aws

ユーザーガイド

Amazon Elastic Compute Cloud

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon Elastic Compute Cloud: ユーザーガイド

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon の商標およびトレードドレスは、Amazon のものではない製品またはサービスと関連付けてはならず、また、お客様に混乱を招くような形や Amazon の信用を傷つけたり失わせたりする形で使用することはできません。Amazon が所有しない商標はすべてそれぞれの所有者に所属します。所有者は必ずしも Amazon と提携していたり、関連しているわけではありません。また、Amazon の後援を受けているとはかぎりません。

Table of Contents

Amazon EC2 とは ... 1
機能 ... 1
関連サービス ... 2
EC2 へのアクセス ... 4
料金 ... 5
見積もり、請求、コストの最適化 ... 6
リソース ... 7
開始方法のチュートリアル ... 8
ステップ 1: インスタンスを起動する ... 10
ステップ 2: インスタンスに接続する ... 12
ステップ 3: インスタンスをクリーンアップする ... 15
次のステップ ... 16
ベストプラクティス ... 18
Amazon マシンイメージ ... 21
AMI の特性 ... 23
起動許可 ... 23
ルートボリュームタイプ ... 24
AMI ルートボリュームタイプを識別する ... 25
仮想化タイプ ... 27
AMI の検索 ... 29
Systems Manager パラメータ ... 32
Systems Manager パブリックパラメータ ... 36
AWS Marketplace の有料 AMI ... 40
AWS Marketplace で AMI を販売する ... 41
有料 AMI の検索 ... 41
有料 AMI の購入 ... 43
製品コードを取得する ... 44
有料サポートの使用 ... 45
有料およびサポートされる AMI の請求書 ... 46
サブスクリプションを管理する ... 46
AMI ライフサイクル ... 48
AMI を作成する ... 49
Amazon S3-backed AMI を作成する ... 58
Windows Sysprep を使用して AMI を作成する ... 99

AMI のコピー ... 116
AMI を保存および復元する ... 134
AMI 祖先 ... 144
AMI ウォーターマーク ... 150
API の使用 ... 156
AMI を非推奨にする ... 181
AMI の無効化 ... 187
AMI の登録解除 ... 193
ブートモード ... 199
UEFI ブートモードの要件 ... 200
AMI ブートモードパラメータ ... 202
インスタンスタイプのブートモード ... 204
インスタンスのブートモード ... 207
オペレーティングシステムのブートモード ... 209
AMI ブートモードを設定する ... 211
UEFI 変数 ... 217
UEFI セキュアブート ... 218
AMI 暗号化 ... 236
インスタンスの起動シナリオ ... 236
イメージコピーのシナリオ ... 240
共有 AMI ... 242
検証済みプロバイダー ... 243
共有 AMI の検索 ... 244
Linux 用の共有 AMI を使用する準備をする ... 247
許可された AMI ... 248
AMI を公開する ... 269
AMI へのパブリックアクセスをブロック ... 273
組織および組織単位での共有 AMI の使用 ... 283
特定の AWS アカウントとの AMI の共有 ... 295
アカウントと AMI の共有をキャンセルする ... 299
共有 Linux AMI の作成に関する推奨事項 ... 300
AMI イベントをモニタリングする ... 306
イベントの詳細 ... 307
available のイベント ... 308
failed のイベント ... 309
deregistered のイベント ... 309

disabled のイベント ... 310
AMI の請求について ... 311
AMI 請求フィールド ... 311
AMI 請求情報の検索 ... 314
請求書に記載されている AMI の請求を確認します ... 316
AMI クォータ ... 316
AMI のクォータの引き上げをリクエストする ... 318
インスタンス ... 319
インスタンスのタイプ ... 320
利用可能なインスタンスタイプ ... 321
ハードウェア仕様 ... 322
ハイパーバイザーのタイプ ... 322
AMI 仮想化タイプ ... 323
Processors ... 323
インスタンスタイプの検索 ... 326
EC2 インスタンスタイプファインダー ... 334
Compute Optimizer 推奨事項 ... 336
インスタンスタイプの変更 ... 339
バーストパフォーマンスインスタンス ... 349
GPU インスタンス ... 403
Mac インスタンス ... 418
EBS 最適化 ... 472
CPU オプション ... 607
AMD SEV-SNP ... 917
プロセッサのステート制御 ... 933
管理されたインスタンス ... 937
管理されたインスタンスの請求 ... 937
管理されたインスタンスを特定する ... 938
マネージドリソースの可視性設定 ... 940
インスタンスの使用を開始する ... 944
ネストされた仮想化 ... 944
仕組み ... 945
考慮事項 ... 945
ネストされた仮想化を有効にした新規インスタンスを起動する ... 946
ネストされた仮想化を使用するように既存のインスタンスを設定する ... 947
請求と購入のオプション ... 948

オンデマンドインスタンス ... 949
リザーブドインスタンス ... 952
スポットインスタンス ... 1022
専用ホスト ... 1126
Dedicated Instances ... 1189
Capacity Reservations ... 1197
起動テンプレート ... 1347
制限事項 ... 1348
アクセス許可 ... 1349
起動インスタンスの制御 ... 1357
作成 ... 1359
修正 (バージョン管理) ... 1376
削除 ... 1380
インスタンスの起動 ... 1383
チュートリアル ... 1386
インスタンスパラメータのリファレンス ... 1410
インスタンス起動ウィザードを使用した起動 ... 1425
起動テンプレートを使用した起動 ... 1428
既存のインスタンスから起動する ... 1436
AWS Marketplace AMI からの起動 ... 1438
インスタンスへの接続 ... 1440
一般的な接続の前提条件 ... 1442
SSH を使用した Linux インスタンスへの接続 ... 1448
RDP を使用した Windows インスタンスへの接続 ... 1465
Session Manager による接続 ... 1475
パブリック IP と EC2 Instance Connect を使用して接続する ... 1476
プライベート IP と EC2 Instance Connect エンドポイントを使用して接続する ... 1514
インスタンスの状態変更 ... 1548
インスタンスの状態別の請求 ... 1548
保留中のインスタンス ... 1550
停止したインスタンス ... 1550
休止したインスタンス ... 1551
インスタンスの再起動 ... 1551
終了したインスタンス ... 1552
インスタンスの状態の違い ... 1552
停止と開始 ... 1555

休止 ... 1575
再起動 ... 1606
終了 ... 1608
リタイア ... 1626
インスタンスの自動復旧 ... 1631
自動インスタンス復旧の主な概念 ... 1632
簡易自動復旧と CloudWatch アクションベースの復旧の違い ... 1634
回復力のあるシステムを構築する ... 1635
自動復旧が発生したかどうかを確認する ... 1636
簡易自動復旧 ... 1637
CloudWatch アクションに基づく復旧 ... 1642
インスタンスメタデータ ... 1647
インスタンスメタデータのカテゴリ ... 1648
動的データのカテゴリ ... 1664
インスタンスメタデータにアクセスする ... 1665
IMDS オプションの設定 ... 1707
起動時のコマンドの実行 ... 1745
例: AMI 作成インデックス値 ... 1773
ホストが EC2 インスタンスであるかどうかを検出する ... 1777
インスタンスアイデンティティドキュメント の検査 ... 1777
システム UUID の検査 ... 1777
システムの仮想マシン生成識別子を調べる ... 1779
インスタンスアイデンティティドキュメント ... 1785
インスタンスアイデンティティドキュメントを取得する ... 1786
インスタンスアイデンティティドキュメントを検証する ... 1788
パブリック証明書 ... 1799
STIG コンプライアンス ... 1862
STIG 設定 ... 1864
STIG スクリプトをダウンロードする ... 1886
Systems Manager で STIG 設定を適用する ... 1908
クロック同期 ... 1911
ハードウェアパケットタイムスタンプ ... 1912
うるう秒 ... 1912
ローカル Amazon Time Sync Service を使用する ... 1914
パブリック Amazon Time Sync Service を使用する ... 1931
Linux インスタンスのタイムスタンプを比較する ... 1933

インスタンスのタイムゾーンを変更する 1935
EC2 Capacity Manager 1937
主な機能 1938
Capacity Manager の有効化 1939
データの整理 1951
コンソールでの Capacity Manager のナビゲーション 1970
データのエクスポート 1972
デバイスドライバーの管理 1977
ネットワークドライバー 1977
グラフィックスドライバー 1978
ストレージデバイスドライバー 1978
その他の Windows デバイスドライバー 1978
AMD ドライバー 1978
NVIDIA ドライバー 1984
Windows に ENA ドライバーをインストールする 2034
Windows PV ドライバー 2055
NVMe ドライバー 2092
その他の Windows デバイスドライバー 2104
Windows インスタンスの設定 2105
Windows 固有のシステム設定 2106
AWS Windows インスタンス用 NVMe ドライバー 2107
Windows 起動エージェント 2108
Windows の EC2 Fast Launch 2268
Windows 管理者のパスワードの変更 2292
Windows システムコンポーネントの追加 2294
Windows に WSL をインストールする 2299
Windows ユーティリティ 2301
Windows インスタンスをアップグレードする 2303
インプレースアップグレードの実行 2304
自動アップグレードの実行 2309
Nitro ベースのインスタンスタイプに移行する 2320
アップグレードのトラブルシューティング 2330
チュートリアル: RDS データベースに EC2 インスタンスを接続する 2331
チュートリアルの目的 2331
Context 2331
アーキテクチャ 2332

考慮事項 2333
チュートリアル完了までの時間 2334
コスト 2334
オプション 1: EC2 コンソールを使用して自動的に接続する 2335
オプション 2: RDS コンソールを使用して自動的に接続する 2347
オプション 3: 手動で接続する 2359
フリート 2371
機能とメリット 2371
フリートにどの方法を使用するか 2372
設定オプション 2374
リクエストタイプ 2375
支出制限 2405
属性ベースのインスタンスタイプの選択 2407
インスタンスの重み付け 2443
配分戦略 2446
キャパシティーの再調整 2453
キャパシティ予約 2459
EC2 フリートの操作 2460
EC2 フリートリクエストの状態 2461
EC2 フリートの前提条件 2462
EC2 フリートの作成 2467
EC2 フリート のタグ付け 2477
EC2 フリートを記述する 2479
EC2 フリート の変更 2484
EC2 フリート の削除 2486
スポットフリートを操作する 2491
スポットフリートリクエストの状態 2492
スポットフリートアクセス許可 2493
スポットフリートを作成する 2505
スポットフリートにタグ付けします。 2515
スポットフリートを記述する 2525
スポットフリートリクエストを変更します。 2527
スポットフリートリクエストをキャンセル (削除) します。 2529
スポットフリートの自動スケーリング 2532
フリートをモニタリングする 2543
CloudWatch を使用してフリートをモニタリングする 2544

EventBridge を使用してフリートをモニタリングする 2547
チュートリアル 2566
チュートリアル: インスタンスの重み付けを使用するように EC2 フリートを設定する 2568
チュートリアル: オンデマンドインスタンスをプライマリキャパシティとして使用するように EC2 フリートを設定する 2572
チュートリアル: ターゲットのキャパシティ予約を使用してオンデマンドインスタンスを起動するように EC2 フリートを設定する 2574
チュートリアル: キャパシティブロックにインスタンスを起動するように EC2 フリートを設定する 2580
チュートリアル: キャパシティ予約に対してインスタンスを起動するように EC2 フリートを設定する 2583
EC2 フリートの CLI 設定の例 2588
例 1: スポットインスタンスをデフォルト購入オプションとして起動する 2589
例 2: オンデマンドインスタンスをデフォルト購入オプションとして起動する 2589
例 3: オンデマンドインスタンスをプライマリ容量として起動する 2590
例 4: 複数のキャパシティー予約を使用して オンデマンドインスタンス を起動する 2590
例 5: 合計ターゲット容量が未使用キャパシティーの予約の数を超えたときに、キャパシティーの予約を使用してオンデマンドインスタンスを起動する 2594
例 6: ターゲットのキャパシティー予約を使用してオンデマンドインスタンスを起動する 2598
例 7: 容量の再調整を設定して代替スポットインスタンスを起動する 2601
例 8: 容量最適化フリートでスポットインスタンスを起動する 2603
例 9: 優先順位のある容量最適化フリートでスポットインスタンスを起動する 2604
例 10: price-capaity-optimized フリートでスポットインスタンスを起動する 2606
例 11: 属性ベースのインスタンスタイプの選択を設定する 2607
スポットフリートの CLI 設定の例 2608
例 1: 最低価格のアベイラビリティーゾーン 2609
例 2: リストからの最低価格のアベイラビリティーゾーン 2610
例 3: リストからの最低価格のインスタンスタイプ 2612
例 4。リクエストの料金を上書きする 2614
例 5: 分散配分戦略を使用する 2615
例 6: インスタンスの重み付けを使用する 2618
例 7: オンデマンドキャパシティを使用する 2620
例 8: キャパシティの再調整を設定する 2621
例 9: キャパシティ最適化フリートを使用する 2623
例 10: 優先順位のあるキャパシティ最適化フリートを使用する 2624
例 11: priceCapacityOptimized フリートを使用する 2625

例 12: 属性ベースのインスタンスタイプの選択を設定する 2626
フリートのクォータ 2627
ターゲットキャパシティのクォータ引き上げをリクエストします 2629
ネットワーク 2630
リージョンとゾーン 2631
リージョン 2632
アベイラビリティーゾーン 2632
Local Zones 2634
Wavelength Zone 2635
AWS Outposts 2637
インスタンスの IP アドレス指定 2639
プライベート IPv4 アドレス 2639
パブリック IPv4 アドレス 2641
パブリック IPv4 アドレスの最適化 2642
IPv6 アドレス 2644
複数の IP アドレス 2645
EC2 インスタンスのホスト名 2647
リンクローカルアドレス 2647
IPv4 アドレス 2648
IPv6 アドレス 2653
セカンダリ IP アドレス 2659
Windows の IPv4 アドレス 2664
EC2 インスタンスのホスト名とドメイン 2671
EC2 インスタンスのホスト名とドメインについて 2672
ホスト名タイプ 2672
自分の IP アドレスを使用する 2688
BYOIP の定義 2689
要件とクォータ 2690
リージョン別の可用性 2691
Local Zone の可用性 2691
前提条件 2694
アドレス範囲をオンボードする 2702
アドレス範囲を使用する 2711
Elastic IP アドレス 2713
Elastic IP アドレスの料金 2713
Elastic IP アドレスの基本 2713

Elastic IP アドレスのクォータ 2715
Elastic IP アドレスの関連付け 2715
Elastic IP アドレスの移管 2721
Elastic IP アドレスをリリース 2726
E メールアプリケーション用の逆引き DNS の使用 2728
ネットワークインターフェイス 2732
ネットワークインターフェイスの概念 2733
ネットワークカード 2736
ネットワークインターフェイスあたりの IP アドレス数 2740
ネットワークインターフェイスの作成 2742
ネットワークインターフェイスのアタッチ 2748
IP アドレスの管理 2753
ネットワークインターフェイス属性の変更 2756
複数のネットワークインターフェイス 2761
リクエスタマネージド型のネットワークインターフェイス 2764
プレフィックスの委任 2766
ネットワークインターフェイスの削除 2778
ネットワーク帯域幅 2778
使用可能なインスタンスの帯域幅 2780
帯域幅の重み付け 2782
インスタンスの帯域幅をモニタリングします。 2821
拡張ネットワーク 2821
Elastic Network Adapter (ENA) 2822
ENA Express 2908
Intel 82599 VF 2948
ネットワークパフォーマンスのモニタリング 2962
Linux のネットワークレイテンシーを改善する 2973
Nitro のパフォーマンスに関する考慮事項 2977
Windows でのネットワークパフォーマンスを最適化する 2985
Elastic Fabric Adapter 2987
EFA の基本 2988
サポートされているインターフェイスとライブラリ 2991
サポートされるインスタンスタイプ 2991
サポートされるオペレーティングシステム 3008
EFA の制限事項 3009
EFA 価格設定 3010

EFAと MPI の開始方法 ... 3010
EFAとNCCL の開始方法 ... 3030
EFA と NIXL の開始方法 ... 3054
ネットワーク帯域幅の最大化 ... 3076
EFA を作成してアタッチする ... 3092
EFA をデタッチおよび削除する ... 3098
EFA のモニタリング ... 3098
EFA インストーラを検証する ... 3106
リリースノート ... 3121
EC2 トポロジー ... 3147
仕組み ... 3148
前提条件 ... 3154
例 ... 3157
プレイスメントグループ ... 3172
プレイスメント戦略 ... 3174
プレイスメントグループの作成 ... 3181
インスタンスの配置を変更する ... 3184
プレイスメントグループの削除 ... 3186
共有プレイスメントグループ ... 3187
AWS Outposts のプレイスメントグループ ... 3189
ネットワーク MTU ... 3191
ジャンボフレーム (9001 MTU) ... 3191
パス MTU 検出 ... 3193
インスタンスの MTU を設定する ... 3194
トラブルシューティング ... 3200
仮想プライベートクラウド ... 3200
デフォルトの VPC ... 3201
非デフォルト VPC ... 3202
インターネットアクセス ... 3202
共有サブネット ... 3203
IPv6 専用サブネット ... 3203
セカンダリネットワーク ... 3203
セカンダリネットワークとは ... 3204
主要なコンセプト ... 3205
アーキテクチャ ... 3205
その他の考慮事項 ... 3205

開始方法 ... 3206
セカンダリネットワークリソースの管理 ... 3208
ネットワーク設計のベストプラクティス ... 3209
トラブルシューティング ... 3210
クォータと制限 ... 3211
セキュリティ ... 3212
データ保護 ... 3213
Amazon EBS のデータセキュリティ ... 3214
保管中の暗号化 ... 3214
転送中の暗号化 ... 3216
インフラストラクチャセキュリティ ... 3218
ネットワークの隔離 ... 3218
物理ホストでの分離 ... 3219
ネットワークトラフィックの制御 ... 3219
耐障害性 ... 3222
コンプライアンス検証 ... 3223
Identity and access management ... 3223
アイデンティティベースのポリシー ... 3224
API のポリシーの例 ... 3236
コンソールのサンプルポリシー ... 3281
AWS 管理ポリシー ... 3294
IAM; ロール ... 3303
更新管理 ... 3316
Windows インスタンスにおけるベストプラクティス ... 3316
高レベルのセキュリティのベストプラクティス ... 3317
更新管理 ... 3318
設定管理 ... 3320
変更管理 ... 3321
Amazon EC2 Windows インスタンスでの監査とアカウンタビリティ ... 3322
キーペア ... 3322
キーペアを作成する ... 3324
キーペアの詳細表示 ... 3331
キーペアの削除 ... 3336
Linux インスタンスでパブリックキーを追加または置き換える ... 3337
フィンガープリントを確認します ... 3339
セキュリティグループ ... 3342

概要 ... 3343
セキュリティグループを作成する ... 3344
インスタンスのセキュリティグループの変更 ... 3346
セキュリティグループを削除する ... 3353
接続追跡 ... 3354
さまざまなユースケースのセキュリティグループのルール ... 3361
NitroTPM ... 3368
要件 ... 3369
NitroTPM 用の Linux AMI の有効化 ... 3371
AMI が NitroTPM に対して有効になっていることを確認する ... 3372
NitroTPM の使用を有効化または停止する ... 3374
インスタンスが NitroTPM に対して有効になっていることを検証する ... 3375
公開承認キーを取得する ... 3377
EC2 インスタンスアテステーション ... 3378
AWS KMS との統合 ... 3380
分離されたコンピューティング環境のアテステーション ... 3381
AWS 責任共有モデル ... 3381
構成証明可能 AMI ... 3381
アテステーション用に AWS KMS を準備する ... 3391
NitroTPM アテステーションドキュメントを取得する ... 3392
AWS KMS との統合 ... 3400
自社のオペレーターからデータを分離する ... 3401
Windows インスタンスの クレデンシャルガード ... 3402
前提条件 ... 3402
サポートされているインスタンスを起動する ... 3403
メモリインテグリティの無効化 ... 3404
クレデンシャルガード を有効にする ... 3405
クレデンシャルガード が実行されていることを確認する ... 3407
AWS PrivateLink ... 3408
インターフェイス VPC エンドポイントを作成する ... 3409
エンドポイントポリシーを作成する ... 3409
ストレージ ... 3411
AWS ストレージ料金表 ... 3412
Amazon EBS ... 3412
EBS ボリューム制限 ... 3413
EBS カード ... 3417

Amazon EC2 インスタンスストア ... 3420
データの永続性 ... 3421
インスタンスストアボリュームの制限 ... 3423
SSD インスタンスストアボリューム ... 3428
インスタンスストアボリュームを追加する ... 3432
M1 および C1 インスタンスのスワップボリュームの有効化 ... 3440
インスタンスストアボリュームの初期化 ... 3444
詳細なパフォーマンス統計インスタンスストアボリューム ... 3445
ルートボリューム ... 3449
Amazon EBS ルートボリュームを持つインスタンス ... 3450
インスタンスストアのルートボリュームを持つインスタンス (Linux インスタンスのみ) ... 3451
インスタンス終了後もルートボリュームを保持する ... 3452
ルートボリュームを置き換える ... 3457
ボリュームのデバイス名 ... 3471
使用できるデバイス名 ... 3471
デバイス名に関する考慮事項 ... 3474
ブロックデバイスマッピング ... 3476
ブロックデバイスマッピングの概念 ... 3476
AMI へのブロックデバイスマッピングの追加 ... 3480
インスタンスへのブロックデバイスマッピングの追加 ... 3485
Windows インスタンスへのボリュームのアタッチとマッピング方法 ... 3494
NVMe ディスクをボリュームにマッピングする ... 3494
非 NVMe ディスクをボリュームにマッピングする ... 3500
Torn Write Prevention ... 3510
サポートされているブロックサイズ ... 3511
要件 ... 3512
インスタンスのサポートを確認する ... 3513
ワークロードを設定する ... 3514
Windows VSS EBS スナップショット ... 3516
VSS とは ... 3517
VSS ベースの Amazon EBS スナップショットソリューションの仕組み ... 3518
VSS の前提条件 ... 3519
VSS スナップショットを作成する ... 3533
VSS スナップショットのトラブルシューティング ... 3544
AWS VSS ソリューションの復元オプション ... 3549
バージョン履歴 ... 3550

オブジェクトストレージ、ファイルストレージ、およびファイルキャッシュ 3555
Amazon S3 3556
Amazon EC2 EFS 3561
Amazon FSx 3565
Amazon File Cache 3571
リソースの管理 3572
リソースのリージョンを選択する 3572
リソースの検索 3574
コンソールの手順 3574
コマンドラインの例 3583
グローバルビュー (クロスリージョン) 3587
AWS グローバルビュー 3587
リソースのタグ付け 3591
タグの基本 3592
リソースのタグ付け 3593
タグの制限 3594
タグとアクセス管理 3595
請求用のリソースのタグ付け 3595
リソースにタグ付けするアクセス許可 3596
タグの追加と削除 3599
タグによるリソースのフィルタリング 3603
インスタンスメタデータを使用したタグの表示 3605
Service Quotas 3611
現在の制限を表示するには 3612
引き上げのリクエスト 3612
ポート 25 を使用した E メール送信の制限 3613
リソースのモニタリング 3614
インスタンスのステータスのモニタリング 3615
ステータスチェック 3616
状態変更イベント 3625
予定されているイベント 3627
CloudWatch を使用したインスタンスのモニタリング 3667
インスタンスアラーム 3667
詳細モニタリングを管理する 3669
CloudWatch メトリクス 3672
CloudWatch エージェントをインストールして設定する 3692

メトリクスの統計 ... 3696
モニタリンググラフを表示する ... 3706
アラームの作成 ... 3707
インスタンスを停止、終了、再起動、または復旧するアラームを作成する ... 3708
EventBridge を使用して自動化する ... 3721
Amazon EC2 イベントタイプ ... 3721
CloudTrail を使用して API 呼び出しをログに記録する ... 3722
CloudTrail の Amazon EC2 API 管理イベント ... 3724
Amazon EC2 API イベントの例 ... 3724
EC2 Instance Connect による接続を監査する ... 3725
.NET および SQL Server アプリケーションのモニタリング ... 3727
無料利用枠の使用状況の追跡 ... 3728
2025 年 7 月 15 日より前とそれ以降の無料利用枠の特典 ... 3728
2025 年 7 月 15 日より前に作成されたアカウントの無料利用枠の使用状況を追跡する ... 3730
請求および使用状況レポート ... 3732
インスタンス ... 3733
ベアメタルインスタンス ... 3733
専用ホスト ... 3734
Dedicated Instances ... 3734
EBS 最適化 ... 3735
キャパシティ予約 ... 3735
EC2 インスタンスのインベントリ ... 3735
地理的場所 ... 3737
ハードウェア仕様 ... 3738
イメージ ... 3739
IP アドレスと DNS ホスト名 ... 3741
「セキュリティグループのルール」 ... 3742
ユーザーデータ ... 3744
関連リソース ... 3745
トラブルシューティング ... 3746
インスタンスの起動に関する問題 ... 3747
無効なデバイス名 ... 3747
インスタンス制限の超過 ... 3748
インスタンス容量の不足 ... 3749
リクエストされた設定は現在サポートされていません。サポートされている設定については、ドキュメントを参照してください。 ... 3749

インスタンスがすぐに終了する ... 3750
アクセス権限の不足 ... 3751
Windows の起動直後に CPU 使用率が高い (Windows インスタンスのみ) ... 3752
IMDSv1 が有効化されたインスタンスの起動が失敗する ... 3753
インスタンスの停止の問題 ... 3754
インスタンスを強制停止する ... 3755
(オプション) 代替インスタンスの作成 ... 3757
インスタンスの終了に関する問題 ... 3760
インスタンスがすぐに終了する ... 3760
インスタンスの削除の遅延 ... 3760
表示されているインスタンスを削除する ... 3763
エラー: インスタンスは終了できない可能性があります。その「disableApiTermination」インスタンス属性を変更します ... 3763
インスタンスが自動的に起動または終了される ... 3763
到達できないインスタンス ... 3764
インスタンスの再起動 ... 3764
インスタンスコンソール出力 ... 3764
接続できないインスタンスのスクリーンショットの取得 ... 3766
Windows インスタンスの一般的なスクリーンショット ... 3767
ホストコンピュータに障害が発生した場合のインスタンスの復旧 ... 3777
インスタンスがオフラインになり、予期せず再起動された ... 3777
Linux インスタンスの SSH の問題 ... 3778
接続の問題の一般的な原因 ... 3779
インスタンスへの接続エラー: 接続タイムアウト ... 3781
エラー: キーを読み込めません..。期待: 任意のプライベートキー ... 3784
エラー: ユーザーキーがサーバーによって認識されない ... 3785
エラー: アクセス許可が拒否されたか、[インスタンス] ポート 22 によって接続が閉じられました。 ... 3787
エラー: Unprotected Private キー ファイル (保護されていないプライベートキーファイル) ... 3790
エラー: プライベートキーの先頭は「-----BEGIN RSA PRIVATE KEY-----」、末尾は「-----END RSA PRIVATE KEY-----」にする必要があります ... 3791
エラー: ホストキーの検証に失敗しました ... 3792
エラー: サーバー refused our key または No supported authentication methods available (サーバーはキーを拒否しましたまたは利用可能なサポートされる認証方法はありません) 3792
インスタンスに対して ping を実行できない ... 3793

エラー: サーバーによる予期しないネットワーク接続の閉鎖 3794
エラー: EC2 インスタントコネクト のホストキーの検証に失敗しました 3794
EC2 インスタントコネクト を使用して Unbntu インスタンスに接続できない 3796
プライベートキーを紛失しました。自分のインスタンスに接続するにはどうすればよいですか? 3797
Linux インスタンスのステータスチェックの失敗 3804
ステータスチェック情報の確認 3805
システムログの取得 3806
Linux インスタンスに対するシステムログエラーのトラブルシューティング 3807
メモリ不足: プロセスの終了 3808
エラー: mmu_update failed (メモリ管理の更新に失敗しました) 3809
I/O エラー (ブロックデバイス障害) 3810
I/O エラー: ローカルでもリモートディスクでもありません (破損した分散ブロックデバイス) 3812
request_module: runaway loop modprobe (古い Linux バージョンでレガシーカーネル modprobe がループしている) 3813
「FATAL: kernel too old」および「fsck: No such file or directory while trying to open /dev」(カーネルと AMI の不一致) 3814
「FATAL: Could not load /lib/modules」または「BusyBox」(カーネルモジュールの欠如) 3815
エラー: 無効のカーネル (EC2 と互換性のないカーネル) 3817
fsck: No such file or directory while trying to open..。(ファイルシステムが見つからない。) 3818
General error mounting filesystems (マウント失敗) 3820
VFS: Unable to mount root fs on unknown-block (ルートファイルシステム不一致) 3823
Error: Unable to determine major/minor number of root device..。(ルートファイルシステム/デバイス不一致) 3824
XENBUS: Device with no driver..。 3826
... days without being checked, check forced (ファイルシステムのチェックが必要です) ... 3827
fsck died with exit status..。(デバイスが見つからない) 3828
GRUB プロンプト (grubdom>) 3829
Bringing up interface eth0: Device eth0 has different MAC address than expected, ignoring。(ハードコードされた MAC アドレス) 3832
SELinux ポリシーを読み込めません。Machine is in enforcing mode。Halting now。(SELinux の誤設定) 3834
XENBUS: Timeout connecting to devices (Xenbus タイムアウト) 3835
Linux インスタンスが間違ったボリュームから起動する 3836

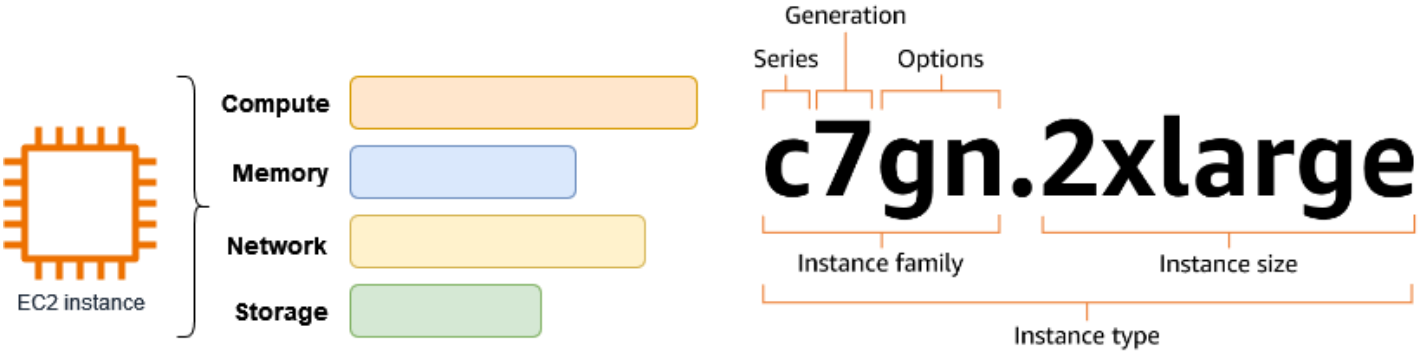
Windows インスタンスの RDP 問題 3838
リモートデスクトップからリモートコンピュータに接続できません 3839
macOS RDP クライアントの使用中にエラーが発生する 3840
RDP にデスクトップではなく黒い画面が表示される 3841
管理者ではないユーザーでインスタンスにリモートでログオンできない 3841
AWS Systems Manager を使用したリモートデスクトップ問題のトラブルシューティング 3842
リモートレジストリを使用して EC2 インスタンスでリモートデスクトップを有効にする 3846
プライベートキーを紛失しました。Windows インスタンスに接続するにはどうすればよいですか? 3847
Windows インスタンスの開始に関する問題 3848
"パスワードは使用できません" 3848
"パスワードはまだ使用できません" 3849
"Windows パスワードを取得できません" 3850
"メタデータサービスを待っています" 3850
"Windows のライセンス認証ができません" 3855
"Windows が正規品ではありません (0x80070005) " 3857
"ライセンスを発行できるターミナルサーバーライセンスサーバーがありません" 3857
「一部の設定は当組織によって管理されています」 3858
Windows インスタンスの問題 3858
AWS Systems Manager Sessions Manager を Windows Server 2025 インスタンスに接続できません 3859
EBS ボリュームが Windows Server 2016 および 2019 で初期化されない 3859
ディレクトリサービス復元モード (DSRM) で EC2 Windows インスタンスを起動する 3861
インスタンスのネットワーク接続が失われる、または、スケジュールされたタスクが予定通りに実行されない 3864
コンソールの出力を取得できない 3864
Windows Server 2012 R2 をネットワークで使用できない 3865
ディスク署名の衝突 3865
ネットワーク経由の Windows インスタンスカーネルデバッグ 3867
前提条件 3867
ステップ 1: デバッグホストに Windows デバッギングツールをインストールする 3868
ステップ 2: デバッグターゲットをセットアップする 3869
ステップ 3: デバッグホストでデバッギングセッションを開始する 3871
ステップ 4: デバッグターゲットを再起動する 3872
デバッギング後にデバッグ設定をクリーンアップする 3872

制限事項 ... 3873
追加のメモ ... 3873
Windows 管理者パスワードのリセット ... 3874
EC2Launch v2 を使用したパスワードのリセット ... 3875
EC2Launch を使用したパスワードのリセット ... 3881
EC2Config を使用したパスワードのリセット ... 3886
Sysprep の問題のトラブルシューティング ... 3892
Linux インスタンス用 EC2Rescue ... 3894
EC2Rescue をインストールする ... 3895
EC2Rescue コマンドを実行する ... 3899
EC2Rescue モジュールの開発 ... 3901
EC2Rescue for Windows インスタンス ... 3908
EC2Rescue GUI を使用したトラブルシューティング ... 3909
EC2Rescue CLI を使用したトラブルシューティング ... 3916
EC2Rescue と Systems Manager を使用したトラブルシューティング ... 3924
EC2 シリアルコンソール ... 3928
前提条件 ... 3929
EC2 シリアルコンソールへのアクセスを設定する ... 3937
EC2 シリアルコンソールに接続する ... 3948
EC2 シリアルコンソールからの切断 ... 3959
EC2 シリアルコンソールを使用してインスタンスをトラブルシューティングする ... 3959
診断割り込みを送信する ... 3968
サポートされるインスタンスタイプ ... 3969
前提条件 ... 3969
診断割り込みの送信 ... 3972
ドキュメント履歴 ... 3973
2018 年以前の履歴 ... 4016

Amazon EC2 とは

Amazon Elastic Compute Cloud (Amazon EC2) は Amazon Web Services (AWS) クラウドでオンデマンドのスケーラブルなコンピューティングキャパシティを提供します。Amazon EC2 を使用することで、ハードウェアのコストを削減できます。これによりアプリケーションの開発とデプロイを迅速に行うことができます。Amazon EC2 を使用すると、必要な数 (またはそれ以下) の仮想サーバーの起動、セキュリティおよびネットワーキングの構成、ストレージの管理ができます。月次または年次の処理やウェブサイトのトラフィックの急増など、計算量の多いタスクを処理するためのキャパシティーを追加 (スケールアップ) できます。使用量が減った場合はキャパシティーを再び減らす (スケールダウン) こともできます。

EC2インスタンスは AWS クラウド上の仮想サーバーです。EC2 インスタンスを起動するときは指定したインスタンスタイプによって、インスタンスで使用できるハードウェアが決定します。各インスタンスタイプはコンピューティング、メモリ、ネットワーク、ストレージリソースが異なるバランスで構成されています。詳細については「Amazon EC2 インスタンスタイプガイド」を参照してください。



Amazon EC2 の機能

Amazon EC2 には次の高度な機能があります。

インスタンス

仮想サーバー。

Amazon マシンイメージ (AMI)

サーバーに必要なコンポーネントをパッケージ化した、インスタンス用に事前に設定されているテンプレート(オペレーティングシステムおよび追加のソフトウェアを含む)。

インスタンスのタイプ

インスタンス用の CPU、メモリ、ストレージ、ネットワーキングキャパシティーのさまざまな設定。

Amazon EBS ボリューム

Amazon エラスティックブロックストア (Amazon EBS) を使用したデータ用の永続的ストレージボリューム。

インスタンスストアボリューム

インスタンスを停止、休止、または終了するときに削除される一時データ用のストレージボリューム。

キーペア

インスタンス用の安全なログイン情報。AWS はパブリックキー、ユーザーはプライベートキーを安全な場所に保存します。

セキュリティグループ

インスタンスに到達できるプロトコル、ポート、送信元 IP の範囲、およびインスタンスが接続できる宛先 IP の範囲を指定できる仮想ファイアウォール。

Amazon EC2 はマーチャントまたはサービスプロバイダーによるクレジットカードデータの処理、ストレージ、および伝送をサポートしており、ペイメントカード業界 (PCI) データセキュリティ基準 (DSS) に準拠していることが確認されています。PCI DSS の詳細 (AWS PCI コンプライアンスパッケージ のコピーをリクエストする方法など) については「PCI DSS レベル 1」を参照してください。

関連サービス

Amazon EC2 で使用できるサービス

Amazon EC2 を使用してデプロイするインスタンスでは 他の AWS のサービス を使用できます。

Amazon EC2 Auto Scaling

アプリケーションの負荷を処理するために適切な数の Amazon EC2 インスタンスがあることを確認できます。

AWS Backup

Amazon EC2 インスタンスとそれらにアタッチされている アマゾン EBS ボリュームのバックアップを自動化できます。

Amazon CloudWatch

インスタンスと Amazon EBS ボリュームをモニタリングできます。

Elastic Load Balancing

アプリケーションの着信トラフィックを複数の インスタンスに自動的に分散できます。

Amazon GuardDuty

不正な、または悪意のある可能性がある EC2 インスタンスの使用を検出します。

EC2 イメージビルダー

カスタマイズされたセキュアで最新のサーバーイメージの作成、管理、デプロイを自動化します。

AWS Launch Wizard

個々の AWS リソースを手動で識別およびプロビジョニングすることなく、サードパーティアプリケーション用の AWS リソースのサイズ設定、設定、デプロイを行えます。

AWS Systems Manager

この安全なエンドツーエンドの管理ソリューションにより、EC2 インスタンスで大規模な操作を実行できます。

その他のコンピューティングサービス

Amazon EC2 を使用する代わりに、別の AWS コンピューティングサービスを使用してインスタンスを起動できます。

Amazon Lightsail

ウェブサイトやウェブアプリケーションの構築にはプロジェクトを迅速にデプロイするのに必要なリソースを低価格で予測可能な月額料金で提供するクラウドプラットフォーム Amazon Lightsail を使用します。Amazon EC2 と Lightsail を比較するにはAmazon Lightsail「 または Amazon EC2」を参照してください。

Amazon Elastic Container Service (Amazon ECS)

コンテナ化されたアプリケーションを EC2 インスタンスのクラスターにデプロイ、管理、スケーリングできます。詳細についてはAWS「 コンテナサービスの選択」を参照してください。

Amazon Elastic Kubernetes Service (Amazon EKS)

AWS で Kubernetes アプリケーションを実行します。詳細についてはAWS「 コンテナサービスの選択」を参照してください。

Amazon EC2 へのアクセス

次のインターフェイスを使用して、Amazon EC2 インスタンスを作成および管理できます。

Amazon EC2 コンソール

Amazon EC2 インスタンスおよびリソースを作成、管理するためのシンプルなウェブインターフェイス。AWS アカウントにサインアップ済みの場合はAWS マネジメントコンソール にサインインし、コンソールのホームページから [EC2] を選択することで、Amazon EC2 コンソールにアクセスできます。

AWS Command Line Interface

コマンドラインシェルでコマンドを使用して AWS サービスとやり取りを行えます。Windows、Mac、Linux でサポートされています。AWS CLI の詳細についてはAWS Command Line Interface「 ユーザーガイド」を参照してください。「AWS CLI コマンドリファレンス」で Amazon EC2 のコマンドを確認できます。

CloudFormation

Amazon EC2 はCloudFormation を使用したリソースの作成をサポートしています。AWS リソースを説明するテンプレートを JSON または YAML 形式で作成すると、CloudFormation はそれらのリソースをプロビジョニングして設定します。CloudFormation テンプレートを再利用して、同じリージョンとアカウント内でも、複数のリージョンとアカウント内でも、同じリソースを複数回プロビジョニングできます。サポートされている Amazon EC2 のリソースタイプとプロパティの詳細についてはAWS CloudFormation「ユーザーガイド」の「EC2 リソースタイプのリファレンス」を参照してください。

AWS SDK

HTTP または HTTPS を介してリクエストを送信する代わりに、言語固有の API を使用してアプリケーションを構築することを希望する場合に備えて、AWS にはソフトウェアデベロッパー向

けのライブラリ、サンプルコード、チュートリアル、その他のリソースが用意されています。これらのライブラリにはリクエストの暗号化署名、リクエストの再試行、エラーレスポンスの処理などのタスクを自動化する基本機能が用意されているので、開発を簡単に始められます。詳細については「AWS で構築するツール」を参照してください 。

AWS Tools for PowerShell

SDK for .NET から公開されている機能に基づいて構築された PowerShell モジュールのセットです。PowerShell用ツール ではPowerShell のコマンドラインから AWS リソースのオペレーションのスクリプトを作成できます。使用を開始する方法については『AWS Tools for PowerShellユーザーガイド』を参照してください。「AWS Tools for PowerShell コマンドレットリファレンス」で、Amazon EC2 のコマンドレットを確認できます。

Query API

Amazon EC2 はクエリ API を提供します。このリクエストはHTTP 動詞 (GET または POST) とクエリパラメータ `Action` で記述する HTTP または HTTPS リクエストです。Amazon EC2 の API アクションの詳細についてはAmazon EC2 API リファレンスの「アクション」を参照してください。

Amazon EC2 の料金表

Amazon EC2 では次の料金オプションが提供されています。

無料利用枠

Amazon EC2 は無料で始めることができます。無料利用枠のオプションについてはAWS 無料利用枠を参照してください。

オンデマンドインスタンス

インスタンスの使用に対し秒単位 (最低時間は 60 秒) で課金され、長期契約や前払い金は不要です。

積立プラン

1〜3 年の期間、1 時間 につき USD で、定期的な使用量を守ることにより Amazon EC2 コストを削減できます。

予約インスタンス

1〜3 年の期間、インスタンスタイプとリージョンを含む特定のインスタンス設定を守ることにより Amazon EC2 コストを削減できます。

スポットインスタンス

未使用の EC2 インスタンスをリクエストして、Amazon EC2 コストを大幅に削減できます。

専用ホスト

オンデマンドで、または Savings Plan の一部として、専用の物理 EC2 サーバーを使用することでコストを削減できます。既存のサーバーバウンドソフトウェアライセンスを使用して、コンプライアンス要件を満たすための支援を受けることができます。

オンデマンドキャパシティ予約

任意の期間、特定のアベイラビリティーゾーンの EC2 インスタンス用にキャパシティーを予約します。

1 秒単位の請求

未使用の分および秒単位のコストを請求から排除します。

Amazon EC2 の課金および料金の詳細なリストと購入モデルの詳細については「Amazon EC2 の料金表」を参照してください。

見積もり、請求、コストの最適化

AWS ユースケースの見積もりを作成するにはAWS 料金見積りツール を使用します。

AWS でデプロイされたクラウドネイティブサービスおよびオープンソースを使用する最新のアーキテクチャに Microsoft ワークロードを変換するコストを見積もるにはMicrosoft ワークロード用の AWS モダナイゼーション計算ツールを使用します。

請求を表示するにはAWS Billing and Cost Management コンソールで請求およびコスト管理ダッシュボードに移動します。請求書には料金の明細が記載された使用状況レポートへのリンクが記載されています。AWS アカウントの請求の詳細についてはAWS 請求とコスト管理 ユーザーガイドを参照してください。

AWS の請求、アカウント、イベントについてご質問がある場合はAWS サポートにお問い合わせください。

プロビジョニングされたサンプル環境の費用を計算するには「クラウドエコノミクスセンター」を参照してください。プロビジョニングされた環境のコストを計算するときはEBS ボリュームのスナップショットストレージなどの付随的コストを必ず含めてください。

AWS Trusted Advisor を使用して、AWS 環境のコスト、セキュリティ、およびパフォーマンスを最適化できます。

AWS Cost Explorer を使用して、EC2 インスタンスのコストと使用状況を分析できます。過去 13 か月までのデータを表示し、次の 12 か月間にどのくらい使用しそうかを予測するこができます。詳細についてはAWS Cost Management「ユーザーガイド」のAWS Cost Explorer「を用いてコストを分析する」を参照してください。

リソース

- Amazon EC2 の機能
- AWS re:Post
- AWS スキルビルダー
- AWS サポート
- 実践的なチュートリアル
- ウェブホスティング
- Windows on AWS

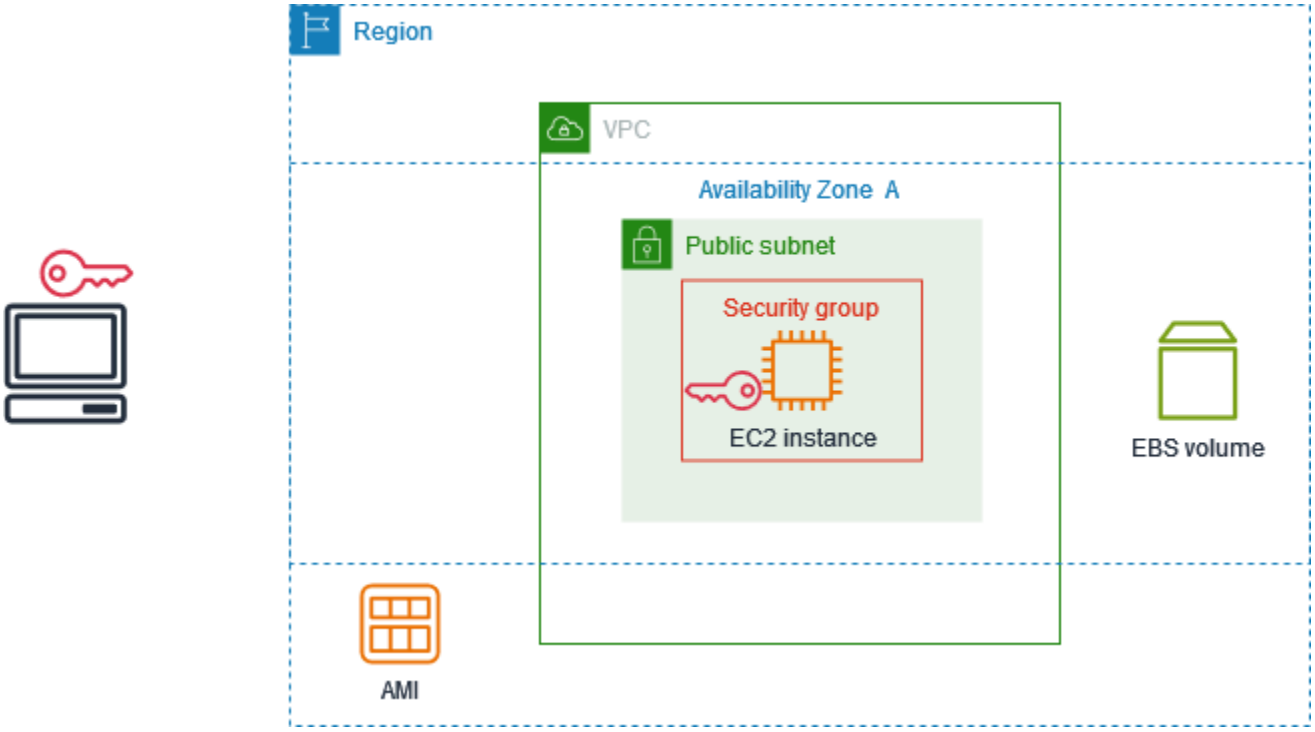
Amazon EC2 の使用を開始する

このチュートリアルを使用して、Amazon Elastic Compute Cloud (Amazon EC2) の使用を開始できます。EC2 インスタンスを起動および接続する方法について説明します。インスタンスは、AWS クラウド内の仮想サーバーです。Amazon EC2 を使用して、インスタンスで実行されるオペレーティングシステムとアプリケーションをセットアップし、設定することができます。

概要

次の図は、このチュートリアルで使用する主要コンポーネントを示しています。

- イメージ – オペレーティングシステムなど、インスタンスで実行されるソフトウェアを含むテンプレート。
- キーペア – インスタンスへの接続時にユーザーの ID を証明するのに使用する一連のセキュリティ認証情報。パブリックキーはインスタンス上、プライベートキーはコンピュータ上にあります。
- ネットワーク – 仮想プライベートクラウド (VPC) は、AWS アカウント専用の仮想ネットワークです。すぐに開始できるように、アカウントには AWS リージョンごとにデフォルトの VPC が付属しており、各デフォルト VPC にはアベイラビリティーゾーンごとにデフォルトのサブネットがあります。
- セキュリティグループ – 仮想ファイアウォールとして機能し、インバウンドトラフィックとアウトバウンドトラフィックを制御します。
- EBS ボリューム – イメージにはルートボリュームが必要です。任意でデータボリュームを追加できます。



このチュートリアルのコスト

AWS アカウントを作成すると、AWS 無料利用枠を利用して、Amazon EC2 を無料で使い始めることができます。

AWS アカウントを作成したのが 2025 年 7 月 15 日より前で、12 か月が経過しておらず、Amazon EC2 の無料利用枠の特典をまだ使い切っていない場合、無料利用枠の特典に収まるオプションを選択できるように支援いたしますので、このチュートリアルを完了するのに一切料金はかかりません。それ以外の場合、インスタンスを起動したとき (アイドル状態のままであっても) から終了するまで、標準の Amazon EC2 使用料金が発生します。

2025 年 7 月 15 日以降に AWS アカウントを作成し、6 か月が経過しておらず、すべてのクレジットを使い切っていない場合、無料利用枠の特典に収まるオプションを選択できるように支援いたしますので、このチュートリアルを完了するのに一切料金はかかりません。

無料利用枠の対象となるかどうかを判断する手順については、「the section called “無料利用枠の使用状況の追跡”」を参照してください。

タスク

- ステップ 1: インスタンスを起動する
- ステップ 2: インスタンスに接続する
- ステップ 3: インスタンスをクリーンアップする

- 次のステップ

ステップ 1: インスタンスを起動する

以下の手順で説明しているように、AWS マネジメントコンソールを使用して EC2 インスタンスを起動できます。このチュートリアルは、最初のインスタンスをすばやく起動できるようにすることを目的としています。そのため、使用できるすべてのオプションを扱っているわけではありません。

インスタンスを起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 画面の上部のナビゲーションバーに、現在の AWS リージョンが表示されます ([オハイオ州] など)。選択したリージョンを使用するか、任意で自分により近いリージョンを選択できます。
3. EC2 コンソールダッシュボードの [インスタンスを起動] ペインで、[インスタンスを起動] を選択してください。
4. [Names and tags] (名前とタグ) の [Name] (名前) には、インスタンス用にわかりやすい名前を入力してください。
5. [Application and OS Images (Amazon Machine Image]) (アプリケーションと OS イメージ (Amazon マシンイメージ) で、次の作業を行います。

   a. [クイックスタート] を選択し、インスタンスのオペレーティングシステム (OS) を選択してください。最初の Linux インスタンスでは、Amazon Linux を選択することをお勧めします。

   b. [Amazon マシンイメージ (AMI)] から、[無料利用枠の対象] となっている AMI を選択してください。
6. [インスタンスタイプ] の [インスタンスタイプ] で、[無料利用枠の対象] と表示されているインスタンスタイプを選択します。
7. [キーペア (ログイン)] の下で、[キーペア名] において既存のキーペアを選択するか、[新しいキーペアの作成] を選択して最初のキーペアを作成します。

> **Warning**
>
> [キーペアなしで続行 (推奨されません)] を選択した場合、このチュートリアルで説明されている方法を使用してインスタンスに接続することはできません。

8. [ネットワーク設定] を見ると、デフォルトの VPC が選択され、ユーザー向けに選択されたアベイラビリティーゾーンでデフォルトのサブネットを使用するオプションが選択され、任意の場所からのインスタンスへの接続を許可するルールを備えたセキュリティグループが設定されていることがわかります (`0.0.0.0/0`)。

   > **Warning**
   >
   > `0.0.0.0/0` を指定すると、世界中の任意の IP アドレスからのトラフィックが有効になります。SSH と RDP プロトコルの場合、これはテスト環境で短時間なら許容できますが、実稼働環境で行うのは安全ではありません。実稼働環境では、適切な個別の IP アドレスまたはアドレス範囲からのアクセスのみを許可するようにしてください。

   最初のインスタンスでは、デフォルト設定を使用することをお勧めします。それ以外の場合は、次のようにネットワーク設定を更新できます。

   - (任意) 特定のデフォルトサブネットを使用するには、[編集] を選択し、サブネットを選択してください。
   - (任意) 別の VPC を使用するには、[編集] を選択し、既存の VPC を選択してください。VPC がパブリックインターネットアクセス用に設定されていない場合、インスタンスに接続できるようになりません。
   - (任意) 特定のネットワークへのインバウンド接続トラフィックを制限するには、[任意の場所] ではなく [カスタム] を選択し、ネットワークの CIDR ブロックを入力してください。
   - (任意) 別のセキュリティグループを選択するには、[既存のセキュリティグループを選択する] を選択し、既存のセキュリティグループを選択してください。セキュリティグループにネットワークからの接続トラフィックを許可するルールがない場合、インスタンスに接続できるようになりません。Linux インスタンスでは、SSH トラフィックを許可する必要があります。Windows インスタンスでは、RDP トラフィックを許可する必要があります。

9. [ストレージを設定] では、ルートボリュームは設定されていますが、データボリュームは設定されていないことに注意してください。テスト目的にはこれで十分です。
10. Summary (概要) パネルでインスタンス設定の要約を確認します。準備が完了したら、[Launch instance] (インスタンスを起動) を選択してください。
11. 起動が成功した場合は、[成功] 通知からインスタンスの ID を選択して [インスタンス] ページを開き、起動のステータスをモニタリングします。
12. インスタンスのチェックボックスをオンにします。インスタンスの初期状態は `pending` です。インスタンスが起動されると、状態は `running` に変わります。[ステータスとアラーム] タブを

選択してください。インスタンスがステータスチェックに合格すると、接続リクエストを受信できる状態になります。

ステップ 2: インスタンスに接続する

使用する手順は、インスタンスのオペレーティングシステムによって異なります。インスタンスに接続できない場合は、「Amazon EC2 Linux インスタンスへの接続に関する問題のトラブルシューティング」を参照してください。

Linux インスタンス

任意の SSH クライアントを使用して Linux インスタンスに接続できます。コンピュータで Windows を実行している場合は、ターミナルを開き、ssh コマンドを実行して SSH クライアントがインストールされていることを確認します。コマンドが見つからない場合は、Windows 用 OpenSSH をインストールします。

SSH を使用してインスタンスに接続するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. 対象のインスタンスを選択し、[Connect] (接続) をクリックしてください。
4. [インスタンスに接続] ページで、[SSH クライアント] タブを選択してください。
5. (任意) インスタンスの起動時にキーペアを作成し、Linux または macOS を実行しているコンピュータにプライベートキー (.pem ファイル) をダウンロードした場合は、サンプルの chmod コマンドを実行してプライベートキーのアクセス許可を設定します。
6. サンプルの SSH コマンドをコピーします。以下の例では、*key-pair-name*.pem がプライベートキーファイルの名前、*ec2-user* がイメージに関連付けられたユーザー名、@ 記号以降の文字列がインスタンスのパブリック DNS 名になります。

   ```
   ssh -i key-pair-name.pem ec2-user@ec2-198-51-100-1.us-east-2.compute.amazonaws.com
   ```

7. コンピュータのターミナルウィンドウで、前の手順で保存した ssh コマンドを実行します。プライベートキーファイルが現在のディレクトリにない場合は、このコマンドでキーファイルへの完全修飾パスを指定する必要があります。

   以下に、応答の例を示します。

```
The authenticity of host 'ec2-198-51-100-1.us-east-2.compute.amazonaws.com
 (198-51-100-1)' can't be established.
ECDSA key fingerprint is l4UB/neBad9tvkgJf1QZWxheQmR59WgrgzEimCG6kZY.
Are you sure you want to continue connecting (yes/no)?
```

8. (任意) セキュリティアラートのフィンガープリントが、インスタンスを初めて起動するときにコンソール出力に含まれるインスタンスフィンガープリントと一致することを確認します。コンソール出力を取得するには、[アクション]、[モニタリングとトラブルシューティング]、[システムログの取得] を選択してください。これらのフィンガープリントが一致しない場合、何者かが中間者 (MITM) 攻撃を試みている可能性があります。一致した場合は、次のステップに進んでください。
9. **yes** と入力してください。

   以下に、応答の例を示します。

```
Warning: Permanently added 'ec2-198-51-100-1.us-
east-2.compute.amazonaws.com' (ECDSA) to the list of known hosts.
```

Windows インスタンス

RDP を使用して Windows インスタンスに接続するには、初期管理者パスワードを取得し、インスタンスに接続するときにこのパスワードを入力する必要があります。インスタンスの起動後、パスワードが利用可能になるまでに数分かかります。アカウントには、GetPasswordData アクションを呼び出すアクセス許可が必要です。詳細については、「Amazon EC2 API へのアクセスを制御するポリシーの例」を参照してください。

管理者アカウントのデフォルトのユーザー名は、AMI に含まれるオペレーティングシステム (OS) の言語によって異なります。正しいユーザー名を確認するには、AMI の OS の言語を特定し、対応するユーザー名を選択してください。例えば、英語 OS の場合、ユーザー名は `Administrator` で、フランス語 OS の場合は `Administrateur`、ポルトガル語 OS の場合は `Administrador` です。OS の言語バージョンに同じ言語のユーザー名がない場合は、ユーザー名 `Administrator` (`Other`) を選択してください。詳細については、Microsoft TechNet Wiki のLocalized Names for Administrator Account in Windowsを参照してください。

初期管理者パスワードを取得するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

2. ナビゲーションペインで、[インスタンス] を選択してください。
3. 対象のインスタンスを選択し、[Connect] (接続) をクリックしてください。
4. [インスタンスに接続] ページで、[RDP クライアント] タブを選択してください。
5. [ユーザー名] で、管理者アカウントのデフォルトのユーザー名を選択してください。選択するユーザー名は、インスタンスの起動に使用した AMI に含まれるオペレーティングシステム (OS) の言語と一致する必要があります。使用する OS と同じ言語のユーザー名がない場合は、[Administrator (Other)] を選択してください。
6. [パスワードを取得] を選択してください。
7. [Windows パスワードを取得] ページで、次の操作を行います。

   a. [プライベートキーファイルのアップロード] を選択し、インスタンスの起動時に指定したプライベートキー `.pem`ファイルに移動します。ファイルを選択した上で、[Open] (開く) を選択して、ファイルの内容をすべてウィンドウにコピーします。

   b. [パスワードを復号化] を選択してください。[Windows パスワードを取得] ページが閉じて、インスタンスのデフォルトの管理者パスワードが、[パスワード] の下に表示されます。前に表示されていた [パスワードを取得] のリンクは削除されます。

   c. パスワードをコピーして、安全な場所に保存します。このパスワードはインスタンスに接続するのに必要です。

次の手順では、Windows のリモートデスクトップ接続クライアント (MSTSC) を使用します。別の RDP クライアントを使用する場合は RDP ファイルをダウンロードし、RDP 接続を確立する手順については RDP クライアントのドキュメントを参照してください。

RDP クライアントを使用して Windows インスタンスに接続するには

1. [インスタンスに接続] ページで、[リモートデスクトップファイルのダウンロード] を選択してください。ファイルのダウンロードが完了したら、[キャンセル] を選択し、[インスタンス] ページに戻ります。RDP ファイルが `Downloads` フォルダにダウンロードされます。
2. `mstsc.exe` を実行して RDP クライアントを開きます。
3. [オプションを表示] を展開して [開く] を選択し、`Downloads` フォルダから .rdp ファイルを選択してください。
4. デフォルトでは、[コンピュータ] はインスタンスのパブリック IPv4 DNS 名で、[ユーザー名] は管理者アカウントです。代わりに IPv6 を使用してインスタンスに接続するには、インスタンスのパブリック IPv4 DNS 名を IPv6 アドレスに置き換えます。必要に応じて、デフォルト設定を確認して変更します。

5. [接続]を選択してください。リモート接続のパブリッシャーが不明であるという警告が表示された場合は、[接続] を選択して続行します。
6. 以前に保存したパスワードを入力し、[OK] を選択してください。
7. 自己署名証明書の性質上、セキュリティ証明書を認証できないという警告が表示されることがあります。次のいずれかを行います。
   - 証明書を信頼する場合は、[はい] を選択してインスタンスに接続します。
   - [Windows] 続行する前に、証明書のサムプリントとシステムログの値を比較して、リモートコンピュータの ID を確認します。[証明書を表示] を選択し、[詳細] タブから [サムプリント] を選択してください。この値を [アクション]、[モニタリングとトラブルシューティング]、[システムログの取得] の `RDPCERTIFICATE-THUMBPRINT` の値と比較します。
   - [Mac OS X] 続行する前に、証明書のフィンガープリントとシステムログの値を比較して、リモートコンピュータの ID を確認します。[証明書を表示] を選択し、[詳細] を展開し、[SHA1 フィンガープリント] を選択してください。この値を [アクション]、[モニタリングとトラブルシューティング]、[システムログの取得] の `RDPCERTIFICATE-THUMBPRINT` の値と比較します。
8. RDP 接続が成功すると、RDP クライアントに Windows ログイン画面が表示され、次に Windows デスクトップが表示されます。代わりにエラーメッセージが表示された場合は、「the section called “リモートデスクトップからリモートコンピュータに接続できません”」を参照してください。RDP 接続が完了したら、RDP クライアントを閉じることができます。

ステップ 3: インスタンスをクリーンアップする

> **Warning**
>
> インスタンスの終了は恒久的で、元に戻すことができません。
> インスタンスを終了すると、そのインスタンスには接続できなくなり、復元することもできません。アタッチされている Amazon EBS ボリュームで、終了時に削除されるように設定されているものも、すべて恒久的に削除され、復元できません。インスタンスストアボリュームに保存されているデータは、すべて恒久的に失われます。詳細については「インスタンスの終了方法」を参照してください。
> インスタンスを終了する前に、終了後も保持する必要があるすべてのデータを永続的ストレージにバックアップしたことを確認してください。

このチュートリアル用に作成したインスタンスを使用した操作が終了したら、インスタンスを終了してクリーンアップする必要があります。クリーンアップする前にこのインスタンスでやることがある場合は、「次のステップ」を参照してください。

インスタンスのステータスが `shutting down` または `terminated` に変更されるとすぐに、そのインスタンスまたは無料利用枠の制限にカウントされる使用量についての料金の発生が停止します。インスタンスを後で使用するために保持しながら、料金や無料利用枠の制限にカウントされる使用量が発生しないようにするために、インスタンスを今すぐ停止し、後で再度起動できます。詳細については、「Amazon EC2 インスタンスの停止と開始」を参照してください。

インスタンスを終了するには

1. ナビゲーションペインで、[インスタンス] を選択してください。インスタンスの一覧で、インスタンスを選択してください。
2. [インスタンスの状態]、[インスタンスの終了 (削除)] の順に選択してください。
3. 確認を求めるメッセージが表示されたら、[終了 (削除)] を選択してください。

   Amazon EC2 によって、インスタンスがシャットダウンおよび終了します。インスタンスの終了後、インスタンスはしばらくの間コンソールに表示されたままですが、エントリは自動的に削除されます。終了したインスタンスを自分でコンソールディスプレイから削除することはできません。

次のステップ

インスタンスを起動した後、次の手順を見てみるといいかもしれません。

- 最初のチュートリアルで、Amazon EC2 のコア概念を確認します。詳細については、「EC2 インスタンスを起動するためのチュートリアル」を参照してください。
- コンソールを使用して Amazon EC2 無料利用枠の使用状況を追跡する方法について説明します。詳細については、「the section called “無料利用枠の使用状況の追跡”」を参照してください。
- 使用量が無料利用枠を超えた場合に通知するように CloudWatch アラームを設定します (2025 年 7 月 15 日より前に作成されたアカウント向け)。詳細については、「AWS Billing User Guide」の「Tracking your AWS 無料利用枠 usage」を参照してください。
- EBS ボリュームの追加。詳細については、「Amazon EBS User Guide」の「Create an Amazon EBS volume」を参照してください。

- Run コマンドを使用してリモートに EC2 インスタンスを管理する方法を説明します。詳細については、AWS Systems Manager ユーザーガイドの「AWS Systems Manager実行コマンド」を参照してください。
- インスタンス購入オプションについてご覧ください。詳細については、「Amazon EC2 の請求および購入オプション」を参照してください。
- インスタンスタイプに関するアドバイスを取得します。詳細については、「EC2 インスタンスタイプファインダーからレコメンデーションを取得する」を参照してください。

Amazon EC2 のベストプラクティス

Amazon EC2 の利点を最大限に高めるために、以下のベストプラクティスを実践することをお勧めします。

セキュリティ

- AWS リソースおよび API へのアクセス管理は、可能な限り ID プロバイダーおよび IAM ロールによる ID フェデレーションを使用します。詳細については、 IAM ユーザーガイドの「ID プロバイダーとフェデレーション」を参照してください。
- セキュリティグループに対して、最小権限となるルールを適用します。
- 定期的にインスタンスのオペレーティングシステムやアプリケーションに対してパッチ処理、更新、および保護を行います。詳細については、「更新管理」を参照してください。Windows オペレーティングシステム固有のガイドラインについては、「Windows インスタンスにおけるセキュリティのベストプラクティス」を参照してください。
- Amazon Inspector を使用して、Amazon EC2 インスタンスを自動的に検出し、ソフトウェアの脆弱性やネットワークへの意図しない公開がないかスキャンします。詳細については、Amazon Inspector ユーザーガイドを参照してください。
- AWS Security Hub CSPM コントロールを使用して、Amazon EC2 リソースをセキュリティのベストプラクティスやセキュリティ基準に照らして監視します。セキュリティハブ CSPM の詳細については、「AWS Security Hub CSPM ユーザーガイド」の「Amazon Elastic Compute Cloud の管理」を参照してください。

ストレージ

- データの永続性、バックアップ、および復元に対するルートボリュームタイプの影響について理解します。詳細については、「ルートボリュームタイプ」を参照してください。
- オペレーティングシステム用およびデータ用として個別に Amazon EBS ボリュームを使用します。データのボリュームがインスタンス終了後も保持されることを確認します。詳細については、「インスタンスの終了時にデータを保持する」を参照してください。
- インスタンスで一時データの格納に使用できるインスタンスストアを使用します。インスタンスを停止、休止、または終了すると、インスタンスストアに格納されたデータは削除されることに注意してください。データベースストレージにインスタンスストアを使用する場合は、耐障害性を確保するレプリケーション係数が設定されたクラスターがあることを確認します。

- EBS ボリュームとスナップショットを暗号化します。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS 暗号化」を参照してください。

リソース管理

- AWS リソースを追跡および識別するために、インスタンスメタデータおよびリソースのカスタムタグを使用します。詳細については、「インスタンスメタデータを使用して EC2 インスタンスを管理する」および「Amazon EC2 リソースのタグ付け」を参照してください。
- Amazon EC2 の現在の制限を表示します。制限の引き上げに対するリクエストは、制限の引き上げが必要となる前に計画してください。詳細については、「Amazon EC2 の Service Quotas」を参照してください。
- AWS Trusted Advisor では、お客様の AWS 環境を検査し、コスト削減、システムの可用性とパフォーマンスの向上、セキュリティギャップの封鎖につながるレコメンデーションをお知らせします。詳細については、AWS サポート ユーザーガイドの AWS Trusted Advisorを参照してください。

バックアップと復旧

- 定期的に Amazon EBS スナップショットを使用して EBS ボリュームをバックアップし、インスタンスから Amazon Machine Image (AMI) を作成して、それ以降にインスタンスを起動するためのテンプレートとして設定を保存します。このユースケースの実現に役立つ AWS サービスの詳細については、「AWS Backup」と「Amazon Data Lifecycle Manager」を参照してください。
- 複数のアベイラビリティゾーンにアプリケーションの重要なコンポーネントをデプロイし、データを適切にレプリケートします。
- インスタンスが再開したときに、動的な IP アドレスを処理するアプリケーションを設計します。詳細については、「Amazon EC2 インスタンスの IP アドレス指定」を参照してください。
- イベントを管理し、対応します。詳細については、「Amazon EC2 リソースのモニタリング」を参照してください。
- フェイルオーバーを処理する準備が整っていることを確認します。基本的な解決策として、手動でネットワークインターフェイスをアタッチすることも、代替インスタンスに Elastic IP アドレスを関連付けることもできます。詳細については、「Elastic Network Interface」を参照してください。自動化されたソリューションとして Amazon EC2 Auto Scaling を使用できます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」を参照してください。
- インスタンスと Amazon EBS ボリュームを復元するプロセスを定期的にテストして、データとサービスが正常に復元されるようにします。

ネットワーク

- アプリケーションの TTL (有効時間) 値を IPv4 と IPv6 で 255 に設定します。小さい値を使用すると、アプリケーショントラフィックの送信中に TTL が期限切れになり、インスタンスの到達可能性の問題が発生する危険性があります。

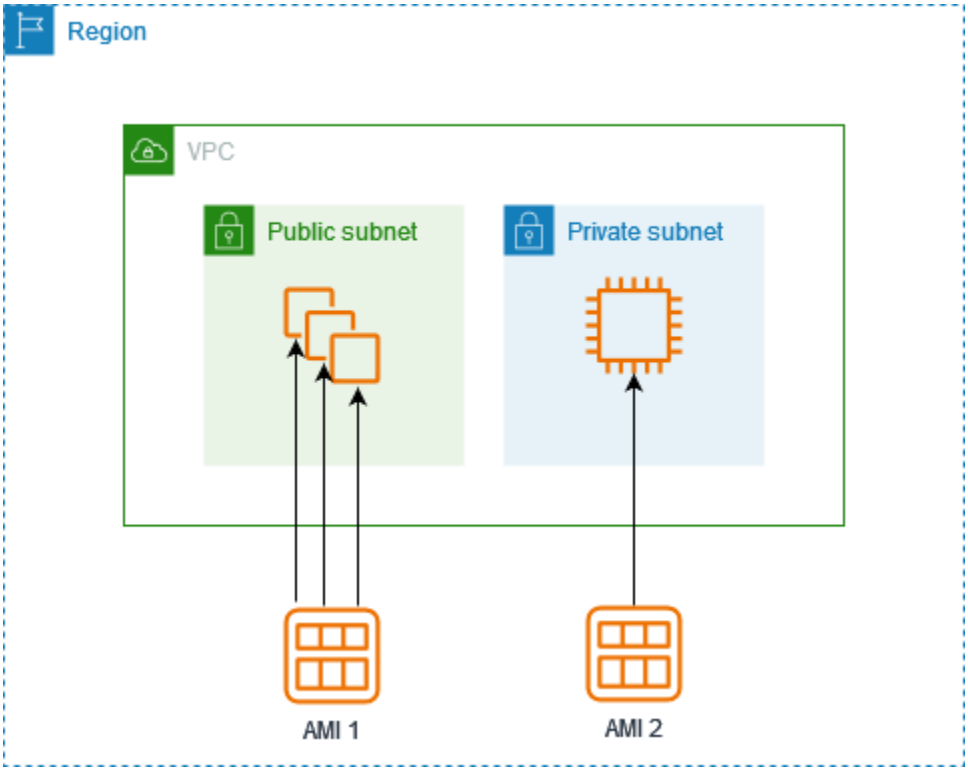
Amazon EC2 の Amazon マシンイメージ

Amazon マシンイメージ (AMI) は、Amazon EC2 インスタンスをセットアップして起動するために必要なソフトウェアを提供するイメージです。各 AMI にはブロックデバイスマッピングも含まれています。このマッピングで、起動するインスタンスにアタッチするブロックデバイスを指定します。インスタンスを起動するときは、AMI を指定する必要があります。この AMI には、インスタンスに選択したインスタンスタイプとの互換性が必要です。AWS が提供する AMI、パブリック AMI、他のユーザーと共有する AMI、AWS Marketplace から購入した AMI を使用できます。

AMI は、以下に基づいて指定します。

- リージョン
- オペレーティングシステム
- プロセッサアーキテクチャ
- ルートボリュームタイプ
- 仮想化タイプ

同じ設定で複数のインスタンスが必要な場合は、1 つの AMI から複数のインスタンスを起動できます。さまざまな設定のインスタンスが必要なときは、次の図に示すように、異なる AMI を使用してインスタンスを起動できます。



Amazon EC2 インスタンスから AMI を作成し、それを使用して同じ設定でインスタンスを起動できます。AMI を別の AWS リージョンにコピーし、そのリージョンでインスタンスを起動するために使用できます。作成した AMI を他のアカウントと共有して、同じ設定でインスタンスを起動することもできます。AWS Marketplace を使用して AMI を販売できます。

内容

- Amazon EC2 の AMI タイプと特性
- EC2 インスタンスの要件を満たす AMI を検索する
- Amazon EC2 インスタンスの AWS Marketplace の有料 AMI
- Amazon EC2 AMI ライフサイクル
- Amazon EC2 ブートモードでのインスタンスの起動動作
- EBS-backed AMI での暗号化の利用
- Amazon EC2 での共有 AMI の使用方法について
- Amazon EventBridge を使用して AMI イベントをモニタリングする
- AMI の請求情報について
- Amazon EC2 の AMI クォータ

Amazon EC2 の AMI タイプと特性

インスタンスを起動するときに選択する AMI には、選択したインスタンスタイプとの互換性が必要です。次の特性に基づき、使用する AMI を選択できます。

- リージョン
- オペレーティングシステム
- プロセッサアーキテクチャ
- 起動許可
- ルートボリュームタイプ
- 仮想化タイプ

起動許可

起動許可により、AMI を使用してインスタンスを起動できるユーザーが決定されます。起動許可は AMI の共有として考えることができます。起動許可を付与すると、AMI を他のユーザーと共有します。起動許可を指定して利用可能な状態を決定できるのは、AMI の所有者のみです。起動許可は次のように分類されます。

起動アクセス許可	説明
パブリック	所有者はすべての AWS アカウントに起動許可を与えます。
明示的	所有者は特定の AWS アカウント、組織、または組織単位 (OU) に起動許可を与えます。
暗示的	所有者には AMI の暗示的起動許可があります。

Amazon や Amazon EC2 コミュニティではさまざまなパブリック AMI を提供しています。詳細については、Amazon EC2 での共有 AMI の使用方法について を参照してください。デベロッパーは自分の AMI に料金を請求できます。詳細については、「Amazon EC2 インスタンスの AWS Marketplace の有料 AMI」を参照してください。

ルートボリュームタイプ

すべての AMI が Amazon EBS-backed と Amazon S3-backed のいずれかに分類されます。

- Amazon EBS-backed AMI – AMI から起動されるインスタンスのルートボリュームが、Amazon EBS スナップショットから作成される Amazon Elastic Block Store (Amazon EBS) ボリュームであることを意味します。Linux AMI と Windows AMI の両方でサポート。
- Amazon S3-backed AMI - AMI から起動されるインスタンスのルートボリュームが、Amazon S3 に保存されたテンプレートから作成されるインスタンスストアボリュームであることを意味します。Linux AMI でのみサポート。Windows AMI はルートボリュームのインスタンスストアをサポートしていません。

詳細については、「Amazon EC2 インスタンスのルートボリューム」を参照してください。

> **Note**
>
> Amazon S3-backed AMI はサポート終了扱いとなっており、新たに使用することは推奨されません。これらの AMI は、C1、C3、D2、I2、M1、M2、M3、R3、X1 の古いインスタンスタイプでのみサポートされています。

次の表では、2 種類の AMI を使用した場合の重要な相違点をまとめています。

特徴	Amazon EBS-backed AMI	Amazon S3-backed AMI
ルートボリューム	EBS ボリューム	インスタンスストアボリューム
インスタンスの起動時間	通常 1 分以内	通常 5 分以内
データの永続性	デフォルトでは、インスタンスを終了するとルートボリュームは削除されます。* EBS ボリュームにある他のデータはすべて、インスタンスの終了後もデフォルトで保持されます。	インスタンスストアボリューム上のデータは、インスタンスの存続中のみ使用できます。

特徴	Amazon EBS-backed AMI	Amazon S3-backed AMI
停止状態	停止状態になっている場合があります。インスタンスが停止して実行されていない場合でも、ルートボリュームは Amazon EBS に保持されます。	停止状態にすることはできません。インスタンスは実行中または終了しています。
変更	インスタンスの停止中に、インスタンスタイプ、カーネル、RAM ディスク、およびユーザーデータが変更可能	インスタンスの属性は、インスタンスを削除するまで固定。
料金	インスタンスの使用量、EBS ボリューム、また、EBS スナップショットとして保存した AMI に対して料金が発生します。	インスタンスの使用量や Amazon S3 に保存した AMI に対して料金が発生します。
AMI の作成/バンドル	単一のコマンドまたは呼び出しを使用	AMI ツールをインストールして使用する必要があります

* デフォルトでは、EBS ルートボリュームには `DeleteOnTermination` フラグが `true` に設定されています。このフラグを変更し、終了後もボリュームを保持する方法については、「Amazon EC2 インスタンスの終了後も Amazon EBS ルートボリュームを保持する」を参照してください。

** `io2` EBS ブロックエクスプレスのみでサポートされています。詳細については、「Amazon EBS ユーザーガイド」の「プロビジョンド IOPS SSD Block Express ボリューム」を参照してください。

AMI によって決まるルートボリュームタイプを識別する

EC2 インスタンスの起動に使用する AMI によってルートボリュームのタイプが決まります。EC2 インスタンスのルートボリュームは、EBS ボリュームまたはインスタンスストアボリュームのどちらかです。

Nitro ベースのインスタンスは、EBS ルートボリュームのみをサポートしています。インスタンスストアのルートボリュームをサポートしているインスタンスタイプは、次の旧世代のインスタンスタイプのみです: C1、C3、D2、I2、M1、M2、M3、R3、X1。

Console

AMI によって決まるルートボリュームタイプを識別するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI] をクリックした後、AMI を選択します。
3. 次のように、[Details] (詳細) タブで [Root device type] (ルートデバイスタイプ) の値を確認します。
   - `ebs` – この AMI から起動されたインスタンスは、EBS ルートボリュームを取得します。
   - `instance store` – この AMI から起動されたインスタンスは、インスタンスストアのルートボリュームを取得します。

AWS CLI

AMI によって決まるルートボリュームタイプを識別するには

describe-images コマンドを使用します。

```
aws ec2 describe-images \
    --image-ids ami-0abcdef1234567890 \
    --query Images[].RootDeviceType
```

以下は出力の例です。

```
ebs
```

PowerShell

AMI によって決まるルートボリュームタイプを識別するには

Get-EC2Image コマンドレットを使用します。

```
(Get-EC2Image `
    -ImageId ami-0abcdef1234567890).RootDeviceType.Value
```

以下は出力の例です。

```
ebs
```

仮想化タイプ

Amazon マシンイメージ では、2 つの仮想化タイプ (準仮想化 (PV) およびハードウェア仮想マシン (HVM)) のどちらかを使用します。PV AMI と HVM AMI の主な違いは、起動の方法と、パフォーマンス向上のための特別なハードウェア拡張機能 (CPU、ネットワーク、ストレージ) を利用できるかどうかという点です。Windows AMI は、HVM AMI です。

以下の表では、HVM と PV AMI を比較しています。

特性	HVM	PV
説明	HVM AMI は、完全に仮想化された一連のハードウェアを備えており、イメージのルートブロックデバイスのマスターブートレコードを実行することによって起動します。この仮想化タイプでは、ベアメタルハードウェア上でオペレーティングシステムが動作するのと同様に、修正を行わなくても仮想マシン上でオペレーティングシステムを直接実行することができます。Amazon EC2 ホストシステムでは、ゲストに提供されている基盤となるハードウェアの一部またはすべてがエミュレートされます。	PV AMI は、PV-GRUB と呼ばれる特別なブートローダーを使用して起動します。このブートローダーによって起動サイクルが開始され、イメージの `menu.lst` ファイルで指定されているカーネルがチェーンロードされます。準仮想化のゲストは、仮想化を明示的にサポートしていないホストハードウェアで実行できます。PV-GRUB の詳細や Amazon EC2 での使用方法については、「ユーザー提供カーネル」を参照してください。
サポートされるインスタンスタイプ	すべての現行世代のインスタンスタイプは HVM AMI をサポートします。	次の旧世代のインスタンスタイプは、PV AMI をサポートします: C1、C3、M1、

特性	HVM	PV
		M3、M2、および T1。現行世代のインスタンスタイプは PV AMI をサポートしません。
ハードウェア拡張のサポート	HVM のゲストは、ホストシステム上の基盤となるハードウェアへの高速なアクセスを可能にするハードウェア拡張を利用できます。拡張ネットワークや GPU 処理を使用するために必要です。専用のネットワークや GPU デバイスに命令を伝達するには、OS がネイティブハードウェアプラットフォームにアクセスできる必要があります。HVM 仮想化ではこのアクセスが可能です。詳細については、「Amazon EC2 インスタンスでの拡張ネットワーキング」を参照してください。	いいえ。拡張ネットワークや GPU 処理などの特別なハードウェア拡張を利用することはできません。
検索方法	コンソールまたは describe-images コマンドを使用して、AMI の仮想化タイプが hvm に設定されていることを確認します。	コンソールまたは describe-images コマンドを使用して、AMI の仮想化タイプが paravirtual に設定されていることを確認します。

PV on HVM

従来、準仮想化のゲストはストレージやネットワークの操作については、HVM のゲストよりも高いパフォーマンスを実現していました。これは、準仮想化のゲストでは I/O 用の特別なドライバー (ネットワークとディスクのハードウェアをエミュレートする際のオーバーヘッドが回避されます) を活用することができたためです。これに対して、HVM のゲストでは、エミュレートされたハードウェアに対する命令を変換する必要がありました。現在では、PV ドライバーを HVM のゲストで利

用できるようになりました。このため、準仮想化された環境で実行するためのができないオペレーティングシステムでも、これらのドライバーを使用することで、ストレージやネットワークの I/O でパフォーマンスの向上を確認することができます。このような PV on HVM ドライバーを使用すると、HVM のゲストで、準仮想化のゲストと同じまたはより優れたパフォーマンスを実現できます。

EC2 インスタンスの要件を満たす AMI を検索する

AMI には、インスタンスの起動に必要な、オペレーティングシステムやルートボリュームのタイプなどのコンポーネントとアプリケーションが含まれています。インスタンスを起動するには、要件を満たす AMI を検索する必要があります。

AMI を選択する際には、起動するインスタンスについて求められる可能性のある次の要件を検討してください。

- AMI ID としての AMI の AWS リージョンは、各リージョンに固有のものです。
- オペレーティングシステム (Linux、Windows など)。
- アーキテクチャ (32 ビット、64 ビット、64 ビット ARM など)。
- ルートボリュームタイプ (Amazon EBS、インスタンスストアなど)。
- プロバイダー (Amazonウェブサービス など)。
- 追加のソフトウェア (SQL サーバー など)。

Console

インスタンス起動ウィザードを使用するときに AMI のリストから選択するか、[Images] ページを使用して使用可能なすべての AMI を検索することができます。

インスタンス起動ウィザードを使用してクイック スタート AMI を検索するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションバーから、インスタンスを起動するリージョンを選択してください。お客様は場所に関係なく、使用できるリージョンをどれでも選択できます。AMI ID は各 AWS リージョンで固有です。
3. コンソールダッシュボードで [インスタンスの作成] を選択してください。
4. [アプリケーションと OS イメージ (Amazon マシンイメージ)] にある [クイックスタート] を選択し、インスタンスのオペレーティングシステム (OS) を選択してください。次

に、[Amazon マシンイメージ (AMI)] で、よく使用されている AMI の 1 つをリストから選択してください。必要な AMI が表示されていない場合は、[その他の AMI を閲覧] を選択して、AMI の全カタログを参照してください。詳細については、「アプリケーションと OS イメージ (Amazon マシンイメージ)」を参照してください。

AMI ページを使用して AMI を検索するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションバーから、インスタンスを起動するリージョンを選択してください。お客様は場所に関係なく、使用できるリージョンをどれでも選択できます。AMI ID は各 AWS リージョンで固有です。
3. ナビゲーションペインで [AMI] を選択してください。
4. (オプション) フィルターと検索オプションを使用すると、表示される AMI を限定して、条件に一致する AMI のみを表示できます。

   例えば、AWS で提供されるすべての AMI を表示するには、[パブリックイメージ] を選択してください。次に、検索オプションを使用して、AMI のリストに表示される範囲を指定します。検索バーをクリックし、メニューから [所有者エイリアス]、[=] 演算子の順に選択し、値として [Amazon] を選択してください。Linux や Windows など、特定のプラットフォームに対応する AMI を検索するには、[検索] バーを再度選択して [プラットフォーム] を選択し、そして [=] 演算子を選択して、表示されたリストからオペレーティングシステムを選択してください。
5. (オプション) [設定] アイコンを選択して、ルートボリュームタイプなど、表示するイメージ属性を選択してください。あるいは、一覧から AMI を選択し、[詳細] タブにそのプロパティを表示できます。
6. AMI を選択する前に、その AMI が インスタントストアバック と Amazon EBS-Backed のどちらであるかを確認し、その違いを認識しておくことが重要です。詳細については、「ルートボリュームタイプ」を参照してください。
7. この AMI からインスタンスを起動するには、インスタンスを選択し、[イメージからのインスタンスの起動] を選択してください。コンソールを使用してインスタンスを起動する方法については、コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する を参照してください。まだインスタンスを起動する準備ができていない場合は、後で使用するために AMI ID を記録します。

AWS CLI

describe-images コマンドを使用して、要件を満たす AMI を見つけます。デフォルトでは、このコマンドは、パブリックの AMI、ユーザーが所有している AMI、ユーザーに共有されている AMI、のすべてを返します。

Amazon が所有する AMI を見つけるには

describe-instances コマンドを `--owners` オプションで使用します。

```
aws ec2 describe-images --owners amazon
```

Windows AMI を見つけるには

次のフィルターを使用すると、Windows AMI のみが表示されます。

```
--filters "Name=platform,Values=windows"
```

EBS-backed AMI を見つけるには

Amazon EBS がバックアップしている AMI のみを表示するときは、次のフィルターを追加します。

```
--filters "Name=root-device-type,Values=ebs"
```

PowerShell

Get-EC2Image コマンドレットを使用して、要件を満たす AMI を見つけます。デフォルトでは、このコマンドレットは、パブリックの AMI、ユーザーが所有している AMI、ユーザーに共有されている AMI、のいずれかをすべて返します。

Amazon が所有する AMI を見つけるには

Get-EC2Image コマンドを `-Owner` パラメータと共に使用します。

```
Get-EC2Image -Owner amazon
```

Windows AMI を見つけるには

次のフィルターを使用すると、Windows AMI のみが表示されます。

```
-Filter @{Name="platform"; Values="windows"}
```

その他の例については、「AWS Tools for PowerShell ユーザーガイド」の「Find an Amazon Machine Image Using Windows PowerShell」を参照してください。

関連リソース

特定のオペレーティングシステムの AMI の詳細については、以下を参照してください。

- Amazon Linux 2023 – 「Amazon Linux 2023 ユーザーガイド」の「Amazon EC2 での AL2023」を参照してください。
- Ubuntu – Canonical Ubuntu ウェブサイトの「Amazon EC2 AMI Locator」を参照してください。
- RHEL – Red Hat ウェブサイトの「Red Hat Enterprise Linux Images (AMI) Amazon ウェブサービス (AWS) で利用可能」を参照してください。
- Windows Server – AWS Windows AMI リファレンス

AWS Marketplace でサブスクライブできる AMI の詳細については、「Amazon EC2 インスタンスの AWS Marketplace の有料 AMI」を参照してください。

Systems Manager を使用してユーザーがインスタンスの起動時に使用する必要がある最新の AMI を見つけやすくする方法については、以下を参照してください。

- Systems Manager パラメータを使用して AMI を参照する
- Systems Manager パブリックパラメータを使用して最新の AMI を参照する

Systems Manager パラメータを使用して AMI を参照する

Amazon EC2 コンソールで EC2 インスタンス起動ウィザードを使用してインスタンスを起動する場合は、リストから AMI を選択するか、AMI ID (このセクションで説明) を指す AWS Systems Manager パラメータを選択できます。オートメーションコードを使用してインスタンスを作成する場合は、AMI ID の代わりに Systems Manager パラメータを指定できます。

システム管理 パラメータは、システム管理 パラメータストアで作成できるユーザー定義のキーと値のペアです。パラメータストアは、アプリケーションの設定値を外部化するための一元的なストアを提供します。詳細については、AWS Systems Manager ユーザーガイドの「AWS システム管理 パラメータストア」を参照してください。

AMI ID をポイントするパラメータを作成する場合は、データ型に aws:ec2:image を指定してください。このデータ型を指定すると、パラメータの作成時または変更時にパラメータ値が AMI ID として検証されます。詳細については、AWS Systems Manager ユーザーガイドの「Amazon マシンイメージ ID のパラメータのネイティブサポート」を参照してください。

コンテンツ

- ユースケース
- アクセス許可
- 制限事項
- システム管理 パラメータを使用したインスタンスの起動

ユースケース

システム管理 パラメータを使用して AMI ID を指すようにすると、ユーザーがインスタンスの起動時に適切な AMI を簡単に選択できるようになります。システム管理 パラメータにより、オートメーションコードのメンテナンスを簡素化することもできます。

ユーザーの利便性の向上

特定の AMI を使用してインスタンスを作成する必要があり、その AMI を定期的に更新する場合は、AMI を見つけるために システム管理 パラメータを選択するようにユーザーに求めることをお勧めします。ユーザーに システム管理 パラメータを選択するように求めると、インスタンスの起動に最新の AMI が使用されるようになります。

例えば、組織内で毎月最新のオペレーティングシステムとアプリケーションパッチが適用された AMI の新しいバージョンを作成するとします。また、ユーザーに最新バージョンの AMI を使用してインスタンスを作成するように求めるとします。ユーザーに最新バージョンを確実に使用させるには、適切な AMI ID をポイントする システム管理 パラメータ (例: golden-ami など) を作成することができます。新しいバージョンの AMI を作成するたびに、パラメータの AMI ID の値を更新して常に最新の AMI をポイントするようにします。ユーザーは毎回同じ システム管理 パラメータを選択し続けるため、AMI の定期的な更新について知る必要はありません。AMI に システム管理 パラメータを使用すると、ユーザーはインスタンスの起動に適切な AMI を簡単に選択できるようになります。

オートメーションコードの保守の簡素化

オートメーションコードを使用してインスタンスを作成する場合は、AMI ID の代わりに システム管理 パラメータを指定できます。新しいバージョンの AMI を作成したら、最新の AMI を指すようにパラメータの AMI ID の値を変更できます。パラメータを参照するオートメーションコードは、新しい

バージョンの AMI を作成するたびに修正する必要はありません。これにより、オートメーションのメンテナンスが簡素化されるため、デプロイのコストの削減に役立ちます。

> **Note**
>
> システム管理 パラメータが指す AMI ID を変更しても、実行中のインスタンスは影響を受けません。

アクセス許可

インスタンス起動ウィザードで AMI ID をポイントする システム管理 パラメータを使用する場合は、IAM ポリシーに次のアクセス許可を追加する必要があります。

- `ssm:DescribeParameters` – システム管理 パラメータを表示および選択するためのアクセス許可を付与します。
- `ssm:GetParameters` – システム管理 パラメータの値を取得するためのアクセス許可を付与します。

また、特定の システム管理 パラメータへのアクセスを制限することもできます。詳細と IAM ポリシーの例については、「例: EC2 インスタンス起動ウィザードの使用」を参照してください。

制限事項

AMI と システム管理 パラメータはリージョンに固有です。リージョン間で同じ システム管理 パラメータ名を使用するには、それぞれのリージョンで同じ名前の システム管理 パラメータを作成します (例: `golden-ami` など)。それぞれのリージョンの システム管理 パラメータでそのリージョンの AMI をポイントします。

パラメータ名では大文字と小文字が区別されます。パラメータ名のバックスラッシュは、パラメータが階層の一部である場合にのみ必要です (例: `/amis/production/golden-ami`)。パラメータが階層の一部でない場合は、バックスラッシュを省略できます。

システム管理 パラメータを使用したインスタンスの起動

インスタンスを起動するときは、AMI ID を指定する代わりに、AMI ID をポイントする Systems Manager パラメータを指定できます。

このパラメータをプログラム的に指定するには、以下の構文を使用します。この構文では、`resolve:ssm` が標準のプレフィクス、`parameter-name` が一意のパラメータ名になります。

```
resolve:ssm:parameter-name
```

システム管理 パラメータでは、バージョンがサポートされています。パラメータの各バージョンには、一意のバージョン番号が割り当てられます。パラメータのバージョンは以下のように参照できます。version は一意のバージョン番号です。デフォルトでは、パラメータのバージョンを指定しない場合は最新バージョンが使用されます。

```
resolve:ssm:parameter-name:version
```

AWS が提供するパブリックパラメータを使用してインスタンスを起動するには、「Systems Manager パブリックパラメータを使用して最新の AMI を参照する」を参照してください。

Console

Systems Manager パラメータを使用して AMI を検索する

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションバーから、インスタンスを起動するリージョンを選択してください。お客様は場所に関係なく、使用できるリージョンをどれでも選択できます。
3. コンソールダッシュボードで [インスタンスの作成] を選択してください。
4. [アプリケーションおよび OS イメージ (Amazon マシンイメージ)] で、[その他の AMI を閲覧] を選択してください。
5. 検索バーの右側にある矢印ボタンを選択し、[システム管理 パラメータで検索] を選択してください。
6. [システム管理 パラメータ] でパラメータを選択してください。対応する AMI ID が [現在、以下に解決されています] の下に表示されます。
7. [検索] を選択してください。AMI ID に一致する AMI がリストに表示されます。
8. リストから AMI を選択し、[選択] を選択してください。

インスタンス起動ウィザードを使用してインスタンスを起動する方法の詳細については、「コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する」を参照してください。

AWS CLI

Systems Manager パラメータを使用してインスタンスを起動する

`--image-id` オプションで run-instances コマンドを使用します。この例では、AMI ID を指定する **`golden-ami`** という名前の Systems Manager パラメータを使用します。

```
--image-id resolve:ssm:/golden-ami
```

Systems Manager パラメータにはバージョンを作成できます。次の例では、**`golden-ami`** パラメータのバージョン 2 を指定します。

```
--image-id resolve:ssm:/golden-ami:2
```

PowerShell

Systems Manager パラメータを使用してインスタンスを起動する

`-ImageId` パラメータで New-EC2Instance コマンドレットを使用します。この例では、AMI ID を指定する **`golden-ami`** という名前の Systems Manager パラメータを使用します。

```
-ImageId "resolve:ssm:/golden-ami"
```

Systems Manager パラメータにはバージョンを作成できます。次の例では、**`golden-ami`** パラメータのバージョン 2 を指定します。

```
-ImageId "resolve:ssm:/golden-ami:2"
```

Systems Manager パブリックパラメータを使用して最新の AMI を参照する

AWS Systems Manager には、AWS によって維持されるパブリック AMI 用のパブリックパラメータがあります。インスタンスの起動時にパブリックパラメータを使用することで、確実に最新の AMI を使用することができます。例えば、パブリックパラメータ `/aws/service/ami-amazon-linux-latest/al2023-ami-kernel-default-arm64` はすべてのリージョンで使用でき、特定のリージョンの arm64 アーキテクチャ対応 Amazon Linux 2023 AMI の最新バージョンを常にポイントします。

パブリックパラメータは次のパスから使用できます。

- Linux – `/aws/service/ami-amazon-linux-latest`
- Windows – `/aws/service/ami-windows-latest`

詳細については、「AWS Systems Manager ユーザーガイド」の「Parameter Store でのパブリックパラメータの使用」を参照してください。

パブリック AMI の SSM パラメータを検索する

パブリック AMI を記述する場合、レスポンスには PublicSsmParameterName フィールドに関連付けられた AWS Systems Manager (SSM) パラメータが含まれます。このパラメータを使用して、常に AMI の最新バージョンを指す SSM パラメータを識別します。

Console

パブリック AMI の SSM パラメータを検索するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. パブリック AMI を選択します。
4. [詳細] タブで、[パブリック SSM パラメータ名] を見つけます。

AWS CLI

パブリック AMI の SSM パラメータを検索するには

describe-images コマンドを使用します。レスポンスには、関連付けられた SSM パラメータを持つパブリック AMI の PublicSsmParameterName フィールドが含まれます。

```
aws ec2 describe-images \
    --image-ids ami-0abcdef1234567890
```

次の出力例では、PublicSsmParameterName フィールドが示されます。一部のフィールドは簡潔さのために省略されています。

```
{
    "Images": [
        {
            "ImageId": "ami-0abcdef1234567890",
            "Name": "al2023-ami-2023.7.20260601.0-kernel-6.1-x86_64",
            ...
            "PublicSsmParameterName": "aws/service/ami-amazon-linux-latest/al2023-
ami-kernel-default-x86_64"
        }
```

```
    ]
}
```

PowerShell

パブリック AMI の SSM パラメータを検索するには

Get-EC2Image コマンドレットを使用します。レスポンスには、関連付けられた SSM パラメータを持つパブリック AMI の PublicSsmParameterName フィールドが含まれます。

```
Get-EC2Image -ImageId ami-0abcdef1234567890
```

次の出力例では、PublicSsmParameterName フィールドが示されます。一部のフィールドは簡潔さのために省略されています。

```
ImageId                : ami-0abcdef1234567890
Name                   : al2023-ami-2023.7.20260601.0-kernel-6.1-x86_64
...
PublicSsmParameterName : aws/service/ami-amazon-linux-latest/al2023-ami-kernel-
default-x86_64
```

Amazon Linux AMI の一覧表示

AWS CLI

現在の AWS リージョンの Linux AMI を一覧表示する

次の get-parameters-by-path コマンドを使用します。--path パラメータの値は Linux AMI に固有の値です。

```
aws ssm get-parameters-by-path \
    --path /aws/service/ami-amazon-linux-latest \
    --query "Parameters[].Name"
```

PowerShell

現在の AWS リージョンの Linux AMI を一覧表示する

Get-SSMParametersByPath コマンドレットを使用します。

```
Get-SSMParametersByPath `
```

```
    -Path "/aws/service/ami-amazon-linux-latest" | `
    Sort-Object Name | Format-Table Name
```

Windows AMI の一覧表示

AWS CLI

現在の AWS リージョンの Windows AMI を一覧表示する

次の get-parameters-by-path コマンドを使用します。--path パラメータの値は Windows AMI に固有の値です。

```
aws ssm get-parameters-by-path \
    --path /aws/service/ami-windows-latest \
    --query "Parameters[].Name"
```

PowerShell

現在の AWS リージョンの Windows AMI を一覧表示する

Get-SSMParametersByPath コマンドレットを使用します。

```
Get-SSMParametersByPath `
    -Path "/aws/service/ami-windows-latest" | `
    Sort-Object Name | Format-Table Name
```

パブリックパラメータを使用したインスタンスの起動

インスタンスの起動時にパブリックパラメータを指定するには、`resolve:ssm:`*`public-parameter`* 構文を使用します。この構文では、`resolve:ssm` が標準のプレフィクス、*`public-parameter`* がパブリックパラメータのパスと名前です。

AWS CLI

パブリックパラメータを使用してインスタンスを作成するには

`--image-id` オプションで run-instances コマンドを使用します。この例では、イメージ ID の Systems Manager パブリックパラメータを指定して、最新の Amazon Linux 2023 AMI でインスタンスを起動します。

```
--image-id resolve:ssm:/aws/service/ami-amazon-linux-latest/al2023-ami-kernel-default-x86_64
```

PowerShell

パブリックパラメータを使用してインスタンスを作成するには

`-ImageId` パラメータで New-EC2Instance コマンドレットを使用します。この例では、イメージ ID の Systems Manager パブリックパラメータを指定して、Windows Server 2022 向けの最新 AMI でインスタンスを起動します。

```
-ImageId "resolve:ssm:/aws/service/ami-windows-latest/Windows_Server-2022-English-Full-Base"
```

Systems Manager パラメータを使用するその他の例については、「AWS Systems Manager Parameter Storeを使用して最新の Amazon Linux AMI ID のクエリする」および「AWS Systems Manager Parameter Storeを使用して最新の Windows AMI をクエリする」を参照してください。

Amazon EC2 インスタンスの AWS Marketplace の有料 AMI

有料 AMI は、AWS Marketplace で販売されている AMI です。AWS Marketplace は、EC2 インスタンスの起動に使用できる AMI など、AWS で実行されるソフトウェアを購入できるオンラインストアです。AWS Marketplace AMI はデベロッパーツールなどカテゴリ別に整理されており、ユーザーは要件に適合する製品を見つけることができます。AWS Marketplace の詳細については、AWS Marketplace のウェブサイトを参照してください。

Red Hat のような組織のサービス契約に付属する AMI など、サードパーティーの AMI を AWS Marketplace で購入できます。また、AMI を作成し、AWS Marketplace で他の Amazon EC2 ユーザーに販売することもできます。安全で信頼性が高く、便利な AMI を作成して、一般公開する手順はきわめて単純で、いくつかのシンプルなガイドラインにしたがうだけです。共有 AMI の作成および使用方法の詳細については、Amazon EC2 での共有 AMI の使用方法について を参照してください。

有料 AMI からのインスタンスの起動は、他の AMI からのインスタンスの起動と同じです。追加パラメータは必要ありません。インスタンスは、AMI の所有者が設定した料金と、Amazon EC2 で m5.small インスタンスタイプを実行する場合の 1 時間あたりの料金など、関連ウェブサービスの標準使用料に基づいて課金されます。税金が加算されることもあります。有料 AMI の所有者は、特定のインスタンスがその有料 AMI から起動されたかどうかを確認できます。

> **Important**
>
> Amazon DevPay は新しい販売者または製品の受付を停止しました。現在は AWS Marketplace が、ソフトウェアとサービスを AWS で販売しており、唯一の統一された e コマースプラットフォームとなっています。AWS Marketplace でソフトウェアをデプロイし販売する方法については、AWS マーケットプレイス での販売を参照してください。AWS Marketplace は Amazon EBS-バック の AMI をサポートしています。

内容

- AWS Marketplace で AMI を販売する
- 有料 AMI の検索
- AWS Marketplace で有料 AMI を購入する
- インスタンスから AWS Marketplace 製品コードを取得する
- サポートされている AWS Marketplace 製品の有料サポートを使用する
- 有料およびサポートされる AMI の請求書
- AWS Marketplace サブスクリプションを管理する

AWS Marketplace で AMI を販売する

AWS Marketplace を使用して AMI を販売できます。AWS Marketplace では体系的に買い物をすることができます。また AWS Marketplace は、Amazon EBS-backed AMI、リザーブドインスタンス、スポットインスタンスなどの AWS 機能もサポートしています。

AWS Marketplace で AMI を販売する方法の詳細については、AWS マーケットプレイス での販売を参照してください。

有料 AMI の検索

有料 AMI は、購入可能な Amazon マシンイメージ (AMI) です。有料 AMI には製品コードもあります。購入可能な AMI は AWS Marketplace で見つけることができます。

Console

有料 AMI を検索するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

2. ナビゲーションペインで [AMI] を選択してください。
3. 最初のフィルタで、[パブリックイメージ] を選択してください。
4. 次のいずれかを行います。
   - 製品コードがわかっている場合は、[製品コード] 、[=] の順に選択し、製品コードを入力してください。
   - 製品コードがわからない場合は、検索バーでフィルター Owner alias=aws-marketplace を指定します。必要に応じて追加のフィルターを指定します。
5. AMI の ID を保存します。

AWS CLI

有料 AMI を検索するには

次の describe-images コマンドを使用します。

```
aws ec2 describe-images --owners aws-marketplace
```

出力には多数のイメージが含まれます。フィルターを指定することで、必要な AMI を識別することができます。AMI を見つけたら、次のコマンドでその ID を指定して製品コードを取得します。

```
aws ec2 describe-images \
    --image-ids ami-0abcdef1234567890 \
    --query Images[*].ProductCodes[].ProductCodeId
```

以下は出力の例です。

```
[
    "cdef1234abc567def8EXAMPLE"
]
```

製品コードがわかっている場合は、結果を製品コードでフィルタリングすることができます。次の例は、指定された製品コードを持つ最新の AMI を返します。

```
aws ec2 describe-images \
    --filters "Name=product-code,Values=cdef1234abc567def8EXAMPLE" \
    --query "sort_by(Images, &CreationDate)[-1].[ImageId]"
```

PowerShell

有料 AMI を検索するには

Get-EC2Image コマンドレットを使用します。

```
Get-EC2Image -Owner aws-marketplace
```

出力には多数のイメージが含まれます。フィルターを指定することで、必要な AMI を識別することができます。AMI を見つけたら、次のコマンドでその ID を指定して製品コードを取得します。

```
(Get-EC2Image -ImageId ami-0abcdef1234567890).ProductCodes
```

以下は出力の例です。

```
ProductCodeId             ProductCodeType
-------------             ---------------
cdef1234abc567def8EXAMPLE marketplace
```

製品コードがわかっている場合は、結果を製品コードでフィルタリングすることができます。次の例は、指定された製品コードを持つ最新の AMI を返します。

```
(Get-EC2Image -Owner aws-marketplace -Filter @{"Name"="product-
code";"Value"="cdef1234abc567def8EXAMPLE"} | sort CreationDate -Descending | Select-
Object -First 1).ImageId
```

AWS Marketplace で有料 AMI を購入する

AMI を使用して Amazon EC2 インスタンスを起動するには、有料 AMI にサインアップする (購入する) 必要があります。

通常、有料 AMI の販売者は、価格や購入サイトへのリンクなど、AMI に関する情報を提供します。リンクをクリックすると、最初に AWS へのログインが求められます。ログイン後、AMI を購入できます。

コンソールを使用した有料 AMI の購入

Amazon EC2 Launch Wizard を使用して有料 AMI を購入できます。詳細については、AWS Marketplace AMI から Amazon EC2 インスタンスを起動する を参照してください。

AWS Marketplace を使用した製品の登録

AWS Marketplace を使用するには、AWS アカウントが必要です。AWS Marketplace 製品からインスタンスを起動するには、Amazon EC2 サービスを利用するためのサインアップと、インスタンスを起動する製品のサブスクリプションが必要です。次のいずれかの方法を使用して、AWS Marketplace で製品をサブスクライブします。

- AWS Marketplace ウェブサイト: 1-Click デプロイメント機能で、事前に設定したソフトウェアをすばやく起動できます。詳細については、「AWS MarketplaceにおけるAMIベースの製品」を参照してください。
- Amazon EC2 Launch Wizard : AMI を検索し、ウィザードからインスタンスを直接起動できます。詳細については、「AWS Marketplace AMI から Amazon EC2 インスタンスを起動する」を参照してください。

インスタンスから AWS Marketplace 製品コードを取得する

インスタンスの AWS Marketplace 製品コードは、インスタンスメタデータを使用して取得できます。インスタンスに製品コードが含まれる場合、Amazon EC2 はそれを返します。メタデータの取得については、EC2 インスタンスのインスタンスメタデータにアクセスする を参照してください。

IMDSv2

Linux

次のコマンドを Linux インスタンスから実行します。

```
TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
    && curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/product-codes
```

Server

次のコマンドレットを Windows インスタンスから実行します。

```
[string]$token = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token-ttl-seconds" = "21600"} `
    -Method PUT -Uri http://169.254.169.254/latest/api/token
```

```
Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} `
    -Method GET -Uri http://169.254.169.254/latest/meta-data/product-codes
```

IMDSv1

Linux

次のコマンドを Linux インスタンスから実行します。

```
curl http://169.254.169.254/latest/meta-data/product-codes
```

Server

次のコマンドを Windows インスタンスから実行します。

```
Invoke-RestMethod -Uri http://169.254.169.254/latest/meta-data/product-codes
```

サポートされている AWS Marketplace 製品の有料サポートを使用する

Amazon EC2 は、デベロッパーがソフトウェア (またはそれに由来する AMI) のサポートを提供できるように手配します。デベロッパーは、お客様がサインアップして使用できるサポート製品を提供することができます。サポート製品にサインアップすると、デベロッパーはお客様に製品コードを渡します。お客様はそのコードをご自分の AMI に関連付ける必要があります。これにより、デベロッパーは、ユーザーのインスタンスがサポート対象であることを確認できます。また、お客様が製品からインスタンスを実行すると、デベロッパーが定めた製品の利用規約にしたがい、お客様に課金されます。

制限事項

- 製品コード属性は、一度設定すると、変更したり削除したりすることはできません。
- リザーブドインスタンス とともにサポート製品を使用することはできません。お客様は常に、サポート製品の販売者が指定した価格を支払います。

AWS CLI

製品コードを AMI に関連付けるには

modify-image-attribute コマンドを使用します。

```
aws ec2 modify-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --product-codes "cdef1234abc567def8EXAMPLE"
```

PowerShell

製品コードを AMI に関連付けるには

Edit-EC2ImageAttribute コマンドレットを使用します。

```
Edit-EC2ImageAttribute `
    -ImageId ami-0abcdef1234567890 `
    -ProductCode "cdef1234abc567def8EXAMPLE"
```

有料およびサポートされる AMI の請求書

有料またはサポートされた AMI の使用料金がお客様のクレジットカードに請求され、その金額を記載した E メールが毎月末に届きます。これは通常の Amazon EC2 使用料金とは別に請求されます。詳細については、AWS Marketplace 購入者ガイドの「製品の支払い」をご参照ください。

AWS Marketplace サブスクリプションを管理する

AWS Marketplace ウェブサイトでは、サブスクリプションの詳細の確認、使用に関するベンダー指示の表示、サブスクリプションの管理などを行うことができます。

サブスクリプションの詳細を確認する

サブスクリプションの詳細を確認するには

1. AWS Marketplace にログインします。
2. [Your マーケットプレイス Account] を選択してください。
3. [Manage your software subscriptions] を選択してください。
4. 現在のすべてのサブスクリプションが表示されます。実行中のインスタンスに接続するためのユーザー名など、製品の使用に関する特定の手順を表示するには、[使用手順] を選択してください。

サブスクリプションのキャンセル

> **Note**
>
> - サブスクリプションをキャンセルしても、その AMI を使用して起動したインスタンスは終了されません。実行中のインスタンスについては、終了するまで引き続き料金が請求されます。サブスクリプションに対する料金の請求を停止するには、この AMI を使用して起動したすべてのインスタンスを終了する必要があります。
> - 受信登録をキャンセルすると、その AMI からインスタンスを起動できなくなります。その AMI を再度使用するには、AWS Marketplace ウェブサイトまたは Amazon EC2 コンソールの起動ウィザードを使用して、その AMI を再度サブスクライブする必要があります。

AWS Marketplace のサブスクリプションをキャンセルするには

1. サブスクリプションに対する料金の請求を停止するには、サブスクリプションによって実行されていたすべてのインスタンスが終了していることを確認してください。

   > **Warning**
   >
   > インスタンスの終了は恒久的で、元に戻すことはできません。
   > インスタンスを終了すると、そのインスタンスには接続できなくなり、復元することもできません。アタッチされている Amazon EBS ボリュームで、終了時に削除されるように設定されているものも、すべて恒久的に削除され、復元できません。インスタンスストアボリュームに保存されているデータは、すべて恒久的に失われます。詳細については、「インスタンスの終了方法」を参照してください。
   > インスタンスを終了する前に、終了後も保持する必要があるすべてのデータを永続的ストレージにバックアップしたことを確認してください。

   a. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
   b. ナビゲーションペインで、[インスタンス] を選択してください。
   c. インスタンスを選択し、[インスタンスの状態]、[インスタンスの終了 (削除)] の順に選択してください。
   d. 確認を求めるメッセージが表示されたら、[終了 (削除)] を選択してください。

2. AWS Marketplace にログインし、[自分の マーケットプレイス アカウント] 、[ソフトウェアサブスクリプションの管理] の順に選択してください。
3. [購読の取り消し] を選択してください。取り消しの確認を求めるプロンプトが表示されます。

Amazon EC2 AMI ライフサイクル

Amazon マシンイメージ (AMI) は、インスタンスのセットアップと起動に必要なソフトウェアを提供するイメージです。インスタンスを起動するときは、AMI を指定する必要があります。Amazon が提供する AMI を使用するか、独自の AMI を作成できます。AMI は、インスタンスを起動する AWS リージョン に配置する必要があります。

AMI のライフサイクルには、AMI の作成、コピー、非推奨化、無効化、削除 (登録解除) が含まれます。

AMI を作成する Amazon は、インスタンスの起動に使用できる AMI を提供していますが、ニーズに合わせてカスタム AMI を作成することもできます。カスタム AMI を作成するには、既存の AMI からインスタンスを起動し、インスタンスをカスタマイズし (ソフトウェアのインストールやオペレーティングシステムの設定など) 、インスタンスから AMI を作成します。インスタンスのカスタマイズはすべて AMI に保存されるため、新しい AMI から起動されたインスタンスにはこれらのカスタマイズが含まれます。

Attestable AMI EC2 インスタンスアテステーションをサポートする AMI を作成するには、「構成証明可能 AMI」を参照してください。

AMI をコピーする AMI を使用してインスタンスを起動できるのは、AMI が配置されている AWS リージョン 内のみです。複数のリージョンで同じ設定でインスタンスを起動する必要がある場合は、AMI を他のリージョンにコピーします。

AMI を非推奨にする AMI を旧版または期限切れとしてマークするには、即時または将来の非推奨日を設定できます。非推奨の AMI は AMI リストに表示されませんが、AMI ID がわかっている場合、ユーザーとサービスは非推奨の AMI を引き続き使用できます。

AMI を無効にする AMI の使用を一時的に使用できないようにするには、AMI を無効にします。AAMI が無効になっていると、その AMI を使用して新しいインスタンスを起動できません。ただし、AMI を再度有効にすると、インスタンスを再度起動できるようになります。なお、AMI から既に起動されている既存のインスタンスは、AMI を無効にしても影響を受けません。

AMI の登録解除 (削除) AMI が不要になった場合は、登録を解除して、新しいインスタンスの起動に使用できなくなります。AMI が保持ルールと一致すると、AMI はごみ箱に移動し、保持期間が終了

する前に復元できます。その後、AMI は完全に削除されます。保持ルールと一致しない場合、そのルールは直ちに完全に削除されます。AMI の登録を解除しても、AMI から起動された既存のインスタンスには影響しないことに注意してください。

AMI ライフサイクルの自動化 Amazon Data Lifecycle Manager を使用して、Amazon EBS-backed AMI とそのバッキングスナップショットの作成、保持、コピー、非推奨、登録削除を自動化できます。EC2 Image Builder を使用して、カスタマイズされた AMI の作成、管理、デプロイを自動化することもできます。詳細については、「Amazon EBS ユーザーガイド」および「EC2 Image Builder ユーザーガイド」の「Automate backups with Amazon Data Lifecycle Manager」を参照してください。

内容

- Amazon EBS-backed AMI を作成する
- Amazon S3-backed AMI を作成する
- Windows Sysprep を使用して Amazon EC2 AMI を作成する
- Amazon EC2 AMI のコピー
- S3 を使用して AMI を保存および復元する
- AMI 祖先を使用して AMI のオリジンをトレースする
- AMI ウォーターマークを使用して AMI を追跡および識別する
- AMI の使用を管理およびモニタリングする
- Amazon EC2 AMI を非推奨にする
- Amazon EC2 AMI を無効にする
- Amazon EC2 AMI の登録を解除する

Amazon EBS-backed AMI を作成する

Amazon EC2 インスタンスまたは Amazon EC2 インスタンスのルートボリュームのスナップショットから、独自の Amazon EBS-backed AMI を作成できます。

インスタンスから Amazon EBS-backed AMI を作成するには、既存の Amazon EBS-backed AMI を使用してインスタンスを起動することから開始します。この AMI は、例えば、AWS Marketplace から取得した AMI、VM Import/Export を使用して作成した AMI、またはユーザーがアクセス可能なその他の任意の AMI です。特定の要件を満たすようにインスタンスをカスタマイズしたら、新しい AMI を作成して登録します。その後、新しい AMI を使用して、カスタマイズした新しいインスタンスを起動できます。

> **Note**
>
> EC2 インスタンスアテステーションをサポートする AMI を作成するには、「構成証明可能 AMI」を参照してください。

以下に説明された手順は、暗号化された Amazon Elastic Block Store (Amazon EBS) ボリューム (ルートボリュームを含む) でバックアップされた Amazon EC2 インスタンスにも、暗号化されていないボリューム同様に機能します。

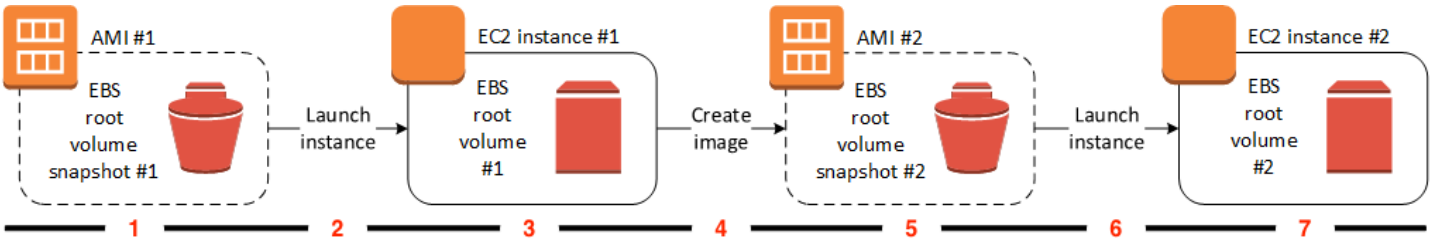
AMI の作成プロセスは、Amazon S3-backed AMI の場合とは異なります。詳細については「Amazon S3-backed AMI を作成する」を参照してください。

内容

- インスタンスからの AMI 作成の概要
- インスタンスから AMI を作成する
- スナップショットから AMI を作成する

インスタンスからの AMI 作成の概要

次の図は、実行中の EC2 インスタンスから Amazon EBS-backed AMI を作成するプロセスをまとめたものです。既存の AMI から開始して、インスタンスを起動してカスタマイズし、そこから新しい AMI を作成し、最後に新しい AMI のインスタンスを起動します。図表内の数字は、次の説明の数値と一致します。



1 — AMI #1: 既存の AMI から始める

作成する AMI に似た既存の AMI を検索します。例えば、AWS Marketplace から取得した AMI、VM Import/Export を使用して作成した AMI、またはユーザーがアクセス可能なその他の任意の AMI です。この AMI を必要に応じてカスタマイズします。

図表内の EBS ルートボリュームスナップショット #1 は、AMI が Amazon EBS-backed AMI であり、ルートボリュームに関する情報がこのスナップショットに格納されていることを示します。

2 — 既存の AMI からインスタンスを起動する

AMI を設定する方法は、新しい AMI のベースとなる AMI からインスタンスを起動し、インスタンスをカスタマイズすることです (図表内の 3)。次に、カスタマイズを含む新しい AMI を作成します (図表内の 4)。

3 — EC2 インスタンス #1: インスタンスをカスタマイズする

インスタンスに接続し、必要に応じてカスタマイズします。新しい AMI には、これらのカスタマイズが含まれます。

インスタンスで次のアクションを実行して、インスタンスをカスタマイズできます。

- ソフトウェアやアプリケーションをインストールする
- データをコピーする
- 起動時間を短縮するために一時ファイルの消去、ハードディスクのデフラグ、占有領域の開放処理を行います。
- 追加の EBS ボリュームをアタッチする

4 – イメージを作成する

インスタンスから AMI を作成する際に、Amazon EC2 がインスタンスをシャットダウンしてから AMI を作成するのは、インスタンス上のすべての動作を停止し、作成プロセス中に一貫した状態が保たれるようにするためです。インスタンスが一貫した状態にあり、適切に AMI を作成できる場合、インスタンスの電源を落として再起動しないように、Amazon EC2 に指定できます。XFS などの一部のファイルシステムでは、アクティビティのフリーズおよびフリーズ解除が可能なため、インスタンスを再起動しなくてもイメージを安全に作成できます。

AMI 作成プロセスの間、Amazon EC2 はインスタンスのルートボリュームとインスタンスにアタッチされているその他の EBS ボリュームのスナップショットを作成します。AMI の登録を解除してスナップショットを削除するまで、スナップショットは課金の対象となります。インスタンスにアタッチされるいずれかのボリュームが暗号化されている場合、新しい AMI は、Amazon EBS 暗号化をサポートするインスタンスでのみ正常に起動します。

ボリュームのサイズによっては、AMI 作成プロセスの完了に数分かかる場合があります (最長で 24 時間かかることもあります)。AMI を作成する前に、ボリュームのスナップショットを作成しておくと、効率が向上する可能性があります。この方法では、AMI を作成する際に作成する必要があるのは小さい差分スナップショットのみになるため、プロセスがよりすばやく完了します (スナップショット作成の合計時間は同じです)。

5 — AMI #2: 新しい AMI

プロセスが完了すると、新しい AMI と、インスタンスのルートボリュームから作成されたスナップショット (スナップショット #2) が得られます。ルートボリュームに加えて、インスタンスストアボリュームまたは EBS ボリュームをインスタンスに追加した場合、新しい AMI のブロックデバイスマッピングにこれらのボリュームの情報が含まれます。

Amazon EC2 では AMI は自動的に登録されます。

6 – 新しい AMI からインスタンスを起動する

新しい AMI を使用してインスタンスを起動できます。

7 — EC2 インスタンス #2: 新しいインスタンス

ユーザーが新しい AMI を使用してインスタンスを起動すると、Amazon EC2 はスナップショットを使用して、そのインスタンスのルートボリュームのために新しい EBS ボリュームを作成します。インスタンスのカスタマイズ時に、インスタンスストアボリュームまたは EBS ボリュームを追加した場合、新しい AMI のブロックデバイスマッピングにこれらのボリュームの情報が含まれ、新しい AMI から起動するインスタンスのブロックデバイスマッピングに自動的にこれらのボリュームの情報が含まれます。新しいインスタンスのブロックデバイスマッピングに指定されているインスタンスストアボリュームは新しく、AMI の作成に使用したインスタンスのインスタンスストアボリュームからのデータは含まれていません。EBS ボリュームのデータは永続的です。詳細については「Amazon EC2 インスタンスでのボリュームのブロックデバイスマッピング」を参照してください。

EBS-backed AMI から新しいインスタンスを作成する場合、本稼働環境に移す前にそのルートボリュームと追加 EBS ストレージの両方を初期化する必要があります。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームの初期化」を参照してください。

インスタンスから AMI を作成する

既存のインスタンスがある場合は、このインスタンスから AMI を作成できます。

Console

AMI を作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開きます。
2. ナビゲーションペインで、[インスタンス] を選択してください。

3. AMI を作成するインスタンスを選択し、[アクション] 、[イメージとテンプレート] の順に選択し、[イメージの作成] をクリックしてください。

> **Tip**
>
> このオプションが無効になっている場合、そのインスタンスは Amazon EBS-backed インスタンスではありません。

4. [イメージの作成] ページで、次の情報を指定します。

   a. [イメージ名] に、最大 127 文字までイメージの一意の名前を入力してください。

   b. [イメージの説明] に、最大 255 文字までイメージの説明を入力します (オプション)。

   c. [インスタンスを再起動] の場合、チェックボックスをオンのまま (デフォルト) にするか、オフにします。

   - [インスタンスを再起動] がオンの場合、Amazon EC2 が新しい AMI を作成するとき、データの保管時にアタッチされたボリュームのスナップショットを取得できるようにインスタンスを再起動して、一貫性のある状態を維持します。
   - [インスタンスを再起動] をクリアすると、Amazon EC2 が新しい AMI を作成しても、インスタンスはシャットダウンおよび再起動されません。

> **Warning**
>
> [インスタンスを再起動] をクリアした場合は、作成されたイメージのファイルシステムの整合性は保証されません。

   d. [インスタンスボリューム] - 次のとおり、ルートボリュームを変更し、Amazon EBS およびインスタンスストアボリュームを追加できます。

      i. ルートボリュームは、最初の行で定義されます。

      - ルートボリュームのサイズを変更するには、[サイズ] に必要な値を入力してください。
      - [終了時に削除] を選択した場合、この AMI から作成されたインスタンスを終了すると、EBS ボリュームが削除されます。[終了時に削除] をオフにした場合は、インスタンスを終了しても、EBS ボリュームは削除されません。詳細については「インスタンスの終了時にデータを保持する」を参照してください。

ii. EBS ボリュームを追加するには、[ボリューム追加] を選択します (これにより、新しい行が追加されます)。[ストレージタイプ] で [EBS] を選択し、行のフィールドに入力してください。作成した AMI からインスタンスを起動すると、追加したボリュームは自動的にそのインスタンスにアタッチされます。空のボリュームはフォーマットしてマウントする必要があります。スナップショットベースのボリュームはマウントする必要があります。

iii. インスタンスストアボリュームを追加するには、「Amazon EC2 AMI にインスタンスストアボリュームを追加する」を参照してください。その後新しい AMI からインスタンスを起動すると、追加されたボリュームは自動的に初期化されてマウントされます。これらのボリュームには、AMI の作成に使用された実行中のインスタンスのインスタンスストアボリュームのデータは含まれません。

e. スナップショットの送信先 – インスタンスボリュームが EBS ローカルスナップショットをサポートしているローカルゾーンにある場合は、AMI のスナップショットを作成する場所を選択します。

- AWS リージョン: ボリュームのローカルゾーンの親リージョンにスナップショットを作成します。
- AWS ローカルゾーン: ボリュームと同じローカルゾーンにスナップショットを作成します。

> **Note**
>
> このオプションは、EBS ローカルスナップショットをサポートするローカルゾーンにのみ、インスタンスがローカルゾーンで作成された場合に限って表示されます。ボリュームがリージョンにある場合、このオプションは表示されず、スナップショットはボリュームと同じリージョンに自動作成されます。詳細については、「Amazon EBS ユーザーガイド」の「Local snapshots in Local Zones」を参照してください。

> **Important**
>
> インスタンスのボリュームのスナップショットはすべて同じ場所に存在する必要があります。既存のスナップショットの場所を確認してください。選択した送信

先とは別の場所に既存のスナップショットがある場合、AMI の作成は失敗します。

f. タグ - AMI とスナップショットに同じタグを付けることも、異なるタグを付けることもできます。

- AMI とスナップショットに同じタグを付けるには、[イメージとスナップショットに一緒にタグを付ける] を選択してください。AMI と作成されるすべてのスナップショットには、同じタグが適用されます。
- AMI とスナップショットに異なるタグを付けるには、[イメージとスナップショットに個別にタグを付ける] を選択してください。AMI と作成されるスナップショットには、異なるタグが適用されます。ただし、すべてのスナップショットに同じタグが付けられます。各スナップショットに異なるタグを付けることはできません。

(オプション) タグを追加するには、[Add tag] を選択し、そのタグのキーと値を入力してください。各タグについて、これを繰り返します。

g. AMI を作成する準備ができたら、[イメージの作成] を選択してください。

5. 作成中に AMI のステータスを表示するには

a. ナビゲーションペインで [AMI] を選択してください。

b. フィルタを [自分が所有] に設定し、リストから AMI を探します。

最初は、ステータスは `pending` ですが、数分後 `available` に変わります。

6. (オプション) 新しい AMI に作成されたスナップショットを表示するには:

a. 前のステップで特定した AMI の ID をメモします。

b. ナビゲーションペインで、[スナップショット] を選択してください。

c. フィルターを [自分が所有] に設定し、新しい AMI ID のスナップショットを [説明] 列で検索します。

ユーザーがこの AMI からインスタンスを起動すると、Amazon EC2 はこのスナップショットを使用して、インスタンスのルートボリュームを作成します。

AWS CLI

AMI を作成するには

create-image コマンドを使用します。

```
aws ec2 create-image \
    --instance-id i-1234567890abcdef0 \
    --name "my-web-server" \
    --description "My web server image" \
    --no-reboot
```

PowerShell

AMI を作成するには

New-EC2Image コマンドレットを使用します。

```
New-EC2Image `
    -InstanceId i-1234567890abcdef0 `
    -Name "my-web-server" `
    -Description "My web server image" `
    -NoReboot $true
```

スナップショットから AMI を作成する

インスタンスのルートボリュームのスナップショットがある場合、このスナップショットから AMI を作成できます。

> **Note**
>
> ほとんどの場合、Windows、RedHat、SUSE、SQL Server の AMI には、正しいライセンス情報が存在する必要があります。詳細については「AMI の請求情報について」を参照してください。スナップショットから AMI を作成する場合、RegisterImage オペレーションはスナップショットのメタデータから正しい請求情報を取得しますが、これには適切なメタデータが必要です。正しい請求情報が適用されたかどうかを確認するには、新しい AMI の [プラットフォームの詳細] フィールドを確認します。フィールドが空であるか、所定のオペレーティングシステムコード (Windows、RedHat、SUSE、SQL など) と一致しない場合、AMI の作成は失敗しているため、この AMI を破棄して「インスタンスから AMI を作成する」の手順に従う必要があります。

Console

スナップショットから AMI を作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開きます。
2. ナビゲーションペインで、[スナップショット] を選択してください。
3. AMI の作成元になるスナップショットを選択してから、[アクション]、[スナップショットからイメージ作成] の順に選択してください。
4. [スナップショットからイメージを作成] ページで、次の情報を指定します。

   a. [イメージ名] に、イメージのわかりやすい名前を入力してください。

   b. [説明] に、イメージの簡単な説明を入力してください。

   c. [アーキテクチャ] で、イメージアーキテクチャを選択してください。32 ビットの場合は [i386]、64 ビットの場合は [x86_64]、64 ビット ARM の場合は [arm64]、64 ビット macOS の場合は [x86_64] をそれぞれ選択してください。

   d. [ルートデバイス名] に、ルートボリュームに使用するデバイス名を入力してください。詳細については「Amazon EC2 インスタンス上のボリュームのデバイス名」を参照してください。

   e. [仮想化タイプ] で、この AMI から起動されたインスタンスで使用する仮想化タイプを選択してください。詳細については「仮想化タイプ」を参照してください。

   f. (準仮想化の場合のみ) [カーネル ID] で、イメージのオペレーティングシステムのカーネルを選択してください。インスタンスのルートボリュームのスナップショットを使用している場合、元のインスタンスと同じカーネル ID を選択してください。不明な場合は、デフォルトのカーネルを使用してください。

   g. (準仮想仮想化の場合のみ) [RAM ディスク ID] で、イメージの RAM ディスクを選択してください。カーネルを選択した場合は、サポートするドライバーとともに特定の RAM ディスクを選択しなければならない可能性があります。

   h. [ブートモード] では、イメージのブートモードを選択するか [デフォルトを使用] を選択し、この AMI でインスタンスを起動したときにインスタンスタイプでサポートされているブートモードで起動するようにします。詳細については「Amazon EC2 AMI のブートモードを設定する」を参照してください。

   i. (オプション) [ブロックデバイスマッピング] で、ルートボリュームをカスタマイズし、データボリュームを追加します。

      ボリュームごとに、サイズ、タイプ、パフォーマンス特性、終了時の削除の動作、および暗号化ステータスを指定できます。ルートボリュームについては、サイズをスナップ

ショットのサイズより小さくすることはできません。ボリュームタイプには、汎用 SSD gp3 がデフォルトで選択されています。

j. (オプション) [タグ] で、新しい AMI に 1 つ以上のタグを追加できます。(オプション) タグを追加するには、[タグ追加] を選択し、そのタグのキーと値を入力してください。各タグについて、これを繰り返します。

k. AMI を作成する準備ができたら、[イメージの作成] を選択してください。

5. (Windows、RedHat、SUSE、SQL Server のみ) 正しい請求情報が適用されたかどうかを確認するには、新しい AMI の [プラットフォームの詳細] フィールドを確認します。フィールドが空であるか、所定のオペレーティングシステムコード (Windows や RedHat など) と一致しない場合、AMI の作成は失敗しているため、この AMI を破棄して「インスタンスから AMI を作成する」の手順に従う必要があります。

AWS CLI

AWS CLI を使用してスナップショットから AMI を作成するには

register-image コマンドを使用します。

```
aws ec2 register-image \
    --name my-image \
    --root-device-name /dev/xvda \
    --block-device-mappings DeviceName=/dev/
xvda,Ebs={SnapshotId=snap-0db2cf683925d191f}
```

PowerShell

PowerShell を使用してスナップショットから AMI を作成するには

Register-EC2Image コマンドレットを使用します。

```
$block = @{SnapshotId=snap-0db2cf683925d191f}
Register-EC2Image `
    -Name my-image `
    -RootDeviceName /dev/xvda `
    -BlockDeviceMapping @{DeviceName="/dev/xvda";Ebs=$block}
```

Amazon S3-backed AMI を作成する

インスタンスの起動時に指定する AMI によってルートボリュームのタイプが決まります。

Amazon S3-backed Linux AMI を作成するには、既存の Amazon S3-backed Linux AMI から起動したインスタンスから始めます。ニーズに合わせてインスタンスをカスタマイズしたら、ボリュームをバンドルし、新しい AMI を登録します。新しい AMI を使用して、カスタマイズした新しいインスタンスを起動できます。

Windows AMI はルートボリュームのインスタンスストアをサポートしていないため、Amazon S3-backed Windows AMI を作成することはできません。

> **Important**
>
> インスタンスストアボリュームをルートボリュームとしてサポートし、Amazon S3-backed AMI を必要とするインスタンスタイプは C1、C3、D2、I2、M1、M2、M3、R3、X1 のみです。

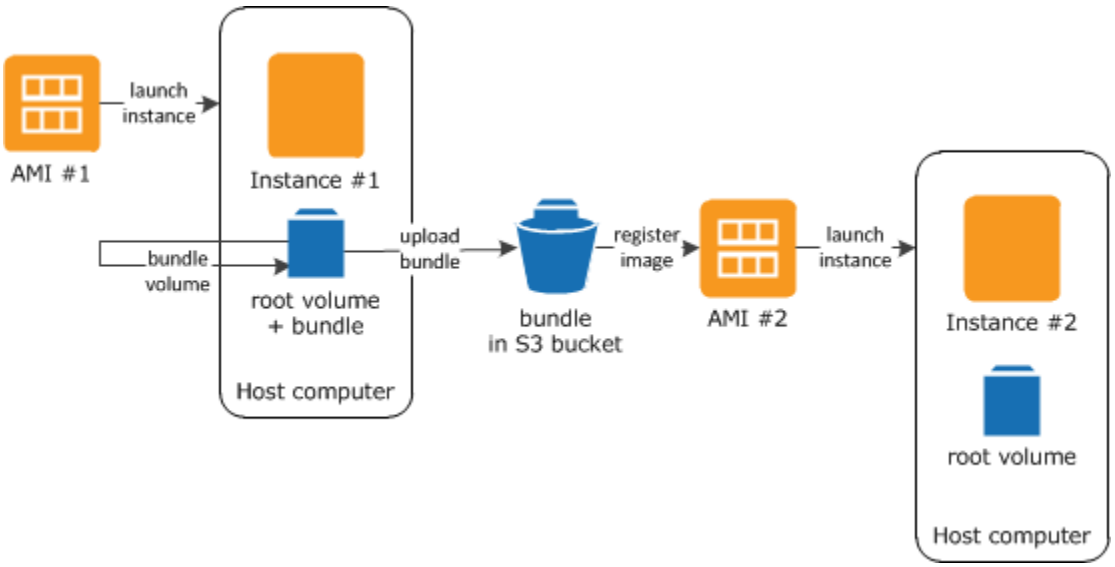
AMI の作成プロセスは、Amazon EBS-backed AMI の場合とは異なります。詳細については、「Amazon EBS-backed AMI を作成する」を参照してください。

内容

- AMI 作成の概要
- 前提条件
- Amazon Linux インスタンスから AMI を作成する
- Amazon EC2 AMI ツールのセットアップ
- Amazon EC2 AMI ツールリファレンス
- Amazon S3-backed AMI を EBS-backed AMI に変換する

AMI 作成の概要

次の図は、インスタンスストアのルートボリュームを持つインスタンスから AMI を作成するプロセスをまとめたものです。



最初に、作成する AMI と同様の AMI からインスタンスを起動します。インスタンスに接続し、それをカスタマイズできます。インスタンスのカスタマイズが終わったら、それをバンドルできます。バンドルプロセスが完了するには数分間かかります。プロセスが完了すると、イメージマニフェスト (`image.manifest.xml`) とルートボリューム用のテンプレートを含むファイル (`image.part.xx`) で構成されるバンドルが作成されます。次に、バンドルを Amazon S3 バケットにアップロードし、AMI を登録します。

> **Note**
>
> Amazon S3-backed Linux AMI の S3 バケットにオブジェクトをアップロードするには、バケットで ACL を有効にする必要があります。有効にしない場合、Amazon EC2 はアップロードするオブジェクトに ACL を設定できません。宛先のバケットが S3 オブジェクトの所有権のバケット所有者強制設定を使用している場合、ACL が無効になるため、この方法は使えません。詳細については、「バケットのオブジェクト所有権のコントロールと ACL の無効化」を参照してください。

お客様が新しい AMI を使用してインスタンスを起動すると、Amazon はユーザーが Amazon S3 にアップロードしたバンドルを使用してインスタンスのルートボリュームを作成します。Amazon S3 のバンドルで使用されるストレージ領域については、お客様がその領域を削除するまでアカウントに料金が発生します。詳細については、「Amazon EC2 AMI の登録を解除する」を参照してください。

ルートボリュームに加えて、インスタンスストアボリュームをインスタンスに追加した場合、新しい AMI のブロックデバイスマッピングにこれらのボリュームの情報が含まれ、新しい AMI から起動す

るインスタンスのブロックデバイスマッピングに自動的にこれらのボリュームの情報が含まれます。詳細については、「Amazon EC2 インスタンスでのボリュームのブロックデバイスマッピング」を参照してください。

前提条件

AMI を作成するには、最初に次のタスクを完了する必要があります。

- AMI ツールをインストールします。詳細については、Amazon EC2 AMI ツールのセットアップ を参照してください。
- AWS CLI をインストールします。詳細については、「AWS CLIを使い始める」を参照してください。
- バンドルに S3 バケットがあり、バケットに ACL が有効になっていることを確認します。ACL の設定の詳細については、「ACL の設定」を参照してください。
  - AWS マネジメントコンソール を使用して S3 バケットを作成するには、https://console.aws.amazon.com/s3/ で Amazon S3 コンソールを開き、[バケツ作成] を選択してください。
  - AWS CLI で S3 バケットを作成するには、「mb」コマンドを使用できます。インストールしている AMI ツールのバージョンが 1.5.18 以降の場合は、`ec2-upload-bundle` コマンドを使用して S3 バケットを作成することもできます。詳細については、「ec2-upload-bundle」を参照してください。
- バンドル内のファイルが S3 バケットで暗号化されていないことを確認します。AMI に暗号化が必要な場合は、代わりに EBS-backed AMI を使用できます。詳細については、「EBS-backed AMI での暗号化の利用」を参照してください。
- AWS アカウント ID があることを確認します。詳細については、「AWS アカウント管理リファレンスガイド」の「AWS アカウント 識別表示」を参照してください。
- AWS CLI を使用するのに必要な認証情報があることを確認します。詳細については、AWS Command Line Interface ユーザーガイドの「AWS CLI に対する認証とアクセスコントロール」を参照してください。
- X.509 証明書および対応するプライベートキーがあることを確認します。
  - X.509 証明書を作成する必要がある場合は、「署名証明書の管理」を参照してください。X.509 証明書とプライベートキーは、AMI の暗号化/復号に使用されます。
  - [中国 (北京)] `$EC2_AMITOOL_HOME/etc/ec2/amitools/cert-ec2-cn-north-1.pem` 証明書を使用します。

- [AWS GovCloud (米国 – 西部)] `$EC2_AMITOOL_HOME/etc/ec2/amitools/cert-ec2-gov.pem` 証明書を使用します。
- インスタンスに接続し、カスタマイズします。例えば、ソフトウェアとアプリケーションをインストールしたり、データをコピーしたり、一時ファイルを削除したり、Linux 設定を変更したりできます。

Amazon Linux インスタンスから AMI を作成する

次の手順では、Amazon Linux 1 を実行するインスタンスストアのルートボリュームを持つインスタンスから AMI を作成する方法について説明します。他の Linux ディストリビューションを実行するインスタンスでは機能しない可能性があります。

AMI ツールの使用準備を整えるには (HVM インスタンスのみ)

1. AMI ツールでは、GRUB のレガシーが正常に起動する必要があります。次のコマンドを使用して GRUB をインストールします。

```
[ec2-user ~]$ sudo yum install -y grub
```

2. 次のコマンドを使用して、パーティション管理パッケージをインストールします。

```
[ec2-user ~]$ sudo yum install -y gdisk kpartx parted
```

インスタンスストアのルートボリュームを持つ Amazon Linux インスタンスから AMI を作成するには

この手順では、「前提条件」に記載された前提条件が満たされていることを前提としています。

次のコマンドでは、*###############*をユーザー自身の情報で置き換えます。

1. インスタンスに認証情報をアップロードします。Amazon ではこれらの認証情報を使用して、お客様と Amazon EC2 だけがお客様の AMI にアクセスできるようにします。

   a. 次のように、認証情報のための一時ディレクトリをインスタンスに作成します。

```
[ec2-user ~]$ mkdir /tmp/cert
```

   それにより、作成したイメージから認証情報を除外できます。

b. scp などの安全なコピーツールを使用して、コンピュータからインスタンスの /tmp/cert ディレクトリに X.509 証明書と対応するプライベートキーをコピーします。次の -i *my-private-key*.pem コマンドの scp オプションは、X.509 プライベートキーではなく、SSH でインスタンスに接続するために使用するプライベートキーです。次に例を示します。

```
you@your_computer:~ $ scp -i my-private-key.pem /
path/to/pk-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem /
path/to/cert-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem ec2-
user@ec2-203-0-113-25.compute-1.amazonaws.com:/tmp/cert/
pk-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem  100%  717     0.7KB/s   00:00
cert-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem  100%  685     0.7KB/s   00:00
```

または、これらがプレーンテキストファイルの場合、証明書とキーをテキストエディタで開き、コンテンツを /tmp/cert の新しいファイルにコピーできます。

2. インスタンス内から ec2-bundle-vol コマンドを実行して、Amazon S3 にアップロードするバンドルを準備します。-e オプションを指定して、認証情報を保存するディレクトリを除外します。デフォルトでは、バンドルプロセスで機密情報を含んでいる可能性があるファイルを除外します。ファイルには、*.sw、*.swo、*.swp、*.pem、*.priv、*id_rsa*、*id_dsa*、*.gpg、*.jks、*/.ssh/authorized_keys、*/.bash_history などがあります。これらのファイルをすべて含めるには、--no-filter オプションを使用します。これらのファイルの一部を含めるには、--include オプションを使用します。

> **Important**
>
> AMI バンドルプロセスは、デフォルトで、ルートボリュームを表す /tmp ディレクトリに、圧縮され暗号化された一連のファイルを作成します。バンドルを格納するのに十分な空きディスク領域が /tmp にない場合、-d */path/to/bundle/storage* オプションを使用して、バンドルを格納する別の場所を指定する必要があります。インスタンスによっては、エフェメラルストレージが /mnt または /media/ephemeral0 にマウントされて使用可能になっている場合があります。または、バンドルを格納する新しい Amazon EBS ボリュームを作成、アタッチ、およびマウントすることもできます。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EKS ボリュームの作成」を参照してください。

a. ec2-bundle-vol コマンドは、root として実行する必要があります。ほとんどのコマンドで、sudo を使用することでアクセス許可を昇格させることができますが、この場合は、環境変数を維持するために sudo -E su を実行する必要があります。

```
[ec2-user ~]$ sudo -E su
```

これで、bash プロンプトにより root ユーザーとして識別されるようになったことと、root シェルにいることを示すハッシュタグにドル記号が置き換えられたことに注意してください。

```
[root ec2-user]#
```

b. AMI のバンドルを作成するには、次のように ec2-bundle-vol コマンドを実行します。

```
[root ec2-user]# ec2-bundle-vol -k /tmp/cert/pk-
HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem -c /tmp/cert/cert-
HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem -u 123456789012 -r x86_64 -e /tmp/cert --
partition gpt
```

> **Note**
>
> 中国 (北京) および AWS GovCloud (米国 – 西部) リージョンについては、--ec2cert パラメータを使用し、前提条件に従って証明書を指定します。

イメージの作成には数分かかります。このコマンドが完了したら、/tmp (またはデフォルト以外の) ディレクトリにバンドルが含まれます (image.manifest.xml、および複数の image.part.*xx* ファイル)。

c. root シェルを終了します。

```
[root ec2-user]# exit
```

3. (オプション) インスタンスストアをさらに追加するには、AMI 用の image.manifest.xml ファイルで、ブロックデバイスマッピングを編集します。詳細については、Amazon EC2 インスタンスでのボリュームのブロックデバイスマッピング を参照してください。

a. image.manifest.xml ファイルのバックアップを作成します。

```
[ec2-user ~]$ sudo cp /tmp/image.manifest.xml /tmp/image.manifest.xml.bak
```

b. image.manifest.xml ファイルの形式を変更し、読み取りと編集が簡単になるようにします。

```
[ec2-user ~]$ sudo xmllint --format /tmp/image.manifest.xml.bak > /tmp/
image.manifest.xml
```

c. テキストエディタで image.manifest.xml のブロックデバイスマッピングを編集します。次の例は、ephemeral1 インスタンスストアボリュームの新しいエントリを示しています。

> **Note**
>
> 無効な種類のファイルの一覧については、「ec2-bundle-vol」を参照してください。

```
<block_device_mapping>
  <mapping>
    <virtual>ami</virtual>
    <device>sda</device>
  </mapping>
  <mapping>
    <virtual>ephemeral0</virtual>
    <device>sdb</device>
  </mapping>
  <mapping>
    <virtual>ephemeral1</virtual>
    <device>sdc</device>
  </mapping>
  <mapping>
    <virtual>root</virtual>
    <device>/dev/sda1</device>
  </mapping>
</block_device_mapping>
```

d. image.manifest.xml ファイルを保存し、テキストエディタを終了します。

4. バンドルを Amazon S3 にアップロードするには、次のように ec2-upload-bundle コマンドを実行します。

```
[ec2-user ~]$ ec2-upload-bundle -b amzn-s3-demo-bucket/bundle_folder/bundle_name -
m /tmp/image.manifest.xml -a your_access_key_id -s your_secret_access_key
```

> **Important**
>
> US East (N. Virginia) 以外のリージョンで AMI を登録するには、--region オプションと、すでにターゲットリージョンに存在するバケットパス、またはターゲットリージョンで作成できる一意のバケットパスの両方でターゲットリージョンを指定する必要があります。

5. (オプション) バンドルを Amazon S3 にアップロードしたら、次の /tmp コマンドを使用して、インスタンスの rm ディレクトリからバンドルを削除できます。

```
[ec2-user ~]$ sudo rm /tmp/image.manifest.xml /tmp/image.part.* /tmp/image
```

> **Important**
>
> -d */path/to/bundle/storage* で Step 2 オプションを使用してパスを指定した場合は、/tmp ではなくそのパスを使用します。

6. AMI を登録するには、次のように register-image コマンドを実行します。

```
[ec2-user ~]$ aws ec2 register-image --image-location amzn-s3-demo-
bucket/bundle_folder/bundle_name/image.manifest.xml --name AMI_name --
virtualization-type hvm
```

> **Important**
>
> ec2-upload-bundle コマンドでリージョンを以前に指定した場合は、このコマンドでもう一度そのリージョンを指定します。

Amazon EC2 AMI ツールのセットアップ

AMI ツールを使用して、Amazon S3-backed Linux AMI を作成および管理できます。ツールを使用するには、Linux インスタンスにインストールする必要があります。AMI ツールは RPM として使用で

きるとともに、RPM をサポートしていない Linux ディストリビューションでは .zip ファイルとして使用できます。

RPM を使用して AMI ツールを設定するには

1. yum などの Linux ディストリビューション用のパッケージマネージャを使用して Ruby をインストールします。次に例を示します。

```
[ec2-user ~]$ sudo yum install -y ruby
```

2. wget や curl などのツールを使用して RPM ファイルをダウンロードします。次に例を示します。

```
[ec2-user ~]$ wget https://s3.amazonaws.com/ec2-downloads/ec2-ami-tools.noarch.rpm
```

3. 次のコマンドを使用して RPM ファイルの署名を確認する:

```
[ec2-user ~]$ rpm -K ec2-ami-tools.noarch.rpm
```

上のコマンドは、ファイルの SHA1 および MD5 ハッシュが OK. であることを示しています。ハッシュが NOT OK であることをコマンドが示している場合、次のコマンドを使用してファイルのヘッダー SHA1 および MD5 ハッシュを表示します。

```
[ec2-user ~]$ rpm -Kv ec2-ami-tools.noarch.rpm
```

次に、ファイルのヘッダー SHA1 および MD5 ハッシュを、以下の検証済み AMI ツールハッシュと比較し、ファイルの正統性を確認します。

- ヘッダー SHA1: a1f662d6f25f69871104e6a62187fa4df508f880
- MD5: 9faff05258064e2f7909b66142de6782

ファイルのヘッダー SHA1 および MD5 ハッシュが検証済み AMI ツールハッシュと一致する場合、次のステップに進みます。

4. 次のコマンドを使用して RPM をインストールします。

```
[ec2-user ~]$ sudo yum install ec2-ami-tools.noarch.rpm
```

5. ec2-ami-tools-version コマンドを使用してインストールした AMI ツールを検証します。

```
[ec2-user ~]$ ec2-ami-tools-version
```

> **Note**
>
> [cannot load such file -- ec2/amitools/version (LoadError)] などのロードエラーを受信した場合は、次のステップを実行し、AMI ツールをインストールした場所を RUBYLIB パスに追加します。

6. (オプション) 前のステップでエラーが発生した場合、AMI ツールをインストールした場所を RUBYLIB パスに追加します。

   a. 追加するパスを調べるには、次のコマンドを実行します。

   ```
   [ec2-user ~]$ rpm -qil ec2-ami-tools | grep ec2/amitools/version
   /usr/lib/ruby/site_ruby/ec2/amitools/version.rb
   /usr/lib64/ruby/site_ruby/ec2/amitools/version.rb
   ```

   上記の例では、以前のロードエラーから失われたファイルは /usr/lib/ruby/site_ruby および /usr/lib64/ruby/site_ruby にあります。

   b. 前のステップの場所を RUBYLIB パスに追加します。

   ```
   [ec2-user ~]$ export RUBYLIB=$RUBYLIB:/usr/lib/ruby/site_ruby:/usr/lib64/ruby/site_ruby
   ```

   c. ec2-ami-tools-version コマンドを使用してインストールした AMI ツールを検証します。

   ```
   [ec2-user ~]$ ec2-ami-tools-version
   ```

zip ファイルを使用して AMI ツールを設定するには

1. Ruby をインストールし、apt-get など、Linux ディストリビューション用のパッケージマネージャを使用して解凍します。次に例を示します。

   ```
   [ec2-user ~]$ sudo apt-get update -y && sudo apt-get install -y ruby unzip
   ```

2. wget や curl などのツールを使用して .zip ファイルをダウンロードします。次に例を示します。

```
[ec2-user ~]$ wget https://s3.amazonaws.com/ec2-downloads/ec2-ami-tools.zip
```

3. /usr/local/ec2 など、適切なインストールディレクトリにファイルを解凍します。

```
[ec2-user ~]$ sudo mkdir -p /usr/local/ec2
$ sudo unzip ec2-ami-tools.zip -d /usr/local/ec2
```

.zip ファイルには、フォルダ (ec2-ami-tools-*x.x.x*) が含まれます。ここで、*x.x.x* はツールのバージョン番号 (例: ec2-ami-tools-1.5.7) です。

4. EC2_AMITOOL_HOME 環境変数を、ツールのインストールディレクトリに設定します。次に例を示します。

```
[ec2-user ~]$ export EC2_AMITOOL_HOME=/usr/local/ec2/ec2-ami-tools-x.x.x
```

5. ツールを PATH 環境変数に追加します。次に例を示します。

```
[ec2-user ~]$ export PATH=$EC2_AMITOOL_HOME/bin:$PATH
```

6. ec2-ami-tools-version コマンドを使用してインストールした AMI ツールを検証できます。

```
[ec2-user ~]$ ec2-ami-tools-version
```

署名証明書の管理

AMI ツールの特定のコマンドでは、デジタル署名用証明書 (X.509 証明書とも呼ばれる) が必要です。証明書を作成し、AWS にアップロードする必要があります。例えば、証明書の作成に OpenSSL などのサードパーティ製のツールを使用できます。

デジタル署名用証明書を作成するには

1. OpenSSL をインストールおよび設定します。
2. プライベートキーを openssl genrsa コマンドを使用して作成し、出力を .pem ファイルで保存します。2048 ビットまたは 4096 ビット RSA キーの作成を推奨しています。

```
openssl genrsa 2048 > private-key.pem
```

3. openssl req コマンドを使用して、証明書を作成します。

```
openssl req -new -x509 -nodes -sha256 -days 365 -key private-key.pem -outform PEM -
out certificate.pem
```

証明書を AWS にアップロードするには、upload-signing-certificate コマンドを使用します。

```
aws iam upload-signing-certificate --user-name user-name --certificate-body
 file://path/to/certificate.pem
```

ユーザーの証明書を一覧表示するには、list-signing-certificates コマンドを使用します。

```
aws iam list-signing-certificates --user-name user-name
```

ユーザーのデジタル署名用証明書を無効化または再有効化するには、update-signing-certificate コマンドを使用します。次のコマンドは証明書を無効にします。

```
aws iam update-signing-certificate --certificate-id OFHPLP4ZULTHYPMSYEX7O4BEXAMPLE --
status Inactive --user-name user-name
```

証明書を削除するには、delete-signing-certificate コマンドを使用します。

```
aws iam delete-signing-certificate --user-name user-name --certificate-
id OFHPLP4ZULTHYPMSYEX7O4BEXAMPLE
```

Amazon EC2 AMI ツールリファレンス

AMI ツールコマンドを使用して、Amazon S3-backed Linux AMI を作成および管理できます。ツールをセットアップする方法は、「Amazon EC2 AMI ツールのセットアップ」を参照してください。

アクセスキーについては、「IAM ユーザーガイド」の「Managing access keys for IAM users」を参照してください。

コマンド

- ec2-ami-tools-version
- ec2-bundle-image
- ec2-bundle-vol
- ec2-delete-bundle

- ec2-download-bundle
- ec2-migrate-manifest
- ec2-unbundle
- ec2-upload-bundle
- AMI ツール用の一般的なオプション

ec2-ami-tools-version

説明

AMI ツールのバージョンについて説明します。

構文

ec2-ami-tools-version

Output

バージョン情報。

例

このコマンド例では、使用中の AMI ツールのバージョン情報を表示します。

```
[ec2-user ~]$ ec2-ami-tools-version
1.5.2 20071010
```

ec2-bundle-image

説明

ループバックファイル内に作成されるオペレーティングシステムイメージから Amazon S3-backed Linux AMI を作成します。

構文

ec2-bundle-image -c *path* -k *path* -u *account* -i *path* [-d *path*] [--ec2cert *path*] [-r *architecture*] [--productcodes *code1*,*code2*,...] [-B *mapping*] [-p *prefix*]

オプション

`-c, --cert` パス

ユーザーの PEM エンコード RSA パブリックキー証明書ファイル。

必須: はい

`-k, --privatekey` パス

PEM エンコードされる RSA キーファイルへのパス。このバンドルをバンドル解除するには、このキーを指定する必要があるため、安全な場所に保管してください。このキーは AWS アカウントに登録されている必要はありません。

必須: はい

`-u, --user` アカウント

ダッシュのない、ユーザーの AWS アカウント ID。

必須: はい

`-i, --image` パス

バンドルするイメージへのパス。

必須: はい

`-d, --destination` パス

バンドルを作成するディレクトリ。

デフォルト: `/tmp`

必須: いいえ

`--ec2cert` パス

イメージマニフェストの暗号化に使用される Amazon EC2 X.509 パブリックキー証明書へのパス。

`us-gov-west-1` および `cn-north-1` リージョンではデフォルト以外のパブリックキー証明書を使用し、その証明書へのパスは、このオプションで指定する必要があります。証明書へのパスは、AMI ツールのインストール方法によって異なります。Amazon Linux の場合、証明書の場所は `/opt/aws/amitools/ec2/etc/ec2/amitools/` です。「Amazon EC2 AMI ツールのセッ

トアップ」の RPM または ZIP ファイルから AMI ツールをインストールした場合、証明書の場所は $EC2_AMITOOL_HOME/etc/ec2/amitools/ です。

必須: us-gov-west-1 および cn-north-1 リージョンのみ。

-r, --arch アーキテクチャ

イメージアーキテクチャ。コマンドラインでアーキテクチャを指定しない場合、バンドルの開始時に入力を求められます。

有効な値: i386 | x86_64

必須: いいえ

--productcodes code1、code2、...

登録時にイメージにアタッチする、カンマ区切りの製品コード。

必須: いいえ

-B, --block-device-mapping マッピング

インスタンスタイプが指定されたデバイスをサポートする場合に、この AMI のインスタンスにブロックデバイスを公開する方法を定義します。

キーと値のペアのカンマ区切りのペアを指定します。名キーは仮想名であり、各値は対応するデバイス名です。仮想名には以下が含まれています。

- ami — インスタンスによって判断されるルートファイルシステムデバイス
- root — カーネルによって判断されるルートファイルシステムデバイス
- swap — インスタンスによって判断されるスワップデバイス
- ephemeralN — N 番目のインスタンスストアボリューム

必須: いいえ

-p, --prefixprefix

バンドル済み AMI ファイルのファイル名プレフィクス。

デフォルト: イメージファイルの名前。例えば、イメージパスが /var/spool/my-image/version-2/debian.img である場合、デフォルトのプレフィクスは debian.img です。

必須: いいえ

`--kernel` kernel_id

廃止。カーネルを設定するには、register-image を使用します。

必須: いいえ

`--ramdisk` ramdisk_id

廃止。必要に応じて、register-image を使用して RAM ディスクを設定します。

必須: いいえ

Output

バンドルプロセスのステージとステータスを記述するステータスメッセージ。

例

この例は、ループバックファイルで作成されたオペレーティングシステムイメージから、バンドルされた AMI を作成します。

```
[ec2-user ~]$ ec2-bundle-image -k pk-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem -c cert-
HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem -u 111122223333 -i image.img -d bundled/ -r x86_64
Please specify a value for arch [i386]:
Bundling image file...
Splitting bundled/image.gz.crypt...
Created image.part.00
Created image.part.01
Created image.part.02
Created image.part.03
Created image.part.04
Created image.part.05
Created image.part.06
Created image.part.07
Created image.part.08
Created image.part.09
Created image.part.10
Created image.part.11
Created image.part.12
Created image.part.13
Created image.part.14
Generating digests for each part...
Digests generated.
Creating bundle manifest...
```

```
ec2-bundle-image complete.
```

ec2-bundle-vol

説明

インスタンスのルートボリュームのコピーを圧縮、暗号化、署名することで、Amazon S3-backed Linux AMI を作成します。

Amazon EC2 はインスタンスから製品コード、カーネル設定、RAM ディスク設定、およびブロックデバイスマッピングを継承しようとします。

デフォルトでは、バンドルプロセスで機密情報を含んでいる可能性があるファイルを除外します。ファイルには、*.sw、*.swo、*.swp、*.pem、*.priv、*id_rsa*、*id_dsa*、*.gpg、*.jks、*/.ssh/authorized_keys、*/.bash_history などがあります。これらのファイルをすべて含めるには、--no-filter オプションを使用します。これらのファイルの一部を含めるには、--include オプションを使用します。

詳細については、Amazon S3-backed AMI を作成する を参照してください。

構文

ec2-bundle-vol -c *path* -k *path* -u *account* [-d *path*] [--ec2cert *path*] [-r *architecture*] [--productcodes *code1*,*code2*,...] [-B *mapping*] [--all] [-e *directory1*,*directory2*,...] [-i *file1*,*file2*,...] [--no-filter] [-p *prefix*] [-s *size*] [--[no-]inherit] [-v *volume*] [-P *type*] [-S *script*] [--fstab *path*] [--generate-fstab] [--grub-config *path*]

オプション

-c, --cert パス

ユーザーの PEM エンコード RSA パブリックキー証明書ファイル。

必須: はい

-k, --privatekey パス

ユーザーの PEM エンコード RSA キーファイルへのパス。

必須: はい

`-u, --user` アカウント

ダッシュのない、ユーザーの AWS アカウント ID。

必須: はい

`-d, --destination` 送信先

バンドルを作成するディレクトリ。

デフォルト: `/tmp`

必須: いいえ

`--ec2cert` パス

イメージマニフェストの暗号化に使用される Amazon EC2 X.509 パブリックキー証明書へのパス。

`us-gov-west-1` および `cn-north-1` リージョンではデフォルト以外のパブリックキー証明書を使用し、その証明書へのパスは、このオプションで指定する必要があります。証明書へのパスは、AMI ツールのインストール方法によって異なります。Amazon Linux の場合、証明書の場所は `/opt/aws/amitools/ec2/etc/ec2/amitools/` です。「Amazon EC2 AMI ツールのセットアップ」の RPM または ZIP ファイルから AMI ツールをインストールした場合、証明書の場所は `$EC2_AMITOOL_HOME/etc/ec2/amitools/` です。

必須: `us-gov-west-1` および `cn-north-1` リージョンのみ。

`-r, --arch` アーキテクチャ

イメージアーキテクチャ。コマンドラインでこれを指定しない場合、バンドルの開始時に入力を求められます。

有効な値: `i386` | `x86_64`

必須: いいえ

`--productcodes` code1、code2、...

登録時にイメージにアタッチする、カンマ区切りの製品コード。

必須: いいえ

`-B, --block-device-mapping` マッピング

インスタンスタイプが指定されたデバイスをサポートする場合に、この AMI のインスタンスにブロックデバイスを公開する方法を定義します。

キーと値のペアのカンマ区切りのペアを指定します。名キーは仮想名であり、各値は対応するデバイス名です。仮想名には以下が含まれています。

- `ami` — インスタンスによって判断されるルートファイルシステムデバイス
- `root` — カーネルによって判断されるルートファイルシステムデバイス
- `swap` — インスタンスによって判断されるスワップデバイス
- `ephemeralN` — N 番目のインスタンスストアボリューム

必須: いいえ

`-a, --all`

リモートでマウントされたファイルシステムのディレクトリを含めて、すべてのディレクトリをバンドルします。

必須: いいえ

`-e, --exclude` directory1、directory2、…

バンドルオペレーションから除外する絶対ディレクトリパスとファイルのリスト。このパラメータは `--all` オプションを上書きします。除外を指定すると、パラメータとともにリストされたディレクトリとサブディレクトリは、ボリュームにバンドルされません。

必須: いいえ

`-i, --include` file1、file2、…

バンドルオペレーションに含めるファイルのリスト。指定されたファイルは、それ以外の場合は AMI から除外されます。これは、機密情報が含まれる可能性があるためです。

必須: いいえ

`--no-filter`

指定した場合、AMI からファイルは除外されません。これは、機密情報が含まれる可能性があるためです。

必須: いいえ

`-p, --prefix` prefix

バンドル済み AMI ファイルのファイル名プレフィクス。

デフォルト: `image`

必須: いいえ

`-s, --size` サイズ

作成するイメージファイルの MB (1024 * 1024 バイト) 単位のサイズ。最大サイズは 10240 MB です。

デフォルト: 10240

必須: いいえ

`--[no-]inherit`

イメージがインスタンスのメタデータを継承するかどうかを示します (デフォルトでは継承します)。`--inherit` を有効にし、インスタンスメタデータにアクセスできない場合、バンドルは失敗します。

必須: いいえ

`-v, --volume` ボリューム

バンドルを作成する、マウントされたボリュームへの絶対パス。

デフォルト: ルートディレクトリ (/)。

必須: いいえ

`-P, --partition` type

ディスクイメージでパーティションテーブルを使用するかどうかを示します。パーティションテーブルタイプを指定しない場合、デフォルトでは、該当する場合はボリュームの親ブロックデバイスで使用されるタイプになります。それ以外の場合、デフォルトは `gpt` です。

有効な値: `mbr` | `gpt` | `none`

必須: いいえ

`-S, --script` スクリプト

バンドルの直前に実行するカスタマイズスクリプト。スクリプトでは単一の引数である、ボリュームのマウントポイントが予期されます。

必須: いいえ

`--fstab` パス

イメージにバンドルする fstab へのパス。これを指定しない場合、Amazon EC2 は /etc/fstab をバンドルします。

必須: いいえ

`--generate-fstab`

Amazon EC2 で提供される fstab を使用してボリュームをバンドルします。

必須: いいえ

`--grub-config`

イメージにバンドルする別の grub 設定ファイルへのパス。デフォルトでは、`ec2-bundle-vol` は `/boot/grub/menu.lst` または `/boot/grub/grub.conf` が、クローンされたイメージ上に存在することを想定します。このオプションにより、別の grub 設定ファイルへのパスを指定することができ、このファイルはデフォルトに上書きしてコピーされます (存在する場合)。

必須: いいえ

`--kernel` kernel_id

廃止。カーネルを設定するには、register-image を使用します。

必須: いいえ

`--ramdiskramdisk_id`

廃止。必要に応じて、register-image を使用して RAM ディスクを設定します。

必須: いいえ

Output

バンドルのステージとステータスを説明するステータスメッセージ。

例

この例では、ローカルマシンのルートファイルシステムのスナップショットを圧縮、暗号化、署名することで、バンドルされた AMI を作成します。

```
[ec2-user ~]$ ec2-bundle-vol -d /mnt -k pk-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem -c
 cert-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem -u 111122223333 -r x86_64
  Copying / into the image file /mnt/image...
```

```
  Excluding:
       sys
       dev/shm
       proc
       dev/pts
       proc/sys/fs/binfmt_misc
       dev
       media
       mnt
       proc
       sys
       tmp/image
       mnt/img-mnt
  1+0 records in
  1+0 records out
  mke2fs 1.38 (30-Jun-2005)
  warning: 256 blocks unused.

  Splitting /mnt/image.gz.crypt...
  Created image.part.00
  Created image.part.01
  Created image.part.02
  Created image.part.03
  ...
  Created image.part.22
  Created image.part.23
  Generating digests for each part...
  Digests generated.
  Creating bundle manifest...
  Bundle Volume complete.
```

ec2-delete-bundle

説明

Amazon S3 ストレージから、指定されたバンドルを削除します。バンドルを削除した後で、対応する AMI からインスタンスを起動することはできません。

構文

ec2-delete-bundle -b *bucket* -a *access_key_id* -s *secret_access_key* [-t *token*] [--url *url*] [--region *region*] [--sigv *version*] [-m *path*] [-p *prefix*] [--clear] [--retry] [-y]

オプション

`-b, --bucket` bucket

バンドルされた AMI に続いてオプションの '/' 区切りパスプレフィクスを含む Amazon S3 バケットの名前

必須: はい

`-a, --access-key` access_key_id

AWS アクセスキー ID。

必須: はい

`-s, --secret-key` secret_access_key

AWS シークレットアクセスキー。

必須: はい

`-t, --delegation-token` トークン

AWS リクエストに渡す委任トークン。詳細については、IAM ユーザーガイドの「IAM の一時的なセキュリティ認証情報」を参照してください。

必須: 一時的なセキュリティ認証情報を使用している場合のみ。

デフォルト: `AWS_DELEGATION_TOKEN` 環境変数の値 (設定されている場合)。

`--region`リージョン

リクエスト署名で使用するリージョン。

デフォルト: `us-east-1`

必須: 署名バージョン 4 を使用する場合は必須

`--sigvversion`

リクエストに署名するときに使用する署名バージョン。

有効な値: 2 | 4

デフォルト: 4

必須: いいえ

`-m, --manifest`パス

マニフェストファイルへのパス。

必須: `--prefix` または `--manifest` のどちらかを指定する必要があります。

`-p, --prefix`prefix

バンドルされた AMI ファイル名プレフィクス。プレフィクス全体を指定します。例えば、プレフィクスが image.img である場合は、`-p image.img` ではなく `-p image` を使用します。

必須: `--prefix` または `--manifest` のどちらかを指定する必要があります。

`--clear`

指定されたバンドルを削除した後で空の場合は、Amazon S3 バケットを削除します。

必須: いいえ

`--retry`

すべての Amazon S3 エラーで、オペレーションあたり最大 5 回まで自動的に再試行します。

必須: いいえ

`-y, --yes`

すべてのプロンプトへの答えが [yes] であると自動的に想定します。

必須: いいえ

Output

Amazon EC2 は、削除プロセスのステージとステータスを示すステータスメッセージを表示します。

例

この例では、Amazon S3 からバンドルを削除します。

```
[ec2-user ~]$ ec2-delete-bundle -b amzn-s3-demo-bucket -a your_access_key_id -
s your_secret_access_key
Deleting files:
amzn-s3-demo-bucket/image.manifest.xml
amzn-s3-demo-bucket/image.part.00
```

```
amzn-s3-demo-bucket/image.part.01
amzn-s3-demo-bucket/image.part.02
amzn-s3-demo-bucket/image.part.03
amzn-s3-demo-bucket/image.part.04
amzn-s3-demo-bucket/image.part.05
amzn-s3-demo-bucket/image.part.06
Continue? [y/n]
y
Deleted amzn-s3-demo-bucket/image.manifest.xml
Deleted amzn-s3-demo-bucket/image.part.00
Deleted amzn-s3-demo-bucket/image.part.01
Deleted amzn-s3-demo-bucket/image.part.02
Deleted amzn-s3-demo-bucket/image.part.03
Deleted amzn-s3-demo-bucket/image.part.04
Deleted amzn-s3-demo-bucket/image.part.05
Deleted amzn-s3-demo-bucket/image.part.06
ec2-delete-bundle complete.
```

ec2-download-bundle

説明

指定された Amazon S3-backed Linux AMI を Amazon S3 ストレージからダウンロードします。

構文

ec2-download-bundle -b *bucket* -a *access_key_id* -s *secret_access_key* -k *path* [--url *url*] [--region *region*] [--sigv *version*] [-m *file*] [-p *prefix*] [-d *directory*] [--retry]

オプション

`-b, --bucket` バケット

バンドルが存在する Amazon S3 バケットの名前。この後に、オプションで '/' 区切りのパスプレフィクスが続きます。

必須: はい

`-a, --access-key` access_key_id

AWS アクセスキー ID。

必須: はい

`-s, --secret-key` secret_access_key

AWS シークレットアクセスキー。

必須: はい

`-k, --privatekey` パス

マニフェストの復号に使用されるプライベートキー。

必須: はい

`--url` url

Amazon S3 サービスの URL。

デフォルト: `https://s3.amazonaws.com/`

必須: いいえ

`--region` region

リクエスト署名で使用するリージョン。

デフォルト: `us-east-1`

必須: 署名バージョン 4 を使用する場合は必須

`--sigv` バージョン

リクエストに署名するときに使用する署名バージョン。

有効な値: 2 | 4

デフォルト: 4

必須: いいえ

`-m, --manifest` ファイル

マニフェストファイル名 (パスなし)。マニフェスト `-m`またはプレフィクス `-p`を指定することをお勧めします。

必須: いいえ

`-p, --prefix` prefix

バンドル済み AMI ファイルのファイル名プレフィクス。

デフォルト: image

必須: いいえ

`-d, --directory` ディレクトリ

ダウンロードしたバンドルが保存されているディレクトリ。ディレクトリが存在している必要があります。

デフォルト: 現在の作業ディレクトリ。

必須: いいえ

`--retry`

すべての Amazon S3 エラーで、オペレーションあたり最大 5 回まで自動的に再試行します。

必須: いいえ

Output

ダウンロードプロセスの多様な段階ステータスを示すメッセージが表示されます。

例

この例では、bundled ディレクトリを作成 (Linux mkdir コマンドを使用) し、Amazon S3 amzn-s3-demo-bucket バケットからバンドルをダウンロードします。

```
[ec2-user ~]$ mkdir bundled
[ec2-user ~]$ ec2-download-bundle -b amzn-s3-demo-bucket/bundles/bundle_name
 -m image.manifest.xml -a your_access_key_id -s your_secret_access_key -k pk-
HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem -d mybundle
Downloading manifest image.manifest.xml from amzn-s3-demo-bucket to mybundle/
image.manifest.xml ...
Downloading part image.part.00 from amzn-s3-demo-bucket/bundles/bundle_name to
 mybundle/image.part.00 ...
Downloaded image.part.00 from amzn-s3-demo-bucket
Downloading part image.part.01 from amzn-s3-demo-bucket/bundles/bundle_name to
 mybundle/image.part.01 ...
Downloaded image.part.01 from amzn-s3-demo-bucket
Downloading part image.part.02 from amzn-s3-demo-bucket/bundles/bundle_name to
 mybundle/image.part.02 ...
Downloaded image.part.02 from amzn-s3-demo-bucket
```

```
Downloading part image.part.03 from amzn-s3-demo-bucket/bundles/bundle_name to
 mybundle/image.part.03 ...
Downloaded image.part.03 from amzn-s3-demo-bucket
Downloading part image.part.04 from amzn-s3-demo-bucket/bundles/bundle_name to
 mybundle/image.part.04 ...
Downloaded image.part.04 from amzn-s3-demo-bucket
Downloading part image.part.05 from amzn-s3-demo-bucket/bundles/bundle_name to
 mybundle/image.part.05 ...
Downloaded image.part.05 from amzn-s3-demo-bucket
Downloading part image.part.06 from amzn-s3-demo-bucket/bundles/bundle_name to
 mybundle/image.part.06 ...
Downloaded image.part.06 from amzn-s3-demo-bucket
```

ec2-migrate-manifest

説明

別のリージョンをサポートするように Amazon S3-backed Linux AMI (証明書、カーネル、RAM ディスクなど) を変更します。

構文

ec2-migrate-manifest -c *path* -k *path* -m *path* {(-a *access_key_id* -s *secret_access_key* --region *region*) | (--no-mapping)} [--ec2cert *ec2_cert_path*] [--kernel *kernel-id*] [--ramdisk *ramdisk_id*]

オプション

`-c, --cert` パス

ユーザーの PEM エンコード RSA パブリックキー証明書ファイル。

必須: はい

`-k, --privatekey` パス

ユーザーの PEM エンコード RSA キーファイルへのパス。

必須: はい

`--manifest` パス

マニフェストファイルへのパス。

必須: はい

`-a, --access-key` access_key_id

AWS アクセスキー ID。

必須: 自動マッピングを使用する場合は必須です。

`-s, --secret-key` secret_access_key

AWS シークレットアクセスキー。

必須: 自動マッピングを使用する場合は必須です。

`--region` region

マッピングファイル内で検索するリージョン。

必須: 自動マッピングを使用する場合は必須です。

`--no-mapping`

カーネルと RAM ディスクの自動マッピングを無効にします。

移行中、Amazon EC2 は、コピー先リージョン用に設計されたカーネルと RAM ディスクで、マニフェストファイルのカーネルと RAM ディスクを置き換えます。`--no-mapping` パラメータを指定しない場合、`ec2-migrate-bundle` は `DescribeRegions` および `DescribeImages` オペレーションを使用して、自動化されたマッピングを実行します。

必須: 自動マッピングに使用される `-a`、`-s`、および `--region` オプションを指定しない場合は必須です。

`--ec2cert` パス

イメージマニフェストの暗号化に使用される Amazon EC2 X.509 パブリックキー証明書へのパス。

`us-gov-west-1` および `cn-north-1` リージョンではデフォルト以外のパブリックキー証明書を使用し、その証明書へのパスは、このオプションで指定する必要があります。証明書へのパスは、AMI ツールのインストール方法によって異なります。Amazon Linux の場合、証明書の場所は `/opt/aws/amitools/ec2/etc/ec2/amitools/` です。「Amazon EC2 AMI ツールのセットアップ」の ZIP ファイルから AMI ツールをインストールした場合、証明書の場所は `$EC2_AMITOOL_HOME/etc/ec2/amitools/` です。

必須: `us-gov-west-1` および `cn-north-1` リージョンのみ。

`--kernel` kernel_id

選択するカーネルの ID。

> **Important**
>
> カーネルと RAM ディスクではなく PV-GRUB を使用することをお勧めします。詳細については、「Amazon Linux 2 ユーザーガイド」の「ユーザー提供カーネル」を参照してください。

必須: いいえ

`--ramdisk` ramdisk_id

選択する RAM ディスクの ID。

> **Important**
>
> カーネルと RAM ディスクではなく PV-GRUB を使用することをお勧めします。詳細については、「Amazon Linux 2 ユーザーガイド」の「ユーザー提供カーネル」を参照してください。

必須: いいえ

Output

バンドルプロセスのステージとステータスを記述するステータスメッセージ。

例

この例では、`my-ami.manifest.xml` マニフェストで指定された AMI を米国から EU にコピーします。

```
[ec2-user ~]$ ec2-migrate-manifest --manifest my-ami.manifest.xml
 --cert cert-HKZYKTAIG2ECMXYIBH3HXV4ZBZQ55CLO.pem --privatekey pk-
HKZYKTAIG2ECMXYIBH3HXV4ZBZQ55CLO.pem --region eu-west-1

Backing up manifest...
Successfully migrated my-ami.manifest.xml It is now suitable for use in eu-west-1.
```

ec2-unbundle

説明

Amazon S3-backed Linux AMI からバンドルを再作成します。

構文

ec2-unbundle -k *path* -m *path* [-s *source_directory*] [-d *destination_directory*]

オプション

`-k, --privatekey` パス

PEM エンコードされる RSA キーファイルへのパス。

必須: はい

`-m, --manifest` パス

マニフェストファイルへのパス。

必須: はい

`-s, --source` source_directory

バンドル含むディレクトリ。

デフォルト: 現在のディレクトリ。

必須: いいえ

`-d, --destination` destination_directory

AMI をバンドル解除するディレクトリ。宛先ディレクトリが存在している必要があります。

デフォルト: 現在のディレクトリ。

必須: いいえ

例

この Linux および UNIX の例では、`image.manifest.xml` ファイルに指定された AMI をバンドル解除します。

```
[ec2-user ~]$ mkdir unbundled
$ ec2-unbundle -m mybundle/image.manifest.xml -k pk-
HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem -s mybundle -d unbundled
$ ls -l unbundled
total 1025008
-rw-r--r-- 1 root root 1048578048 Aug 25 23:46 image.img
```

Output

バンドル解除プロセスの多様な段階ステータスを示すメッセージが表示されます。

ec2-upload-bundle

説明

Amazon S3-backed Linux AMI のバンドルを Amazon S3 にアップロードし、アップロードされたオブジェクトで適切なアクセスコントロールリスト (ACL) を設定します。詳細については、「Amazon S3-backed AMI を作成する」を参照してください。

> **Note**
>
> Amazon S3-backed Linux AMI の S3 バケットにオブジェクトをアップロードするには、バケットで ACL を有効にする必要があります。有効にしない場合、Amazon EC2 はアップロードするオブジェクトに ACL を設定できません。宛先のバケットが S3 オブジェクトの所有権のバケット所有者強制設定を使用している場合、ACL が無効になるため、この方法は使えません。詳細については、「バケットのオブジェクト所有権のコントロールと ACL の無効化」を参照してください。

構文

```
ec2-upload-bundle -b bucket -a access_key_id -s secret_access_key [-t
token] -m path [--url url] [--region region] [--sigv version] [--acl acl]
[-d directory] [--part part] [--retry] [--skipmanifest]
```

オプション

`-b, --bucket` バケット

バンドルを保存する Amazon S3 バケットの名前。その後にオプションで '/' 区切りのパスプレフィクスが続きます。バケットが存在しない場合、バケット名を使用できる場合はバケットが作

成されます。さらに、バケットが存在せず、AMI ツールのバージョンが 1.5.18 以降の場合、このコマンドはバケットの ACL を設定します。

必須: はい

`-a, --access-key` access_key_id

AWS アクセスキー ID。

必須: はい

`-s, --secret-key` secret_access_key

お客様の AWS シークレットアクセスキー。

必須: はい

`-t, --delegation-token` トークン

AWS リクエストに渡す委任トークン。詳細については、IAM ユーザーガイドの「IAM の一時的なセキュリティ認証情報」を参照してください。

必須: 一時的なセキュリティ認証情報を使用している場合のみ。

デフォルト: `AWS_DELEGATION_TOKEN` 環境変数の値 (設定されている場合)。

`-m, --manifest` パス

マニフェストファイルへのパス。マニフェストファイルはバンドルプロセス中に作成され、バンドルを含むディレクトリにあります。

必須: はい

`--url` url

廃止。バケットの場所が (`--region` ではなく) EU に制約されない限り、代わりに `eu-west-1` オプションを使用します。`--location` フラグは、その特定の場所の制限を対象にする唯一の方法です。

Amazon S3 エンドポイントサービスの URL。

デフォルト: `https://s3.amazonaws.com/`

必須: いいえ

`--region` region

宛先の S3 バケットに対してリクエスト署名で使用するリージョン。

- バケットが存在せず、リージョンを指定しない場合、ツールは (us-east-1 で) 場所の制約のないバケットを作成します。
- バケットが存在せず、リージョンを指定した場合、ツールは指定したリージョンでバケットを作成します。
- バケットが存在し、リージョンを指定しない場合、ツールはバケットの場所を使用します。
- バケットが存在し、リージョンとして us-east-1 を指定した場合、ツールはエラーメッセージなしでバケットの実際の場所を使用し、一致する既存のファイルは上書きされます。
- バケットが存在し、バケットの実際の場所に一致しない (us-east-1 以外の) リージョンを指定した場合、ツールはエラーで終了します。

バケットが (EU ではなく) eu-west-1 の場所に制約されている場合は、代わりに --location フラグを使用します。--location フラグは、その特定の場所の制限を対象にする唯一の方法です。

デフォルト: us-east-1

必須: 署名バージョン 4 を使用する場合は必須

--sigv バージョン

リクエストに署名するときに使用する署名バージョン。

有効な値: 2 | 4

デフォルト: 4

必須: いいえ

--acl acl

バンドルされたイメージのアクセスコントロールリストのポリシー。

有効な値: public-read | aws-exec-read

デフォルト: aws-exec-read

必須: いいえ

-d, --directory ディレクトリ

バンドルされた AMI 部分を含むディレクトリ。

デフォルト: マニフェストファイルを含むディレクトリ (-m オプションを参照)。

必須: いいえ

--part パート

指定された部分とそれ以降のすべての部分のアップロードを開始します。例えば、--part 04。

必須: いいえ

--retry

すべての Amazon S3 エラーで、オペレーションあたり最大 5 回まで自動的に再試行します。

必須: いいえ

--skipmanifest

マニフェストをアップロードしません。

必須: いいえ

--location の場所

廃止。バケットの場所が (--region ではなく) EU に制約されない限り、代わりに eu-west-1 オプションを使用します。--location フラグは、その特定の場所の制限を対象にする唯一の方法です。

宛先 Amazon S3 バケットの場所の制約。バケットが存在し、バケットの実際の場所に一致しない場所を指定する場合、ツールはエラーで終了します。バケットが存在し、場所を指定しない場合、ツールはバケットの場所を使用します。バケットが存在しない場合に場所を指定すると、ツールは、指定した場所でバケットを作成します。バケットが存在せず、場所を指定しない場合、ツールは (us-east-1 で) 場所の制約のないバケットを作成します。

デフォルト: --region を指定した場合、場所はその指定したリージョンに設定されます。--region を指定しない場合、場所はデフォルトで us-east-1 になります。

必須: いいえ

Output

Amazon EC2 は、アップロードプロセスのステージとステータスを示すステータスメッセージを表示します。

例

この例では、image.manifest.xml マニフェストで指定されたバンドルをアップロードします。

```
[ec2-user ~]$ ec2-upload-bundle -b amzn-s3-demo-bucket/bundles/bundle_name -m
 image.manifest.xml -a your_access_key_id -s your_secret_access_key
Creating bucket...
Uploading bundled image parts to the S3 bucket amzn-s3-demo-bucket ...
Uploaded image.part.00
Uploaded image.part.01
Uploaded image.part.02
Uploaded image.part.03
Uploaded image.part.04
Uploaded image.part.05
Uploaded image.part.06
Uploaded image.part.07
Uploaded image.part.08
Uploaded image.part.09
Uploaded image.part.10
Uploaded image.part.11
Uploaded image.part.12
Uploaded image.part.13
Uploaded image.part.14
Uploading manifest ...
Uploaded manifest.
Bundle upload completed.
```

AMI ツール用の一般的なオプション

AMI ツールのほとんどで、以下の任意のパラメータを使用できます。

`--help, -h`

ヘルプメッセージを表示します。

`--version`

バージョンと著作権表記を表示します。

`--manual`

手動のエントリを表示します。

`--batch`

インタラクティブなプロンプトを制約するバッチモードで実行します。

`--debug`

問題のトラブルシューティング時に役立つ可能性がある情報を表示します。

Amazon S3-backed AMI を EBS-backed AMI に変換する

所有している Amazon S3-backed Linux AMI は、Amazon EBS-Backed Linux AMI に変換できます。

> **Important**
> 所有していない AMI を変換することはできません。

Amazon S3-backed AMI を Amazon EBS-backed AMI に変換するには

1. Amazon EBS-backed AMI から Amazon Linux インスタンスを起動します。詳細については、コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する を参照してください。Amazon Linux インスタンスには、AWS CLI および AMI ツールがプリインストールされています。
2. Amazon S3-backed AMI をバンドルするのに使用した X.509 プライベートキーをインスタンスにアップロードします。Amazon はこのキーを使用して、お客様と Amazon EC2 だけがお客様の AMI にアクセスできるようにします。

   a. 次のように、X.509 プライベートキーのインスタンスに一時ディレクトリを作成します。

   ```
   [ec2-user ~]$ mkdir /tmp/cert
   ```

   b. scp などの安全なコピーツールを使用して、コンピュータから /tmp/cert ディレクトリに X.509 プライベートキーをコピーします。次のコマンドの *my-private-key* パラメータは、SSH でインスタンスに接続するために使用するプライベートキーです。例:

   ```
   you@your_computer:~ $ scp -i my-private-key.pem /
   path/to/pk-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem ec2-
   user@ec2-203-0-113-25.compute-1.amazonaws.com:/tmp/cert/
   pk-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem  100%  717     0.7KB/s   00:00
   ```

3. 環境変数を設定して、AWS CLI を使用します。詳細については、「環境変数」を参照してください。

   a. (推奨) AWS アクセスキー、シークレットキーおよびセッショントークンの環境変数を設定します。

   ```
   [ec2-user ~]$ export AWS_ACCESS_KEY_ID=your_access_key_id
   [ec2-user ~]$ export AWS_SECRET_ACCESS_KEY=your_secret_access_key
   ```

```
[ec2-user ~]$ export AWS_SESSION_TOKEN=your_session_token
```

b. AWS アクセスキーおよびシークレットキーの環境変数を設定します。

```
[ec2-user ~]$ export AWS_ACCESS_KEY_ID=your_access_key_id
[ec2-user ~]$ export AWS_SECRET_ACCESS_KEY=your_secret_access_key
```

4. 新しい AMI の Amazon エラスティックブロックストア (Amazon EBS) ボリュームを準備します。

a. create-volume コマンドを使用して、インスタンスと同じアベイラビリティーゾーンに空の EBS ボリュームを作成します。コマンド出力のボリューム ID を書き留めてください。

> **Important**
>
> この EBS ボリュームは、元のインスタンスストアのルートボリュームと同じサイズ以上である必要があります。

```
aws ec2 create-volume \
    --size 10 \
    --region us-west-2 \
    --availability-zone us-west-2b
```

b. attach-volume コマンドを使用して、Amazon EBS-backed インスタンスにボリュームをアタッチします。

```
aws ec2 attach-volume \
    --volume-id vol-01234567890abcdef \
    --instance-id i-1234567890abcdef0 \
    --region us-west-2
```

5. バンドルのフォルダを作成します。

```
[ec2-user ~]$ mkdir /tmp/bundle
```

6. /tmp/bundle コマンドを使用して、インスタンスストアバック AMI のバンドルを ec2-download-bundle にダウンロードします。

```
[ec2-user ~]$ ec2-download-bundle -b amzn-s3-demo-bucket/bundle_folder/bundle_name
 -m image.manifest.xml -a $AWS_ACCESS_KEY_ID -s $AWS_SECRET_ACCESS_KEY --
privatekey /path/to/pk-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem -d /tmp/bundle
```

7. ec2-unbundle コマンドを使用して、バンドルからイメージファイルを再作成します。

   a. バンドルフォルダにディレクトリを変更します。

   ```
   [ec2-user ~]$ cd /tmp/bundle/
   ```

   b. ec2-unbundle コマンドを実行します。

   ```
   [ec2-user bundle]$ ec2-unbundle -m image.manifest.xml --privatekey /path/to/pk-
   HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem
   ```

8. バンドルを解除したイメージから新しい EBS ボリュームにファイルをコピーします。

   ```
   [ec2-user bundle]$ sudo dd if=/tmp/bundle/image of=/dev/sdb bs=1M
   ```

9. バンドルを解除した新しいパーティションのボリュームを調査します。

   ```
   [ec2-user bundle]$ sudo partprobe /dev/sdb1
   ```

10. ブロックデバイスの一覧を表示してマウントするデバイス名を選択してください。

    ```
    [ec2-user bundle]$ lsblk
    NAME         MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
    /dev/sda    202:0    0   8G  0 disk
    ##/dev/sda1 202:1    0   8G  0 part /
    /dev/sdb    202:80   0  10G  0 disk
    ##/dev/sdb1 202:81   0  10G  0 part
    ```

    この例では、マウントするパーティションは /dev/sdb1 ですが、デバイス名はおそらく異なります。ボリュームが仕切られていない場合は、マウントするデバイスは /dev/sdb に似ています (デバイスパーティションの末尾に数値なし)。

11. 新しい EBS ボリュームのマウントポイントを作成し、ボリュームをマウントします。

    ```
    [ec2-user bundle]$ sudo mkdir /mnt/ebs
    [ec2-user bundle]$ sudo mount /dev/sdb1 /mnt/ebs
    ```

12. EBS ボリュームの /etc/fstab ファイルを任意のテキストエディタ (vim や nano など) で開き、インスタンスストア (エフェメラル) ボリュームのエントリがあれば削除します。EBS ボリュームが /mnt/ebs に取付けられるため、fstab ファイルは /mnt/ebs/etc/fstab にあります。

```
[ec2-user bundle]$ sudo nano /mnt/ebs/etc/fstab
#
LABEL=/     /           ext4    defaults,noatime  1   1
tmpfs       /dev/shm    tmpfs   defaults        0   0
devpts      /dev/pts    devpts  gid=5,mode=620  0   0
sysfs       /sys        sysfs   defaults        0   0
proc        /proc       proc    defaults        0   0
/dev/sdb        /media/ephemeral0       auto    defaults,comment=cloudconfig    0
     2
```

この例では、最後の行を削除する必要があります。

13. ボリュームをアンマウントし、インスタンスからデタッチします。

```
[ec2-user bundle]$ sudo umount /mnt/ebs
[ec2-user bundle]$ aws ec2 detach-volume --volume-id vol-01234567890abcdef --
region us-west-2
```

14. 次のように、新しい EBS ボリュームから AMI を作成します。

    a. 新しい EBS ボリュームのスナップショットを作成します。

    ```
    [ec2-user bundle]$ aws ec2 create-snapshot --region us-west-2 --description
     "your_snapshot_description" --volume-id vol-01234567890abcdef
    ```

    b. スナップショットが完了していることを確認します。

    ```
    [ec2-user bundle]$ aws ec2 describe-snapshots --region us-west-2 --snapshot-
    id snap-0abcdef1234567890
    ```

    c. 元の AMI で使用されたプロセッサアーキテクチャ、仮想化タイプ、カーネルイメージ (aki) を、describe-images コマンドを使用して特定します。このステップでは、元の Amazon S3-backed AMI の AMI ID が必要です。

    ```
    [ec2-user bundle]$ aws ec2 describe-images --region us-west-2 --image-
    id ami-0abcdef1234567890 --output text
    ```

```
IMAGES x86_64 amazon/amzn-ami-pv-2013.09.2.x86_64-s3 ami-8ef297be amazon
 available public machine aki-fc8f11cc instance-store paravirtual xen
```

この例では、アーキテクチャは x86_64 で、カーネルイメージ ID は aki-fc8f11cc です。次のステップでこれらの値を使用します。前述のコマンドの出力では ari ID もリストされますので、これも書き留めます。

d. 新しい EBS ボリュームのスナップショット ID と前のステップで書き留めた値を使用して、新しい AMI を登録します。前述のコマンド出力に ari ID がリストされていた場合は、その ID を次のコマンドで --ramdisk-id *ari_id*. を使用して指定します。

```
[ec2-user bundle]$ aws ec2 register-image --region us-west-2 --
name your_new_ami_name --block-device-mappings DeviceName=device-
name,Ebs={SnapshotId=snap-0abcdef1234567890} --virtualization-type paravirtual
 --architecture x86_64 --kernel-id aki-fc8f11cc --root-device-name device-name
```

15. (オプション) 新しい AMI からインスタンスを起動できることをテストした後で、この手順で作成した EBS ボリュームを削除できます。

```
aws ec2 delete-volume --volume-id vol-01234567890abcdef
```

Windows Sysprep を使用して Amazon EC2 AMI を作成する

Microsoft システム準備 (Windows Sysprep) ツールは、新しいイメージをキャプチャする前にインスタンス固有のシステム設定を削除して、オペレーティングシステムの一般化バージョンを作成します。

EC2 Image Builder を使用して、ソフトウェアと設定が事前にインストール、定義、カスタマイズされたセキュアな最新の「ゴールデン」サーバーイメージの作成、管理、デプロイを自動化することをお勧めします。

Windows Sysprep で、Windows 起動エージェント (EC2Launch v2、EC2Launch、EC2Config) を使って標準化された AMI を作成することもできます。

> **Important**
>
> Windows Sysprep を使用してインスタンスのバックアップを作成しないでください。Windows Sysprep はシステム固有の情報を削除します。この情報を削除すると、インスタンスのバックアップに予期しない結果が生じる場合があります。

Windows Sysprep のトラブルシューティングについては、「Amazon EC2 Windows インスタンスの Sysprep の問題をトラブルシューティングする」を参照してください。

内容

- Windows Sysprep のステップ
- [開始する前に]
- EC2Launch v2 で Windows Sysprep を使って AMI を作成する
- EC2Launch で Windows Sysprep を使って AMI を作成する
- EC2Config で Windows Sysprep を使って AMI を作成する

Windows Sysprep のステップ

Windows Sysprep は、次のステップを通じて実行されます。

- 一般化: Sysprep ツールはイメージに固有の情報と設定を削除します。Windows Sysprep は、例えばセキュリティ識別子 (SID)、コンピュータ名、イベントログおよび特定のドライバーなどを削除します。このステップを完了すると、オペレーティングシステム (OS) は AMI を作成する準備が整いました。

  > **Note**
  >
  > Windows 起動エージェントで Windows Sysprep を実行すると、`PersistAllDeviceInstalls` はデフォルトで true に設定されているため、システムによってドライバーの削除が禁止されます。

- 特定化: プラグアンドプレイはコンピュータをスキャンして、検出されたデバイス用のドライバをインストールします。Sysprep ツールは、コンピュータ名や SID など OS に固有の要件を生成します。必要に応じて、このフェーズでコマンドを実行できます。
- アウトオブボックスエクスペリエンス (OOBE): システムによって Windows セットアップの省略バージョンが実行され、システム言語、タイムゾーン、登録組織などの情報を入力するように求められます。Windows 起動エージェントで Windows Sysprep を実行すると、応答ファイルによってこのステップが自動化されます。

[開始する前に]

- Windows Sysprep を実行する前に、Windows Sysprep を実行する単一の管理者アカウント以外のすべてのローカルユーザーアカウントとすべてのアカウントプロファイルを削除することをお勧めします。追加のアカウントとプロファイルを使用して Windows Sysprep を実行すると、プロファイルデータの損失や Windows Sysprep の完了の失敗など、予期しない動作が発生する可能性があります。
- 詳細については、「Sysprep Overview」を参照してください。
- Sysprep でサポートされているサーバーロールについては、「Sysprep Support for Server Roles」を参照してください。

EC2Launch v2 で Windows Sysprep を使って AMI を作成する

EC2Launch v2 エージェントがインストールされたインスタンスからイメージを作成すると、EC2Launch v2 はイメージが作成されたときに特定のタスクを実行します。これには Windows Sysprep を使用した作業が含まれます。詳細については「the section called “Windows Sysprep のステップ”」を参照してください。

内容

- Windows Sysprep のアクション
- Sysprep 後
- EC2Launch v2 で Windows Sysprep を実行する

Windows Sysprep のアクション

イメージを準備するために、Windows Sysprep と EC2Launch v2 は以下のアクションを実行します。

1. [EC2Launch の設定] ダイアログボックスで [Sysprep によるシャットダウン] を選択すると、システムは `ec2launch sysprep` コマンドを実行します。
2. EC2Launch v2 は、`unattend.xml` のレジストリ値を読み取ることで、`HKEY_USERS\.DEFAULT\Control Panel\International\LocaleName` ファイルの内容を編集します。このファイルは `C:\ProgramData\Amazon\EC2Launch\sysprep` ディレクトリにあります。
3. システムは `BeforeSysprep.cmd` を実行します。このコマンドは、次のレジストリキーを作成します。

reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t REG_DWORD /d 1 /f

レジストリキーは再度有効になるまで RDP 接続を無効にします。RDP 接続を無効にすることは、安全上の観点より必要となります。これは、Windows Sysprep 実行後の最初のブートセッション中、RDP が接続できるわずかな時間の間に管理者パスワードが空白となるためです。

4. EC2Launch v2 サービスは、以下のコマンドを実行して Windows Sysprep を呼び出します。

sysprep.exe /oobe /generalize /shutdown /unattend: "C:\ProgramData\Amazon\EC2Launch\sysprep\unattend.xml"

ステップの一般化

- EC2Launch v2 は、コンピュータ名や SID などイメージに固有の情報と設定を削除します。インスタンスがドメインのメンバーである場合は、そのドメインから削除されます。`unattend.xml` 応答ファイルには、このステップに影響する以下の設定が含まれています。
  - PersistAllDeviceInstalls: この設定は、Windows セットアップがデバイスを削除したり再設定することを防ぐことによってイメージ準備プロセスを高速化します。これは、Amazon AMI を実行するためには特定のドライバーが必要となり、これらのドライバーの再検出には時間がかかるためです。
  - DoNotCleanUpNonPresentDevices: この設定では、現在存在しないデバイス用のプラグアンドプレイ情報を保持します。
- Windows Sysprep は AMI の作成の準備完了後に OS をシャットダウンします。システムは、新しいインスタンスを起動するか、または元のインスタンスを起動します。

ステップの特定化

システムは、コンピュータ名や SID など OS に固有の要件を生成します。またシステムは、`unattend.xml` 応答ファイルで指定した設定に基づいて、以下のアクションを実行します。

- CopyProfile: Windows Sysprep は、組み込まれた管理者のプロファイルを含むすべてのユーザープロファイルを削除するように設定できます。この設定は、組み込まれた管理者アカウントを保持するため、アカウントで作成したすべてのカスタム設定は新しいイメージに引き継がれます。デフォルト値は `True` です。

CopyProfile は、デフォルトのプロファイルを既存のローカル管理者プロファイルに置き換えます。Windows Sysprep の実行後にログインしたすべてのアカウントは、最初のログイン時にそのプロファイルとその内容のコピーを受け取ります。

新しいイメージに引き継ぐ特定のユーザープロファイルがない場合、この設定を `False` に変更します。Windows Sysprep はすべてのユーザープロファイルを削除します (これにより時間とディスク領域が節約されます)。

- タイムゾーン: タイムゾーンはデフォルトで世界時 (UTC) に設定されます。
- 順序 1 の同期コマンド: システムは次のコマンドを実行して、管理者アカウントを有効化し、パスワード条件を指定します。

```
net user Administrator /ACTIVE:YES /LOGONPASSWORDCHG:NO /EXPIRES:NEVER /
PASSWORDREQ:YES
```

- 順序 2 の同期コマンド: システムは、管理者パスワードを組み換えます。このセキュリティ対策の目的は、`setAdminAccount` のタスクを設定していない場合、Windows Sysprep の完了後にインスタンスへのアクセスを防ぐことです。

  システムは、ローカルの起動エージェントディレクトリ (`C:\Program Files\Amazon \EC2Launch\`) から次のコマンドを実行します。

```
EC2Launch.exe internal randomize-password --username Administrator
```

- リモートデスクトップ接続を有効にするため、システムはターミナルサーバーの `fDenyTSConnections` レジストリキーを false に設定します。

OOBE のステップ

1. システムは EC2Launch v2 応答ファイルを使用して以下の設定を指定します。
   - `<InputLocale>en-US</InputLocale>`
   - `<SystemLocale>en-US</SystemLocale>`
   - `<UILanguage>en-US</UILanguage>`
   - `<UserLocale>en-US</UserLocale>`
   - `<HideEULAPage>true</HideEULAPage>`
   - `<HideWirelessSetupInOOBE>true</HideWirelessSetupInOOBE>`
   - `<ProtectYourPC>3</ProtectYourPC>`

- <BluetoothTaskbarIconEnabled>false</BluetoothTaskbarIconEnabled>
- <TimeZone>UTC</TimeZone>
- <RegisteredOrganization>Amazon.com</RegisteredOrganization>
- <RegisteredOwner>EC2</RegisteredOwner>

> **Note**
>
> 一般化ステップと特殊化ステップの実行中、EC2Launch v2 によって OS のステータスがモニタリングされます。OS が Sysprep のステップにあることが EC2Launch v2 によって検出された場合、以下のメッセージがシステムログに発行されます。
> Windows is being configured. SysprepState=IMAGE_STATE_UNDEPLOYABLE

2. システムは EC2Launch v2 を実行します。

Sysprep 後

Windows Sysprep が完了すると、EC2Launch v2 によって以下のメッセージがコンソール出力に送信されます。

```
Windows sysprep configuration complete.
```

次に、EC2Launch v2 は以下のアクションを実行します。

1. agent-config.yml ファイルの内容を読み取り、設定されたタスクを実行します。
2. preReady ステージのすべてのタスクを実行します。
3. 完了したら、Windows is ready メッセージをインスタンスのシステムログに送信します。
4. PostReady ステージのすべてのタスクを実行します。

EC2Launch v2 の詳細については、「EC2 Windows インスタンスの起動時に EC2Launch v2 エージェントを使用してタスクを実行する」を参照してください。

EC2Launch v2 で Windows Sysprep を実行する

以下の手順に従って、EC2Launch v2 で Windows Sysprep を使用して標準化 AMI を作成します。

1. Amazon EC2 コンソールで、複製する AMI の場所を特定します。

2. Windows インスタンスを起動して接続します。
3. カスタム設定

   a. Windows の [スタート] メニューから、[Amazon EC2Launch 設定] を検索して選択します。[Amazon EC2Launch 設定] ダイアログボックスのオプションと設定の詳細については、「EC2Launch v2 設定を Windows インスタンスに構成する」を参照してください。

   b. 変更を加えた場合、シャットダウンする前に [保存] を選択してください。
4. [Sysprep を使用してシャットダウン] または [Sysprep を使用せずにシャットダウン] を選択します。

Windows Sysprep を実行しインスタンスをシャットダウンするかどうか確認を求められたら [はい] をクリックします。EC2Launch v2 は Windows Sysprep を実行します。次に、インスタンスからログオフされ、インスタンスがシャットダウンされます。Amazon EC2 コンソールの [Instances (インスタンス)] ページでは、インスタンスの状態が `Running` から `Stopping` に変わった後、`Stopped` に変わります。この状態になれば、インスタンスから AMI を安全に作成できます。

コマンドラインから Windows Sysprep ツールを手動で呼び出すには、次のコマンドを使います。

```
"%programfiles%\amazon\ec2launch\ec2launch.exe" sysprep --shutdown=true
```

EC2Launch で Windows Sysprep を使って AMI を作成する

EC2Launch エージェントがインストールされたインスタンスからイメージを作成すると、EC2Launch はイメージが作成されたときに特定のタスクを実行します。これには Windows Sysprep を使用した作業が含まれます。詳細については「the section called “Windows Sysprep のステップ”」を参照してください。

EC2Launch は、AMI でイメージ準備プロセスを自動化および保護する Windows Sysprep 用のデフォルトの応答ファイルとバッチファイルを提供します。これらのファイルの変更はオプションです。デフォルトでは、これらのファイルは `C:\ProgramData\Amazon\EC2-Windows\Launch\Sysprep` ディレクトリにあります。

> **Important**
>
> Windows Sysprep を使用してインスタンスのバックアップを作成しないでください。Windows Sysprep はシステム固有の情報を削除します。この情報を削除すると、インスタンスバックアップで意図しない結果が生じる可能性があります。

内容

- Windows Sysprep の EC2Launch 応答ファイルとバッチファイル
- EC2Launch で Windows Sysprep を実行する
- カスタム AMI の起動時に Server 2016 以降のメタデータ/KMS ルートを更新する

Windows Sysprep の EC2Launch 応答ファイルとバッチファイル

Windows Sysprep の EC2Launch 応答ファイルとバッチファイルには以下のものが含まれます。

`Unattend.xml`

これがデフォルトの応答ファイルです。`SysprepInstance.ps1` を実行するか、ユーザーインターフェイスで ShutdownWithSysprep を選択すると、このファイルから設定が読み取られます。

`BeforeSysprep.cmd`

このバッチファイルをカスタマイズし、EC2Launch が Windows Sysprep を実行する前にコマンドを実行します。

`SysprepSpecialize.cmd`

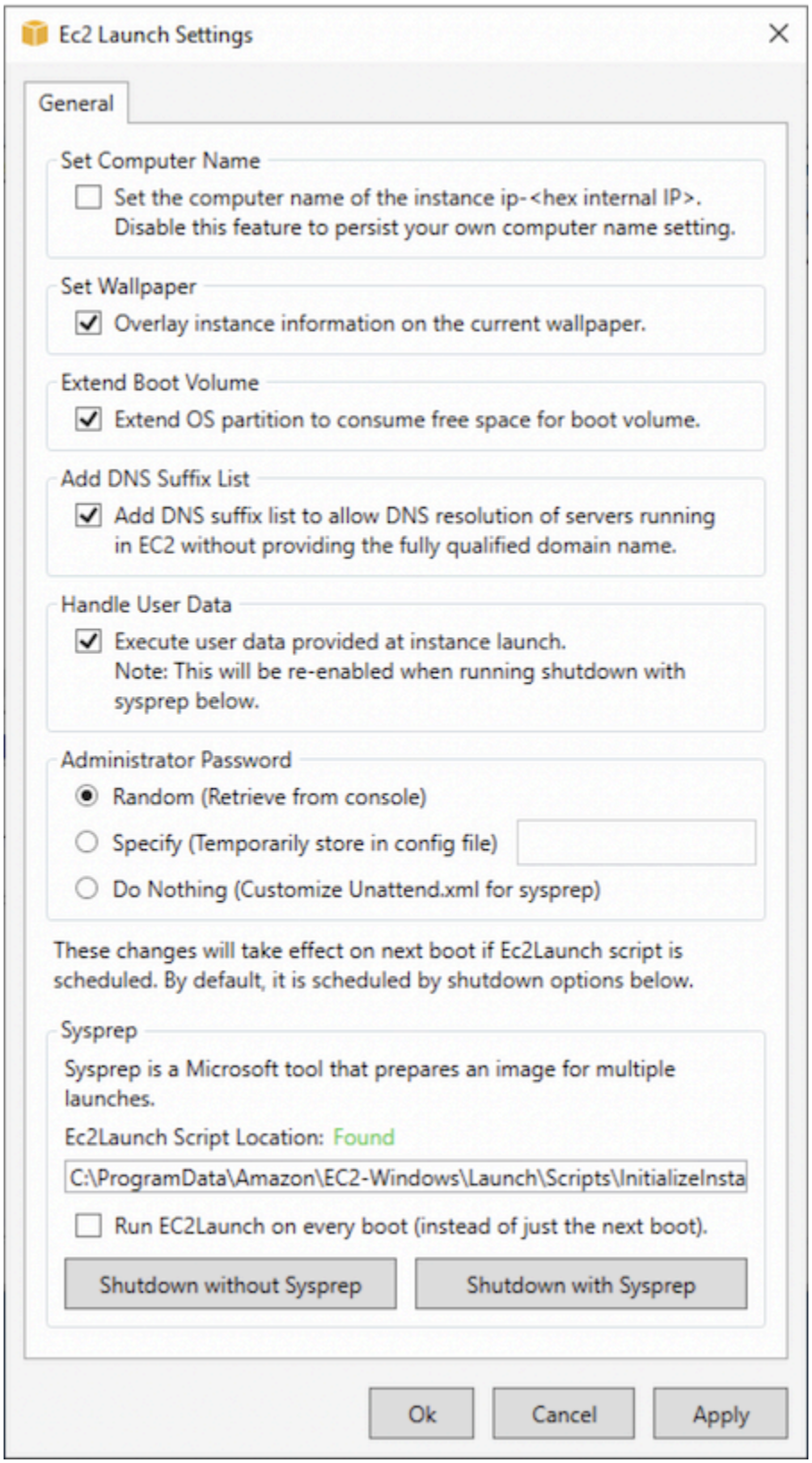
このバッチファイルをカスタマイズして、Windows Sysprep の特定化ステップ中にコマンドを実行します。

EC2Launch で Windows Sysprep を実行する

Windows Server 2016 以降の完全インストール (デスクトップ体験を含む) では、EC2Launch を使用して手動で、または [EC2 起動設定] アプリケーションを使用して、Windows Sysprep を実行できます。

[EC2 起動設定] アプリケーションを使用して Windows Sysprep を実行するには

1. Amazon EC2 コンソールで、Windows Server 2016 以降の AMI を見つけるか作成します。
2. AMI から Windows インスタンスを起動します。
3. Windows インスタンスに接続し、カスタマイズします。
4. [EC2LaunchSettings] アプリケーションを検索して実行します。このアプリケーションは、デフォルトでは `C:\ProgramData\Amazon\EC2-Windows\Launch\Settings` ディレクトリにあります。



5. 必要に応じて、オプションを選択または選択解除します。これらの設定は `LaunchConfig.json` ファイルに保存されます。

6. [Administrator Password] で、以下のいずれかを行います。

   - [Random] を選択します。EC2Launch は、ユーザーのキーを使用してパスワードを生成し、暗号化します。この設定はインスタンス起動後に無効になるため、インスタンスを再起動したり、停止して起動した場合でもパスワードは保持されます。
   - [Specify] を選択し、システム要件を満たすパスワードを入力します。このパスワードはクリアテキストとして `LaunchConfig.json` に保存され、Windows Sysprep で管理者パスワードが設定されると削除されます。ここでシャットダウンした場合、パスワードはすぐに設定されます。EC2Launch は、ユーザーのキーを使用してパスワードを暗号化します。
   - [DoNothing] を選択して、`unattend.xml` ファイルでパスワードを指定します。`unattend.xml` でパスワードを指定しない場合、管理者アカウントは無効になります。

7. [Shutdown with Sysprep (Sysprep を使用してシャットダウン)] を選択します。

EC2Launch を使用して手動で Windows Sysprep を実行するには

1. Amazon EC2 コンソールで、複製する Windows Server 2016 以降の Datacenter エディション AMI を見つけるか作成します。
2. Windows インスタンスを起動して接続します。
3. インスタンスをカスタマイズします。
4. `LaunchConfig.json` ファイルで設定を指定します。デフォルトでは、このファイルは `C:\ProgramData\Amazon\EC2-Windows\Launch\Config` ディレクトリにあります。

   `adminPasswordType` で、次のいずれかの値を指定します。

   `Random`

   EC2Launch は、ユーザーのキーを使用してパスワードを生成し、暗号化します。この設定はインスタンス起動後に無効になるため、インスタンスを再起動したり、停止して起動した場合でもパスワードは保持されます。

   `Specify`

   EC2Launch は、`adminPassword` で指定したパスワードを使用します。指定したパスワードがシステム要件を満たさない場合は、代わりに EC2Launch によってランダムなパスワードが生成されます。このパスワードはクリアテキストとして `LaunchConfig.json` に保存され、Windows Sysprep で管理者パスワードが設定されると削除されます。EC2Launch は、ユーザーのキーを使用してパスワードを暗号化します。

DoNothing

EC2Launch は、unattend.xml ファイルで指定したパスワードを使用します。unattend.xml でパスワードを指定しない場合、管理者アカウントは無効になります。

5. (オプション) 必要に応じて、unattend.xml およびその他の設定ファイルで設定を指定します。手動のインストールを計画している場合は、これらのファイルに変更を加える必要はありません。デフォルトでは、このファイルは C:\ProgramData\Amazon\EC2-Windows\Launch\Sysprep ディレクトリにあります。
6. Windows PowerShell で、./InitializeInstance.ps1 -Schedule を実行します。デフォルトでは、このスクリプトは C:\ProgramData\Amazon\EC2-Windows\Launch\Scripts ディレクトリにあります。このスクリプトは、次の起動中に初期化するようインスタンスをスケジュールします。次のステップで SysprepInstance.ps1 スクリプトを実行する前に、このスクリプトを実行する必要があります。
7. Windows PowerShell で、./SysprepInstance.ps1 を実行します。デフォルトでは、このスクリプトは C:\ProgramData\Amazon\EC2-Windows\Launch\Scripts ディレクトリにあります。

ユーザーは自動的にログオフさせられ、インスタンスがシャットダウンします。Amazon EC2 コンソールの [Instances (インスタンス)] ページを見ると、インスタンスの状態が Running から Stopping に、そして Stopped へ変わるのがわかります。この状態になれば、インスタンスから AMI を安全に作成できます。

カスタム AMI の起動時に Server 2016 以降のメタデータ/KMS ルートを更新する

カスタム AMI の起動時に Server 2016 以降のメタデータ/KMS ルートを更新するには、以下のいずれかの操作を行います。

- EC2LaunchSettings GUI (C:\ProgramData\Amazon\EC2-Windows\Launch\Settings\Ec2LaunchSettings.exe) を実行し、Windows Sysprep を使用してシャットダウンするオプションを選択します。
- AMI を作成する前に、EC2LaunchSettings を実行し、Windows Sysprep を使用しないでシャットダウンします。これにより、次回の起動時に実行される EC2 Launch の初期化タスクが設定され、インスタンスのサブネットに基づいてルートが設定されます。
- PowerShell から AMI を作成する前に、EC2 Launch の初期化タスクを手動で再スケジュールします。

> **Important**
>
> タスクを再スケジューリングする前に、デフォルトではパスワードリセット動作となることに注意してください。

- Windows のライセンス認証またはインスタンスメタデータの通信に関するエラーが発生している実行中のインスタンスのルートを更新するには、「Windows のライセンス認証を行うことができません」を参照してください。

EC2Config で Windows Sysprep を使って AMI を作成する

EC2Config サービスがインストールされたインスタンスからイメージを作成すると、EC2Config はイメージが作成されたときに特定のタスクを実行します。これには Windows Sysprep を使用した作業が含まれます。詳細については「the section called “Windows Sysprep のステップ”」を参照してください。

内容

- Windows Sysprep のアクション
- Sysprep 後
- EC2Config サービスで Windows Sysprep を実行する

Windows Sysprep のアクション

イメージを準備するために、Windows Sysprep と EC2Config サービスは次のアクションを実行します。

1. [EC2 サービスのプロパティ] ダイアログボックスで [Sysprep を使用してシャットダウンする] を選択すると、システムは ec2config.exe -sysprep コマンドを実行します。
2. EC2Config サービスは `BundleConfig.xml` ファイルの内容を読み込みます。デフォルトでは、このファイルは `C:\Program Files\Amazon\Ec2ConfigService\Settings` ディレクトリにあります。

   `BundleConfig.xml` ファイルには、以下の設定が含まれています。これらの設定は変更できます：

- AutoSysprep: Windows Sysprep を自動で使用するかどうかを示します。[EC2 サービスプロパティ] ダイアログボックスで Windows Sysprep を実行した場合、この値を変更する必要はありません。デフォルト値は No です。
- SetRDPCertificate: リモートデスクトップサーバーに自己署名証明書を設定します。これによって、リモートデスクトッププロトコール (RDP) を安全に使用して、インスタンスに接続できます。新しいインスタンスに証明書が必要な場合は、値を Yes に変更します。この設定は、Windows Server 2012 のインスタンスでは使用されません。これらのオペレーティングシステムが独自の証明書を生成できるためです。デフォルト値は No です。
- SetPasswordAfterSysprep: 新しく起動したインスタンスにランダムなパスワードを設定し、ユーザー起動キーで暗号化して、暗号化されたパスワードをコンソールに出力します。新しいインスタンスにランダムに暗号化されたパスワードを設定しない場合は、値 No に変更します。デフォルト値は Yes です。
- PreSysprepRunCmd: 実行するコマンドの場所。コマンドは、デフォルトでは C:\Program Files\Amazon\Ec2ConfigService\Scripts\BeforeSysprep.cmd ディレクトリにあります。

3. システムは BeforeSysprep.cmd を実行します。このコマンドは、次のレジストリキーを作成します。

```
reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server" /v
 fDenyTSConnections /t REG_DWORD /d 1 /f
```

   レジストリキーは再度有効になるまで RDP 接続を無効にします。RDP 接続を無効にすることは、安全上の観点より必要となります。これは、Windows Sysprep 実行後の最初のブートセッション中、RDP が接続できるわずかな時間の間に管理者パスワードが空白となるためです。

4. EC2Config サービスは、以下のコマンドを実行して Windows Sysprep を呼び出します。

```
sysprep.exe /unattend: "C:\Program Files\Amazon\Ec2ConfigService\sysprep2008.xml" /
oobe /generalize /shutdown
```

ステップの一般化

- このツールは、コンピュータ名や SID のようなイメージに固有の情報と設定を削除します。インスタンスがドメインのメンバーである場合は、そのドメインから削除されます。sysprep2008.xml 応答ファイルには、このステップに影響する以下の設定が含まれています。

  - PersistAllDeviceInstalls: この設定は、Windows セットアップがデバイスを削除したり再設定することを防ぐことによってイメージ準備プロセスを高速化します。これは、Amazon AMI を実行するためには特定のドライバーが必要となり、これらのドライバーの再検出には時間がかかるためです。
  - DoNotCleanUpNonPresentDevices: この設定では、現在存在しないデバイス用のプラグアンドプレイ情報を保持します。
- Windows Sysprep は AMI の作成の準備完了後に OS をシャットダウンします。システムは、新しいインスタンスを起動するか、または元のインスタンスを起動します。

ステップの特定化

システムは、コンピュータ名や SID など OS に固有の要件を生成します。またシステムは、sysprep2008.xml 応答ファイルで指定した設定に基づいて、次のアクションを実行します。

- CopyProfile: Windows Sysprep は、組み込まれた管理者のプロファイルを含むすべてのユーザープロファイルを削除するように設定できます。この設定は、組み込まれた管理者アカウントを保持するため、アカウント作成したすべてのカスタム設定は新しいイメージに引き継がれます。デフォルト値は True です。

  CopyProfile は、デフォルトのプロファイルを既存のローカル管理者プロファイルに置き換えます。Windows Sysprep の実行後にログインしたすべてのアカウントは、最初のログイン時にそのプロファイルとその内容のコピーを受け取ります。

  新しいイメージに引き継ぐことを希望する特定のユーザープロファイルがない場合、この設定を「いいえ」に変更します。Windows Sysprep はすべてのユーザーを削除します。これによって時間とディスク領域が節約されます。
- タイムゾーン: タイムゾーンはデフォルトで世界時 (UTC) に設定されます。
- 順序 1 の同期コマンド: システムは次のコマンドを実行して、管理者アカウントを有効化し、パスワード条件を指定します。

  net user Administrator /ACTIVE:YES /LOGONPASSWORDCHG:NO /EXPIRES:NEVER /PASSWORDREQ:YES
- 順序 2 の同期コマンド: システムは、管理者パスワードを組み換えます。このセキュリティ対策の目的は、ec2setpassword 設定を有効にしていない場合、Windows Sysprep の完了後にインスタンスへのアクセスを防ぐことです。

  C:\Program Files\Amazon\Ec2ConfigService\ScramblePassword.exe" -u Administrator

- 順序 3 の同期コマンド: システムは次のコマンドを実行します。

  C: \Program Files\Amazon\Ec2ConfigService\Scripts\SysprepSpecializePhase.cmd

  このコマンドは、RDP を再度有効にする次のレジストリキーを追加します。

  reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t REG_DWORD /d 0 /f

OOBE のステップ

1. EC2Config サービス応答ファイルを使用すると、システムは次の設定を指定します。
   - <InputLocale>en-US</InputLocale>
   - <SystemLocale>en-US</SystemLocale>
   - <UILanguage>en-US</UILanguage>
   - <UserLocale>en-US</UserLocale>
   - <HideEULAPage>true</HideEULAPage>
   - <HideWirelessSetupInOOBE>true</HideWirelessSetupInOOBE>
   - <NetworkLocation>その他</NetworkLocation>
   - <ProtectYourPC>3</ProtectYourPC>
   - <BluetoothTaskbarIconEnabled>false</BluetoothTaskbarIconEnabled>
   - <TimeZone>UTC</TimeZone>
   - <RegisteredOrganization>Amazon.com</RegisteredOrganization>
   - <RegisteredOwner>Amazon</RegisteredOwner>

   > **Note**
   >
   > 一般化と特定化ステップにおいて、EC2Config サービスは OS のステータスをモニタリングします。OS が Sysprep のステップにあることが EC2Config によって検出された場合、以下のメッセージがシステムログに発行されます。
   > EC2ConfigMonitorState: 0 Windows is being configured.
   > SysprepState=IMAGE_STATE_UNDEPLOYABLE

2. OOBE フェーズが完了すると、システムは `SetupComplete.cmd` から `C:\Windows\Setup\Scripts\SetupComplete.cmd` を実行します。2015 年 4 月以前の Amazon パブリック AMI では、このファイルは空となり、イメージには何も実行されません。2015 年 4 月以降の

パブリック AMI では、ファイルに call "C:\Program Files\Amazon\Ec2ConfigService\Scripts \PostSysprep.cmd" の値が含まれます。

3. システムが PostSysprep.cmd を実行し、次の操作を行います。
   - ローカル管理者パスワード期限切れにならないよう設定します。パスワードの期限が切れた場合、管理者がログインできないことがあります。
   - MSSQLServer マシン名を設定すると (インストールされている場合)、名前は AMI で同期されます。

Sysprep 後

Windows Sysprep が完了したら、EC2Config サービスは次のメッセージをコンソール出力へ送信します。

```
Windows sysprep configuration complete.
   Message: Sysprep Start
   Message: Sysprep End
```

そして、EC2Config は次のアクションを実行します。

1. config.xml ファイルのコンテンツを読み取り、すべての有効なプラグインを示します。
2. 「Windows の準備が終了する前」のすべてのプラグインを同時に実行します。
   - Ec2SetPassword
   - Ec2SetComputerName
   - Ec2InitializeDrives
   - Ec2EventLog
   - Ec2ConfigureRDP
   - Ec2OutputRDPCert
   - Ec2SetDriveLetter
   - Ec2WindowsActivate
   - Ec2DynamicBootVolumeSize
3. これが完了すると、「Windows の準備完了」のメッセージをインスタンスのシステム ログに送信します。
4. 「Windows の準備が終了した後」のすべてのプラグインを同時に実行します
   - Amazon CloudWatch Logs

- UserData
- AWS Systems Manager (Systems Manager)

Windows プラグインの詳細については、「EC2Config サービスを使用して、EC2 レガシー Windows オペレーティングシステムインスタンスの起動時にタスクを実行する」を参照してください。

EC2Config サービスで Windows Sysprep を実行する

以下の手順に従って、Windows Sysprep と EC2Config サービスを使用して標準化 AMI を作成します。

1. Amazon EC2 コンソールで複製を希望する AMI を見つけるか、作成します。
2. Windows インスタンスを起動して接続します。
3. カスタマイズする。
4. EC2Config サービス応答ファイルで特定設定を指定します。

   ```
   C:\Program Files\Amazon\Ec2ConfigService\sysprep2008.xml
   ```
5. Windows の [スタート] メニューから [すべてのプログラム] を選び、次に [EC2ConfigService 設定] を選択します。
6. [Ec2 サービスプロパティ] ダイアログボックスで [イメージ] タブを選択します。Ec2 サービスプロパティダイアログボックスのオプションと設定についての詳細は、「Ec2 サービスプロパティ」を参照してください。
7. 管理者パスワードのオプションを選択してから、[Shutdown with Sysprep] または [Shutdown without Sysprep] を選択します。EC2Config は、選択したパスワードオプションに基づいて設定ファイルを編集します。
   - ランダム: EC2Config はパスワードを生成してユーザーのキーで暗号化し、暗号化されたパスワードをコンソールに表示します。この設定は初回起動後に無効になるため、インスタンスを再起動したり、停止して起動した場合でもパスワードは保持されます。
   - 指定: パスワードは、Windows Sysprep 応答ファイルに暗号化されていない形式 (平文) で保存されます。Windows Sysprep が次に実行されると、管理者パスワードに設定されます。ここでシャットダウンした場合、パスワードはすぐに設定されます。サービスを再開すると、管理者パスワードは削除されます。このパスワードは後で取得できないため覚えておくことが重要です。
   - 既存のパスワードを保持: Windows Sysprep の実行時や EC2Config の再起動時に、管理者アカウントの既存のパスワードは変更されません。このパスワードは後で取得できないため覚えておくことが重要です。

8. [OK] を選択してください。

Windows Sysprep を実行しインスタンスをシャットダウンするかどうか確認を求められたら [はい] をクリックします。これにより、EC2Config が Windows Sysprep を実行します。次に、ユーザーは自動的にログオフさせられ、インスタンスがシャットダウンします。Amazon EC2 コンソールの [インスタンス] ページを見ると、インスタンスの状態が Running から Stopping、最終的に Stopped に変わるのがわかります。この状態になれば、インスタンスから AMI を安全に作成できます。

コマンドラインから Windows Sysprep ツールを手動で呼び出すには、次のコマンドを使います。

```
"%programfiles%\amazon\ec2configservice\"ec2config.exe -sysprep""
```

> **Note**
>
> CMD シェルが C:\Program Files\Amazon\EC2ConfigService\ ディレクトリ内に既に存在する場合、コマンドの二重引用符は不要です。

ただし、この操作は慎重に行ってください。Ec2ConfigService\Settings フォルダで指定した XML ファイルオプションが正しくないと、インスタンスに接続できなくなる場合があります。設定ファイルの詳細については、「EC2Config の設定ファイル」を参照してください。コマンドラインから Windows Sysprep を設定して実行する例については、「Ec2ConfigService\Scripts\InstallUpdates.ps1」を参照してください。

Amazon EC2 AMI のコピー

複数のリージョンを横断する、一貫性のある Amazon EC2 インスタンス設定が必要な場合は、すべてのインスタンスを起動するための単一の Amazon マシンイメージ (AMI) をテンプレートとして使用します。ただし、AMI はリージョンに固有のリソースであるため、特定の AWS リージョンでインスタンスを起動するときは、そのリージョンに AMI を配置する必要があります。したがって複数のリージョンで同じ AMI を使用するときは、その AMI をソースのリージョンからターゲットとなる各リージョンにコピーする必要があります。

AMI をコピーする方法は、同じパーティション内のリージョン間でコピーするか、異なるパーティションのリージョン間でコピーするかによって異なります。

- クロスリージョンコピー – AMI を同じパーティション内のリージョン間でコピーします。例えば、商用パーティション内のリージョン間でコピーするなどです。この方法については本トピックで説明します。
- クロスパーティションコピー – AMI を特定のパーティションから別のパーティションにコピーします。例えば、商用パーティションから AWS GovCloud (US) パーティションにコピーするなどです。この方法の詳細については「AMI を保存および復元する」を参照してください。
- クロスアカウントコピー – 別の AWS アカウントがお使いの AWS アカウントと共有している AMI のコピーを作成します。この方法については本トピックで説明します。

クロスリージョンコピーとクロスアカウンコピーの完了までに要する時間は、ベストエフォートベースとなります。完了時間をコントロールする必要がある場合は、15 分から 48 時間までの範囲で完了時間を指定して、AMI が所定の時間内にコピーされるようにすることができます。時間ベースの AMI コピー操作には追加料金が適用されます。詳細については、「Amazon EBS ユーザーガイド」の「料金と請求」を参照してください。

内容

- 考慮事項
- コスト
- Amazon EC2 AMI をコピーするための権限を付与する
- AMI のコピー
- 保留中の AMI コピー操作の中止
- Amazon EC2 AMI コピーの仕組み

考慮事項

- AMI をコピーするためのアクセス許可 – IAM ポリシーを使用すると、AMI をコピーするためのアクセス許可をユーザーに付与したり、それを拒否したりできます。2024 年 10 月 28 日以降、ソース AMI の `CopyImage` アクションにリソースレベルのアクセス許可を指定できます。新しい AMI のリソースレベルのアクセス許可は以前と同様に利用できます。
- 起動許可と Amazon S3 バケット許可 – AWS は、起動許可と Amazon S3 バケット許可をソース AMI から新しい AMI にコピーしません。コピー操作が完了したら、新しいAMIに起動パーミッションと Amazon S3バケットパーミッションが適用できます。
- タグ – コピーできるのは、ソース AMI にアタッチされているユーザー定義の AMI タグだけです。システムタグ (`aws:` プレフィックスが付いている) や、他の AWS アカウント がアタッチした

ユーザー定義タグはコピーされません。AMI をコピーするときに、新しい AMI とそのバックアップスナップショットに新しいタグをアタッチできます。

- 時間ベースの AMI コピーのクォータ – 累積スナップショットコピースループットクォータに達すると、それ以降の時間ベースの AMI コピーリクエストは失敗します。詳細については「Amazon EBS ユーザーガイド」の「クォータ」を参照してください。
- サポートされている送信元と送信先のコピー – コピーできるかどうか、および新しい AMI で許可される送信先はソース AMI の場所によって決まります。
  - 1 つのリージョンにあるソース AMI は、そのリージョン内、別のリージョン、そのリージョンに関連付けられた Outpost、またはそのリージョン内のローカルゾーンにコピーできます。
  - 1 つのローカルゾーンにあるソース AMI は、そのローカルゾーン内、そのローカルゾーンの親リージョン、または同じ親リージョンを持つ他の特定のローカルゾーンにコピーできます。
  - Outpost にあるソース AMI はコピーできません。
- 送信元と送信先の CLI パラメータ – CLI を使用する場合、次のパラメータを使用して、コピーする AMI の送信元の場所と新しい AMI の送信先を指定できます。コピーオペレーションはコピー先リージョンで開始する必要があります。`--region` パラメータを省略すると、AWS CLI 設定で構成されているデフォルトのリージョンがコピー先になります。

送信先	送信元パラメータ	送信先パラメータ
リージョン間のコピー	`--source-region`	`--region`
リージョンから Outpost	`--source-region`	`--destination-outpost-arn` (Outpost の ARN)
リージョンからローカルゾーン	`--source-region` ローカルゾーンの親リージョンである必要があります。	`--destination-availability-zone` (ローカルゾーンの名前) または `--destination-availability-zone-id` (ローカルゾーンの ID)
ローカルゾーンからリージョン	`--source-region` ローカルゾーンの親リージョンである必要があります。	`--region` ローカルゾーンの親リージョンである必要があります。

送信先	送信元パラメータ	送信先パラメータ
	送信元のローカルゾーンは、指定されたソース AMI ID の場所から継承されます。	
ローカルゾーンからローカルゾーン	`--source-region` ローカルゾーンの親リージョンである必要があります。 送信元のローカルゾーンは、指定されたソース AMI ID の場所から継承されます。	`--destination-availability-zone` (ローカルゾーンの名前) または `--destination-availability-zone-id` (ローカルゾーンの ID)

コスト

完了時間を指定しない場合、AMI のコピーに料金はかかりません。ただし、時間を指定して AMI をコピーした場合は、追加料金がかかります。詳細については、「Amazon EBS ユーザーガイド」の「料金と請求」を参照してください。

標準のストレージ料金とデータ転送料金が適用されます。EBS-backed AMI をコピーする場合は、追加の EBS スナップショットのストレージに対して料金が発生します。

Amazon EC2 AMI をコピーするための権限を付与する

EBS-backed AMI または Amazon S3-backed AMI をコピーするには、次の IAM アクセス許可が必要です。

- `ec2:CopyImage` – AMI をコピーするアクセス許可。EBS-backed AMI の場合、AMI のバックアップするスナップショットをコピーするアクセス許可も付与します。
- `ec2:CreateTags` – ターゲット AMI にタグ付けするアクセス許可。EBS-backed AMI の場合、ターゲット AMI のバックアップするスナップショットにタグ付けするアクセス許可も付与します。

インスタンスストアバック AMI をコピーする場合は、追加で次の IAM アクセス許可が必要です。

- `s3:CreateBucket` – 新しい AMI のターゲットリージョンに S3 バケットを作成するアクセス許可
- `s3:PutBucketOwnershipControls` – 新しく作成された S3 バケットACLs を有効にして `aws-exec-read` 既定 ACL でオブジェクトを書き込めるようにするアクセス許可
- `s3:GetBucketAcl` – ソースバケットの ACL を読み取るアクセス許可
- `s3:ListAllMyBuckets` – ターゲットリージョンで AMI の既存の S3 バケットを検出するアクセス許可
- `s3:GetObject` – ソースバケットのオブジェクトを読み取るアクセス許可
- `s3:PutObject` – ターゲットバケットにオブジェクトを書き込むアクセス許可
- `s3:PutObjectAcl` – ターゲットバケットの新しいオブジェクトのアクセス許可を書き込むアクセス許可

> **Note**
>
> 2024 年 10 月 28 日以降、ソース AMI の `CopyImage` アクションにリソースレベルのアクセス許可を指定できます。ターゲット AMI のリソースレベルのアクセス許可は以前と同様に利用できます。詳細については、「サービス認可リファレンス」の「Amazon EC2 で定義されたアクション」でテーブル内の [CopyImage] を参照してください。

EBS-backed AMI をコピーし、ターゲット AMI とスナップショットにタグ付けするための IAM ポリシーの例

次のポリシー例では、EBS-backed AMI をコピーし、ターゲット AMI とそのバックアップするスナップショットにタグを付けるアクセス許可を付与します。

> **Note**
>
> 2024 年 10 月 28 日以降、`Resource` 要素でスナップショットを指定できます。詳細については、「サービス認可リファレンス」の「Amazon EC2 で定義されたアクション」でテーブル内の [CopyImage] を参照してください。

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [{
        "Sid": "PermissionToCopyAllImages",
        "Effect": "Allow",
        "Action": [
            "ec2:CopyImage",
            "ec2:CreateTags"
        ],
        "Resource": [
            "arn:aws:ec2:*::image/*",
            "arn:aws:ec2:*::snapshot/*"
        ]
    }]
}
```

EBS-backed AMI をコピーし、新しいスナップショットのタグ付けを拒否する IAM ポリシーの例

`ec2:CopySnapshot` アクセス許可は、`ec2:CopyImage` アクセス許可を取得すると自動的に付与されます。新しいバッキングスナップショットにタグ付けするアクセス許可は明示的に拒否し、`ec2:CreateTags` アクションの `Allow` 効果を上書きできます。

次のポリシー例では、EBS-backed AMI をコピーするアクセス許可を付与しますが、ターゲット AMI の新しいバックアップするスナップショットのタグ付けは拒否します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": [
                "ec2:CopyImage",
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*::image/*",
                "arn:aws:ec2:*::snapshot/*"
            ]
        },
        {
            "Effect": "Deny",
```

```
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:::snapshot/*"
        }
    ]
}
```

Amazon S3-backed AMI をコピーし、ターゲット AMI にタグ付けするための IAM ポリシーの例

次のポリシー例では、指定されたソースバケットの Amazon S3-backed AMI を指定されたリージョンにコピーし、ターゲット AMI にタグ付けするアクセス許可を付与します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Sid": "PermissionToCopyAllImages",
            "Effect": "Allow",
            "Action": [
                "ec2:CopyImage",
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:*::image/*"
        },
        {
            "Effect": "Allow",
            "Action": "s3:ListAllMyBuckets",
            "Resource": [
                "arn:aws:s3:::*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-source-bucket/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:CreateBucket",
```

```
                "s3:GetBucketAcl",
                "s3:PutObjectAcl",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::amis-for-111122223333-in-us-east-2-hash"
            ]
        }
    ]
}
```

AMI ソースバケットの Amazon リソースネーム (ARN) を検索するには、https://console.aws.amazon.com/ec2/ で Amazon EC2 コンソールを開き、ナビゲーションペインで [AMI] を選択し、[ソース] 列でバケット名を特定します。

> **Note**
>
> `s3:CreateBucket` アクセス許可は、初めて Amazon S3-backed AMI を個々のリージョンにコピーするときにのみ必要です。その後、リージョンですでに作成された Amazon S3 バケットは、そのリージョンにコピーする将来のすべての AMI に保存されます。

AMI のコピー

コピーできるのは、自分が所有している AMI、または別のアカウントから共有された AMI です。サポートされている送信元と送信先の組み合わせについては、「考慮事項」を参照してください。

Console

AMI をコピーするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. コンソールのナビゲーションバーから、AMI を含むリージョンを選択してください。
3. ナビゲーションペインで、[AMI] を選択し、リージョンで利用できる AMI のリストを表示します。
4. コピーする AMI が表示されない場合は、別のフィルターを選択してください。AMI は、[自己所有]、[プライベートイメージ]、[パブリックイメージ]、および [無効化されたイメージ] でフィルタリングできます。

5. コピーする AMI を選択して、[アクション]、[AMI のコピー] の順に選択してください。
6. [Amazon マシンイメージ (AMI) をコピー] のページで、次の情報を指定します。

   a. [AMI コピー名] : 新しい AMI の名前。この名前にはオペレーティングシステム情報を含めることができます (Amazon EC2 は AMI の詳細を表示するときにこの情報を提供しません)。

   b. [AMI コピーの説明] : デフォルトでは、オリジナルからコピーを見分けられるように、ソース AMI に関する情報が説明に含まれています。この説明は必要に応じて変更できます。

   c. [送信先リージョン] : AMI をコピーするリージョン。詳細については、「リージョン間のコピー」および「アカウント間のコピー」を参照してください。

   d. [タグのコピー]: このチェックボックスをオンにし、AMI のコピー時にユーザー定義の AMI タグを含めます。システムタグ (aws: プレフィックスが付いている) や、他の AWS アカウント がアタッチしたユーザー定義タグはコピーされません。

   e. 時間ベースのコピー: 次のように、指定した時間内にコピーを完了させるか、ベストエフォートベースで完了させるかを指定できます。

      - 指定した時間内にコピーを完了させるには:
        - [時間ベースのコピーを有効にする] を選択します。
        - [完了時間] に、コピーに割り当てる時間 (分) を 15 分単位で入力します。完了期間は、AMI に関連付けられているすべてのスナップショットに適用されます。

          詳細については、「Amazon EBS ユーザーガイド」の「料金と請求」を参照してください。
      - ベストエフォートベースでコピーを完了するには:
        - [時間ベースのコピーを有効にする] の選択を外したままにします。

   f. (EBS バックされた AMI のみ) [AMI コピーの EBS スナップショットを暗号化]: ターゲットスナップショットを暗号化するか、別のキーを使用して再暗号化する場合、このチェックボックスを選択してください。デフォルトで暗号化を有効にしている場合、[AMI コピーの EBS スナップショットを暗号化] チェックボックスがオンになり、オフにできません。詳細については、「暗号化とコピー」を参照してください。

   g. (EBS-backed AMI のみ) [KMS キー]: ターゲットスナップショットを暗号化するための KMS キー。

   h. [タグ]: 新しい AMI と新しいスナップショットに同じタグを付けることも、異なるタグでタグ付けすることもできます。

- 新しい AMI と新しいスナップショットに同じタグを付けるには、[イメージとスナップショットに対し一緒にタグを付けます] を選択してください。新しい AMI と作成されるすべてのスナップショットには、同じタグが適用されます。
- 新しい AMI と新しいスナップショットに異なるタグを付けるには、[イメージとスナップショットに対し個別にタグを付けます] を選択してください。新しい AMI と作成されるスナップショットには、異なるタグが適用されます。ただし、作成されるすべての新しいスナップショットには同じタグが付けられることに注意してください。新しいそれぞれのスナップショットに異なるタグを付けることはできません。

(オプション) タグを追加するには、[タグ追加] を選択し、そのタグのキーと値を入力してください。各タグについて、これを繰り返します。

i. AMI をコピーする準備ができたら、[AMI のコピー] を選択してください。

新しい AMI の初期ステータスは Pending です。ステータスが Available になると、AMI のコピー操作は完了です。

AWS CLI

リージョン間で AMI をコピーするには

copy-image コマンドを使用します。コピー元リージョンおよび送信先リージョンの両方を指定する必要があります。コピー元のリージョンは、--source-region パラメータを使用して指定します。--region パラメータを使用して送信先リージョンを指定できます (このパラメータを省略すると AWS CLI 設定で構成されているデフォルトのリージョンが使用されます)。

```
aws ec2 copy-image \
    --source-image-id ami-0abcdef1234567890 \
    --source-region us-west-2 \
    --name my-ami \
    --region us-east-1
```

AMI のコピー時にターゲットスナップショットを暗号化する場合は、追加のパラメータを (--encrypted および --kms-key-id) を指定する必要があります。

リージョンからローカルゾーンに AMI をコピーするには

copy-image コマンドを使用します。送信元と送信先の両方を指定する必要があります。コピー元のリージョンは、--source-region パラメータを使用して指定します。送信先のローカ

ルゾーンは、--destination-availability-zone パラメータを使用して指定します (--destination-availability-zone-id を使用することもできます)。リージョン内の AMI は、同じリージョン内のローカルゾーンにのみコピーできます。

```
aws ec2 copy-image \
    --source-image-id ami-0abcdef1234567890 \
    --source-region cn-north-1 \
    --destination-availability-zone cn-north-1-pkx-1a \
    --name my-ami \
    --region cn-north-1
```

ローカルゾーンからリージョンに AMI をコピーするには

copy-image コマンドを使用します。送信元と送信先の両方を指定する必要があります。コピー元のリージョンは、--source-region パラメータを使用して指定します。--region パラメータを使用して送信先リージョンを指定します (このパラメータを省略すると AWS CLI 設定で構成されているデフォルトのリージョンが使用されます)。送信元のローカルゾーンは、指定されたソース AMI ID の場所から継承されます。ローカルゾーンの AMI は、そのローカルゾーンの親リージョンにのみコピーできます。

```
aws ec2 copy-image \
    --source-image-id ami-0abcdef1234567890 \
    --source-region cn-north-1 \
    --name my-ami \
    --region cn-north-1
```

ローカルゾーン間で AMI をコピーするには

copy-image コマンドを使用します。送信元と送信先の両方を指定する必要があります。送信元のローカルゾーンのリージョンは、--source-region パラメータを使用して指定します。送信先のローカルゾーンは、--destination-availability-zone パラメータを使用して指定します (--destination-availability-zone-id を使用することもできます)。送信元のローカルゾーンは、指定されたソース AMI ID の場所から継承されます。--region パラメータを使用して送信先ローカルゾーンの親リージョンを指定します (このパラメータを省略すると AWS CLI 設定で構成されているデフォルトのリージョンが使用されます)。

```
aws ec2 copy-image \
    --source-image-id ami-0abcdef1234567890 \
    --source-region cn-north-1 \
    --destination-availability-zone cn-north-1-pkx-1a \
```

```
    --name my-ami \
    --region cn-north-1
```

PowerShell

リージョン間で AMI をコピーするには

Copy-EC2Image コマンドレットを使用します。コピー元リージョンおよび送信先リージョンの両方を指定する必要があります。コピー元のリージョンは、-SourceRegion パラメータを使用して指定します。送信先リージョンは、-Region パラメータ、または Set-AWSDefaultRegion コマンドレットを使用して指定できます。

```
Copy-EC2Image `
    -SourceImageId ami-0abcdef1234567890 `
    -SourceRegion us-west-2 `
    -Name my-ami `
    -Region us-east-1
```

AMI のコピー時にターゲットスナップショットを暗号化する場合は、追加のパラメータを (-Encrypted および -KmsKeyId) を指定する必要があります。

リージョンからローカルゾーンに AMI をコピーするには

Copy-EC2Image コマンドレットを使用します。送信元と送信先の両方を指定する必要があります。コピー元のリージョンは、-SourceRegion パラメータを使用して指定します。送信先のローカルゾーンは、-DestinationAvailabilityZone パラメータを使用して指定します (-DestinationAvailabilityZoneId を使用することもできます)。リージョン内の AMI は、同じリージョン内のローカルゾーンにのみコピーできます。

```
Copy-EC2Image `
    -SourceImageId ami-0abcdef1234567890 `
    -SourceRegion cn-north-1 `
    -DestinationAvailabilityZone cn-north-1-pkx-1a `
    -Name my-ami `
    -Region cn-north-1
```

ローカルゾーンからリージョンに AMI をコピーするには

Copy-EC2Image コマンドレットを使用します。送信元と送信先の両方を指定する必要があります。コピー元のリージョンは、-SourceRegion パラメータを使用して指定します。送信先リージョンは、-Region パラメータ、または Set-AWSDefaultRegion コマンドレットを使用して指定

します。送信元のローカルゾーンは、指定されたソース AMI ID の場所から継承されます。ローカルゾーンの AMI は、そのローカルゾーンの親リージョンにのみコピーできます。

```
Copy-EC2Image `
    -SourceImageId ami-0abcdef1234567890 `
    -SourceRegion cn-north-1 `
    -Name my-ami `
    -Region cn-north-1
```

ローカルゾーン間で AMI をコピーするには

Copy-EC2Image コマンドレットを使用します。送信元と送信先の両方を指定する必要があります。送信元のローカルゾーンのリージョンは、-SourceRegion パラメータを使用して指定します。送信先のローカルゾーンは、-DestinationAvailabilityZone パラメータを使用して指定します (-DestinationAvailabilityZoneId を使用することもできます)。送信元のローカルゾーンは、指定されたソース AMI ID の場所から継承されます。送信先のローカルゾーンの親リージョンは、-Region パラメータ、または Set-AWSDefaultRegion コマンドレットを使用して指定できます。

```
Copy-EC2Image `
    -SourceImageId ami-0abcdef1234567890 `
    -SourceRegion cn-north-1 `
    -DestinationAvailabilityZone cn-north-1-pkx-1a `
    -Name my-ami `
    -Region cn-north-1
```

保留中の AMI コピー操作の中止

次の手順を使用して、保留中の AMI コピーを停止できます。

Console

AMI のコピー操作を中止するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションバーのリージョンセレクターから対象のリージョンを選択してください。
3. ナビゲーションペインで [AMI] を選択してください。
4. コピーを中止する AMI を選択し、[アクション]、[AMI を登録解除] を選択してください。
5. 確認を求めるメッセージが表示されたら、[AMI の登録解除] を選択してください。

AWS CLI

AMI のコピー操作を中止するには

deregister-image コマンドを使用します。

```
aws ec2 deregister-image --image-id ami-0abcdef1234567890
```

PowerShell

を使用して AMI のコピー操作を中止するには

Unregister-EC2Image コマンドレットを使用します。

```
Unregister-EC2Image -ImageId ami-0abcdef1234567890
```

Amazon EC2 AMI コピーの仕組み

ソース AMI をコピーすると、見た目は同じでもまったく別のターゲット AMI とも呼ばれる新しい AMI になります。ターゲット AMI にはそれ独自の AMI ID があります。ソース AMI は、ターゲット AMI に影響を及ぼさずに変更または登録解除できます。逆の場合も同様です。

EBS-backed AMI を使用すると、それぞれのバックアップするスナップショットは、同一だが区別されるターゲットスナップショットにコピーされます。AMI を新しいリージョンにコピーすると、スナップショットは完全な (増分ではない) コピーになります。暗号化されていないバッキングスナップショットを暗号化するか、新しい KMS キーに暗号化すると、スナップショットは完全な (増分ではない) コピーになります。以降の AMI のコピー操作では、バッキングスナップショットの増分コピーが作成されます。

内容

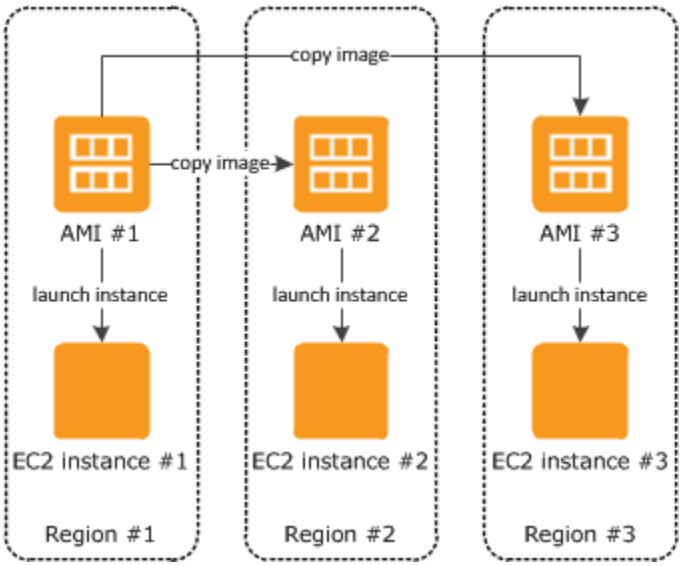
- リージョン間のコピー
- アカウント間のコピー
- 時間ベースの AMI コピー操作
- 暗号化とコピー

リージョン間のコピー

地理的に分散したリージョンに AMI をコピーすると、次のような利点があります。

- 一貫性のあるグローバルなデプロイメント: 1 つのリージョンから別のリージョンに AMI をコピーすることで、一貫性のあるインスタンスを同じ AMI から別のリージョンに起動できます。
- スケーラビリティ: ユーザーの場所にかかわらず、ユーザーのニーズに合ったグローバルアプリケーションをより簡単に設計できます。
- パフォーマンス: アプリケーションを配布したり、アプリケーションの重要なコンポーネントをユーザーの近くに配置したりすることでパフォーマンスを向上できます。また、インスタンスの種類やその他の AWS サービスなど、リージョン固有の機能を活用することもできます。
- 高可用性: アプリケーションを設計し、AWS リージョン全体にわたってデプロイして可用性を高めることができます。

次の図は、ソース AMI と異なるリージョンにある 2 つのコピーされた AMI、およびそこから起動される EC2 インスタンスの関係を示します。AMI からインスタンスを起動すると、AMI が存在する同じリージョンに存在します。ソース AMI を変更し、それらの変更をターゲットリージョンの AMI に反映させる場合、ソース AMI をターゲットリージョンに再度コピーする必要があります。



Amazon S3-backed AMI を最初にリージョンにコピーするときに、そのリージョンにコピーされた AMI に Amazon S3 バケットを作成します。そのリージョンにコピーするすべての Amazon S3-backed AMI が、このバケットに保存されます。バケット名の形式は次のとおりです: AMI-for-####-in-#####-#### 例: `amis-for-123456789012-in-us-east-2-yhjmxvp6`。

前提条件

AMI をコピーする前に、ソース AMI のすべての内容が、異なるリージョンでの実行をサポートするように更新されていることを確認する必要があります。例えば、データベース接続文字列や同様のアプリケーション設定データが、適切なリソースを指すように更新する必要があります。それ以外の場

合、対象のリージョンの新しい AMI から起動したインスタンスは元のリージョンのリソースをまだ使用している可能性があり、それによりパフォーマンスとコストに影響が及ぶことがあります。

制限事項

- コピー先のリージョンには、AMI の同時コピー操作が 300 個に制限されます。これは、時間ベースの AMI コピー操作にも適用されます。
- 準仮想化 (PV) AMI がサポートされていないリージョンに、PV AMI をコピーすることはできません。詳細については、「仮想化タイプ」を参照してください。

アカウント間のコピー

他の AWS アカウント の AMI が AWS アカウント と共有されている場合は、共有 AMI をコピーできます。これは、クロスアカウントコピーと呼ばれています。共有される AMI はソース AMI です。ソース AMI をコピーする際に、新しい AMI を作成します。新しい AMI は、多くの場合、ターゲット AMI と呼ばれます。

AMI のコスト

- 共有されている AMI の場合、リージョン内のストレージについて、共有されている AMI のアカウントに課金されます。
- 自分のアカウントと共有された AMI をコピーした場合、アカウントのコピー先の AMI の所有者は自分になります。
  - ソース AMI の所有者は、標準の Amazon EBS または Amazon S3 転送料金が課金されます。
  - 宛先リージョン内のターゲット AMI のストレージについて料金が課金されます。

リソースのアクセス許可

別のアカウントと共有した AMI をコピーするときは、ソース AMI の所有者から、AMI 自体の読み取り許可だけでなく、それをバックアップしているストレージの読み取り許可も、付与してもらう必要があります。ストレージは、関連 EBS スナップショット (Amazon EBS-backed AMI の場合) か、関連 S3 バケット (Amazon S3-backed AMI の場合) のどちらかです。共有 AMI に暗号化されたスナップショットがある場合、所有者はキーをユーザーと共有する必要があります。リソースのアクセス許可付与についての詳細は、EBS スナップショットの場合は「Amazon EBS ユーザーガイド」の「Amazon EBS スナップショットを他の AWS アカウント と共有」を参照してください。S3 バケットの場合、「Amazon S3 ユーザーガイド」の「Amazon S3 で ID とアクセスの管理」を参照してください。

> **Note**
>
> ソース AMI にアタッチされたタグは、複数のアカウントでターゲット AMI にコピーされません。

時間ベースの AMI コピー操作

単一の関連付けられたスナップショットを持つ EBS-backed AMI に対して時間ベースの AMI コピー操作を開始すると、個々の時間ベースのスナップショットコピー操作と同じ動作をし、同じスループット制限が適用されます。

複数の関連スナップショットを持つ EBS-backed AMI に対して時間ベースの AMI コピー操作を開始すると、同時に実行される時間ベースのスナップショットコピー操作と同じように動作し、同じスループット制限が適用されます。関連付けられたスナップショットごとに個別のスナップショットコピーリクエストが発生し、それぞれが累積スナップショットコピースループットクォータに影響します。指定した完了期間は、関連付けられている各スナップショットに適用されます。

詳細については、「Amazon EBS ユーザーガイド」の「料金と請求」を参照してください。

暗号化とコピー

次の表は、各種 AMI コピーのシナリオにおける暗号化サポートを示します。暗号化されたスナップショットを生成するために暗号化されていないスナップショットをコピーすることはできますが、暗号化されていないスナップショットを生成するために暗号化されたスナップショットをコピーすることはできません。

シナリオ	説明	サポート対象
1	非暗号化から非暗号化	はい
2	暗号化から暗号化	はい
3	非暗号化から暗号化	はい
4	暗号化から非暗号化	いいえ

> **Note**
>
> CopyImage アクション中の暗号化は Amazon EBS-backed AMI にのみ適用されます。Amazon S3-backed AMI はスナップショットを使用しないため、コピーを使用して暗号化ステータスを変更することはできません。

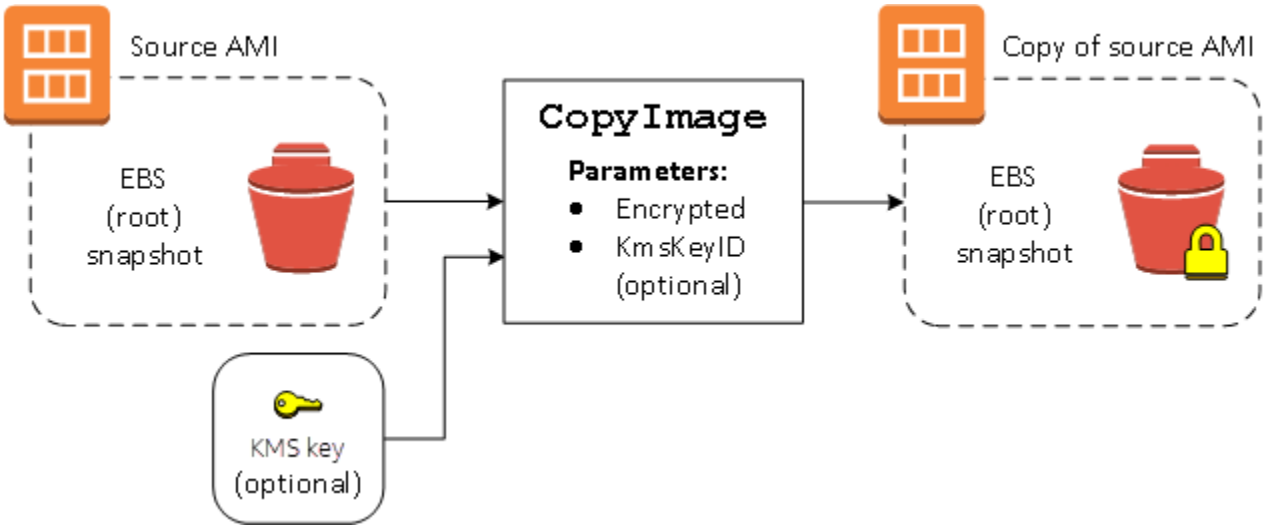
暗号化パラメータを指定しないで AMI をコピーすると、デフォルトでバックアップするスナップショットは元の暗号化ステータスと一緒にコピーされます。したがって、ソース AMI が暗号化されていないスナップショットでバックアップされている場合、結果のターゲットスナップショットも暗号化されません。同様に、ソース AMI のスナップショットが暗号化されている場合、結果のターゲットスナップショットも同じ AWS KMS キーで暗号化されます。複数のスナップショットでバックアップされている AMI の場合、各ターゲットスナップショットは対応するソーススナップショットの暗号化状態を保持します。

AMI コピー中にターゲットバッキングスナップショットの暗号化状態を変更するには、暗号化パラメータを指定できます。次の例では、ターゲット AMI の暗号化状態を変更するために CopyImage アクションに暗号化パラメータが指定されるデフォルトではないケースを示しています。

暗号化されていないソース AMI の暗号化されたターゲット AMI へのコピー

このシナリオでは、暗号化されていないルートスナップショットでバックアップされた AMI は、暗号化されたルートスナップショットを持つ AMI にコピーされます。CopyImage アクションは、カスタマー管理された型キーなど、2 つの暗号化パラメータで呼び出されます。その結果、ルートスナップショットの暗号化ステータスが変更され、ターゲット AMI はソーススナップショットと同じデータを含むルートスナップショットにバックアップされますが、指定されたキーを使用して暗号化されます。両方の AMI でスナップショットのストレージコストと、いずれかの AMI から起動するインスタンスの料金が発生します。

> **Note**
>
> デフォルトで暗号化を有効にすると、AMI 内のすべてのスナップショットで Encrypted パラメータを true に設定したのと同じ効果があります。



Encrypted パラメータを設定すると、このインスタンスの単一のスナップショットが暗号化されます。KmsKeyId パラメータを指定しない場合は、デフォルトのカスタマー管理された型キーを使用して、スナップショットのコピーが暗号化されます。

暗号化されたスナップショットを持つ AMI のコピーの詳細については、「EBS-backed AMI での暗号化の利用」を参照してください。

S3 を使用して AMI を保存および復元する

Amazon マシンイメージ (AMI) を Amazon S3 バケットに保存し、AMI を別の S3 バケットにコピーして、S3 バケットから復元できます。S3 バケットを使用して AMI を保存および復元することで、AMI をある AWS パーティションから別のパーティション (例えば、主要な商用パーティションから AWS GovCloud (US) パーティション) にコピーできます。AMI を S3 バケットに保存することで、AMI のアーカイブコピーを作成することもできます。

S3 を使用した AMI の保存および復元のサポート対象の API は、CreateStoreImageTask、DescribeStoreImageTasks、および CreateRestoreImageTask です。

CopyImage は、AWS パーティション内の AMI のコピーの際に使用することが推奨される API です。ただし、CopyImage は、AMI を別のパーティションにコピーできません。

AWS パーティションの詳細については、「IAM ユーザーガイド」の「Amazon リソースネーム (ARN)」ページの「#######」を参照してください。

> **Warning**
>
> AWS パーティションまたは AWS リージョン間でデータを移動する場合、適用されるすべての法令およびビジネス要件 (適用される政府の規制およびデータ所在地に関する要件を含みますが、これらに限られません) を確実に遵守してください。

内容

- ユースケース
- 制限事項
- コスト
- AMI の保存と復元の仕組み
- 保存イメージタスクを作成する

ユースケース

保存 API と復元 API を使用して、次の操作を実行します。

- AWS パーティション間で AMI をコピーする
- AMI のアーカイブコピーを作成する

AWS パーティション間で AMI をコピーする

S3 バケットを使用して AMI を保存および復元することで、ある AWS パーティションから別のパーティションに、またはある AWS リージョンから別のリージョンに AMI をコピーできます。次の例では、主要な商用パーティションから AWS GovCloud (US) パーティションに、具体的には `us-east-2` リージョンから `us-gov-east-1` リージョンに AMI をコピーします。

あるパーティションから別のパーティションに AMI をコピーするには、次の手順に従います。

- `CreateStoreImageTask` を使用して、現在のリージョンの S3 バケットに AMI を保存します。この例では、S3 バケットは `us-east-2` にあります。
- `DescribeStoreImageTasks` を使用して、保存タスクの進行状況をモニタリングします。タスクが完了すると、オブジェクトが S3 バケットに表示されます。
- 任意の手順を使用して、保存された AMI オブジェクトをターゲットパーティションの S3 バケットにコピーします。この例では、S3 バケットは `us-gov-east-1` にあります。

> **Note**
>
> パーティションごとに異なる AWS 認証情報が必要なため、S3 オブジェクトをあるパーティションから別のパーティションに直接コピーすることはできません。パーティション間で S3 オブジェクトをコピーするプロセスは、このドキュメントの対象外です。例として、次のコピープロセスを提供していますが、お客様のセキュリティ要件を満たすコピープロセスを使用する必要があります。
>
> - パーティション間で 1 つの AMI をコピーするためのコピープロセスはシンプルです。ソースバケットから中間ホスト (EC2 インスタンスやラップトップなど) にオブジェクトをダウンロードし、中間ホストからターゲットバケットにオブジェクトをアップロードするだけです。プロセスの各段階で、パーティションの AWS 認証情報を使用します。
> - より持続的な使用のために、コピーを管理するアプリケーションの開発をご検討ください。S3 マルチパートダウンロードとアップロードを使用することも考慮に値します。

- `CreateRestoreImageTask` を使用して、ターゲットパーティションの S3 バケットから AMI を復元します。この例では、S3 バケットは `us-gov-east-1` にあります。
- その状態が使用可能になるタイミングを確認するために、AMI を記述して復元タスクの進行状況をモニタリングします。また、スナップショットを記述することで、復元される AMI を構成するスナップショットの進行状況 (%) をモニタリングすることもできます。

AMI のアーカイブコピーを作成する

AMI を S3 バケットに保存することで、AMI のアーカイブコピーを作成できます。AMI は S3 内の 1 つのオブジェクトにパックされ、すべての AMI メタデータ (共有情報を除く) は、保存された AMI の一部として保持されます。AMI データは、ストレージプロセスの一環として圧縮されます。簡単に圧縮できるデータを含む AMI は、S3 で小さめのオブジェクトとなります。コストを削減するために、より安価な S3 ストレージ階層を使用できます。詳細については、Amazon S3 ストレージクラスおよび Amazon S3 の料金をご参照ください。

制限事項

- AMI を保存するには、AWS アカウント が AMI とそのスナップショットを所有しているか、AMI とそのスナップショットをアカウントと直接共有する必要があります。公開されているだけの AMI は保存できません。
- これらの API を使用して保存できるのは、EBS-backed AMI だけです。

- 準仮想化 (PV) AMI はサポートされていません。
- 保存可能な AMI の上限サイズ (圧縮前) は、5,000 GB です。
- 保存イメージリクエストのクォータ: 進行中の 1,200 GB の保存作業 (スナップショットデータ)。
- 復元イメージリクエストのクォータ: 進行中の 600 GB の復元作業 (スナップショットデータ)。
- 保存タスク中は、スナップショットを削除してはならず、保存を実行する IAM プリンシパルにはスナップショットへのアクセス権が必要です。それ以外の場合は、保存プロセスが失敗します。
- 同じ S3 バケットに AMI の複数のコピーを作成することはできません。
- S3 バケットに保存されている AMI は、元の AMI ID では復元できません。AMI エイリアシングを使用すると、これを軽減できます。
- 現在、保存 API と復元 API は、AWS Command Line Interface、AWS SDK、および Amazon EC2 API を使用する場合にのみサポートされます。Amazon EC2 コンソールを使用して AMI を保存および復元することはできません。

コスト

S3 を使用して AMI を保存および復元する場合、保存 API と復元 API で使用されるサービス、およびデータ転送について料金が発生します。API は、S3 と EBS Direct API を使用します (これらの API がスナップショットデータにアクセスするために内部的に使用されます)。詳細については、Amazon S3 の料金および Amazon EBS の料金をご参照ください。

AMI の保存と復元の仕組み

S3 を使用して AMI を保存および復元するには、次の API を使用します。

- `CreateStoreImageTask` – AMI を S3 バケットに保存する
- `DescribeStoreImageTasks` – AMI 保存タスクの進行状況を示す
- `CreateRestoreImageTask` – S3 バケットから AMI を復元する

API の仕組み

- CreateStoreImageTask
- DescribeStoreImageTasks
- CreateRestoreImageTask
- ファイルパス

CreateStoreImageTask

`CreateStoreImageTask` API は、AMI を単一のオブジェクトとして S3 バケットに保存します。

API は、AMI とそのスナップショットからすべてのデータを読み取るタスクを作成し、S3 マルチパートアップロードを使用して S3 オブジェクトにデータを保存します。API は、リージョン固有でない AMI メタデータの大部分を含む AMI のすべてのコンポーネント、および AMI に含まれるすべての EBS スナップショットを取得し、S3 内の単一のオブジェクトにパックします。データは、S3 で使用される領域の量を削減するために、アップロードプロセスの一環として圧縮されるので、S3 内のオブジェクトは AMI 内のスナップショットのサイズの合計よりも小さくなる可能性があります。

この API を呼び出すアカウントに AMI タグとスナップショットタグが表示されている場合、それらは保持されます。

S3 のオブジェクトは AMI と同じ ID を持っていますが、`.bin` 拡張子が付いています。AMI 名、AMI の説明、AMI の登録日、AMI の所有者アカウント、および保存オペレーションのタイムスタンプといったデータも S3 オブジェクトに S3 メタデータタグとして保存されます。

タスクを完了するのにかかる時間は、AMI のサイズによって異なります。また、タスクがキューに入れられているため、進行中の他のタスクの数にも依存します。`DescribeStoreImageTasks` API を呼び出してタスクの進行状況を追跡できます。

進行中のすべての AMI のサイズの合計は、アカウントごとに 1,200 GB の EBS スナップショットデータに制限されます。進行中のタスクが制限未満になるまで、それ以降のタスクの作成は拒否されます。例えば、200 GB のスナップショットデータを持つ AMI と 400 GB のスナップショットデータを持つ別の AMI が現在保存されている場合、別のリクエストが受け入れられます。これは、進行中の合計が 600 GB で、制限未満であるためです。ただし、1,200 GB のスナップショットデータを持つ 1 つの AMI が現在保存されようとしている場合は、タスクが完了するまでそれ以降のタスクは拒否されます。

DescribeStoreImageTasks

`DescribeStoreImageTasks` API は、AMI の保存タスクの進行状況を示します。指定した AMI のタスクを記述できます。AMI を指定しない場合、過去 31 日間に処理されたすべての保存イメージタスクのページ分割されたリストが表示されます。

各 AMI タスクについて、応答では、タスクが `InProgress`、`Completed`、または `Failed` のいずれであるかが示されます。`InProgress` のタスクの場合、応答では、進行状況がパーセンテージで示されます。

タスクは時系列の逆順にリストされます。

現時点では、前月のタスクのみを表示できます。

CreateRestoreImageTask

`CreateRestoreImageTask` API は、`CreateStoreImageTask` リクエストを使用して以前に作成された S3 オブジェクトから AMI を復元するタスクを開始します。

復元タスクは、保存タスクが実行されたリージョンと同じリージョンまたは別のリージョンで実行できます。

AMI オブジェクトの復元ソースである S3 バケットは、復元タスクがリクエストされたリージョンと同じリージョンに存在する必要があります。AMI はこのリージョンで復元されます。

AMI は、保存された AMI の値に対応する名前、説明、ブロックデバイスマッピングなどのメタデータとともに復元されます。このアカウントの名前は、リージョン内の AMI に対して一意である必要があります。名前を指定しない場合、新しい AMI は元の AMI と同じ名前になります。AMI は、復元プロセス時に生成される新しい AMI ID を取得します。

AMI 復元タスクを完了するのにかかる時間は、AMI のサイズによって異なります。また、タスクがキューに入れられているため、進行中の他のタスクの数にも依存します。タスクの進行状況は、AMI (describe-images) またはその EBS スナップショット (describe-snapshots) を記述することで確認できます。タスクが失敗すると、AMI とスナップショットは失敗の状態に移行されます。

進行中のすべての AMI のサイズの合計は、アカウントあたり 600 GB (復元後のサイズに基づく) の EBS スナップショットデータに制限されます。進行中のタスクが制限未満になるまで、それ以降のタスクの作成は拒否されます。

ファイルパス

AMI を保存および復元するときは、次の方法でファイルパスを使用できます。

- AMI を S3 に保存する場合、ファイルパスをバケット名に追加できます。システム内部では、バケット名からパスが分離され、AMI を保存するために生成されたオブジェクトキーにそのパスが追加されます。完全なオブジェクトパスは、API 呼び出しからのレスポンスに表示されます。
- AMI を復元する場合、オブジェクトキーパラメータを使用できるので、オブジェクトキー値の先頭にパスを追加できます。

例: ファイルパスが追加されたバケット名

AMI を保存する場合、バケット名の後にファイルパスを指定します。

```
amzn-s3-demo-bucket/path1/path2
```

作成されるオブジェクトキーを次に示します。

```
path1/path2/ami-0abcdef1234567890.bin
```

AMI を復元するときは、バケット名とオブジェクトキーの両方を指定します。例については「the section called “保存イメージタスクを作成する”」を参照してください。

保存イメージタスクを作成する

AMI を S3 バケットに保存するときに、保存イメージタスクが作成されます。保存イメージタスクを使用して、プロセスの進行状況と結果をモニタリングできます。

内容

- AMI のセキュリティ保護
- S3 を使用して AMI を保存および復元するためのアクセス権限
- 保存イメージタスクを作成する
- 復元イメージタスクを作成する

AMI のセキュリティ保護

AMI のコンテンツを保護するために十分なセキュリティをもって S3 バケットが確実に設定されていること、および AMI オブジェクトがバケット内に残っている限り、セキュリティが確実に維持されるようにすることが重要です。これを実行できない場合は、これらの API の使用はお勧めしません。S3 バケットへのパブリックアクセスが許可されていないことを確認します。必須ではありませんが、AMI を保存する S3 バケットのためにサーバー側の暗号化を有効にすることをお勧めします。

S3 バケットに適切なセキュリティを設定する方法については、次のセキュリティトピックをご参照ください。

- Amazon S3 ストレージへのパブリックアクセスのブロック
- Amazon S3 バケット向けのサーバー側のデフォルトの暗号化動作の設定
- AWS Config ルールの s3-bucket-ssl-requests-only に準拠するには、どの S3 バケットポリシーを使用できますか?

- Amazon S3 サーバーアクセスログ記録の有効化

AMI スナップショットが S3 オブジェクトにコピーされると、データは TLS 接続を介してコピーされます。暗号化されたスナップショットを使用して AMI を保存できますが、スナップショットは保存プロセスの一部として復号されます。

S3 を使用して AMI を保存および復元するためのアクセス権限

IAM プリンシパルが Amazon S3 を使用して AMI を保存または復元する場合は、必要な許可を付与する必要があります。

次のポリシーの例には、IAM プリンシパルが保存タスクと復元タスクを実行できるようにするために必要なすべてのアクションが含まれています。

特定のリソースへのアクセス権のみをプリンシパルに付与する IAM ポリシーを作成することもできます。ポリシーの例については、「IAM ユーザーガイド」の「AWS リソースのアクセス管理」を参照してください。

> **Note**
>
> AMI を構成するスナップショットが暗号化されている場合、またはアカウントの暗号化がデフォルトで有効になっている場合は、IAM プリンシパルに KMS キーを使用するための許可が付与されている必要があります。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:DeleteObject",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:PutObject",
                "s3:PutObjectTagging",
                "s3:AbortMultipartUpload",
```

```
                "ebs:CompleteSnapshot",
                "ebs:GetSnapshotBlock",
                "ebs:ListChangedBlocks",
                "ebs:ListSnapshotBlocks",
                "ebs:PutSnapshotBlock",
                "ebs:StartSnapshot",
                "ec2:CreateStoreImageTask",
                "ec2:DescribeStoreImageTasks",
                "ec2:CreateRestoreImageTask",
                "ec2:GetEbsEncryptionByDefault",
                "ec2:DescribeTags",
                "ec2:CreateTags"
            ],
            "Resource": "*"
        }
    ]
}
```

保存イメージタスクを作成する

AMI を S3 バケットに保存するには、まず保存イメージタスクを作成します。タスクを完了するのにかかる時間は、AMI のサイズによって異なります。タスクが成功または失敗するまで、タスクの進行状況を追跡できます。

AWS CLI

保存イメージタスクを作成するには

create-store-image-task コマンドを使用します。

```
aws ec2 create-store-image-task \
    --image-id ami-0abcdef1234567890 \
    --bucket amzn-s3-demo-bucket
```

以下は出力の例です。

```
{
  "ObjectKey": "ami-0abcdef1234567890.bin"
}
```

保存イメージタスクの進行状況を表示するには

describe-store-image-tasks コマンドを使用します。

```
aws ec2 describe-store-image-tasks \
    --image-ids ami-0abcdef1234567890 \
    --query StoreImageTaskResults[].StoreTaskState \
    --output text
```

以下は出力の例です。

```
InProgress
```

PowerShell

保存イメージタスクを作成するには

New-EC2StoreImageTask コマンドレットを使用します。

```
New-EC2StoreImageTask `
    -ImageId ami-0abcdef1234567890 `
    -Bucket amzn-s3-demo-bucket
```

以下は出力の例です。

```
ObjectKey         : ami-0abcdef1234567890.bin
```

保存イメージタスクの進行状況を表示するには

Get-EC2StoreImageTask コマンドレットを使用します。

```
(Get-EC2StoreImageTask -ImageId ami-0abcdef1234567890).StoreTaskState
```

以下は出力の例です。

```
InProgress
```

復元イメージタスクを作成する

復元された AMI の名前を指定する必要があります。このアカウントの名前は、リージョン内の AMI に対して一意である必要があります。復元された AMI は、新しい AMI ID を取得します。

AWS CLI

復元イメージタスクを作成するには

create-restore-image-task コマンドを使用します。

```
aws ec2 create-restore-image-task \
    --object-key ami-0abcdef1234567890.bin \
    --bucket amzn-s3-demo-bucket \
    --name "my-restored-ami"
```

以下は出力の例です。

```
{
   "ImageId": "ami-1234567890abcdef0"
}
```

PowerShell

復元イメージタスクを作成するには

New-EC2RestoreImageTask コマンドレットを使用します。

```
New-EC2RestoreImageTask `
    -ObjectKey ami-0abcdef1234567890.bin `
    -Bucket amzn-s3-demo-bucket `
    -Name "my-restored-ami"
```

以下は出力の例です。

```
ImageId         : ami-1234567890abcdef0
```

AMI 祖先を使用して AMI のオリジンをトレースする

AMI 祖先は、すべての祖先 AMI の ID とリージョンを返すことで、AMI のオリジンをトレースするのに役立ちます。AMI を作成またはコピーすると、新しい AMI はソース (親) AMI の ID とリージョンを保持します。これにより、ルート AMI に遡るまで AMI のチェーンを追跡することができます。

主な利点

AMI 祖先を使用すると、次のことができます。

- AMI 派生を追跡して、内部ポリシーに準拠していることを確認する。
- 祖先 AMI でセキュリティの問題が見つかったときに、脆弱性を持つ可能性がある AMI を特定する。
- 複数のリージョンにわたる AMI オリジンの可視性を維持する。

トピック

- AMI 祖先の仕組み
- 考慮事項
- AMI 祖先を表示する
- ソースを AMI を識別

AMI 祖先の仕組み

AMI 祖先は、指定された AMI の作成に使用された親 AMI、親の親など、ルート AMI に遡るまで AMI を識別します。処理の流れ

- 各 AMI には、ソース (親) AMI の ID とリージョンが表示されます。
- 選択した AMI から始まる祖先エントリのリストには、各親 AMI が順番に表示されます。
- 祖先エントリのリストは、ルート AMI に到達するまでトレースバックすることができます。ルート AMI は次のいずれかです。
  - 検証済みプロバイダーからのパブリック AMI (所有者エイリアスによって識別されます。これは `amazon` または `aws-marketplace` です)。
  - 祖先が記録されていない AMI。例えば、RegisterImage を使用してスナップショットのセットから直接 AMI を作成する場合、インスタンスから AMI を作成する場合とは異なり、追跡できるソース AMI はありません。
  - ソース AMI が別のパーティションの AMI。
  - リスト内の 50 番目の AMI。祖先リスト内の AMI の最大数は 50 です。

考慮事項

- ソース AMI の ID とリージョンは、CreateImage、CopyImage、または CreateRestoreImageTask を使用して作成された AMI でのみ使用できます。
- CreateImage (インスタンスから AMI を作成) を使用して作成された AMI の場合、ソース AMI ID はインスタンスの起動に使用される AMI の ID です。

- ソース AMI 情報は、次の場合は使用できません。
  - RegisterImage を使用して作成された AMI (スナップショットから作成されているため)。
  - 一部の古い AMI。
- ソース AMI 情報は、次の場合に保持されます。
  - AMI がリージョン間でコピーされている場合。
  - ソース AMI が登録解除 (削除) されている場合。
  - お客様がソース AMI にアクセスできない場合。
- 各祖先リストが 50 AMI に制限されている場合。

AMI 祖先を表示する

次の方法を使用して、AMI の祖先を表示できます。

Console

AMI の祖先を表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. AMI を選択し、[AMI 祖先] タブを選択します。
4. [AMI 祖先エントリ] テーブルには、祖先リスト内のすべての AMI が一覧表示されます。
   - AMI ID – 祖先リスト内の各 AMI の識別子。テーブルの最初のエントリは選択した AMI です。その後に祖先が続きます。
   - ソース AMI ID – AMI ID 列の AMI が作成された AMI の ID。ダッシュ (-) は、AMI 祖先リストの末尾を示します。
   - ソース AMI リージョン – ソース AMI がある AWS リージョン です。
   - 祖先レベル – 祖先リスト内の位置。意味を次に記します。
     - 0 (入力 AMI) は、お客様が祖先を知りたいと考えている、選択した AMI を示します。
     - 数字が増えると、古い祖先が表示されます。
     - ***n*** (元の AMI) はルート AMI を示し、数は祖先リストをどこまで遡るかを示します。
   - 作成日 – AMI が作成された日時。UTC 形式。
   - 所有者エイリアス – AMI 所有者のエイリアス (amazon など)。ダッシュ (-) は、AMI に所有者エイリアスがないことを示します。

AWS CLI

AMI 祖先を表示するには

get-image-ancestry コマンドを使用して、AMI の ID を指定します。

```
aws ec2 get-image-ancestry \
    --image-id ami-1111111111EXAMPLE \
    --region us-east-1
```

以下は出力の例です。出力には、AMI が祖先順に一覧表示されます。最初のエントリは指定された (入力) AMI で、その後に親、親の親などが続き、ルート AMI で終わります。

```
{
    "ImageAncestryEntries": [
        {
            "CreationDate": "2025-01-17T18:37:50.000Z",
            "ImageId": "ami-1111111111EXAMPLE", // Input AMI
            "SourceImageId": "ami-2222222222EXAMPLE",
            "SourceImageRegion": "us-east-1"

        },
        {
            "CreationDate": "2025-01-17T18:37:50.000Z",
            "ImageId": "ami-2222222222EXAMPLE", // Parent AMI
            "SourceImageId": "ami-3333333333EXAMPLE",
            "SourceImageRegion": "us-east-1"
        },
        ...
        {
            "CreationDate": "2025-01-17T18:37:50.000Z",
            "ImageId": "ami-8888888888EXAMPLE", // Root AMI
            "ImageOwnerAlias": "aws-marketplace",
            "SourceImageId": "ami-9999999999EXAMPLE",
            "SourceImageRegion": "us-east-2"
        }
    ]
}
```

PowerShell

AMI 祖先を表示するには

Get-EC2ImageAncestry コマンドレットを使用します。

```
Get-EC2ImageAncestry -ImageId ami-1111111111EXAMPLE
```

以下は出力の例です。出力には、AMI が祖先順に一覧表示されます。最初のエントリは指定された (入力) AMI で、その後に親、親の親などが続き、ルート AMI で終わります。

```
ImageAncestryEntries : {
    @{
        CreationDate = "2025-01-17T18:37:50.000Z"
        ImageId = "ami-1111111111EXAMPLE"    # Input AMI
        SourceImageId = "ami-2222222222EXAMPLE"
        SourceImageRegion = "us-east-1"
    },
    @{
        CreationDate = "2025-01-17T18:37:50.000Z"
        ImageId = "ami-2222222222EXAMPLE"    # Parent AMI
        SourceImageId = "ami-3333333333EXAMPLE"
        SourceImageRegion = "us-east-1"
    },
    ...
    @{
        CreationDate = "2025-01-17T18:37:50.000Z"
        ImageId = "ami-8888888888EXAMPLE"    # Root AMI
        ImageOwnerAlias = "aws-marketplace"
        SourceImageId = "ami-9999999999EXAMPLE"
        SourceImageRegion = "us-east-2"
    }
}
```

ソースを AMI を識別

AMI の作成に使用される直接の親 (ソース) AMI のみを特定する必要がある場合は、次の方法を使用できます。

Console

選択した AMI の作成に使用されるソース AMI を特定するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。

3. AMI を選択して詳細を表示します。

   ソース AMI 情報は、[ソース AMI ID] および [ソース AMI リージョン] のフィールドに表示されます。

AWS CLI

指定した AMI の作成に使用されるソース AMI を特定するには

describe-images コマンドを使用します。

```
aws ec2 describe-images \
    --region us-east-1 \
    --image-ids ami-0abcdef1234567890 \
    --query "Images[].{ID:SourceImageId,Region:SourceImageRegion}"
```

以下は出力の例です。

```
[
    {
        "ID": "ami-0abcdef1234567890",
        "Region": "us-west-2"
    }
]
```

PowerShell

指定した AMI の作成に使用されるソース AMI を特定するには

Get-EC2Image コマンドレットを使用します。

```
Get-EC2Image -ImageId ami-0abcdef1234567890 | Select SourceImageId,
 SourceImageRegion
```

以下は出力の例です。

```
SourceImageId           SourceImageRegion
-------------           -----------------
ami-0abcdef1234567890 us-west-2
```

AMI ウォーターマークを使用して AMI を追跡および識別する

AMI ウォーターマークは、プライベート AMI にアタッチして出所を追跡し、ガバナンスポリシーを適用するための識別子です。ウォーターマークは AMI ライフサイクル全体で保持されます。

- ウォーターマーク付き AMI から起動された実行中のインスタンスから新しい AMI を作成すると、新しい AMI はウォーターマークを継承します。
- ウォーターマーク付き AMI をコピーする場合、コピーにはウォーターマークが保持されます。
- ウォーターマーク付き AMI を別のアカウントと共有した場合、ウォーターマークは受信者に表示されたままになります。

主な利点

- アカウントとリージョン間の出所を追跡する – 承認されたベースイメージから派生する AMI を特定します。
- アカウント全体で関連する AMI をフィルタリングして検索します。
- AMI コンシューマーがプロジェクトまたは組織に関連付けられた信頼された AMI を検出して識別するのに役立ちます。
- ウォーターマークと許可された AMI を組み合わせることによってガバナンスを強制し、承認されたウォーターマークを持つ AMI のみにインスタンスの起動を制限します。

トピック

- AMI ウォーターマークの仕組み
- 必要なアクセス許可
- AMI にウォーターマークをアタッチする
- AMI からウォーターマークをデタッチする
- AMI ウォーターマークを表示する
- ウォーターマークで AMI をフィルタリング

AMI ウォーターマークの仕組み

AMI ウォーターマークは、AMI に付加する構造化された識別子です。ウォーターマークの主な特徴を以下のとおりです。

- 永続化 – AMI にウォーターマークを添付すると、そのウォーターマークはすべての派生 AMI にも引き継がれます。
- 所有者のみ – AMI 所有者のみが AMI にウォーターマークをアタッチできます。
- 誰でも表示可能 – AMI にアクセスできるユーザーは誰でもウォーターマークを表示できます。
- 5 の制限 – AMI には最大 5 つのウォーターマークを含めることができます。
- パブリック AMI では使用不可能 – ウォーターマークがある場合、ウォーターマークをパブリック AMI にアタッチすることや AMI をパブリックにすることはできません。
- フィルタリング可能 – `describe-images` を使用すると、AMI ウォーターマークでフィルタリングできます。

ウォーターマークの形式

ウォーターマークは、次のフィールドを持つ構造化オブジェクトです。

- `WatermarkKey` – ウォーターマークの一意の識別子。*`account-id`*`:`*`watermark-name`* で構成されます。account ID は、REST API 所有者の 12 桁の AWS account ID です。ウォーターマークの名称部分は、お客様が指定した名前です。
- `SourceImageRegion` – 最初にウォーターマークをアタッチした AMI のリージョン。
- `SourceImageId` – ウォーターマークを最初にアタッチした AMI。
- `SourceImageCreationDate` – ウォーターマークを最初にアタッチした AMI の作成日。
- `WatermarkCreationTime` – ウォーターマークを適用したときのタイムスタンプ。

ウォーターマーク名は 3 ~ 128 文字で、英数字、括弧 (())、角括弧 ([])、スペース、ピリオド (.)、スラッシュ (/)、ダッシュ (-)、一重引用符 (')、アットマーク (@)、またはアンダースコア (_) を含めることができます。

必要なアクセス許可

AMI ウォーターマークを使用するには、次の IAM アクセス許可が必要です。

- `ec2:AttachImageWatermark` – AMI にウォーターマークをアタッチします。
- `ec2:DetachImageWatermark` – AMI からウォーターマークをデタッチします。
- `ec2:DescribeImages` – AMI のウォーターマークを表示します。

AMI にウォーターマークをアタッチする

コンソール、AWS CLI、または PowerShell を使用して、AMI にウォーターマークをアタッチできます。

Console

AMI にウォーターマークをアタッチするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. AMI を選択します。
4. 詳細 タブの ウォーターマーク セクションで、ウォーターマークの管理 を選択します。
5. ウォーターマーク名を入力し、アタッチ を選択します。

AWS CLI

AMI にウォーターマークをアタッチするには

attach-image-watermark コマンドを使用します。

```
aws ec2 attach-image-watermark \
    --image-id ami-1111111111EXAMPLE \
    --image-watermark-name "prod-baseline"
```

以下は出力の例です。

```
{
    "WatermarkKey": "123456789012:prod-baseline"
}
```

PowerShell

AMI にウォーターマークをアタッチするには

Add-EC2ImageWatermark コマンドレットを使用します。

```
Add-EC2ImageWatermark `
    -ImageId ami-1111111111EXAMPLE `
```

```
    -ImageWatermarkName "prod-baseline"
```

1 つの AMI に最大 5 つのウォーターマークをアタッチできます。

AMI からウォーターマークをデタッチする

コンソール、AWS CLI、または PowerShell を使用して、AMI からウォーターマークをデタッチできます。

Console

AMI からウォーターマークをデタッチするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. AMI を選択します。
4. 詳細 タブの ウォーターマーク セクションで、ウォーターマークの管理 を選択します。
5. 削除するウォーターマークを選択し、削除 を選択します。

AWS CLI

AMI からウォーターマークをデタッチするには

detach-image-watermark コマンドを使用します。

```
aws ec2 detach-image-watermark \
    --image-id ami-1111111111EXAMPLE \
    --image-watermark-key "111122223333:prod-baseline"
```

PowerShell

AMI からウォーターマークをデタッチするには

Remove-EC2ImageWatermark cmdlet を使用します。

```
Remove-EC2ImageWatermark `
    -ImageId ami-1111111111EXAMPLE `
    -ImageWatermarkKey "111122223333:prod-baseline"
```

> **Note**
>
> AMI からウォーターマークをデタッチしても、そのウォーターマークが既に存在する派生 AMI からは削除されません。ウォーターマークが永続性を維持するには、ウォーターマークを管理する必要がある信頼できる管理者にのみ `ec2:DetachImageWatermark` のアクセス許可を付与します。

AMI ウォーターマークを表示する

コンソール、AWS CLI、または PowerShell を使用して、AMI のウォーターマークを表示できます。

Console

AMI のウォーターマークを表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. AMI を選択します。
4. 詳細 タブのウォーターマークセクションで ウォーターマーク を表示します。

AWS CLI

AMI のウォーターマークを表示するには

describe-images コマンドを使用します。

```
aws ec2 describe-images \
    --image-ids ami-046863d776a820ccd \
    --region us-east-1
```

レスポンスには、各 AMI に対応する `ImageWatermarks` の 配列が含まれます。

```
{
    "Images": [
        {
            "ImageId": "ami-046863d776a820ccd",
            "Public": false,
            "OwnerId": "123456789012",
            ...
```

```
            "ImageWatermarks": [
                {
                    "WatermarkKey": "111122223333:prod-baseline",
                    "Region": "us-east-1",
                    "SourceImageId": "ami-0b752bf1df193a6c4",
                    "SourceImageCreationDate": "2024-07-10T08:15:00",
                    "CreationDate": "2024-07-12T14:30:00"
                },
                {
                    "WatermarkKey": "222222222222:security-approved",
                    "Region": "eu-north-1",
                    "SourceImageId": "ami-12345678",
                    "SourceImageCreationDate": "2024-06-01T10:00:00",
                    "CreationDate": "2024-06-05T09:45:00"
                }
            ]
        }
    ]
}
```

PowerShell

AMI のウォーターマークを表示するには

Get-EC2Image コマンドレットを使用します。

```
(Get-EC2Image -ImageId ami-046863d776a820ccd).ImageWatermarks
```

ウォーターマークで AMI をフィルタリング

コンソール、AWS CLI、PowerShell を使用して AMI をウォーターマークでフィルタリングできます。

Console

AMI ウォーターマークでフィルタリングするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. 検索バーで、ウォーターマークキー フィルターを選択し、ウォーターマークキーの値を入力します。

AWS CLI

AMI ウォーターマークでフィルタリングするには

`image-watermark-key` フィルターで「describe-images」コマンドを使用します。

```
aws ec2 describe-images \
    --filters "Name=image-watermark-key,Values=111122223333:prod-baseline"
```

これにより、コピー操作によって継承された派生 AMI など、指定されたウォーターマークを持つにアクセスできるすべての AMI が返されます。

PowerShell

AMI ウォーターマークでフィルタリングするには

`-Filter` パラメータで Get-EC2Image コマンドレットを使用します。

```
Get-EC2Image `
    -Filter @{Name="image-watermark-key"; Values="111122223333:prod-baseline"}
```

これにより、コピー操作によって継承された派生 AMI など、指定されたウォーターマークを持つにアクセスできるすべての AMI が返されます。

AMI の使用を管理およびモニタリングする

AWS には、AMI の使用状況を効果的に管理およびモニタリングするのに役立ついくつかの機能が用意されています。共有 AMI を使用しているアカウントの追跡、AMI が最後に使用された日時の特定、AWS アカウント で特定の AMI を参照しているリソースの検出を行うことができます。

AMI の使用状況を管理およびモニタリングするための機能の概要を次の表に示します。

機能	ユースケース	主な利点
AMI の使用状況レポート	AMI を使用している AWS アカウントと、各 AMI の使用量を可視化します。	• AMI の登録解除または無効化を安全に行うことができるように、AMI を参照している AWS アカウントと

機能	ユースケース	主な利点
		リソースタイプを特定します。 • ストレージコストを削減するために登録を解除する未使用の AMI を特定します。 • 最も使用されている AMI を特定します。
最後に使用された追跡	AMI が最後に使用された日時を確認します。	• AMI の登録解除を安全に行うことができるように未使用の AMI を特定します。 • ストレージコストを削減するために登録を解除する未使用の AMI を特定します。
AMI の参照チェック	AWS リソースが最新の準拠 AMI を使用していることを確認します。	• アカウントでの AMI の使用を監査します。 • 特定の AMI が参照されている場所を確認します。 • リソースを更新して最新の AMI を参照することでコンプライアンスを維持します。

トピック

- AMI の使用状況を表示する
- Amazon EC2 AMI が最後に使用された日時を確認する

- 指定した AMI を参照しているリソースを識別します。

AMI の使用状況を表示する

Amazon マシンイメージ (AMI) を他の AWS アカウント (特定の AWS アカウント、組織、組織単位 (OU)、またはパブリック) と共有する場合、AMI の使用状況レポートを作成することで、これらの AMI がどのように使用されているかを確認できます。レポートは、以下を可視化します。

- EC2 インスタンスまたは起動テンプレートで AMI を使用している AWS アカウント
- 各 AMI を参照している EC2 インスタンスまたは起動テンプレートの数

AMI の使用状況レポートは、以下を行って AMI をより効果的に管理するのに役立ちます。

- AMI の登録解除または無効化を安全に行うことができるように、AMI を参照している AWS アカウントとリソースタイプを特定します。
- ストレージコストを削減するために登録を解除する未使用の AMI を特定します。
- 最も使用されている AMI を特定します。

内容

- AMI 使用状況レポートの仕組み
- AMI の使用状況レポートを作成する
- AMI の使用状況レポートを表示する
- AMI の使用状況レポートを削除する
- レポートに関するクォータ

AMI 使用状況レポートの仕組み

AMI 使用状況レポートを作成するときは、以下を指定します。

- レポート対象の AMI。
- チェックする AWS アカウント (特定のアカウントまたはすべてのアカウント)。
- チェックするリソースタイプ (EC2 インスタンス、起動テンプレート、またはその両方)。
- 起動テンプレートの場合、チェックするバージョンの数 (デフォルトでは 20 の最新バージョン)。

Amazon EC2 によって AMI ごとに個別のレポートが作成されます。各レポートは以下を提供します。

- AMI を使用している AWS アカウントのリスト。
- アカウントごとのリソースタイプ別に AMI を参照しているリソースの数。起動テンプレートの場合、起動テンプレートの複数のバージョンで AMI が参照されているときはカウントは 1 になります。

> **Important**
>
> AMI の使用状況レポートを生成する場合、最新のアクティビティが含まれていない可能性があります。過去 24 時間のインスタンスアクティビティと、過去数日間の起動テンプレートアクティビティがレポートに表示されないことがあります。

Amazon EC2 は、作成から 30 日後にレポートを自動的に削除します。EC2 コンソールからレポートをダウンロードしてローカルに保持できます。

AMI の使用状況レポートを作成する

AMI がどのように使用されているかを確認するには、まず AMI の使用状況レポートを作成し、レポートするアカウントとリソースタイプを指定する必要があります。レポートが作成されたら、レポートの内容を表示できます。EC2 コンソールからレポートをダウンロードすることもできます。

Console

AMI の使用状況レポートを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. AMI を選択し、[アクション]、[AMI の使用状況]、[自分の AMI 使用状況を表示] の順に選択します。
4. [AMI 使用状況レポートの作成] ページで、次の操作を行います。
   a. [リソースタイプ] で、レポートする 1 つ以上のリソースタイプを選択します。
   b. [アカウント ID] で、次のいずれかを実行します。

- [アカウント ID を指定] を選択し、レポートする各アカウントの [アカウント ID を追加] を選択します。
- すべてのアカウントでレポートするには、[すべてのアカウントを含める] を選択します。

c. [AMI 使用状況レポートの作成] を選択します。

5. AMI ページで、[AMI の使用状況] タブを選択します。
6. 詳細を表示するレポート ID を選択します。

AWS CLI

複数のアカウントの AMI 使用状況レポートを作成するには

次の必須パラメータで create-image-usage-report コマンドを使用します。

- `--image-id` – レポートする AMI の ID。
- `--resource-types` – チェックするリソースのタイプ。次の例では、チェックするリソースタイプは EC2 インスタンスと起動テンプレートです。さらに、チェックする起動テンプレートのバージョンの数も指定されています (`version-depth=100`)。

複数の特定のアカウントをレポートするには、 `--account-ids` パラメータを使用してレポートする各アカウントの ID を指定します。

```
aws ec2 create-image-usage-report \
    --image-id ami-0abcdef1234567890 \
    --account-ids 111122223333 444455556666 123456789012 \
    --resource-types ResourceType=ec2:Instance \
      'ResourceType=ec2:LaunchTemplate,ResourceTypeOptions=[{OptionName=version-
depth,OptionValues=100}]'
```

すべてのアカウントの AMI 使用状況レポートを作成するには

指定した AMI を使用しているすべてのアカウントをレポートするには、同じコマンドを使用しますが、`--account-ids` パラメータを省略します。

```
aws ec2 create-image-usage-report \
    --image-id ami-0abcdef1234567890 \
    --resource-types ResourceType=ec2:Instance \
```

```
      'ResourceType=ec2:LaunchTemplate,ResourceTypeOptions=[{OptionName=version-
depth,OptionValues=100}]'
```

以下は出力の例です。

```
{
    "ReportId": "amiur-00b877d192f6b02d0"
}
```

レポート作成ステータスをモニタリングするには

describe-image-usage-reports コマンドを使用して、レポート ID を指定します。

```
aws ec2 describe-image-usage-reports --report-ids amiur-00b877d192f6b02d0
```

以下は出力の例です。State フィールドの初期値は pending です。レポートエントリを表示するには、状態が available である必要があります。

```
{
    "ImageUsageReports": [
        {
            "ImageId": "ami-0e9ae3dc21c2b3a64",
            "ReportId": "amiur-abcae3dc21c2b3999",
            "ResourceTypes": [
                {"ResourceType": "ec2:Instance"}
            ],
            "State": "pending",
            "CreationTime": "2025-09-29T13:27:12.322000+00:00",
            "ExpirationTime": "2025-10-28T13:27:12.322000+00:00"
        }
    ]
}
```

PowerShell

複数のアカウントの AMI 使用状況レポートを作成するには

以下の必須パラメータで New-EC2ImageUsageReport コマンドレットを使用します。

- -ImageId – レポートする AMI の ID。

- -ResourceType – チェックするリソースのタイプ。次の例では、チェックするリソースタイプは EC2 インスタンスと起動テンプレートです。さらに、チェックする起動テンプレートのバージョンの数も指定されています ('version-depth' = *100*)。

複数の特定のアカウントをレポートするには、 -AccountId パラメータを使用してレポートする各アカウントの ID を指定します。

```
New-EC2ImageUsageReport `
    -ImageId ami-0abcdef1234567890 `
    -AccountId 111122223333 444455556666 123456789012 `
    -ResourceType @(
        @{ResourceType = 'ec2:Instance'},
        @{ResourceType = 'ec2:LaunchTemplate'ResourceTypeOptions = @{'version-depth'
 = 100}
        })
```

すべてのアカウントの AMI 使用状況レポートを作成するには

指定した AMI を使用しているすべてのアカウントをレポートするには、同じコマンドを使用しますが、-AccountId パラメータを省略します。

```
New-EC2ImageUsageReport `
    -ImageId ami-0abcdef1234567890 `
    -ResourceType @(
        @{ResourceType = 'ec2:Instance'},
        @{ResourceType = 'ec2:LaunchTemplate'ResourceTypeOptions = @{'version-depth'
 = 100}
        })
```

以下は出力の例です。

```
ReportId
--------
amiur-00b877d192f6b02d0
```

レポート作成ステータスをモニタリングするには

Get-EC2ImageUsageReport コマンドを使用してレポート ID を指定します。

```
Get-EC2ImageUsageReport -ReportId amiur-00b877d192f6b02d0
```

以下は出力の例です。State フィールドの初期値は pending です。レポートエントリを表示するには、状態が available である必要があります。

```
ImageUsageReports
-----------------
{@{ImageId=ami-0e9ae3dc21c2b3a64; ReportId=amiur-abcae3dc21c2b3999;
 ResourceTypes=System.Object[]; State=pending; CreationTime=2025-09-29;
 ExpirationTime=2025-10-28}}
```

AMI の使用状況レポートを表示する

AMI に対して過去 30 日間に作成したすべての使用状況レポートを表示できます。Amazon EC2 は、作成から 30 日後にレポートを自動的に削除します。

レポートごとに、AMI を使用している AWS アカウントを表示できます。アカウントごとに、AMI を参照しているリソースの数をリソースタイプ別に確認できます。レポートの作成が開始されたときを表示することもできます。この情報は、レポートが完了 (コンソール) または available (AWS CLI) の場合に表示されます。

> **Important**
>
> AMI の使用状況レポートを生成する場合、最新のアクティビティが含まれていない可能性があります。過去 24 時間のインスタンスアクティビティと、過去数日間の起動テンプレートアクティビティがレポートに表示されないことがあります。

Console

AMI の使用状況レポートを表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. AMI を選択します。
4. [My usage reports] タブを選択します。

   レポートリストには以下が表示されます。

   - 選択した AMI に対して過去 30 日間に生成されたすべてのレポート。
   - レポートごとに、[レポート開始時間] 列にレポートが作成された日付が表示されます。

5. レポートの ID を選択すると、その内容が表示されます。
6. AMI の詳細ページの [My usage reports] タブに戻るには、[この AMI のすべてのレポートを表示] を選択します。

AWS CLI

指定した AMI のすべての AMI 使用状況レポートを一覧表示するには

describe-image-usage-reports コマンドを使用して、レポートのリストを取得する AMI の ID を指定します。

```
aws ec2 describe-image-usage-reports --image-ids ami-0abcdef1234567890
```

以下は出力の例です。スキャンされたリソースタイプ、レポートの作成日と有効期限とともに各レポート ID が一覧表示されます。この情報を使用して、エントリを表示するレポートを識別できます。

```
{
  "ImageUsageReports": [
    {
      "ImageId": "ami-0abcdef1234567890",
      "ReportId": "amiur-1111111111111111",
      "ResourceTypes": [
        {
          "ResourceType": "ec2:Instance"
        }
      ],
      "State": "available",
      "CreationTime": "2025-09-29T13:27:12.322000+00:00",
      "ExpirationTime": "2025-10-28T13:27:12.322000+00:00",
      "Tags": []
    },
    {
      "ImageId": "ami-0abcdef1234567890",
      "ReportId": "amiur-22222222222222222",
      "ResourceTypes": [
        {
          "ResourceType": "ec2:Instance"
        },
        {
          "ResourceType": "ec2:LaunchTemplate"
```

```
        }
      ],
      "State": "available",
      "CreationTime": "2025-10-01T13:27:12.322000+00:00",
      "ExpirationTime": "2025-10-30T13:27:12.322000+00:00",
      "Tags": []
    }
  ],
  "NextToken": "opaque"
}
```

指定した AMI の AMI 使用状況レポートの内容を表示するには

describe-image-usage-report-entries コマンドを使用して AMI の ID を指定します。指定した AMI のすべてのレポートが返され、AMI を使用したアカウントとそのリソース数が表示されます。

```
aws ec2 describe-image-usage-report-entries --image-ids ami-0abcdef1234567890
```

以下は出力の例です。

```
{
  "ImageUsageReportEntries": [
    {
      "ImageId": "ami-0abcdef1234567890",
      "ResourceType": "ec2:Instance",
      "AccountId": "123412341234",
      "UsageCount": 15,
      "ReportCreationTime": "2025-09-29T13:27:12.322000+00:00",
      "ReportId": "amiur-1111111111111111"
    },
    {
      "ImageId": "ami-0abcdef1234567890",
      "ResourceType": "ec2:Instance",
      "AccountId": "123412341234",
      "UsageCount": 2,
      "ReportCreationTime": "2025-10-01T13:27:12.322000+00:00",
      "ReportId": "amiur-2222222222222222"
    },
    {
      "ImageId": "ami-0abcdef1234567890",
      "ResourceType": "ec2:Instance",
      "AccountId": "001100110011",
      "UsageCount": 39,
```

```
      "ReportCreationTime": "2025-10-01T13:27:12.322000+00:00",
      "ReportId": "amiur-22222222222222222"
    }
  ],
  "NextToken": "opaque"
}
```

指定したレポートの AMI 使用状況レポートの内容を表示するには

describe-image-usage-report-entries コマンドを使用してレポートの ID を指定します。指定したレポートのすべてのエントリが返され、AMI を使用したアカウントとそのリソース数が表示されます。

```
aws ec2 describe-image-usage-report-entries --report-ids amiur-11111111111111111
```

以下は出力の例です。

```
{
  "ImageUsageReportEntries": [
    {
      "ImageId": "ami-0abcdef1234567890",
      "ResourceType": "ec2:Instance",
      "AccountId": "123412341234",
      "UsageCount": 15,
      "ReportCreationTime": "2025-09-29T13:27:12.322000+00:00",
      "ReportId": "amiur-11111111111111111"
    },
    {
      "ImageId": "ami-0abcdef1234567890",
      "ResourceType": "ec2:LaunchTemplate",
      "AccountId": "123412341234",
      "UsageCount": 4,
      "ReportCreationTime": "2025-09-29T13:27:12.322000+00:00",
      "ReportId": "amiur-11111111111111111"
    },
    {
      "ImageId": "ami-0abcdef1234567890",
      "ResourceType": "ec2:LaunchTemplate",
      "AccountId": "001100110011",
      "UsageCount": 2,
      "ReportCreationTime": "2025-09-29T13:27:12.322000+00:00",
      "ReportId": "amiur-11111111111111111"
```

```
    }
  ],
  "NextToken": "opaque"
}
```

PowerShell

指定した AMI のすべての AMI 使用状況レポートを一覧表示するには

Get-EC2ImageUsageReport コマンドレットを使用して、レポートのリストを取得する AMI の ID を指定します。

```
Get-EC2ImageUsageReport -ImageId ami-0abcdef1234567890
```

以下は出力の例です。スキャンされたリソースタイプ、レポートの作成日と有効期限とともに各レポート ID が一覧表示されます。この情報を使用して、エントリを表示するレポートを識別できます。

```
@{
    ImageUsageReports = @(
        @{
            ImageId = "ami-0abcdef1234567890"
            ReportId = "amiur-1111111111111111"
            ResourceTypes = @(
                @{
                    ResourceType = "ec2:Instance"
                }
            )
            State = "available"
            CreationTime = "2025-09-29T13:27:12.322000+00:00"
            ExpirationTime = "2025-10-28T13:27:12.322000+00:00"
        },
        @{
            ImageId = "ami-0abcdef1234567890"
            ReportId = "amiur-2222222222222222"
            ResourceTypes = @(
                @{
                    ResourceType = "ec2:Instance"
                }
            )
            State = "available"
            CreationTime = "2025-09-30T13:27:12.322000+00:00"
```

```
            ExpirationTime = "2025-10-29T13:27:12.322000+00:00"
        },
        @{
            ImageId = "ami-0abcdef1234567890"
            ReportId = "amiur-3333333333333333"
            ResourceTypes = @(
                @{
                    ResourceType = "ec2:Instance"
                }
            )
            State = "available"
            CreationTime = "2025-10-01T13:27:12.322000+00:00"
            ExpirationTime = "2025-10-30T13:27:12.322000+00:00"
        }
    )
    NextToken = "opaque"
}
```

指定した AMI の AMI 使用状況レポートの内容を表示するには

Get-EC2ImageUsageReportEntry コマンドレットを使用して AMI の ID を指定します。指定した AMI のすべてのレポートが返され、AMI を使用したアカウントとそのリソース数が表示されます。

```
Get-EC2ImageUsageReportEntry -ImageId ami-0abcdef1234567890
```

以下は出力の例です。

```
ImageUsageReportEntries : {@{
    ImageId = "ami-0abcdef1234567890"
    ResourceType = "ec2:Instance"
    AccountId = "123412341234"
    UsageCount = 15
    ReportCreationTime = "2025-09-29T13:27:12.322000+00:00"
    ReportId = "amiur-1111111111111111"
    }, @{
    ImageId = "ami-0abcdef1234567890"
    ResourceType = "ec2:Instance"
    AccountId = "123412341234"
    UsageCount = 7
    ReportCreationTime = "2025-09-30T13:27:12.322000+00:00"
    ReportId = "amiur-2222222222222222"
```

```
    }...}
NextToken : opaque
```

指定したレポートの AMI 使用状況レポートの内容を表示するには

Get-EC2ImageUsageReportEntry コマンドレットを指定してレポートの ID を指定します。指定したレポートのすべてのエントリが返され、AMI を使用したアカウントとそのリソース数が表示されます。

```
Get-EC2ImageUsageReportEntry -ReportId amiur-1111111111111111
```

以下は出力の例です。

```
ImageUsageReportEntries : {@{
    ImageId = "ami-0abcdef1234567890"
    ResourceType = "ec2:Instance"
    AccountId = "123412341234"
    UsageCount = 15
    ReportCreationTime = "2025-09-29T13:27:12.322000+00:00"
    ReportId = "amiur-1111111111111111"
    }, @{
    ImageId = "ami-0abcdef1234567890"
    ResourceType = "ec2:LaunchTemplate"
    AccountId = "123412341234"
    UsageCount = 4
    ReportCreationTime = "2025-09-29T13:27:12.322000+00:00"
    ReportId = "amiur-1111111111111111"
    }, @{
    ImageId = "ami-0abcdef1234567890"
    ResourceType = "ec2:LaunchTemplate"
    AccountId = "************"
    UsageCount = 2
    ReportCreationTime = "2025-09-29T13:27:12.322000+00:00"
    ReportId = "amiur-1111111111111111"
    }}
NextToken : opaque
```

AMI の使用状況レポートを削除する

Amazon EC2 は、レポートの作成から 30 日後に自動的にレポートを削除します。その前に手動で削除することができます。

Console

AMI の使用状況レポートを削除するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. AMI を選択します。
4. [AMI の使用状況] タブを選択します。
5. 削除するレポートの横にあるオプションボタンを選択して、[削除] を選択します。

AWS CLI

AMI の使用状況レポートを削除するには

delete-image-usage-report コマンドを使用して、レポートの ID を指定します。

```
aws ec2 delete-image-usage-report --report-id amiur-0123456789abcdefg
```

PowerShell

AMI の使用状況レポートを削除するには

Remove-EC2ImageUsageReport コマンドレットを使用して、レポートの ID を指定します。

```
Remove-EC2ImageUsageReport -ReportId amiur-0123456789abcdefg
```

レポートに関するクォータ

AMI 使用状況レポートの作成には、次のクォータが適用されます。AWS リージョン ごとにクォータが適用されます。

説明	クォータ
AWS アカウント ごとの進行中の (pending) AMI 使用状況レポート	2,000
AMI ごとの進行中の (pending) AMI 使用状況レポート	1

Amazon EC2 AMI が最後に使用された日時を確認する

Amazon EC2 は、インスタンスを起動するために AMI が最後に使用された日時を自動的に追跡します。長期間インスタンスの起動に使用されていない AMI がある場合は、その AMI を登録解除または非推奨化の候補とするかどうかを検討してください。

考慮事項

- インスタンスを起動するために AMI が使用される場合、その発生から 24 時間経過した後に報告されます。
- 最終起動時間を取得するには、AMI の所有者である必要があります。
- AMI 使用状況データは、2017 年 4 月以降利用可能です。

Console

AMI の最終起動時間を表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左のナビゲーションペインで [AMI] を選択してください。
3. フィルターバーから、[Owned by me] (自己所有) を選択します。
4. AMI のチェックボックスをオンにします。
5. [詳細] タブで [最終起動時刻] を見つけます。

AWS CLI

AMI を記述して最終起動時間を表示するには

describe-images コマンドを使用します。出力に `LastLaunchedTime` が含まれていない場合は、自分がその AMI を所有しているか確認します。

```
aws ec2 describe-images \
    --image-id ami-0abcdef1234567890 \
    --query Images[].LastLaunchedTime \
    --output text
```

以下は出力の例です。

```
2025-02-17T20:22:19Z
```

AMI の最終起動時間の属性を表示するには

describe-image-attribute コマンドを使用します。ユーザーは、指定した AMI の所有者でなければなりません。

```
aws ec2 describe-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --attribute lastLaunchedTime \
    --query LastLaunchedTime.Value \
    --output text
```

以下は出力の例です。

```
2025-02-17T20:22:19Z
```

PowerShell

AMI を記述して最終起動時間を表示するには

Get-EC2Image コマンドレットを使用します。出力に LastLaunchedTime が含まれていない場合は、自分がその AMI を所有しているか確認します。

```
(Get-EC2Image -ImageId ami-0abcdef1234567890).LastLaunchedTime
```

以下は出力の例です。

```
2025-02-17T20:22:19Z
```

AMI の最終起動時間の属性を表示するには

Get-EC2ImageAttribute コマンドレットを使用します。ユーザーは、指定した AMI の所有者でなければなりません。

```
(Get-EC2ImageAttribute `
    -ImageId ami-0abcdef1234567890 `
    -Attribute LastLaunchedTime).LastLaunchedTime
```

以下は出力の例です。

```
2025-02-17T20:22:19Z
```

指定した AMI を参照しているリソースを識別します。

指定した Amazon マシンイメージ (AMI) を参照する AWS リソースは、AMI がパブリックかプライベートか、または誰が所有しているかに関係なく識別できます。この可視性により、リソースが最新の準拠 AMI を使用することが保証されます。

主な利点

AMI の参照をチェックすると、次のことに役立ちます。

- アカウントでの AMI の使用を監査します。
- 特定の AMI が参照されている場所を確認します。
- リソースを更新して最新の AMI を参照することでコンプライアンスを維持します。

トピック

- サポートされているリソース
- AMI 参照チェックの仕組み
- 必要な IAM 許可
- AMI の参照を確認する手順

サポートされているリソース

AMI の参照は以下でチェックできます。

- EC2 インスタンス
- 起動テンプレート
- SSM パラメータ
- Image Builder のイメージレシピ
- Image Builder コンテナレシピ

AMI 参照チェックの仕組み

基本オペレーション

AMI 参照チェックを実行する際は、以下の手順を実行します。

- 確認する AMI を指定します。
- スキャンするリソースタイプを選択します。
- 指定した AMI を参照するリソースのリストを受け取ります。

リソースタイプの選択

コンソールで、スキャンするリソースタイプを選択します。

CLI では、次の CLI パラメータのいずれかまたは両方を使用して、スキャンするリソースタイプを指定します。

- `IncludeAllResourceTypes`: サポートされているすべてのリソースタイプをスキャンします。
- `ResourceTypes`: 指定したリソースタイプをスキャンします。

応答の範囲設定

`ResourceTypes` パラメータを使用して `ResourceTypeOptions` の値をカスタマイズすることで、EC2 インスタンスと起動テンプレートの応答範囲を指定できます。コンソールと `IncludeAllResourceTypes` パラメータはどちらもデフォルトのオプション値を使用します。`ResourceTypes` と `IncludeAllResourceTypes` を一緒に使用すると、`ResourceTypes` オプション値がデフォルトよりも優先されます。

デフォルト値を以下に示します。

リソースタイプ	範囲オプション (**`OptionName`**)	目的	**`OptionValue`** とコンソールのデフォルト値

EC2 インスタンス	`state-name`	インスタンスの状態でフィルタリングする	`pending`、`running`、`shutt`
			`down` 、`terminate`
			`d` 、`stopping`、`stopped`
			(すべての状態)

起動テンプレート	`version-depth`	確認する起動テンプレートのバージョンの数を指定します (最	`10` (最新バージョン)

リソースタイプ	範囲オプション (**OptionName**)	目的	**OptionValue** とコンソールのデフォルト値
		新バージョンから開始)。	

必要な IAM 許可

DescribeImageReferences API を使用して、指定された AMI を参照するリソースを特定するには、リソースを記述するために次の IAM アクセス許可が必要です。

- `ec2:DescribeInstances`
- `ec2:DescribeLaunchTemplates`
- `ec2:DescribeLaunchTemplateVersions`
- `ssm:DescribeParameters`
- `ssm:GetParameters`
- `imagebuilder:ListImageRecipes`
- `imagebuilder:ListContainerRecipes`
- `imagebuilder:GetContainerRecipe`

DescribeImageReferences API を使用するための IAM ポリシーの例

次のポリシー例では、DescribeImageReferences API を使用するアクセス許可を付与します。これには、EC2 インスタンス、起動テンプレート、Systems Manager パラメータ、Image Builder イメージレシピ、Image Builder コンテナレシピを記述するアクセス許可が含まれます。

JSON

```
{
 "Version":"2012-10-17",
 "Statement": [
  {
   "Effect": "Allow",
   "Action": "ec2:DescribeImageReferences",
   "Resource": "*"
  },
```

```
  {
   "Effect": "Allow",
   "Action": [
    "ec2:DescribeInstances",
    "ec2:DescribeLaunchTemplates",
    "ec2:DescribeLaunchTemplateVersions",
    "ssm:DescribeParameters",
    "ssm:GetParameters",
    "imagebuilder:ListImageRecipes",
    "imagebuilder:ListContainerRecipes",
    "imagebuilder:GetContainerRecipe"
   ],
   "Resource": "*",
   "Condition": {
    "ForAnyValue:StringEquals": {
     "aws:CalledVia": [
      "ec2-images.amazonaws.com"
     ]
    }
   }
  }
 ]
}
```

Important

自分でポリシーを作成する代わりに、AWS マネージドポリシー AmazonEC2ImageReferencesAccessPolicy を使用することを強くお勧めします。必要なアクセス許可のみを提供するカスタム IAM ポリシーを作成するには時間と専門知識が必要で、新しいリソースタイプが利用可能になったときに更新が必要になります。AmazonEC2ImageReferencesAccessPolicy マネージドポリシー:

- DescribeImageReferences API を使用するために必要なすべてのアクセス許可を付与します (EC2 インスタンス、起動テンプレート、Systems Manager パラメータ、Image Builder コンテナとイメージレシピを記述するためのアクセス許可が含まれます)。
- 利用可能になった新しいリソースタイプを自動的にサポートします (IncludeAllResourceTypes パラメータを使用する場合、特に重要です)。

`AmazonEC2ImageReferencesAccessPolicy` ポリシーは IAM アイデンティティ (ユーザー、グループ、ロール) にアタッチできます。
このポリシーに含まれるアクセス許可を確認するには、「AWS マネージドポリシーリファレンス」の「AmazonEC2ImageReferencesAccessPolicy」を参照してください。

AMI の参照を確認する手順

次の手順を使用して、指定した AMI を参照している AWS リソースを特定します。

Console

指定した AMI を参照するリソースを識別するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. 参照をチェックする 1 つまたは複数の AMI を選択します。
4. [アクション]、[AMI の使用状況]、[参照先リソースを表示] の順に選択します。
5. [選択した AMI を参照しているリソースを表示] ページで次の操作を実行します。
   a. [リソースタイプ] で 1 つ以上のリソースタイプを選択します。
   b. [リソースの表示] を選択します。
6. [選択した AMI を参照するリソース] セクションが表示されます。リストには、指定された AMI を参照するリソースが表示されます。各行には、以下に関する情報が表示されます。
   - [AMI ID] – 参照されている AMI の ID。
   - [リソースタイプ] – AMI を参照しているリソースのリソースタイプ。
   - [リソース ID] – AMI を参照しているリソースの ID。

AWS CLI

特定のリソースタイプの AMI 参照を確認するには

`--resource-types` パラメータで describe-image-references コマンドを使用します。次の例では、EC2 インスタンス (インスタンスの状態による範囲指定)、起動テンプレート (最新の 20 個

の起動テンプレートバージョンによる範囲指定)、その他の特定のリソースタイプをチェックします。

```
aws ec2 describe-image-references \
    --image-ids ami-0abcdef1234567890 ami-1234567890abcdef0 \
    --resource-types \
        'ResourceType=ec2:Instance,ResourceTypeOptions=[{OptionName=state-
name,OptionValues=[running,pending]}]' \
        'ResourceType=ec2:LaunchTemplate,ResourceTypeOptions=[{OptionName=version-
depth,OptionValues=[20]}]' \
        'ResourceType=ssm:Parameter' \
        'ResourceType=imagebuilder:ImageRecipe' \
        'ResourceType=imagebuilder:ContainerRecipe'
```

以下は出力の例です。

```
{
    "ImageReferences": [
        {
            "ImageId": "ami-0abcdef1234567890",
            "ResourceType": "ec2:Instance",
            "Arn": "arn:aws:ec2:us-east-1:123456789012:instance/i-1234567890abcdef0"
        },
        {
            "ImageId": "ami-1234567890abcdef0",
            "ResourceType": "ec2:LaunchTemplate",
            "Arn": "arn:aws:ec2:us-east-1:123456789012:launch-template/
lt-1234567890abcdef0"
        }
    ]
}
```

サポートされているすべてのリソースタイプの AMI 参照を確認するには

`--include-all-resource-types` パラメータで describe-image-references コマンドを使用します。

```
aws ec2 describe-image-references \
    --image-ids ami-0abcdef1234567890 ami-1234567890abcdef0 \
    --include-all-resource-types
```

サポートされているすべてのリソースタイプと特定のオプションの AMI 参照を確認するには

`--include-all-resource-types` と `--resource-types` の両方のパラメータで describe-image-references コマンドを使用します。この例では、実行中または保留中のインスタンスに対する EC2 インスタンスの応答範囲を指定しながら、すべてのリソースタイプをチェックします。

```
aws ec2 describe-image-references \
    --image-ids ami-0abcdef1234567890 ami-1234567890abcdef0 \
    --include-all-resource-types \
    --resource-types
 'ResourceType=ec2:Instance,ResourceTypeOptions=[{OptionName=state-
name,OptionValues=[running,pending]}]'
```

PowerShell

特定のリソースタイプの AMI 参照を確認するには

`-ResourceType` パラメータで Get-EC2ImageReference コマンドレットを使用します。次の例では、EC2 インスタンス (インスタンスの状態による範囲指定)、起動テンプレート (最新の 20 個の起動テンプレートバージョンによる範囲指定)、その他の特定のリソースタイプをチェックします。

```
Get-EC2ImageReference `
    -ImageId 'ami-0abcdef1234567890', 'ami-1234567890abcdef0' `
    -ResourceType @(
        @{
            ResourceType = 'ec2:Instance'
            ResourceTypeOptions = @(
                @{
                    OptionName = 'state-name'
                    OptionValues = @('running', 'pending')
                }
            )
        },
        @{
            ResourceType = 'ec2:LaunchTemplate'
            ResourceTypeOptions = @(
                @{
                    OptionName = 'version-depth'
                    OptionValues = @('20')
                }
            )
        },
        @{
```

```
            ResourceType = 'ssm:Parameter'
        },
        @{
            ResourceType = 'imagebuilder:ImageRecipe'
        },
        @{
            ResourceType = 'imagebuilder:ContainerRecipe'
        }
    )
```

サポートされているすべてのリソースタイプの AMI 参照を確認するには

-IncludeAllResourceTypes パラメータで Get-EC2ImageReference コマンドレットを使用します。

```
Get-EC2ImageReference `
    -ImageId 'ami-0abcdef1234567890', 'ami-1234567890abcdef0' `
    -IncludeAllResourceTypes
```

サポートされているすべてのリソースタイプと特定のオプションの AMI 参照を確認するには

-IncludeAllResourceTypes と -ResourceType の両方のパラメータで Get-EC2ImageReference コマンドレットを使用します。この例では、実行中または保留中のインスタンスに対する EC2 インスタンスの応答範囲を指定しながら、すべてのリソースタイプをチェックします。

```
Get-EC2ImageReference `
    -ImageId 'ami-0abcdef1234567890', 'ami-1234567890abcdef0' `
    -IncludeAllResourceTypes `
    -ResourceType @(
        @{
            ResourceType = 'ec2:Instance'
            ResourceTypeOptions = @(
                @{
                    OptionName = 'state-name'
                    OptionValues = @('running', 'pending')
                }
            )
        }
    )
```

Amazon EC2 AMI を非推奨にする

AMI の使用を避けることで、それが古く、使用すべきではないことを示せます。AMI が非推奨となる将来の日付を特定し、AMI が使用期限切れになるタイミングを知ることも可能です。例えば、現在有効な管理が行われていない AMI の使用を避けたり、新しいバージョンで置き換えられている AMI を避けたりすることができます。新しいユーザーが古い AMI を使用することを防止するため、デフォルトで、非推奨の AMI は AMI のリストに表示されていません。ただし、既存のユーザーおよび起動サービス (起動テンプレートや Auto Scaling グループなど) では、ID を指定することで、非推奨の AMI を引き続き使用できます。AMI を削除して、ユーザーとサービスが使用できないようにするには、その AMI を登録解除します。

AMIが非推奨となった後は、以下が実施されます。

- AMI ユーザーの場合、非推奨の AMI は (ID を指定した場合や、非推奨の AMI を表示する必要があると指定した場合を除き) DescribeImages API 呼び出しに表示されなくなります。AMIの 所有者に対しては、非推奨の AMI は引き続き DescribeImages API 呼び出しに表示されます。
- AMI ユーザーは、非推奨の AMI をは EC2 コンソール経由で選択できなくなります。例えば、非推奨の AMI は、インスタンスの起動ウィザードの AMI カタログに表示されません。AMI 所有者の EC2 コンソール上には、非推奨の AMI が引き続き表示されます。
- AMI ユーザーで、非推奨となった AMI の ID がわかっている場合は、API、CLI、または SDK により、非推奨の AMI を使用しながらインスタンスの起動を継続することができます。
- 起動テンプレートや Auto Scaling グループなどの起動サービスは、非推奨の AMI を引き続き参照できます。
- 今後非推奨となる予定の AMI を使用して起動された EC2 インスタンスは影響を受けず、停止、起動、および再起動が可能です。

プライベート AMI とパブリック AMI の両方を非推奨にすることができます。

内容

- コスト
- 考慮事項
- AMI を非推奨にする
- 非推奨の AMI の詳細表示
- AMI の非推奨化をキャンセルする

コスト

AMI を非推奨にしても、その AMI は削除されません。AMI 所有者には、その AMI のスナップショットのための料金が引き続き請求されます。スナップショットの支払いを停止するには、AMI 所有者は、登録解除により AMI を削除する必要があります。

考慮事項

- AMI を非推奨にするには、AMI の所有者である必要があります。
- インスタンスを起動するために最近使用されていない AMI は、非推奨や登録解除の対象となる可能性が高いです。詳細については、「the section called “AMI が最後に使用された日時を確認する”」を参照してください。
- Amazon Data Lifecycle Manager EBS-backed AMI ポリシーを作成することで、EBS-backed AMI の廃止を自動化できます。詳細については、「Create AMI lifecycle policies」を参照してください。
- すべてのパブリック AMI を非推奨にする日をデフォルトで AMI 作成日の 2 年後とします。非推奨にする日は 2 年より前の日付に設定できます。非推奨にする日を取り消す場合や、非推奨にする日をもっと先の日付に変える場合は、AMI を特定の AWS アカウントとのみ共有するようにして、AMI を非公開にする必要があります。

AMI を非推奨にする

AMI を非推奨にする日時を指定することができます。ユーザーは、この AMI の所有者である必要があります。

非推奨日の上限は 10 年後ですが、パブリック AMI は例外で、上限は作成日から 2 年です。過去の日付を指定することはできません。

Console

AMI を特定の日付に非推奨にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左のナビゲーターで [AMI] を選択します。
3. フィルターバーから、[Owned by me] (自己所有) を選択します。
4. AMI を選択し、[Actions] (アクション)、[Manage AMI Deprecation] (AMI 非推奨を管理) の順に選択します。複数の AMI を選択して、複数の AMI の同じ非推奨日を一度に設定できます。

5. [有効化] のチェックボックスをオンにしたら、非推奨となった日付および時刻を入力します。
6. [保存] を選択します。

AWS CLI

AMI を特定の日付に非推奨にするには

enable-image-deprecation コマンドを使用します。秒の値を指定した場合は、Amazon EC2 により最も近い分に丸められます。

```
aws ec2 enable-image-deprecation \
    --image-id ami-0abcdef1234567890 \
    --deprecate-at "2025-04-15T13:17:12.000Z"
```

PowerShell

AMI を特定の日付に非推奨にするには

Enable-EC2ImageDeprecation コマンドレットを使用します。秒の値を指定した場合は、Amazon EC2 により最も近い分に丸められます。

```
Enable-EC2ImageDeprecation `
    -ImageId ami-0abcdef1234567890 `
    -DeprecateAt 2025-04-15T13:17:12.000Z
```

非推奨の AMI の詳細表示

AMI が非推奨になった日時を表示し、すべての AMI を非推奨になった日付でフィルタリングすることができます。

Console

AMI が非推奨になった日付を表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左側のナビゲーターで [AMI] をクリックした後、AMI を選択します。

3. [非推奨になった時刻] フィールドにチェックをオンにします (AMI の横にあるチェックボックスをオンにした場合、[詳細] タブにあります)。このフィールドには、AMI の非推奨の日次が表示されます。フィールドが空の場合は、AMI は非推奨ではありません。

非推奨になった日付で AMI をフィルタリングするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左のナビゲーターで [AMI] を選択します。
3. フィルターバーから、[Owned by me] (自己所有) または [Private images] (プライベートイメージ) を選択します (プライベートイメージには、共有されている AMI のほかに、所有している AMI も含まれます)。
4. 検索バーで **`Deprecation time`** と入力し (文字を入力すると、[Deprecation time] (非推奨となった時刻) のフィルターが表示されます)、演算子と日時を選択します。

AWS CLI

すべての AMI を記述すると、結果は、ユーザーが AMI ユーザーであるか AMI 所有者であるかに応じて変わってきます。

- AMI ユーザー – デフォルトでは、ユーザーがすべての AMI を記述すると、ユーザーに共有されているが、ユーザーが所有者ではない非推奨の AMI は、結果から除外されます。非推奨の AMI を結果に含めるには、`--include-deprecated` オプションを指定します。
- AMI 所有者 – ユーザーがすべての AMI を記述すると、ユーザーが所有するすべての AMI (非推奨の AMI を含む) が、結果に含まれます。`--no-include-deprecated` オプションを使用して、所有する非推奨の AMI を除外することはできません。

非推奨の AMI を含めて、アカウントのすべての AMI を記述するには

次の describe-images コマンドを使用します。

```
aws ec2 describe-images
    --owners 123456789012 \
    --include-deprecated
```

アカウントの非推奨の AMI を記述するには

次の describe-images コマンドを使用します。

```
aws ec2 describe-images \
    --owners self \
    --query "Images[?DeprecationTime!=null].ImageId" \
    --output text
```

以下は出力の例です。

```
ami-0abcdef1234567890
```

AMI が非推奨となった日付を表示するには

次の describe-images コマンドを使用します。出力に DeprecationTime がない場合、AMI は非推奨ではなく、将来的に非推奨に設定されることもありません。

```
aws ec2 describe-images \
    --image-ids ami-0abcdef1234567890 \
    --query Images[].DeprecationTime \
    --output text
```

以下は出力の例です。

```
2025-05-01T00:00:00.000Z
```

PowerShell

アカウントの非推奨の AMI を一覧表示するには

Get-EC2Image コマンドレットを使用します。

```
(Get-EC2Image -Owner self | Where-Object {$_.DeprecationTime -ne $null}).ImageId
```

以下は出力の例です。

```
ami-0abcdef1234567890
```

AMI が非推奨となった日付を表示するには

Get-EC2Image コマンドレットを使用します。出力に DeprecationTime がない場合、AMI は非推奨ではなく、将来的に非推奨に設定されることもありません。

```
(Get-EC2Image -ImageId ami-0abcdef1234567890).DeprecationTime
```

以下は出力の例です。

```
2025-05-01T00:00:00.000Z
```

AMI の非推奨化をキャンセルする

AMI の非推奨はキャンセルすることができ、それにより、非推奨の日時が削除されます。この手順を実行するには、AMI の所有者である必要があります。

Console

非推奨となっている AMI を復旧するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左のナビゲーターで [AMI] を選択します。
3. フィルターバーから、[Owned by me] (自己所有) を選択します。
4. AMI を選択し、[Actions] (アクション)、[Manage AMI Deprecation] (AMI 非推奨を管理) の順に選択します。複数の AMI を選択して、複数の AMI の非推奨を一度にキャンセルできます。
5. [有効化] チェックボックスをオフにしたら、[保存] を選択します。

AWS CLI

AMI の非推奨をキャンセルするには

次の disable-image-deprecation コマンドを使用します。

```
aws ec2 disable-image-deprecation --image-id ami-0abcdef1234567890
```

PowerShell

AMI の非推奨をキャンセルするには

Disable-EC2ImageDeprecation コマンドレットを使用します。

```
Disable-EC2ImageDeprecation -ImageId ami-0abcdef1234567890
```

Amazon EC2 AMI を無効にする

AMI を無効にして、インスタンスの起動に使用されないようにできます。無効な AMI から新しいインスタンスを起動することはできません。無効化された AMI を再度有効にして、インスタンスの起動時に再び使用できるようにすることができます。

プライベート AMI とパブリック AMI の両方を無効化できます。

使用頻度は低いが、長期間保持する必要がある無効な EBS-backed AMI のストレージコストを削減するには、関連するスナップショットをアーカイブできます。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS スナップショットのアーカイブ」を参照してください。

内容

- AMI 無効化の仕組み
- コスト
- 前提条件
- 必要な IAM 許可
- AMI の無効化
- 無効化された AMI の説明
- 無効化された AMI を再度有効にする

AMI 無効化の仕組み

> **Warning**
>
> AMI を無効にすると、AMI のすべての起動権限が削除されます。

AMI が無効になっている場合:

- AMI の状態は `disabled` に変わります。
- 無効化された AMI は共有できません。AMI が公開されていたか、以前に共有されていた場合は、非公開になります。AMI が AWS アカウント、組織または組織単位で共有されていた場合、無効になっている AMI にはアクセスできなくなります。
- 無効化された AMI は、デフォルトで DescribeImages API 呼び出しに表示されません。

- 無効化された AMI は [自分が所有] コンソールフィルタには表示されません。無効になっている AMI を検索するには、[無効化されたイメージ] コンソールフィルタを使用してください。
- 無効化された AMI は、EC2 コンソールのインスタンス起動時に選択できません。たとえば、無効化された AMI はインスタンスの起動ウィザードの AMI カタログに表示されません。また、起動テンプレート作成時にも表示されません。
- 起動テンプレートや Auto Scaling グループなどの起動サービスは、無効化された AMI を引き続き参照できます。無効化された AMI からのそれ以降のインスタンスの起動は失敗するため、使用可能な AMI のみを参照するように、起動テンプレートと Auto Scaling グループを更新することをお勧めします。
- 今後無効化される予定の AMI を使用して起動された EC2 インスタンスは影響を受けず、停止、起動、および再起動が可能です。
- 無効になっている AMI に関連するスナップショットは削除できません。関連するスナップショットを削除しようとすると `snapshot is currently in use` エラーになります。

AMI が再び有効になると:

- AMI の状態が `available` に変わり、インスタンスの起動に使用できるようになります。
- AMI は共有できます。
- AMI を無効にしたときに AMI にアクセスできなくなった AWS アカウント、組織、および組織単位は、自動的にアクセスを回復できませんが、AMI を再び共有することは可能です。

コスト

AMI を無効化しても、その AMI は削除されません。EBS-backed AMI の場合は、AMI の EBS スナップショットの料金を引き続きお支払いいただきます。AMI を残しておきたい場合は、スナップショットをアーカイブすることでストレージコストを削減できる場合があります。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS スナップショットのアーカイブ」を参照してください。AMI とそのスナップショットを保持したくない場合は、AMI を登録解除し、スナップショットを削除する必要があります。詳細については、「AMI の登録解除」を参照してください。

前提条件

AMI を無効または再度有効にするには、AMI の所有者である必要があります。

必要な IAM 許可

AMI を無効化する、および再度有効化するには、次の IAM 権限が必要です。

- `ec2:DisableImage`
- `ec2:EnableImage`

AMI の無効化

AMI は EC2 コンソールまたは AWS Command Line Interface (AWS CLI) を使用して無効にできます。この手順を実行するには、AMI の所有者である必要があります。

Console

AMI を無効化するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左のナビゲーションペインで [AMI] を選択してください。
3. フィルターバーから、[Owned by me] (自己所有) を選択します。
4. AMI を選択し、[アクション]、[AMI を無効にする] の順に選択します。複数の AMI を選択し、まとめて無効化することもできます。
5. [AMI を無効にする] ウィンドウで、[AMI を無効にする] を選択します。

AWS CLI

AMI を無効化するには

次の disable-image コマンドを使用します。

```
aws ec2 disable-image --image-id ami-0abcdef1234567890
```

PowerShell

AMI を無効化するには

Disable-EC2Image コマンドレットを使用します。

```
Disable-EC2Image -ImageId ami-0abcdef1234567890
```

無効化された AMI の説明

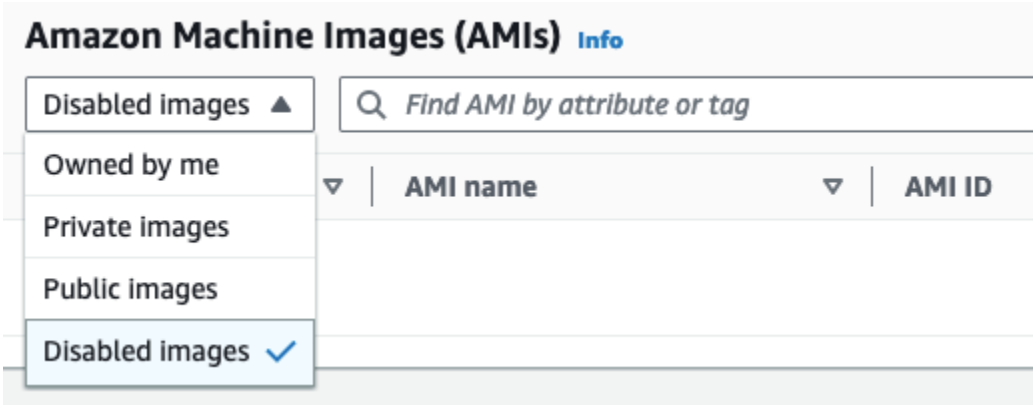
無効化された AMI は EC2 コンソールと AWS CLI を使用して表示できます。

無効化された AMI を表示するには AMI 所有者である必要があります。無効化された AMI は非公開になるため、所有者以外には表示されません。

Console

無効化された AMI を表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左のナビゲーションペインで [AMI] を選択してください。
3. フィルターバーから [無効化された画像] を選択します。



AWS CLI

デフォルトでは、すべての AMI を記述した場合、無効化された AMI は結果に含まれません。非推奨の AMI を結果に含めるには、`--include-disabled` オプションを指定します。AMI が無効になっている場合、AMI の `State` フィールドは `disabled` になります。

無効になっている AMI も含めてアカウントのすべての AMI を記述するには

次の describe-images コマンドを使用します。

```
aws ec2 describe-images \
    --owners 123456789012 \
    --include-disabled
```

アカウントの無効になっている AMI を一覧表示するには

次の describe-images コマンドを使用します。

```
aws ec2 describe-images \
```

```
    --owners self \
    --include-disabled \
    --filters Name=state,Values=disabled \
    --query Images[].ImageId \
    --output text
```

以下は出力の例です。

```
ami-0abcdef1234567890
```

AMI のステータスを記述するには

次の describe-images コマンドを使用します。出力に DeprecationTime がない場合、AMI は非推奨ではなく、将来的に非推奨に設定されることもありません。

```
aws ec2 describe-images \
    --image-ids ami-0abcdef1234567890 \
    --query Images[].State \
    --output text
```

以下は出力の例です。

```
disabled
```

PowerShell

デフォルトでは、すべての AMI を記述した場合、無効化された AMI は結果に含まれません。無効になった AMI を結果に含めるには、-IncludeDisabled パラメータを指定します。AMI が無効になっている場合、AMI の State フィールドは disabled になります。

アカウントの無効になっている AMI を一覧表示するには

Get-EC2Image コマンドレットを使用します。

```
(Get-EC2Image `
    -Owner self `
    -IncludeDisabled $true | Where-Object {$_.State -eq "disabled"}).ImageId
```

以下は出力の例です。

```
ami-0abcdef1234567890
```

AMI のステータスを記述するには

Get-EC2Image コマンドレットを使用します。

```
(Get-EC2Image -ImageId ami-0abcdef1234567890).State.Value
```

以下は出力の例です。

```
disabled
```

無効化された AMI を再度有効にする

無効化された AMI を再度有効にすることができます。この手順を実行するには、AMI の所有者である必要があります。

Console

無効化した AMI を再度有効にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左のナビゲーションペインで [AMI] を選択してください。
3. フィルターバーから [無効化された画像] を選択します。
4. AMI を選択し、[アクション]、[AMI を有効化] の順に選択します。複数の AMI を選択し、まとめて再有効化を選択することもできます。
5. [AMI を有効化] ウィンドウで、[有効化] を選択します。

AWS CLI

無効化した AMI を再度有効にするには

次の enable-image コマンドを使用します。

```
aws ec2 enable-image --image-id ami-0abcdef1234567890
```

PowerShell

無効化した AMI を再度有効にするには

Enable-EC2Image コマンドレットを使用します。

```
Enable-EC2Image -ImageId ami-0abcdef1234567890
```

Amazon EC2 AMI の登録を解除する

AMI の登録を解除すると、Amazon EC2 により AMI は完全に削除されます。AMI の登録を解除すると、それを使用して新しいインスタンスを起動できなくなります。AMI の利用が終わったら、その登録を解除することを検討するとよいでしょう。

AMI が誤ってまたは悪意によって登録解除されるのを防ぐために、登録解除保護を有効にすることができます。EBS-backed AMI を誤って登録解除した場合、完全に削除されるまでの許容期間内に復元した場合にのみ、ごみ箱から復元できます。

AMI の登録解除時に、関連付けられたスナップショットを同時に削除することも可能です。ただし、スナップショットが複数の AMI に関連付けられている場合、AMI は登録解除できますが、スナップショットはたとえ削除対象として指定していても削除されません。削除されていないスナップショットには、引き続きストレージコストが発生します。

AMI の登録を解除しても、AMI から既に起動したインスタンスは影響を受けません。これらのインスタンスを引き続き使用できます。デフォルトでは、AMI の登録を解除しても、その AMI の作成プロセスで作成されたスナップショットには影響はありません。これらのスナップショットのインスタンスとストレージコストには、引き続き使用料が発生します。したがって、不要なコストの発生を避けるには、インスタンスをすべて終了し、不要なスナップショットをすべて削除することをお勧めします。スナップショットは、登録解除時に自動的に削除することも、あるいは登録解除後に手動で削除することも可能です。詳細については、「未使用のリソースによるコストを回避する」を参照してください。

AMI から起動されたインスタンスがその後登録解除された場合でも、`describe-instance-image-metadata` AWS CLI コマンドを使用して AMI に関するいくつかの概要情報を表示できます。インスタンスのメタデタの詳細については、を参照してくださ.い。

内容

- 考慮事項
- AMI の登録解除

- 未使用のリソースによるコストを回避する
- Amazon EC2 AMI を登録解除から保護する

考慮事項

- アカウントが所有していない AMI の登録を解除することはできません。
- AWS Backup サービスで管理されている AMI の登録解除に、Amazon EC2 を使用することはできません。代わりに、AWS Backup を使用して、バックアップボールト内の対応するリカバリポイントを削除します。詳細については、「AWS Backup デベロッパーガイド」の「バックアップの削除」を参照してください。

AMI の登録解除

Amazon EBS-backed AMI と Amazon S3-backed AMI の登録を解除できます。EBS-backed AMI の場合、関連付けられたスナップショットを同時に削除することも可能です。ただし、スナップショットが他の AMI に関連付けられている場合は、たとえ削除対象として指定していても削除されません。

Console

AMI の登録を解除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [AMI] を選択してください。
3. フィルターバーから [自己所有] を選択して使用可能な AMI を一覧表示するか、[無効化されたイメージ] を選択して無効になっている AMI を一覧表示します。
4. 登録を解除する AMI を選択してください。
5. [アクション]、[AMI の登録解除] の順に選択してください。
6. (オプション) 関連付けられたスナップショットを登録解除時に削除するには、[関連付けられたスナップショットの削除] チェックボックスをオンにします。

> **Note**
>
> スナップショットが他の AMI に関連付けられている場合は、たとえこのチェックボックスを選択してもスナップショットは削除されないので注意してください。

7. [AMI の登録解除] を選択します。

   コンソールで AMI がリストから削除されるまで、数分ほどかかります。ステータスを更新するには、[リフレッシュ] を選択してください。

AWS CLI

AMI の登録を解除するには

次の deregister-image コマンドを使用します。

```
aws ec2 deregister-image --image-id ami-0abcdef1234567890
```

AMI の登録を解除し、関連するスナップショットを削除するには

次の deregister-image コマンドを使用して、--delete-associated-snapshots パラメータを指定します。スナップショットが他の AMI に関連付けられている場合、このパラメータを指定してもスナップショットは削除されないので注意してください。

```
aws ec2 deregister-image \
    --image-id ami-0abcdef1234567890 \
    --delete-associated-snapshots
```

PowerShell

AMI の登録を解除するには

Unregister-EC2Image コマンドレットを使用します。

```
Unregister-EC2Image -ImageId ami-0abcdef1234567890
```

AMI の登録を解除し、関連するスナップショットを削除するには

Unregister-EC2Image コマンドレットを使用して、-DeleteAssociatedSnapshots パラメータを指定します。スナップショットが他の AMI に関連付けられている場合、このパラメータを指定してもスナップショットは削除されないので注意してください。

```
Unregister-EC2Image `
    -ImageId ami-0abcdef1234567890 `
    -DeleteAssociatedSnapshots
```

未使用のリソースによるコストを回避する

デフォルトでは、AMI の登録を解除しても、その AMI に関連付けられたリソースは完全には削除されません。これらのリソースには、EBS-backed AMI のスナップショットと、Amazon S3-backed AMI の Amazon S3 内のファイルが含まれます。AMI の登録を解除しても、AMI から起動されたインスタンスが終了または停止することはありません。

スナップショットとファイルの保存には引き続き費用がかかり、実行中のインスタンスにも費用が発生します。

このような不要なコストの発生を避けるために、必要ないリソースを削除することをお勧めします。

EBS-backed AMI

- AMI の登録解除時に、関連付けられたスナップショットを削除します。詳細については、「AMI の登録解除」を参照してください。
- 関連付けられたスナップショットを削除せずに AMI を登録解除した場合は、手動でスナップショットを削除できます。AMI の作成時に作成されたインスタンスルートボリュームのスナップショットには、次の説明形式があります。

```
Created by CreateImage(i-1234567890abcdef0) for ami-0abcdef1234567890
```

- AMI から起動したインスタンスが不要になった場合は、それらを停止または終了できます。インスタンスを一覧表示するには、AMI の ID でフィルタリングします。

Amazon S3-backed AMI

- ec2-delete-bundle (AMI ツール) コマンドを使用して、Amazon S3 のバンドルを削除します。
- バンドルを削除した後に Amazon S3 バケットが空になり、不要になった場合は、そのバケットを削除できます。
- AMI から起動したインスタンスが不要になった場合は、それらを削除できます。インスタンスを一覧表示するには、AMI の ID でフィルタリングします。

Amazon EC2 AMI を登録解除から保護する

AMI の登録解除保護をオンにして、偶発的な削除や悪意のある削除を防ぐことができます。登録解除保護をオンにすると、IAM アクセス許可に関係なく、ユーザーによる AMI の削除はできなくなります。AMI の登録を解除するには、まず AMI の登録解除保護を無効にする必要があります。

AMI の登録解除保護を有効にする際、24 時間のクールダウン期間を持たせるオプションがあります。このクールダウン期間は、登録解除保護を無効にした後も有効のままになる時間です。このクールダウン期間中、AMI は登録解除できません。クールダウン期間が終了すると、AMI の登録解除が可能になります。

デフォルトでは、すべての既存および新規 AMI では登録解除保護はオフになっています。

登録解除保護をオンにする

登録解除保護をオンにするには、次の手順を使用します。

Console

登録解除保護をオンにするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. フィルターバーから [自己所有] を選択して使用可能な AMI を一覧表示するか、[無効化されたイメージ] を選択して無効になっている AMI を一覧表示します。
4. 登録解除保護を有効にする AMI を選択し、[アクション]、[AMI 登録解除保護の管理] の順に選択してください。
5. [AMI 登録解除保護の管理] ダイアログボックスでは、クールダウン期間を有効または無効にして、登録解除保護を有効にできます。以下のオプションのいずれかを選択してください。
   - [24 時間のクールダウン期間をつけて有効にする] — クールダウン期間を設定すると、登録解除保護をオフにしてから 24 時間は AMI の登録を解除できません。
   - [クールダウンなしで有効化] — クールダウン期間を設定しないと、登録解除保護がオフにしたときに AMI をすぐに登録解除できます。
6. [保存] を選択します。

AWS CLI

登録解除保護をオンにするには

enable-image-deregistration-protection コマンドを使用します。オプションのクールダウン期間を有効にするには、`--with-cooldown` オプションを含めます。

```
aws ec2 enable-image-deregistration-protection \
    --image-id ami-0abcdef1234567890 \
```

```
    --with-cooldown
```

PowerShell

登録解除保護をオンにするには

Enable-EC2ImageDeregistrationProtection コマンドレットを使用します。オプションのクールダウン期間を有効にするには、-WithCooldown パラメータを true に設定します。

```
Enable-EC2ImageDeregistrationProtection `
    -ImageId ami-0abcdef1234567890 `
    -WithCooldown $true
```

登録解除保護をオフにする

登録解除保護をオフにするには、次の手順を使用します。

AMI の登録解除保護をオンにしたときに 24 時間のクールダウン期間を設定した場合、登録解除保護をオフにしても、AMI をすぐに登録解除することはできません。このクールダウン期間は 24 時間の、登録解除保護を無効にした後も有効のままになる期間です。このクールダウン期間中、AMI は登録解除できません。クールダウン期間が終了すると、AMI を登録解除できます。

Console

登録解除保護をオフにするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. フィルターバーから [自己所有] を選択して使用可能な AMI を一覧表示するか、[無効化されたイメージ] を選択して無効になっている AMI を一覧表示します。
4. 登録解除保護を無効にする AMI を選択し、[アクション]、[AMI 登録解除保護の管理] の順に選択してください。
5. [AMI 登録解除保護の管理] ダイアログボックスで、[無効にする] を選択してください。
6. [保存] を選択します。

AWS CLI

登録解除保護をオフにするには

disable-image-deregistration-protection コマンドを使用します。

```
aws ec2 disable-image-deregistration-protection --image-id ami-0abcdef1234567890
```

PowerShell

登録解除保護をオフにするには

Disable-EC2ImageDeregistrationProtection コマンドレットを使用します。

```
Disable-EC2ImageDeregistrationProtection -ImageId ami-0abcdef1234567890
```

Amazon EC2 ブートモードでのインスタンスの起動動作

コンピュータが起動して最初に実行されるソフトウェアが、プラットフォームの初期化を行い、そのプラットフォーム固有の操作を実行するためのオペレーティングシステム用のインタフェースを提供する必要があります。

Amazon EC2 では、統合拡張ファームウェアインターフェイス (UEFI) とレガシー BIOS の、2 種類のブートモードソフトウェアがサポートされます。

AMI で使用可能なブートモードパラメータ

AMI のブートモードパラメータ値には、`uefi`、`legacy-bios` または `uefi-preferred` のどちらかを指定できます。AMI ブートモードパラメータの設定はオプションです。ブートモードパラメータがない AMI の場合、これらの AMI から起動されるインスタンスでは、インスタンスタイプごとのデフォルトのブートモード値が使用されます。

AMI ブートモードパラメータの目的

AMI ブートモードパラメータは、インスタンスの起動時に使用するブートモードを Amazon EC2 に通知します。ブートモードパラメータが `uefi` に設定されている場合、EC2 は UEFI でのインスタンスの起動を試みます。オペレーティングシステムが UEFI をサポートするように設定されていない場合、インスタンスの起動が失敗します。

UEFI Preferred ブートモードパラメータ

`uefi-preferred` ブートモードパラメータを使用して、UEFI とレガシー BIOS の両方をサポートする AMI を作成できます。ブートモードパラメータが `uefi-preferred` に設定されている場合、インスタンスタイプごとの EFI がサポートされている場合、インスタンスは UEFI での起動になりま

す。インスタンスタイプが UEFI をサポートしていない場合、インスタンスはレガシー BIOS で起動されます。

> **Warning**
>
> UEFI セキュアブートなどの一部の機能は、UEFI で起動するインスタンスでのみ使用できます。UEFI をサポートしないインスタンスタイプで `uefi-preferred` AMI ブートモードパラメータを使用すると、インスタンスはレガシー BIOS として起動し、UEFI 依存機能は無効になります。UEFI に依存する機能の可用性を重視する場合は、AMI ブートモードパラメータを `uefi` に設定します。

インスタンスタイプごとのデフォルトのブートモード

- Graviton インスタンスタイプ: UEFI
- Intel および AMD インスタンスタイプ: レガシー BIOS

ゾーンのサポート

UEFI ブートは Wavelength ゾーンではサポートされていません。

ブートモードのトピック

- UEFI ブートモードで EC2 インスタンスを起動するための要件
- Amazon EC2 AMI のブートモードパラメータを確認する
- EC2 インスタンスタイプがサポートしているブートモードを確認する
- EC2 インスタンスのブートモードを確認する
- EC2 インスタンスのオペレーティングシステムのブートモードを確認する
- Amazon EC2 AMI のブートモードを設定する
- Amazon EC2 インスタンスの UEFI 変数
- Amazon EC2 インスタンスの UEFI 安全ブート

UEFI ブートモードで EC2 インスタンスを起動するための要件

インスタンスのブートモードは、AMI の設定、その中に含まれるオペレーティングシステム、およびインスタンスタイプによって決まります。UEFI ブートモードでインスタンスを起動するには、以下の要件を満たしている必要があります。

AMI

UEFI では、AMI は次のように設定する必要があります。

- オペレーティングシステム – AMI に含まれるオペレーティングシステムです。UEFI の使用を設定する必要があり、この設定がない場合はインスタンスの起動に失敗します。詳細については、「EC2 インスタンスのオペレーティングシステムのブートモードを確認する」を参照してください。
- AMI ブートモードパラメータ - AMI のブートモードパラメータは `uefi` または `uefi-preferred` に設定します。詳細については、「Amazon EC2 AMI のブートモードパラメータを確認する」を参照してください。

Linux – 以下の Linux AMI は UEFI をサポートしています。

- Amazon Linux 2023
- Amazon Linux 2 (重力子 インスタンスタイプのみ)

その他の Linux AMI の場合、AMI の設定、VM Import/Export を介した AMI のインポート、または CloudEndure を介した AMI のインポートを行う必要があります。

Windows – 以下の Windows AMI は UEFI をサポートしています。

- Windows_Server-2025-* (`BIOS-` 名前のプレフィックスが付いた AMI を除く)
- TPM-Windows_Server-2022-English-Full-Base
- TPM-Windows_Server-2022-English-Core-Base
- TPM-Windows_Server-2019-English-Full-Base
- TPM-Windows_Server-2019-English-Core-Base
- TPM-Windows_Server-2016-English-Full-Base
- TPM-Windows_Server-2016-English-Core-Base

インスタンスタイプ

AWS Nitro System に構築されているすべてのインスタンス (ベアメタルインスタンス、DL1、G4ad、P4、u-3tb1、u-6tb1、u-9tb1、u-12tb1、u-18tb1、u-24tb1 および VT1 を除く) では、UEFI とレガシー BIOS の両方がサポートされます。詳細については、「the section called “インスタンスタイプのブートモード”」を参照してください。

次の表は、インスタンスのブートモード (インスタンス起動モードの結果列で示される) が AMI のブートモードパラメータ (列 1)、AMI に含まれるオペレーティングシステムのブートモード設定 (列

2)、およびインスタンスタイプのブートモードサポート (列 3) の組み合わせによって決定されることを示しています。

AMI ブートモードパラメータ	オペレーティングシステムのブートモード設定	インスタンスタイプのブートモードサポート	インスタンスのブートモードの結果
UEFI	UEFI	UEFI	UEFI
レガシー BIOS	レガシー BIOS	レガシー BIOS	レガシー BIOS
UEFI Preferred	UEFI	UEFI	UEFI
UEFI Preferred	UEFI	UEFI とレガシー BIOS	UEFI
UEFI Preferred	レガシー BIOS	レガシー BIOS	レガシー BIOS
UEFI Preferred	レガシー BIOS	UEFI とレガシー BIOS	レガシー BIOS
ブートモードが指定されていません - ARM	UEFI	UEFI	UEFI
ブートモードが指定されていません - x86	レガシー BIOS	UEFI とレガシー BIOS	レガシー BIOS

Amazon EC2 AMI のブートモードパラメータを確認する

AMI ブートモードパラメータの設定はオプションです。AMI のブートモードパラメータ値には、`uefi`、`legacy-bios` または `uefi-preferred` のどちらかを指定できます。

一部の AMI には、ブートモードパラメータがありません。AMI にブートモードパラメータがない場合、AMI から起動されるインスタンスでは、インスタンスタイプごとのデフォルト値が使用されます。重力子 ではこの設定は `uefi` となり、Intel および AMD インスタンスタイプでは `legacy-bios` となります。

Console

AMI のブートモードパラメータを確認するには ()

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI] をクリックした後、AMI を選択してください。
3. [ブートモード] フィールドを調べます。

   - uefi という値は、AMI が UEFI をサポートしていることを示します。
   - uefi-preferred という値は、AMI が UEFI とレガシー BIOS の両方をサポートしていることを示します。
   - 値がない場合、AMI から起動されるインスタンスは、インスタンスタイプのデフォルト値を使用します。

インスタンスの起動時に AMI のブートモードパラメータを確認するには

インスタンスの起動ウィザードを使用してインスタンスを起動する場合、AMI を選択するステップで、[ブートモード] フィールドを表示します。詳細については、「アプリケーションと OS イメージ (Amazon マシンイメージ)」を参照してください。

AWS CLI

AMI のブートモードパラメータを確認するには ()

describe-images コマンドを使用して、AMI のブートモードを確認します。

```
aws ec2 describe-images \
    --region us-east-1 \
    --image-id ami-0abcdef1234567890 \
    --query Images[].BootMode \
    --output text
```

以下は出力の例です。

```
uefi
```

出力で、uefi という値はその AMI が UEFI をサポートしていることを示しています。uefi-preferred の値により、その AMI が UEFI とレガシー BIOS の両方をサポートしていることを示されます。値がない場合、AMI から起動されるインスタンスは、インスタンスタイプのデフォルト値を使用します。

PowerShell

AMI のブートモードパラメータを確認するには ()

Get-EC2Image コマンドレットを使用して、AMI のブートモードを確認します。

```
Get-EC2Image -Region us-east-1 `
    -ImageId ami-0abcdef1234567890 | Format-List Name, BootMode, TpmSupport
```

以下は出力の例です。

```
Name       : TPM-Windows_Server-2016-English-Full-Base-2023.05.10
BootMode   : uefi
TpmSupport : v2.0
```

出力で、BootMode という値は AMI のブートモードを示しています。uefi という値により、その AMI が UEFI をサポートしていることを示します。uefi-preferred の値により、その AMI が UEFI とレガシー BIOS の両方をサポートしていることを示されます。値がない場合、AMI から起動されるインスタンスは、インスタンスタイプのデフォルト値を使用します。

EC2 インスタンスタイプがサポートしているブートモードを確認する

インスタンスタイプがサポートしているブートモードを確認できます。

Amazon EC2 コンソールには、インスタンスタイプがサポートしているブートモードは表示されません。

AWS CLI

describe-instance-types コマンドを使用して、インスタンスタイプがサポートしているブートモードを確認します。--query パラメータは、サポートされているブートモードのみを返すように出力をフィルタリングします。

次の例では、指定されたインスタンスタイプが UEFI ブートモードとレガシー BIOS ブートモードの両方をサポートしていることを示しています。

```
aws ec2 describe-instance-types \
    --instance-types m5.2xlarge \
    --query "InstanceTypes[*].SupportedBootModes"
```

以下は出力の例です。

```
[
    [
        "legacy-bios",
        "uefi"
    ]
]
```

次の例は、t2.xlarge がレガシー BIOS のみをサポートしていることを示しています。

```
aws ec2 describe-instance-types \
    --instance-types t2.xlarge \
    --query "InstanceTypes[*].SupportedBootModes"
```

以下は出力の例です。

```
[
    [
        "legacy-bios"
    ]
]
```

PowerShell

Get-EC2InstanceType コマンドレットを使用して、インスタンスタイプがサポートしているブートモードを確認します。

次の例では、m5.2xlarge が UEFI ブートモードとレガシー BIOS ブートモードの両方をサポートしていることを示しています。

```
Get-EC2InstanceType -InstanceType m5.2xlarge | Format-List InstanceType,
 SupportedBootModes
```

以下は出力の例です。

```
InstanceType       : m5.2xlarge
SupportedBootModes : {legacy-bios, uefi}
```

次の例は、t2.xlarge がレガシー BIOS のみをサポートしていることを示しています。

```
Get-EC2InstanceType -InstanceType t2.xlarge | Format-List InstanceType,
 SupportedBootModes
```

以下は出力の例です。

```
InstanceType       : t2.xlarge
SupportedBootModes : {legacy-bios}
```

UEFI をサポートするインスタンスタイプを確認するには

UEFI をサポートするインスタンスタイプを確認できます。Amazon EC2 コンソールには、インスタンスタイプの UEFI のサポートは表示されません。

AWS CLI

利用可能なインスタンスタイプは、AWS リージョン ごとに異なります。リージョンで使用可能であり、UEFI をサポートしているインスタンスタイプを確認するには、describe-instance-types コマンドを使用します。`--filters` パラメータを含めることで結果の範囲を UEFI をサポートするインスタンスタイプに規定し、`--query` パラメータを含めることで出力の範囲を `InstanceType` の値に規定します。

```
aws ec2 describe-instance-types \
    --filters Name=supported-boot-mode,Values=uefi \
    --query "InstanceTypes[*].[InstanceType]" \
    --output text | sort
```

PowerShell

利用可能なインスタンスタイプは、AWS リージョン ごとに異なります。リージョンで使用可能であり、UEFI をサポートしているインスタンスタイプを確認するには、Get-EC2InstanceType コマンドレットを使用します。

```
Get-EC2InstanceType -Filter @{Name="supported-boot-mode"; Values="uefi"} |
 Sort-Object @{E={$_.ProcessorInfo.SupportedArchitectures}}, InstanceType |
 Format-Table InstanceType, SupportedBootModes, `
      @{Name="SupportedArchitectures"; E={$_.ProcessorInfo.SupportedArchitectures}}
```

UEFI 安全ブート をサポートし不揮発性変数を保持する、インスタンスタイプを確認するには

ベアメタルインスタンスでは UEFI 安全ブート と不揮発性変数はサポートされないため、以下の例では出力から除外します。UEFI 安全ブート の詳細については、「Amazon EC2 インスタンスの UEFI 安全ブート」を参照してください。

AWS CLI

describe-instance-types コマンドを使用し、出力からベアメタルインスタンスを除外します。

```
aws ec2 describe-instance-types \
    --filters Name=supported-boot-mode,Values=uefi Name=bare-metal,Values=false \
    --query "InstanceTypes[*].[InstanceType]" \
    --output text | sort
```

PowerShell

Get-EC2InstanceType コマンドレットを使用し、出力からベアメタルインスタンスを除外します。

```
Get-EC2InstanceType -Filter `
      @{Name="supported-boot-mode"; Values="uefi"}, `
      @{Name="bare-metal"; Values="false"} |
 Sort-Object @{E={$_.ProcessorInfo.SupportedArchitectures}}, InstanceType |
 Format-Table InstanceType, SupportedBootModes, `
      @{Name="SupportedArchitectures"; E={$_.ProcessorInfo.SupportedArchitectures}}
```

EC2 インスタンスのブートモードを確認する

インスタンスのブートモードは、Amazon EC2 コンソールの [ブートモード] フィールドに表示され、AWS CLI の currentInstanceBootMode パラメータによって表示されます。

インスタンスの起動時、そのブートモードパラメータの値は、インスタンスの起動に使用された AMI のブートモードパラメータの値によって決まります。

- uefi のブートモードパラメータを持つ AMI は、uefi の currentInstanceBootMode パラメータを持つインスタンスを作成します。
- legacy-bios のブートモードパラメータを持つ AMI は、 legacy-bios の currentInstanceBootMode パラメータを持つインスタンスを作成します。
- uefi-preferred のブートモードパラメータを持つ AMI は、インスタンスタイプが UEFI をサポートしている場合はという uefi の currentInstanceBootMode パラメータを持つインス

タンスを作成します。それ以外の場合は、legacy-bios の currentInstanceBootMode パラメータがのインスタンスを作成します。

- ブートモードのパラメータ値を持たない AMI は、AMI アーキテクチャが ARM か x86 か、サポートされているインスタンスタイプのブートモードによって決まる currentInstanceBootMode パラメータ値を持つインスタンスを作成します。デフォルトのブートモードは、重力子 インスタンスタイプでは uefi、Intel と AMD インスタンスタイプでは legacy-bios です。

Console

インスタンスのブートモードを確認するには ()

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [インスタンス] を選択し、インスタンスを選択してください。
3. [詳細] タブを開き、[ブートモード] フィールドを確認します。

AWS CLI

インスタンスのブートモードを確認するには ()

インスタンスのブートモードを決定するには describe-instances を使用します。インスタンスの作成に使用された AMI のブートモードを確認することもできます。

```
aws ec2 describe-instances \
    --region us-east-1 \
    --instance-ids i-1234567890abcdef0 \
    --query Reservations[].Instances[].BootMode \
    --output text
```

以下は出力の例です。

```
uefi
```

PowerShell

インスタンスのブートモードを確認するには ()

インスタンスのブートモードを決定するには Get-EC2Image コマンドレットを使用します。インスタンスの作成に使用された AMI のブートモードを確認することもできます。

```
(Get-EC2Instance `
    -InstanceId i-1234567890abcdef0).Instances | Format-List BootMode,
 CurrentInstanceBootMode, InstanceType, ImageId
```

以下は出力の例です。

```
BootMode                : uefi
CurrentInstanceBootMode : uefi
InstanceType            : c5a.large
ImageId                 : ami-0abcdef1234567890
```

EC2 インスタンスのオペレーティングシステムのブートモードを確認する

Amazon EC2 のブートモードは AMI のブートモードに従います。このブートモードがインスタンスの起動に使用されます。インスタンスのオペレーティングシステムが UEFI 用に設定されているかどうかを確認するには、SSH (Linux インスタンス) または RDP (Windows インスタンス) を使用してインスタンスに接続する必要があります。

インスタンスのオペレーティングシステムの説明を使用してください。

Linux

インスタンスのオペレーティングシステムのブートモードを特定するには

1. SSH を使用しての Linux インスタンスへの接続
2. オペレーティングシステムのブートモードを表示するには、以下のいずれかを実行します。

   - 以下のコマンドを実行してください。

     ```
     [ec2-user ~]$ sudo /usr/sbin/efibootmgr
     ```

     UEFI ブートモードで起動されたインスタンスで想定される出力

     ```
     BootCurrent: 0001
     Timeout: 0 seconds
     BootOrder: 0000,0001
     Boot0000* UiApp
     Boot0001* UEFI Amazon Elastic Block Store vol-xyz
     ```

- 次のコマンドを実行して、/sys/firmware/efi ディレクトリが存在するか確認します。このディレクトリは、インスタンスが UEFI を使用して起動する場合のみ存在します。このディレクトリが存在しない場合、このコマンドは Legacy BIOS Boot Detected を返します。

```
[ec2-user ~]$ [ -d /sys/firmware/efi ] && echo "UEFI Boot Detected" || echo
 "Legacy BIOS Boot Detected"
```

UEFI ブートモードで起動されたインスタンスで想定される出力

```
UEFI Boot Detected
```

レガシー BIOS ブートモードで起動されたインスタンスで想定される出力

```
Legacy BIOS Boot Detected
```

- 次のコマンドを実行して、dmesg 出力に EFI が 含まれていることを確認します。

```
[ec2-user ~]$ dmesg | grep -i "EFI"
```

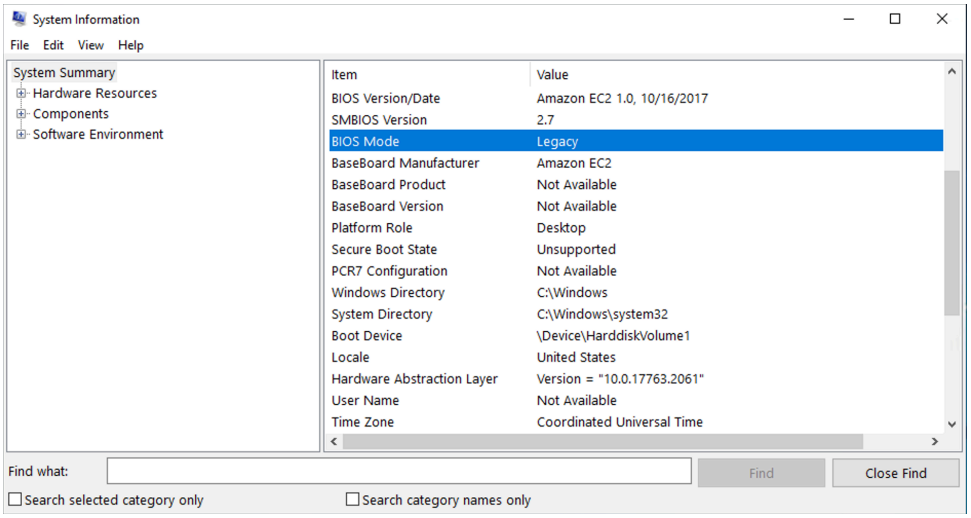
UEFI ブートモードで起動されたインスタンスで想定される出力

```
[    0.000000] efi: Getting EFI parameters from FDT:
[    0.000000] efi: EFI v2.70 by EDK II
```

Windows

インスタンスのオペレーティングシステムのブートモードを特定するには

1. RDP を使用しての Windows インスタンスへの接続
2. [システム情報] を開き、[BIOS モード] 行を確認します。



Amazon EC2 AMI のブートモードを設定する

デフォルトでは、AMI には、その作成に使用される EC2 インスタンスのブートモードが継承されます。例えば、レガシー BIOS で実行されている EC2 インスタンスから AMI を作成する場合、新しい AMI のブートモードは `legacy-bios` になります。ブートモードが `uefi-preferred` の EC2 インスタンスから AMI を作成する場合、新しい AMI のブートモードは `uefi-preferred` になります。

AMI を登録する際に、AMI のブートモードを `uefi`、`legacy-bios`、または `uefi-preferred` に設定できます。

AMI ブートモードを `uefi-preferred` に設定すると、インスタンスは次のように起動します。

- UEFI とレガシー BIOS の両方をサポートするインスタンスタイプ (例えば `m5.large` など) の場合、インスタンスは UEFI を使用して起動します。
- レガシー BIOS のみをサポートするインスタンスタイプ (例えば `m4.large` など) の場合、インスタンスはレガシー BIOS を使用して起動します。

AMI ブートモードを `uefi-preferred` に設定した場合、オペレーティングシステムは UEFI と Legacy BIOS の両方を起動する機能をサポートしている必要があります。

既存のレガシー BIOS ベースのインスタンスを UEFI に、または既存の UEFI ベースのインスタンスをレガシー BIOS に変換するには、まず、選択したブートモードをサポートするように、インスタンスのボリュームとオペレーティングシステムを変更する必要があります。次に、ボリュームのスナップショットを作成します。最後に、スナップショットから AMI を作成します。

考慮事項

- AMI ブートモードパラメータを設定しても、オペレーティングシステムは指定されたブートモードに自動的に変更されません。最初に、インスタンスのボリュームとオペレーティングシステムに対し、選択したブートモードを使用した起動をサポートするための適切な変更を加える必要があります。これを行わないと、作成される AMI を使用できません。例えば、レガシー BIOS ベースの Windows インスタンスを UEFI に変換する場合、Microsoft の MBR2GPT ツールを使用して、システムディスクを MBR から GPT に変換できます。ここでの変更内容は、オペレーティングシステムにより異なります。詳細については、オペレーティングシステムのマニュアルを参照してください。
- register-image コマンドまたは Register-EC2Image コマンドレットを使用して、NitroTPM と UEFI Preferred の両方をサポートする AMI を作成することはできません。
- UEFI セキュアブートなどの一部の機能は、UEFI で起動するインスタンスでのみ使用できます。UEFI をサポートしないインスタンスタイプで `uefi-preferred` AMI ブートモードパラメータを使用すると、インスタンスはレガシー BIOS として起動し、UEFI 依存機能は無効になります。UEFI に依存する機能の可用性を重視する場合は、AMI ブートモードパラメータを `uefi` に設定します。

AWS CLI

AMI のブートモードを設定するには

1. インスタンスのボリュームとオペレーティングシステムに対し、選択したブートモードでの起動をサポートするための適切な変更を加えます。ここでの変更内容は、オペレーティングシステムにより異なります。詳細については、オペレーティングシステムのマニュアルを参照してください。

   > **Warning**
   >
   > このステップを実行しないと、AMI は使用不可能になります。

2. インスタンスのボリューム ID を、describe-instances コマンドを使用して確認します。次のステップでは、このボリュームのスナップショットを作成します。

```
aws ec2 describe-instances \
    --instance-ids i-1234567890abcdef0 \
    --query Reservations[].Instances[].BlockDeviceMappings
```

以下は出力の例です。

```
[
    [
        {
            "DeviceName": "/dev/xvda",
            "Ebs": {
                "AttachTime": "2024-07-11T01:05:51+00:00",
                "DeleteOnTermination": true,
                "Status": "attached",
                "VolumeId": "vol-1234567890abcdef0"
            }
        }
    ]
]
```

3. ボリュームのスナップショットを作成するには、create-snapshot コマンドを使用します。前のステップで取得したボリューム ID を使用します。

```
aws ec2 create-snapshot \
    --volume-id vol-01234567890abcdef \
    --description "my snapshot"
```

以下は出力の例です。

```
{
    "Description": "my snapshot",
    "Encrypted": false,
    "OwnerId": "123456789012",
    "Progress": "",
    "SnapshotId": "snap-0abcdef1234567890",
    "StartTime": "",
    "State": "pending",
    "VolumeId": "vol-01234567890abcdef",
    "VolumeSize": 30,
    "Tags": []
}
```

4. スナップショットの状態が completed になるまで待ってから、次のステップに進みます。スナップショットの状態を取得するには、describe-snapshots コマンドを前のステップのスナップショット ID と共に使用します。

```
aws ec2 describe-snapshots \
    --snapshot-ids snap-0abcdef1234567890 \
    --query Snapshots[].State \
    --output text
```

以下は出力の例です。

```
completed
```

5. 新しい AMI を作成するには、register-image コマンドを使用します。CreateSnapshot の出力の SnapshotId の値を使用します。

   - ブートモードを UEFI に設定するには、uefi の値を使用した --boot-mode パラメータを追加します。

   ```
   aws ec2 register-image \
      --description "my image" \
      --name "my-image" \
      --block-device-mappings "DeviceName=/dev/
   sda1,Ebs={SnapshotId=snap-0abcdef1234567890,DeleteOnTermination=true}" \
      --root-device-name /dev/sda1 \
      --virtualization-type hvm \
      --ena-support \
      --boot-mode uefi
   ```

   - ブートモードを uefi-preferred に設定するには、--boot-mode の値を uefi-preferred に設定します

   ```
   aws ec2 register-image \
      --description "my description" \
      --name "my-image" \
      --block-device-mappings "DeviceName=/dev/
   sda1,Ebs={SnapshotId=snap-0abcdef1234567890,DeleteOnTermination=true}" \
      --root-device-name /dev/sda1 \
      --virtualization-type hvm \
      --ena-support \
      --boot-mode uefi-preferred
   ```

6. (オプション) 新しく作成した AMI が、指定したブートモードに設定されていることを確認するには、describe-images コマンドを使用します。

```
aws ec2 describe-images \
    --image-id ami-1234567890abcdef0 \
    --query Images[].BootMode \
    --output text
```

以下は出力の例です。

```
uefi
```

PowerShell

AMI のブートモードを設定するには

1. インスタンスのボリュームとオペレーティングシステムに対し、選択したブートモードでの起動をサポートするための適切な変更を加えます。ここでの変更内容は、オペレーティングシステムにより異なります。詳細については、オペレーティングシステムのマニュアルを参照してください。

   > **Warning**
   >
   > このステップを実行しないと、AMI は使用不可能になります。

2. インスタンスのボリューム ID を確認するには、Get-EC2Instance コマンドレットを使用します。

```
(Get-EC2Instance `
    -InstanceId i-1234567890abcdef0).Instances.BlockDeviceMappings.Ebs
```

以下は出力の例です。

```
AssociatedResource  :
AttachTime          : 7/11/2024 1:05:51 AM
DeleteOnTermination : True
Operator            :
Status              : attached
VolumeId            : vol-01234567890abcdef
```

3. ボリュームのスナップショットを作成するには、New-EC2Snapshot コマンドレットを使用します。前のステップで取得したボリューム ID を使用します。

```
New-EC2Snapshot `
    -VolumeId vol-01234567890abcdef `
    -Description "my snapshot"
```

以下は出力の例です。

```
AvailabilityZone          :
Description               : my snapshot
Encrypted                 : False
FullSnapshotSizeInBytes   : 0
KmsKeyId                  :
OwnerId                   : 123456789012
RestoreExpiryTime         :
SnapshotId                : snap-0abcdef1234567890
SseType                   :
StartTime                 : 4/25/2025 6:08:59 PM
State                     : pending
StateMessage              :
VolumeId                  : vol-01234567890abcdef
VolumeSize                : 30
```

4. スナップショットの状態が completed になるまで待ってから、次のステップに進みます。スナップショットの状態を取得するには、Get-EC2Snapshot コマンドレットを前のステップのスナップショット ID と共に使用します。

```
(Get-EC2Snapshot `
    -SnapshotId snap-0abcdef1234567890).State.Value
```

以下は出力の例です。

```
completed
```

5. 新しい AMI を作成するには、Register-EC2Image コマンドレットを使用します。New-EC2Snapshot の出力の SnapshotId の値を使用します。

   - ブートモードを UEFI に設定するには、uefi の値を使用した -BootMode パラメータを追加します。

```
$block = @{SnapshotId=snap-0abcdef1234567890}
Register-EC2Image `
   -Description "my image" `
   -Name "my-image" `
   -BlockDeviceMapping @{DeviceName="/dev/xvda";Ebs=$block} `
   -RootDeviceName /dev/xvda `
   -EnaSupport $true `
   -BootMode uefi
```

- ブートモードを uefi-preferred に設定するには、-BootMode の値を uefi-preferred に設定します

```
$block = @{SnapshotId=snap-0abcdef1234567890}
Register-EC2Image `
   -Description "my image" `
   -Name "my-image" `
   -BlockDeviceMapping @{DeviceName="/dev/xvda";Ebs=$block} `
   -RootDeviceName /dev/xvda `
   -EnaSupport $true `
   -BootMode uefi-preferred
```

6. (オプション) 新しく作成した AMI が、指定したブートモードに設定されていることを確認するには、Get-EC2Image コマンドレットを使用します。

```
(Get-EC2Image `
    -ImageId ami-1234567890abcdef0).BootMode.Value
```

以下は出力の例です。

```
uefi
```

Amazon EC2 インスタンスの UEFI 変数

ブートモードが UEFI に設定されているインスタンスを起動すると、変数の key-value ストアが作成されます。このストアは、UEFI およびインスタンスオペレーティングシステムで UEFI 変数を格納するために使用できます。

UEFI 変数は、ブートローダーとオペレーティングシステムにより使用されるもので、システムの起動初期における処理を指定します。これにより、オペレーティングシステムは、(ブート順序や UEFI 安全ブート キーの管理など) ブートプロセスに関する特定の設定を行えます。

> **Warning**
>
> インスタンス (およびインスタンス上で実行されている可能性のある任意のソフトウェア) に接続できるユーザー、またはインスタンスで GetInstanceUefiData API を使用するアクセス許可を持つユーザーは誰でも変数を読み取ることができます。パスワードや個人識別情報などの機密データを UEFI 変数ストアに保存しないでください。

UEFI 変数の永続性

- 2022 年 5 月 10 日以前に起動されたインスタンスの UEFI 変数は、再起動または停止時に消去されます。
- 2022 年 5 月 11 日以降に起動されたインスタンスの場合、不揮発性として設定された UEFI 変数であれば、再起動および停止/開始時にも保持されます。
- ベアメタルインスタンスの場合は、インスタンスの停止/開始オペレーションの全体を通して、不揮発性の UEFI 変数は保持されません。

Amazon EC2 インスタンスの UEFI 安全ブート

UEFI 安全ブート は、長期の使用実績がある Amazon EC2 の安全なブートプロセスを基に構築されたものです。これには詳細な防御が追加されているので、ユーザーは、起動後も持続する脅威からソフトウェアを保護できます。インスタンスが起動できるのは、暗号化キーで署名されたソフトウェアのみになります。キーは、UEFI の不揮発性変数ストアのキーデータベースに保存されています。UEFI 安全ブート は、インスタンスのブートフローが不正な変更を受けることを防止します。

内容

- Amazon EC2 インスタンスで UEFI 安全ブート を使用する方法
- Amazon EC2 で UEFI 安全ブート を使用するための要件
- Amazon EC2 インスタンスで UEFI 安全ブート が有効化されているかどうかを確認する
- カスタム UEFI 安全ブート キーを使用して Linux AMI を作成する
- UEFI 安全ブート の AWS バイナリ BLOB を作成する

Amazon EC2 インスタンスで UEFI 安全ブート を使用する方法

UEFI 安全ブート は、UEFI の中で規定された機能であり、これにより、ブートチェーンの状態を検証することができます。ファームウェア自体による初期化後は、暗号を使用した検証が行われた UEFI バイナリコードのみを、実行するように設計されています。これらのバイナリコードには、UEFI ドライバーやメインブートローダーに加え、ブートチェーンによりロードされるコンポーネントも含まれます。

UEFI 安全ブート では、信頼チェーンで使用される 4 つのキーデータベースが指定されています。このデータベースは、UEFI の変数ストアに格納されています。

信頼チェーンには、以下が含まれます。

プラットフォームキー (PK) データベース

PK データベースは信頼チェーンのルートに置かれます。これには、キー交換キー (KEK) データベースを更新する際に信頼チェーンで使用される、単一のパブリック PK キーが含まれています。

PK データベースを変更するためには、プライベート PK キーを使用して、その更新リクエストに署名する必要があります。この変更処理には、空の PK キーの書き込みによる PK データベースの削除も含まれます。

キー交換キー (KEK) データベース

KEK データベースでは、公開 KEK キーがリストされています。これらのキーは、署名データベース (db) と拒否リストデータベース (dbx) を更新する際に、信頼チェーンが使用します。

パブリック KEK データベースを変更するには、プライベート PK キーを使用して、更新のリクエストに署名をする必要があります。

署名 (DB) データベース

db データベースには、すべての UEFI ブートバイナリを検証するために信頼チェーンが使用する、パブリックキーとハッシュがリストされています。

db データベースを変更するには、プライベート PK キーまたはプライベート KEK キーを使用して、更新リクエストに署名する必要があります。

署名拒否リスト (dbx) データベース

dbx データベースは、信頼されていないパブリックキーとバイナリハッシュをリストします。このリストは、信頼チェーンが失効ファイルとして使用します。

dbx データベースは、常に、他のすべてのキーデータベースよりも優先されます。

dbx データベースを変更するには、プライベート PK キーまたはプライベート KEK キーを使用して、その更新リクエストに署名する必要があります。

UEFI フォーラム (https://uefi.org/revocationlistfile) には、既知の問題のあるバイナリコードと証明書を多数リストした dbx が公開されており、いつでも使用できます。

> **Important**
>
> UEFI 安全ブート は、任意の UEFI バイナリでシグニチャ検証を適用します。UEFI 安全ブート 内での UEFI バイナリの実行を許可するには、上記で説明したいずれかのプライベート db キーを使用して、そのバイナリに署名します。

デフォルトでは、UEFI 安全ブート は無効になっており、システムは `SetupMode` になっています。`SetupMode` の状態であるシステムでは、暗号による署名なしですべてのキー変数の更新が可能です。PK が設定されると、UEFI 安全ブート が有効化されるとともに、SetupMode が取り消されます。

Amazon EC2 で UEFI 安全ブート を使用するための要件

サポートされている AMI とサポートされているインスタンス タイプで Amazon EC2 インスタンスを起動すると、そのインスタンスは UEFI 安全ブート データベースで UEFI ブートバイナリを自動的に検証します。追加の設定は必要ありません。UEFI 安全ブート は、インスタンスを起動した後に設定することもできます。

> **Note**
>
> UEFI 安全ブート は、インスタンスとそのオペレーティングシステムについて、ブートフローが変更されないように保護します。UEFI 安全ブート が有効になっているソース AMI から新しい AMI を作成し、コピープロセス中に AMI 内で `UefiData` を変更するなど、特定のパラメータを変更すると、UEFI 安全ブート を無効にできます。

内容

- サポートされている AMI
- サポートされるインスタンスタイプ

サポートされている AMI

Linux AMI

Linux インスタンスを起動するには、Linux AMI で UEFI セキュアブートが有効になっている必要があります。

Amazon Linux は、AL2023 リリース 2023.1 から UEFI セキュアブートをサポートしています。ただし UEFI セキュアブートは、デフォルトの AMI では有効になっていません。詳細については、「AL2023 ユーザーガイド」の「UEFI Secure Boot」を参照してください。古いバージョンの Amazon Linux AMI では、UEFI セキュアブートは有効ではありません。サポートされている AMI を使用するには、独自の Linux AMI でいくつかの設定ステップを実行する必要があります。詳細については、「カスタム UEFI 安全ブート キーを使用して Linux AMI を作成する」を参照してください。

Windows AMI

Windows インスタンスを起動するには、Windows AMI で UEFI セキュアブートが有効になっている必要があります。Microsoft キーで UEFI Secure Boot 用に事前設定された AWS Windows AMI を見つけるには、「AWSWindows AMI リファレンス」の「NitroTPM と UEFI Secure Boot で設定された Windows Server AMI を見つける」を参照してください。

現時点では、import-image コマンドにより、UEFI 安全ブート 使用する Windows をインポートすることはできません。

サポートされるインスタンスタイプ

UEFI をサポートするすべての仮想インスタンスタイプは、UEFI 安全ブート もサポートします。UEFI 安全ブート をサポートするインスタンスタイプについては、UEFI ブートモードの要件 を参照してください。

> **Note**
>
> ベアメタルインスタンスタイプは UEFI 安全ブート をサポートしていません。

Amazon EC2 インスタンスで UEFI 安全ブート が有効化されているかどうかを確認する

次のステップを使用して、Amazon EC2 で UEFI 安全ブート が有効になっているかどうかを確認できます。

Linux インスタンス

Linux インスタンスが UEFI 安全ブート に対して有効になっているかどうかを確認するには `mokutil` ユティリティを使用できます。`mokutil` がインスタンスにインストールされていない場合は、これをインストールする必要があります。Amazon Linux 2 のインストールステップについては、「Find and install software packages on an Amazon Linux 2 instance」を参照してください。その他の Linux ディストリビューションについては、それぞれの個別のドキュメントを参照してください。

Linux インスタンスが UEFI 安全ブート に対して有効になっているかどうかを確認するには

インスタンスに接続し、ターミナルウィンドウで `root` として次のコマンドを実行します。

```
mokutil --sb-state
```

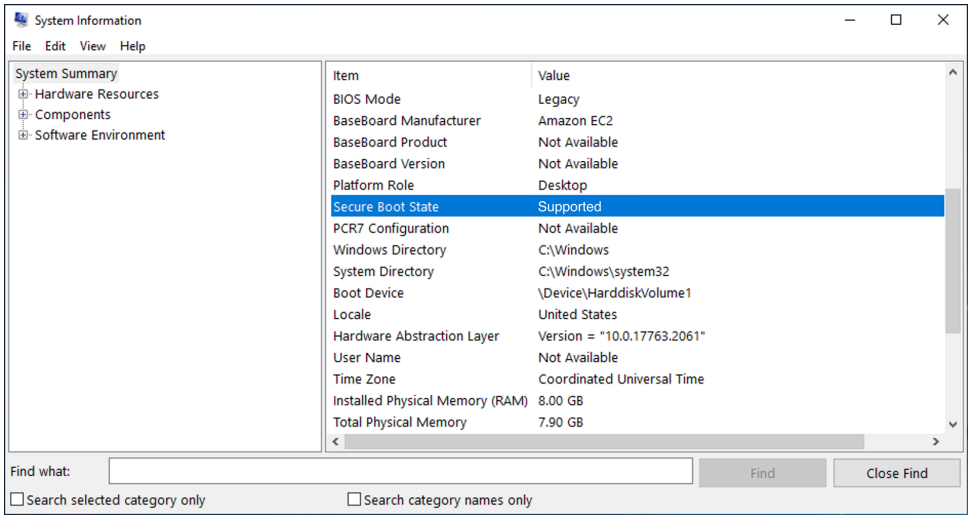
以下は出力の例です。

- UEFI 安全ブート が有効な場合、出力には `SecureBoot enabled` が含まれます。
- UEFI 安全ブート が有効化されていない場合は、出力に `SecureBoot disabled` または `Failed to read SecureBoot` が含まれます。

Windows インスタンス

Windows インスタンスで UEFI 安全ブート が有効になっているかどうかを確認するには

1. インスタンスに接続します。
2. msinfo32 ツールを開きます。
3. [安全ブート State] (安全ブート の状態) フィールドを確認します。UEFI 安全ブート が有効になっている場合、次の図に示すように、この値は [サポート対象] です。



Windows PowerShell コマンドレット `Confirm-SecureBootUEFI` を使用して安全ブートのステータスを確認することもできます。コマンドの詳細については、Microsoft ドキュメントの「Confirm-SecureBootUEFI」を参照してください。

カスタム UEFI 安全ブート キーを使用して Linux AMI を作成する

この手順では、UEFI 安全ブートとカスタムメイドのプライベートキーを使用して Linux AMI を作成する方法を示します。Amazon Linux は、AL2023 リリース 2023.1 から UEFI セキュアブートをサポートしています。詳細については、「Amazon Linux 2023 ユーザーガイド」の「UEFI Secure Boot on AL2023」を参照してください。

> **Important**
>
> 次の手順は、上級ユーザーのみを対象としています。これらの手順を使用するには、SSL および Linux ディストリビューションのブートフローに関する十分な知識が必要です。

前提条件

- 以下のツールを使用します。
  - OpenSSL – https://www.openssl.org/
  - efivar – https://github.com/rhboot/efivar
  - efitools – https://git.kernel.org/pub/scm/linux/kernel/git/jejb/efitools.git/
  - get-instance-uefi-data コマンド

- Linux インスタンスは、UEFI ブートモードをサポートする Linux AMI で起動され、不揮発性データを使用している必要があります。

UEFI 安全ブート キーを使用せずに新規で作成されたインスタンスには、SetupMode が適用されます。この状態で、独自のキーを登録することが可能です。一部の AMI では UEFI 安全ブート が事前設定されており、既定のキーを変更することはできません。AMI のキーを変更する場合は、その AMI をベースに、新らたな AMI を作成する必要があります。

変数ストア内のキーを伝達するには 2 つの方法があり、それぞれを、以下の オプション A とオプション B の中で説明します。オプション A では、実際のハードウェアのフローを模倣することで、インスタンス内からこれを行う方法について説明します。オプション B では、AMI の作成時に base64 でエンコードされたファイルとして渡される、バイナリ BLOB を作成する方法について説明します。どちらのオプションでも、最初に、信頼チェーンに使用するためのキーペアを 3 つ作成する必要があります。

UEFI 安全ブート をサポートする Linux AMI を作成するには、まず 3 つのキーペアを作成し、オプション A またはオプション B のいずれかを実行します。両方は実行しません。

- タスク 1: キーペアを作成する
- タスク 2 – オプション A: インスタンス内から変数ストアにキーを追加する
- タスク 2 – オプション B: 値が事前に設定済みの変数ストアを含むバイナリ BLOB を作成する

タスク 1: キーペアを作成する

UEFI 安全ブート は、プラットフォームキー (PK)、キー交換キー (KEK)、署名データベース (db) という 3 つのキーデータベースに基づいおり、これらのデータベースは信頼チェーンで使用されます。

インスタンスでは、これらの各キーを作成する必要があります。UEFI 安全ブート 標準が有効な形式でパブリックキーの準備を行うには、それぞれのキー用に証明書を作成します。DER では SSL 形式 (バイナリエンコード用の形式) を定義しています。その後、各証明書を UEFI 署名リストに変換します。このリストはバイナリ形式で、UEFI 安全ブート による解析が可能です。最後に、関連するキーで各証明書に署名します。

タスク

- キーペアの作成を準備するには
- キーペア 1: プラットフォームキー (PK) を作成します。
- キーペア 2: キー交換キー (KEK) を作成します

- キーペア 3: 署名データベース (db) を作成します
- ブートイメージ (カーネル) にプライベートキーで署名する

キーペアの作成を準備するには

キーペアを作成する前に、キー生成処理で使用するための、グローバルで一意の識別子 (GUID) を作成します。

1. インスタンスに接続します。
2. シェルプロンプトで、次のコマンドを実行します。

```
uuidgen --random > GUID.txt
```

キーペア 1: プラットフォームキー (PK) を作成します。

PK は UEFI 安全ブート インスタンスの信頼のルートです。プライベート PK は KEK を更新するために使用されます。このキーはその後、認証されたキーを署名データベース (db) に追加するためにも使用されます。

キーペアの作成には、X.509 標準が適用されます。標準の詳細については、ウィキペディアの「X.509」を参照してください。

PK を作成するには

1. キーを作成します。変数 PK に名前を付ける必要があります

```
openssl req -newkey rsa:4096 -nodes -keyout PK.key -new -x509 -sha256 -days 3650 -
subj "/CN=Platform key/" -out PK.crt
```

   以下の各パラメータが指定されます。

   - `-keyout PK.key` – プライベートキーファイル。
   - `-days 3650` – 証明書が有効な日数。
   - `-out PK.crt` – UEFI 変数の作成に使用される証明書。
   - CN=*Platform key* – キーの共通名 (CN)。ここでは、*##########* の代わりに組織の独自の名前を入力できます。
2. 証明書を作成します。

```
openssl x509 -outform DER -in PK.crt -out PK.cer
```

3. 証明書を UEFI 署名リストに変換します。

```
cert-to-efi-sig-list -g "$(< GUID.txt)" PK.crt PK.esl
```

4. UEFI 署名リストに (自己署名の) プライベート PKで署名します。

```
sign-efi-sig-list -g "$(< GUID.txt)" -k PK.key -c PK.crt PK PK.esl PK.auth
```

キーペア 2: キー交換キー (KEK) を作成します

プライベート KEK は db にキーを追加するために使用されます。このデータベースは、システム上で起動が許可された署名のリストです。

KEK を作成するには

1. キーを作成します。

```
openssl req -newkey rsa:4096 -nodes -keyout KEK.key -new -x509 -sha256 -days 3650 -
subj "/CN=Key Exchange Key/" -out KEK.crt
```

2. 証明書を作成します。

```
openssl x509 -outform DER -in KEK.crt -out KEK.cer
```

3. 証明書を UEFI 署名リストに変換します。

```
cert-to-efi-sig-list -g "$(< GUID.txt)" KEK.crt KEK.esl
```

4. プライベート PK で署名リストに署名します。

```
sign-efi-sig-list -g "$(< GUID.txt)" -k PK.key -c PK.crt KEK KEK.esl KEK.auth
```

キーペア 3: 署名データベース (db) を作成します

db リストには、システム上で起動することが認証されているキーが記載されています。このリストを変更する場合は、プライベート KEK が必要です。ブートイメージは、このステップで作成したプライベートキーで署名されます。

db を作成するには

1. キーを作成します。

```
openssl req -newkey rsa:4096 -nodes -keyout db.key -new -x509 -sha256 -days 3650 -
subj "/CN=Signature Database key/" -out db.crt
```

2. 証明書を作成します。

```
openssl x509 -outform DER -in db.crt -out db.cer
```

3. 証明書を UEFI 署名リストに変換します。

```
cert-to-efi-sig-list -g "$(< GUID.txt)" db.crt db.esl
```

4. プライベート KEK を使用して署名リストに署名します。

```
sign-efi-sig-list -g "$(< GUID.txt)" -k KEK.key -c KEK.crt db db.esl db.auth
```

ブートイメージ (カーネル) にプライベートキーで署名する

Ubuntu 22.04 の場合、以下のイメージに署名が必要です。

```
/boot/efi/EFI/ubuntu/shimx64.efi
/boot/efi/EFI/ubuntu/mmx64.efi
/boot/efi/EFI/ubuntu/grubx64.efi
/boot/vmlinuz
```

イメージに署名するには

イメージに署名するには、以下の構文を使用します。

```
sbsign --key db.key --cert db.crt --output /boot/vmlinuz /boot/vmlinuz
```

> **Note**
>
> 署名は、すべての新しいカーネルに対して行う必要があります。通常、*/boot/vmlinuz* は、最近インストールしたカーネルへのシンボリックリンクとなります。

ブートチェーンおよび必要なイメージは、ディストリビューションのドキュメントで確認してください。

[1] ArchWiki コミュニティの皆さんから提供された、すべての作業に感謝します。PK の作成、KEK の作成、DB の作成、およびイメージへの署名のためのコマンドは、Creating keys から提供されており、ArchWiki メンテナンスチームおよび (または) ArchWiki のコントリビューターによって作成されたものです。

タスク 2 – オプション A: インスタンス内から変数ストアにキーを追加する

3 つのキーペアの作成が完了すると、以下のステップにより、インスタンスに接続し、インスタンス内から変数ストアにキーを追加できるようになります。または、the section called “タスク 2 – オプション B: 値が事前に設定済みの変数ストアを含むバイナリ BLOB を作成する” の手順を完了します。

オプション A のステップ:

- ステップ 1: UEFI 安全ブート をサポートするインスタンスを起動する
- ステップ 2: UEFI 安全ブート をサポートするようにインスタンスを設定する
- ステップ 3: インスタンスから AMI を作成する

ステップ 1: UEFI 安全ブート をサポートするインスタンスを起動する

次の前提条件の下でインスタンスを起動することで、UEFI 安全ブート をサポートするようにインスタンスを構成する準備が整います。インスタンスでの UEFI 安全ブート のサポートは、起動時にみ有効化が可能で、その後に有効にすることはできません。

前提条件

- AMI - Linux AMI では、UEFI ブートモードをサポートする必要があります。AMI が UEFI ブートモードをサポートしていることを確認するには、AMI ブートモードパラメータで uefi を指定する必要があります。詳細については、「Amazon EC2 AMI のブートモードパラメータを確認する」を参照してください。

AWS では、重力子 ベースのインスタンスタイプ向けに UEFI をサポートするように構成された Linux AMI のみを提供します。AWS では、UEFI ブートモードをサポートする x86_64 Linux AMI を提供していません。すべてのアーキテクチャで UEFI ブートモードをサポートするように独自の AMI を構成できます。UEFI ブートモードをサポートするように AMI を設定するには、独自の AMI に対して、複数の設定ステップを実行する必要があります。詳細については、「Amazon EC2 AMI のブートモードを設定する」を参照してください。

- インスタンスタイプ — UEFI をサポートするすべての仮想インスタンスタイプは、UEFI 安全ブート もサポートします。ベアメタルインスタンスタイプは UEFI 安全ブート をサポートしていません。UEFI 安全ブート をサポートするインスタンスタイプについては、UEFI ブートモードの要件を参照してください。
- UEFI 安全ブート の立ち上げ後に、インスタンスの起動を行います。UEFI 安全ブート のサポートが可能なのは、2022 年 5 月 10 日 (UEFI 安全ブート リリース日) より後に起動されたインスタンスのみです。

インスタンスを起動したら、UEFI データが存在するかどうかを調べ、UEFI 安全ブート のサポートを設定できる状態であることを確認します (ステップ 2 に進みます)。UEFI データが見つかれば、不揮発性データが保持されていることになります。

インスタンスがステップ 2 に進める状態かどうかを確認するには

get-instance-uefi-data コマンドを使用して、インスタンス ID を指定します。

```
aws ec2 get-instance-uefi-data --instance-id i-1234567890abcdef0
```

出力に UEFI データが含まれていれば、そのインスタンスはステップ 2 に進める状態です。空の出力が表示される場合、そのインスタンスで UEFI 安全ブート をサポートする設定は行えません。この状態は、UEFI 安全ブート サポートが利用可能になる前に起動されたインスタンスで発生します。この場合、新しいインスタンスを起動して再試行します。

ステップ 2: UEFI 安全ブート をサポートするようにインスタンスを設定する

インスタンスの UEFI 変数ストアにキーペアを登録します。

> **Warning**
>
> ブートイメージへの署名は、キーの登録後に行う必要があります。そうしないと、インスタンスを起動できなくなります。

署名済みの UEFI 署名リスト (PK、KEK、および db) は、作成後に UEFI ファームウェアに登録する必要があります。

PK 変数に対する書き込みは、以下の場合にのみ行うことができます。

- SetupMode変数が1の場合、PKはまだ登録されていない。これは、次のコマンドを使用して確認します。1 または 0 のどちらかが出力されている。

```
efivar -d -n 8be4df61-93ca-11d2-aa0d-00e098032b8c-SetupMode
```

- 新しい PK が、既存の PK のプライベートキーによって署名されている。

UEFI 変数ストアにキーを登録するには

インスタンスで以下のコマンドを実行する必要があります。

SetupMode が有効 (値が 1) になっていれば、インスタンスで以下のコマンドを実行することでキーを登録できます。

```
[ec2-user ~]$ efi-updatevar -f db.auth db
```

```
[ec2-user ~]$ efi-updatevar -f KEK.auth KEK
```

```
[ec2-user ~]$ efi-updatevar -f PK.auth PK
```

UEFI 安全ブート が有効になっていることを確認するには

UEFI 安全ブート が有効であることを確認するには、「Amazon EC2 インスタンスで UEFI 安全ブート が有効化されているかどうかを確認する」に示したステップに従います。

この段階で、get-instance-uefi-data CLI コマンドにより UEFI 変数ストアをエクスポートすることが可能です。あるいは、次のステップに進みブートイメージに署名して、それを UEFI 安全ブート-対応のインスタンスで再起動できます。

ステップ 3: インスタンスから AMI を作成する

インスタンスから AMI を作成するには、コンソールまたは CreateImage API、CLI、または SDK を使用します。コンソールでのステップについては、「Amazon EBS-backed AMI を作成する」を参照してください。API でのステップについては、「CreateImage」を参照してください。

> **Note**
>
> `CreateImage` API は、インスタンスの UEFI 変数ストアを AMI に自動的にコピーします。コンソールは `CreateImage` API を使用します。この AMI を使用してインスタンスを起動した場合、インスタンスにも AMI と同じ UEFI 変数ストアが含まれます。

タスク 2 – オプション B: 値が事前に設定済みの変数ストアを含むバイナリ BLOB を作成する

3 つのキーペアの作成後、事前に値が設定され UEFI 安全ブート キーが指定された変数ストアを含む、バイナリ BLOB を作成できます。または、the section called “タスク 2 – オプション A: インスタンス内から変数ストアにキーを追加する” の手順を完了します。

> **Warning**
>
> ブートイメージへの署名は、キーの登録前に行う必要があります。そうしないと、インスタンスを起動できなくなります。

オプション B のステップ:

- ステップ 1: 変数ストアを新規で作成するか既存の変数ストアを更新する
- ステップ 2: AMI の作成時にバイナリ BLOB をアップロードする

ステップ 1: 変数ストアを新規で作成するか既存の変数ストアを更新する

python-uefivars ツールを使用すれば、インスタンスを実行しなくてもオフラインで変数ストアを作成できます。このツールでは、キーから新しい変数ストアを作成できます。現在、このスクリプトでは、EDK2 形式、AWS 形式、および上位レベルのツールで編集しやすい JSON 表現がサポートされています。

インスタンスを実行せずに変数ストアをオフラインで作成するには

1. 次のリンクからツールをダウンロードします。

   ```
   https://github.com/awslabs/python-uefivars
   ```

2. 次のコマンドを実行して、キーから新しい変数ストアを作成します。これにより、base64 でエンコードされたバイナリ BLOB が、*`your_binary_blob`* bin として作成されます。このツールでは、`-I` パラメータ経由でバイナリ BLOB を更新することもできます。

```
./uefivars.py -i none -o aws -O your_binary_blob.bin -P PK.esl -K KEK.esl --db
 db.esl --dbx dbx.esl
```

ステップ 2: AMI の作成時にバイナリ BLOB をアップロードする

register-image により、UEFI 変数ストアデータを渡します。--uefi-data パラメータではバイナリ BLOB を指定し、また --boot-mode パラメータでは uefi を指定します。

```
aws ec2 register-image \
    --name uefi_sb_tpm_register_image_test \
    --uefi-data $(cat your_binary_blob.bin) \
    --block-device-mappings "DeviceName=/dev/sda1,Ebs=
 {SnapshotId=snap-0123456789example,DeleteOnTermination=true}" \
    --architecture x86_64 \
    --root-device-name /dev/sda1 \
    --virtualization-type hvm \
    --ena-support \
    --boot-mode uefi
```

UEFI 安全ブート の AWS バイナリ BLOB を作成する

次のステップにより、AMI の作成中に UEFI 安全ブート 変数をカスタマイズすることができます。このステップで使用される KEK は、2021 年 9 月現在のものです。Microsoft により KEK が更新された場合は、その最新の KEK を使用する必要があります。

AWS でバイナリ BLOB を作成するには

1. 空の PK 署名リストを作成します。

```
touch empty_key.crt
cert-to-efi-sig-list empty_key.crt PK.esl
```

2. KEK 証明書をダウンロードします。

```
https://go.microsoft.com/fwlink/?LinkId=321185
```

3. UEFI 署名リスト (`siglist`) で KEK 証明書をラップします。

```
sbsiglist --owner 77fa9abd-0359-4d32-bd60-28f4e78f784b --type x509 --output
 MS_Win_KEK.esl MicCorKEKCA2011_2011-06-24.crt
```

4. Microsoft から db 証明書をダウンロードします。

```
https://www.microsoft.com/pkiops/certs/MicWinProPCA2011_2011-10-19.crt
https://www.microsoft.com/pkiops/certs/MicCorUEFCA2011_2011-06-27.crt
```

5. db 署名リストを生成します。

```
sbsiglist --owner 77fa9abd-0359-4d32-bd60-28f4e78f784b --type x509 --output
 MS_Win_db.esl MicWinProPCA2011_2011-10-19.crt
sbsiglist --owner 77fa9abd-0359-4d32-bd60-28f4e78f784b --type x509 --output
 MS_UEFI_db.esl MicCorUEFCA2011_2011-06-27.crt
cat MS_Win_db.esl MS_UEFI_db.esl > MS_db.esl
```

6. Unified Extensible Firmware Interface Forum は、DBX ファイルを提供しなくなりました。これらは、Microsoft によって GitHub で提供されるようになりました。https://github.com/microsoft/secureboot_objects の Microsoft セキュアブート更新リポジトリから最新の DBX 更新をダウンロードしてください。
7. 署名付きの更新バイナリを解凍します。

   以下のスクリプトコンテンツを使用して SplitDbxContent.ps1 を作成します。または、Install-Script -Name SplitDbxContent を使用して PowerShell Gallery からスクリプトをインストールすることもできます。

```
<#PSScriptInfo

.VERSION 1.0

.GUID ec45a3fc-5e87-4d90-b55e-bdea083f732d

.AUTHOR Microsoft Secure Boot Team

.COMPANYNAME Microsoft

.COPYRIGHT Microsoft

.TAGS Windows Security

.LICENSEURI
```

```
.PROJECTURI

.ICONURI

.EXTERNALMODULEDEPENDENCIES

.REQUIREDSCRIPTS

.EXTERNALSCRIPTDEPENDENCIES

.RELEASENOTES
Version 1.0: Original published version.

#>

<#
.DESCRIPTION
 Splits a DBX update package into the new DBX variable contents and the signature
 authorizing the change.
 To apply an update using the output files of this script, try:
 Set-SecureBootUefi -Name dbx -ContentFilePath .\content.bin -SignedFilePath .
\signature.p7 -Time 2010-03-06T19:17:21Z -AppendWrite'
.EXAMPLE
.\SplitDbxAuthInfo.ps1 DbxUpdate_x64.bin
#>


# Get file from script input
$file  = Get-Content -Encoding Byte $args[0]

# Identify file signature
$chop = $file[40..($file.Length - 1)]
if (($chop[0] -ne 0x30) -or ($chop[1] -ne 0x82 )) {
    Write-Error "Cannot find signature"
    exit 1
}

# Signature is known to be ASN size plus header of 4 bytes
$sig_length = ($chop[2] * 256) + $chop[3] + 4
$sig = $chop[0..($sig_length - 1)]

if ($sig_length -gt ($file.Length + 40)) {
    Write-Error "Signature longer than file size!"
```

```
    exit 1
}

# Content is everything else
$content = $file[0..39] + $chop[$sig_length..($chop.Length - 1)]

# Write signature and content to files
Set-Content -Encoding Byte signature.p7 $sig
Set-Content -Encoding Byte content.bin $content
```

次のスクリプトを使用して、署名付き DBX ファイルを解凍します。

```
PS C:\Windows\system32> SplitDbxContent.ps1 .\dbx.bin
```

これにより、signature.p7 および content.bin の 2 つのファイルが生成されます。content.bin は次のステップで使用します。

8. uefivars.py スクリプトを使用して UEFI 変数ストアを作成します。

```
./uefivars.py -i none -o aws -O uefiblob-microsoft-keys-empty-pk.bin -P ~/PK.esl -K
 ~/MS_Win_KEK.esl --db ~/MS_db.esl  --dbx ~/content.bin
```

9. バイナリ BLOB と UEFI 変数ストアを確認します。

```
./uefivars.py -i aws -I uefiblob-microsoft-keys-empty-pk.bin -o json | less
```

10. 再度、同じツールに渡すと、BLOB を更新できます。

```
./uefivars.py -i aws -I uefiblob-microsoft-keys-empty-pk.bin -o aws -O uefiblob-
microsoft-keys-empty-pk.bin -P ~/PK.esl -K ~/MS_Win_KEK.esl --db ~/MS_db.esl  --dbx
 ~/content.bin
```

正常な出力

```
Replacing PK
Replacing KEK
Replacing db
Replacing dbx
```

EBS-backed AMI での暗号化の利用

Amazon EBS スナップショットを使用した AMI は Amazon EBS 暗号化の利点を活かすことができます。データおよびルートボリュームの両方のスナップショットを暗号化して AMI にアタッチできます。インスタンスを起動し、完全な EBS 暗号化サポートも含めてイメージをコピーできます。これらのオペレーションの暗号化パラメータは、AWS KMS が利用できるすべてのリージョンでサポートされています。

暗号化された EBS ボリュームを持つ EC2 インスタンスは、他のインスタンスと同様に AMIs から起動します。また、暗号化されていない EBS スナップショットでバックアップされている AMI からインスタンスを起動するとき、起動中に一部またはすべてのボリュームを暗号化できます。

EBS ボリュームと同様に、AMI のスナップショットはデフォルトの AWS KMS key または指定したカスタマーマネージド型キーで暗号化できます。いずれの場合も、選択した KMS キー を使用するためのアクセス権限が必要です。

暗号化されたスナップショットを持つ AMI は、AWS アカウント間で共有できます。詳細については、Amazon EC2 での共有 AMI の使用方法について を参照してください。

EBS-backed AMI トピックでの暗号化

- インスタンスの起動シナリオ
- イメージコピーのシナリオ

インスタンスの起動シナリオ

AMI から Amazon EC2 インスタンスを起動するには、`RunInstances` または直接 Amazon EC2 API や CLI を使用して、AWS マネジメントコンソール アクションを実行します。その際、ブロックデバイスマッピングから提供されるパラメータを指定します。詳細については、「Amazon EC2 インスタンスでのボリュームのブロックデバイスマッピング」を参照してください。AWS CLI からブロックデバイスマッピングを制御する例については、「EC2 インスタンスを起動、リスト、および終了する」を参照してください。

デフォルトでは、明示的な暗号化パラメータがない場合、AMI のソーススナップショットから EBS ボリュームを復元しているときに、`RunInstances` アクションは AMI のソーススナップショットの既存の暗号化状態を維持します。デフォルトでの暗号化が有効になっている場合、AMI から作成したすべてのボリュームが暗号化されます (作成元のスナップショットが暗号化されているかどうかは

関係ありません)。デフォルトでの暗号化が有効にされていない場合、インスタンスは AMI の暗号化状態を維持します。

インスタンスを起動し、同時に、暗号化パラメータを指定して、新しい暗号化状態を生成されるボリュームに適用することもできます。そのため、以下の動作が観察されます。

暗号化パラメータなしでの起動

- デフォルトでの暗号化が有効にされている場合を除き、暗号化されていないスナップショットは、暗号化されていないボリュームに復元されます。デフォルトでの暗号化が有効にされている場合は、新しく作成されるすべてのボリュームが暗号化されます。
- 所有する暗号化されたスナップショットは、同じ KMS キー に暗号化されるボリュームに復元されます。
- 所有していない (例えば、AMI が共有されている) 暗号化されたスナップショットは、ユーザーの AWS アカウントのデフォルト KMS キーで暗号化されているボリュームに復元されます。

デフォルトの動作は、暗号化パラメータを指定してオーバーライドできます。利用できるパラメータは、`Encrypted` と `KmsKeyId` です。`Encrypted` パラメータのみを設定すると、次のような結果になります。

`Encrypted` を設定し、**`KmsKeyId`** を指定しない場合のインスタンス起動動作

- 暗号化されていないスナップショットは、ユーザーの AWS アカウントのデフォルト KMS キーで暗号化されている EBS ボリュームに復元されます。
- 所有する暗号化されたスナップショットは、同じ KMS キー により暗号化された EBS ボリュームに復元されます。(つまり、`Encrypted` パラメータには効果がありません。)
- 所有していない (つまり、AMI が共有されている) 暗号化されたスナップショットは、ユーザーの AWS アカウントのデフォルト KMS キーで暗号化されているボリュームに復元されます。(つまり、`Encrypted` パラメータには効果がありません。)

`Encrypted` と `KmsKeyId` 両方のパラメータを設定すると、暗号化オペレーションにデフォルトではない KMS キー を指定できます。結果として次のように動作します。

`Encrypted` と **`KmsKeyId`** が両方設定されたインスタンス

- 暗号化されていないスナップショットは、指定された KMS キー により暗号化された EBS ボリュームに復元されます。

- 暗号化されたスナップショットは、元の KMS キー ではなく、指定された KMS キー に暗号化された EBS ボリュームに復元されます。

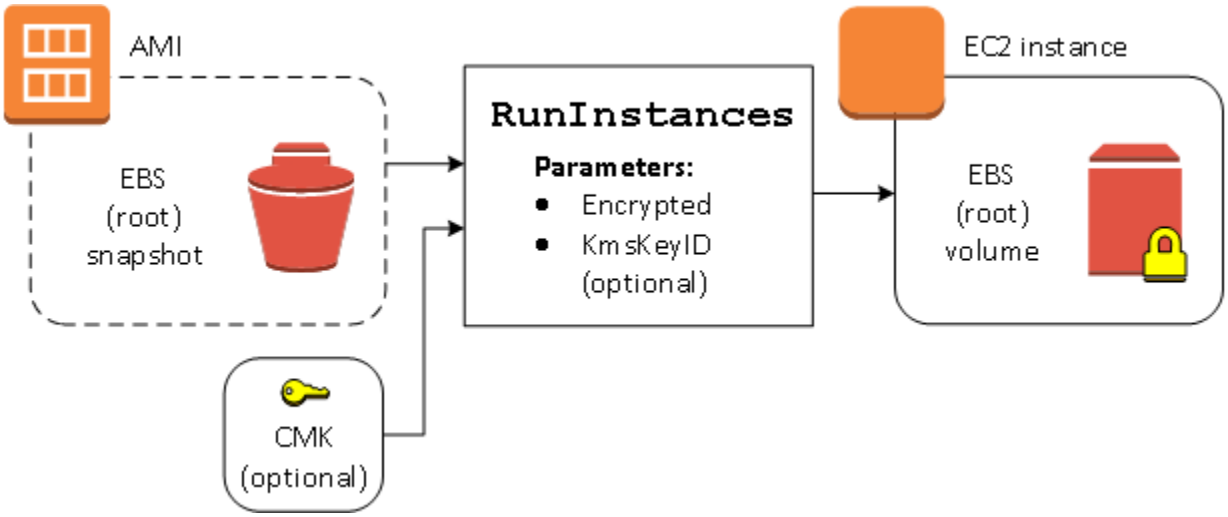
Encrypted パラメータも設定せずに KmsKeyIdを送信するとエラーが発生します。

以下のセクションでは、デフォルトではない暗号化パラメータを使用して AMI からインスタンスを起動する例を示します。これらの各シナリオでは、RunInstances アクションに指定するパラメータにより、スナップショットからボリュームを復元中に暗号化の状態が変化します。

コンソールを使用して AMI からインスタンスを起動する方法については、「Amazon EC2 インスタンスの起動」を参照してください。

起動時にボリュームを暗号化する

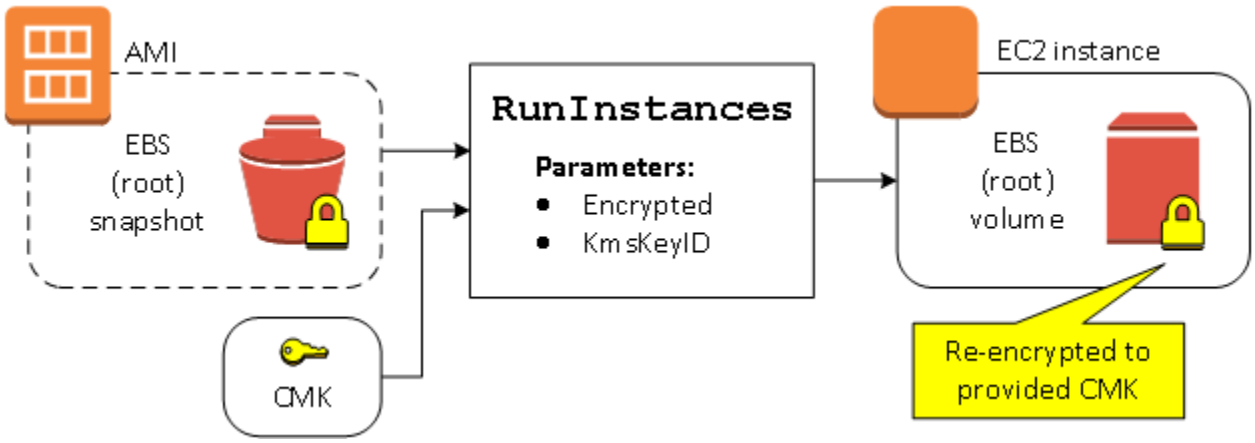
この例では、暗号化されていないスナップショットでバックアップされた AMI を使用して、暗号化された EBS ボリュームのある EC2 インスタンスを起動します。



Encrypted パラメータのみを使用すると、このインスタンスのボリュームが暗号化されます。KmsKeyId パラメータの指定はオプションです。KMS キー ID を指定しない場合、AWS アカウントのデフォルト KMS キーを使用して、ボリュームを暗号化します。所有する別の KMS キー にボリュームを暗号化するには、KmsKeyId パラメータを指定します。

起動時にボリュームを再暗号化する

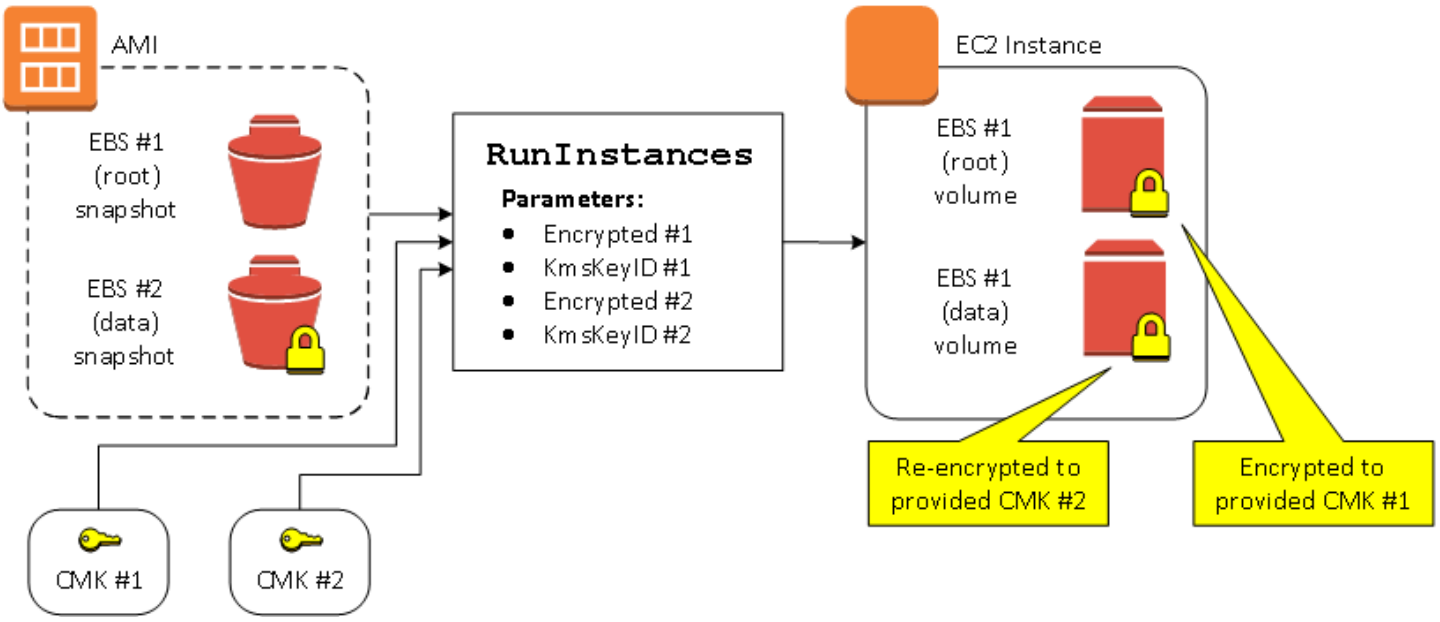
この例では、暗号化されたスナップショットでバックアップされた AMI を使用して、新しい KMS キー により暗号化された EBS ボリュームのある EC2 インスタンスを起動します。



AMI を所有していて、暗号化パラメータを指定しない場合、作成されるインスタンスではスナップショットと同じ KMS キーでボリュームが暗号化されます。他のアカウントから共有された AMI に対して暗号化パラメータを指定しない場合、ボリュームはデフォルト KMS キー により暗号化されます。図のように暗号化パラメータが指定されている場合、ボリュームは指定された KMS キー により暗号化されます。

起動時に複数のボリュームの暗号化状態を変更する

このより複雑な例では、複数のスナップショット (暗号化状態はそれぞれ異なります) でバックアップされた AMI を使用して、新しく暗号化されたボリュームと再暗号化されたボリュームがある EC2 インスタンスを起動します。



このシナリオでは、RunInstances アクションにソーススナップショットそれぞれに対する暗号化パラメータが指定されます。可能な暗号化パラメータがすべて指定されると、AMI を所有しているかどうかに関係なく、作成されるインスタンスは同じです。

イメージコピーのシナリオ

Amazon EC2 AMI をコピーするには、CopyImage または直接 Amazon EC2 API や CLI を使用して、AWS マネジメントコンソール アクションを実行します。

デフォルトでは、明示的な暗号化パラメータがない場合、コピー中 CopyImage アクションは AMI のソーススナップショットの既存の暗号化状態を維持します。AMI をコピーし、同時に、暗号化パラメータを指定して、新しい暗号化状態を関連付けられている EBS スナップショットに適用することもできます。そのため、以下の動作が観察されます。

暗号化パラメータなしでのコピー

- デフォルトでの暗号化が有効にされている場合を除き、暗号化されていないスナップショットは、別の暗号化されていないスナップショットにコピーされます。デフォルトでの暗号化が有効にされている場合は、新しく作成されるすべてのスナップショットが暗号化されます。
- 所有する暗号化されたスナップショットは、同じ KMS キー で暗号化されたスナップショットにコピーされます。
- 所有していない (例えば、AMI が共有されている) 暗号化されたスナップショットは、ユーザーの AWS アカウントのデフォルト KMS キーで暗号化されているスナップショットにコピーされます。

これらすべてのデフォルトの動作は、暗号化パラメータを指定してオーバーライドできます。利用できるパラメータは、Encrypted と KmsKeyId です。Encrypted パラメータのみを設定すると、次のような結果になります。

Encrypted を設定し、**KmsKeyId** を指定しない場合のコピーイメージ動作

- 暗号化されていないスナップショットは、AWS アカウントのデフォルト KMS キーで暗号化されたスナップショットにコピーされます。
- 暗号化されたスナップショットは、同じ KMS キー により暗号化されたスナップショットにコピーされます。(つまり、Encrypted パラメータには効果がありません。)
- 所有していない (例えば、AMI が共有されている) 暗号化されたスナップショットは、ユーザーの AWS アカウントのデフォルト KMS キーで暗号化されているボリュームにコピーされます。(つまり、Encrypted パラメータには効果がありません。)

Encrypted と KmsKeyId 両方のパラメータを設定すると、暗号化オペレーションにカスタマー管理 KMS キー を指定できます。結果として次のように動作します。

Encrypted と KmsKeyId の両方を設定した場合のコピーイメージ動作

- 暗号化されていないスナップショットは、指定された KMS キー により暗号化されたスナップショットにコピーされます。
- 暗号化されたスナップショットは、元の KMS キー ではなく、指定された KMS キー に暗号化されたスナップショットにコピーされます。

Encrypted パラメータも設定せずに KmsKeyIdを送信するとエラーが発生します。

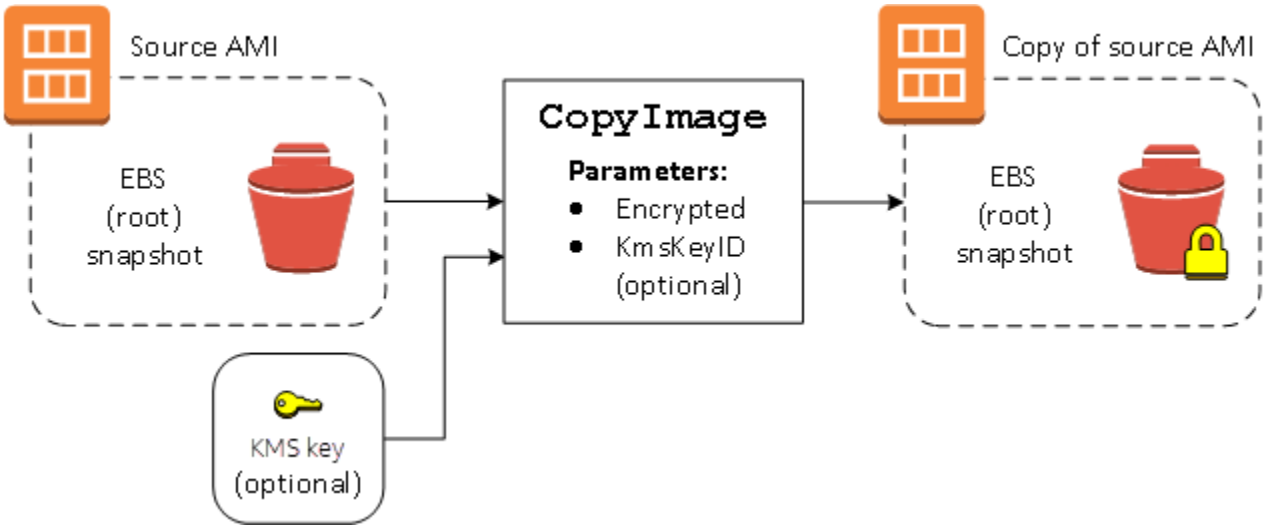
以下のセクションでは、デフォルトではない暗号化パラメータを使用して AMI をコピーし、結果として暗号化状態が変化する例を示します。

コンソールを使用する手順の詳細については、「Amazon EC2 AMI のコピー」を参照してください。

コピー時に暗号化されていないイメージを暗号化する

このシナリオでは、暗号化されていないルートスナップショットでバックアップされた AMI は、暗号化されたルートスナップショットを持つ AMI にコピーされます。CopyImage アクションは、カスタマー管理された型キーなど、2 つの暗号化パラメータで呼び出されます。その結果、ルートスナップショットの暗号化ステータスが変更され、ターゲット AMI はソーススナップショットと同じデータを含むルートスナップショットにバックアップされますが、指定されたキーを使用して暗号化されます。両方の AMI でスナップショットのストレージコストと、いずれかの AMI から起動するインスタンスの料金が発生します。

> **Note**
>
> デフォルトで暗号化を有効にすると、AMI 内のすべてのスナップショットで Encrypted パラメータを true に設定したのと同じ効果があります。



`Encrypted` パラメータを設定すると、このインスタンスの単一のスナップショットが暗号化されます。`KmsKeyId` パラメータを指定しない場合は、デフォルトのカスタマーマネージド型キーを使用して、スナップショットのコピーが暗号化されます。

> **Note**
>
> 複数のスナップショットがあるイメージをコピーして、それぞれの暗号化状態を個々に設定することもできます。

Amazon EC2 での共有 AMI の使用方法について

共有 AMI は、デベロッパーが作成し、他のデベロッパーが利用できるようにした AMI です。Amazon EC2 を始める最も簡単な方法は、必要なコンポーネントが含まれている共有 AMI を使用して、カスタムコンテンツを追加することです。独自の AMI を作成し、他のユーザーと共有することもできます。

共有 AMI は、ご自分の判断で使用してください。Amazon は、他の Amazon EC2 ユーザーとの間で共有される AMI の統合性や安全性を保証できません。そのため、共有 AMI を取り扱う際は、ご自分のデータセンターに外部のコードをデプロイすることを検討する場合と同じように、充分な注意を払う必要があります。検証済みのプロバイダーなど、信頼できるソースから AMI を取得することをお勧めします。

検証済みプロバイダー

Amazon EC2 コンソールでは、Amazon または検証済み Amazon パートナーが所有するパブリック AMI には [検証済みプロバイダー] のマークが付されます。

また、describe-images AWS CLI コマンドを使用して、検証済みプロバイダーからのパブリック AMI を識別することもできます。Amazon または検証済みパートナーが所有するパブリックイメージには、`amazon`、`aws-backup-vault`、`aws-marketplace` のいずれかのエイリアス付き所有者が存在します。CLI 出力では、これらの値が `ImageOwnerAlias` について表示されます。他のユーザーは、AMI にエイリアスを設定できません。これを利用すれば、Amazon または検証済みパートナーから AMI を簡単に見つけられます。

検証済みプロバイダーになるには、AWS Marketplace で販売者として登録する必要があります。登録が完了すると、AMI を AWS Marketplace で一覧表示できます。詳細については、「AWS Marketplace 販売者ガイド」の「販売者としての開始方法」および「AMI ベースの製品」を参照してください。

共有 AMI のトピック

- Amazon EC2 インスタンスで使用する共有 AMI を検索する
- Linux 用の共有 AMI を使用する準備をする
- 許可された AMI を使用して Amazon EC2 で AMI の検出と使用を制御する
- Amazon EC2 で使用するために AMI を公開する
- AMI のパブリックアクセスのブロックについて
- 組織および組織単位での共有 AMI の使用
- 特定の AWS アカウントとの AMI の共有
- お客様の AWS アカウント と AMI の共有をキャンセルする
- 共有 Linux AMI の作成に関する推奨事項

他のトピックに関する情報をお探しの場合は

- AMI の作成については、「the section called “Amazon S3-backed AMI を作成する”」または「the section called “AMI を作成する”」を参照してください。
- AWS Marketplace でのアプリケーションの構築、配信、保守の詳細については、AWS Marketplace ドキュメントをご参照ください。

Amazon EC2 インスタンスで使用する共有 AMI を検索する

Amazon EC2 コンソールまたはコマンドラインを使用すると、Amazon EC2 インスタンスで使用するパブリックまたはプライベートの共有 AMI を検索できます。

AMI はリージョンのリソースです。共有 AMI (パブリックまたはプライベート) を検索するときには、それの共有元のリージョンから実行する必要があります。AMI を他のリージョンで利用できるようにするには、AMI をそのリージョンにコピーし、共有します。詳細については、「Amazon EC2 AMI のコピー」を参照してください。

Console

コンソールには AMI フィルターフィールドが用意されています。[検索] フィールドで指定されたフィルターを使用して、検索範囲を指定することもできます。

共有または AMI を見つけるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. 最初のフィルターで、次のいずれかのオプションを選択します。
   - プライベートイメージ – 共有されているすべての AMI を一覧表示します。
   - パブリックイメージ – すべてのパブリック AMI を一覧表示します。
4. (オプション) Amazon の公開イメージのみを表示するには、[検索] フィールドを選択し、メニューオプションから [所有者エイリアス] 、[=]、[Amazon] の順に選択します。
5. (オプション) フィルターを追加して、要件を満たす AMI に検索範囲を設定します。

「検証済みプロバイダー」で共有パブリック AMI を見つける方法

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI カタログ] を選択してください。
3. [コミュニティ AMI] を選択してください。
4. [結果を絞り込む] ペインで、[検証済みプロバイダー] を選択します。[検証済みプロバイダー] のラベルは、AMI が Amazon または検証済みパートナーからのものであることを示します。

AWS CLI

AMI を一覧表示するには、- describe-images マンド () を使用します。次の例のように、興味のある種類の AMI に絞って一覧表示できます。

すべてのパブリック AMI を一覧表示する方法

次のコマンドは、所有しているパブリック AMI を含むすべてのパブリック AMI を一覧表示します。

```
aws ec2 describe-images --executable-users all
```

明示的な起動許可を持つ AMI を一覧表示する方法

次のコマンドを使用すると、お客様が明示的な起動許可を持つ AMI が一覧表示されます。このリストには、お客様が所有する AMI は含まれていません。

```
aws ec2 describe-images --executable-users self
```

検証済みプロバイダーが所有する AMI を一覧表示する方法

次のコマンドにより、「検証済みプロバイダー」が所有する AMI が一覧表示されます。検証済みプロバイダー (Amazon または検証済みパートナーのいずれか) が所有するパブリック AMI には、アカウントフィールドで amazon、aws-backup-vault、aws-marketplace として表示されるエイリアス付き所有者が存在します。これは、検証済みプロバイダーからの AMI を簡単に見つけるのに役立ちます。他のユーザーは、AMI にエイリアスを設定できません。

```
aws ec2 describe-images \
    --owners amazon aws-marketplace \
    --query 'Images[*].[ImageId]' \
    --output text
```

アカウントが所有する AMI を一覧表示する方法

次のコマンドを実行すると、指定した AWS アカウント が所有する AMI が一覧表示されます。

```
aws ec2 describe-images --owners 123456789012
```

フィルタを使用して AMI をスコープする方法

表示される AMI の数を減らすには、フィルタを使用して、興味のある種類の AMI に限定して表示します。例えば、次のフィルタを使用すると、EBS-backed AMI のみが表示されます。

```
--filters "Name=root-device-type,Values=ebs"
```

PowerShell

AMI を一覧表示するには、Get-EC2Image コマンドレット (Tools for Windows PowerShell) を使用します。次の例のように、興味のある種類の AMI に絞って一覧表示できます。

すべてのパブリック AMI を一覧表示する方法

次のコマンドは、所有しているパブリック AMI を含むすべてのパブリック AMI を一覧表示します。

```
Get-EC2Image -ExecutableUser all
```

明示的な起動許可を持つ AMI を一覧表示する方法

次のコマンドを使用すると、お客様が明示的な起動許可を持つ AMI が一覧表示されます。このリストには、お客様が所有する AMI は含まれていません。

```
Get-EC2Image -ExecutableUser self
```

検証済みプロバイダーが所有する AMI を一覧表示する方法

次のコマンドにより、「検証済みプロバイダー」が所有する AMI が一覧表示されます。検証済みプロバイダー (Amazon または検証済みパートナーのいずれか) が所有するパブリック AMI には、アカウントフィールドで amazon、aws-backup-vault、aws-marketplace として表示されるエイリアス付き所有者が存在します。これは、検証済みプロバイダーからの AMI を簡単に見つけるのに役立ちます。他のユーザーは、AMI にエイリアスを設定できません。

```
Get-EC2Image -Owner amazon aws-marketplace
```

アカウントが所有する AMI を一覧表示する方法

次のコマンドを実行すると、指定した AWS アカウント が所有する AMI が一覧表示されます。

```
Get-EC2Image -Owner 123456789012
```

フィルタを使用して AMI をスコープする方法

表示される AMI の数を減らすには、フィルタを使用して、興味のある種類の AMI に限定して表示します。例えば、次のフィルタを使用すると、EBS-backed AMI のみが表示されます。

```
-Filter @{Name="root-device-type"; Values="ebs"}
```

Linux 用の共有 AMI を使用する準備をする

Linux で共有 AMI を使用する前に、次の手順を実行して、インスタンスへの好ましくないアクセスを許可する認証情報が第三者により事前にインストールされていないことと、機密データを第三者に送信する可能性があるリモートロギングが事前設定されていないことを確認します。システムセキュリティ改善についての詳細は、AMI で使用される Linux ディストリビューションの文書を確認してください。

インスタンスへのアクセスを誤って失わないように、SSH セッションを 2 つ開始して、見覚えのない認証情報を削除し、その後も SSH を使用してインスタンスにログインできることが確認されるまで、2 つ目のセッションを開いておくことをお勧めします。

1. 未許可のパブリック SSH キーを特定し、無効にします。ファイル内の唯一のキーは、AMI の起動に使用したキーである必要があります。次のコマンドを使用すると、authorized_keys ファイルが見つかります。

   ```
   [ec2-user ~]$ sudo find / -name "authorized_keys" -print -exec cat {} \;
   ```

2. ルートユーザーにはパスワードベースの認証を無効にします。sshd_config ファイルを開き、次のように PermitRootLogin 行を編集します。

   ```
   PermitRootLogin without-password
   ```

   または、ルートユーザーとしてインスタンスにログインする機能を無効にできます。

   ```
   PermitRootLogin No
   ```

   sshd サービスを再起動します。

3. インスタンスにログインできるユーザーが他にないか確認します。スーパーユーザー権限を持つユーザーが特に危険です。不明のアカウントがあれば、そのパスワードを削除するか、ロックします。

4. 開いていても使用していないポートと、着信接続をリスニングしている実行中のネットワークサービスをチェックします。
5. 事前設定されているリモートロギングを防ぐには、既存の設定ファイルを削除し、`rsyslog` サービスを再起動してください。例:

```
[ec2-user ~]$ sudo rm /etc/rsyslog.conf
[ec2-user ~]$ sudo service rsyslog restart
```

6. すべての cron ジョブが正当であることを確認します。

セキュリティ上のリスクとして考えられるパブリック AMI を発見した際には、AWS セキュリティチームにご連絡ください。詳細については、「AWS セキュリティセンター」を参照してください。

許可された AMI を使用して Amazon EC2 で AMI の検出と使用を制御する

AWS アカウント での Amazon マシンイメージ (AMI) の検出と使用を制御するには、許可された AMI の機能を使用できます。アカウント内で AMI を表示して使用するために満たす必要がある基準を指定します。条件を有効にすると、インスタンスを起動するユーザーは、指定された条件に準拠する AMI のみを表示し、アクセスできます。例えば、信頼された AMI プロバイダーのリストを条件として指定できます。これらのプロバイダーからの AMI のみが表示され、使用できます。

許可された AMI 設定を有効にする前に、監査モードを有効にして、どの AMI を表示および使用不可にするかをプレビューできます。これにより、必要に応じて条件を絞り込み、目的の AMI のみがアカウント内のユーザーに表示されるようにすることができます。さらに、describe-instance-image-metadata コマンドを実行し、指定した条件を満たしていない AMI で起動されたインスタンスを特定します。この情報は、準拠した AMI を使用するように起動設定を更新する (起動テンプレートで別の AMI を指定するなど) か、これらの AMI を許可するように条件を調整するかどうかを決定するのに役立ちます。

許可された AMI 設定は、アカウントレベルで直接指定するか、宣言ポリシーを使用して指定します。これらの設定は、AMI の使用を制御する各 AWS リージョンで設定する必要があります。宣言型ポリシーを使用すると、複数のリージョンと複数のアカウントで同時に設定を適用できます。宣言ポリシーが使用されている場合、アカウント内で直接設定を変更することはできません。このトピックでは、アカウント内で設定を直接設定する方法について説明します。宣言ポリシーの使用の詳細については、「AWS Organizations ユーザーガイド」の「宣言ポリシー」を参照してください。

> **Note**
>
> 許可された AMI 機能は、アカウントと共有されている AMI とパブリック AMI の検出と使用のみを制御します。アカウントが所有する AMI は制限されません。設定した条件にかかわらず、特定のアカウントで作成した AMI は、常にそのアカウントのユーザーが検出、使用することができます。

許可された AMI の主な利点

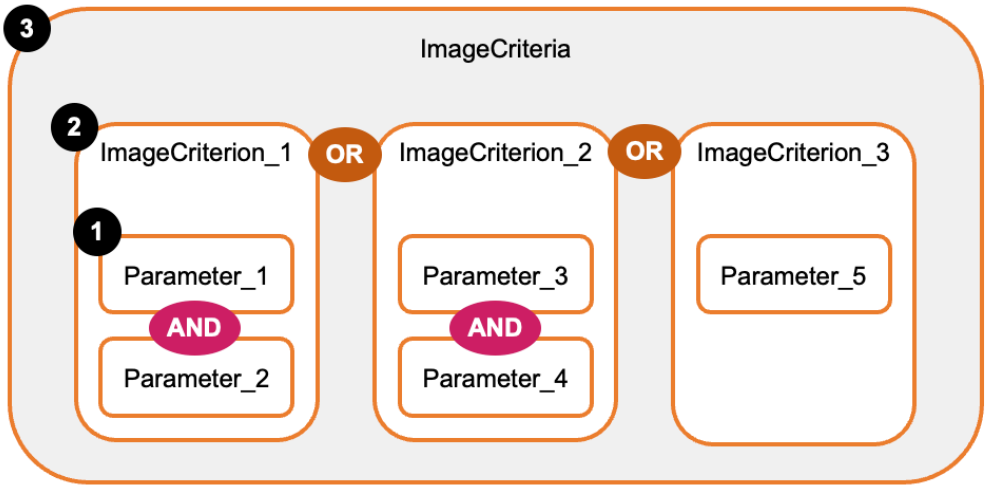
- コンプライアンスとセキュリティ: ユーザーは、指定された基準を満たす AMI のみを検出して使用できるため、準拠されていない AMI を使用するリスクが軽減されます。
- 効率的な管理: 許可される AMI の数を減らすことで、残りの AMI の管理が簡単になり、効率が向上します。
- アカウントレベルの一元化された実装: アカウント内で直接、または宣言ポリシーを使用して、アカウントレベルで許可された AMI 設定を構成します。これにより、アカウント全体で AMI の使用を一元的かつ効率的に制御できます。

内容

- 許可された AMI の仕組み
- 許可された AMI を実装するためのベストプラクティス
- 必要な IAM 許可
- 許可された AMI の設定を管理する

許可された AMI の仕組み

お使いのアカウントで、どの AMI を検出して使用できるかを制御するには、AMI を評価するための一連の基準を定義します。次の図に示すように、基準は 1 つまたは複数の `ImageCriterion` で構成されます。説明は図の後に続きます。



設定には 3 つのレベルがあります。

- 1 – パラメータ値
  - 複数値パラメータ:
    - `ImageProviders`
    - `ImageNames`
    - `MarketplaceProductCodes`

      AMI は、パラメータ内の任意の値と一致した場合に許可されます。

      例: `ImageProviders` = amazon またはアカウント 111122223333 または アカウント 444455556666 (パラメータ値の評価ロジックは図に示されていません。)
    - `ImageWatermarks`
  - 単一値パラメータ:
    - `CreationDateCondition`
    - `DeprecationTimeCondition`
- 2 – `ImageCriterion`
  - 複数のパラメータを AND ロジックでグループ化します。
  - AMI が許可されるには、`ImageCriterion` 内のすべてのパラメータと一致する必要があります。
  - 例: `ImageProviders` = amazon AND `CreationDateCondition` = 300 日以下
- 3 – `ImageCriteria`
  - 複数の `ImageCriterion` を OR ロジックでグループ化します。

- AMI は、任意の ImageCriterion と一致した場合に許可されます。
- AMI を評価するための完全な設定を形成します。

トピック

- 許可された AMI のパラメータ
- 許可された AMI の設定
- 基準の評価方法
- 制限
- 許可されている操作

許可された AMI のパラメータ

次のパラメータを設定すると ImageCriterion を作成できます。

`ImageProviders`

AMI が許可されている AMI プロバイダーです。

有効な値は、次のように AWS、および AWS アカウント ID で定義されたエイリアスであり、次のようなものがあります。

- amazon – Amazon または検証済みのプロバイダーによって作成された AMI を識別するエイリアス
- aws-marketplace – AWS Marketplace で検証済みプロバイダーによって作成された AMI を識別するエイリアス
- aws-backup-vault – 論理エアギャップバックアップボールトアカウントに存在する AWS バックアップ AMI を識別するエイリアス。AWS バックアップの論理エアギャップボールト機能を使用する場合は、このエイリアスが AMI プロバイダーとして含まれていることを確認してください。
- AWS アカウント ID – 1 つ以上の 12 桁の AWS アカウント ID
- none – アカウントによって作成された AMI のみを検出して使用できることを示します。パブリックまたは共有 AMI を検出して使用することはできません。これを指定すると、他の条件は指定できなくなります。

`ImageNames`

完全一致またはワイルドカード (? または *) を使用した、許可された AMI の名前。

`MarketplaceProductCodes`

許可された AMI の AWS Marketplace 製品コード。

`CreationDateCondition`

許可された AMI の最大有効期間。

`DeprecationTimeCondition`

許可された AMI が非推奨になってからの最大経過期間。

`ImageWatermarks`

ウォーターマークフィルターのリスト。フィルターでは、AMI 上のウォーターマークがフィルターエントリ (OR ロジック) と一致する場合に AMI が許可されます。各フィルターには、次のフィールドが含まれます。

- `WatermarkKey` – 一致するウォーターマークキー (`account-id:watermark-name`)
- `SourceImageRegion` – 最初にウォーターマークをアタッチした AMI のリージョン。
- `MaximumDaysSinceSourceImageCreated` – ウォーターマークを適用したときのソース AMI の最大有効期間
- `MaximumDaysSinceWatermarkCreated` – ウォーターマークを適用してからの最大日数

各条件の有効な値と制約については、「Amazon EC2 API リファレンス」の「ImageCriterionRequest」を参照してください。

許可された AMI の設定

許可された AMI のコア設定は、許可された AMI の基準を定義する `ImageCriteria` 設定です。次の JSON 構造は、指定できるパラメータを示しています。

```
{
    "State": "enabled" | "disabled" | "audit-mode",
    "ImageCriteria" : [
        {
            "ImageProviders": ["string",...],
            "MarketplaceProductCodes": ["string",...],
            "ImageNames":["string",...],
            "CreationDateCondition" : {
                "MaximumDaysSinceCreated": integer
            },
            "DeprecationTimeCondition" : {
```

```
                "MaximumDaysSinceDeprecated": integer
            },
            "ImageWatermarks": [
                {
                    "WatermarkKey": "string",
                    "SourceImageRegion": "string",
                    "MaximumDaysSinceSourceImageCreated": integer,
                    "MaximumDaysSinceWatermarkCreated": integer
                },...
            ]
         },
         ...
}
```

ImageCriteria の例

以下の ImageCriteria の例では 5 つの ImageCriterion を設定します。AMI は、これらの ImageCriterion のいずれかと一致する場合に許可されます。条件の評価方法に関する詳細は、「基準の評価方法」を参照してください。

```
{
    "ImageCriteria": [
        // ImageCriterion 1: Allow AWS Marketplace AMIs with product code
 "abcdefg1234567890"
        {
            "MarketplaceProductCodes": [
                "abcdefg1234567890"
            ]
        },
        // ImageCriterion 2: Allow AMIs from providers whose accounts are
        // "123456789012" OR "123456789013" AND AMI age is less than 300 days
        {
            "ImageProviders": [
                "123456789012",
                "123456789013"
            ],
            "CreationDateCondition": {
                "MaximumDaysSinceCreated": 300
            }
        },
        // ImageCriterion 3: Allow AMIs from provider whose account is "123456789014"
        // AND with names following the pattern "golden-ami-*"
        {
```

```
            "ImageProviders": [
                "123456789014"
            ],
            "ImageNames": [
                "golden-ami-*"
            ]
        },
        // ImageCriterion 4: Allow AMIs from Amazon or verified providers
        // AND which aren't deprecated
        {
            "ImageProviders": [
                "amazon"
            ],
            "DeprecationTimeCondition": {
                "MaximumDaysSinceDeprecated": 0
            }
        },
        // ImageCriterion 5: Allow AMIs that carry an approved watermark
        // from account "111122223333" AND the watermark was applied less than 90 days
 ago
        {
            "ImageWatermarks": [
                {
                    "WatermarkKey": "111122223333:prod-baseline",
                    "MaximumDaysSinceWatermarkCreated": 90
                }
            ]
        }
    ]
}
```

基準の評価方法

以下の表は、AMI が許可されるかどうかを判断する評価ルールを説明し、各レベルで AND 演算子または OR 演算子がどのように適用されるかを示しています。

評価レベル	オペレーター	許可された AMI になるための要件
ImageProviders 、ImageNames 、MarketplaceProduct	OR	AMI は、各パラメータリストの少なくとも 1 つの値と一致する必要があります。

評価レベル	オペレーター	許可された AMI になるための要件
Codes 、ImageWaterrmarks のパラメータ値		
ImageCriterion	AND	AMI は、各 ImageCriterion のすべてのパラメータと一致する必要があります。
ImageCriteria	OR	AMI は、ImageCriterion のいずれかと一致する必要があります。

上記の評価ルールを使用して、それらを ImageCriteria の例 に適用する方法を見てみましょう。

- ImageCriterion 1: AWS Marketplace 製品コード abcdefg1234567890 を持つ AMI を許可する

  OR
- ImageCriterion 2: 次の両方の基準を満たす AMI を許可する
  - 123456789012 OR 123456789013 のいずれかのアカウントが所有している
    - AND
  - 過去 300 日以内に作成された

  OR
- ImageCriterion 3: 次の両方の条件を満たす AMI を許可する:
  - 123456789014 アカウントが所有している
    - AND
  - golden-ami-* パターンで命名されている

  OR
- ImageCriterion 4: 次の両方の条件を満たす AMI を許可する:
  - Amazon か検証済みのプロバイダーが公開している (amazon エイリアスで指定)
    - AND

- 非推奨 (非推奨になってからの最大日数が 0)

OR

- ImageCriterion 5: ウォーターマークが一致する AMI を許可します。
  - ウォーターマークキー 111122223333:prod-baseline
    - AND
  - 90 日より前にウォーターマークを適用した

制限

ImageCriteria には以下までを含めることができます:

- 10 ImageCriterion

各 ImageCriterion には以下までを含めることができます:

- ImageProviders の 200 個の値
- ImageNames の 50 個の値
- MarketplaceProductCodes の 50 個の値
- ImageWatermarks の 50 個の値

制限の例

前述の ImageCriteria の例 の使用:

- 5 つの ImageCriterion があります。上限の 10 に達するまで、最大 5 個を追加できます。
- 最初の ImageCriterion には、MarketplaceProductCodes に 1 つの値があります。この ImageCriterion には、上限の 50 に達するまで、最大 49 個を追加できます。
- 2 番目の ImageCriterion には、ImageProviders に 2 つの値があります。この ImageCriterion には、上限の 200 に達するまで、最大 198 個を追加できます。
- 3 番目の ImageCriterion には、ImageNames に 1 つの値があります。この ImageCriterion には、上限の 50 に達するまで、最大 49 個を追加できます。

許可されている操作

許可された AMI 機能には、イメージ条件を管理するための 3 つのオペレーション状態 (有効、無効、監査モード) があります。これにより、イメージ条件を有効または無効にしたり、必要に応じて確認したりできます。

有効

許可された AMI が有効になっている場合:

- `ImageCriteria` が適用されます。
- 許可された AMI のみが EC2 コンソールとイメージを使用する API (イメージを使用する他のアクションを記述、コピー、保存、実行するなど) で検出できます。
- インスタンスは、許可された AMI を使用してのみ起動できます。

Disabled

許可された AMI が無効になっている場合:

- `ImageCriteria` は適用されません。
- AMI の検出可能性や使用に制限はありません。

オーディオモード

監査モードの場合:

- `ImageCriteria` は適用されますが、AMI の検出可能性や使用に制限はありません。
- EC2 コンソールでは、AMI ごとに、許可されたイメージフィールドには、許可された AMI が有効になっているときに、AMI を検出してアカウント内のユーザーが利用可能であるかを示す [はい] または [いいえ] が表示されます。
- コマンドラインでは、`describe-image` オペレーションのレスポンスに `"ImageAllowed": true` または `"ImageAllowed": false` が含まれ、許可された AMI が有効になっている場合に、AMI が検出可能になり、アカウント内のユーザーが利用可能であるかを示します。
- EC2 コンソールでは、許可された AMI が有効になっている場合に、アカウント内のユーザーが検出または使用できない AMI の横には [許可されていません] と表示されます。

許可された AMI を実装するためのベストプラクティス

許可された AMI、スムーズな移行を確保し、AWS 環境への潜在的な中断を最小限に抑えるために、これらのベストプラクティスを検討してください。

1. 監査モードを有効にする

   まず、監査モードで許可された AMI を有効にします。この状態では、実際にアクセスを制限することなく、どの AMI が基準の影響を受けるかを確認し、リスクのない評価期間を提供できます。

2. 許可された AMI 基準を設定する

   組織のセキュリティポリシー、コンプライアンス要件、運用ニーズに合った AMI プロバイダーを慎重に確立します。

   > **Note**
   >
   > Amazon ECS、Amazon EKS、AWS Lambda マネージドインスタンスなどの AWS マネージドサービスを使用する場合は、AWS が作成した AMI を許可するために amazon エイリアスを指定することが推奨されます。これらのサービスは、Amazon が公開した AMI に依存してインスタンスを起動します。
   > AMI `CreationDateCondition` に制限を設定するときは注意が必要です。過度に制限された日付条件 (AMI は作成から 5 日未満でなければならないなど) を設定すると、AMI が、AWS か他のプロバイダーのいずれのものかにかかわらず、指定された期間内に更新されなかった場合にインスタンスの起動が失敗する可能性があります。
   > コントロールと特異度を向上させるために、`ImageNames` と `ImageProviders` を組み合わせることが推奨されます。`ImageNames` のみを使用すると、AMI を一意に識別できない可能性があります。

3. 期待されるビジネスプロセスへの影響を確認する

   コンソールまたは CLI を使用して、指定した基準を満たさない AMI で起動されたインスタンスを特定できます。この情報は、準拠した AMI を使用するように起動設定を更新する (起動テンプレートで別の AMI を指定するなど) か、これらの AMI を許可するように条件を調整するかどうかを決定するのに役立ちます。

   コンソール: ec2-instance-launched-with-allowed-ami AWS Config ルールを使用し、実行中または停止中のインスタンスが、許可された AMI 基準を満たす AMI で起動されたかどうかを確認します。ルールは、許可された AMI 基準を AMI が満たしていない場合は [NON_COMPLIANT]、満た

している場合は [COMPLIANT] です。ルールは、許可された AMI 設定が 有効 または 監査モード に設定されている場合にのみ機能します。

CLI: describe-instance-image-metadata コマンドを実行し、レスポンスをフィルタリングして、指定した条件を満たさない AMI で起動されたインスタンスを特定します。

コンソールと CLI での手順については、「許可されていない AMI から起動したインスタンスを見つける」を参照してください。

4. 許可された AMI を有効にする

   基準が期待されるビジネスプロセスに悪影響を及ぼさないことを確認したら、許可された AMI を有効にします。

5. インスタンスの起動をモニタリングする

   Amazon EMR、Amazon ECR、Amazon EKS、AWS Elastic Beanstalk など、アプリケーションおよび使用する AWS マネージドサービス全体で AMI からのインスタンス起動を引き続きモニタリングします。予期しない問題がないか確認し、許可された AMI 基準に必要な調整を行います。

6. 新しい AMI のパイロット

   現在の許可されている AMI 設定に準拠していないサードパーティ AMI をテストするために、AWS は次のアプローチを推奨します。

   - 別の AWS アカウント を使用する: ビジネスクリティカルなリソースにアクセスできないアカウントを作成します。このアカウントで許可された AMI 設定が有効になっていない、またはテストする AMI が明示的に許可されていることを確認し、テストできるようにします。
   - 別の AWS リージョン でテストする: サードパーティー AMI は使用できるが、許可された AMI 設定をまだ有効にしていないリージョンを使用します。

   こうすることで、新しいAMIのテスト中でも、ビジネスクリティカルなリソースの安全性を確保することができます。

必要な IAM 許可

許可された AMI 機能を使用するには、次の IAM アクセス許可が必要です:

- `GetAllowedImagesSettings`
- `EnableAllowedImagesSettings`
- `DisableAllowedImagesSettings`

- `ReplaceImageCriteriaInAllowedImagesSettings`

許可された AMI の設定を管理する

許可された AMI の設定を管理できます。これらの設定は、アカウントごとに、リージョン別で行われます。

タスク

- 許可されたAMIを有効にする
- 許可された AMI 基準を設定する
- 許可されたAMIを無効にする
- 許可された AMI 基準を取得する
- 許可されている AMI を見つける
- 許可されていない AMI から起動したインスタンスを見つける

許可されたAMIを有効にする

許可された AMI を有効にし、その基準を指定することができます。監査モードから開始することをお勧めします。監査モードでは、その基準の影響を受ける AMI を、アクセスを実際に制限することなく確認できます。

Console

許可されたAMIを有効にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、ダッシュボードを選択してください。
3. [アカウント属性] カードの [設定] で、[許可されている AMI] を選択します。
4. [許可された AMI] タブで、[管理] を選択します。
5. [許可された AMI 設定] で、[監査モード] または [有効] を選択します。監査モードで開始し、基準をテストした後に、このステップに戻って許可された AMI を有効にするのがお勧めです。
6. (オプション) [AMI 基準] には、JSON 形式の基準を入力します。
7. [更新] を選択します。

AWS CLI

許可されたAMIを有効にするには

enable-allowed-images-settings コマンドを使用します。

```
aws ec2 enable-allowed-images-settings --allowed-images-settings-state enabled
```

代わりに監査モードを有効にする場合は、enabled ではなく audit-mode を指定します。

```
aws ec2 enable-allowed-images-settings --allowed-images-settings-state audit-mode
```

PowerShell

許可されたAMIを有効にするには

Enable-EC2AllowedImagesSetting コマンドレットを使用します。

```
Enable-EC2AllowedImagesSetting -AllowedImagesSettingsState enabled
```

代わりに監査モードを有効にする場合は、enabled ではなく audit-mode を指定します。

```
Enable-EC2AllowedImagesSetting -AllowedImagesSettingsState audit-mode
```

許可された AMI 基準を設定する

許可された AMI を有効にすると、許可された AMI 基準の設定または置き換えが可能になります。

正しい設定と有効な値については、「許可された AMI の設定」と「許可された AMI のパラメータ」を参照してください。

Console

許可された AMI 基準を設定するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、ダッシュボードを選択してください。
3. [アカウント属性] カードの [設定] で、[許可されている AMI] を選択します。
4. [許可された AMI] タブで、[管理] を選択します。

5. [AMI 基準] に、基準を JSON 形式で入力します。
6. [更新] を選択します。

AWS CLI

許可された AMI 基準を設定するには

replace-image-criteria-in-allowed-images-settings コマンドを使用して、許可された AMI の条件を含む JSON ファイルを指定します。

```
aws ec2 replace-image-criteria-in-allowed-images-settings --cli-input-json
 file://file_name.json
```

PowerShell

許可された AMI 基準を設定するには

Set-EC2ImageCriteriaInAllowedImagesSetting コマンドレットを使用して、許可された AMI の条件を含む JSON ファイルを指定します。

```
$imageCriteria = Get-Content -Path .\file_name.json | ConvertFrom-Json
Set-EC2ImageCriteriaInAllowedImagesSetting -ImageCriterion $imageCriteria
```

許可されたAMIを無効にする

許可された AMI は、次の手順で無効にすることができます。

Console

許可されたAMIを無効にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、ダッシュボードを選択してください。
3. [アカウント属性] カードの [設定] で、[許可されている AMI] を選択します。
4. [許可された AMI] タブで、[管理] を選択します。
5. 許可された AMI設定で、無効を選択してください。
6. [Update] (更新) を選択してください。

AWS CLI

許可されたAMIを無効にするには

disable-allowed-images-settings コマンドを使用します。

```
aws ec2 disable-allowed-images-settings
```

PowerShell

許可されたAMIを無効にするには

Disable-EC2AllowedImagesSetting コマンドレットを使用します。

```
Disable-EC2AllowedImagesSetting
```

許可された AMI 基準を取得する

許可された AMI 設定と、許可された AMI 基準の現在の状態を取得できます。

Console

許可された AMI の状態と基準を取得するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、ダッシュボードを選択してください。
3. [アカウント属性] カードの [設定] で、[許可されている AMI] を選択します。
4. [許可された AMI] タブでは、[許可された AMI 設定] が [有効]、[無効]、[監査モード] のいずれかに設定されています。
5. 許可された AMI の状態が [有効] または [監査モード] である場合、[AMI 基準] に、AMI 基準が JSON 形式で表示されます。

AWS CLI

許可された AMI の状態と基準を取得するには

get-allowed-images-settings コマンドを使用します。

```
aws ec2 get-allowed-images-settings
```

次の出力例では、状態は audit-mode で、イメージの条件がアカウントで設定されています。

```
{
    "State": "audit-mode",
    "ImageCriteria": [
        {
            "MarketplaceProductCodes": [
                "abcdefg1234567890"
            ]
        },
        {
            "ImageProviders": [
                "123456789012",
                "123456789013"
            ],
            "CreationDateCondition": {
                "MaximumDaysSinceCreated": 300
            }
        },
        {
            "ImageProviders": [
                "123456789014"
            ],
            "ImageNames": [
                "golden-ami-*"
            ]
        },
        {
            "ImageProviders": [
                "amazon"
            ],
            "DeprecationTimeCondition": {
                "MaximumDaysSinceDeprecated": 0
            }
        }
    ],
    "ManagedBy": "account"
}
```

PowerShell

許可された AMI の状態と基準を取得するには

Get-EC2AllowedImagesSetting コマンドレットを使用します。

```
Get-EC2AllowedImagesSetting | Select-Object `
    State, `
    ManagedBy, `
    @{Name='ImageProviders'; Expression={($_.ImageCriteria.ImageProviders)}}, `
    @{Name='MarketplaceProductCodes';
 Expression={($_.ImageCriteria.MarketplaceProductCodes)}}, `
    @{Name='ImageNames'; Expression={($_.ImageCriteria.ImageNames)}}, `
    @{Name='MaximumDaysSinceCreated';
 Expression={($_.ImageCriteria.CreationDateCondition.MaximumDaysSinceCreated)}}, `
    @{Name='MaximumDaysSinceDeprecated';
 Expression={($_.ImageCriteria.DeprecationTimeCondition.MaximumDaysSinceDeprecated)}}
```

次の出力例では、状態は audit-mode で、イメージの条件がアカウントで設定されています。

```
State      : audit-mode
ManagedBy  : account
ImageProviders            : {123456789012, 123456789013, 123456789014, amazon}
MarketplaceProductCodes   : {abcdefg1234567890}
ImageNames                : {golden-ami-*}
MaximumDaysSinceCreated   : 300
MaximumDaysSinceDeprecated: 0
```

許可されている AMI を見つける

現在の許可された AMI 基準で許可されている、または許可されていない AMI を見つけることができます。

> **Note**
>
> 許可された AMI は監査モードになっている必要があります。

Console

AMI が許可された AMI 基準を満たしているかどうかを確認するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. AMI を選択します。

4. 詳細タブ (チェックボックスを選択した場合) または概要エリア (AMI ID を選択した場合) で、許可されたイメージフィールドを見つけます。

   - [はい] – AMI は許可された AMI 基準を満たしています。この AMI は、ユーザーが許可された AMI を有効にした後に、そのユーザーのアカウントに含まれるユーザーが使用できます。
   - [いいえ] – AMI は許可された AMI 基準を満たしていません。

5. ナビゲーションペインで AMI カタログ を選択してください。

   未 許可 とマークされた AMI は、許可された AMI の基準を満たしていない AMI を示します。許可された AMI が有効になっている場合、この AMI はアカウント内のユーザーには表示または使用できなくなります。

AWS CLI

AMI が許可された AMI 基準を満たしているかどうかを確認するには

describe-images コマンドを使用します。

```
aws ec2 describe-images \
    --image-id ami-0abcdef1234567890 \
    --query Images[].ImageAllowed \
    --output text
```

以下は出力の例です。

```
True
```

許可された AMI 基準を満たす AMI を見つけるには

describe-images コマンドを使用します。

```
aws ec2 describe-images \
    --filters "Name=image-allowed,Values=true" \
    --max-items 10 \
    --query Images[].ImageId
```

以下は出力の例です。

```
ami-000eaaa8be2fd162a
```

```
ami-000f82db25e50de8e
ami-000fc21eb34c7a9a6
ami-0010b876f1287d7be
ami-0010b929226fe8eba
ami-0010957836340aead
ami-00112c992a47ba871
ami-00111759e194abcc1
ami-001112565ffcafa5e
ami-0011e45aaee9fba88
```

PowerShell

AMI が許可された AMI 基準を満たしているかどうかを確認するには

Get-EC2Image コマンドレットを使用します。

```
(Get-EC2Image -ImageId ami-0abcdef1234567890).ImageAllowed
```

以下は出力の例です。

```
True
```

許可された AMI 基準を満たす AMI を見つけるには

Get-EC2Image コマンドレットを使用します。

```
Get-EC2Image `
    -Filter @{Name="image-allows";Values="true"} `
    -MaxResult 10 | `
    Select ImageId
```

以下は出力の例です。

```
ami-000eaaa8be2fd162a
ami-000f82db25e50de8e
ami-000fc21eb34c7a9a6
ami-0010b876f1287d7be
ami-0010b929226fe8eba
ami-0010957836340aead
ami-00112c992a47ba871
ami-00111759e194abcc1
ami-001112565ffcafa5e
```

```
ami-0011e45aaee9fba88
```

許可されていない AMI から起動したインスタンスを見つける

許可された AMI 基準を満たしていない AMI を使って起動したインスタンスを、特定することができます。

Console

インスタンスが、許可されていない AMI を使って起動したかどうかを確認するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択します。
4. [詳細] タブの [インスタンスの詳細] で、[許可されたイメージ] 見つけます。
   - はい – AMI は許可された AMI 基準を満たしています。
   - いいえ – AMI は許可された AMI 基準を満たしていません。

AWS CLI

許可されていない AMI を使って起動したインスタンスを見つけるには

describe-instance-image-metadata コマンドを `image-allowed` フィルターと共に使用します。

```
aws ec2 describe-instance-image-metadata \
    --filters "Name=image-allowed,Values=false" \
    --query "InstanceImageMetadata[*].[InstanceId,ImageMetadata.ImageId]" \
    --output table
```

以下は出力の例です。

```
-----------------------------------------------------
|          DescribeInstanceImageMetadata          |
+----------------------+--------------------------+
|  i-08fd74f3f1595fdbd |  ami-09245d5773578a1d6   |
|  i-0b1bf24fd4f297ab9 |  ami-07cccf2bd80ed467f   |
|  i-026a2eb590b4f7234 |  ami-0c0ec0a3a3a4c34c0   |
|  i-006a6a4e8870c828f |  ami-0a70b9d193ae8a799   |
|  i-0781e91cfeca3179d |  ami-00c257e12d6828491   |
```

```
|  i-02b631e2a6ae7c2d9 |  ami-0bfddf4206f1fa7b9  |
+----------------------+-------------------------+
```

PowerShell

許可されていない AMI を使って起動したインスタンスを見つけるには

Get-EC2InstanceImageMetadata コマンドレットを使用します。

```
Get-EC2InstanceImageMetadata `
    -Filter @{Name="image-allowed";Values="false"} | `
    Select InstanceId, @{Name='ImageId'; Expression={($_.ImageMetadata.ImageId)}}
```

以下は出力の例です。

```
InstanceId          ImageId
----------          -------
i-08fd74f3f1595fdbd ami-09245d5773578a1d6
i-0b1bf24fd4f297ab9 ami-07cccf2bd80ed467f
i-026a2eb590b4f7234 ami-0c0ec0a3a3a4c34c0
i-006a6a4e8870c828f ami-0a70b9d193ae8a799
i-0781e91cfeca3179d ami-00c257e12d6828491
i-02b631e2a6ae7c2d9 ami-0bfddf4206f1fa7b9
```

AWS Config

ec2-instance-launched-with-allowed-ami AWS Config ルールを追加し、それを要件に合わせて設定してから、それを使用してインスタンスを評価します。

詳細については、「AWS Config デベロッパーガイド」の「Adding AWS Config Rules」および「ec2-instance-launched-with-allowed-ami」を参照してください。

Amazon EC2 で使用するために AMI を公開する

AMI をすべての AWS アカウント と共有することで公開できます。

AMI がパブリックに共有されないようにしたい場合は、AMI のパブリックアクセスをブロックできます。これにより、AMI を公開しようとするあらゆる試みがブロックされ、不正アクセスや AMI データの悪用を防ぐのに役立ちます。パブリックアクセスのブロックを有効にしても、既に公開されている AMI には影響しないことに注意してください。AMI は引き続き公開されています。詳細については、「AMI のパブリックアクセスのブロックについて」を参照してください。

特定のアカウントのみが AMI を使用してインスタンスを起動可能にする方法については、「特定の AWS アカウントとの AMI の共有」を参照してください。

内容

- 考慮事項
- すべての AWS アカウントで AMI を共有 (パブリックに共有)

考慮事項

AMI を公開する前に、以下の点を検討してください。

- 所有権 — AMI を公開するには、お客様の AWS アカウント がその AMI を所有している必要があります。
- リージョン – AMI はリージョンのリソースです。共有した AMI は、共有したリージョンでのみ使用できます。AMI を他のリージョンで利用できるようにするには、AMI をそのリージョンにコピーし、共有します。詳細については、「Amazon EC2 AMI のコピー」を参照してください。
- パブリックアクセスをブロック – AMI をパブリックに共有するには、AMI をパブリックに共有する各リージョンで AMI のパブリックアクセスのブロックを無効にする必要があります。AMI をパブリックに共有した後で、AMI のパブリックアクセスのブロックを再度有効にして、AMI がそれ以上パブリックに共有されないようにできます。
- 公開できない AMI - 次のコンポーネントが含まれる AMI は公開できません (ただし、AMI を特定の AWS アカウント と共有することはできます)。
  - 暗号化されたボリューム
  - 暗号化されたボリュームのスナップショット
  - 製品コード
- 機密データが公開されないようにする - AMI を共有するときに機密データが公開されないようにするには、「共有 Linux AMI の作成に関する推奨事項」のセキュリティ考慮事項を読み、推奨アクションに従います。
- 使用 — AMI を共有する場合、ユーザーは AMI からのインスタンスのみを起動できます。AMI はそれを削除、共有、または変更することはできません。ただし、AMI を使用してインスタンスを起動した後は、起動したインスタンスから AMI を作成できます。
- 自動非推奨 – すべてのパブリック AMI を非推奨にする日はデフォルトで AMI 作成日の 2 年後になっています。非推奨にする日は 2 年より前の日付に設定できます。非推奨にする日を取り消す場合や、非推奨にする日をもっと先の日付に変える場合は、AMI を特定の AWS アカウント とのみ共有するようにして、AMI を非公開にする必要があります。

- 旧型の AMI の削除 — パブリック AMI の廃止日が過ぎ、その AMI で新しいインスタンスが 6 か月以上起動されなかった場合、AWS はそのパブリック共有プロパティを削除し、古くなった AMI がパブリック AMI リストに表示されないようにします。
- 請求 — 他の AWS アカウント がお客様の AMI を使用してインスタンスを起動しても、お客様には請求されません。AMI を使用してインスタンスを起動するアカウントには、起動するインスタンスに対して請求されます。

すべての AWS アカウントで AMI を共有 (パブリックに共有)

AMI を公開すると、コンソールの [コミュニティ AMI] で使用できるようになります。これには、EC2 コンソールの左側のナビゲーターにある [AMI カタログ] から、またはコンソールを使用してインスタンスを起動するときにアクセスできます。AMI は、公開してから [Community AMI] に表示されるまでに、しばらく時間がかかることもあります。

Console

[To make an AMI public]

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. リストから AMI を選択し、[アクション] から [AMI 権限の編集] を選択してください。
4. [AMI の可用性] で、[パブリック] を選択してください。
5. [変更の保存] をクリックしてください。

AWS CLI

各 AMI には、所有者以外でその AMI を使用してインスタンスを起動できる AWS アカウント を制御する `launchPermission` プロパティがあります。AMI の `launchPermission` プロパティを変更することで、AMI を公開したり (この場合、すべての AWS アカウント に起動許可が与えられます) 、指定した AWS アカウント とのみ AMI を共有したりすることができます。

AMI の起動許可を持っているアカウントの一覧に対してアカウント ID の追加または削除ができます。AMI を公開するには、`all` グループを指定します。パブリック起動許可と明示的起動許可の両方を指定できます。

[To make an AMI public]

1. 次のように、modify-image-attribute コマンドを使用して、指定した AMI の launchPermission リストに all グループを追加します。

```
aws ec2 modify-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --launch-permission "Add=[{Group=all}]"
```

2. AMI の起動許可を確認するには、describe-image-attribute コマンドを使用します。

```
aws ec2 describe-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --attribute launchPermission
```

3. (オプション) AMI をプライベートに戻すには、その起動許可から all グループを削除します。AMI の所有者には常に起動許可が与えられるため、このコマンドの影響を受けないことにご注意ください。

```
aws ec2 modify-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --launch-permission "Remove=[{Group=all}]"
```

PowerShell

各 AMI には、所有者以外でその AMI を使用してインスタンスを起動できる AWS アカウント を制御する launchPermission プロパティがあります。AMI の launchPermission プロパティを変更することで、AMI を公開したり (この場合、すべての AWS アカウント に起動許可が与えられます) 、指定した AWS アカウント とのみ AMI を共有したりすることができます。

AMI の起動許可を持っているアカウントの一覧に対してアカウント ID の追加または削除ができます。AMI を公開するには、all グループを指定します。パブリック起動許可と明示的起動許可の両方を指定できます。

[To make an AMI public]

1. 次のように、Edit-EC2ImageAttribute コマンドを使用して、指定した AMI の launchPermission リストに all グループを追加します。

```
Edit-EC2ImageAttribute `
```

```
    -ImageId ami-0abcdef1234567890 `
    -Attribute launchPermission `
    -OperationType add `
    -UserGroup all
```

2. AMI の起動許可を確認するには、次の Get-EC2ImageAttribute コマンドを使用します。

```
Get-EC2ImageAttribute `
    -ImageId ami-0abcdef1234567890 `
    -Attribute launchPermission
```

3. (オプション) AMI をプライベートに戻すには、その起動許可から all グループを削除します。AMI の所有者には常に起動許可が与えられるため、このコマンドの影響を受けないことにご注意ください。

```
Edit-EC2ImageAttribute `
    -ImageId ami-0abcdef1234567890 `
    -Attribute launchPermission `
    -OperationType remove `
    -UserGroup all
```

AMI のパブリックアクセスのブロックについて

AMI がパブリックに共有されないようにするには、AMI のパブリックアクセスのブロックを有効にします。

パブリックアクセスのブロックを有効にすると、AMI を公開しようとする試みは自動的にブロックされます。ただし、既にパブリック AMI がある場合は、公開されたままになります。

AMI をパブリックに共有するには、パブリックアクセスのブロックを無効にする必要があります。共有が完了したら、AMI が意図せずパブリックに共有されないように、パブリックアクセスのブロックを再度有効にするのがベストプラクティスです。

> **Note**
>
> この設定は、アカウントレベルで直接、または宣言ポリシーを使用して設定されます。AMI のパブリック共有を防止する各 AWS リージョンで設定する必要があります。宣言型ポリシーを使用すると、複数のリージョンと複数のアカウントで同時に設定を適用できます。宣言ポリシーが使用されている場合、アカウント内で直接設定を変更することはできません。このトピックでは、アカウント内で設定を直接設定する方法について説明します。宣言ポリ

シーの使用の詳細については、「AWS Organizations ユーザーガイド」の「宣言ポリシー」を参照してください。

管理者ユーザーのみが AMI のパブリックアクセスのブロックを有効または無効にできるように、IAM アクセス許可を管理者ユーザーに制限できます。

トピック

- デフォルト設定
- AMI のパブリックアクセスのブロック設定を管理する

デフォルト設定

[AMI のパブリックアクセスをブロック] 設定は、アカウントが新規か既存か、およびパブリック AMI の有無に応じて、デフォルトで有効または無効になります。次のテーブルは、デフォルト設定の一覧です。

AWS アカウント	AMI のデフォルト設定ではパブリックアクセスをブロックします。
新しいアカウント	有効
パブリック AMI のない既存のアカウント [1]	有効
1 つ以上のパブリック AMI がある既存のアカウント	Disabled

[1] 2023 年 7 月 15 日以降のアカウントに 1 つ以上のパブリック AMI があった場合、その後すべての AMI を非公開にしたとしても、[AMI のパブリックアクセスをブロック] はデフォルトで無効になっています。

AMI のパブリックアクセスのブロック設定を管理する

AMI を公開して共有できるかどうかを制御する AMI 用のパブリックアクセスのブロック設定を管理できます。Amazon EC2 コンソールまたは AWS CLI を使用して、AMI の現在のパブリックアクセスのブロック状態を有効化、無効化、または表示できます。

AMI のパブリックアクセスのブロック状態を表示する

AMI のパブリックアクセスのブロック状態を表示すると、AMI のパブリック共有がアカウントでブロックされているかどうかを確認できます。AMI のパブリック共有がブロックされているかどうかを確認するには、それぞれの AWS リージョン で状態を確認する必要があります。

必要なアクセス許可

AMI の現在のパブリックアクセスのブロック設定を取得するには、`GetImageBlockPublicAccessState` IAM アクセス許可が必要です。

Console

指定したリージョンで AMI のパブリックアクセスのブロック状態を表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/を開いてください。
2. 画面上部のナビゲーションバーで、AMI のパブリックアクセスのブロック状態を表示するリージョンを選択してください。
3. ナビゲーションペインで、ダッシュボードを選択してください。
4. [アカウント属性] カードの [設定] で、[データ保護とセキュリティ] を選択します。
5. [AMI のパブリックアクセスをブロック] で [パブリックアクセス] フィールドを確認します。値は [新しいパブリック共有をブロック中] または [新しいパブリック共有が可能] です。

AWS CLI

AMI のパブリックアクセスのブロック状態を取得するには

get-image-block-public-access-state コマンドを使用します。この値は `block-new-sharing` または `unblocked` のどちらかです。

例: 特定のリージョンの場合

```
aws ec2 get-image-block-public-access-state --region us-east-1
```

`ManagedBy` フィールドは、設定を構成したエンティティを示します。この例では、`account` は設定がアカウントで直接設定されたことを示します。`declarative-policy` という値は、その設定が宣言的ポリシーによって構成されたことを意味します。詳細については、「AWS OrganizationsIAM ユーザーガイド」の「管理されたポリシー」を参照してください。

```
{
    "ImageBlockPublicAccessState": "block-new-sharing",
    "ManagedBy": "account"
}
```

例: アカウントの全リージョンの場合

```
echo -e "Region   \t Public Access State" ; \
echo -e "-------------- \t ----------------------" ; \
for region in $(
    aws ec2 describe-regions \
        --region us-east-1 \
        --query "Regions[*].[RegionName]" \
        --output text
    );
    do (output=$(
        aws ec2 get-image-block-public-access-state \
            --region $region \
            --output text)
        echo -e "$region \t $output"
    );
done
```

以下は出力の例です。

```
Region           Public Access State
--------------   ----------------------
ap-south-1       block-new-sharing
eu-north-1       unblocked
eu-west-3        block-new-sharing
...
```

PowerShell

AMI のパブリックアクセスのブロック状態を取得するには

Get-EC2ImageBlockPublicAccessState コマンドレットを使用します。この値は block-new-sharing または unblocked のどちらかです。

例: 特定のリージョンの場合

```
Get-EC2ImageBlockPublicAccessState -Region us-east-1
```

以下は出力の例です。

```
block-new-sharing
```

例: アカウントの全リージョンの場合

```
(Get-EC2Region).RegionName | `
    ForEach-Object {
        [PSCustomObject]@{
            Region   = $_
            PublicAccessState = (Get-EC2ImageBlockPublicAccessState -Region $_)
        }
} | `
Format-Table -AutoSize
```

以下は出力の例です。

```
Region         PublicAccessState
------         -----------------
ap-south-1     block-new-sharing
eu-north-1     block-new-sharing
eu-west-3      block-new-sharing
...
```

AMI のパブリックアクセスのブロックを有効にする

AMI がパブリックに共有されないようにするには、AMI のパブリックアクセスのブロックを有効にします。AMI がパブリックに共有されないようにする AWS リージョン ごとに、AMI のパブリックアクセスのブロックを有効にする必要があります。既にパブリック AMI がある場合は、引き続き公開されます。

必要なアクセス許可

AMI のパブリックアクセスのブロック設定を有効にするには、EnableImageBlockPublicAccess IAM アクセス許可が必要です。

考慮事項

- この設定を行うには、最大 10 分かかる場合があります。この間、パブリックアクセス状態を記述すると、レスポンスは unblocked になります。設定が完了すると、レスポンスは block-new-sharing になります。

Console

指定したリージョンで AMI のパブリックアクセスのブロックを有効にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/を開いてください。
2. 画面上部のナビゲーションバーで、AMI のパブリックアクセスをブロックするリージョンを選択してください。
3. ナビゲーションペインで、ダッシュボードを選択してください。
4. [アカウント属性] カードの [設定] で、[データ保護とセキュリティ] を選択します。
5. [AMI のパブリックアクセスをブロック] で [管理] を選択してください。
6. [新しいパブリック共有をブロック] のチェックボックスをオンにしたら、[更新] を選択してください。

AWS CLI

AMI のパブリックアクセスのブロックを有効にするには

enable-image-block-public-access コマンドを使用します。

例: 特定のリージョンの場合

```
aws ec2 enable-image-block-public-access \
--region us-east-1 \
--image-block-public-access-state block-new-sharing
```

以下は出力の例です。

```
{
    "ImageBlockPublicAccessState": "block-new-sharing"
}
```

例: アカウントの全リージョンの場合

```
echo -e "Region   \t Public Access State" ; \
echo -e "-------------- \t ----------------------" ; \
for region in $(
    aws ec2 describe-regions \
        --region us-east-1 \
        --query "Regions[*].[RegionName]" \
```

```
        --output text
    );
    do (output=$(
        aws ec2 enable-image-block-public-access \
            --region $region \
            --image-block-public-access-state block-new-sharing \
            --output text)
        echo -e "$region \t $output"
    );
done
```

以下は出力の例です。

```
Region           Public Access State
--------------   ----------------------
ap-south-1       block-new-sharing
eu-north-1       block-new-sharing
eu-west-3        block-new-sharing
...
```

PowerShell

AMI のパブリックアクセスのブロックを有効にするには

Enable-EC2ImageBlockPublicAccess コマンドを使用します。

例: 特定のリージョンの場合

```
Enable-EC2ImageBlockPublicAccess `
    -Region us-east-1 `
    -ImageBlockPublicAccessState block-new-sharing
```

以下は出力の例です。

```
Value
-----
block-new-sharing
```

例: アカウントの全リージョンの場合

```
(Get-EC2Region).RegionName | `
    ForEach-Object {
```

```
    [PSCustomObject]@{
        Region            = $_
        PublicAccessState = (
        Enable-EC2ImageBlockPublicAccess `
         -Region $_ `
         -ImageBlockPublicAccessState block-new-sharing)
    }
} | `
Format-Table -AutoSize
```

以下は出力の例です。

```
Region         PublicAccessState
------         -----------------
ap-south-1     block-new-sharing
eu-north-1     block-new-sharing
eu-west-3      block-new-sharing
...
```

AMI のパブリックアクセスのブロックを無効にする

アカウント内のユーザーが AMI をパブリックに共有できるようにするには、アカウントレベルでパブリックアクセスのブロックを無効にします。AMI がパブリックに共有できるようにする AWS リージョン ごとに、AMI のパブリックアクセスのブロックを無効にする必要があります。

必要なアクセス許可

AMI のパブリックアクセスのブロック設定を無効にするには、`DisableImageBlockPublicAccess` IAM アクセス許可が必要です。

考慮事項

- この設定を行うには、最大 10 分かかる場合があります。この間、パブリックアクセス状態を記述すると、レスポンスは `block-new-sharing` になります。設定が完了すると、レスポンスは `unblocked` になります。

Console

指定したリージョンで AMI のパブリックアクセスのブロックを無効にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/を開いてください。

2. 画面上部のナビゲーションバーで、AMI のパブリックアクセスのブロックを無効にするリージョンを選択してください。
3. ナビゲーションペインで、ダッシュボードを選択してください。
4. [アカウント属性] カードの [設定] で、[データ保護とセキュリティ] を選択します。
5. [AMI のパブリックアクセスをブロック] で [管理] を選択してください。
6. [新しいパブリック共有のブロック] のチェックボックスをオフにしたら、[更新] を選択してください。
7. 確認を求められたら、「**confirm**」と入力してから、[パブリック共有の許可] を選択してください。

AWS CLI

AMI のパブリックアクセスのブロックを無効にするには

disable-image-block-public-access コマンドを使用します。

例: 特定のリージョンの場合

```
aws ec2 disable-image-block-public-access --region us-east-1
```

以下は出力の例です。

```
{
   "ImageBlockPublicAccessState": "unblocked"
}
```

例: アカウントの全リージョンの場合

```
echo -e "Region   \t Public Access State" ; \
echo -e "-------------- \t ----------------------" ; \
for region in $(
    aws ec2 describe-regions \
        --region us-east-1 \
        --query "Regions[*].[RegionName]" \
        --output text
    );
    do (output=$(
```

```
        aws ec2 disable-image-block-public-access \
            --region $region \
            --output text)
        echo -e "$region \t $output"
    );
done
```

以下は出力の例です。

```
Region           Public Access State
--------------   ----------------------
ap-south-1       unblocked
eu-north-1       unblocked
eu-west-3        unblocked
...
```

PowerShell

AMI のパブリックアクセスのブロックを無効にするには

Disable-EC2ImageBlockPublicAccess コマンドレットを使用します。

例: 特定のリージョンの場合

```
Disable-EC2ImageBlockPublicAccess -Region us-east-1
```

以下は出力の例です。

```
Value
-----
unblocked
```

例: アカウントの全リージョンの場合

```
(Get-EC2Region).RegionName | `
    ForEach-Object {
    [PSCustomObject]@{
        Region            = $_
        PublicAccessState = (Disable-EC2ImageBlockPublicAccess -Region $_)
    }
```

```
} | `
Format-Table -AutoSize
```

以下は出力の例です。

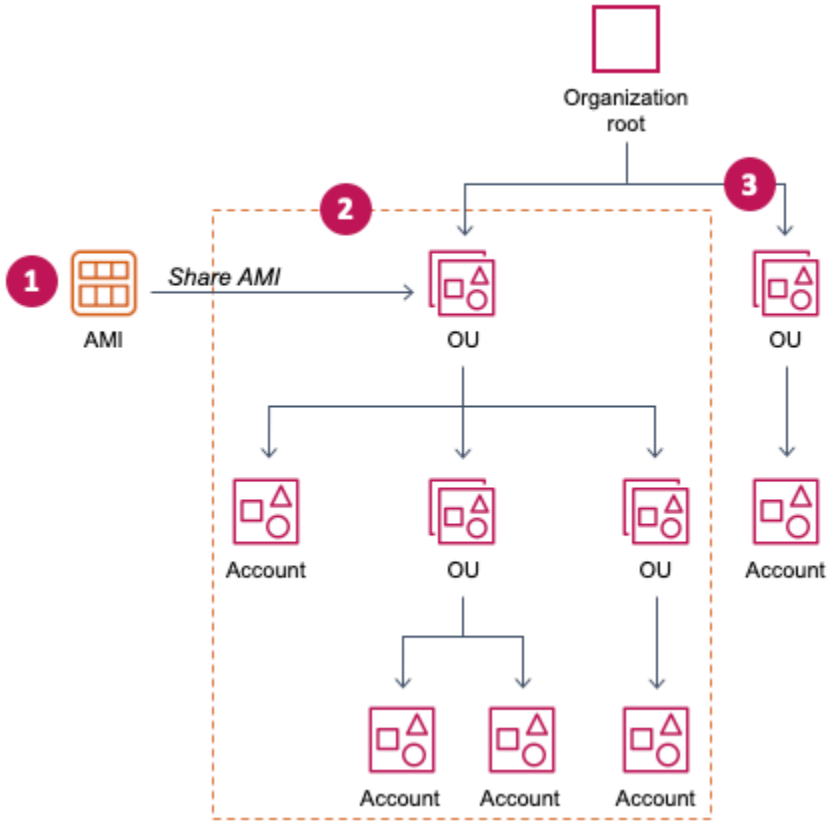
```
Region         PublicAccessState
------         -----------------
ap-south-1     unblocked
eu-north-1     unblocked
eu-west-3      unblocked
...
```

組織および組織単位での共有 AMI の使用

AWS Organizations は、作成し一元管理する組織に、複数の AWS アカウント を統合するためのアカウント管理サービスです。AMI は、特定のアカウントと共有するだけでなく、組織または組織単位 (OU) と共有することもできます。

組織とは、AWS アカウント を統合して一元管理するために作成するエンティティのことです。アカウントを階層ツリーのような構造に編成して、ルートを最上部に置いて組織単位をその組織ルート下にネストすることができます。各アカウントは、ルートに直接追加するか、階層内の OU のいずれかに配置することができます。詳細については、「AWS Organizations ユーザーガイド」の「AWS 組織の用語およびコンセプト」を参照してください。

AMI を組織または OU と共有すると、すべての子アカウントが AMI にアクセスできます。例えば、次の図では、AMI は最上位レベルの OU と共有されています (1 の数字の矢印で示されます)。その最上位レベルの OU の下にネストされているすべての OU やアカウント (2 の数字の点線で示したもの) も AMI にアクセスできます。点線の外側にある組織や OU (数字の 3 で示されている) のアカウントは、AMI が共有されている OU の子供ではないため、AMI へのアクセス権はありません。



トピック

- 考慮事項
- 組織または組織単位の ARN を取得する
- 組織と OU に KMS キーの使用を許可する
- 組織または OU の AMI 共有を管理する

考慮事項

特定の組織または組織単位で AMI を共有する場合は、以下について検討してください。

- 所有権 — AMI を共有するには、お客様の AWS アカウント がその AMI を所有している必要があります。
- 共有制限 — AMI の所有者は、メンバーではない組織や OU を含め、任意の組織または OU と AMI を共有できます。

  リージョン内で AMI を共有できるエンティティの最大数については、「Amazon EC2 Service Quotas」をご覧ください。

- タグ - ユーザー定義タグ (AMI にアタッチするタグ) は共有できません。AMI を共有する場合、ユーザー定義タグは AMI が共有されている組織または OU のどの AWS アカウント にも使用できません。
- ARN 形式 — コマンドで組織または OU を指定する場合は、正しい ARN 形式を必ず使用してください。ID のみを指定するとエラーになります。例えば、`o-123example` や `ou-1234-5example` を指定するとエラーになります。

  正しい ARN 形式:

  - 組織の ARN: `arn:aws:organizations::`*`111122223333`*`:organization/`*`organization-id`*
  - OU ARN: `arn:aws:organizations::`*`111122223333`*`:ou/`*`organization-id`*`/`*`ou-id`*

  説明は以下のとおりです。

  - *`111122223333`* は、管理アカウントの 12 桁アカウント ID の例です。管理アカウント番号がわからない場合は、管理アカウント番号を含む ARN を取得するための組織または組織単位を記述できます。詳細については、組織または組織単位の ARN を取得する を参照してください。
  - *`organization-id`* は組織 ID であり、例えば、`o-123example` となります。
  - *`ou-id`* は組織単位 ID であり、例えば、`ou-1234-5example` となります。

  ARN の形式の詳細については、「IAM ユーザーガイド」の「Amazon リソースネーム (ARN)」を参照してください。
- 暗号化とキー — 暗号化されていないスナップショットと暗号化されたスナップショットによってバックアップされた AMI を共有できます。
  - 暗号化されたスナップショットは、カスタマー管理された型キーを使用して暗号化する必要があります。デフォルトの AWS 管理キーで暗号化されたスナップショットでバックアップされた AMI を共有することはできません。
  - 暗号化されたスナップショットによってバックアップされた AMI を共有する場合、スナップショットの暗号化に使用されたカスタマー管理されたキーの使用を組織または OU に許可する必要があります。詳細については、組織と OU に KMS キーの使用を許可する を参照してください。
- リージョン – AMI はリージョンのリソースです。共有した AMI は、共有したリージョンでのみ使用できます。AMI を他のリージョンで利用できるようにするには、AMI をそのリージョンにコピーし、共有します。詳細については、Amazon EC2 AMI のコピー を参照してください。

- 使用 — AMI を共有する場合、ユーザーは AMI からのインスタンスのみを起動できます。AMI はそれを削除、共有、または変更することはできません。ただし、AMI を使用してインスタンスを起動した後は、起動したインスタンスから AMI を作成できます。
- 請求 — 他の AWS アカウント がお客様の AMI を使用してインスタンスを起動しても、お客様には請求されません。AMI を使用してインスタンスを起動するアカウントには、起動するインスタンスに対して請求されます。

組織または組織単位の ARN を取得する

組織と組織単位 ARN には、12 桁の管理アカウント番号が含まれています。管理アカウント番号がわからない場合は、組織と組織単位を記述して、それぞれの ARN を取得できます。以下の例では、123456789012 が管理アカウントのアカウント ID です。

必要なアクセス許可

ARN を取得する前に、組織と組織単位を記述する権限が必要です。次のポリシーで、これらの権限が付与されます。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "organizations:Describe*"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS CLI

組織の ARN を取得するには

describe-organization コマンドを使用します。組織 ARN のみを返す `--query` オプションを追加します。

```
aws organizations describe-organization --query 'Organization.Arn'
```

以下は出力の例です。

```
"arn:aws:organizations::123456789012:organization/o-1234567abc"
```

組織単位の ARN を取得するには

describe-organizational-unit コマンドを使用します。--query パラメータを使用して、組織単位の ARN のみを返します。

```
aws organizations describe-organizational-unit \
    --organizational-unit-id ou-a123-b4567890 \
    --query 'OrganizationalUnit.Arn'
```

以下は出力の例です。

```
"arn:aws:organizations::123456789012:ou/o-1234567abc/ou-a123-b4567890"
```

PowerShell

組織の ARN を取得するには

Get-ORGOrganization コマンドレットを使用します。

```
(Get-ORGOrganization).Arn
```

以下は出力の例です。

```
arn:aws:organizations::123456789012:organization/o-1234567abc
```

組織単位の ARN を取得するには

Get-ORGOrganizationalUnit コマンドレットを使用します。

```
(Get-ORGOrganizationalUnit -OrganizationalUnitId "ou-a123-b4567890").Arn
```

以下は出力の例です。

```
arn:aws:organizations::123456789012:ou/o-1234567abc/ou-a123-b4567890
```

組織と OU に KMS キーの使用を許可する

暗号化されたスナップショットによってバックアップされた AMI を共有する場合、組織または OU がスナップショットの暗号化に使用されの使用を許可する必要もあります。

> **Note**
>
> 暗号化されたスナップショットは、カスタマー 管理された型キーを使用して暗号化す る必要があります。デフォルトの AWS 管理キーで暗号化されたスナップショットでバックアップされた AMI を共有することはできません。

KMS キーへのアクセスを制御するには、キーポリシーで aws:PrincipalOrgID および aws:PrincipalOrgPaths 条件キーを使用して、指定したアクションに対する許可を特定のプリンシパルにのみ許可します。そのプリンシパルは、ユーザー、IAM ロール、フェデレーションユーザー、または AWS アカウント ルートユーザーです。

条件キーは次のように使用されます。

- aws:PrincipalOrgID – 指定された ID で表される組織に属するプリンシパルを許可します。
- aws:PrincipalOrgPaths – 指定されたパスで表される OU に属するプリンシパルを許可します。

組織または OU に KMS キーを使用するアクセス権限を付与するには、次のステートメントをキーポリシーに追加します。

```
{
    "Sid": "Allow access for organization root",
    "Effect": "Allow",
    "Principal": "*",
    "Action": [
        "kms:Describe*",
        "kms:List*",
        "kms:Get*",
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:CreateGrant"
```

```
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:PrincipalOrgID": "o-123example"
        }
    }
}
```

特定の OU (およびその OU に属するアカウント) に KMS キーを使用するアクセス許可を付与するには、次の例のようなポリシーを使用できます。

```
{
        "Sid": "Allow access for specific OUs and their descendants",
        "Effect": "Allow",
        "Principal": "*",
        "Action": [
            "kms:Describe*",
            "kms:List*",
            "kms:Get*",
            "kms:Encrypt",
            "kms:Decrypt",
            "kms:ReEncrypt*",
            "kms:GenerateDataKey*",
            "kms:CreateGrant"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:PrincipalOrgID": "o-123example"
            },
            "ForAnyValue:StringLike": {
                "aws:PrincipalOrgPaths": [
                    "o-123example/r-ab12/ou-ab12-33333333/*",
                    "o-123example/r-ab12/ou-ab12-22222222/*"
                ]
            }
        }
}
```

その他の条件ステートメントの例については、「IAM ユーザーガイド」の「aws:PrincipalOrgID」と「aws:PrincipalOrgPaths」を参照してください。

クロスアカウントアクセス権限の付与の詳細については、「AWS Key Management Service Developer Guide」の「Allowing users in other accounts to use a KMS key」を参照してください。

組織または OU の AMI 共有を管理する

組織および組織単位 (OU) との AMI の共有を管理することで、組織および OU が Amazon EC2 インスタンスを起動できるかどうかを制御できます。

AMI が共有されている組織と OU を表示する

AMI を共有した組織と OU を見つけることができます。

Console

を使用して AMI を共有した組織およびOUを確認するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. リストで AMI を選択し、[アクセス許可] タブをクリックし、[共有組織/OU] までスクロールします。

   共有されている AMI を見つけるには、「Amazon EC2 インスタンスで使用する共有 AMI を検索する」を参照してください。

AWS CLI

を使用して AMI を共有した組織およびOUを確認するには

describe-image-attribute コマンドを `launchPermission` 属性と共に使用します。

```
aws ec2 describe-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --attribute launchPermission
```

以下に、レスポンスの例を示します。

```
{
    "ImageId": "ami-0abcdef1234567890",
    "LaunchPermissions": [
        {
```

```
            "OrganizationalUnitArn": "arn:aws:organizations::111122223333:ou/
o-123example/ou-1234-5example"
        }
    ]
}
```

PowerShell

を使用して AMI を共有した組織およびOUを確認するには

Get-EC2ImageAttribute コマンドレットを使用します。

```
Get-EC2ImageAttribute `
    -ImageId ami-0abcdef1234567890 `
    -Attribute launchPermission
```

組織または OU で AMI を共有する

組織または OU で AMI を共有できます。

> **Note**
>
> AMI を共有するために、AMI の参照先の Amazon EBS スナップショットを共有する必要はありません。共有する必要があるのは AMI 自体だけです。起動の際に、参照先の EBS スナップショットへのインスタンスアクセスが自動的に提供されます。ただし、AMI が参照するスナップショットを暗号化するために使用した KMS キーは共有する必要があります。詳細については、「組織と OU に KMS キーの使用を許可する」を参照してください。

Console

AMI を組織または OU と共有するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. リストで AMI を選択し、[アクション] から [AMI 権限の編集] を選択してください。
4. [AMI の利用状況] で、[プライベート] を選択してください。
5. [共有組織/OU] の隣で、[組織/OU ARN 追加] を選択してください。

6. [組織/OU ARN] で、AMI を共有する組織 ARN または OU ARN を入力し、[AMI の共有] を選択してください。ID だけでなく、完全な ARN を指定する必要があることに注意してください。

   この AMI を複数の組織または OU と共有するには、この手順を繰り返して、必要なすべての組織または OU を追加します。
7. 完了したら、[変更保存] を選択してください。
8. (オプション) AMI を共有した組織または OU を表示するには、リストから AMI を選択し、[アクセス許可] タブをクリックし、[共有組織/OU] までスクロールします。共有されている AMI を見つけるには、「Amazon EC2 インスタンスで使用する共有 AMI を検索する」を参照してください。

AWS CLI

AMI を組織または OU と共有するには

modify-image-attribute コマンドを使用すると、指定した組織に対し、指定した AMI の起動許可が与えられます。

```
aws ec2 modify-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --launch-permission
 "Add=[{OrganizationArn=arn:aws:organizations::123456789012:organization/
o-123example}]"
```

を使用して AMI を OUと共有するには

[modify-image-attribute] コマンドを使用すると、指定した OU に対し、指定した AMI の起動許可が与えられます。ID だけでなく、完全な ARN を指定する必要があることに注意してください。

```
aws ec2 modify-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --launch-permission
 "Add=[{OrganizationalUnitArn=arn:aws:organizations::123456789012:ou/o-123example/
ou-1234-5example}]"
```

PowerShell

AMI を共有するには、次の例のように Edit-EC2ImageAttribute コマンド (Windows PowerShell用ツール) を使用します。

AMI を組織または OU と共有するには

次のコマンドを使用すると、指定した組織に対し、指定した AMI の起動許可が与えられます。

```
Edit-EC2ImageAttribute `
    -ImageId ami-0abcdef1234567890 `
    -Attribute launchPermission `
    -OperationType add `
    -OrganizationArn "arn:aws:organizations::123456789012:organization/o-123example"
```

組織または OU と AMI の共有を停止するには

次のコマンドを使用すると、指定した組織から指定した AMI の起動許可が削除されます。

```
Edit-EC2ImageAttribute `
    -ImageId ami-0abcdef1234567890 `
    -Attribute launchPermission `
    -OperationType remove `
    -OrganizationArn "arn:aws:organizations::123456789012:organization/o-123example"
```

すべての組織、OU、および AWS アカウント と AMI の共有を停止するには

次のコマンドを使用すると、指定した AMI からパブリック起動許可と明示的起動許可がすべて削除されます。AMI の所有者には常に起動許可が与えられるため、このコマンドの影響を受けないことにご注意ください。

```
Reset-EC2ImageAttribute `
    -ImageId ami-0abcdef1234567890 `
    -Attribute launchPermission
```

組織または OU での AMI 共有を停止する

組織または OU での AMI 共有を停止できます。

> **Note**
>
> AMI が共有されている組織または OU 内にある場合、特定のアカウントと AMI の共有を停止することはできません。アカウントの起動権限を削除して AMI の共有を停止しようとすると、Amazon EC2 は成功メッセージを返します。ただし、AMI は引き続きアカウントと共有されます。

Console

組織または OU と AMI の共有を停止するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/) を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. リストで AMI を選択し、[アクション] から [AMI 権限の編集] を選択してください。
4. [共有組織/OU] で、AMI の共有を停止する組織または OU を選択し、[選択を削除] を選択してください。
5. 完了したら、[変更保存] を選択してください。
6. (オプション) AMI の組織または OU との共有の停止を確認するには、リストから AMI を選択し、[アクセス許可] タブをクリックし、[共有組織/OU] までスクロールします。

AWS CLI

組織または OU と AMI の共有を停止するには

modify-image-attribute コマンドを使用します。この例では、指定した組織から指定した AMI の起動許可が削除されます。

```
aws ec2 modify-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --launch-permission
 "Remove=[{OrganizationArn=arn:aws:organizations::123456789012:organization/
o-123example}]"
```

すべての組織、OU、および AWS アカウント と AMI の共有を停止するには

reset-image-attribute コマンドを使用します。この例では、指定した AMI からパブリック起動許可と明示的起動許可がすべて削除されます。AMI の所有者には常に起動許可が与えられるため、このコマンドの影響を受けないことにご注意ください。

```
aws ec2 reset-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --attribute launchPermission
```

PowerShell

組織または OU と AMI の共有を停止するには

Edit-EC2ImageAttribute コマンドレットを使用します。この例では、指定した組織から指定した AMI の起動許可が削除されます。

```
Edit-EC2ImageAttribute `
    -ImageId ami-0abcdef1234567890 `
    -Attribute launchPermission `
    -OperationType remove `
    -OrganizationArn "arn:aws:organizations::123456789012:organization/o-123example"
```

すべての組織、OU、および AWS アカウント と AMI の共有を停止するには

Reset-EC2ImageAttribute コマンドレットを使用します。この例では、指定した AMI からパブリック起動許可と明示的起動許可がすべて削除されます。AMI の所有者には常に起動許可が与えられるため、このコマンドの影響を受けないことにご注意ください。

```
Reset-EC2ImageAttribute `
    -ImageId ami-0abcdef1234567890 `
    -Attribute LaunchPermission
```

特定の AWS アカウントとの AMI の共有

AMI を公開せず、特定の AWS アカウント とだけ共有することもできます。これに必要なものは AWS アカウント ID のみです。

AWS アカウント ID は、AWS アカウント を一意に識別する 12 桁の数値です (012345678901 など)。詳細については、「AWS アカウント管理 リファレンスガイド」の「AWS アカウント 識別子の表示」を参照してください。

考慮事項

特定の AWS アカウント で AMI を共有する場合は、以下について検討してください。

- 所有権 — AMI を共有するには、お客様の AWS アカウント がその AMI を所有している必要があります。
- 共有制限 – リージョン内で AMI を共有できるエンティティの最大数については、「Amazon EC2 Service Quotas」をご覧ください。
- タグ - ユーザー定義タグ (AMI にアタッチするタグ) は共有できません。AMI を共有する場合、ユーザー定義タグは AMI が共有されている AWS アカウント では使用できません。

- スナップショット –AMI を共有するために、AMI の参照先の Amazon EBS スナップショットを共有する必要はありません。共有できるのは AMI 自体のみです。システムは起動時に参照される EBS スナップショットへのインスタンス アクセスを提供します。ただし、AMI が参照するスナップショットを暗号化するために使用した KMS キー は共有する必要があります。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS スナップショットの共有」を参照してください。
- 暗号化とキー — 暗号化されていないスナップショットと暗号化されたスナップショットによってバックアップされた AMI を共有できます。
  - 暗号化されたスナップショットは、KMS キーを使用して暗号化する必要があります。デフォルトの AWS 管理キーで暗号化されたスナップショットでバックアップされた AMI を共有することはできません。
  - 暗号化されたスナップショットによってバックアップされた AMI を共有する場合、スナップショットの暗号化に使用された KMS キーの使用を AWS アカウント に許可する必要があります。詳細については、「組織と OU に KMS キーの使用を許可する」を参照してください。暗号化にカスタマー管理されたキーを使用する際に、Auto Scaling インスタンスの起動に必要となるキーポリシーを設定するには、「Amazon EC2 Auto Scaling ユーザーガイド」の「暗号化ボリュームで使用するために必要な AWS KMS key ポリシー」を参照してください。
- リージョン – AMI はリージョンのリソースです。共有した AMI は、そのリージョンでのみ使用できます。AMI を他のリージョンで利用できるようにするには、AMI をそのリージョンにコピーし、共有します。詳細については、Amazon EC2 AMI のコピー を参照してください。
- 使用 — AMI を共有する場合、ユーザーは AMI からのインスタンスのみを起動できます。AMI はそれを削除、共有、または変更することはできません。ただし、AMI を使用してインスタンスを起動した後は、インスタンスから AMI を作成できます。
- 共有 AMI のコピー — 別のアカウントのユーザーが共有 AMI をコピーする場合は、AMI をバックアップするストレージに対する読み取り権限をそのユーザーに付与する必要があります。詳細については、「アカウント間のコピー」を参照してください。
- 請求 — 他の AWS アカウント がお客様の AMI を使用してインスタンスを起動しても、お客様には請求されません。AMI を使用してインスタンスを起動するアカウントには、起動するインスタンスに対して請求されます。

Console

明示的な起動許可を与えるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

2. ナビゲーションペインで [AMI] を選択してください。
3. リストで AMI を選択し、[アクション] から [AMI 権限の編集] を選択してください。
4. [プライベート] を選択してください。
5. [共有アカウント] で、[アカウント ID の追加] () を選択してください。
6. AWS アカウント ID には、AMI を共有したい AWS アカウント ID を入力し、[AMI の共有] を選択してください。

   この AMI を複数のアカウントで共有するには、必要なアカウント ID がすべて追加されるまでステップ 5 と 6 を繰り返します。
7. 完了したら、変更を保存 を選択してください。
8. (オプション) AMI を共有した AWS アカウント ID を表示するには、リストから AMI を選択し、[アクセス許可] タブを開いてください。共有されている AMI を見つけるには、「Amazon EC2 インスタンスで使用する共有 AMI を検索する」を参照してください。

AWS CLI

AMI を共有するには、次の例のように modify-image-attribute マンド () を使用します。

明示的な起動許可を与えるには

次の例では、指定した AWS アカウントに対し、指定した AMI の起動許可が与えられます。

```
aws ec2 modify-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --launch-permission "Add=[{UserId=123456789012}]"
```

アカウントに与えた起動許可を取り消すには

次の例では、指定した AWS アカウントから指定した AMI の起動許可が削除されます。

```
aws ec2 modify-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --launch-permission "Remove=[{UserId=123456789012}]"
```

すべての起動許可を取り消すには

次の例では、指定した AMI からパブリック起動許可と明示的起動許可がすべて削除されます。AMI の所有者には常に起動許可が与えられるため、このコマンドの影響を受けないことにご注意ください。

```
aws ec2 reset-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --attribute launchPermission
```

PowerShell

AMI を共有するには、次の例のように Edit-EC2ImageAttribute コマンド (Windows PowerShell用ツール) を使用します。

明示的な起動許可を与えるには

次の例では、指定した AWS アカウントに対し、指定した AMI の起動許可が与えられます。

```
Edit-EC2ImageAttribute `
    -ImageId ami-0abcdef1234567890 `
    -Attribute launchPermission `
    -OperationType add `
    -UserId "123456789012"
```

アカウントに与えた起動許可を取り消すには

次の例では、指定した AWS アカウントから指定した AMI の起動許可が削除されます。

```
Edit-EC2ImageAttribute `
    -ImageId ami-0abcdef1234567890 `
    -Attribute launchPermission -OperationType remove `
    -UserId "123456789012"
```

すべての起動許可を取り消すには

次の例では、指定した AMI からパブリック起動許可と明示的起動許可がすべて削除されます。AMI の所有者には常に起動許可が与えられるため、このコマンドの影響を受けないことにご注意ください。

```
Reset-EC2ImageAttribute `
    -ImageId ami-0abcdef1234567890 `
    -Attribute launchPermission
```

お客様の AWS アカウント と AMI の共有をキャンセルする

AMI の起動許可にアカウントを追加することで、Amazon マシンイメージ (AMI) を特定の AWS アカウント と共有できます。AMI が AWS アカウント と共有されていて、そのアカウントとの共有が不要になった場合は、AMI の起動許可からアカウントを削除できます。この操作は、`cancel-image-launch-permission` AWS CLI コマンドを実行して行うことができます。このコマンドを実行すると、指定した AMI の起動許可から AWS アカウント が削除されます。お客様の AWS アカウントと共有されている AMI を見つけるには、「Amazon EC2 インスタンスで使用する共有 AMI を検索する」を参照してください。

例えば、共有された未使用または廃止予定の AMI を含むインスタンスを起動する可能性を減らすために、AMI をアカウントで共有することをキャンセルする場合があります。AMI をアカウントと共有することをキャンセルすると、describe-images の出力や EC2 コンソールの AMI リストには表示されなくなります。

トピック

- 制限事項
- アカウントと AMI の共有をキャンセルする

制限事項

- AWS アカウント とだけ共有されている AMI の起動許可からアカウントを削除できます。組織または組織単位 (OU) と共有されている AMI の起動許可からアカウントを削除したり、パブリック AMI へのアクセスを削除したりすることに `cancel-image-launch-permission` を使用することはできません。
- AMI の起動許可からアカウントを完全に削除することはできません。AMI の所有者は、お客様のアカウントと再び AMI を共有できます。
- AMI はリージョンのリソースです。`cancel-image-launch-permission` の実行時には、AMI が配置されているリージョンを指定する必要があります。コマンドの中でリージョンを指定するか、AWS_DEFAULT_REGION 環境変数を使用します。
- AWS CLI および SDK のみが、AMI の起動許可からのアカウントの削除をサポートしています。EC2 コンソールは現在このアクションに対応していません。

アカウントと AMI の共有をキャンセルする

> **Note**
>
> お客様のアカウントと AMI の共有をキャンセルすると、元に戻すことはできません。AMI へのアクセスを回復するには、AMI 所有者がお客様のアカウントと AMI を共有する必要があります。

AWS CLI

アカウントとの AMI の共有をキャンセルするには

cancel-image-launch-permission コマンドを使用します。

```
aws ec2 cancel-image-launch-permission \
    --image-id ami-0abcdef1234567890 \
    --region us-east-1
```

PowerShell

アカウントとの AMI の共有をキャンセルするには

Stop-EC2ImageLaunchPermission コマンドレットを使用します。

```
Stop-EC2ImageLaunchPermission `
    -ImageId ami-0abcdef1234567890 `
    -Region us-east-1
```

共有 Linux AMI の作成に関する推奨事項

攻撃対象領域を縮小し、作成する AMI の信頼性を向上させるためには、次のガイドラインを使用します。

> **Important**
>
> セキュリティのガイドラインのリストは、いずれも完全ではありません。共有 AMI を注意深く作成し、機密データが漏洩される可能性について十分考慮してください。

コンテンツ

- ルートユーザーのパスワードベースのリモートログインを無効にする
- ローカルルートアクセスを無効にする
- SSH ホストキーペアの削除
- パブリックキー認証情報のインストール
- sshd DNS チェックの無効化 (任意)
- 機密データを削除する

AWS Marketplace の AMI を構築する場合は、AWS Marketplace 販売者ガイドの「AMI 構築のベストプラクティス」で、ガイドライン、ポリシー、ベストプラクティスをご参照ください。

ルートユーザーのパスワードベースのリモートログインを無効にする

パブリック AMI に固定のルートパスワードを使用することは、セキュリティの面で危険であり、すぐに知られるおそれがあります。初回ログイン後にパスワードを変更するようにユーザーに依存していますが、変更されるまでの一瞬の間にパスワードが悪用される危険性があります。

この問題を解決するには、ルートユーザーのパスワードベースのリモートログインを無効にします。

ルートユーザーのパスワードベースのリモートログインを無効にするには

1. テキストエディタで /etc/ssh/sshd_config ファイルを開き、次の行を見つけ出します:

   ```
   #PermitRootLogin yes
   ```

2. 行を次のように変更します:

   ```
   PermitRootLogin without-password
   ```

   この設定ファイルの場所は、ディストリビューションに応じて、または OpenSSH を実行していない場合は、異なることがあります。このような場合は、関連資料を参照してください。

ローカルルートアクセスを無効にする

共有 AMI を使用する際のベストプラクティスは、直接ルートログインを無効にすることです。これを行うには、実行中のインスタンスにログインし、次のコマンドを発行します。

```
[ec2-user ~]$ sudo passwd -l root
```

> **Note**
>
> このコマンドが sudo の使用に影響を及ぼすことはありません。

SSH ホストキーペアの削除

パブリック AMI から派生した AMI を共有する場合は、/etc/ssh にある既存の SSH ホストキーペアを削除します。これにより、他のユーザーがお客様の AMI を使用してインスタンスを起動したときに、SSH は、新しい固有の SSH キーペアを生成するように強制されるため、セキュリティが強化され、「中間者」攻撃の可能性を減らします。

システムにある次のすべてのキーファイルを削除します。

- ssh_host_dsa_key
- ssh_host_dsa_key.pub
- ssh_host_key
- ssh_host_key.pub
- ssh_host_rsa_key
- ssh_host_rsa_key.pub
- ssh_host_ecdsa_key
- ssh_host_ecdsa_key.pub
- ssh_host_ed25519_key
- ssh_host_ed25519_key.pub

次のコマンドを使用して、これらのファイルをすべて確実に削除できます。

```
[ec2-user ~]$ sudo shred -u /etc/ssh/*_key /etc/ssh/*_key.pub
```

> **Warning**
>
> shred などの安全な削除ユーティリティでは、ストレージメディアからファイルのすべてのコピーが削除されない場合があります。ファイルの非表示のコピーは、ジャーナルファイルシステム (Amazon Linux のデフォルト ext4 を含む)、スナップショット、バック

アップ、RAID、および一時キャッシュによって作成することができます。詳細については、shred ドキュメントを参照してください。

Important

パブリック AMI から既存の SSH ホストキーペアを削除することを忘れた場合、ルーチン監査プロセスから、AMI のインスタンスを実行するすべての顧客に向けて、セキュリティ上のリスクがある可能性について通知されます。短い猶予期間の後に、AMI にプライベートのマークが付けられます。

パブリックキー認証情報のインストール

パスワードを使用したログインを防ぐように AMI を構成したら、ユーザーが別のメカニズムを使用してログインできるようにしておく必要があります。

ユーザーは、Amazon EC2 を使用すると、インスタンスの起動時にパブリックプライベートキーペア名を指定できます。`RunInstances` API 呼び出し (またはコマンドライン API ツール) で有効なキーペア名を指定すると、パブリックキー (`CreateKeyPair` または `ImportKeyPair` の呼び出し後に Amazon EC2 がサーバー上に保持するキーペアの一部) を、インスタンスメタデータに対する HTTP Query を介してインスタンスで使用できるようになります。

SSH を使用してログインするには、AMI が起動時にキー値を取得し、それを `/root/.ssh/authorized_keys` (または AMI 上のその他のユーザーアカウントの同等項目) に付加する必要があります。ユーザーはキーペアを使用して AMI のインスタンスを起動し、ルートパスワードを入力せずにログインできます。

Amazon Linux や Ubuntu を初めとする多くのディストリビューションでは、`cloud-init` パッケージを使用して、設定されたユーザーのパブリックキー認証情報を挿入します。`cloud-init` をサポートしていないディストリビューションの場合は、システムスタートアップスクリプト (例: `/etc/rc.local`) に次のコードを追加して、起動時にルートユーザーに対して指定したパブリックキーを取り込みます。

Note

次の例では、IP アドレス http://169.254.169.254/ はリンクローカルアドレスであり、インスタンスからのみ有効です。

IMDSv2

```
if [ ! -d /root/.ssh ] ; then
        mkdir -p /root/.ssh
        chmod 700 /root/.ssh
fi
# Fetch public key using HTTP
TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-
token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-
data/public-keys/0/openssh-key > /tmp/my-key
if [ $? -eq 0 ] ; then
        cat /tmp/my-key >> /root/.ssh/authorized_keys
        chmod 700 /root/.ssh/authorized_keys
        rm /tmp/my-key
fi
```

IMDSv1

```
if [ ! -d /root/.ssh ] ; then
        mkdir -p /root/.ssh
        chmod 700 /root/.ssh
fi
# Fetch public key using HTTP
curl http://169.254.169.254/latest/meta-data/public-keys/0/openssh-key > /tmp/my-key
if [ $? -eq 0 ] ; then
        cat /tmp/my-key >> /root/.ssh/authorized_keys
        chmod 700 /root/.ssh/authorized_keys
        rm /tmp/my-key
fi
```

この設定は、あらゆるユーザーに適用できます。root ユーザーに限定する必要はありません。

> **Note**
>
> この AMI に基づいたインスタンスを再バンドルすると、起動時に使用されたキーが組み込まれます。キーへの組み込みを阻止するには、authorized_keys ファイルの を空にする (ファイルを削除する) か、このファイルを再バンドルから除外します。

sshd DNS チェックの無効化 (任意)

sshd DNS チェックを無効にすると、sshd セキュリティが若干低下します。ただし、DNS の解決策が失敗した場合は、SSH ログインが引き続き機能します。sshd チェックを無効にしなかった場合、DNS の解決策が失敗すると、すべてのログインが阻止されます。

sshd DNS チェックを無効にするには

1. テキストエディタで /etc/ssh/sshd_config ファイルを開き、次の行を見つけ出します:

```
#UseDNS yes
```

2. 行を次のように変更します:

```
UseDNS no
```

> **Note**
>
> この設定ファイルの場所は、ディストリビューションに応じて、または OpenSSH を実行していない場合は、異なることがあります。このような場合は、関連資料を参照してください。

機密データを削除する

共有する AMI に、機密性のあるデータやソフトウェアは保管しないことをお勧めします。共有 AMI を起動するユーザーは、それを再バンドルしたり、自分のものとして登録したりできる可能性があります。以下のガイドラインに従って、見落としやすいセキュリティ上のリスクを回避してください:

- --exclude *directory* で ec2-bundle-vol オプションを使用して、バンドル操作に含めたくない機密情報が入っているディレクトリおよびサブディレクトリをスキップすることをお勧めします。特に、イメージをバンドルするときに、すべてのユーザー所有の SSH パブリックキー/プライベートキーペアおよび SSH authorized_keys ファイルを除外します。Amazon パブリック AMI で、これらのファイルは、ルートユーザーの場合は /root/.ssh、通常のユーザーの場合は /home/*user_name*/.ssh/ に配置されています。詳細については、「ec2-bundle-vol」を参照してください。

- バンドルの前に必ずシェル履歴を削除してください。同じ AMI で複数のバンドルのアップロードを試行すると、シェル履歴にアクセスキーが含まれます。次の例は、インスタンス内からのバンドルの前に実行される最後のコマンドとなる必要があります。

```
[ec2-user ~]$ shred -u ~/.*history
```

> **Warning**
>
> 上記の警告で示した shred の制限は、ここにも適用されます。
> bash は、終了時に現在のセッション履歴をディスクに書き込むことに注意してください。~/.bash_history を削除後にインスタンスをログアウトし、再度ログインすると、~/.bash_history が再作成され、前のセッション中に実行されたすべてのコマンドが含まれています。
> bash 以外の他のプログラムもディスクに履歴を書き込むため、注意して不要な dot ファイルと dot ディレクトリを削除または除外します。

- 実行中のインスタンスをバンドルするには、プライベートキーと X.509 証明書が必要です。これらの証明書およびその他の証明書を、バンドルされていない場所 (インスタンスストアなど) に書き込みます。

Amazon EventBridge を使用して AMI イベントをモニタリングする

Amazon マシンイメージ (AMI) の状態に変更があった場合、Amazon EC2 はイベントを生成し、それを Amazon EventBridge (旧 Amazon CloudWatch Events) に送信します。イベントは、EventBridge のデフォルトのイベントバスに、JSON 形式で送信されます。Amazon EventBridge を使用することで、これらのイベントの検出と対応が行えるようになります。EventBridge では、イベントに応答してアクションをトリガーするためのルールを作成します。例えば、AMI 作成プロセスが完了したことを検出し、Amazon SNS トピックを呼び出して E メール通知をユーザーに送信する、EventBridge ルールを作成できます。

AMI が以下のいずれかの状態に遷移すると、Amazon EC2 は `EC2 AMI State Change` イベントを生成します。

- `available`
- `failed`
- `deregistered`

- `disabled`

イベントは、ベストエフォートベースで生成されます。

次のテーブルは、AMI の操作と AMI が入力できる状態を示しています。テーブルの [はい] は、対応する操作が実行されたときに AMI が入力できる状態を示しています。

AMI オペレーション	available	failed	deregistered	disabled
CopyImage	可能	はい		
CreateImage	はい	はい		
CreateRestoreImageTask	はい	はい		
DeregisterImage			はい	
DisableImage				はい
EnableImage	はい			
RegisterImage	はい	あり		

EC2 AMI State Change のイベント

- イベントの詳細
- available のイベント
- failed のイベント
- deregistered のイベント
- disabled のイベント

イベントの詳細

イベント内の以下のフィールドを使用すると、アクションをトリガーするルールを作成できます。

`"source": "aws.ec2"`

イベントが Amazon EC2 からのものであるかを特定します。

`"detail-type": "EC2 AMI State Change"`

イベント名を特定します。

`"detail": { "ImageId": "ami-0abcdef1234567890", "State": "available", }`

AMI ID と AMI の状態 (`available`、`failed`、`deregistered`、または `disabled`) を指定します。

詳細については、「Amazon EventBridge ユーザーガイド」で以下のトピックを参照してください。

- Amazon EventBridge イベント
- Amazon EventBridge のイベントパターン
- Amazon EventBridge ルール

Lambda 関数を作成する方法と、その Lambda 関数を実行する EventBridge ルールのチュートリアルについては、「AWS Lambda デベロッパーガイド」の「チュートリアル: EventBridge を使用して Amazon EC2 インスタンスの状態を口グに記録する」を参照してください。

available のイベント

以下に、`CreateImage`、`CopyImage`、`RegisterImage`、`CreateRestoreImageTask`、または `EnableImage` が正常に処理された後、AMI が `available` 状態に遷移する際に Amazon EC2 が生成するイベントの例を示します。

`"State": "available"` は、 のオペレーションが正常に処理されたことを示します。

```
{
    "version": "0",
    "id": "example-9f07-51db-246b-d8b8441bcdf0",
    "detail-type": "EC2 AMI State Change",
    "source": "aws.ec2",
    "account": "012345678901",
    "time": "yyyy-mm-ddThh:mm:ssZ",
    "region": "us-east-1",
    "resources": ["arn:aws:ec2:us-east-1::image/ami-0abcdef1234567890"],
    "detail": {
        "RequestId": "example-9dcc-40a6-aa77-7ce457d5442b",
```

```
        "ImageId": "ami-0abcdef1234567890",
        "State": "available",
        "ErrorMessage": ""
    }
}
```

failed のイベント

以下に、CreateImage、CopyImage、RegisterImage、または CreateRestoreImageTask が正常に処理された後、AMI が failed 状態に遷移する際に Amazon EC2 が生成するイベントの例を示します。

以下のフィールドにより、関連する情報が提供されます。

- "State": "failed" – オペレーションが失敗したことを示します。
- "ErrorMessage": "" – オペレーション失敗の理由を示します。

```
{
    "version": "0",
    "id": "example-9f07-51db-246b-d8b8441bcdf0",
    "detail-type": "EC2 AMI State Change",
    "source": "aws.ec2",
    "account": "012345678901",
    "time": "yyyy-mm-ddThh:mm:ssZ",
    "region": "us-east-1",
    "resources": ["arn:aws:ec2:us-east-1::image/ami-0abcdef1234567890"],
    "detail": {
        "RequestId": "example-9dcc-40a6-aa77-7ce457d5442b",
        "ImageId": "ami-0abcdef1234567890",
        "State": "failed",
        "ErrorMessage": "Description of failure"
    }
}
```

deregistered のイベント

以下に、DeregisterImage が正常に処理された後、AMI が deregistered 状態に遷移する際に Amazon EC2 が生成するイベントの例を示します。オペレーションが失敗した場合、イベントの生成は行われません。DeregisterImage は同期オペレーションであるため、処理が失敗した場合は直ちに認識されます。

"State": "deregistered" は、DeregisterImage のオペレーションが正常に処理されたことを示します。

```
{
    "version": "0",
    "id": "example-9f07-51db-246b-d8b8441bcdf0",
    "detail-type": "EC2 AMI State Change",
    "source": "aws.ec2",
    "account": "012345678901",
    "time": "yyyy-mm-ddThh:mm:ssZ",
    "region": "us-east-1",
    "resources": ["arn:aws:ec2:us-east-1::image/ami-0abcdef1234567890"],
    "detail": {
        "RequestId": "example-9dcc-40a6-aa77-7ce457d5442b",
        "ImageId": "ami-0abcdef1234567890",
        "State": "deregistered",
        "ErrorMessage": ""
    }
}
```

disabled のイベント

以下に、DisableImage が正常に処理された後、AMI が disabled 状態に遷移する際に Amazon EC2 が生成するイベントの例を示します。オペレーションが失敗した場合、イベントの生成は行われません。DisableImage は同期オペレーションであるため、処理が失敗した場合は直ちに認識されます。

"State": "disabled" は、DisableImage のオペレーションが正常に処理されたことを示します。

```
{
    "version": "0",
    "id": "example-9f07-51db-246b-d8b8441bcdf0",
    "detail-type": "EC2 AMI State Change",
    "source": "aws.ec2",
    "account": "012345678901",
    "time": "yyyy-mm-ddThh:mm:ssZ",
    "region": "us-east-1",
    "resources": ["arn:aws:ec2:us-east-1::image/ami-0abcdef1234567890"],
    "detail": {
        "RequestId": "example-9dcc-40a6-aa77-7ce457d5442b",
        "ImageId": "ami-0abcdef1234567890",
```

```
        "State": "disabled",
        "ErrorMessage": ""
    }
}
```

AMI の請求情報について

インスタンスの起動時に選択できる Amazon マシンイメージ (AMI) は多数あり、さまざまなオペレーティングシステムプラットフォームと機能をサポートしています。AWS からの最終的な請求金額に対し、インスタンスの起動時に選択した AMI がどのように影響するかは、関連するオペレーティングシステムプラットフォームと請求情報を調べることで知ることができます。オンデマンドまたは スポットインスタンス を起動するか、リザーブドインスタンス を購入する前に、この操作を行ってください。

以下、どのように AMI を事前調査することで、ニーズに最も適した AMI を選択できるかを示した例を 2 つ紹介します。

- スポットインスタンス では、AMI プラットフォームの詳細を使用して、AMI が スポットインスタンス でサポートされていることが確認できます。
- リザーブドインスタンス を購入する際、AMI プラットフォームの詳細にマップするオペレーティングシステムプラットフォーム (プラットフォーム) を選択するようにできます。

インスタンスの料金の詳細については、「Amazon EC2 料金表」を参照してください。

コンテンツ

- AMI 請求情報フィールド
- AMI の請求と使用状況の詳細の検索
- 請求書に記載されている AMI の請求を確認します

AMI 請求情報フィールド

次のフィールドは、AMI に関連付けられた請求情報を提供します。

プラットフォームの詳細

> AMI の請求コードに関連付けられたプラットフォームの詳細。例えば、`Red Hat Enterprise Linux`。

使用オペレーション

Amazon EC2 インスタンスのオペレーション、および AMI に関連付けられている請求コード。例えば、`RunInstances:0010`。[使用オペレーション] は、AWS のコストと使用状況レポート (CUR) の 明細項目/オペレーション列と、AWS API 価格表 に対応しています。

これらのフィールドは、Amazon EC2 コンソールの 「インスタンス」ページまたは「AMI」ページ、あるいは [describe-images] コマンドまたは [Get-EC2Image] コマンドによって返されるレスポンスで表示できます。

サンプルデータ: プラットフォーム別の使用オペレーション

次の表は、Amazon EC2 コンソールの 「インスタンス」ページまたは「AMI」ページ、あるいは [describe-images] コマンドまたは [Get-EC2Image] コマンドによって返されるレスポンスに表示される、プラットフォームの詳細と使用オペレーションの値の一部を一覧表示しています。

プラットフォームの詳細	使用オペレーション [2]
Linux/UNIX	RunInstances
Red Hat BYOL Linux	RunInstances:00g0 [3]
Red Hat Enterprise Linux	RunInstances:0010
Red Hat Enterprise Linux with HA	RunInstances:1010
Red Hat Enterprise Linux with SQL Server Standard and HA	RunInstances:1014
Red Hat Enterprise Linux with SQL Server Enterprise and HA	RunInstances:1110
Red Hat Enterprise Linux with SQL Server Standard	RunInstances:0014
Red Hat Enterprise Linux with SQL Server Web	RunInstances:0210

プラットフォームの詳細	使用オペレーション [2]
Red Hat Enterprise Linux with SQL Server Enterprise	RunInstances:0110
SQL Server Enterprise	RunInstances:0100
SQL Server Standard	RunInstances:0004
SQL Server Web	RunInstances:0200
SUSE Linux	RunInstances:000g
Ubuntu Pro	RunInstances:0g00
Windows	RunInstances:0002
Windows BYOL	RunInstances:0800
Windows with SQL Server Enterprise [1]	RunInstances:0102
Windows with SQL Server Standard [1]	RunInstances:0006
Windows with SQL Server Web [1]	RunInstances:0202

[1] 2 つのソフトウェアライセンスが 1 つの AMI に関連付けられている場合、[プラットフォームの詳細] フィールドには両方が表示されます。

[2] スポットインスタンス を実行している場合、[AWS のコストと使用状況レポート] の lineitem/Operation は、ここに記載されている [使用オペレーション] の値と異なる場合があります。例えば、`lineitem/Operation` に `RunInstances:0010:SV006` が表示されている場合は、Amazon EC2 が、ゾーン 6 の米国東部 (バージニア北部) で Red Hat Enterprise Linux スポットインスタンス時間を実行していることを示します。

[3] こちらは、使用状況レポートに RunInstances (Linux/UNIX) のように表示されます。

AMI の請求と使用状況の詳細の検索

請求書の AMI 料金を確認するには、次のプロパティが役立ちます。

- プラットフォームの詳細
- 使用操作
- AMI ID

Console

AMI の AMI 請求情報を見つけるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. AMI を選択します。
4. [詳細] タブで、[プラットフォーム詳細] と [使用オペレーション] を見つけます。

インスタンスの AMI 請求情報を見つけるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択します。
4. [詳細] タブで、[インスタンスの詳細] を展開し、[プラットフォーム詳細] と [使用オペレーション] を見つけます。

AWS CLI

AMI の AMI 請求情報を見つけるには

describe-images コマンドを使用します。

```
aws ec2 describe-images \
    --image-ids ami-0abcdef1234567890 \
    --query "Images[].
{PlatformDetails:PlatformDetails,UsageOperation:UsageOperation}"
```

Linux AMI の出力例を次に示します。

```
[
    {
        "PlatformDetails": "Linux/UNIX",
        "UsageOperation": "RunInstances"
    }
]
```

インスタンスの AMI 請求情報を見つけるには

describe-instances コマンドを使用します。

```
aws ec2 describe-instances \
    --instance-ids i-1234567890abcdef0 \
    --query "Reservations[].Instances[].
{PlatformDetails:PlatformDetails,UsageOperation:UsageOperation}"
```

Windows インスタンスの出力例を次に示します。

```
[
    {
        "PlatformDetails": "Windows",
        "UsageOperation": "RunInstances:0002"
    }
]
```

PowerShell

AMI の AMI 請求情報を見つけるには

Get-EC2Image コマンドレットを使用します。

```
Get-EC2Image `
    -ImageId ami-0abcdef1234567890 | `
    Format-List PlatformDetails, UsageOperation
```

Linux AMI の出力例を次に示します。

```
PlatformDetails : Linux/UNIX
UsageOperation  : RunInstances
```

インスタンスの AMI 請求情報を見つけるには

Get-EC2Instance コマンドレットを使用します。

```
(Get-EC2Instance `
    -InstanceId i-1234567890abcdef0).Instances | `
    Format-List PlatformDetails, UsageOperation
```

Windows インスタンスの出力例を次に示します。

```
PlatformDetails : Windows
UsageOperation  : RunInstances:0002
```

請求書に記載されている AMI の請求を確認します

AWS のコストと使用状況レポート (CUR) で示されたインスタンスの請求情報が、そのインスタンスの起動に使用した AMI に関連付けられた請求情報と一致していることを確認することで、予定外のコストの発生を防ぐことができます。

請求情報を確認するには、CUR でインスタンス ID を見つけ、lineitem/Operation 列で対応する値を確認します。値は、AMI に関連付けられた [使用オペレーション] の値と一致する必要があります。

例えば、AMI ami-0123456789EXAMPLEには次の請求情報があります。

- プラットフォームの詳細 = Red Hat Enterprise Linux
- 使用オペレーション = RunInstances:0010

この AMI を使用してインスタンスを起動した場合は、CUR でインスタンス ID を検索し、lineitem/Operation 列で対応する値が確認できます。この例では、値は RunInstances:0010 であることが必要です。

Amazon EC2 の AMI クォータ

AMI を作成および共有する際には、以下のクォータが適用されます。AWS リージョン ごとにクォータが適用されます。

クォータ名	説明	リージョンあたりのデフォルトのクォータ
AMI	リージョンごとに許可されているパブリック AMI およびプライベート AMI の最大数。これらには、利用可能な AMI と保留中の AMI、およびごみ箱にある AMI が含まれます。	50,000
パブリック AMI	リージョンごとに許可されているパブリック AMI の最大数 (ごみ箱内のパブリック AMI を含む)。	5
AMI 共有	リージョン内で AMI を共有できるエンティティ (組織、組織単位 (OU)、アカウント) の最大数。AMI を組織または OU と共有する場合、組織内のアカウント数や OU 数はクォータにカウントされません。	1,000

クォータを超えて AMI をさらに作成または共有したい場合は、以下を実行できます。

- AMI またはパブリック AMI のクォータの合計を超える場合は、未使用のイメージの登録を解除することを検討してください。
- パブリック AMI のクォータを超える場合は、1 つ以上のパブリック AMI をプライベートにすることを検討してください。
- AMI の共有クォータを超える場合は、個別のアカウントではなく、組織または OU と AMI を共有することを検討してください。
- AMI のクォータの引き上げをリクエストします。

AMI のクォータの引き上げをリクエストする

AMI のデフォルトクォータを超える容量が必要な場合は、クォータの引き上げをリクエストできます。

AMI のクォータの引き上げをリクエストするには

1. https://console.aws.amazon.com/servicequotas/ で Service Quotas コンソールを開きます。
2. ナビゲーションペインで、[AWS サービス] を選択します。
3. リストから [Amazon Elastic Compute Cloud (Amazon EC2)] を選択するか、検索ボックスにサービスの名前を入力します。
4. 引き上げをリクエストするには、AMI クォータを選択します。選択できる AMI クォータは次のとおりです:
   - AMI
   - パブリック AMI
   - AMI 共有
5. [Request quota increase] (クォータの引き上げのリクエスト) を選択します。
6. [Change quota value] (クォータ値の変更) に新しいクォータ値を入力し、[Request] (リクエスト) を選択します。

保留中または最近解決されたリクエストを表示するには、ナビゲーションペインから [ダッシュボード] を選択します。保留中のリクエストの場合は、リクエストのステータスを選択してリクエストの受信をオープンします。リクエストの初期ステータスは [Pending] (保留中) です。ステータスが [Quota requested] (クォータをリクエスト済み) に変わると、[Support Center case number] (サポートセンターのケース番号) にケース番号が表示されます。リクエストのチケットを開くには、ケース番号を選択します。

リクエストが解決されると、クォータの [適用されたクォータ値] が新しい値に設定されます。

詳細については、Service Quotas ユーザーガイドを参照してください。

Amazon EC2 インスタンス

Amazon EC2 インスタンスは AWS クラウド環境の仮想サーバーです。インスタンスを最初に起動した時点 (インスタンスの起動と呼びます) から、インスタンスを削除する時点 (インスタンスの終了と呼びます) まで、インスタンスを完全に制御できます。インスタンスの起動時に、さまざまなオペレーティングシステムから選択できます。インスタンスに接続し、ニーズに合わせてカスタマイズできます。例えば、オペレーティングシステムの設定、オペレーティングシステムの更新のインストール、インスタンスへのアプリケーションのインストールを行うことができます。

Amazon EC2 は、幅広いインスタンスタイプを提供します。アプリケーションの実行に必要なコンピューティングリソース、メモリ、ストレージ、ネットワークパフォーマンスを提供するインスタンスタイプを選択できます。

Amazon EC2 については、お客様が利用された分のみのお支払いとなります。インスタンスの請求は、インスタンスを起動して実行状態に移行すると開始されます。インスタンスを停止すると請求は停止し、インスタンスの起動時に再開されます。インスタンスを終了すると、シャットダウン状態への移行時に請求が停止します。

Amazon EC2 には、インスタンスのパフォーマンスとコストを最適化するために使用できる機能が用意されています。例えば、Amazon EC2 Fleet または Amazon EC2 Auto Scaling を使用して、インスタンス使用率の変化に応じてキャパシティをスケールアップまたはスケールダウンできます。スポットインスタンスまたは Savings Plans を使用して、インスタンスのコストを削減できます。

マネージドインスタンスは、Amazon EKS オートモードなどのサービスプロバイダーによって管理されます。マネージドインスタンスの設定を直接変更することはできません。マネージドインスタンスは、マネージドフィールドの true 値によって識別されます。詳細については、Amazon EC2 マネージドインスタンスを参照してください。

機能とタスク

- Amazon EC2 インスタンスタイプ
- Amazon EC2 マネージドインスタンス
- ネストされた仮想化を使用して Amazon EC2 インスタンスでハイパーバイザーを実行する
- Amazon EC2 の請求および購入オプション
- Amazon EC2 起動テンプレートにインスタンス起動パラメータを保存する
- Amazon EC2 インスタンスの起動
- EC2 インスタンスに接続する

- Amazon EC2 インスタンスの状態変更
- インスタンスの自動復旧
- インスタンスメタデータを使用して EC2 インスタンスを管理する
- ホストが EC2 インスタンスであるかどうかを検出する
- Amazon EC2 インスタンスのインスタンスアイデンティティドキュメント
- EC2 インスタンスの STIG コンプライアンス
- EC2 インスタンス上の正確なクロックと時刻の同期
- EC2 Capacity Manager
- EC2 インスタンスのデバイスドライバーを管理する
- Amazon EC2 Windows インスタンスの設定
- EC2 Windows インスタンスのより新しいバージョンの Windows Server へのアップグレード
- チュートリアル: Amazon RDS データベースに Amazon EC2 インスタンスを接続する

Amazon EC2 インスタンスタイプ

インスタンスを起動するときは指定したインスタンスタイプによって、インスタンスに使用するホストコンピュータのハードウェアが決まります。インスタンスタイプごとに、コンピューティング、メモリ、およびストレージの機能が異なっており、これらの機能に基づいたインスタンスファミリーにグループ化されています。インスタンスタイプは、インスタンス上で実行するアプリケーションやソフトウェアの要件に基づいて選択します。機能およびユースケースの詳細については、「Amazon EC2 インスタンスのタイプ」を参照してください。

Amazon EC2 ではCPU、メモリ、インスタンスストレージなどホストコンピュータの一部のリソースを、特定のインスタンス専用に割り当てます。ネットワークやディスクサブシステムなどホストコンピュータでの他のリソースはAmazon EC2 によりインスタンス間で共有されます。ホストコンピュータの各インスタンスが、これらの共有リソースの 1 つを可能な限り利用しようとする場合、それぞれのインスタンスはそのリソースの共有分を等しく受け取ります。ただし、リソースの使用率が低い場合は 1 つのインスタンスがそのリソースのより多くの部分を利用できます。

各インスタンスタイプは共有リソースからからより高い、またはより低い最小性能を提供します。例えば、高速の I/O パフォーマンスを実行するインスタンスタイプは共有リソースに対してより大きな割り当てを取得します。共有リソースをより大きく配分することによって、I/O 性能のばらつきを抑えることもできます。ほとんどのアプリケーションでは中程度の I/O 性能があれば十分です。ただし、より高い、またはより一貫した I/O パフォーマンスを必要とするアプリケーションの場合はより I/O パフォーマンスの高いインスタンスタイプを使用することを検討してください。

内容

- 利用可能なインスタンスタイプ
- ハードウェア仕様
- ハイパーバイザーのタイプ
- AMI 仮想化タイプ
- Processors
- Amazon EC2 インスタンスタイプの検索
- EC2 インスタンスタイプファインダーからレコメンデーションを取得する
- Compute Optimizer から EC2 インスタンスの推奨事項を取得する
- Amazon EC2 インスタンスタイプの変更
- バーストパフォーマンスインスタンス
- GPU インスタンスによるパフォーマンスアクセラレーション
- Amazon EC2 Mac インスタンス
- Amazon EBS 最適化インスタンスタイプ
- Amazon EC2 インスタンスに関する CPU オプション
- Amazon EC2 インスタンスの AMD SEV-SNP
- Amazon EC2 Linux インスタンスのプロセッサのステート制御

利用可能なインスタンスタイプ

Amazon EC2 では幅広いインスタンスタイプの選択肢があり、さまざなユースケースに合わせて最適化できます。インスタンスタイプはCPU、メモリ、ストレージ、およびネットワーク容量のさまざまな組み合わせで構成され、アプリケーションに適したリソースの組み合わせを柔軟に選択できます。各インスタンスタイプには 1 つ以上のインスタンスサイズが含まれているため、ターゲットワークロードの要件に合わせてリソースをスケーリングできます。

インスタンスタイプの命名規則

名前はインスタンスファミリー、世代、プロセッサファミリー、機能、サイズに基づいています。詳細については「Amazon EC2 インスタンスタイプガイド」の「Naming conventions」を参照してください。

インスタンスタイプの検索

サポート対象のリージョン、コンピューティングリソース、ストレージリソースなどの要件を満たすインスタンスタイプを判断するには「Amazon EC2 インスタンスタイプの検索」および「Amazon EC2 インスタンスタイプガイド」の「Amazon EC2 インスタンスタイプの仕様」を参照してください。

ハードウェア仕様

インスタンスタイプの詳細な仕様については「Amazon EC2 インスタンスタイプガイド」の「Specifications」を参照してください。料金については「Amazon EC2 オンデマンド料金」をご覧ください。

お客様のニーズに最適なインスタンスタイプを決定するにはインスタンスを起動し、独自のベンチマークアプリケーションを使用することをお勧めします。支払いはインスタンス秒単位であるため、決定する前に複数のインスタンスタイプをテストすると、便利なうえ、コストを抑えることができます。決定を行った後でも、ニーズが変化したときはインスタンスタイプを変更できます。詳細については「Amazon EC2 インスタンスタイプの変更」を参照してください。

ハイパーバイザーのタイプ

Amazon EC2 はハイパーバイザーとして Xen と Nitro をサポートしています。

Nitro ベースのインスタンス

- 汎用: M5 | M5a | M5ad | M5d | M5dn | M5n | M5zn | M6a | M6g | M6gd | M6i | M6id | M6idn | M6in | M7a | M7g | M7gd | M7i | M7i-flex | M8a | M8azn | M8g | M8gb | M8gd | M8gn | M8i | M8id | M8i-flex | M8in | M8idn | M8ine | M8ib | M8idb | M9g | M9gd | T3 | T3a | T4g
- コンピューティング最適化: C5 | C5a | C5ad | C5d | C5n | C6a | C6g | C6gd | C6gn | C6i | C6id | C6in | C7a | C7g | C7gd | C7gn | C7i | C7i-flex | C8a | C8g | C8gb | C8gd | C8gn | C8i | C8id | C8i-flex | C8in | C8ine | C8ib | C9g | C9gd
- メモリ最適化: R5 | R5a | R5ad | R5b | R5d | R5dn | R5n | R6a | R6g | R6gd | R6i | R6id | R6idn | R6in | R7a | R7g | R7gd | R7i | R7iz | R8a | R8g | R8gb | R8gd | R8gn | R8i | R8id | R8i-flex | R8in |R8idn | R8ib |R8idb | U-3tb1 | U-6tb1 | U-9tb1 | U-12tb1 | U-18tb1 | U-24tb1 | U7i-6tb | U7i-8tb | U7i-12tb | U7in-16tb | U7in-24tb | U7in-32tb | U7inh-32tb | X2gd | X2idn | X2iedn | X2iezn | X8g | X8aedz | X8i | z1d
- ストレージ最適化: D3 | D3en | I3en | I4g | I4i | I7i | I7ie | I8g | I8ge | Im4gn | Is4gen
- 高速コンピューティング: DL1 | DL2q | F2 | G4ad | G4dn | G5 | G5g | G6 | G6e | G6f | Gr6 | Gr6f | G7 | G7e | Inf1 | Inf2 | P4d | P4de | P5 | P5e | P5en | P6-B200 | P6-B300 | P6e-GB200 | Trn1 | Trn1n | Trn2 | Trn2u | VT1

- ハイパフォーマンスコンピューティング: Hpc6a | Hpc6id | Hpc7a | Hpc7g | Hpc8a
- 前の世代: A1 | P3dn

Nitro Hypervisor に対応しているバージョンの詳細については「Amazon EC2 インスタンスタイプガイド」の「Network feature support」を参照してください。

Xen ベースのインスタンス

- 汎用: M1 | M2 | M3 | M4 | T1 | T2
- コンピューティング最適化: C1 | C3 | C4
- メモリ最適化: R3 | R4 | X1 | X1e
- ストレージ最適化: D2 | H1 | I2 | I3
- 高速コンピューティング: G3 | P3

AMI 仮想化タイプ

インスタンスの仮想化タイプはインスタンスの起動に使用する AMI によって決まります。現行世代のインスタンスタイプはハードウェア仮想マシン (HVM) のみをサポートしています。以前の世代のインスタンスタイプの中には準仮想化 (PV) をサポートするものがあり、一部の AWS リージョンは PV インスタンスをサポートしています。詳細については「仮想化タイプ」を参照してください。

最適なパフォーマンスを得るために、HVM AMI を使用することをお勧めします。さらに、拡張ネットワーキングのメリットを活用するにはHVM AMI が必要です。HVM 仮想化はAWS プラットフォームによって提供されるハードウェアアシストテクノロジーを使用します。HVM 仮想化を使用すると、ゲスト VM はネイティブハードウェアプラットフォーム上で動作しているかのように動作します。ただし、パフォーマンスの向上のために PV ネットワークとストレージドライバは使用します。

Processors

EC2 インスタンスはさまざまなプロセッサをサポートしています。

Processors

- インテルプロセッサ
- AMD プロセッサ
- AWS Graviton プロセッサ
- AWS Trainium

- AWS Inferentia

インテルプロセッサ

Intel プロセッサで動作する Amazon EC2 インスタンスには以下のプロセッサ機能が含まれている可能性があります。Intel プロセッサで動作するすべてのインスタンスが、これらのプロセッサ機能をすべてサポートするとは限りません。各インスタンスタイプで使用できる機能の詳細については「Amazon EC2 インスタンスタイプ」を参照してください。

- インテルの AES New Instructions (AES-NI) — インテルの AES-NI 暗号化命令セットはオリジナルの Advanced Encryption Standard (AES) アルゴリズムを改良し、より高速なデータ保護とより優れたセキュリティを提供します。現行世代の全 EC2 インスタンスがこのプロセッサ機能をサポートしています。
- Intel Advanced Vector Extensions (Intel AVX、Intel AVX2、および Intel AVX-512) — 浮動小数点 (FP) 集約型のアプリケーション用に設計された命令セット拡張で、Intel AVX および Intel AVX2 は 256 ビット、Intel AVX-512 は 512 ビットです。Intel AVX 命令は画像およびオーディオ/ビデオ処理、科学的シミュレーション、財務分析、および 3D モデリングと分析などのアプリケーションに対するパフォーマンスを向上させます。これらの機能はHVM AMI で起動されたインスタンスのみで利用できます。
- Intel Turbo Boost Technology — Intel Turbo Boost Technology プロセッサーは定格の動作周波数よりも高速にコアを自動的に実行します。
- Intel Deep Learning Boost (Intel DL Boost) — AI の深層学習のユースケースを高速化します。第 2 世代インテル Xeon スケーラブルプロセッサでは新しいベクトルニューラルネットワーク命令 (VNNI/INT8) を使ってインテル AVX-512 を拡張します。これにより、画像認識/セグメント化、物体検出、音声認識、言語翻訳、レコメンデーションシステム、強化学習などにおける深層学習の推論パフォーマンスは旧世代のインテル Xeon スケーラブルプロセッサ (FP32) よりも大幅に向上します。VNNI はすべての Linux ディストリビューションと互換性があるわけではありません。

  `M5n`、`R5n`、`M5dn`、`M5zn`、`R5b`、`R5dn`、`D3`、`D3en` および `C6i` インスタンスではVNNI をサポートしています。`C5` および `C5d` インスタンスでは`12xlarge`、`24xlarge`、`metal` インスタンスのみ VNNI をサポートしています。

これは64 ビット CPU の命名に関する業界の慣習の影響であり、ややわかりにくいものになっています。チップ製造元の Advanced Micro Devices (AMD) はIntel x86 命令セットをベースとして商業的に初めて成功した 64 ビットアーキテクチャを導入しました。その結果、このアーキテクチャーはチップ製造元にかかわらず AMD64 と幅広く呼ばれています。Windows および複数の Linux ディス

トリビューションがこの慣習に従っています。インスタンスがインテルハードウェアで実行されているにもかかわらず、Ubuntu または Windows で実行されているインスタンス上の内部システム情報に CPU アーキテクチャが AMD64 と表示されるのはこのためです。

AMD プロセッサ

AMD EPYC プロセッサで動作する Amazon EC2 インスタンスはワークロードのコストとパフォーマンスのどちらも最適化できます。こうしたインスタンスでサポート可能なプロセッサ機能には次のものがあります。AMD プロセッサで動作するすべてのインスタンスが、これらのプロセッサ機能をすべてサポートするとは限りません。各インスタンスタイプで使用できる機能の詳細については「Amazon EC2 インスタンスタイプ」を参照してください。

- AMD Secure Memory Encryption (SME)
- AMD Transparent Single Key Memory Encryption (TSME)
- AMD Advanced Vector Extensions (AVX)
- AMD Secure Encrypted Virtualization-Secure Nested Paging (SEV-SNP)
- Vector Neural Network Instructions (VNNI)
- BFloat16

AWS Graviton プロセッサ

AWS Graviton は Amazon EC2 インスタンスで実行されるワークロードに最高のコストパフォーマンスを提供するように設計されたプロセッサファミリーです。

詳細については「Getting started with Graviton」を参照してください。

AWS Trainium

AWS Trainium を搭載したインスタンスは高性能で費用対効果の高い深層学習トレーニングを目的として構築されています。このインスタンスを使用すると、音声認識、レコメンデーション、不正検出、イメージや動画の分類など、幅広いアプリケーションで使用される自然言語処理、コンピュータビジョン、レコメンダーモデルをトレーニングできます。PyTorch や TensorFlow などのよく使用される ML フレームワークで、既存のワークフローを使用できます。

AWS Inferentia

AWS Inferentia を搭載したインスタンスは機械学習を高速化するように設計されており、高性能で低レイテンシーの機械学習推論を実現します。これらのインスタンスは自然言語処理、オブジェクトの

検出と分類、コンテンツのパーソナライズとフィルタリング、音声認識などのアプリケーション向け深層学習 (DL) モデルをデプロイするために最適化されています。

使用を開始するにはさまざまな方法があります。

- 機械学習モデルの使用を開始する最も簡単な方法であり、フルマネージド型のサービスである SageMaker を使用します。詳細については、「Amazon SageMaker AI 開発者ガイド」の「SageMaker の使用開始 を参照してください。
- 深層学習 AMI を使用して Inf1 または Inf2 インスタンスを起動します。詳細については AWS デベロッパーガイドの DLAMI を使用した AWS Deep Learning AMIs Inferentiaを参照してください。
- 独自の AMI を使用して Inf1 または Inf2 インスタンスを起動し、AWS Neuron SDK をインストールします。これにより、AWS Inferentia の深層学習モデルをコンパイル、実行、プロファイリングできます。
- Inf1 または Inf2 インスタンスと Amazon ECS 最適化 AMI を使用してコンテナインスタンスを起動します。詳細については、Amazon Elastic Container Service デベロッパーガイドの「Amazon Linux 2 (Inferentia) AMI」を参照してください。
- Inf1 インスタンスを実行するノードを持つ Amazon EKS クラスターを作成します。詳細については、Amazon EKS ユーザーガイドの「Inferentia のサポート」を参照してください。

Amazon EC2 インスタンスタイプの検索

インスタンスの起動前には、使用するインスタンスタイプを選択しなければなりません。選択するインスタンスタイプは、コンピューティング、メモリ、ストレージリソースなど、ワークロードが必要とするリソースに応じて異なる場合があります。ワークロードに適したインスタンスタイプをいくつか特定し、テスト環境でそれらのパフォーマンスを評価することは有益な場合があります。負荷時のアプリケーションのパフォーマンスを測定するための、代替手段はありません。

EC2 インスタンスタイプファインダーを使用することで、EC2 インスタンスタイプに関する提案やガイダンスを得ることができます。詳細については、「the section called “EC2 インスタンスタイプファインダー”」を参照してください。

EC2 インスタンスをすでに実行している場合は、AWS Compute Optimizer を使用すると、パフォーマンスの向上、コストの削減、またはその両方に使用するインスタンスタイプの、推奨事項を取得できます。詳細については、the section called “Compute Optimizer 推奨事項” を参照してください。

タスク

- コンソールを使用したインスタンスタイプの検索

- AWS CLI を使用してインスタンスタイプを記述する
- AWS CLI を使用したインスタンスタイプの検索
- Tools for PowerShell を使用してインスタンスタイプを見つける

コンソールを使用したインスタンスタイプの検索

Amazon EC2 コンソールを使用して、ニーズに合ったインスタンスタイプを検索できます。

コンソールを使用したインスタンスタイプを検索するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションバーから、インスタンスを起動するリージョンを選択してください。お客様は場所に関係なく、使用できるリージョンをどれでも選択できます。
3. ナビゲーションペインで、[インスタンスタイプ] を選択します。
4. (オプション) プリファレンス (歯車) アイコンを選択して オンデマンド Linux 料金などの表示するインスタンスタイプ属性を選択し、次に [確認] を選択します。または、インスタンスタイプの名前を選択して詳細ページを開き、コンソールで使用できるすべての属性を表示します。このコンソールには、API またはコマンドラインで使用できる属性のすべては表示されません。
5. インスタンスタイプ属性を使用して、ニーズを満たすインスタンスタイプのみがリスト表示されるようにフィルタリングします。例えば、以下の属性でフィルターをかけることができます。

   - [アベイラビリティーゾーン] — アベイラビリティーゾーン、ローカルゾーン、Wavelength Zone の名前。詳細については、the section called “リージョンとゾーン” を参照してください。
   - vCPU または コア — vCPUs またはコアの数。
   - [Memory (GiB)] (メモリ (GiB)) — メモリサイズ (GiB 単位)。
   - [Network performance] (ネットワークパフォーマンス) — ネットワークパフォーマンス (ギガビット)。
   - [Local instance storage] (ローカルインスタンスストレージ) — インスタンスタイプにローカルインスタンスストレージがあるかどうかを示します (`true|false`)。
6. (オプション) 並べて比較するには、複数のインスタンスタイプに対応するチェックボックスをオンにします。比較は、画面下部に表示されます。
7. (オプション) インスタンスタイプのリストをカンマ区切り値 (.csv) ファイルで保存してさらに見直せるようにするには、[Actions] (アクション)、[Download list] (リスト CSV をダウンロード)

の順に選択します。このファイルには、設定したフィルターに一致するすべてのインスタンスタイプが含まれます。

8. (オプション) ニーズを満たすインスタンスタイプを使用してインスタンスを起動するには、インスタンスタイプのチェックボックスをオンにして [Actions] (アクション)、[Launch instance] (インスタンスを起動) の順に選択します。詳細については、「コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する」を参照してください。

AWS CLI を使用してインスタンスタイプを記述する

describe-instance-types コマンドを使用して、特定のインスタンスタイプを記述できます。

インスタンスタイプを詳細に記述するには

次のコマンドは、指定されたインスタンスタイプで使用可能なすべての詳細を表示します。出力には時間がかかるため、ここでは省略します。

```
aws ec2 describe-instance-types \
    --instance-types t2.micro \
    --region us-east-2
```

インスタンスタイプを記述し、出力をフィルタリングする

次のコマンドは、指定されたインスタンスタイプのネットワーキングの詳細を表示します。

```
aws ec2 describe-instance-types \
    --instance-types t2.micro \
    --region us-east-2 \
    --query "InstanceTypes[].NetworkInfo"
```

以下は出力例です。

```
[
    {
        "NetworkPerformance": "Low to Moderate",
        "MaximumNetworkInterfaces": 2,
        "MaximumNetworkCards": 1,
        "DefaultNetworkCardIndex": 0,
        "NetworkCards": [
            {
                "NetworkCardIndex": 0,
                "NetworkPerformance": "Low to Moderate",
```

```
                "MaximumNetworkInterfaces": 2,
                "BaselineBandwidthInGbps": 0.064,
                "PeakBandwidthInGbps": 1.024
            }
        ],
        "Ipv4AddressesPerInterface": 2,
        "Ipv6AddressesPerInterface": 2,
        "Ipv6Supported": true,
        "EnaSupport": "unsupported",
        "EfaSupported": false,
        "EncryptionInTransitSupported": false,
        "EnaSrdSupported": false
    }
]
```

次のコマンドは、指定されたインスタンスタイプで使用可能なメモリを表示します。

```
aws ec2 describe-instance-types \
    --instance-types t2.micro \
    --region us-east-2 \
    --query "InstanceTypes[].MemoryInfo"
```

以下は出力例です。

```
[
    {
        "SizeInMiB": 1024
    }
]
```

AWS CLI を使用したインスタンスタイプの検索

describe-instance-types コマンドと describe-instance-type-offerings コマンドを使用して、ニーズを満たすインスタンスタイプを検出します。

例

- 例 1: アベイラビリティーゾーンでインスタンスタイプを検出する
- 例 2: 使用可能なメモリサイズでインスタンスタイプを検出する
- 例 3: 使用可能なインスタンスストレージでインスタンスタイプを検出する
- 例 4: 休止をサポートするインスタンスタイプを検出する

例 1: アベイラビリティーゾーンでインスタンスタイプを検出する

次の例では、指定されたアベイラビリティーゾーンに使用可能なインスタンスタイプのみを表示しています。

```
aws ec2 describe-instance-type-offerings \
    --location-type "availability-zone" \
    --filters "Name=location,Values=us-east-2a" \
    --region us-east-2 \
    --query "InstanceTypeOfferings[*].[InstanceType]" --output text | sort
```

出力は、アルファベット順に並んだインスタンスタイプのリストです。次に、出力の冒頭部分のみを示します。

```
a1.2xlarge
a1.4xlarge
a1.large
a1.medium
a1.metal
a1.xlarge
c4.2xlarge
   ...
```

例 2: 使用可能なメモリサイズでインスタンスタイプを検出する

次の例では、64 GiB (65536 MiB) のメモリを搭載した現行世代のインスタンスタイプのみを表示しています。

```
aws ec2 describe-instance-types \
    --filters "Name=current-generation,Values=true" "Name=memory-info.size-in-
mib,Values=65536" \
    --region us-east-2 \
    --query "InstanceTypes[*].[InstanceType]" --output text | sort
```

出力は、アルファベット順に並んだインスタンスタイプのリストです。次に、出力の冒頭部分のみを示します。

```
c5a.8xlarge
c5ad.8xlarge
c6a.8xlarge
c6g.8xlarge
```

```
c6gd.8xlarge
c6gn.8xlarge
c6i.8xlarge
c6id.8xlarge
c6in.8xlarge
   ...
```

例 3: 使用可能なインスタンスストレージでインスタンスタイプを検出する

次の例では、インスタンスストアボリュームを備えたすべての R7 インスタンスのインスタンスストレージの合計サイズを表示しています。

```
aws ec2 describe-instance-types \
    --filters "Name=instance-type,Values=r7*" "Name=instance-storage-
supported,Values=true" \
    --region us-east-2 \
    --query "InstanceTypes[].[InstanceType, InstanceStorageInfo.TotalSizeInGB]" \
    --output table
```

以下は出力例です。

```
---------------------------
|  DescribeInstanceTypes  |
+----------------+--------+
|  r7gd.xlarge   |  237   |
|  r7gd.8xlarge  |  1900  |
|  r7gd.16xlarge |  3800  |
|  r7gd.medium   |  59    |
|  r7gd.4xlarge  |  950   |
|  r7gd.2xlarge  |  474   |
|  r7gd.metal    |  3800  |
|  r7gd.large    |  118   |
|  r7gd.12xlarge |  2850  |
+----------------+--------+
```

例 4: 休止をサポートするインスタンスタイプを検出する

次の例では、休止をサポートするインスタンスタイプを表示しています。

```
aws ec2 describe-instance-types \
    --filters "Name=hibernation-supported,Values=true" \
    --region us-east-2 \
```

```
    --query "InstanceTypes[*].[InstanceType]" \
    --output text | sort
```

出力は、アルファベット順に並んだインスタンスタイプのリストです。次に、出力の冒頭部分のみを示します。

```
c4.2xlarge
c4.4xlarge
c4.8xlarge
c4.large
c4.xlarge
c5.12xlarge
c5.18xlarge
c5.2xlarge
c5.4xlarge
c5.9xlarge
...
```

Tools for PowerShell を使用してインスタンスタイプを見つける

Get-EC2InstanceType および Get-EC2InstanceTypeOffering コマンドレットを使用して、ニーズに合ったインスタンスタイプを見つけることができます。

例

- アベイラビリティーゾーンでインスタンスタイプを検出する
- 使用可能なメモリサイズでインスタンスタイプを検出する
- 使用可能なインスタンスストレージでインスタンスタイプを検出する
- 休止をサポートするインスタンスタイプを検出する

アベイラビリティーゾーンでインスタンスタイプを検出する

次の例では、指定されたアベイラビリティーゾーンに使用可能なインスタンスタイプのみを表示しています。

```
(Get-EC2InstanceTypeOffering `
    -LocationType "availability-zone" `
    -Region us-east-2 `
    -Filter @{Name="location"; Values="us-east-2a"}).InstanceType | Sort-Object `
```

使用可能なメモリサイズでインスタンスタイプを検出する

次の例では、64 GiB (65536 MiB) のメモリを搭載した現行世代のインスタンスタイプのみを表示しています。

```
(Get-EC2InstanceType `
    -Filter @{Name="current-generation"; Values="true"},
            @{Name="memory-info.size-in-mib"; Values="65536"}).InstanceType | Sort-
Object
```

使用可能なインスタンスストレージでインスタンスタイプを検出する

次の例では、インスタンスストアボリュームを備えたすべての R7 インスタンスのインスタンスストレージの合計サイズを表示しています。

```
Get-EC2InstanceType `
    -Filter @{Name="instance-type"; Values="r7*"},
            @{Name="instance-storage-supported"; Values="true"} | `
     Select InstanceType, @{Name="TotalSizeInGB";
 Expression={($_.InstanceStorageInfo.TotalSizeInGB)}}
```

以下は出力例です。

```
InstanceType  TotalSizeInGB
------------  -------------
r7gd.8xlarge           1900
r7gd.16xlarge          3800
r7gd.xlarge             237
r7gd.4xlarge            950
r7gd.medium              59
r7gd.2xlarge            474
r7gd.large              118
r7gd.metal             3800
r7gd.12xlarge          2850
```

休止をサポートするインスタンスタイプを検出する

次の例では、休止をサポートするインスタンスタイプを表示しています。

```
(Get-EC2InstanceType `
    -Filter @{Name="hibernation-supported"; Values="true"}).InstanceType | Sort-Object
```

EC2 インスタンスタイプファインダーからレコメンデーションを取得する

新しいワークロード – EC2 インスタンスタイプファインダーは、ユースケース、ワークロードタイプ、CPU メーカーの優先設定、価格とパフォーマンスの優先度の他に、ユーザーが指定できる追加のパラメータも考慮します。そして、このデータを使用して、新しいワークロードに最適な Amazon EC2 インスタンスタイプの推奨とガイダンスを提供します。

利用可能なインスタンスタイプの数が多いため、ワークロードに適したインスタンスタイプを見つけるには時間と手間がかかることがあります。EC2 インスタンスタイプファインダーを使用することで、最新のインスタンスタイプを常に把握し、ワークロードに最適なコストパフォーマンスを実現できます。

Amazon EC2 コンソールを使用して、EC2 インスタンスタイプの提案とガイダンスを取得できます。Amazon Q に直接アクセスして、インスタンスタイプのアドバイスを求めることもできます。詳細については、「Amazon Q Developer ユーザーガイド」を参照してください。

既存のワークロードに適したインスタンスタイプをお探しの場合は、AWS Compute Optimizer を使用します。詳細については「Compute Optimizer から EC2 インスタンスの推奨事項を取得する」を参照してください。

EC2 インスタンスタイプファインダーを使用する

Amazon EC2 コンソールでは、起動テンプレートの作成時に、または [インスタンスタイプ] ページで、インスタンス起動ウィザード内の EC2 インスタンスタイプファインダーからインスタンスタイプの推奨を取得できます。

Amazon EC2 コンソールで EC2 インスタンスタイプファインダーを使用して EC2 インスタンスタイプの推奨とガイダンスを取得するには、次の手順を使用します。手順のアニメーションを見る場合は、「アニメーションを表示する: EC2 インスタンスタイプファインダーを使用してインスタンスタイプの推奨を取得する」を参照してください。

EC2 インスタンスタイプファインダーを使用してインスタンスタイプの推奨を取得するには

1. 次のいずれかを使用してプロセスを開始します。

   - インスタンスを起動するための手順に従います。[インスタンスタイプ] の横にある [アドバイスを取得] リンクを選択します。
   - 次の手順に従って起動テンプレートを作成します。[インスタンスタイプ] の横にある [アドバイスを取得] リンクを選択します。

- ナビゲーションペインで、[インスタンスタイプ] を選択し、[インスタンスタイプファインダー] ボタンを選択します。

2. [インスタンスタイプの選択に関するアドバイスを得る] 画面で、次の操作を行います。

   a. [ワークロードタイプ]、[ユースケース]、[優先度]、および [CPU メーカー] のオプションを選択して、インスタンスタイプの要件を指定します。

   b. (任意) ワークロードのより詳細な要件を指定するには、次の操作を行います。

      i. [詳細パラメータ] を展開します。

      ii. パラメータを追加するには、パラメータを選択してから [追加] を選び、パラメータの値を指定します。追加する追加条件ごとに同じ操作を行います。最小値または最大値を指定しない場合は、フィールドを空のままにします。

      iii. 追加した後にパラメータを削除するには、パラメータの横にある [X] を選択します。

   c. [インスタンスタイプに関するアドバイスを取得] を選択します。

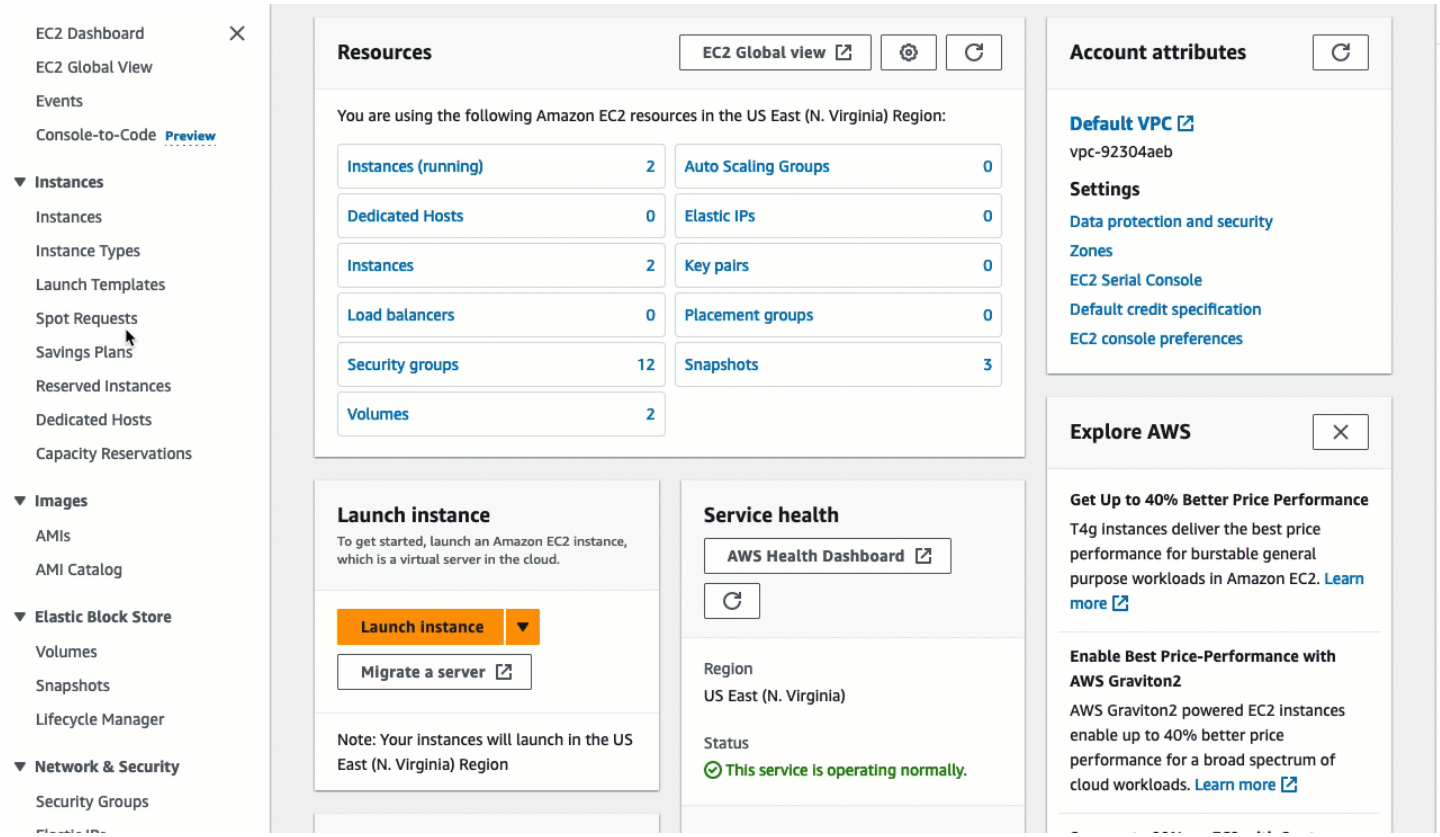
      Amazon EC2 では、指定した要件に一致するインスタンスファミリーの推奨が提供されます。

3. 推奨されたインスタンスファミリー内の各インスタンスタイプの詳細を表示するには、[推奨インスタンスファミリーの詳細を表示] を選択します。

4. 要件を満たすインスタンスタイプを選択し、[アクション]、[インスタンスを起動] または [アクション]、[起動テンプレートを作成] を選択します。

   あるいは、インスタンス起動ウィザードまたは起動テンプレートページでプロセスを開始し、元のフローに戻る場合は、使用するインスタンスタイプを書き留めます。次に、インスタンス起動ウィザードまたは起動テンプレートにおいて、[インスタンスタイプ] でインスタンスタイプを選択し、インスタンスを起動する手順または起動テンプレートを作成する手順を完了します。

アニメーションを表示する: EC2 インスタンスタイプファインダーを使用してインスタンスタイプの推奨を取得する



Compute Optimizer から EC2 インスタンスの推奨事項を取得する

AWS Compute Optimizer から、パフォーマンスの向上、コストの削減、またはその両方に役立つ、Amazon EC2 の推奨事項が得られます。これらの推奨事項を使用して、新しいインスタンスタイプに変更するかどうかを判断できます。

推奨事項を作成するために、Compute Optimizer は既存のインスタンスの仕様と使用率メトリクスを分析します。次に、コンパイルされたデータを使用して、既存のワークロードを処理するのに最適な Amazon EC2 インスタンスタイプを推奨します。推奨事項は、時間あたりのインスタンス料金とともに返されます。詳細については、「AWS Compute Optimizer ユーザーガイド」の「Amazon EC2 インスタンスメトリクス」を参照してください。

内容

- 要件
- 結果の分類
- 推奨事項の表示

- 推奨事項の評価に関する考慮事項

要件

Compute Optimizer から推奨事項を取得するには、まず Compute Optimizer にオプトインする必要があります。詳細については、「AWS Compute Optimizer ユーザーガイド」の「AWS Compute Optimizer の使用開始」を参照してください。

Compute Optimizer は、一部のインスタンスタイプに関する推奨事項を生成しますが、すべてのインスタンスタイプについて推奨事項を生成するわけではありません。サポートされていないインスタンスタイプを使用している場合、Compute Optimizer は推奨事項を生成しません。サポートされているインスタンスタイプのリストについては、「AWS Compute Optimizer ユーザーガイド」の「Amazon EC2 インスタンスの要件」を参照してください。

結果の分類

Compute Optimizer は、EC2 インスタンスの検出結果を以下のように分類します。

- プロビジョニング不足 – EC2 インスタンスは、インスタンス仕様の CPU、メモリ、ネットワークなどの 1 つ以上の要素がワークロードのパフォーマンス要件を満たしていない場合に、プロビジョニング不足と見なされます。EC2 インスタンスがプロビジョニング不足である場合、アプリケーションのパフォーマンスが低下することがあります。
- 過剰プロビジョニング – EC2 インスタンスは、インスタンス仕様の CPU、メモリ、ネットワークなどの 1 つ以上の要素をサイズダウンしてもワークロードのパフォーマンス要件を満たす場合や、どの仕様要素もプロビジョニング不足でない場合に、過剰プロビジョニングと見なされます。EC2 インスタンスの過剰プロビジョニングは、余分なインフラストラクチャコストを発生させる場合があります。
- 最適化 – EC2 インスタンスは、インスタンス仕様の CPU、メモリ、ネットワークなどのすべての要素がワークロードのパフォーマンス要件を満たし、インスタンスが過剰プロビジョニングされていない場合に、最適化されていると見なされます。最適化された EC2 インスタンスは、最適なパフォーマンスとインフラストラクチャコストでワークロードを実行します。最適化されたインスタンスとして、Compute Optimizer は新世代のインスタンスタイプを推奨する場合があります。
- なし – このインスタンスに対する推奨事項はありません。この結果になる可能性があるのは、Compute Optimizer にオプトインしてから 12 時間未満である場合、インスタンスの実行時間が 30 時間未満である場合、またはインスタンスタイプが Compute Optimizer でサポートされていない場合です。

推奨事項の表示

Compute Optimizer にオプトインすると、Compute Optimizer が EC2 インスタンスに関して生成した検出結果を Amazon EC2 コンソールに表示できます。次に、Compute Optimizer コンソールにアクセスして推奨事項を表示できます。最近オプトインした場合は、結果が EC2 コンソールに反映されるまで最大 12 時間かかることがあります。

Amazon EC2 コンソールを使用してインスタンスの推奨事項を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンス ID を選択して、インスタンスの詳細ページを開きます。
4. インスタンス詳細ページの上部にある概要セクションで、[AWS Compute Optimizer の検出結果] を見つけます。検出結果がある場合は、検出結果の分類と詳細を表示するリンクが表示されます。ない場合は、「このインスタンスで利用可能な推奨事項はありません (No recommendations available for this instance)」と表示されます。
5. 検出結果がある場合は、[詳細を表示] を選択します。Compute Optimizer コンソールに [EC2 インスタンスの推奨事項] ページが開きます。現在のインスタンスタイプには [現在] というラベルが付けられます。最大 3 つのインスタンスタイプが [オプション 1]、[オプション 2]、[オプション 3] というラベル付きで推奨されます。このページには、インスタンスの最近の CloudWatch メトリクスデータも表示されます。

すべてのリージョンのすべてのインスタンスに関する推奨事項を表示するには

Compute Optimizer コンソールを使用して、すべてのリージョンのすべての Amazon EC2 インスタンスの推奨事項を表示できます。詳細については、「AWS Compute Optimizer ユーザーガイド」の「EC2 インスタンスレコメンデーションの表示」と「EC2 インスタンスの詳細の表示」を参照してください。

推奨事項の評価に関する考慮事項

推奨事項を受け取ったら、それに対応するかどうかを決定する必要があります。インスタンスタイプを変更する前に、次の点を考慮してください。

- 推奨情報は使用状況を予測するものではありません。推奨事項は、直近の 14 日間の使用履歴に基づいています。将来のリソースニーズを満たすことが予想されるインスタンスタイプを必ず選択します。

- グラフ化されたメトリクスを参考にして、実際の使用量がインスタンスの容量よりも低いかどうかを判断します。メトリクスデータ (平均、ピーク、パーセンタイル) を CloudWatch で表示し、EC2 インスタンスの推奨事項をさらに評価することもできます。例えば、一日の CPU パーセンテージメトリクスがどのように変化するか、ピークに対応する必要があるかどうかに注目します。詳細については、Amazon CloudWatch ユーザーガイドの「使用可能なメトリクスの表示」を参照してください。
- Compute Optimizer は、バーストパフォーマンスインスタンス (T3、T3a、および T2 インスタンス) の推奨事項を提供する場合があります。定期的にベースラインを超えてバーストする場合は、新しいインスタンスタイプの vCPU に基づいて引き続きバーストを実行できることを確認します。詳細については、バーストパフォーマンスインスタンスに関する主要な概念 を参照してください。
- リザーブドインスタンスを購入した場合、オンデマンドインスタンスはリザーブドインスタンスとして請求される場合があります。現在のインスタンスタイプを変更する前に、まずリザーブドインスタンスの使用率と適用範囲に対する影響を評価します。
- 可能であれば、新世代のインスタンスへの交換を検討します。
- 別のインスタンスファミリーに移行する場合は、仮想化、アーキテクチャー、ネットワークタイプなどの点で、現在のインスタンスタイプと新しいインスタンスタイプに互換性があることを確認してください。詳細については、インスタンスタイプ変更の互換性 を参照してください。
- 最後に、推奨事項ごとに提供されるパフォーマンスリスク評価を検討します。パフォーマンスリスクは、推奨されるインスタンスタイプがワークロードのパフォーマンス要件を満たすかどうかを検証するために費やす必要のある作業量を示します。また、変更前と変更後に厳格な負荷テストおよびパフォーマンステストを行うことをお勧めします。

Amazon EC2 インスタンスタイプの変更

ニーズが変わるにつれて、インスタンスの利用率が高すぎたり (インスタンスタイプが小さすぎる)、低すぎたりする (インスタンスタイプが大きすぎる) ことに気付く場合があります。この場合は、インスタンスタイプを変更することでインスタンスのサイズを変更できます。例えば、ワークロードに対して `t2.micro` インスタンスが小さすぎる場合は、`t2.large` などのより大きな T2 インスタンスタイプに変更することで、サイズを大きくすることができます。また、`m5.large` などの別のインスタンスタイプに変更することもできます。また、IPv6 への対応など、いくつかの機能を利用するために、旧世代のインスタンスタイプから現世代のインスタンスタイプに変更することもできます。

既存のワークロードを処理するのに最適なインスタンスタイプにレコメンデーションが必要な場合は、AWS Compute Optimizer を使用することができます。詳細については、「Compute Optimizer から EC2 インスタンスの推奨事項を取得する」を参照してください。

インスタンスタイプを変更する場合、新しいインスタンスタイプのレートの課金が開始されます。すべてのインスタンスタイプのオンデマンド料金については、「Amazon EC2 オンデマンド料金」を参照してください。

インスタンスタイプを変更せずにインスタンスにストレージを追加するには、EBS ボリュームをインスタンスに追加します。詳細については、「Amazon EBS ユーザーガイド」の「インスタンスへの Amazon EBS ボリュームのアタッチ」を参照してください。

どの手順に従うべきですか？

インスタンスタイプを変更するには、さまざまな手順があります。使用する手順は、インスタンスのルートボリューム、およびインスタントタイプがインスタンスの現在の設定と互換性があるかどうかによって異なります。互換性が決定される方法については、「インスタンスタイプ変更の互換性」を参照してください。

次の表を使用して、従う手順を決定します。

ルートボリューム	互換性	以下の手順に従います
EBS	互換性あり	インスタンスタイプを変更する
EBS	互換性なし	新しいインスタンスタイプに移行する
インスタンスストア	該当しない	新しいインスタンスタイプに移行する

インスタンスタイプ変更の互換性

インスタンスの現在の設定が目的のインスタンスタイプと互換性がある場合にのみ、インスタンスタイプを変更できます。使用するインスタンスタイプに既存のインスタンス設定との互換性がない場合、インスタンスタイプに対応した設定を持つ新しいインスタンスを起動して、アプリケーションを新しいインスタンスに移行する必要があります。

互換性は、次のように決定されます。

仮想化タイプ

Linux AMI では、2 つの仮想化タイプ (準仮想化 (PV) およびハードウェア仮想マシン (HVM)) のどちらかを使用します。PV AMI から起動したインスタンスの場合、HVM のみのインスタンスタイプに変更することはできません。詳細については、仮想化タイプを参照してください。インスタンスの仮想化タイプを確認するには、Amazon EC2 コンソールで [Instances] (インスタンス) 画面の詳細ペインの [Virtualization] (仮想化) の値を参照します。

アーキテクチャ

AMI はプロセッサのアーキテクチャに固有であるため、プロセッサアーキテクチャーが現在のインスタンスタイプと同じインスタンスタイプを選択する必要があります。例:

- 現在のインスタンスタイプが、Arm アーキテクチャに基づくプロセッサである場合、対象となるインスタンスタイプは、Arm アーキテクチャベースのプロセッサ (C6g や M6g など) をサポートするものに制限されます。
- 32 ビット AMI をサポートするのは以下のインスタンスタイプのみです。`t2.nano`、`t2.micro`、`t2.small`、`t2.medium`、`c3.large`、`t1.micro`、`m1.small`、`m1.me` および `c1.medium`。32 ビットインスタンスのインスタンスタイプを変更する場合は、これらのインスタンスタイプに制限されます。

ネットワークアダプター

ドライバーのネットワークアダプターを別のネットワークアダプターに切り替えると、オペレーティングシステムが新しいアダプターを作成したときに、ネットワークアダプターの設定がリセットされます。設定を再構成するには、管理者権限を持つローカルアカウントへのアクセスが必要な場合があります。ネットワークアダプターを別のネットワークアダプターに切り替える例を次に示します。

- AWS PV (T2 インスタンス) からインテル 82599 VF (M4 インスタンス)
- インテル 82599 VF (ほとんどの M4 インスタンス) から ENA (M5 インスタンス)
- ENA (M5インスタンス) から高帯域幅の ENA (M5nインスタンス)

拡張ネットワーク

拡張ネットワークをサポートするインスタンスタイプでは、必要なドライバーがインストールされていなければなりません。例えば、Nitro ベースのインスタンスには Elastic Network Adapter (ENA) ドライバーがインストールされた EBS-backed AMI が必要です。拡張ネットワーキングをサポートしていないインスタンスタイプから、拡張ネットワークをサポートするインスタンスタイプに変更するには、インスタンスに ENA ドライバーまたは ixgbevf ドライバーを適切にインストールする必要があります。

> **Note**
>
> ENA Express を有効にしてインスタンスのサイズを変更する場合、新しいインスタンスタイプも ENA Express をサポートしている必要があります。ENA Express をサポートしているインスタンスタイプのリストは、「ENA Express でサポートされるインスタンスタイプ」を参照してください。
> ENA Express をサポートするインスタンスタイプから ENA Express をサポートしないインスタンスタイプに変更するには、インスタンスをサイズ変更する前に、ENA Express が現在有効になっていないことを確認します。

NVMe

Nitro ベースのインスタンスでは、EBS ボリュームは NVMe ブロックデバイスとして公開されます。NVMe をサポートしないインスタンスタイプから NVMe をサポートするインスタンスタイプに変更するには、まずインスタンスに NVMe ドライバーをインストールする必要があります。また、ブロックデバイスマッピングで指定したデバイスのデバイス名は、NVMe デバイス名 (`/dev/nvme[0-26]n1`) を使用して変更されます。

[Linux インスタンス] したがって、`/etc/fstab` を使用してブート時にファイルシステムをマウントするには、デバイス名の代わりに UUID/Label を使用する必要があります。

ボリュームの制限

インスタンスにアタッチできる Amazon EBS ボリュームの最大数はインスタンスのタイプとサイズによって異なります。詳細については、「Amazon EC2 インスタンスの Amazon EBS ボリューム制限」を参照してください。

インスタンスタイプまたはインスタンスサイズに変更できるのは、現在インスタンスにアタッチされているボリュームと同じ数またはそれ以上のボリュームをサポートするものに限られます。現在アタッチされているボリュームの数をサポートしていないインスタンスタイプまたはインスタンスサイズに変更すると、リクエストは失敗します。例えば、32 個のボリュームが接続されている `m7i.4xlarge` インスタンスから、最大 27 個のボリュームをサポートする `m6i.4xlarge` インスタンスに変更すると、リクエストは失敗します。

NitroTPM

NitroTPM が有効な AMI と NitroTPM をサポートするインスタンスタイプを使用してインスタンスを起動した場合、インスタンスは NitroTPM が有効な状態で起動します。同じく NitroTPM をサポートするインスタンスタイプにのみ変更できます。

Amazon EC2 インスタンスのインスタンスタイプを変更する

必要なインスタンスタイプがインスタンスの現在の設定と互換性がある場合は、次の手順を使用して Amazon EBS-backed インスタンスのインスタンスタイプを変更します。詳細については、「the section called “互換性”」を参照してください。

考慮事項

- インスタンスタイプを変更するには、そのインスタンスを停止する必要があります。インスタンスが停止している間のダウンタイムを予定しておいてください。インスタンスを停止し、インスタンスタイプの変更を行うと、数分かかります。インスタンスを再起動すると、アプリケーションの起動スクリプトによってかかる時間が変動する場合があります。詳細については、Amazon EC2 インスタンスの停止と開始 を参照してください。
- インスタンスを停止して起動すると、インスタンスは新しいハードウェアに移動されます。インスタンスに Elastic IP ではないパブリック IPv4 アドレスがある場合には、このアドレスはリリースされて、インスタンスは新しいパブリック IPv4 アドレスになります。インスタンスのライフサイクル全体における IP アドレスの動作の詳細については、「インスタンスの状態の違い」を参照してください。
- [Spot Instance] (スポットインスタンス) のインスタンスタイプを変更することはできません。
- [Windows インスタンス インスタンスタイプを変更する前に、AWS PV ドライバーパッケージを更新することをお勧めします。詳細については、「the section called “PV ドライバーのアップグレード”」を参照してください。
- インスタンスが Auto Scaling グループにある場合、Amazon EC2 Auto Scaling サービスはインスタンスを異常と判断して停止し、場合によってはそれを終了して代替インスタンスを起動します。インスタンスタイプを変更するときに、そのグループのスケーリングプロセスを中断することで、これを防ぐことができます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「スケーリングプロセスの中断と再開」を参照してください。
- NVMe インスタンスストアボリュームを使用してインスタンスのインスタンスタイプを変更すると、AMI またはインスタンスブロックデバイスマッピングで指定されていない場合でもすべての NVMe インスタンスストアボリュームが利用可能であるため、追加のインスタンスストアボリュームが更新されたインスタンスに存在するようになる可能性があります。それ以外の場合、更新したインスタンスには、元のインスタンスの起動時に指定したのと同じ数のインスタンスストアボリュームが設定されます。
- インスタンスにアタッチできる Amazon EBS ボリュームの最大数は、インスタンスのタイプとサイズによって異なります。インスタンスにすでにアタッチされているボリュームの数をサポートし

ていないインスタンスタイプまたはインスタンスサイズに変更することはできません。詳細については、「Amazon EC2 インスタンスの Amazon EBS ボリューム制限」を参照してください。

- [Linux インスタンス] `AWSSupport-MigrateXenToNitroLinux` ランブックを使用して、互換性のあるインスタンスを Xen インスタンスタイプから Nitro インスタンスタイプに移行できます。詳細については「AWS Systems Manager 自動 ランブックリファレンス」の AWSSupport-MigrateXenToNitroLinux runbook を参照してください。
- [Windows インスタンス] 互換性のある Windows インスタンスを Xen インスタンスタイプから Nitro インスタンスタイプに移行する方法に関するその他のガイダンスについては、「最新世代のインスタンスタイプへの移行」を参照してください。

Amazon EBS-backed インスタンスのインスタンスタイプを変更するには

1. (オプション) 新しいインスタンスのタイプに既存のインスタンスにインストールされていないドライバーが必要な場合は、インスタンスに接続してドライバーをインストールする必要があります。詳細については、「インスタンスタイプ変更の互換性」を参照してください。
2. [Windows インスタンス] 静的 IP アドレス指定を使用するように Windows インスタンスを設定し、拡張ネットワーキングをサポートしないインスタンスタイプから拡張ネットワーキングをサポートするインスタンスタイプに変更する場合、静的 IP アドレス指定を再設定すると IP アドレスが競合する可能性について警告が表示される場合があります。これを防ぐには、インスタンスタイプを変更する前に、インスタンスのネットワークインターフェイスで DHCP を有効にします。インスタンスから、[Network and Sharing Center] (ネットワークと共有センター) を開き、ネットワークインターフェイスの [Internet Protocol Version 4 (TCP/IPv4) Properties] (インターネットプロトコルバージョン 4 (TCP/IPv4) のプロパティ) を開いて、[Obtain an IP address automatically] (IP アドレスを自動的に取得する) を選択します。インスタンスタイプを変更し、ネットワークインターフェイス上で静的 IP アドレッシングを設定します。
3. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
4. ナビゲーションペインで、[インスタンス] を選択してください。
5. インスタンスを選択し、[Instance state (インスタンスの状態)]、[Stop instance (インスタンスの停止)] の順に選択してください。確認を求められたら、[Stop] を選択してください。インスタンスが停止するまで、数分かかる場合があります。
6. インスタンスが選択された状態で、[Actions (アクション)]、[Instance settings (インスタンス設定)]、[Change instance type (インスタンスタイプの変更)] の順に選択します。状態が `stopped` ではないインスタンスの場合、このオプションはグレー表示されています。
7. [Change Instance Type] (インスタンスタイプの変更) で、次のいずれかの操作を行います。

a. [Instance type] (インスタンスタイプ) で、使用するインスタンスタイプを選択します。

   インスタンスタイプがリストにない場合、インスタンスタイプはインスタンスの設定と互換性がありません。代わりに、新しい EC2 インスタンスを起動して新しいインスタンスタイプに移行する の手順をすべて行います。

b. (オプション) 選択したインスタンスタイプが EBS 最適化をサポートしている場合は、[EBS-optimized] (EBS 最適化) を選択して EBS 最適化を有効にするか、[EBS-optimized] (EBS 最適化) を選択解除して EBS 最適化を無効にします。

   選択したインスタンスタイプがデフォルトで EBS に最適化されている場合、[EBS-Optimized] (EBS 最適化) は選択状態になっており、これを選択解除することはできません。

c. (オプション) 新しいインスタンスタイプで vCPU オプションを設定します。

   既存のインスタンスのインスタンスタイプを変更すると、Amazon EC2 は、可能であれば既存のインスタンスから新しいインスタンスに CPU オプション設定を適用します。新しいインスタンスタイプがこれらの設定をサポートしていない場合、CPU オプションは なし にリセットされます。このオプションでは、新しいインスタンスタイプのデフォルトの vCPU 数が使用されます。

   選択したインスタンスタイプが vCPU 設定をサポートしている場合は、[詳細] パネルで [CPU オプションを指定] を選択して、新しいインスタンスタイプの vCPU を設定します。

d. [変更] を選択して、新しい設定を受け入れます。

8. インスタンスをスタートするには、インスタンスを選択後、[Instance state] (インスタンスの状態)、[Start instance] (インスタンスの開始) の順にクリックします。インスタンスが `running` 状態になるまで、数分かかる場合があります。インスタンスが起動しない場合は、「インスタンスタイプ変更のトラブルシューティング」を参照してください。

9. [Windows インスタンス] インスタンスが EC2Launch v1 を使用した Windows Server 2016 または Windows Server 2019 を実行する場合は、Windows インスタンスに接続し、インスタンスタイプが変更された後に次の EC2Launch PowerShell スクリプトを実行してインスタンスを設定します。

> **Important**
>
> 管理者パスワードはインスタンス初期化 EC2 Launch スクリプトを有効にするとリセットされます。初期化タスクの設定で指定することで、管理者パスワードのリセットを無効にするように設定ファイルを変更できます。パスワードリセットを無効にす

るステップについては、「初期化タスクの設定」(EC2Launch) または「設定の変更」(EC2Launch v2) を参照してください。

```
PS C:\> C:\ProgramData\Amazon\EC2-Windows\Launch\Scripts\InitializeInstance.ps1 -
Schedule
```

新しい EC2 インスタンスを起動して新しいインスタンスタイプに移行する

EC2 インスタンスのインスタンスタイプを変更できるのは、そのインスタンスが、目的の新しいインスタンスタイプとの互換性がある設定を持つ EBS-backed インスタンスである場合のみです。それ以外の場合、設定またはインスタンスが新しいインスタンスタイプと互換性がない場合、またはインスタンスストアベースのインスタンスである場合は、必要なインスタンスタイプと互換性のある代替インスタンスを起動する必要があります。互換性が決定される方法の詳細については、「インスタンスタイプ変更の互換性」を参照してください。

データ移行プロセスの概要

- 元のインスタンスのデータをバックアップします。
- 使用する新しいインスタンスタイプと互換性がある設定で新しいインスタンスを起動して、元のインスタンスにアタッチされた EBS ボリュームをアタッチします。
- アプリケーションを新しいインスタンスにインストールします。
- データを復元します。
- 元のインスタンスに Elastic IP アドレスがある場合は、ユーザーが中断することなくアプリケーションを使用し続けることができるように、Elastic IP アドレスを新しいインスタンスに関連付ける必要があります。

インスタンスを新しいインスタンスに移行するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 次のように、まだ必要なデータをバックアップします。
   - インスタンスに接続し、インスタンスストアボリュームのデータを永続ストレージにコピーします。

- EBS ボリュームのスナップショットを作成して、同じデータを持つ新しいボリュームを作成したり、ボリュームを元のインスタンスからデタッチして新しいインスタンスにアタッチしたりできます。

3. ナビゲーションペインで、[インスタンス] を選択してください。
4. [Launch Instances] (インスタンスの起動) を選択します。インスタンスを設定する場合は、次の操作を行います。

   a. 使用するインスタンスタイプをサポートする AMI を選択します。例えば、新しいインスタンスタイプのプロセッサタイプをサポートする AMI を選択する必要があります。また、現行世代のインスタンスタイプには HVM AMI が必要です。
   b. 起動する新しいインスタンスタイプを選択します。使用するインスタンスタイプが使用できない場合は、選択した AMI の構成と互換性がありません。
   c. 同じトラフィックが新しいインスタンスに到達できるようにする場合は、元のインスタンスで使用されているものと同じ VPC とセキュリティグループを選択します。
   d. 新しいインスタンスの設定が完了したら、手順を実行してキーペアを選択し、インスタンスを起動します。インスタンスが `running` 状態になるまで、数分かかる場合があります。
5. EBS スナップショットにデータをバックアップした場合は、スナップショットからボリュームを作成し、新しいインスタンスにそのボリュームをアタッチします。

   EBS ボリュームを元のインスタンスから新しいインスタンスに移動するには、元のインスタンスからボリュームをデタッチし、新しいインスタンスにそのボリュームをアタッチします。
6. アプリケーションと必要なソフトウェアを新しいインスタンスにインストールします。
7. 元のインスタンスのインスタンスストアボリュームからバックアップしたデータを復元します。
8. 元のインスタンスに Elastic IP アドレスがある場合は、次のようにそれを新しいインスタンスに割り当てます。

   a. ナビゲーションペインで [Elastic IP] を選択します。
   b. 元のインスタンスに関連付ける Elastic IP アドレスを選択して、[Actions (アクション)]、[Disassociate Elastic IP address (Elastic IP アドレスの関連付けの解除)] の順に選択します。確認を求めるメッセージが表示されたら、[Disassociate (関連付け解除)] を選択します。
   c. Elastic IP アドレスがまだ選択された状態で、[Actions (アクション)]、[Associate Elastic IP address (Elastic IP アドレスの関連付け)] の順に選択します。
   d. [リソースタイプ] で、[Instance (インスタンス)] を選択します。
   e. [インスタンス] で、新しいインスタンスを選択します。

f. (オプション) [プライベート IP アドレス] で、Elastic IP アドレスを関連付けるプライベート IP アドレスを指定します。

g. [Associate] を選択します。

9. (オプション) 不要になった場合は、元のインスタンスを終了できます。インスタンスを選択し、新しいインスタンスではなく元のインスタンスを終了させようとしていることを確認し (名前や起動時間の確認など)、[Instance state] (インスタンスの状態)、[Terminate instance] (インスタンスを終了) を選択します。

インスタンスタイプ変更のトラブルシューティング

以下の情報は、インスタンスタイプの変更時に発生する可能性のある問題の診断と修復に役立ちます。

インスタンスタイプを変更してもインスタンスが起動しない

考えられる原因: 新しいインスタンスタイプの要件が満たされていない

インスタンスが起動しない場合、新しいインスタンスタイプの要件の 1 つが満たされていない可能性があります。詳細については、タイプを変更した後、Linuxインスタンスが起動しなくなったのはなぜですか?を参照してください。

考えられる原因: AMI がインスタンスタイプをサポートしていない

EC2 コンソールを使用してインスタンスタイプを変更する場合、選択した AMI でサポートされているインスタンスタイプのみを使用できます。しかし、AWS CLI を使ってインスタンスを起動すると、互換性のない AMI とインスタンスタイプを指定してしまうことがあります。AMI とインスタンスタイプに互換性がない場合、インスタンスは起動できません。詳細については、インスタンスタイプ変更の互換性を参照してください。

考えられる原因: インスタンスがクラスタープレイスメントグループに属している

インスタンスがクラスタープレイスメントグループに属しており、インスタンスタイプを変更した後にインスタンスの起動に失敗した場合、以下を試してください。

1. クラスタープレイスメントグループ内のすべてのインスタンスを停止します。
2. 影響を受けるインスタンスのインスタンスタイプを変更します。
3. クラスタープレイスメントグループ内のすべてのインスタンスを起動します。

インスタンスタイプを変更した後、アプリケーションからアプリケーションまたは Web サイトにアクセスできない

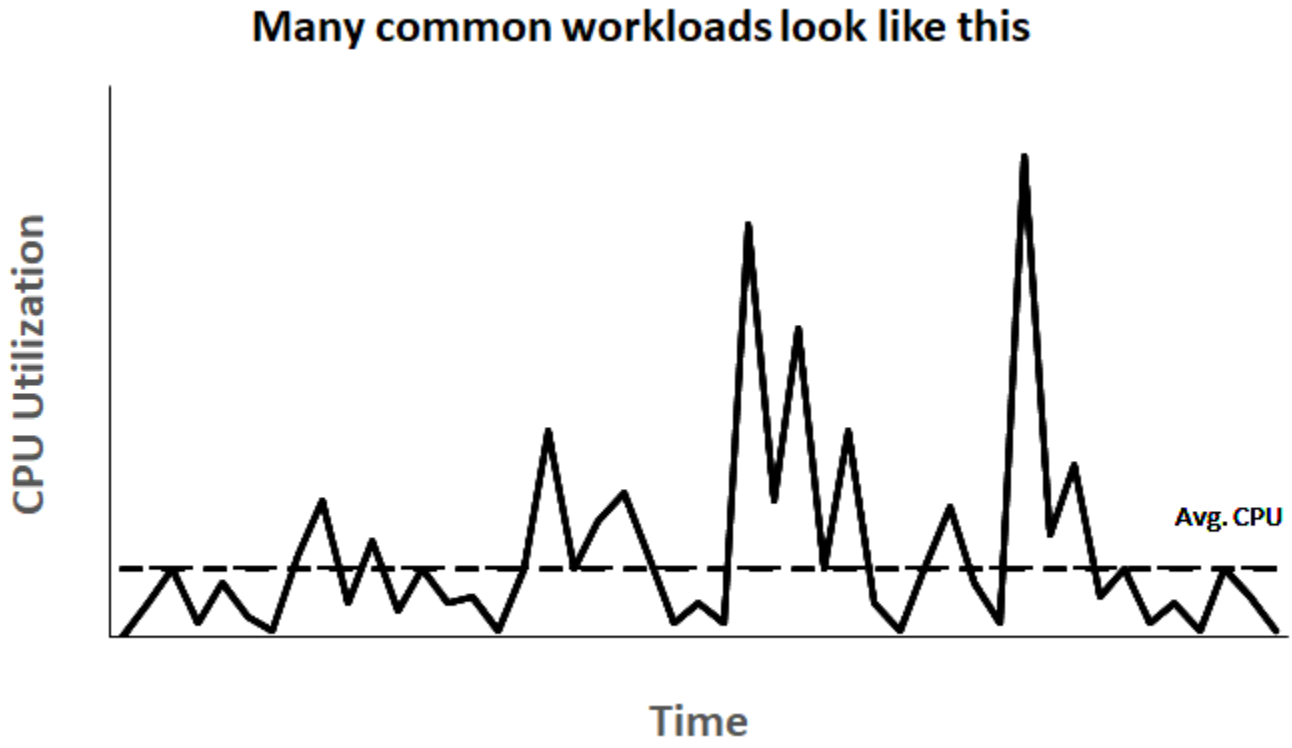
考えられる原因: パブリック IPv4 アドレスが解放されている

インスタンスタイプを変更する場合は、まずインスタンスを停止する必要があります。ユーザーがインスタンスを停止すると、パブリック IPv4 アドレスを解放し、お客様のインスタンスに新しいパブリック IPv4 アドレスを付与します。

インスタンスの停止と開始の間にパブリック IPv4 アドレスを保持するには、Elastic IP アドレスを使用することをお勧めします。詳細については、「Elastic IP アドレス」を参照してください。

バーストパフォーマンスインスタンス

多くの汎用ワークロードでは、平均的な状態はビジーではないので、持続的で高いレベルの CPU パフォーマンスを必要としません。以下のグラフは、現在 AWS クラウドのお客様が実行している多くの一般的なワークロードにおける、CPU の使用率を示しています。



これらの CPU 使用率が低 ~ 中程度のワークロードが、CPU サイクルを浪費するような場合には、使用量以上の料金が発生することになります。こういった問題に対処するには、低コストのバースト汎用インスタンス (T インスタンス) を活用します。

T インスタンスファミリーは、ベースラインの CPU パフォーマンスを提供しながら、いつでも必要な時間だけ、能力をベースライン以上にバーストさせる機能を備えています。ベースライン CPU は汎用ワークロードにおける大部分のニーズを満たすように構成されており、大規模なマイクロサービス、ウェブサーバー、中小規模のデータベース、データのログ記録、コードリポジトリ、仮想デスクトップ、開発/テスト環境、ビジネスクリティカルなアプリケーションなどに対応できます。T インスタンスでは、コンピューティング、メモリ、ネットワークリソースがバランスしているので、CPU 使用率が低～中程度の広範な汎用アプリケーションを実行するための、費用対効果が最も高い方法が提供されます。コストの面では、M インスタンスと比較しても最大 15% の節約となります。また、2 個の vCPU と 0.5 GiB のメモリで動作する小型かつ経済的なインスタンスサイズを選択すれば、さらにコストの削減が可能です。ナノ、マイクロ、Small、中 など小型の T インスタンスサイズは、必要なメモリが少なく、また想定される CPU 使用率も高くないワークロードに適しています。

> **Note**
>
> このトピックは、バースト可能な CPU について説明します。バースト可能なネットワークパフォーマンスの詳細については、「Amazon EC2 インスタンスのネットワーク帯域幅」を参照してください。

EC2 バーストインスタンスタイプ

EC2 バーストインスタンスには、T4g、T3a、T3 インスタンスタイプ、および旧世代の T2 インスタンスタイプが含まれます。

T4g インスタンスタイプは、最新世代のバーストインスタンスです。このタイプは、パフォーマンスに対して最良の価格設定が行われており、すべての EC2 インスタンスタイプ中でも、最も低いコストでの利用が可能です。T4g インスタンスタイプは、ARM ベースの AWSGraviton2 プロセッサーで動作します。また、オペレーティングシステムベンダー、独立系ソフトウェアベンダー、さらに一般的な AWS サービスとアプリケーションからの、広範なエコシステムサポートも利用できます。

次の表に、各バーストインスタンスタイプ間の主な違いをまとめます。

タイプ	説明	プロセッサファミリ
最新世代		
T4g	最低コストの EC2 インスタンスタイプで、T3 と比較した場	Arm Neoverse N1 コア搭載の AWS Graviton2 プロセッサ

タイプ	説明	プロセッサファミリ
	合コストパフォーマンスは最大 40% 向上し、使用料金は 20% 低減	
T3a	T3 インスタンスと比較してコストを 10% 低減する、最低料金の x86 ベースインスタンス	AMD 第1世代 EPYC プロセッサー
T3	x86 ワークロードでピーク時に最良のコストパフォーマンスを発揮、旧世代の T2 インスタンスと比較した場合コストパフォーマンスが最大 30% 向上	インテル Xeon スケーラブル (Skylake、Cascade Lake プロセッサー)
前の世代		
T2	旧世代のバーストインスタンス	インテル Xeon プロセッサ

インスタンスの料金体系と、その他の仕様については、「Amazon EC2 の料金」および「Amazon EC2 インスタンスタイプ」を参照してください。バースト可能なネットワークパフォーマンスの詳細については、「Amazon EC2 インスタンスのネットワーク帯域幅」を参照してください。

2025 年 7 月 15 日より前に AWS アカウントを作成していて、12 か月が経過していない場合は、特定の使用制限内で `t2.micro` インスタンスを無料で使用できます (`t2.micro` を利用できないリージョンでは `t3.micro` インスタンスを使用できます)。2025 年 7 月 15 日以降に AWS アカウントを作成した場合は、`t3.micro`、`t3.small`、`t4g.micro`、`t4g.small` インスタンスタイプを 6 か月間、またはクレジットを使い切るまで使用できます。詳細については、「AWS 無料利用枠」を参照してください。

T インスタンスでサポートされる購入オプション

- On-Demand Instances
- Reserved Instances
- ハードウェア専有インスタンス (T3 のみ)

- Dedicated Hosts (T3 のみ、standardモードのみ)
- スポットインスタンス

詳細については「Amazon EC2 の請求および購入オプション」を参照してください。

目次

- ベストプラクティス
- バーストパフォーマンスインスタンスに関する主要な概念
- バーストパフォーマンスインスタンスの Unlimited モード
- バーストパフォーマンスインスタンスのスタンダードモード
- バーストパフォーマンスインスタンスを設定する
- バーストインスタンスの CPU クレジットをモニタリングする

ベストプラクティス

これらのベストプラクティスに従って、バーストパフォーマンスインスタンスの利点を最大限に活用してください。

- 選択するインスタンスのサイズが、オペレーティングシステムおよびアプリケーションの最小メモリ要件を満たしていることを確認します。多量のメモリおよび CPU リソースを消費するグラフィカルユーザーインターフェースを使用するオペレーティングシステム (Windows など) は、多くのユースケースで t3.micro 以上のインスタンスサイズを必要とする場合があります。時間の経過とともに、メモリおよび CPU に対するワークロードからの要求が増大した場合のために、T インスタンスには、同じインスタンスタイプで大きなインスタンスサイズにスケールできる柔軟性が備わっています。あるいは、別のインスタンスタイプを選択することも可能です。
- アカウントの AWS Compute Optimizer を有効にし、ワークロードに関する Compute Optimizer 推奨事項を確認します。Compute Optimizer は、パフォーマンスを向上させるためにインスタンスをアップサイズする必要があるかや、コスト削減のためにダウンサイズする必要があるかを評価する際に役立ちます。Compute Optimizer は、シナリオに応じて異なるインスタンスタイプを推奨する場合もあります。詳細については、「AWS Compute Optimizer ユーザーガイド」の「EC2 インスタンスのレコメンデーションの表示」を参照してください。

バーストパフォーマンスインスタンスに関する主要な概念

従来の Amazon EC2 インスタンスタイプの場合は CPU リソースが固定されています。一方バーストパフォーマンスインスタンスの場合は、CPU 使用率にベースラインレベルを定義した上で、そのレベルを超えて CPU 使用率をバーストさせることが可能となっています。これにより料金は、ベースラインの CPU 使用率に加えて、バーストとして追加された分に対してのみ支払えば良いことになり、コンピューティングのコストを削減できます。ベースライン使用率とバースト機能は、CPU クレジットで管理します。バーストパフォーマンスインスタンスは、CPU 使用率にクレジットを使用する唯一のインスタンスタイプです。

各バーストパフォーマンスインスタンスは、CPU 使用率がベースラインを下回っている間は継続的にクレジットを獲得し、ベースラインを上回っている間は継続的にクレジットを消費します。獲得または消費されたクレジットの量は、インスタンスの CPU 使用率によって異なります。

- CPU 使用率がベースラインを下回っている場合、獲得するクレジットは消費するクレジットよりも大きくなります。
- CPU 使用率がベースラインと等しい場合、獲得するクレジットは消費するクレジットと等しくなります。
- CPU 使用率がベースラインよりも高い場合、消費するクレジットが獲得するクレジットよりも高くなります。

獲得したクレジットが消費したクレジットよりも大きい場合、その差額は蓄積されたクレジットと呼ばれ、後でベースラインを超えて CPU 使用率をバーストさせる際に使用できます。一方、消費したクレジットが獲得したクレジットよりも多い場合のインスタンスの動作は、クレジット設定モード (スタンダードモードまたは Unlimited モード) によって異なります。

スタンダードモードでは、消費したクレジットが獲得したクレジットよりも多い場合、インスタンスは、ベースラインを越えて CPU 使用率をバーストさせるために、蓄積されたクレジットを使用します。蓄積されたクレジットに残額がない場合、インスタンスは CPU 使用率を徐々にベースラインまで低下させ、より多くのクレジットが蓄積されるまで、ベースラインを超えてバーストすることはできなくなります。

Unlimited モードでは、CPU 使用率がベースラインを超えてバーストした場合、インスタンスは最初に蓄積されたクレジットを使用します。その後、蓄積されたクレジットの残額がなくなった場合には、インスタンスは余剰クレジットを消費してバーストを維持します。その CPU 利用率がベースラインを下回った場合、獲得した CPU クレジットを使用して、先に消費された余剰クレジットの支払いが行われます。CPU クレジットを獲得して余剰クレジットを支払う機能により、Amazon EC2 は

24 時間にわたるインスタンスの CPU 使用率を平均化できるようになります。24 時間の平均 CPU 使用率がベースラインを超えたインスタンスには、vCPU 時間あたりの超過の使用量に対して、均一追加料金が発生します。

内容

- 主要な概念と定義
- CPU クレジットの獲得
- CPU クレジットの獲得率
- CPU クレジット蓄積制限
- CPU 存続期間の蓄積
- ベースライン使用率

主要な概念と定義

バーストパフォーマンスインスタンスには、以下の主要な概念と定義が適用されます。

CPU 使用率

CPU 使用率とは、割り当てられた EC2 コンピューティングユニットのうち、現在インスタンス上で使用されているものが占める割合のことです。このメトリクスは、割り当てられた CPU サイクルの中で、インスタンスで使用されているサイクルの割合を測定します。CloudWatch の CPU 使用率に関するメトリクスでは、コアごとの CPU 使用率ではなく、インスタンスごとの CPU 使用率を示しています。インスタンスの CPU ベースラインに関する仕様は、インスタンスごとの CPU 使用率とも関連しています。AWS マネジメントコンソール または AWS CLI を使用して CPU 使用率を測定する方法については、特定のインスタンスの統計を取得するを参照してください。

CPU クレジット

vCPU 時間の単位。

例:

1 CPU クレジット = 1 vCPU × 100% 使用率 × 1 分

1 CPU クレジット = 1 vCPU × 50% 使用率 × 2 分

1 CPU クレジット = 2 vCPU × 25% 使用率 × 2 分

ベースライン使用率

ベースライン使用率とは、CPU クレジットの獲得数と CPU クレジットの使用数が一致する場合に、正味のクレジット残高が 0 の状態で CPU を使用できるレベルのことです。ベースライン使用率はベースラインとも呼ばれます。ベースライン使用率は vCPU の使用率のパーセンテージとして表され、次のように計算されます。ベースライン使用率 % = (獲得したクレジットの数 ÷ vCPU の数) ÷ 60 分

各バーストパフォーマンスインスタンスタイプのベースライン使用率については、「クレジットの表」を参照してください。

獲得クレジット

実行中のインスタンスが継続的に獲得するクレジットです。

1 時間あたりの獲得クレジット数 =ベースライン使用率 (%) × vCPU 数 (個)× 60 (分)

例:

vCPU を 2 個使用し、ベースライン使用率が 5% に設定された t3.nano では次の計算のように、1 時間あたり 6 クレジットを獲得します。

2 個の vCPU × 5% のベースライン × 60 分 = 1 時間あたり 6 クレジット

消費または使用されたクレジット

実行中のインスタンスにより継続的に使用されるクレジットです。

1 分あたりに使用される CPU クレジット = vCPU 数 (個) × CPU 使用率 (%) × 1 分

蓄積されたクレジット

インスタンスの使用量がベースラインの使用率よりも少ないので、消費されなかった CPU クレジットです。つまり、蓄積されたクレジット = 獲得クレジット — 使用されたクレジット (ともにベースラインより低い場合)、となります。

例:

仮に t3.nano の CPU 使用率が 2% で、ベースラインである 5% を 1 時間の間下回っていた場合、蓄積されたクレジットは次のように計算されます。

蓄積された CPU クレジット = (1 時間あたりの獲得クレジット — 1 時間あたりの使用クレジット) = 6 — 2 (vCPU 個数) × 2 (CPU 使用率 %) × 60 (分) = 6 — 2.4 = 3.6 (1 時間あたりに蓄積されたクレジット)

クレジット蓄積制限

インスタンスのサイズによって異なりますが、通常は 24 時間以内に獲得できるクレジットの最大数と等しくなります。

例:

t3.nanoの場合、クレジット蓄積制限 = 24 × 6 = 144クレジット

起動クレジット

これは、スタンダードモードに設定された T2 インスタンスにのみ適用されます。起動クレジットは、新しい T2 インスタンスに割り当てられるもので、CPU クレジットの数に制限があります。スタンダードモードで起動することで、ベースラインを超えたバーストが可能になります。

余剰クレジット

蓄積されたクレジットの残高が枯渇したインスタンスが消費するクレジットです。余剰クレジットは、長期間高パフォーマンスを維持するバーストインスタンスのために設計されており、使用できるのは Unlimited モードでのみです。余剰クレジット残高は、Unlimited モードのインスタンスがバーストのために使用した、クレジットの数を判断するために使用されます。

スタンダードモード

クレジットの設定モードです。このモードのインスタンスでは、クレジット残高に蓄積されたクレジットを消費することで、そのベースラインを超えたバーストが可能です。

Unlimited モード

クレジットの設定モードです。必要な期間にわたって高い CPU 使用率を維持することで、インスタンスがベースラインを超えてバーストすることを可能にします。24 時間ごとのインスタンスの平均 CPU 使用率またはインスタンスの存続期間のいずれか短い方の時間で、インスタンスの平均 CPU 使用率がベースライン以下になった場合、1 時間ごとのインスタンス価格は自動的にすべての CPU 使用率スパイクをカバーします。長時間にわたって高い CPU 使用率でインスタンスを実行する場合には、vCPU 時間ごとに均一追加料金が発生します。

次の表は、バーストインスタンスタイプ間の主なクレジットの違いをまとめたものです。

タイプ	サポートされる CPU クレジットのタイプ	クレジットの設定モード	インスタンスの開始から停止までの間に蓄積された CPU クレジットのライフスパン
最新世代			
T4g	獲得クレジット、蓄積されたクレジット、消費されたクレジット、余剰クレジット (Unlimited モードのみ)	スタンダード、Unlimited (デフォルト)	7 日間 (クレジットは、インスタンスが停止した後、7 日間維持されます)
T3a	獲得クレジット、蓄積されたクレジット、消費されたクレジット、余剰クレジット (Unlimited モードのみ)	スタンダード、Unlimited (デフォルト)	7 日間 (クレジットは、インスタンスが停止した後、7 日間維持されます)
T3	獲得クレジット、蓄積されたクレジット、消費されたクレジット、余剰クレジット (Unlimited モードのみ)	スタンダード、Unlimited (デフォルト)	7 日間 (クレジットは、インスタンスが停止した後、7 日間維持されます)
前の世代			
T2	獲得クレジット、蓄積されたクレジット、使用されたクレジット、起動クレジット (スタンダードモードのみ)、余剰ク	スタンダード (デフォルト)、Unlimited	0 日 (インスタンスが停止するとクレジットは失われます)

タイプ	サポートされる CPU クレジットのタイプ	クレジットの設定モード	インスタンスの開始から停止までの間に蓄積された CPU クレジットのライフスパン
	レジット (Unlimited モードのみ)		

> **Note**
>
> Dedicated Host で起動される T3 インスタンスでは、Unlimited モードはサポートされていません。

CPU クレジットの獲得

各バーストパフォーマンスインスタンスは、インスタンスサイズに応じて、1 時間当たりの CPU クレジットを絶えず一定の割合で (ミリ秒レベルの細かさで) 獲得します。クレジットを蓄積または消費する会計処理もミリ秒レベルの細かさで実施されるため、CPU クレジットの浪費について心配する必要はありません。CPU の短期バーストでは CPU クレジットのごく一部しか使用されません。

バーストパフォーマンスインスタンスが使用する CPU リソースが、ベースライン使用率に必要な CPU リソースよりも少ない場合 (アイドル時など)、未使用の CPU クレジットが CPU クレジット残高に蓄積されます。バーストパフォーマンスインスタンスがベースライン使用率を超えてバーストする必要がある場合は、蓄積されたクレジットを消費します。CPU 使用率を増やす必要がある場合、バーストパフォーマンスインスタンスが蓄積したクレジットが多いほど、ベースラインを超えてバーストできる時間が増えます。

次の表は、バーストパフォーマンスインスタンスのタイプ、1 時間あたりに CPU クレジットを獲得するレート、インスタンスが蓄積できる獲得 CPU クレジットの最大数、インスタンスあたりの vCPU 数、およびコア全体に対する割合で表したベースライン使用率 (単一の vCPU を使用した場合) の一覧です。

インスタンスタイプ	1 時間あたりに受け取る CPU クレジット	蓄積可能な最大獲得クレジット*	vCPU 数	vCPU あたりのベースライン使用率
T2				
t2.nano	3	72	1	5%
t2.micro	6	144	1	10%
t2.small	12	288	1	20%
t2.medium	24	576	2	20%**
t2.large	36	864	2	30%**
t2.xlarge	54	1296	4	22.5%**
t2.2xlarge	81.6	1958.4	8	17%**
T3				
t3.nano	6	144	2	5%**
t3.micro	12	288	2	10%**
t3.small	24	576	2	20%**
t3.medium	24	576	2	20%**
t3.large	36	864	2	30%**
t3.xlarge	96	2304	4	40%**
t3.2xlarge	192	4608	8	40%**
T3a				
t3a.nano	6	144	2	5%**
t3a.micro	12	288	2	10%**

インスタンスタイプ	1 時間あたりに受け取る CPU クレジット	蓄積可能な最大獲得クレジット*	vCPU 数	vCPU あたりのベースライン使用率
t3a.small	24	576	2	20%**
t3a.medium	24	576	2	20%**
t3a.large	36	864	2	30%**
t3a.xlarge	96	2304	4	40%**
t3a.2xlarge	192	4608	8	40%**
T4g				
t4g.nano	6	144	2	5%**
t4g.micro	12	288	2	10%**
t4g.small	24	576	2	20%**
t4g.medium	24	576	2	20%**
t4g.large	36	864	2	30%**
t4g.xlarge	96	2304	4	40%**
t4g.2xlarge	192	4608	8	40%**

* 蓄積できるクレジットの数は、24 時間で獲得できるクレジットの数と同じです。

** 表内のベースライン使用率は vCPU 別の割合です。CloudWatch では、CPU 使用率は vCPU 別に表示されます。例えば、ベースラインレベルで動作する t3.large インスタンスの CPU 使用率は、CloudWatch の CPU メトリクスに 30% として表示されます。ベースライン使用率の計算方法については、ベースライン使用率を参照してください。

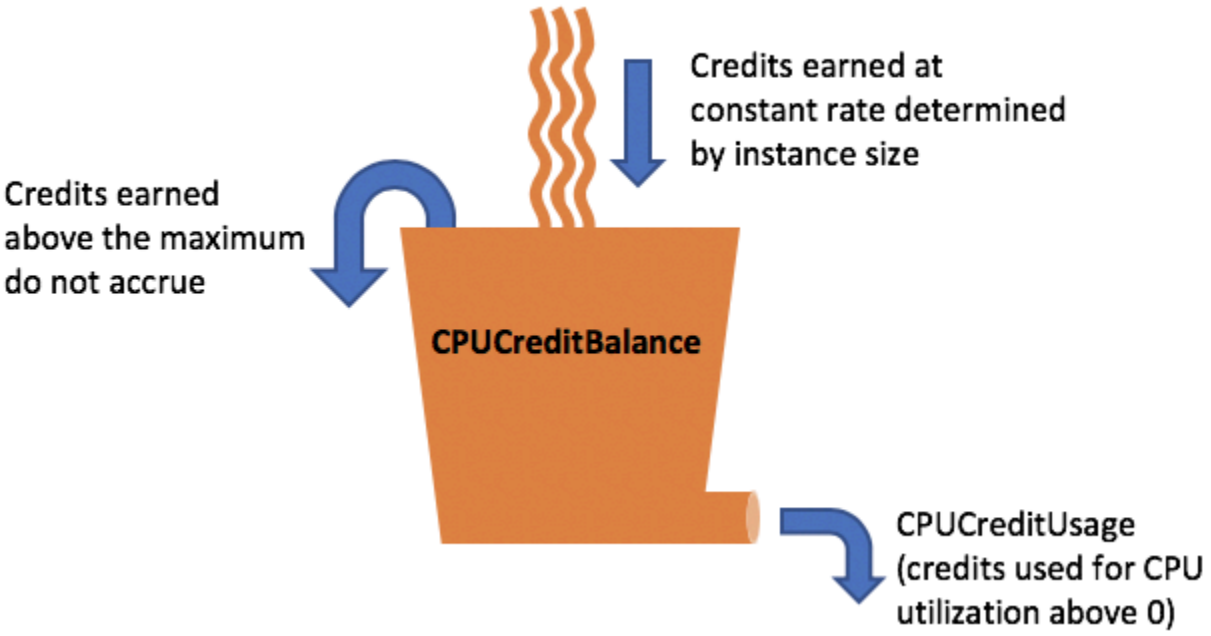
*** 各vCPUは、インテル Xeon コアまたは AMD EPYC コアのスレッドに対応します (T2 と T4g インスタンスを除く)。

CPU クレジットの獲得率

1 時間あたりに獲得する CPU クレジット数は、インスタンスのサイズによって決まります。例えば、t3.nano は 1 時間あたり 6 クレジットを獲得しますが、t3.small は 1 時間あたり 24 クレジットを獲得します。前記の表は、すべてのインスタンスのクレジット獲得率を示しています。

CPU クレジット蓄積制限

実行中のインスタンスで獲得されたクレジットが失効することはありませんが、インスタンスが蓄積できる獲得クレジットの数には制限があります。制限は、CPU クレジット残高により決まります。下記の図に示されているとおり、制限に到達すると、獲得された新しいクレジットはすべて破棄されます。フルバケットは CPU クレジット残高制限を示し、スピルオーバーは制限を超えた新しく獲得されたクレジットを示します。



CPU クレジット残高制限は、各 インスタンスのサイズによって異なります。例えば、t3.micro インスタンスは CPU クレジット残高で最大 288 の獲得 CPU クレジットを蓄積できます。前記の表は、各 インスタンスに累積できる獲得クレジットの最大数を示しています。

T2 スタンダードインスタンスは、起動クレジットも獲得します。起動クレジットは、CPU クレジット残高制限に対してカウントされません。T2 インスタンスがその起動クレジットを消費しておら

ず、獲得クレジットを蓄積しながら 24 時間以上アイドル状態が続いた場合、CPU クレジット残高は制限を上回って表示されます。詳細については「起動クレジット」を参照してください。

T4g、T3a、および T3 インスタンスでは、起動クレジットを獲得させることはできません。これらのインスタンスはデフォルトで unlimited として起動するため、起動クレジットなしでも起動後すぐにバーストできます。Dedicated Host で起動された T3 インスタンスはstandard(デフォルト)unlimitedモードは Dedicated Host の T3 インスタンスではサポートされていません。

CPU 存続期間の蓄積

実行中のインスタンスの CPU クレジットは失効しません。

T2 では、CPU クレジット残高は、インスタンスが停止して起動すると引き継がれません。T2 インスタンスを停止した場合、蓄積されたすべてのクレジットが失われます。

T4g、T3a、および T3では、インスタンスが停止した後も CPU クレジット残高は 7 日間保持され、その後に失われます。7 日以内にインスタンスを起動する場合、クレジットは失われません。

詳細については、CloudWatch メトリクスの表の CPUCreditBalance を参照してください。

ベースライン使用率

ベースライン使用率とは、CPU クレジットの獲得数と CPU クレジットの使用数が一致する場合に、正味のクレジット残高が 0 の状態で CPU を使用できるレベルのことです。ベースライン使用率はベースラインとも呼ばれます。

ベースライン使用率は、vCPU の使用に対する割合として表され、次のように計算されます。

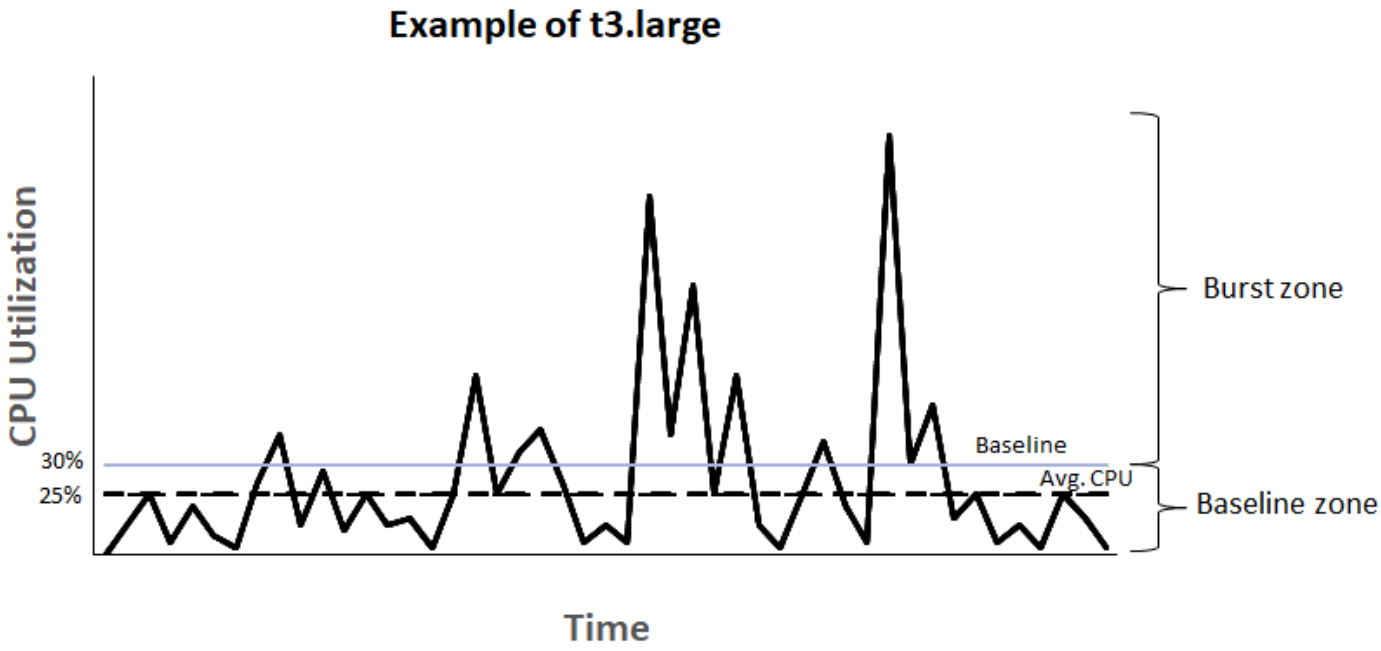
```
(number of credits earned/number of vCPUs)/60 minutes = % baseline
utilization
```

例えば、2 つの vCPU を持つ t3.nano インスタンスが 1 時間あたり 6 クレジットを獲得すると、ベースライン使用率は 5% になります。これは、次のように計算されます。

```
(6 credits earned/2 vCPUs)/60 minutes = 5% baseline utilization
```

2 つの vCPU を持つ t3.large インスタンスが 1 時間あたり 36 クレジットを獲得すると、ベースライン使用率は 30% になります ((36/2)/60)。

次のグラフは、平均 CPU 使用率がベースラインを下回っている t3.large の例を示しています。



バーストパフォーマンスインスタンスの Unlimited モード

unlimited として設定したバーストパフォーマンスインスタンスは、必要に応じた期間にわたり、高い CPU 使用率を保持できます。24 時間ごとのインスタンスの平均 CPU 使用率またはインスタンスの存続期間のいずれか短い方の時間で、インスタンスの平均 CPU 使用率がベースライン以下になった場合、1 時間ごとのインスタンス価格は自動的にすべての CPU 使用率スパイクをカバーします。

汎用のワークロードではほとんどの場合、unlimitedとして設定されたインスタンスは追加料金なしで十分なパフォーマンスを提供します。長時間にわたって高い CPU 使用率でインスタンスを実行する場合には、vCPU 時間ごとに均一追加料金が発生します。料金の詳細については、「Amazon EC2 の料金」および「T2/T3/T4g Unlimited モードの料金」を参照してください。

2025 年 7 月 15 日より前に AWS アカウントを作成し、t2.micro または t3.micro インスタンスをAWS 無料利用枠の範囲で unlimited モードで使用している場合、ローリング期間の 24 時間における平均使用率が、そのインスタンスのベースライン使用率を超過すると、料金が発生することがあります。

T4g、T3a、および T3 インスタンスは (デフォルトを変更していない場合) デフォルトでは unlimited で起動します。24 時間の平均 CPU 使用率がベースラインを超えた場合は、余剰クレジットに対して課金されます。スポットインスタンス を unlimited として起動し、CPU クレジットを計上するためのアイドル時間なしに、すぐに短期間使用する場合は、余剰クレジットの料金が

発生します。コストの増加を抑えるには、スポットインスタンス を標準モードで起動することをお勧めします。詳細については、余剰クレジットにより料金が発生することがあるおよびバーストパフォーマンスインスタンスを起動するを参照してください。

> **Note**
>
> Dedicated Host で起動された T3 インスタンスはstandard(デフォルト)unlimitedモードは Dedicated Host の T3 インスタンスではサポートされていません。

内容

- バーストインスタンスの無制限モードの概念
  - 無制限のバーストパフォーマンスインスタンスの仕組み
  - Unlimited モードと固定 CPU を使用する場合
  - 余剰クレジットにより料金が発生することがある
  - unlimited として設定されたバーストパフォーマンスのコストを教えてください。
  - T2 Unlimited インスタンスの起動クレジットはありません
  - 無制限モードの有効化
  - Unlimited とスタンダードを切り替えるとクレジットはどうなるか
  - クレジット使用状況のモニタリング
- バーストインスタンスの無制限モードの例
  - 例 1: T3 Unlimited でのクレジット使用についての説明
  - 例 2: T2 Unlimited でのクレジット使用についての説明

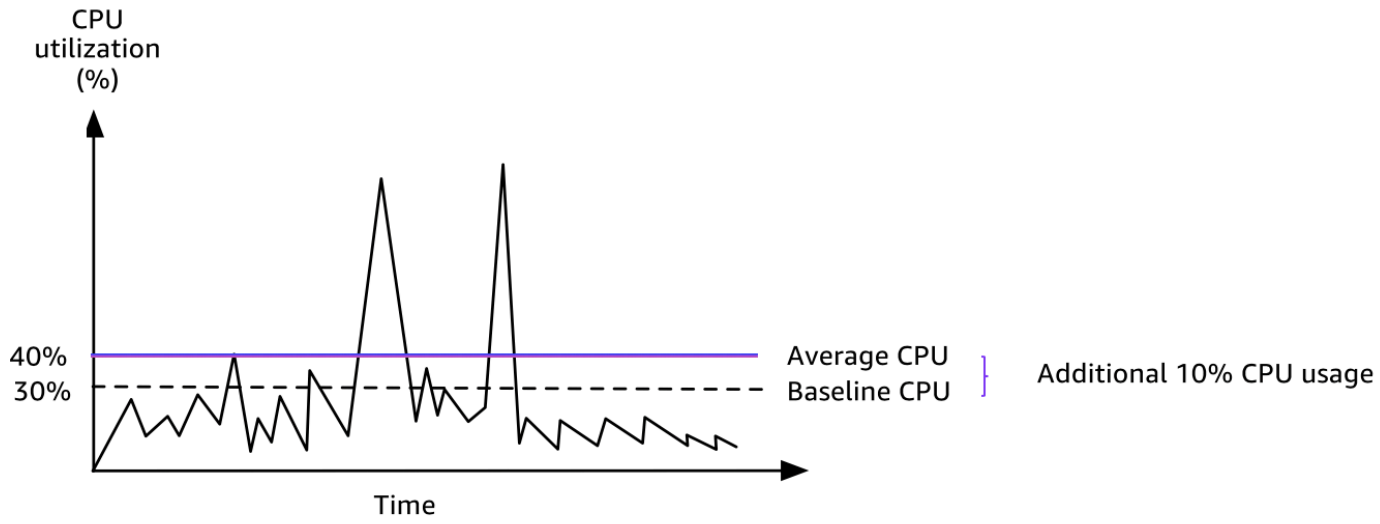
バーストインスタンスの無制限モードの概念

unlimited モードはバーストパフォーマンスインスタンスのクレジットの設定オプションです。これにより、実行中または停止中のインスタンスをいつでも有効または無効にできます。各 AWS リージョンのアカウントレベルで、バーストパフォーマンスインスタンスファミリーごとに、デフォルトのクレジットオプションとして unlimited を設定できます。アカウント内のすべての新しいバーストパフォーマンスインスタンスは、このデフォルトのクレジットオプションを使用して起動されます。

無制限のバーストパフォーマンスインスタンスの仕組み

unlimited として設定したバーストパフォーマンスインスタンスが CPU クレジット残高を使い切った場合、ベースラインを超えてバーストするには余剰クレジットを使用できます。その CPU 利用率がベースラインを下回った場合、獲得した CPU クレジットを使用して、先に消費された余剰クレジットの支払いが行われます。CPU クレジットを獲得して余剰クレジットを支払う機能により、Amazon EC2 は 24 時間にわたるインスタンスの CPU 使用率を平均化できるようになります。24 時間の平均 CPU 使用率がベースラインを超えたインスタンスには、vCPU 時間あたりの超過の使用量に対して、均一追加料金が発生します。

以下のグラフは、t3.large の CPU 使用率を示します。t3.large のベースラインの CPU 使用率は 30% です。インスタンスが 24 時間にわたって平均 30% 以下の CPU 使用率で実行されている場合、コストはインスタンスの 1 時間あたりの料金ですでにカバーされているため、追加料金はかかりません。ただし、ここでのグラフに示されているように、24 時間の平均 CPU 使用率 40% でインスタンスが実行されている場合、そのインスタンスでの超過 CPU 使用量 10% に対しては、vCPU 時間あたりに均一追加料金が発生します。



各インスタンスタイプの vCPU あたりのベースライン使用率、および各インスタンスタイプが獲得するクレジット数の詳細については、クレジットの表を参照してください。

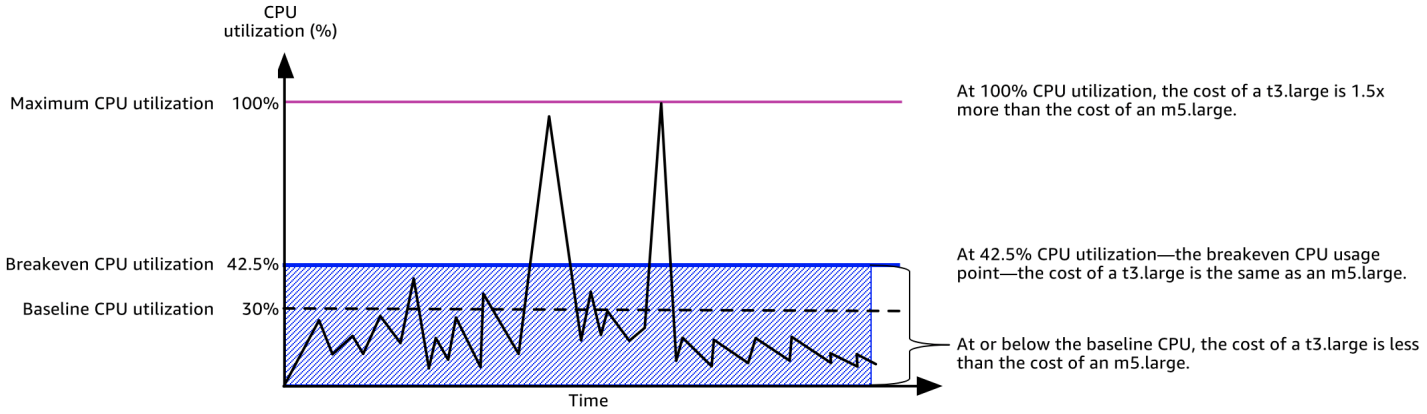
Unlimited モードと固定 CPU を使用する場合

unlimited モード (T3 など) でバーストパフォーマンスインスタンスと、固定パフォーマンスインスタンス (M5 など) のどちらを使用するかを決める場合は、損益分岐点 CPU 使用率を判断する必要があります。バーストパフォーマンスインスタンスの損益分岐点 CPU 使用率は、バーストパフォー

マンスインスタンスが固定パフォーマンスインスタンスと同じコストになるポイントです。損益分岐点 CPU 使用率は、次のことを判断するのに役立ちます。

- 24 時間の平均 CPU 使用率が損益分岐点の CPU 使用率以下である場合は、バーストパフォーマンスインスタンスを unlimited モードで使用すると、固定パフォーマンスインスタンスと同じパフォーマンスを維持しながら、バーストパフォーマンスインスタンスを低価格で使用できます。
- 24 時間の平均 CPU 使用率が損益分岐点 CPU 使用率を上回る場合、バーストパフォーマンスインスタンスは、同等サイズの固定パフォーマンスインスタンスよりもコストが高くなります。T3 インスタンスが 100% CPU で継続的にバーストする場合、同等サイズの M5 インスタンスの約 1.5 倍の価格を支払うことになります。

次のグラフは、t3.large のコストが m5.large と同じ場合の損益分岐点の CPU 使用率を示しています。t3.large の損益分岐点の CPU 使用率は 42.5% です。平均 CPU 使用率が 42.5% の場合、t3.large の実行コストは m5.large と同じです。平均 CPU 使用率が 42.5% を超える場合は、より高価になります。ワークロードの平均 CPU 使用率が 42.5% 未満であれば、m5.large と同じパフォーマンスを得ながら、t3.large を低コストで使用できます。



次の表は、unlimited モードまたは固定パフォーマンスインスタンスでバーストパフォーマンスインスタンスを使用する方が安価な場合を判断できるように、損益分岐点 CPU 使用率のしきい値を計算する方法を示しています。表の列には A から K のラベルが付けられています。

インスタンスタイプ	vCPU	T3 の料金 */時間	M5 の料金 */時間	料金の違い	vCPU あたりの T3 ベースライン使用率 (%)	余剰クレジットに対する vCPU 時間あたりの料金	vCPU 時間 (分) あたりの料金	vCPU ごとに利用可能な追加のバースト (分)	利用可能な追加 CPU (%)	損益分岐 CPU %
A	B	C	D	E = D - C	F	G	H = G / 60	I = E / H	J = (I / 60) / B	K = F + J
t3.large	2	0.0835 USD	0.096 USD	0.0125 USD	30%	0.05 USD	0.000833 USD	15	12.5%	42.5%

* 料金は us-east-1 および Linux OS に基づいています。

テーブルは以下の情報を提供します。

- 列 A は、インスタンスタイプ `t3.large` を示します。
- 列 B は、`t3.large` の vCPU の数を示します。
- 列 C は、1 時間あたりの `t3.large` の料金を示します。
- 列 D は、1 時間あたりの `m5.large` の料金を示します。
- 列 E は、`t3.large` と `m5.large` の差額を示しています。
- 列 F は、`t3.large` の vCPU あたりのベースライン使用率 (30%) を示しています。ベースラインでは、インスタンスの 1 時間あたりのコストが CPU 使用率のコストになります。
- G 列は、獲得したクレジットを使い切った後のインスタンスが 100% の CPU 使用率でバーストした場合に請求される、vCPU 時間あたりの均一追加料金を示しています。

- H 列は、獲得したクレジットを使い切った後のインスタンスが 100% の CPU 使用率でバーストした場合に請求される、vCPU 分あたりの均一追加料金を示しています。
- 列 I は、t3.large と同じ 1 時間あたりの料金が発生している間に、100% CPU で m5.large が 1 時間あたりにバーストできる追加の時間 (分) を示しています。
- J 列は、インスタンスが m5.large と同じ 1 時間あたりの料金が発生している間にバーストする可能性がある、ベースラインを超えた追加の CPU 使用率 (%) を示しています。
- 列 K は、t3.large 以上支払わなくても m5.large がバーストする可能性がある損益分岐点 CPU 使用率 (%) を示しています。この使用率を超えた t3.large のコストは m5.large よりも高くなります。

次の表は、同様のサイズの M5 インスタンスタイプと比較した、T3 インスタンスタイプの損益分岐点 CPU 使用率 (%) を示しています。

T3 インスタンスタイプ	M5 と比較した T3 の損益分岐点 CPU 使用率 (%)
t3.large	42.5%
t3.xlarge	52.5%
t3.2xlarge	52.5%

余剰クレジットにより料金が発生することがある

インスタンスの平均 CPU 使用率がベースライン以下の場合、インスタンスに追加料金は発生しません。インスタンスは 24 時間でクレジット最大数を獲得 (例えば、t3.micro インスタンスは 24 時間で最大 288 クレジット獲得可能) するため、課金されることなく余剰クレジットを最大まで消費できます。

ただし、CPU 使用率がベースラインを上回ったままの場合、インスタンスは消費した余剰クレジットを支払うのに十分なクレジットを獲得できません。支払われない余剰クレジットに対して、vCPU 時間ごとに均一追加料金が発生します。料金の詳細については、T2/T3/T4g Unlimited モードの料金およびを参照してください。

先に消費された余剰クレジットは、以下のいずれかの状況に当てはまると料金が発生します。

- 消費された余剰クレジットが、インスタンスが 24 時間に獲得できる最大クレジット数を超えている。最大数を越えて消費された余剰クレジットは、時間の最後に課金されます。
- インスタンスが停止または終了した。
- インスタンスは unlimited から standard に切り替わります。

消費された余剰クレジットは、CloudWatch メトリクス CPUSurplusCreditBalance により追跡されます。課金された余剰クレジットは、CloudWatch メトリクス CPUSurplusCreditsCharged で追跡できます 。詳細については「バーストパフォーマンスインスタンスの追加 CloudWatch メトリクス」を参照してください。

unlimited として設定されたバーストパフォーマンスのコストを教えてください。

余剰クレジットを使用していて、それらが獲得クレジットで返済されない場合は (「余剰クレジットにより料金が発生することがある」を参照)、余剰クレジットに対して vCPU 時間あたりの定額追加料金をお支払いいただきます。料金は、「Amazon EC2 オンデマンド料金」ページの「T2/T3/T4g Unlimited モードの料金」セクションに記載されています。

T2 Unlimited インスタンスの起動クレジットはありません

T2 スタンダードインスタンスが起動クレジットを受け取っても、T2 Unlimited インスタンスは起動クレジットを受け取りません。T2 Unlimited インスタンスは、24 時間のローリング枠内または存続期間のどちらか短いほうで平均 CPU 使用率がベースラインを越えない限り、追加料金なしでいつでもベースラインを超えるバーストができます。したがって、T2 Unlimited インスタンスは、起動直後の高パフォーマンスを実現するために起動クレジットを必要としません。

T2 インスタンスが standard から unlimited に切り替えられた場合、残りの CPUCreditBalance が引き継がれる前に、蓄積された起動クレジットが CPUCreditBalance から削除されます。

T4g、T3a、および T3 インスタンスは、デフォルトで無制限モードで起動するため、起動クレジットを受け取ることはありません。そのため、起動時にすぐにバーストする可能性があります。T4g、T3a、および T3 インスタンスのクレジット設定を Unlimited モードにすることで、ベースラインを超えてバーストさせるために必要な量の CPU リソースを、必要な期間だけ使用できるようになります。

無制限モードの有効化

実行中または停止中のインスタンスで、unlimited から standard、standard から unlimited へいつでも切り替えることができます。詳細については、起動時にクレジット仕様を設定するおよびバーストパフォーマンスインスタンスのクレジット仕様を管理するを参照してください。

各 unlimited リージョンのアカウントレベルで、バーストパフォーマンスインスタンスファミリーごとに、デフォルトのクレジットオプションとして AWS を設定できます。アカウント内のすべての新しいバーストパフォーマンスインスタンスは、このデフォルトのクレジットオプションを使用して起動されます。詳細については「アカウントのデフォルトのクレジット仕様を管理する」を参照してください。

Amazon EC2 コンソールまたは unlimited を使用して、バーストパフォーマンスインスタンスが standard あるいは AWS CLI のどちらで設定されているを確認できます。詳細については「バーストパフォーマンスインスタンスを設定する」を参照してください。

Unlimited とスタンダードを切り替えるとクレジットはどうなるか

CPUCreditBalance は、インスタンスが蓄積したクレジットの数を追跡する CloudWatch メトリクスです。CPUSurplusCreditBalance は、インスタンスが消費した余剰クレジットの数を追跡する CloudWatch メトリクスです。

unlimited として設定されたインスタンスを standard に変更すると、以下の状況が発生します

- CPUCreditBalance 値は変更されずに引き継がれます。
- CPUSurplusCreditBalance 値にはすぐに課金されます。

standard インスタンスが unlimited に切り替えられると、以下の状況が発生します。

- CPUCreditBalance 値に含まれる、蓄積された獲得クレジットが引き継がれます。
- T2 スタンダードインスタンスでは、起動クレジットがすべて CPUCreditBalance 値から削除され、蓄積された獲得クレジットを含む残りの CPUCreditBalance 値が引き継がれます。

クレジット使用状況のモニタリング

インスタンスが、ベースラインが提供するよりも多くクレジットを消費していないか確認するには、CloudWatch メトリクスを使用して使用率を追跡し、クレジット使用量を通知する時間ごとのアラームを設定できます。詳細については「バーストインスタンスの CPU クレジットをモニタリングする」を参照してください。

バーストインスタンスの無制限モードの例

次の例では、unlimited として設定されているインスタンスのクレジットの使用について説明します。

例

- 例 1: T3 Unlimited でのクレジット使用についての説明
- 例 2: T2 Unlimited でのクレジット使用についての説明

例 1: T3 Unlimited でのクレジット使用についての説明

この例では、t3.nano として起動した unlimited インスタンスの CPU 使用率、獲得クレジットおよび余剰クレジットを使用して CPU 使用率を保持する方法を示します。

t3.nano インスタンスは、24 時間のローリング期間に渡って最大で 144 CPU クレジットを獲得し、それを 144 分の vCPU 使用と引き換えることができます。CPU クレジット残高 (CloudWatch メトリクス CPUCreditBalance で示される) が消耗すると、余剰 CPU クレジット — まだ獲得していない — を消費して必要なだけバーストします。t3.nano インスタンスは 24 時間あたり最大 144 クレジットを獲得するため、すぐに課金されることなく余剰クレジットを最大まで消費できます。CPU クレジットを 144 以上消費した場合、差分については時間の最後に課金されます。

以下のグラフにある例の目的は、CPUCreditBalance を使いきった後でも余剰クレジットを使用してインスタンスをバーストさせる方法を示すことです。以下のワークフローは、グラフの番号付きの点を参照します。

P1 – グラフの 0 時において、インスタンスは unlimited として起動され、すぐにクレジットを獲得します。このインスタンスは起動時からアイドル状態になり (CPU 使用率は 0%)、クレジットは消費されません。すべての未消費のクレジットはクレジット残高に蓄積されます。最初の 24 時間は、CPUCreditUsage は 0 で、CPUCreditBalance 値は、最大の 144 に達します。

P2 – 次の 12 時間では、CPU 使用率はベースラインの 5% を下回る 2.5% です。インスタンスは消費するよりも多くのクレジットを獲得しますが、CPUCreditBalance 値は、最大 144 クレジットを超えることはできません。

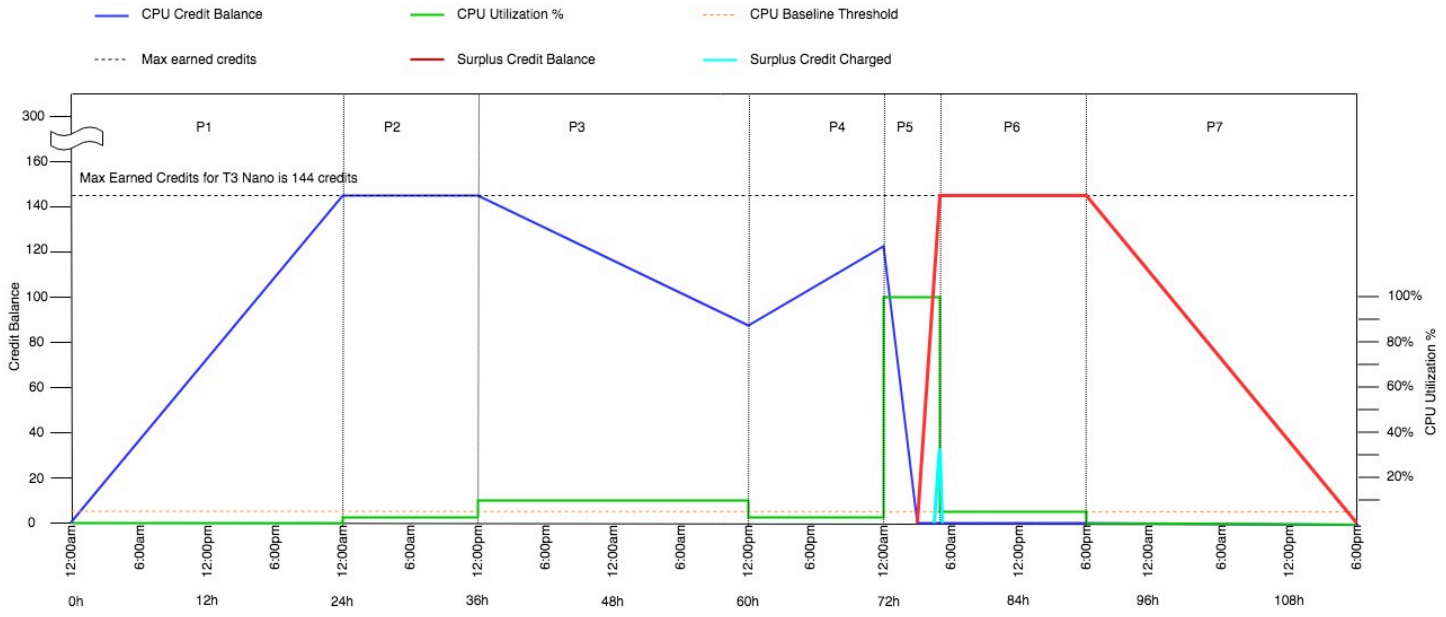
P3 – 次の 24 時間では、CPU 使用率は 7% (ベースラインを上回る) で 57.6 クレジットの消費を必要とします。インスタンスは獲得するよりも多くのクレジットを消費し、CPUCreditBalance 値は、86.4 クレジットに低減します。

P4 – 次の 12 時間では、CPU 使用率は 2.5% に減少し (ベースラインを下回る) で 36 クレジットの消費を必要とします。同時に、インスタンスは 72 クレジットを獲得します。インスタンスは消費するよりも多くのクレジットを獲得し、CPUCreditBalance 値は、122 クレジットに増加します。

P5 – 次の 5 時間で、インスタンスは 100% の CPU 使用率でバーストし、バーストを保持するために 570 クレジットを消費します。この期間内の約 1 時間で、インスタンスは CPUCreditBalance 全体の 122 クレジットを使い切り、高い CPU 使用率を維持するために余剰クレジットを使い始めます。この期間の余剰クレジット数は合計 448 (570-122=448) です。CPUSurplusCreditBalance 値が 144 CPU クレジット (t3.nano インスタンスが 24 時間に獲得できるクレジットの最大数) に達すると、その後に消費される余剰クレジットは獲得クレジットで相殺することはできません。その後に消費される余剰クレジットの量は 304 (448-144=304) クレジットで、時間の終了後に 304 クレジットに対して少額の追加料金が発生します。

P6 – 次の 13 時間では、CPU 使用率は 5% (ベースライン) です。インスタンスは消費したのと同量のクレジットを獲得するため、CPUSurplusCreditBalance の支払いを超過しません。CPUSurplusCreditBalance 値は、144 クレジットのままです。

P7 – この例の過去 24 時間では、インスタンスはアイドル状態で、CPU 使用率は 0% です。この間、インスタンスは、CPUSurplusCreditBalance の支払いに使用する 144 クレジットを獲得します。



例 2: T2 Unlimited でのクレジット使用についての説明

この例では、t2.nano として起動した unlimited インスタンスの CPU 使用率、獲得クレジットおよび余剰クレジットを使用して CPU 使用率を保持する方法を示します。

t2.nano インスタンスは、24 時間のローリング期間に渡って最大で 72 CPU クレジットを獲得し、それを 72 分の vCPU 使用と引き換えることができます。CPU クレジット残高 (CloudWatch メトリクス CPUCreditBalance で示される) が消耗すると、余剰 CPU クレジット — まだ獲得していない — を消費して必要なだけバーストします。t2.nano インスタンスは 24 時間あたり最大 72 クレジットを獲得するため、すぐに課金されることなく余剰クレジットを最大まで消費できます。CPU クレジットを 72 以上消費した場合、差分については時間の最後に課金されます。

以下のグラフにある例の目的は、CPUCreditBalance を使いきった後でも余剰クレジットを使用してインスタンスをバーストさせる方法を示すことです。グラフ中のタイムライン開始時点で、インスタンスは 24 時間に獲得可能なクレジットの最大数と同じクレジット残高を蓄積しているものとします。以下のワークフローは、グラフの番号付きの点を参照します。

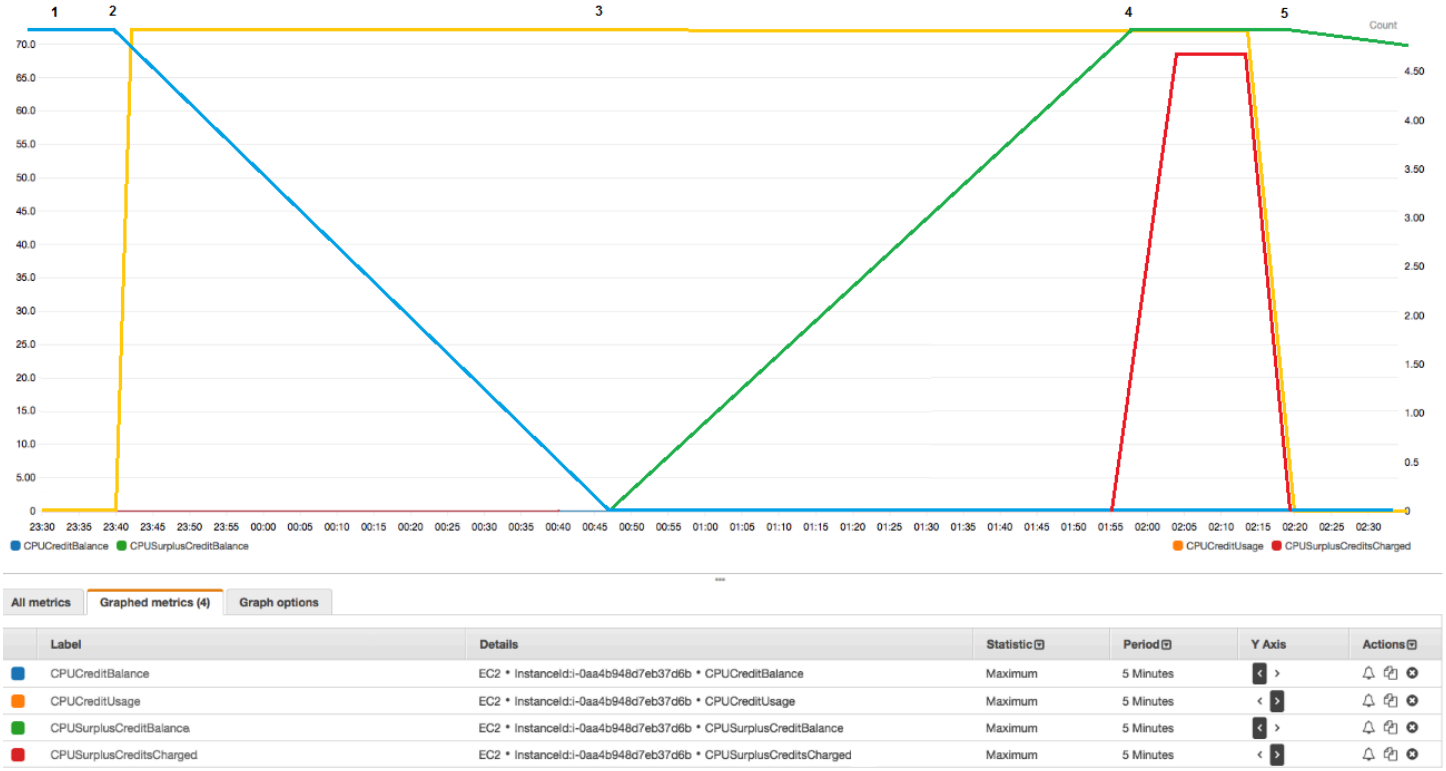
1 – 最初の 10 分間、CPUCreditUsage は 0 で、CPUCreditBalance 値は最大の 72 のままです。

2 – 23:40 に CPU 使用率が増加すると、インスタンスは CPU クレジットを消費し CPUCreditBalance 値が減少します。

3 – 00:47 頃、インスタンスは CPUCreditBalance 全体を使い切り、高い CPU 使用率を維持するために余剰クレジットを使い始めます。

4 – CPUSurplusCreditBalance 値が 72 CPU クレジットに達する 1:55 まで余剰クレジットが消費されます。これは、t2.nano インスタンスが 24 時間で獲得できる最大値と同じです。その後に消費される余剰クレジットは、24 時間以内の獲得クレジットで相殺することはできません。そのため、時間終了時に少額の追加料金が発生します。

5 – インスタンスは 2:20 頃まで余剰クレジットを消費し続けます。この時点で、CPU 使用率がベースラインを下回ると、インスタンスは 1 時間あたり 3 クレジット (または 5 分ごとに 0.25 クレジット) を獲得し始めます。これは、CPUSurplusCreditBalance の支払いに使用されます。CPUSurplusCreditBalance 値が 0 まで減った後、インスタンスは 5 分ごとに 0.25 クレジットの割合で CPUCreditBalance に獲得クレジットを蓄積し始めます。



請求書の計算 (Linux インスタンス)

超過クレジットは vCPU 時間あたり 0.05 USD かかります。インスタンスは、1:55 から 2:20 の間におよそ 25 余剰クレジットを消費し、これは 0.42 vCPU 時間に相当します。このインスタンスの追加料金は、0.42 vCPU 時間 x 0.05 USD/vCPU 時間 = 0.021 USD、四捨五入して 0.02 USD です。これが、この T2 Unlimited インスタンスの月末請求書です。

Amazon Elastic Compute Cloud running Linux/UNIX		
$0.0058 per On Demand Linux t2.nano Instance Hour	720.000 Hrs	$4.18

Amazon Elastic Compute Cloud T2 CPU Credits		
$0.05 per vCPU-Hour of T2 CPU credits	0.420 vCPU-Hours	$0.02

請求書の計算 (Windows インスタンス)

超過クレジットは vCPU 時間あたり 0.096 USD かかります。インスタンスは、1:55 から 2:20 の間におよそ 25 余剰クレジットを消費し、これは 0.42 vCPU 時間に相当します。このインスタンスの追加料金は、0.42 vCPU 時間 x 0.096 USD/vCPU 時間 = 0.04032 USD、四捨五入して 0.04 USD です。これが、この T2 Unlimited インスタンスの月末請求書です。

Amazon Elastic Compute Cloud running Windows		
$0.0081 per On Demand Windows t2.nano Instance Hour	720.000 Hrs	$5.83

Amazon Elastic Compute Cloud T2 CPU Credits		
$0.096 per vCPU-Hour of T2 CPU credits	0.420 vCPU-Hours	$0.04

1 時間ごとの料金の発生を通知する請求アラートを設定して、必要に応じてアクションを実行できます。

バーストパフォーマンスインスタンスのスタンダードモード

`standard` として設定したバーストパフォーマンスインスタンスは、平均 CPU 使用率がインスタンスのベースライン CPU 使用率を一貫して下回るワークロードに適しています。ベースラインより上にバーストする場合、インスタンスは CPU クレジット残高に蓄積されたクレジットを消費します。インスタンスの蓄積されたクレジットが少なくなると、CPU 使用率は徐々にベースラインパフォーマンスレベルまで下がるため、蓄積された CPU クレジット残高を使い切った場合でも、パフォーマンスが急激に低下することはありません。詳細については「バーストパフォーマンスインスタンスに関する主要な概念」を参照してください。

内容

- バーストインスタンスのスタンダードモードの概念
  - スタンダードのバーストパフォーマンスインスタンスの仕組み
  - 起動クレジット
  - 起動クレジット制限
  - 起動クレジットと獲得クレジットの違い
- バーストインスタンスのスタンダードモードの例
  - 例 1: T3 スタンダードでのクレジット使用についての説明
  - 例 2: T2 スタンダードでのクレジット使用についての説明
    - 期間 1: 1 ~ 24 時間
    - 期間 2: 25 ~ 36– 時間
    - 期間 3: 37 ~ 61 時間
    - 期間 4: 62 ~ 72 時間
    - 期間 5: 73 ~ 75 時間
    - 期間 6: 76 ~ 90 時間

- 期間 7: 91 ~ 96 時間

バーストインスタンスのスタンダードモードの概念

standard モードはバーストパフォーマンスインスタンスの設定オプションです。これにより、実行中または停止中のインスタンスをいつでも有効または無効にできます。各 AWS リージョンのアカウントレベルで、バーストパフォーマンスインスタンスファミリーごとに、デフォルトのクレジットオプションとして standard を設定できます。アカウント内のすべての新しいバーストパフォーマンスインスタンスは、このデフォルトのクレジットオプションを使用して起動されます。

スタンダードのバーストパフォーマンスインスタンスの仕組み

standard に設定されているバーストパフォーマンスインスタンスが実行状態の場合、1 時間当たりの獲得クレジットを絶えず一定の割合で (ミリ秒レベルの細かさで) 獲得します。T2 スタンダードインスタンスが停止すると、蓄積されたクレジットがすべて失われ、クレジット残高はゼロにリセットされます。再起動されると、新しい起動クレジットのセットを受け取り、獲得したクレジットの蓄積を始めます。T4g、T3a、および T3 スタンダードインスタンスでは、CPU クレジット残高は、インスタンスが停止した後も 7 日間保持された後失われます。7 日以内にインスタンスを起動する場合、クレジットは失われません。

T2 Standard インスタンスは、獲得クレジットと起動クレジットの 2 種類の CPU クレジットを受け取ります。T2 スタンダードインスタンスが実行状態の場合、1 時間当たりの獲得クレジットを絶えず一定の割合で (ミリ秒レベルの細かさで) 獲得します。スタート時のインススタンスは、良いスタートアップエクスペリエンスのためのクレジットをまだ獲得していません。したがって、スタートアップエクスペリエンスを積み重ねるために、スタート時にクレジットを獲得します。インスタンスは、獲得クレジットを蓄積しながら最初にそのクレジットを消費します。

T4g、T3a、および T3 インスタンスは、Unlimited モードをサポートしているため、起動クレジットを受け取りません。T4g、T3a、および T3 インスタンスのクレジット設定を Unlimited モードにすることで、ベースラインを超えてバーストさせるために必要な量の CPU リソースを、必要な期間だけ使用できるようになります。

起動クレジット

T2 スタンダードインスタンスは、起動時またはスタート時に vCPU あたり 30 起動クレジットを獲得します。T1 スタンダードインスタンスは 15 起動クレジットを獲得します。例えば、t2.micro インスタンスは vCPU が 1 つのため 30 起動クレジット、t2.xlarge インスタンスには vCPU が 4 つあるため 120 起動クレジットを取得します。起動クレジットは、インスタンスが獲得クレジット

を蓄積できるようになる前に、起動してすぐにバーストできるよう、最適な起動エクスペリエンスを提供するために設計されています。

起動クレジットは、獲得クレジットよりも先に消費されます。未使用の起動クレジットは CPU クレジット残高に蓄積されますが、CPU クレジット残高制限に対してカウントされません。例えば、t2.micro インスタンスの CPU クレジット残高制限は 144 獲得クレジットです。起動された後 24 時間アイドルのままであった場合、その CPU クレジット残高は 174 に到達し (30 起動クレジット + 144 獲得クレジット)、制限を上回ります。ただし、インスタンスが 30 起動クレジットを消費した後は、クレジット残高が 144 を超えることはありません。各インスタンスサイズの CPU クレジット残高制限の詳細については、クレジットの表を参照してください。

次の表は、起動または開始の際に受け取る初期 CPU クレジットの割り当てと vCPU の数を示しています。

インスタンスタイプ	起動クレジット	vCPU
t1.micro	15	1
t2.nano	30	1
t2.micro	30	1
t2.small	30	1
t2.medium	60	2
t2.large	60	2
t2.xlarge	120	4
t2.2xlarge	240	8

起動クレジット制限

T2 スタンダードインスタンスが起動クレジットを受け取る回数には制限があります。デフォルトの制限は、リージョンごとにローリング期間の 24 時間あたり各アカウントで合計で 100 回の T2 スタンダードインスタンスの起動または開始と設定されています。例えば、24 時間以内にインスタンスが 100 回停止および開始した場合、24 時間以内に 100 インスタンスが起動された場合、または他の

組み合わせが 100 回の開始と同じである場合、制限に到達します。新しいアカウントでは、使用量に基づいて増える下限が設定される場合があります。

> **Tip**
>
> ワークロードに必要なパフォーマンスを常に確実に得るには、バーストパフォーマンスインスタンスの Unlimited モード に切り替えるか、またはより大きいインスタンスサイズの使用を検討してください。

起動クレジットと獲得クレジットの違い

次の表に、起動クレジットと獲得クレジットの違いを示します。

	起動クレジット	獲得クレジット
クレジットの獲得率	T2 スタンダードインスタンスは、起動時またはスタート時に vCPU あたり 30 起動クレジットを獲得します。 T2 インスタンスが `unlimited` から `standard` に切り替えられた場合、切り替えの時点では起動クレジットを取得しません。	各 T2 インスタンスは、インスタンスサイズに応じて、1 時間当たりの CPU クレジットを絶えず一定の割合で (ミリ秒レベルの細かさで) 獲得します。インスタンスサイズごとに獲得される CPU クレジット数の詳細については、クレジットの表を参照してください。
クレジットの獲得制限	起動クレジット受け取り制限は、リージョンごとにローリング期間の 24 時間あたり各アカウントで合計で 100 回の T2 スタンダードインスタンスの起動または開始と設定されています。新しいアカウントでは、使用量に基づいて増える下限が設定される場合があります。	T2 インスタンスは、CPU クレジット残高制限より多くのクレジットを蓄積することはできません。CPU クレジット残高がその制限に到達した場合、制限に到達した後に獲得されたクレジットはすべて破棄されます。起動クレジットは制限に対してはカウントされません。各 T2 インスタンスサイズの CPU クレジット残高制限の詳細については、クレジットの表を参照してください。

	起動クレジット	獲得クレジット
クレジットの使用	起動クレジットは、獲得クレジットよりも先に消費されます。	獲得クレジットは、すべての起動クレジットを消費した後にのみ消費されます。
クレジットの有効期限	T2 スタンダードインスタンスが実行中の場合、起動クレジットは期限切れになりません。T2 スタンダードインスタンスが停止し、T2 Unlimited に切り替えられた場合、すべての起動クレジットが失われます。	T2 インスタンスが実行中の場合、蓄積した獲得クレジットは期限切れになりません。T2 インスタンスが停止すると、蓄積された獲得クレジットはすべて失われます。

蓄積された起動クレジットと蓄積された獲得クレジットの数は、CloudWatch メトリクス `CPUCreditBalance` によって追跡されます 詳細については、CloudWatch メトリクスの表の `CPUCreditBalance` を参照してください。

バーストインスタンスのスタンダードモードの例

次の例では、インスタンスが `standard` として設定された場合の、クレジットの使用について説明します。

例

- 例 1: T3 スタンダードでのクレジット使用についての説明
- 例 2: T2 スタンダードでのクレジット使用についての説明

例 1: T3 スタンダードでのクレジット使用についての説明

この例では、`t3.nano` として起動した `standard` インスタンスが、獲得クレジットを、獲得、蓄積、消費する方法について示します。クレジット残が、蓄積された獲得クレジットを反映するかについて示します。

実行中の `t3.nano` インスタンスは、24 時間ごとに 144 クレジットを獲得します。このクレジット残の制限は、144 の獲得クレジットです。制限に到達すると、獲得された新しいクレジットはすべて破棄されます。獲得および蓄積できるクレジット数の詳細については、クレジットの表を参照してください。

T3 スタンダードインスタンスを起動し、すぐに使用することができます。または、T3 スタンダードインスタンスを起動し、何日間かアイドル状態にしてから、そこでアプリケーションを実行する場合があります。インスタンスを使用中か、アイドル状態であるかによって、クレジットが消費されるか、あるいは蓄積されるかが決まります。インスタンスが起動してから 24 時間アイドル状態のままの場合、蓄積できる獲得クレジットの最大数となり、クレジット残高が制限に達します。

この例では、起動後に 24 時間アイドル状態のままとなるインスタンスについて説明します。また、96 時間にわたる 7 つの期間で、クレジットが獲得、蓄積、消費、破棄される率と、各期間の終了時点でのクレジット残高の値を示します。

以下のワークフローは、グラフの番号付きの点を参照します。

P1 – グラフの 0 時において、インスタンスは `standard` として起動され、すぐにクレジットを獲得します。このインスタンスは起動時からアイドル状態になり (CPU 使用率は 0%)、クレジットは消費されません。すべての未消費のクレジットはクレジット残高に蓄積されます。最初の 24 時間は、`CPUCreditUsage` は 0 で、`CPUCreditBalance` 値は、最大の 144 に達します。

P2 – 次の 12 時間では、CPU 使用率はベースラインの 5% を下回る 2.5% です。インスタンスは消費するよりも多くのクレジットを獲得しますが、`CPUCreditBalance` 値は、最大 144 クレジットを超えることはできません。制限を超えて獲得されたクレジットはすべて破棄されます。

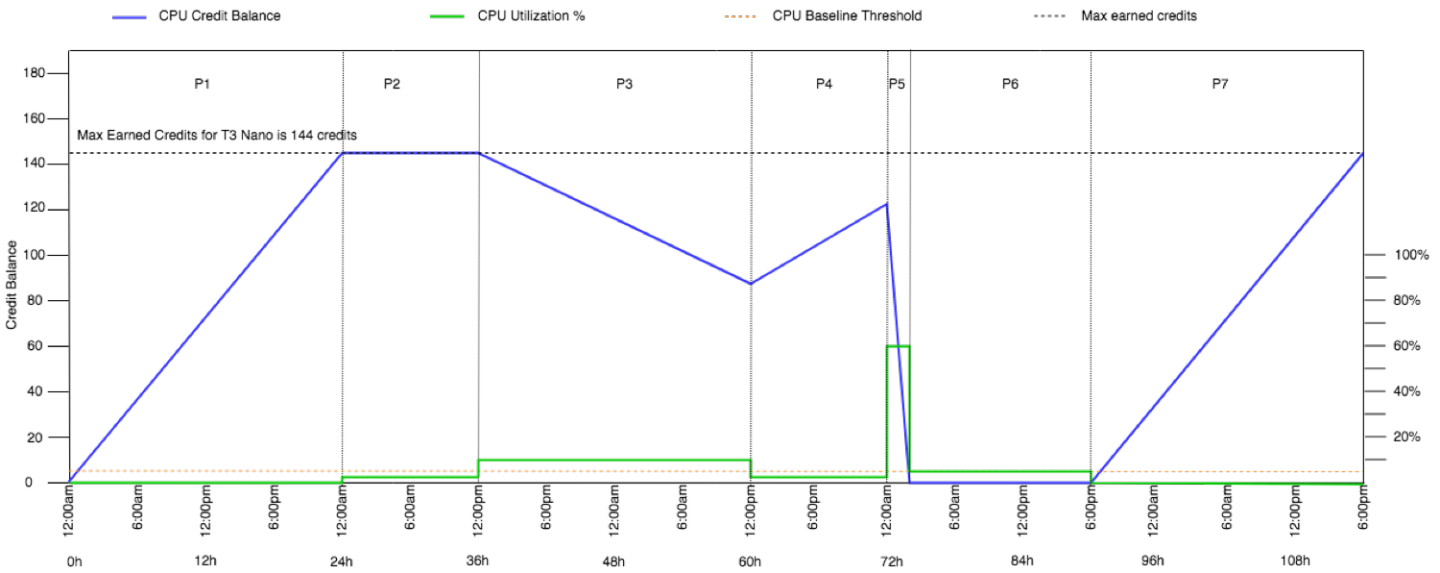
P3 – 次の 24 時間では、CPU 使用率は 7% (ベースラインを上回る) で 57.6 クレジットの消費を必要とします。インスタンスは獲得するよりも多くのクレジットを消費し、`CPUCreditBalance` 値は、86.4 クレジットに低減します。

P4 – 次の 12 時間では、CPU 使用率は 2.5% に減少し (ベースラインを下回る) で 36 クレジットの消費を必要とします。同時に、インスタンスは 72 クレジットを獲得します。インスタンスは消費するよりも多くのクレジットを獲得し、`CPUCreditBalance` 値は、122 クレジットに増加します。

P5 – 次の 2 時間で、インスタンスは 60% の CPU 使用率でバーストし、122 クレジットの全体 `CPUCreditBalance` 値を使い切ります。この期間の終わりに、`CPUCreditBalance` は 0 になり、CPU 使用率はベースライン使用率レベルの 5% まで強制的に落とされます。ベースラインで、インスタンスは消費した分のクレジットを獲得します。

P6 – 次の 14 時間では、CPU 使用率は 5% (ベースライン) です。インスタンスは消費した分のクレジットを獲得します。`CPUCreditBalance` 値は、0 のままです。

P7 – この例の過去 24 時間では、インスタンスはアイドル状態で、CPU 使用率は 0% です。この間、インスタンスは、`CPUCreditBalance` に蓄積する 144 クレジットを獲得します。



例 2: T2 スタンダードでのクレジット使用についての説明

この例では、t2.nano が起動クレジットおよび獲得クレジットを獲得、蓄積、消費する際に、standard インスタンスがどのように起動されるかについて示します。クレジット残高に、蓄積された獲得クレジットだけでなく、蓄積された起動クレジットがどのように反映されるかについて示します。

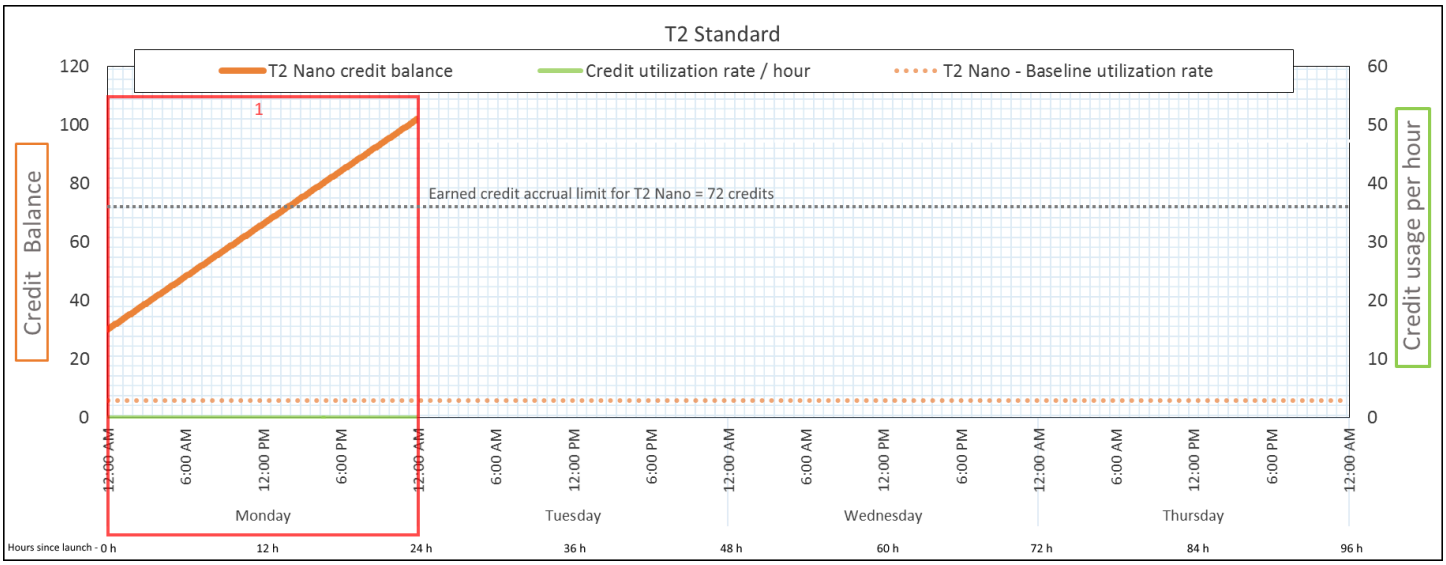
t2.nano インスタンスは、起動時に 30 起動クレジットを獲得し、24 時間ごとに 72 クレジットを獲得します。このクレジット残高の制限は 72 獲得クレジットです。起動クレジットはこの制限に対してカウントされません。制限に到達すると、獲得された新しいクレジットはすべて破棄されます。獲得および蓄積できるクレジット数の詳細については、クレジットの表を参照してください。の制限事項の詳細については、起動クレジット制限を参照してください。

T2 スタンダードインスタンスを起動し、すぐに使用することができます。または、T2 スタンダードインスタンスを起動し、何日間かアイドル状態にしてから、そこでアプリケーションを実行する場合があります。インスタンスを使用中か、アイドル状態であるかによって、クレジットが消費されるか、あるいは蓄積されるかが決まります。インスタンスが起動後に 24 時間アイドル状態のままである場合、クレジット残高は制限を超えているように表示されます。これは、蓄積された獲得クレジットと蓄積された起動クレジットの両方が残高に反映されるためです。ただし、CPU を使用すると、起動クレジットが最初に使用されます。その後、この制限は、蓄積できる獲得クレジットの最大数を常に反映します。

この例では、起動後に 24 時間アイドル状態のままとなるインスタンスについて説明します。また、96 時間にわたる 7 つの期間で、クレジットが獲得、蓄積、消費、破棄される率と、各期間の終了時点でのクレジット残高の値を示します。

期間 1: 1 ~ 24 時間

グラフの 0 時において、T2 インスタンスは standard として起動され、すぐに 30 クレジットを獲得します。実行状態の間はクレジットを獲得します。このインスタンスは起動時からアイドル状態になり (CPU 使用率は 0%)、クレジットは消費されません。すべての未消費のクレジットはクレジット残高に蓄積されます。起動後約 14 時間で、クレジット残高は 72 (30 起動クレジット + 42 獲得クレジット) となり、これはインスタンスが 24 時間に獲得できる数と同等になります。起動後 24 時間で、クレジット残高は 72 クレジットを超えます。これは、未消費の起動クレジットがクレジット残高に蓄積されるためです (クレジット残高は— 102 クレジット: 30 起動クレジット + 72 獲得クレジット)。



クレジットの消費率	24 時間あたり 0 クレジット (0% の CPU 使用率)
クレジットの獲得率	24 時間あたり 72 クレジット
クレジットの破棄率	24 時間あたり 0 クレジット
クレジット残高	102 クレジット (30 起動クレジット + 72 獲得クレジット)

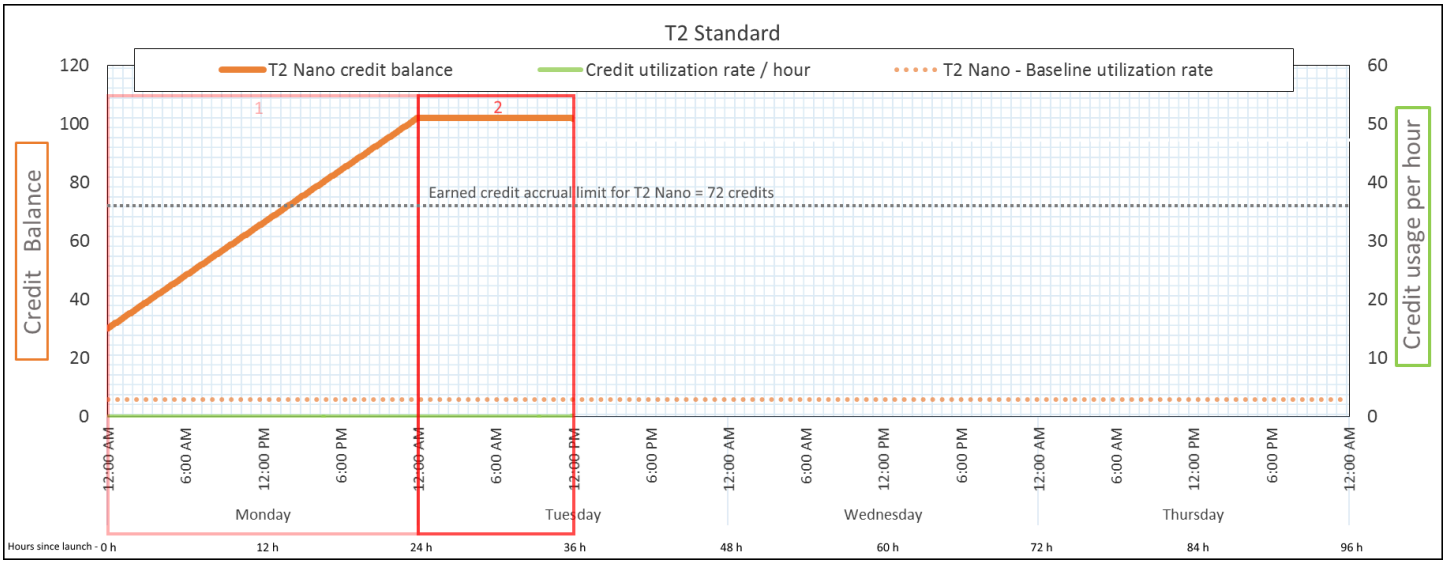
結論

起動後に CPU の使用がない場合、インスタンスは 24 時間に獲得できるよりも多くのクレジットを蓄積します (30 起動クレジット + 72 獲得クレジット = 102 クレジット)。

実際のシナリオでは、EC2 インスタンスは起動中および実行中にも少数のクレジットを消費します。それにより、残高がこの例の理論的な最大値に達することを防ぎます。

期間 2: 25 ~ 36– 時間

次の 12 時間中に、インスタンスは引き続きアイドル状態のままとなり、クレジットを獲得しますが、クレジット残高は増えません。102 クレジット (30 起動クレジット + 72 獲得クレジット) で頭打ちとなります。クレジット残高は制限である 72 の蓄積された獲得クレジットに達したため、新しく獲得されたクレジットは破棄されます。



クレジットの消費率	24 時間あたり 0 クレジット (0% の CPU 使用率)
クレジットの獲得率	24 時間あたり 72 クレジット (1 時間で 3 クレジット)
クレジットの破棄率	24 時間あたり 72 クレジット (100% のクレジット獲得率)
クレジット残高	102 クレジット (30 起動クレジット + 72 獲得クレジット)— 残高は変更されません

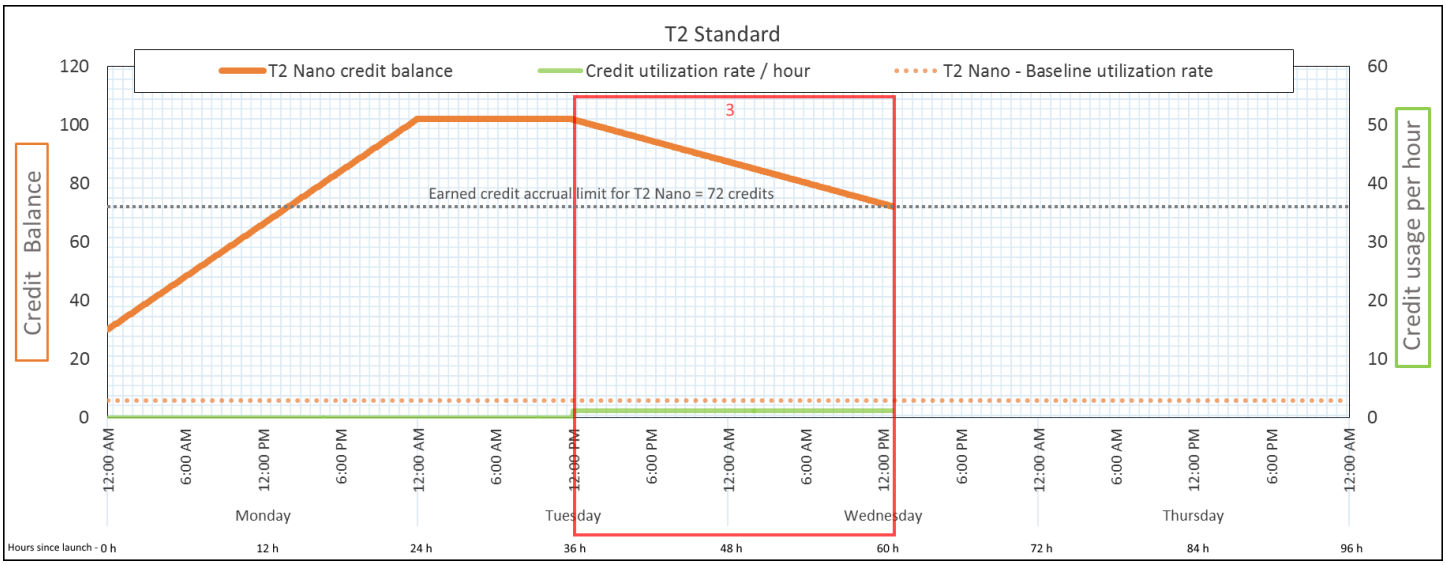
結論

インスタンスはクレジットを継続して獲得しますが、クレジット残高が制限に達した場合、獲得クレジットはそれ以上蓄積されません。制限に到達すると、新しく獲得されたクレジットはすべて破棄さ

れます。起動クレジットは、クレジット残高制限に対してカウントされません。残高に蓄積された起動クレジットが含まれている場合、残高は制限を超えているように表示されます。

期間 3: 37 ~ 61 時間

次の 25 時間で、インスタンスは 2% の CPU を使用します。これには 30 クレジットが必要です。同じ期間に 75 クレジットを取得しますが、クレジット残高は減ります。残高が減るのは、蓄積された起動クレジットが最初に消費されますが、クレジット残高が既に 72 獲得クレジットという制限に達しているため、新しく獲得されたクレジットは破棄されるためです。



クレジットの消費率	24 時間あたり 28.8 クレジット (1 時間ごとに 1.2 クレジット、2% の CPU 使用率、40% のクレジット獲得率)— 25 時間以上で 30 クレジット
クレジットの獲得率	24 時間あたり 72 クレジット
クレジットの破棄率	24 時間あたり 72 クレジット (100% のクレジット獲得率)
クレジット残高	72 クレジット (30 起動クレジットが消費され、72 獲得クレジットが未使用のまま)

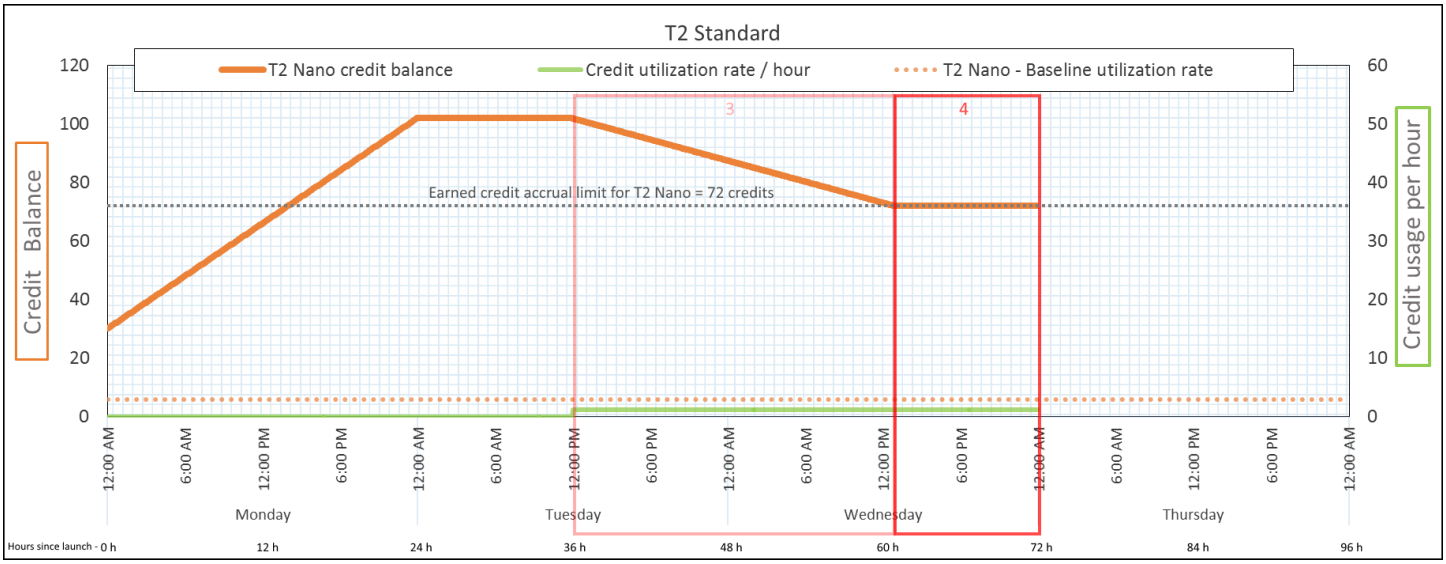
結論

インスタンスは、獲得クレジットを消費する前に、起動クレジットを最初に消費します。起動クレジットは、クレジット制限に対してカウントされません。起動クレジットが消費された後で、24 時間に獲得できる数よりも残高が高くなることはありません。さらに、インスタンスの実行中は、それ以上クレジットを獲得することはできません。

期間 4: 62 ~ 72 時間

次の 11 時間で、インスタンスは 2% の CPU を使用します。これには 13.2 クレジットが必要です。これは前の期間の CPU 使用率と同じですが、残高は減りません。72 クレジットのままです。

残高が減らないのは、クレジットの獲得率がクレジットの消費率よりも高いためです。また、インスタンスは 13.2 クレジットを消費する時間に、33 クレジットを獲得します。ただし、残高の制限は 72 クレジットであるため、制限を超えて獲得されたクレジットは破棄されます。残高は 72 で頭打ちとなります。これは期間 2 の 102 クレジットという頭打ちとは異なりますが、蓄積された起動クレジットがないためです。



クレジットの消費率	24 時間あたり 28.8 クレジット (1 時間ごとに 1.2 クレジット、2% の CPU 使用率、40% のクレジット獲得率)— 11 時間以上で 13.2 クレジット
クレジットの獲得率	24 時間あたり 72 クレジット
クレジットの破棄率	24 時間あたり 43.2 クレジット (60% のクレジット獲得率)

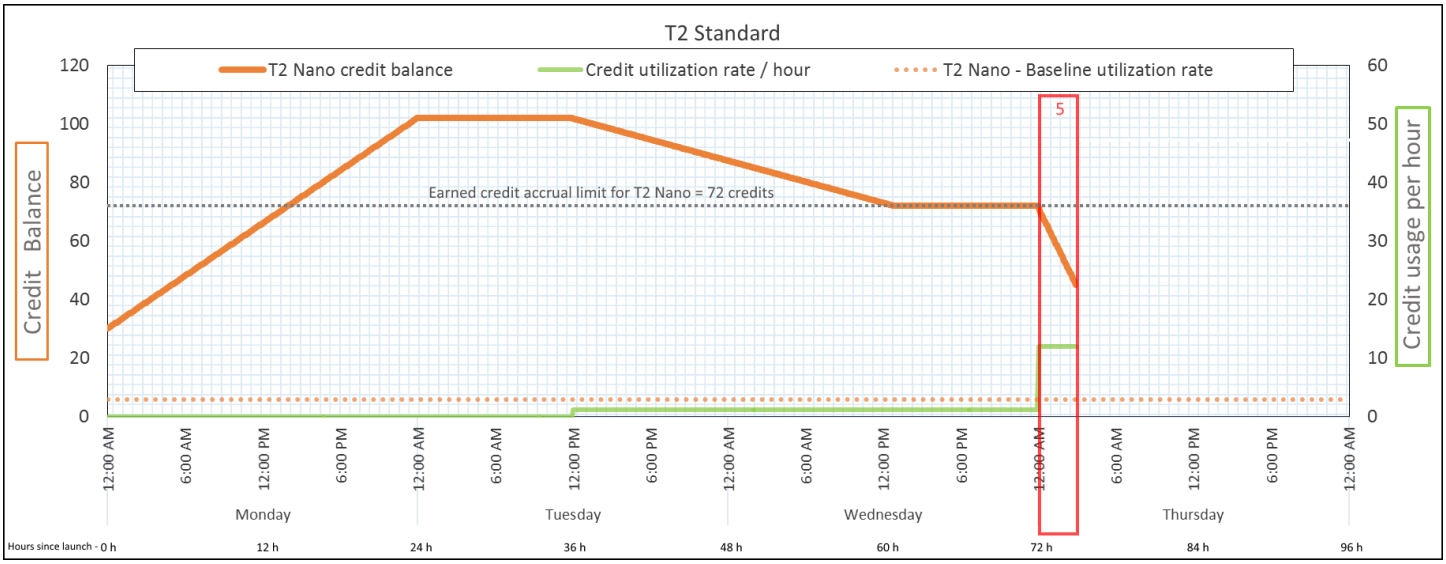
クレジット残高	72 クレジット (0 起動クレジット、72 獲得クレジット)— 残高は上限

結論

起動クレジットの消費後、クレジット残高の制限はインスタンスが 24 時間に獲得できるクレジット数によって決まります。インスタンスが、消費するよりも多くのクレジットを獲得した場合、制限を超えて新しく獲得されたクレジットは破棄されます。

期間 5: 73 ~ 75 時間

次の 3 時間で、インスタンスは 20% の CPU 使用率でバーストします。これには 36 クレジットが必要です。インスタンスは同じ 3 時間で 9 クレジットを獲得します。これにより、実際の残高は 27 クレジット減ります。3 時間の最後に、クレジット残高は 45 の蓄積された獲得クレジットとなります。



クレジットの消費率	24 時間あたり 288 クレジット (1 時間ごとに 12 クレジット、20% の CPU 使用率、400% のクレジット獲得率)— 3 時間以上で 36 クレジット
クレジットの獲得率	24 時間あたり 72 クレジット (3 時間で 9 クレジット)
クレジットの破棄率	24 時間あたり 0 クレジット

クレジット残高	45 クレジット (前の残高 (72) - 消費したクレジット (36) + 獲得したクレジット (9))— 残高は 24 時間あたり 216 クレジットの率で減少 (消費率 288/24 + 獲得率 72/24 = 残高減少率 216/24)

結論

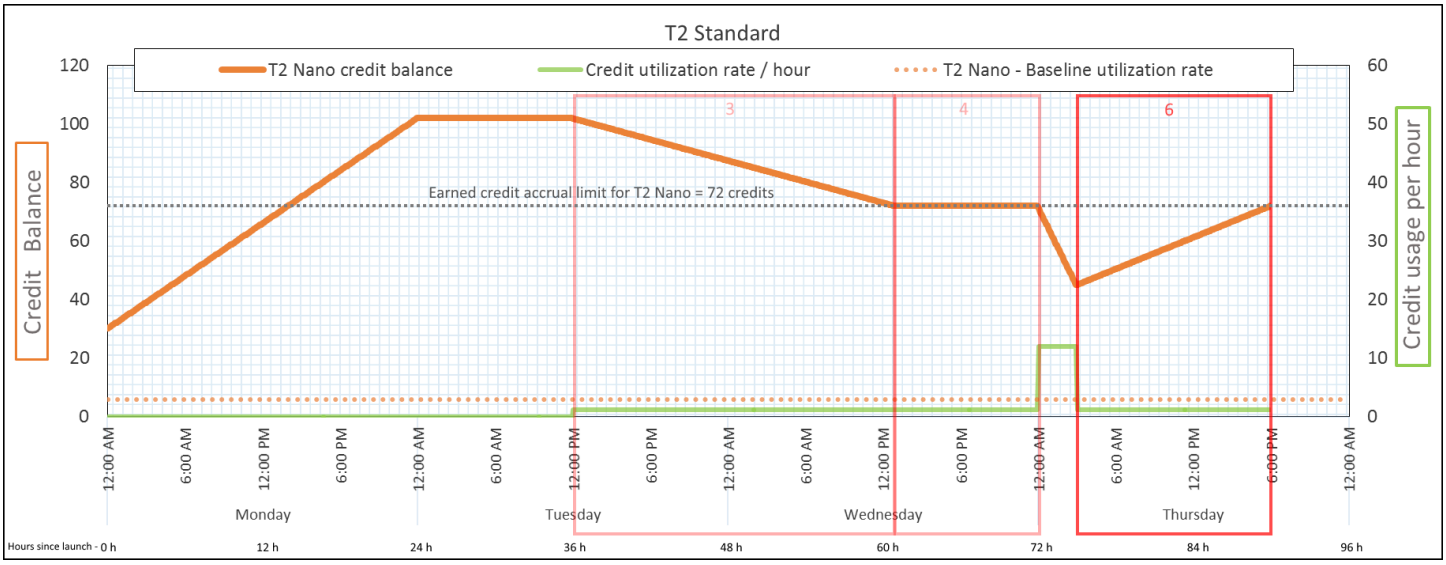
インスタンスが、獲得するよりも多くのクレジットを消費する場合、クレジット残高は減ります。

期間 6: 76 ~ 90 時間

次の 15 時間で、インスタンスは 2% の CPU を使用します。これには 18 クレジットが必要です。これは、期間 3 および 4 と同じ CPU 使用率です。ただし、期間 3 で残高が減り、期間 4 で頭打ちになりましたが、この期間の残高は増えます。

期間 3 で、蓄積された起動クレジットが消費されました。また、クレジットの制限を超えて獲得されたクレジットは破棄され、クレジット残高が減りました。期間 4 で、インスタンスが消費したクレジットは獲得したクレジットよりも少なくなりました。限度額を超えて獲得したクレジットはすべて破棄されたため、残高は最大 72 クレジットとなりました。

この期間に蓄積された起動クレジットはなく、残高の蓄積された獲得クレジットの数は制限を下回っています。獲得クレジットは破棄されません。さらに、インスタンスは消費するよりも多くのクレジットを獲得し、クレジット残高が増えます。



クレジットの消費率	24 時間あたり 28.8 クレジット (1 時間ごとに 1.2 クレジット、2% の CPU 使用率、40% のクレジット獲得率)— 15 時間以上で 18 クレジット
クレジットの獲得率	24 時間あたり 72 クレジット (15 時間で 45 クレジット)
クレジットの破棄率	24 時間あたり 0 クレジット
クレジット残高	72 クレジット (残高は 24 時間あたり 43.2 クレジットの率で増えます— 変更率 = 消費率 28.8/24 + 獲得率 72/24)

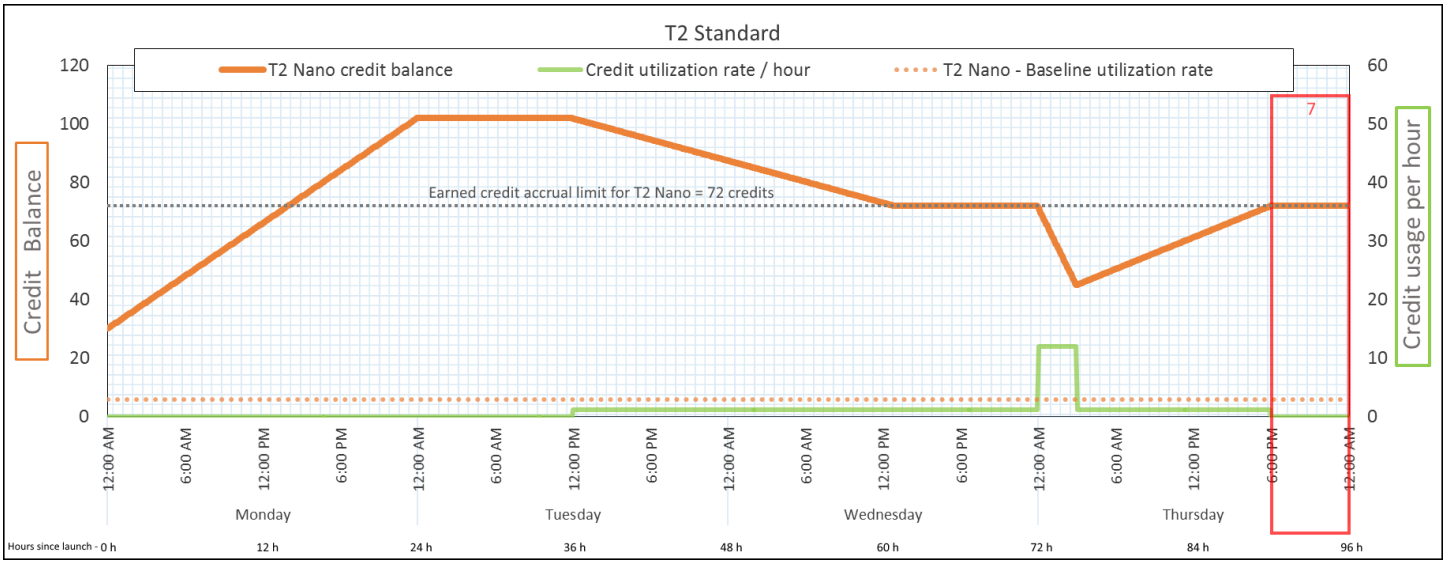
結論

インスタンスが、獲得するよりも少ないクレジットを消費する場合、クレジット残高は増えます。

期間 7: 91 ~ 96 時間

次の 6 時間は、インスタンスはアイドル状態になり (—CPU 使用率は 0%—)、クレジットは消費されません。これは、期間 2 の —CPU 使用率と同じですが、残高は 102 クレジットで頭打ちになりません。インスタンスのクレジット残高の制限である 72 クレジットで頭打ちになります。

期間 2 で、クレジット残高には蓄積された 30 起動クレジットが含まれます。起動クレジットは期間 3 で消費されました。実行中のインスタンスはそれ以上起動クレジットを取得できません。クレジット残高の制限に達すると、制限を超えて獲得されたクレジットは破棄されます。



クレジットの消費率	24 時間あたり 0 クレジット (0% の CPU 使用率)
クレジットの獲得率	24 時間あたり 72 クレジット
クレジットの破棄率	24 時間あたり 72 クレジット (100% のクレジット獲得率)
クレジット残高	72 クレジット (0 起動クレジット、72 獲得クレジット)

結論

インスタンスはクレジットを継続して獲得しますが、クレジット残高の制限に達した場合、獲得クレジットはそれ以上蓄積されません。制限に到達すると、新しく獲得されたクレジットはすべて破棄されます。クレジット残高の制限は、インスタンスが 24 時間に獲得できるクレジット数によって決まります。クレジット残高の制限の詳細については、クレジットの表を参照してください。

バーストパフォーマンスインスタンスを設定する

バーストパフォーマンスインスタンス (T インスタンス) の起動、モニタリング、および変更の手順は似ています。主な違いは、起動時のデフォルトのクレジット指定です。

各 T インスタンスファミリーには、以下のデフォルトクレジット仕様が設定されています。

- T4g、T3a、および T3 インスタンスを `unlimited` で起動する

- 専有ホストで standard として T3 インスタンスを起動のみ行えます。
- T2 インスタンスを standard として起動

アカウントのクレジット指定のデフォルト設定を変更できます。

タスク

- 起動時にクレジット仕様を設定する
- クレジット仕様を unlimited に設定するように Auto Scaling グループを設定する
- バーストパフォーマンスインスタンスのクレジット仕様を管理する
- アカウントのデフォルトのクレジット仕様を管理する

起動時にクレジット仕様を設定する

T インスタンスは、unlimited または standard のクレジット仕様で起動できます。

次の手順では、EC2 コンソールまたは AWS CLI を使用する方法について説明します。Auto Scaling グループの使用の詳細については、「クレジット仕様を unlimited に設定するように Auto Scaling グループを設定する」を参照してください。

Console

起動時にインスタンスのクレジット仕様を設定するには

1. インスタンスを起動するための手順に従います。
2. [Instance type] (インスタンスタイプ) で、T インスタンスタイプを選択してください。
3. [Advanced Details] (高度な詳細) を展開します。[クレジット仕様] で、クレジット仕様を選択します。
4. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択してください。

AWS CLI

起動時にインスタンスのクレジット仕様を設定するには

--credit-specification オプションで run-instances コマンドを使用します。

```
--credit-specification CpuCredits=unlimited
```

PowerShell

起動時にインスタンスのクレジット仕様を設定するには

`-CreditSpecification_CpuCredit` パラメータで New-EC2Instance コマンドレットを使用します。

```
-CreditSpecification_CpuCredit unlimited
```

クレジット仕様を unlimited に設定するように Auto Scaling グループを設定する

T インスタンスが起動または開始する際、優れたブートストラップエクスペリエンスには CPU クレジットが必要です。Auto Scaling グループを使用してインスタンスを起動する場合は、インスタンスを `unlimited` として設定することをお勧めします。そうする場合、インスタンスは Auto Scaling グループによって自動的に起動または再開されたときに余剰クレジットを使用します。余剰クレジットを使用することで、パフォーマンスの制限を防ぐことができます。

起動テンプレートの作成

インスタンスを Auto Scaling グループで `unlimited` として起動するには、起動に起動テンプレートを使用する必要があります。起動設定では、インスタンスを `unlimited` として起動することはサポートされていません。

Console

クレジット仕様を設定する起動テンプレートを作成するには

1. 「Amazon EC2 Auto Scaling ユーザーガイド」の「詳細設定を使用して起動テンプレートを作成する」を参照してください。
2. [Launch template contents] ((テンプレート内容の起動) の [Instance type] (インスタンスタイプ) で、インスタンスサイズを選択してください。
3. インスタンスを Auto Scaling グループで `unlimited` として起動するには、[Advanced details] (高度な詳細) の [Credit specification] (クレジット指定) で [Unlimited] (無制限) を選択してください。
4. 起動テンプレートパラメータの定義が終了したら、[Create launch template] (起動テンプレートの作成) を選択してください。

AWS CLI

クレジット仕様を設定する起動テンプレートを作成するには

create-launch-template コマンドを使用します。

```
aws ec2 create-launch-template \
    --launch-template-name my-launch-template \
    --version-description FirstVersion \
    --launch-template-data CreditSpecification={CpuCredits=unlimited}
```

PowerShell

クレジット仕様を設定する起動テンプレートを作成するには

New-EC2LaunchTemplate コマンドレットを使用します。起動テンプレートデータのクレジット仕様を次のように定義します。

```
$creditSpec = New-Object Amazon.EC2.Model.CreditSpecificationRequest
$creditSpec.CpuCredits = "unlimited"
$launchTemplateData = New-Object Amazon.EC2.Model.RequestLaunchTemplateData
$launchTemplateData.CreditSpecification = $creditSpec
```

起動テンプレートによる Auto Scaling グループの関連付け

起動テンプレートを Auto Scaling グループに関連付けるには、起動テンプレートを使用して Auto Scaling グループを作成するか、または既存の Auto Scaling グループに起動テンプレートを追加します。

Console

起動テンプレートを使用して Auto Scaling グループを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 画面の上部のナビゲーションバーで、起動テンプレートを作成したときに使用したのと同じリージョンを選択してください。
3. ナビゲーションペインで [Auto Scaling グループ]、[Auto Scaling グループの作成] の順に選択してください。

4. [Launch Template (起動テンプレート)] で、起動テンプレートを選択し、[次のステップ] を選択してください。
5. Auto Scaling グループ用のフィールドに入力してください。[Review page (確認ページ)] で設定の確認を終えたら、[Create Auto Scaling group (Auto Scaling グループの作成)] を選択してください。詳細については、『Amazon EC2 Auto Scaling ユーザーガイド』の起動テンプレートを使用した Auto Scaling グループの作成 を参照してください。

既存の Auto Scaling グループに起動テンプレートを追加するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 画面の上部のナビゲーションバーで、起動テンプレートを作成したときに使用したのと同じリージョンを選択してください。
3. ナビゲーションペインで、[Auto Scaling Groups] をクリックしてください。
4. Auto Scaling グループの一覧から Auto Scaling グループを選択し、[アクション]、[編集] の順に選択してください。
5. [Details (詳細)] タブの [Launch Template (起動テンプレート)] で起動テンプレートを選択して、[Save (保存)] を選択してください。

AWS CLI

起動テンプレートを使用して Auto Scaling グループを作成するには

create-auto-scaling-グループコマンドを使用して、 パラメ `--launch-template` タを指定します。

既存の Auto Scaling グループに起動テンプレートを追加するには

update-auto-scaling-group コマンドを使用して、`--launch-template` パラメータを指定します。

PowerShell

起動テンプレートを使用して Auto Scaling グループを作成するには

New-ASAutoScalingGroup コマンドレットを使用して、`-LaunchTemplate_LaunchTemplateId` または `-LaunchTemplate_LaunchTemplateName`パラメータを指定します。

既存の Auto Scaling グループに起動テンプレートを追加するには

Update-ASAutoScalingGroup コマンドレットを使用して、-LaunchTemplate_LaunchTemplateId または -LaunchTemplate_LaunchTemplateNameパラメータを指定します。

バーストパフォーマンスインスタンスのクレジット仕様を管理する

実行中または停止中の T インスタンスのクレジット指定は、unlimited と standard の間でいつでも切り替えることができます。

unlimited モードでは、インスタンスが余剰クレジットを使用することがあり、追加料金が発生する可能性があることに注意してください。詳細については「余剰クレジットにより料金が発生することがある」を参照してください。

Console

インスタンスのクレジット仕様を管理するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左のナビゲーションペインの [インスタンス] を選択してください。
3. (オプション) インスタンスを選択します。[詳細] タブで、[クレジット仕様] を見つけます。この値は unlimited または standard のどちらかです。
4. (オプション) 複数のインスタンスのクレジット仕様を同時に変更するには、それらをすべて選択します。
5. [Actions (アクション)]、[Instance settings (インスタンス設定)]、[Change credit specification (クレジット指定の変更)] の順に選択してください。このオプションは、T インスタンスを選択した場合にのみ有効になります。
6. [無制限モード] では、各インスタンス ID の横にあるチェックボックスをオンまたはオフにします。

AWS CLI

インスタンスのクレジット仕様を取得するには

describe-instance-credit-specifications コマンドを使用します。インスタンス ID を指定しない場合、クレジット仕様が unlimited のすべてのインスタンスが返されます。出力には、unlimited のクレジット仕様を持つ以前に設定されたインスタンスも含まれます。例えば、T3 インスタンスを M4 インスタンスにサイズ変更し、unlimited に設定している場合、Amazon EC2 は M4 インスタンスを返します。

```
aws ec2 describe-instance-credit-specifications \
    --instance-id i-1234567890abcdef0 \
    --query InstanceCreditSpecifications[].CpuCredits \
    --output text
```

以下は出力の例です。

```
unlimited
```

インスタンスのクレジット仕様を設定するには

modify-instance-credit-specification コマンドを使用します。

```
aws ec2 modify-instance-credit-specification \
    --region us-east-1 \
    --instance-credit-specification
 "InstanceId=i-1234567890abcdef0,CpuCredits=unlimited"
```

PowerShell

インスタンスのクレジット仕様を取得するには

Get-EC2CreditSpecification コマンドレットを使用します。

```
(Get-EC2CreditSpecification `
    -InstanceId i-1234567890abcdef0).CpuCredits
```

以下は出力の例です。

```
unlimited
```

インスタンスのクレジット仕様を設定するには

Edit-EC2InstanceCreditSpecification コマンドレットを使用します。

```
Edit-EC2InstanceCreditSpecification `
    -Region us-east-1 `
    -InstanceCreditSpecification @({InstanceId="i-1234567890abcdef0"
 CpuCredits="unlimited"})
```

アカウントのデフォルトのクレジット仕様を管理する

各 T インスタンスファミリーには、デフォルトクレジット仕様が付属しています。各 AWS リージョンのアカウントレベルで、T インスタンスファミリーごとにデフォルトのクレジット仕様を変更できます。デフォルトのクレジット仕様の有効な値は、`unlimited` および `standard` です。

EC2 コンソールのインスタンス起動ウィザードを使用してインスタンスを起動している場合、アカウントレベルのデフォルトのクレジット指定は、お客様により設定されたクレジット指定の値により上書きされます。AWS CLI を使用してインスタンスを起動する場合には、アカウント内のすべての新しい T インスタンスは、デフォルトのクレジット指定を使用して起動されます。既存の実行中または停止中のインスタンスのクレジット指定には影響しません。

考慮事項

インスタンスファミリーのデフォルトのクレジット指定は、継続した 5 分間に 1 回のみ変更でき、継続した 24 時間中に最大 4 回変更できます。

Console

デフォルトのクレジット仕様を管理するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. AWS リージョン を変更するにはページの右上隅にあるリージョンセレクターを使用します。
3. ナビゲーションペインで、ダッシュボードを選択してください。
4. [アカウント属性] カードの [設定] で、[デフォルトのクレジット指定] を選択します。
5. [管理] をクリックしてください。
6. インスタンスファミリーごとに、[無制限] または [標準] を選択した上で、[更新] をクリックしてください。

AWS CLI

デフォルトのクレジット仕様を取得するには

get-default-credit-specification コマンドを使用します。

```
aws ec2 get-default-credit-specification \
    --region us-east-1 \
    --instance-family t2 \
```

```
    --query InstanceFamilyCreditSpecifications[].CpuCredits \
    --output text
```

以下は出力の例です。

```
standard
```

デフォルトのクレジット仕様を設定するには

modify-default-credit-specification コマンドを使用します。次の例では、値を unlimited に設定します。

```
aws ec2 modify-default-credit-specification \
    --region us-east-1 \
    --instance-family t2 \
    --cpu-credits unlimited
```

PowerShell

デフォルトのクレジット仕様を取得するには

Get-EC2DefaultCreditSpecification コマンドレットを使用します。

```
(Get-EC2DefaultCreditSpecification `
    -Region us-east-1 `
    -InstanceFamily t2).CpuCredits
```

以下は出力の例です。

```
standard
```

デフォルトのクレジット仕様を設定するには

Edit-EC2DefaultCreditSpecification コマンドレットを使用します。次の例では、値を unlimited に設定します。

```
Edit-EC2DefaultCreditSpecification `
    -Region us-east-1 `
    -InstanceFamily t2 `
    -CpuCredit unlimited
```

バーストインスタンスの CPU クレジットをモニタリングする

EC2 はメトリクスを Amazon CloudWatch に送信します。CPU クレジットのメトリクスは、CloudWatch コンソールの Amazon EC2 インスタンスごとのメトリクスで確認するか、AWS CLI を使用して各インスタンスのメトリクスを一覧表示することで確認できます。詳細については「インスタンスに対して利用可能な CloudWatch メトリクス」を参照してください。

内容

- バーストパフォーマンスインスタンスの追加 CloudWatch メトリクス
- CPU クレジット使用状況の計算

バーストパフォーマンスインスタンスの追加 CloudWatch メトリクス

バーストパフォーマンスインスタンスには、以下の追加の CloudWatch メトリクスがあり、5 分ごとに更新されます。

- `CPUCreditUsage` – 測定期間に消費された CPU クレジットの数。
- `CPUCreditBalance` – インスタンスが蓄積する CPU クレジット数。CPU がバーストし、CPU クレジットが獲得するよりも速い速度で使用される際に、このバランスは枯渇します。
- `CPUSurplusCreditBalance` – `CPUCreditBalance` 値がゼロになったときに CPU 使用率を保持するために使用される余剰 CPU クレジットの数。
- `CPUSurplusCreditsCharged` – 24 時間で獲得できる CPU クレジットの最大数を越えた、追加料金が発生する分の余剰 CPU クレジットの数。

最後の 2 つのメトリクスは `unlimited` として設定されたインスタンスにのみ適用されます。

バーストパフォーマンスインスタンスの CloudWatch メトリクスの説明を次の表に示します。詳細については「インスタンスに対して利用可能な CloudWatch メトリクス」を参照してください。

メトリクス	説明
`CPUCreditUsage`	CPU 使用率に関してインスタンスで消費される CPU クレジットの数。1 つの CPU クレジットは、1 個の vCPU が 100% の使用率で 1 分間実行されること、または、vCPU、使用率、時間の同等の組み合わせ (例えば、1 個の vCPU が 50% の使用率で 2 分間実行されるか、2 個の vCPU が 25% の使用率で 2 分間実行される) に相当します。

メトリクス	説明
	CPU クレジットメトリクスは、5 分間隔でのみ利用可能です。5 分を超える期間を指定する場合は、Average 統計の代わりに Sum 統計を使用します。 単位: クレジット (vCPU 分)
CPUCreditBalance	インスタンスが起動または開始後に蓄積した獲得 CPU クレジットの数。T2 スタンダードの場合、CPUCreditBalance には蓄積された起動クレジットの数も含まれます。 クレジットは獲得後にクレジット残高に蓄積され、消費されるとクレジット残高から削除されます。クレジット残高にはインスタンスサイズによって決まる上限があります。制限に到達すると、獲得された新しいクレジットはすべて破棄されます。T2 スタンダードの場合、起動クレジットは制限に対してカウントされません。 CPUCreditBalance のクレジットはインスタンスがそのベースライン CPU 使用率を超えてバーストするために消費できます。 インスタンスが実行中の場合、CPUCreditBalance のクレジットは期限切れになりません。T4g、T3a、または T3 インスタンスが停止すると、CPUCreditBalance 値は 7 日間保持されます。その後、蓄積されたすべてのクレジットが失われます。T2 インスタンスが停止すると、CPUCreditBalance 値は保持されず、蓄積されたすべてのクレジットが失われます。 CPU クレジットメトリクスは、5 分間隔でのみ利用可能です。 単位: クレジット (vCPU 分)

メトリクス	説明
CPUSurplusCreditBalance	CPUCreditBalance 値がゼロの場合に unlimited インスタンスによって消費された余剰クレジットの数。 CPUSurplusCreditBalance 値は獲得した CPU クレジットによって支払われます。余剰クレジットの数が、24 時間にインスタンスが獲得できるクレジットの最大数を超えている場合、最大数を超えて消費された余剰クレジットに対しては料金が発生します。 単位: クレジット (vCPU 分)
CPUSurplusCreditsCharged	獲得 CPU クレジットにより支払われないために追加料金が発生した、消費された余剰クレジットの数。 消費された余剰クレジットは以下のいずれかの状況に当てはまると料金が発生します。 • 消費された余剰クレジットが、インスタンスが 24 時間に獲得できる最大クレジット数を超えている。最大数を越えて消費された余剰クレジットは時間の最後に課金されます。 • インスタンスが停止または終了した。 • インスタンスは unlimited から standard に切り替わります。 単位: クレジット (vCPU 分)

CPU クレジット使用状況の計算

インスタンスの CPU クレジット使用状況は、前述の表で説明したインスタンス CloudWatch メトリクスを使用して計算されます。

Amazon EC2 は、メトリクスを 5 分ごとに CloudWatch に送信します。前のメトリクス値の参照はいつでも、5 分前に送信された、直前のメトリクス値を意味します。

スタンダードインスタンスの CPU クレジット使用状況の計算

- CPU クレジット残高は、CPU 利用率がベースラインを下回り、前の 5 分間に消費したクレジットが獲得したクレジットより少なかった場合に増加します。
- CPU クレジット残高は、CPU 利用率がベースラインを上回り、前の 5 分間に消費したクレジットが獲得したクレジットよりも多かった場合に減少します。

数学的に、これは次の式で表されます。

Example

```
CPUCreditBalance = prior CPUCreditBalance + [Credits earned per hour * (5/60) -
 CPUCreditUsage]
```

インスタンスのサイズは、インスタンスが 1 時間あたりに獲得できるクレジットの数と、クレジット残高に蓄積できる獲得クレジットの数を決定します。1 時間あたりに獲得するクレジット数と、各インスタンスサイズのクレジット残高制限については、クレジットの表を参照してください。

例

この例では、t3.nano インスタンスを使用します。インスタンスの CPUCreditBalance 値を計算するには、前述の式を次のように使用します。

- CPUCreditBalance – 計算する現在のクレジット残高。
- prior CPUCreditBalance – 5 分前のクレジット残高。この例では、インスタンスは 2 クレジットを蓄積しています。
- Credits earned per hour – t3.nano インスタンスは 1 時間あたり 6 クレジット獲得します。
- 5/60 – CloudWatch メトリクスのパブリッシュ間の 5 分間隔を表します。1 時間あたりに獲得するクレジットに 60 分の 5 (5 分) を掛けて、インスタンスが過去 5 分間に獲得したクレジット数を求めます。t3.nano インスタンスは、5 分ごとに 0.5 クレジットを獲得します。
- CPUCreditUsage – インスタンスが過去 5 分間に消費したクレジット数。この例では、インスタンスは過去 5 分間に 1 クレジットを消費しました。

これらの値を使用して、CPUCreditBalance の値を計算できます。

Example

```
CPUCreditBalance = 2 + [0.5 - 1] = 1.5
```

無制限インスタンスの CPU クレジット使用状況の計算

バーストパフォーマンスインスタンスがベースラインを超えてバーストする必要がある場合、余剰クレジットを消費する前に、蓄積されたクレジットが常に消費されます。蓄積した CPU クレジット残高を使いきると、必要な期間だけ余剰クレジットを使用してバーストできます。CPU 利用率がベースラインを下回った場合、インスタンスが獲得クレジットを蓄積する前に常に余剰クレジットが支払われます。

この 5 分間に発生するアクティビティを反映するため、次の式では Adjusted balance という用語を使用します。CPUCreditBalance および CPUSurplusCreditBalance の CloudWatch メトリクスの値に達するため、この値を使用します。

Example

```
Adjusted balance = [prior CPUCreditBalance - prior CPUSurplusCreditBalance] + [Credits
 earned per hour * (5/60) - CPUCreditUsage]
```

0 に対する Adjusted balance の値は、インスタンスが獲得したすべてのクレジットがバーストに消費され、余剰クレジットは消費されなかったことを示します。結果として、CPUCreditBalance と CPUSurplusCreditBalance は 0 に設定されます。

Adjusted balance の正の値は、インスタンスが獲得クレジットを蓄積し、前の余剰クレジットが (ある場合) 支払われたことを示します。結果として、Adjusted balance の値は CPUCreditBalance に割り当てられ、CPUSurplusCreditBalance は 0に設定されます。インスタンスサイズは、蓄積可能な最大クレジット数を決定します。

Example

```
CPUCreditBalance = min [max earned credit balance, Adjusted balance]
CPUSurplusCreditBalance = 0
```

Adjusted balance の負の値は、インスタンスが蓄積したすべての獲得クレジットに加えて、余剰クレジットもバーストに消費されたことを示します。結果として、Adjusted balance の値は CPUSurplusCreditBalance と CPUCreditBalance に割り当てられ、0に設定されます。繰り返しになりますが、インスタンスサイズは、蓄積可能な最大クレジット数を決定します。

Example

```
CPUSurplusCreditBalance = min [max earned credit balance, -Adjusted balance]
CPUCreditBalance = 0
```

消費される余剰クレジットがインスタンスに蓄積可能な最大クレジットを越えた場合、余剰クレジット残高は前述の式に示すように最大に設定されます。残りの余剰クレジットは CPUSurplusCreditsCharged メトリクスで示すように課金されます。

Example

```
CPUSurplusCreditsCharged = max [-Adjusted balance - max earned credit balance, 0]
```

最後に、CPUSurplusCreditBalance により追跡された余剰クレジットもインスタンスの終了時に課金されます。インスタンスを unlimited から standard に切り替えると、残りの CPUSurplusCreditBalance も課金されます。

GPU インスタンスによるパフォーマンスアクセラレーション

GPU ベースのインスタンスでは、数千のコンピューティングコアを持つ NVIDIA GPU にアクセスできます。これらのインスタンスを使用すると、CUDA または Open Computing Language (OpenCL) パラレルコンピューティングフレームワークを活用することにより、サイエンス、エンジニアリング、およびレンダリングアプリケーションを高速化できます。また、ゲームストリーミング、3D アプリケーションストリーミング、およびその他のグラフィックスワークロードを含む、グラフィックアプリケーションにも使用できます。

GPU ベースのインスタンスをアクティブ化または最適化するには、次のように適切なドライバーをインストールする必要があります。

- NVIDIA GPU がアタッチされたインスタンス (P3 または G4dn インスタンスなど) に NVIDIA ドライバーをインストールするには、「NVIDIA ドライバー」を参照してください。
- AMD GPU がアタッチされたインスタンス (G4ad インスタンスなど) に AMD ドライバーをインストールするには、「AMD ドライバー」を参照してください。

内容

- Amazon EC2 GPU ベースのインスタンスで NVIDIA GRID 仮想アプリケーションをアクティブ化する
- Amazon EC2 インスタンスの GPU 設定を最適化する

- G4ad Linux インスタンスでデュアル 4K ディスプレイをセットアップする
- GPU アクセラレーテッドインスタンスの使用を開始する

Amazon EC2 GPU ベースのインスタンスで NVIDIA GRID 仮想アプリケーションをアクティブ化する

NVIDIA GPU を搭載した GPU ベースのインスタンスで GRID 仮想アプリケーションをアクティブ化するには (NVIDIA GRID 仮想ワークステーションはデフォルトで有効になっています)、ドライバーの製品タイプを定義する必要があります。使用するプロセスは、インスタンスのオペレーティングシステムによって異なります。

Linux インスタンス

Linux インスタンスで GRID 仮想アプリケーションを有効にするには

1. 提供されるテンプレートファイルから /etc/nvidia/gridd.conf ファイルを作成します。

```
[ec2-user ~]$ sudo cp /etc/nvidia/gridd.conf.template /etc/nvidia/gridd.conf
```

2. お好きなテキストエディタで /etc/nvidia/gridd.conf ファイルを開きます。
3. FeatureType 行を見つけ、それを 0 と等しくなるように設定します。次に、IgnoreSP=TRUE の行を追加します。

```
FeatureType=0 IgnoreSP=TRUE
```

4. ファイルを保存して終了します。
5. インスタンスを再起動し、新しい設定を取得します。

```
[ec2-user ~]$ sudo reboot
```

Windows インスタンス

Windows インスタンスで GRID 仮想アプリケーションを有効にするには

1. regedit.exe を実行して、レジストリエディタを開きます。
2. HKEY_LOCAL_MACHINE\SOFTWARE\NVIDIA Corporation\Global\GridLicensing に移動します。

3. 右側のペインでコンテキスト (右クリック) メニューを開き、[New]、[DWORD] の順に選択してください。
4. [Name (名前)] にFeatureTypeと入力し、Enter キーを押します。
5. [FeatureType] でコンテキスト (右クリック) メニューを開き、[Modify] を選択してください。
6. 値のデータには、NVIDIA GRID 仮想アプリケーションで 0 を入力し、[OK] を選択してください。
7. 右側のペインでコンテキスト (右クリック) メニューを開き、[New]、[DWORD] の順に選択してください。
8. [名前] では、[IgnoreSP] を入力し、次に Enter と入力してください。
9. [IgnoreSP] でコンテキスト (右クリック) メニューを開き、[Modify] を選択してください。
10. [Value data] に1と入力し、[OK] を選択してください。
11. レジストリエディタを閉じます。

Amazon EC2 インスタンスの GPU 設定を最適化する

NVIDIA GPU インスタンスで最大のパフォーマンスを実現するためには、いくつかの最適化方法の中から GPU 設定を選択できます。これらのインスタンスタイプの一部では、NVIDIA ドライバーは自動ブースト機能を使用しますが、これは GPU クロック速度に左右されます。自動ブーストを無効にし、GPU クロック速度を最大周波数に設定することで、安定して GPU インスタンスで最大パフォーマンスを実現できます。

Linux での GPU 設定の最適化

1. GPU 設定を永続的になるように設定します。このコマンドの実行には数分かかることがあります。

```
[ec2-user ~]$ sudo nvidia-persistenced
```

2. [G3 インスタンスのみ] インスタンス上のすべての GPU の自動ブースト機能を無効にします。

```
[ec2-user ~]$ sudo nvidia-smi --auto-boost-default=0
```

3. すべての GPU クロックを最大周波数に設定します。次のコマンドで指定されるメモリとグラフィッククロック速度を使用します。

一部のバージョンの NVIDIA ドライバーでは、アプリケーションのクロック速度の設定をサポートしていないため、"Setting applications clocks is not supported for GPU..." エラーが表示されますが、無視できます。

- G3 インスタンス:

```
[ec2-user ~]$ sudo nvidia-smi -ac 2505,1177
```

- G4dn インスタンス:

```
[ec2-user ~]$ sudo nvidia-smi -ac 5001,1590
```

- G5 インスタンス:

```
[ec2-user ~]$ sudo nvidia-smi -ac 6250,1710
```

- G6、G6f、Gr6、およびGr6f インスタンス:

```
[ec2-user ~]$ sudo nvidia-smi -ac 6251,2040
```

- G6e インスタンス:

```
[ec2-user ~]$ sudo nvidia-smi -ac 9001,2520
```

- G7e インスタンス:

```
[ec2-user ~]$ sudo nvidia-smi -ac 12481,2430
```

- G7 インスタンス:

```
[ec2-user ~]$ sudo nvidia-smi -ac 12501,2415
```

- P3 および P3dn インスタンス:

```
[ec2-user ~]$ sudo nvidia-smi -ac 877,1530
```

- P4d インスタンス:

```
[ec2-user ~]$ sudo nvidia-smi -ac 1215,1410
```

- P4de インスタンス:

```
[ec2-user ~]$ sudo nvidia-smi -ac 1593,1410
```

- P5 インスタンス:

```
[ec2-user ~]$ sudo nvidia-smi -ac 2619,1980
```

- P5e および P5en インスタンス:

```
[ec2-user ~]$ sudo nvidia-smi -ac 3201,1980
```

- P6-B200 インスタンス:

```
[ec2-user ~]$ sudo nvidia-smi -ac 3996,1965
```

- P6-B300 インスタンス:

```
[ec2-user ~]$ sudo nvidia-smi -ac 3996,2032
```

Windows での GPU 設定の最適化

1. PowerShell ウィンドウを開き、NVIDIA のインストールフォルダに移動します。

```
PS C:\> cd "C:\Windows\System32\DriverStore\FileRepository\nvgridsw_aws.inf_*\"
```

2. [G3 インスタンスのみ] インスタンス上のすべての GPU の自動ブースト機能を無効にします。

```
PS C:\> .\nvidia-smi --auto-boost-default=0
```

3. すべての GPU クロックを最大周波数に設定します。次のコマンドで指定されるメモリとグラフィッククロック速度を使用します。

   一部のバージョンの NVIDIA ドライバーでは、アプリケーションのクロック速度の設定をサポートしていないため、"Setting applications clocks is not supported for GPU..." エラーが表示されますが、無視できます。

   - G3 インスタンス:

```
PS C:\> .\nvidia-smi -ac "2505,1177"
```

- G4dn インスタンス:

```
PS C:\> .\nvidia-smi -ac "5001,1590"
```

- G5 インスタンス:

```
PS C:\> .\nvidia-smi -ac "6250,1710"
```

- G6、G6f、Gr6、およびGr6f インスタンス:

```
PS C:\> .\nvidia-smi -ac "6251,2040"
```

- G6e インスタンス:

```
PS C:\> .\nvidia-smi -ac "9001,2520"
```

- P3 および P3dn インスタンス:

```
PS C:\> .\nvidia-smi -ac "877,1530"
```

G4ad Linux インスタンスでデュアル 4K ディスプレイをセットアップする

G4ad インスタンスを起動すると、デュアル 4K ディスプレイをセットアップできます。

AMD ドライバーをインストールしてデュアル画面を設定するには

1. Linux インスタンスに接続して、デュアル 4K (2x4k) 向けにターゲットとする GPU の PCI Bus アドレスを取得します。

   ```
   lspci -vv | grep -i amd
   ```

   以下のような出力結果が取得できます。

   ```
   00:1e.0 Display controller: Advanced Micro Devices, Inc. [*AMD*/ATI] Device 7362 (rev
    c3)
   Subsystem: Advanced Micro Devices, Inc. [AMD/ATI] Device 0a34
   ```

2. 上記の出力では、PCI バスアドレスは 00:1e.0 であることに注意してください。`/etc/modprobe.d/amdgpu.conf` という名前のファイルを作成して追加します。

```
options amdgpu virtual_display=0000:00:1e.0,2
```

3. Linux に AMD ドライバーをインストールするには、「EC2 インスタンス用の AMD ドライバー」を参照してください。AMD GPU ドライバーが既にインストールされている場合は、dkms を通じて amdgpu カーネルモジュールを再構築する必要があります。
4. 以下の xorg.conf ファイルを使用して、デュアル (2x4K) スクリーントポロジを定義し、`/etc/X11/xorg.conf`: のファイルに保存します。

```
~$ cat /etc/X11/xorg.conf
Section "ServerLayout"
    Identifier     "Layout0"
    Screen          0 "Screen0"
    Screen        1 "Screen1"
    InputDevice     "Keyboard0" "CoreKeyboard"
    InputDevice     "Mouse0" "CorePointer"
    Option          "Xinerama" "1"
EndSection
Section "Files"
    ModulePath "/opt/amdgpu/lib64/xorg/modules/drivers"
    ModulePath "/opt/amdgpu/lib/xorg/modules"
    ModulePath "/opt/amdgpu-pro/lib/xorg/modules/extensions"
    ModulePath "/opt/amdgpu-pro/lib64/xorg/modules/extensions"
    ModulePath "/usr/lib64/xorg/modules"
    ModulePath "/usr/lib/xorg/modules"
EndSection
Section "InputDevice"
    # generated from default
    Identifier     "Mouse0"
    Driver         "mouse"
    Option         "Protocol" "auto"
    Option         "Device" "/dev/psaux"
    Option         "Emulate3Buttons" "no"
    Option         "ZAxisMapping" "4 5"
EndSection
Section "InputDevice"
    # generated from default
    Identifier     "Keyboard0"
    Driver         "kbd"
EndSection

Section "Monitor"
```

```
    Identifier     "Virtual"
    VendorName     "Unknown"
    ModelName      "Unknown"
    Option         "Primary" "true"
EndSection

Section "Monitor"
    Identifier     "Virtual-1"
    VendorName     "Unknown"
    ModelName      "Unknown"
    Option         "RightOf" "Virtual"
EndSection

Section "Device"
    Identifier     "Device0"
    Driver         "amdgpu"
    VendorName     "AMD"
    BoardName      "Radeon MxGPU V520"
    BusID          "PCI:0:30:0"
EndSection

Section "Device"
    Identifier     "Device1"
    Driver         "amdgpu"
    VendorName     "AMD"
    BoardName      "Radeon MxGPU V520"
    BusID          "PCI:0:30:0"
EndSection

Section "Extensions"
    Option         "DPMS" "Disable"
EndSection

Section "Screen"
    Identifier     "Screen0"
    Device         "Device0"
    Monitor        "Virtual"
    DefaultDepth   24
    Option         "AllowEmptyInitialConfiguration" "True"
    SubSection "Display"
        Virtual    3840 2160
        Depth      32
    EndSubSection
EndSection
```

```
Section "Screen"
    Identifier     "Screen1"
    Device         "Device1"
    Monitor        "Virtual"
    DefaultDepth   24
    Option         "AllowEmptyInitialConfiguration" "True"
    SubSection "Display"
        Virtual    3840 2160
        Depth      32
    EndSubSection
EndSection
```

5. インタラクティブデスクトップの設定手順に従って、DCV を設定します。
6. DCV の設定が完了したら、再起動します。
7. ドライバーが機能していることを確認します。

```
dmesg | grep amdgpu
```

レスポンスは次のようになります。

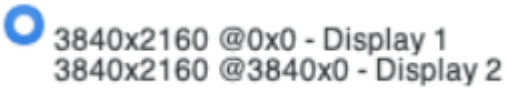
```
Initialized amdgpu
```

8. DISPLAY=:0 xrandr -q の出力で、2 つの仮想ディスプレイが接続されていることを確認できます。

```
~$ DISPLAY=:0 xrandr -q
Screen 0: minimum 320 x 200, current 3840 x 1080, maximum 16384 x 16384
Virtual connected primary 1920x1080+0+0 (normal left inverted right x axis y axis)
 0mm x 0mm
 4096x3112  60.00
 3656x2664  59.99
 4096x2160  60.00
 3840x2160  60.00
 1920x1200  59.95
 1920x1080  60.00
 1600x1200  59.95
 1680x1050  60.00
 1400x1050  60.00
 1280x1024  59.95
 1440x900 59.99
 1280x960 59.99
```

```
 1280x854 59.95
 1280x800 59.96
 1280x720 59.97
 1152x768 59.95
 1024x768 60.00 59.95
 800x600  60.32 59.96 56.25
 848x480  60.00 59.94
 720x480  59.94
 640x480  59.94 59.94
Virtual-1 connected 1920x1080+1920+0 (normal left inverted right x axis y axis) 0mm x
 0mm
 4096x3112  60.00
 3656x2664  59.99
 4096x2160  60.00
 3840x2160  60.00
 1920x1200  59.95
 1920x1080  60.00
 1600x1200  59.95
 1680x1050  60.00
 1400x1050  60.00
 1280x1024  59.95
 1440x900 59.99
 1280x960 59.99
 1280x854 59.95
 1280x800 59.96
 1280x720 59.97
 1152x768 59.95
 1024x768 60.00 59.95
 800x600  60.32 59.96 56.25
 848x480  60.00 59.94
 720x480  59.94
640x480  59.94 59.94
```

9. DCV に接続するときは、解像度を 2x4K に変更し、デュアルモニタのサポートが DCV によって登録されていることを確認します。



Linux のインタラクティブデスクトップをセットアップする

Linux インスタンスに AMD GPU ドライバーがインストールされ、amdgpu が使用されていることを確認したら、インタラクティブデスクトップマネージャーをインストールできます。最高の互換性とパフォーマンスを得るには、MATE デスクトップ環境をお勧めします。

前提条件

テキストエディタを開き、次のファイルを`xorg.conf`という名前のファイルとして保存します。このファイルはインスタンスで必要になります。

```
Section "ServerLayout"
Identifier     "Layout0"
Screen          0 "Screen0"
InputDevice     "Keyboard0" "CoreKeyboard"
InputDevice     "Mouse0" "CorePointer"
EndSection
Section "Files"
ModulePath "/opt/amdgpu/lib64/xorg/modules/drivers"
ModulePath "/opt/amdgpu/lib/xorg/modules"
ModulePath "/opt/amdgpu-pro/lib/xorg/modules/extensions"
ModulePath "/opt/amdgpu-pro/lib64/xorg/modules/extensions"
ModulePath "/usr/lib64/xorg/modules"
ModulePath "/usr/lib/xorg/modules"
EndSection
Section "InputDevice"
# generated from default
Identifier     "Mouse0"
Driver         "mouse"
Option         "Protocol" "auto"
Option         "Device" "/dev/psaux"
Option         "Emulate3Buttons" "no"
Option         "ZAxisMapping" "4 5"
EndSection
Section "InputDevice"
# generated from default
Identifier     "Keyboard0"
Driver         "kbd"
EndSection
Section "Monitor"
Identifier     "Monitor0"
VendorName     "Unknown"
ModelName      "Unknown"
```

```
EndSection
Section "Device"
Identifier     "Device0"
Driver         "amdgpu"
VendorName     "AMD"
BoardName      "Radeon MxGPU V520"
BusID          "PCI:0:30:0"
EndSection
Section "Extensions"
Option         "DPMS" "Disable"
EndSection
Section "Screen"
Identifier     "Screen0"
Device         "Device0"
Monitor        "Monitor0"
DefaultDepth   24
Option         "AllowEmptyInitialConfiguration" "True"
SubSection "Display"
    Virtual    3840 2160
    Depth      32
EndSubSection
EndSection
```

Amazon Linux 2 でインタラクティブデスクトップをセットアップするには

1. EPEL リポジトリをインストールします。

```
[ec2-user ~]$ sudo amazon-linux-extras install epel -y
```

2. MATE デスクトップをインストールします。

```
[ec2-user ~]$ sudo amazon-linux-extras install mate-desktop1.x -y
[ec2-user ~]$ sudo yum groupinstall "MATE Desktop" -y
[ec2-user ~]$ sudo systemctl disable firewalld
```

3. xorg.conf ファイルを /etc/X11/xorg.conf にコピーします。
4. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

5. (オプション) Amazon DCV サーバーをインストールして、Amazon DCV を高パフォーマンスの表示プロトコルとして使用してから、お好みのクライアントを使用して Amazon DCV セッションに接続します。

Ubuntu でインタラクティブデスクトップをセットアップするには

1. MATE デスクトップをインストールします。

```
$ sudo apt install xorg-dev ubuntu-mate-desktop -y
$ sudo apt purge ifupdown -y
```

2. `xorg.conf` ファイルを `/etc/X11/xorg.conf` にコピーします。
3. インスタンスを再起動します。

```
$ sudo reboot
```

4. 適切なバージョンの Ubuntu 用の AMF エンコーダーをインストールします。

```
$ sudo apt install ./amdgpu-pro-20.20-*/amf-amdgpu-pro_20.20-*_amd64.deb
```

5. (オプション) Amazon DCV サーバーをインストールして、Amazon DCV を高パフォーマンスの表示プロトコルとして使用してから、お好みのクライアントを使用して Amazon DCV セッションに接続します。
6. DCV のインストール後、DCV ユーザーに動画のアクセス権限を付与します。

```
$ sudo usermod -aG video dcv
```

CentOS で対話型デスクトップをセットアップするには

1. EPEL リポジトリをインストールします。

```
$ sudo yum update -y
$ sudo yum install epel-release -y
```

2. MATE デスクトップをインストールします。

```
$ sudo yum groupinstall "MATE Desktop" -y
$ sudo systemctl disable firewalld
```

3. `xorg.conf` ファイルを `/etc/X11/xorg.conf` にコピーします。
4. インスタンスを再起動します。

```
$ sudo reboot
```

5. (オプション) Amazon DCV サーバーをインストールして、Amazon DCV を高パフォーマンスの表示プロトコルとして使用してから、お好みのクライアントを使用して Amazon DCV セッションに接続します。

GPU アクセラレーテッドインスタンスの使用を開始する

以下のリストに示すような最新世代の GPU アクセラレーテッドインスタンスタイプは、深層学習やハイパフォーマンスコンピューティング (HPC) アプリケーションに最良のパフォーマンス機能を提供します。機能の詳細については、インスタンスタイプのリンクを選択してください。

- P6 ファミリー
- P6 ファミリー
- P5 ファミリー

高速化インスタンスタイプのインスタンスタイプ仕様に関する完全なリストについては、「Amazon EC2 Instance Types」リファレンスで「Accelerated computing」を参照してください。

ソフトウェア設定

最新世代の GPU アクセラレーテッドインスタンスタイプの使用を開始する最も簡単な方法は、すべての必須ソフトウェアで事前設定されている AWS Deep Learning AMI からインスタンスを起動することです。GPU アクセラレーテッドインスタンスタイプで使用する最新の AWS Deep Learning AMIs については、「AWS Deep Learning AMIs デベロッパーガイド」の「P6 でサポートされる DLAMI」を参照してください。

深層学習または HPC アプリケーションをホストするインスタンスを起動するためにカスタム AMI を構築する必要がある場合は、ベースイメージ上に以下の最小ソフトウェアバージョンをインストールすることをお勧めします。

インスタンスタイプ	NVIDIA ドライバー	CUDA	NVIDIA GDRCopy	EFA インストーラ	NCCL	EFA K8s [1]
G7	595	13.2	2.5.1	1.48.0	2.29.7-1	0.5.10
G7e	575	12.9	2.5	1.45.0	2.28.3	0.5.10
P5	530	12.1	2.3	1.24.1	2.18.3	0.4.4
P5.4xlarge	530	12.1	2.3	1.43.1 [2]	2.18.3	0.4.4
P5e	550	12.1	2.3	1.24.1	2.18.3	0.5.5
P5en	550	12.1	2.3	1.24.1	2.18.3	0.5.6
P6-B200	570	12.8	2.5	1.41.0	2.26.2-1	0.5.10
P6e-GB200	570	12.8	2.5	1.41.0	2.26.2-1	0.5.10
P6-B300	580	13.0	2.5	1.44.0	2.28.3	0.5.10

[1]「EFA K8」の列には、`aws-efa-k8s-device-plugin` に推奨される最低限のバージョンが含まれています。

[2] GPU-to-GPU 通信で Elastic Fabric Adapter (EFA) および NVIDIA Collective Communications Library (NCCL) が使用される場合、`P5.4xlarge` インスタンスに影響する互換性の問題が発生します。この問題を軽減するには `FI_HMEM_DISABLE_P2P` 環境変数を 1 に設定し、EFA バージョン 1.43.1 以降がインストールされていることを確認してください。

> **Note**
>
> バージョン 1.41.0 の EFA インストーラを使用している場合、`aws-ofi-nccl plugin` が付属しています。以前のバージョンの EFA インストーラでは、`aws-ofi-nccl plugin` バージョン `1.7.2-aws` 以降を使用してください。

また、より深い C ステートを使用しないようにインスタンスを設定することをお勧めします。詳細については、Amazon Linux 2 ユーザーガイドの「より深い C ステートの制限による高パフォーマン

スと低レイテンシー」を参照してください。最新の AWS Deep Learning Base GPU AMI は、より深い C ステートを使用しないように事前設定されています。

ネットワークと Elastic Fabric Adapter (EFA) 設定については、「複数のネットワークカードを使用して Amazon EC2 インスタンスのネットワーク帯域幅を最大化する」を参照してください。

Amazon EC2 Mac インスタンス

EC2 Mac インスタンスは、iPhone、iPad、Mac、Vision Pro、Apple Watch、Apple TV、Safari などの Apple プラットフォーム用アプリケーションの開発、構築、テスト、署名に最適です。Mac インスタンスには、SSH または Apple Remote Desktop (ARD) を使用して接続できます。

> **Note**
>
> [unit of billing] (請求の単位) は [dedicated host] (専有ホスト) です。そのホストで実行されているインスタンスには追加料金はかかりません。

Amazon EC2 Mac インスタンスは macOS オペレーティングシステムをネイティブでサポートしています。

- EC2 x86 Mac インスタンス (`mac1.metal`)は 3.2 GHz Intel 第 8 世代 (Coffee Lake) Core i7 プロセッサ、6 個の物理コアと 12 個の論理コア、32 GiB のメモリを搭載した 2018 Mac mini ハードウェアに構築されています。
- EC2 M1 Mac インスタンス (`mac2.metal`) は、Apple シリコン M1 プロセッサ、8 個の CPU コア、8 個の GPU コア、16 GiB のメモリ、16 コアの Apple Neural Engine を搭載した 2020 Mac mini ハードウェアに構築されています。
- EC2 M1 Ultra Mac インスタンス (`mac2-m1ultra.metal`) は、Apple シリコン M1 Ultra プロセッサ、20 個の CPU コア、64 個の GPU コア、128 GiB のメモリ、32 コアの Apple Neural Engine を搭載した 2022 Mac Studio ハードウェアに構築されています。
- EC2 M2 Mac インスタンス (`mac2-m2.metal`) は、Apple シリコン M2 プロセッサ、8 個の CPU コア、10 個の GPU コア、24 GiB のメモリ、16 コアの Apple Neural Engine を搭載した 2023 Mac Studio ハードウェアに構築されています。
- EC2 M2 Pro Mac インスタンス (`mac2-m2pro.metal`) は、Apple シリコン M2 Pro プロセッサ、12 個の CPU コア、19 個の GPU コア、32 GiB のメモリ、16 コアの Apple Neural Engine を搭載した 2023 Mac mini ハードウェアに構築されています。

- EC2 M4 Mac インスタンス (`mac-m4.metal`) は、Apple シリコン M4 プロセッサ、10 個の CPU コア、10 個の GPU コア、24 GiB のメモリ、16 コアの Apple Neural Engine を搭載した 2024 Mac mini ハードウェアに構築されています。
- EC2 M4 Pro Mac インスタンス (`mac-m4pro.metal`) は、Apple シリコン M4 Pro プロセッサ、14 個の CPU コア、20 個の GPU コア、48 GiB のメモリ、16 コアの Apple Neural Engine を搭載した 2024 Mac mini ハードウェアに構築されています。

Amazon EC2 Mac 専有ホストは、専有ホストの自動復旧と再起動ベースのホストメンテナンスをサポートします。

内容

- 考慮事項
- インスタンスの準備状況
- EC2 macOS AMI
- EC2 macOS Init
- macOS 用の Amazon EC2 System Monitor
- 関連リソース
- AWS マネジメントコンソール または AWS CLI を使用して Mac インスタンスを起動する
- SSH または GUI を使用して Mac インスタンスに接続する
- Amazon EC2 Mac インスタンス上のオペレーティングシステムとソフトウェアの更新
- Mac インスタンスの EBS ボリュームのサイズを増やす
- Amazon EC2 Mac インスタンスを停止または終了する
- Amazon EC2 Mac インスタンスのシステム整合性保護を構成する
- Amazon EC2 Mac 専有ホストでサポートされている macOS バージョンを特定する
- macOS AMI の通知へのサブスクライブ
- AWS Systems Manager Parameter Store API を使用して macOS AMI ID を取得する
- Amazon EC2 macOS AMI リリースノート

考慮事項

Mac インスタンスには、次の考慮事項が適用されます。

- Mac インスタンスは、Dedicated Hosts のベアメタルインスタンスとしてのみ使用でき、Dedicated Host をリリースできる前の最低割り当て期間は 24 時間です。Dedicated Host ごとに 1 つの Mac インスタンスを起動できます。Dedicated Host は、AWS アカウント、 AWS 組織内の 組織単位、あるいは AWS 組織全体と共有することができます。
- Mac インスタンスはさまざまな AWS リージョン で使用できます。それぞれの AWS リージョンで利用可能な Mac インスタンスのリストについては、「リージョン別の Amazon EC2 インスタンスタイプ」を参照してください。
- Mac インスタンスは オンデマンドインスタンス としてのみ使用できます。スポットインスタンスまたは リザーブドインスタンス では使用できません。Savings Plan を購入すると、Mac インスタンスでコストを節約できます。
- 異なる Mac インスタンスタイプと特定の macOS Amazon マシンイメージ (AMI) の互換性はさまざまです。詳細については、Amazon EC2 macOS AMI リリースノートを参照してください。
- EBS ホットプラグはサポートされています。
- AWS は、Apple ハードウェアの内部 SSD を管理またはサポートしません。代わりに、Amazon EBS ボリュームを使用することが強く推奨されます。EBS ボリュームは、他の EC2 インスタンスと同等の伸縮性、可用性、耐久性などの利点を Mac インスタンスにもたらします。
- 最適なパフォーマンスを得るには、Mac インスタンスで 10,000 IOPS と 400 MiB/秒のスループットの Amazon EBS ボリュームを使用することをお勧めします。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームの種類」を参照してください。
- Mac インスタンスは、Amazon EC2 Auto Scaling をサポートしています。
- x86 Mac インスタンスでは、自動ソフトウェア更新は無効化されています。インスタンスを本番環境に置く前に、更新を適用し、インスタンスでテストすることをお勧めします。詳細については、Amazon EC2 Mac インスタンス上のオペレーティングシステムとソフトウェアの更新を参照してください。
- Mac インスタンスを停止または終了すると、Dedicated Host でスクラブワークフローが実行されます。詳細については、Amazon EC2 Mac インスタンスを停止または終了するを参照してください。
-

> **Important**
>
> Apple Intelligence 機能は、外部ボリュームから Mac ハードウェアを起動する場合は使用できません。EC2 Mac インスタンスはデフォルトで外部 EBS ボリュームから起動するため、Apple Intelligence 機能はサポートされていません。

- **Warning**

  FileVault は使用しません。FileVault を有効にすると、パーティションがロックされているため、ホストの起動に失敗します。データ暗号化が必要な場合は、Amazon EBS 暗号化を使用して、ブートの問題やパフォーマンスへの影響を回避します。Amazon EBS の暗号化では、暗号化オペレーションはインスタンスをホストするサーバー上で実行されるため、インスタンスとそれにアタッチされた EBS ストレージ間に保管中のデータと転送中のデータの両方のセキュリティが確保されます。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS 暗号化」を参照してください。

インスタンスの準備状況

Mac インスタンスを起動したら、インスタンスに接続する前に、インスタンスの準備が整うまで待機する必要があります。x86 Mac インスタンスまたは Apple Silicon Mac インスタンスを含む AWS 提供の AMI の場合、起動時間は約 6 分から 20 分の範囲です。選択した Amazon EBS のボリュームサイズ、ユーザーデータへの追加スクリプトの導入、カスタム macOS AMI への追加ロードソフトウェアなどにより、起動時間が長くなる場合があります。

以下のような小さなシェルスクリプトを使用すれば、describe-instance-status API をポーリングし、インスタンスが接続できる状態になったかどうかを確認できます。次のコマンドで、インスタンス ID の例を独自の インスタンス ID に置き換えます。

```
for i in $(seq 1 200); do aws ec2 describe-instance-status --instance-
ids=i-1234567890abcdef0 \
    --query='InstanceStatuses[0].InstanceStatus.Status'; sleep 5; done;
```

EC2 macOS AMI

Amazon EC2 macOS は、Amazon EC2 Mac インスタンスで実行されるデベロッパーワークロードに対して、安定性、安全性、高パフォーマンスの環境を実現するように設計されています。EC2 macOS AMI には、起動設定ツールや一般的な AWS ライブラリやツールなど、AWS との統合を容易にするパッケージが含まれています。

EC2 macOS AMI の詳細については、「Amazon EC2 macOS AMI リリースノート」を参照してください。

AWS は、更新済みの EC2 macOS AMI を定期的に提供します。これには、AWS が所有するパッケージの更新と、完全にテストされた macOS の最新バージョンが含まれます。さらに、AWS は更

新された AMI を提供し、さらに完全にテストおよび検証できるとすぐに最新のマイナーバージョンアップデートやメジャーバージョンアップデートを提供します。Mac インスタンスのデータやカスタマイズ情報を保持する必要がない場合は、現在の AMI を使用して新しいインスタンスを起動してから、前のインスタンスを終了することで、最新の更新プログラムを取得できます。また、Mac インスタンスに適用するアップデートを選択できます。

macOS AMI の通知をサブスクライブする方法については、macOS AMI の通知へのサブスクライブを参照してください。

EC2 macOS Init

EC2 macOS Init は、起動時に EC2 Mac インスタンスを初期化するために使用します。優先順位グループを使用して、タスクの論理グループを同時に実行します。

launchd plist ファイルは `/Library/LaunchDaemons/com.amazon.ec2.macos-init.plist` です。EC2 macOS Init 用のファイルは、`/usr/local/aws/ec2-macos-init` にあります。

詳細については、https://github.com/aws/ec2-macos-init を参照してください。

macOS 用の Amazon EC2 System Monitor

macOS 用の Amazon EC2 System Monitor は、Amazon CloudWatch で CPU 使用率メトリクスを使用できるようにします。このメトリクスは、カスタムシリアルデバイス経由で 1 分間隔で CloudWatch に送信されます。このエージェントを有効または無効にするには、次の手順に従います。このエージェントは、デフォルトでは有効になっています。

```
sudo setup-ec2monitoring [enable | disable]
```

> **Note**
>
> macOS 用の Amazon EC2 System Monitor は、現在、Apple Silicon Mac インスタンスではサポートされていません。

関連リソース

料金については、「料金」を参照してください。

Mac インスタンスの詳細については、「Amazon EC2 Mac インスタンス」を参照してください。

Mac インスタンスのハードウェア仕様とネットワークパフォーマンスについて詳しくは、「汎用インスタンス」を参照してください。

AWS マネジメントコンソール または AWS CLI を使用して Mac インスタンスを起動する

EC2 Mac インスタンスには専有ホストが必要です。まず、アカウントにホストを割り当ててから、そのホストにインスタンスを作成する必要があります。

AWS マネジメントコンソール または AWS CLI を使用して Mac インスタンスを起動できます。

コンソールを使用した Mac インスタンスの起動

Mac インスタンスを Dedicated Host 上で起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 以下のように専有ホストを割り当てます。
    a. ナビゲーションペインで [Dedicated Hosts] を選択してください。
    b. [Dedicated Host の割り当て] を選択し、次の操作を行います。
        i. [インスタンスファミリー] では Mac のインスタンスファミリーを選択します。リストにインスタンスファミリーが表示されない場合、それは、現在選択されているリージョンではサポートされていません。
        ii. [インスタンスタイプ] では、選択したインスタンスファミリーに基づいてインスタンスタイプを選択します。
        iii. [アベイラビリティーゾーン] で、専有ホストのアベイラビリティーゾーンを選択してください。
        iv. [Quantity] (数量) は 1 のままにします。
        v. [割り当て] を選択してください。
3. 次のように、ホストにインスタンスを作成します。
    a. 作成した Dedicated Host を選択し、次の操作を実行します。
        i. [アクション] (アクション)、[Launch instance(s) onto host] (ホストにインスタンスを作成) の順に選択してください。
        ii. [Application and OS Images (Amazon Machine Image)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で、macOS AMI を選択してください。

iii. [インスタンスタイプ] では Mac のインスタンスタイプを選択します。

iv. [高度な詳細] で、[テナンシー]、[テナンシーホスト指定]、および [テナンシーのホスト ID] が、作成した専有ホストに基づいて事前設定されていることを確認します。必要に応じて [Tenancy affinity] (テナンシーのアフィニティ) を更新します。

v. ウィザードを完了し、必要に応じて EBS ボリューム、セキュリティグループ、およびキーペアを指定します。

vi. [Summary] (サマリー) パネルで、[Launch instance] (インスタンスの起動) を選択してください。

b. 確認ページは、インスタンスが起動中であることを通知します。[View all instances] (すべてのインスタンスの表示) を選択して確認ページを閉じ、コンソールに戻ります。インスタンスの初期状態は `pending` です。インスタンスの状態が `running` に変わると、ステータスチェックに合格すると、インスタンスは準備完了になります。

AWS CLI を使用した Mac インスタンスの起動

専有ホストを割り当てる

以下の allocate-hosts コマンドを使用して、お使いの Mac インスタンスに専有ホストを割り当て、`instance-type` を有効な Mac インスタンスタイプに置き換え、`region` と `availability-zone` はお使いの環境に適したものに置き換えます。

```
aws ec2 allocate-hosts --region us-east-1 --instance-type mac1.metal --availability-
zone us-east-1b --auto-placement "on" --quantity 1
```

ホスト上でインスタンスを起動する

次の run-instances コマンドを使用して Mac インスタンスを起動します。ここでも、`instance-type` を 有効な Mac インスタンスタイプに、`region` と `availability-zone` を以前使用していたものにそれぞれ置き換えます。

```
aws ec2 run-instances --region us-east-1 --instance-type mac1.metal --placement
 Tenancy=host --image-id ami_id --key-name my-key-pair
```

インスタンスの初期状態は `pending` です。インスタンスの状態が `running` に変わると、ステータスチェックに合格すると、インスタンスは準備完了になります。インスタンスのステータス情報を表示するには、次の describe-instance-status コマンドを使用します。

```
aws ec2 describe-instance-status --instance-ids i-017f8354e2dc69c4f
```

ステータスチェックに合格した実行中のインスタンスの出力例を次に示します。

```
{
    "InstanceStatuses": [
        {
            "AvailabilityZone": "us-east-1b",
            "InstanceId": "i-017f8354e2dc69c4f",
            "InstanceState": {
                "Code": 16,
                "Name": "running"
            },
            "InstanceStatus": {
                "Details": [
                    {
                        "Name": "reachability",
                        "Status": "passed"
                    }
                ],
                "Status": "ok"
            },
            "SystemStatus": {
                "Details": [
                    {
                        "Name": "reachability",
                        "Status": "passed"
                    }
                ],
                "Status": "ok"
            }
        }
    ]
}
```

SSH または GUI を使用して Mac インスタンスに接続する

SSH またはグラフィカルユーザーインターフェイス (GUI) を使用して Mac インスタンスに接続できます。

複数のユーザーが同時に OS にアクセスできます。ポート 5900 の画面共有サービスが組み込まれているため、通常、1:1 user:GUI セッションがあります。macOS 内で SSH を使用すると、sshd_config ファイルの [最大セッション] 制限まで、複数のセッションがサポートされます。

SSH を使用したインスタンスへの接続

Amazon EC2 Mac インスタンスは、デフォルトではリモートルート SSH を許可しません。ec2-user アカウントは、SSH を使用してリモートでログインするように設定されています。ec2-user アカウントにも sudo 権限があります。インスタンスに接続したら、他のユーザーを追加できます。

SSH を使用したインスタンスへの接続をサポートするには、SSH アクセスを許可するキーペアとセキュリティグループを使用してインスタンスを起動し、インスタンスにインターネット接続があることを確認します。インスタンスに接続するときに、キーペアの .pem ファイルを指定します。

SSH クライアントを使用して Mac インスタンスに接続するには、次の手順に従います。インスタンスの接続でエラーが発生した場合は、Amazon EC2 Linux インスタンスへの接続に関する問題のトラブルシューティングを参照してください。

SSH を使用してインスタンスに接続するには

1. コマンドラインでsshと入力して、ローカルコンピュータに SSH クライアントがインストールされていることを確認します。コンピュータがコマンドを認識しない場合は、お使いのオペレーティングシステム用の SSH クライアントを検索し、インストールします。
2. インスタンスのパブリック DNS 名を取得します。Amazon EC2 コンソールを使用して、[詳細] タブと [ネットワーク] タブの両方でパブリック DNS 名を確認できます。AWS CLI を使用して、describe-instances コマンドを使用してパブリック DNS 名を見つけることができます。
3. インスタンスの起動時に指定したキーペアの .pem ファイルを見つけます。
4. 次の ssh コマンドを使用してインスタンスに接続し、インスタンスのパブリック DNS 名と .pem ファイルを指定します。

```
ssh -i /path/key-pair-name.pem ec2-user@instance-public-dns-name
```

パスワード認証は、パスワードのブルートフォース攻撃を防ぐために無効になっています。SSH 設定を変更する前に、/usr/local/aws/ec2-macos-init/init.toml を開いて secureSSHDConfig を false に設定します。

インスタンスのグラフィカルユーザーインターフェイス (GUI) に接続する

VNC、Apple Remote Desktop (ARD)、または Apple Screen Sharing アプリケーション (macOS に付属) を使用してインスタンスの GUI に接続するには、以下の手順に従います。

> **Note**
>
> macOS 10.14 以降では、画面共有がシステム環境設定で有効になっている場合のみ制御できます。

ARD クライアントまたは VNC クライアントを使用してインスタンスに接続するには

1. ローカルコンピュータに、ARD をサポートしている ARD クライアントまたは VNC クライアントが、インストールされていることを確認します。macOS では、組み込みの画面共有アプリケーションを利用できます。インストールされていない場合は、お使いのオペレーティングシステム向けの ARD を検索し、インストールします。
2. ローカルコンピュータから、SSH を使用してインスタンスに接続します。
3. 次のように passwd コマンドを使用して、ec2-user アカウントのパスワードを設定します。

   ```
   [ec2-user ~]$ sudo passwd ec2-user
   ```

4. 次のコマンドを使用して macOS スクリーン共有をインストールして起動します。

   ```
   [ec2-user ~]$ sudo launchctl enable system/com.apple.screensharing
   sudo launchctl load -w /System/Library/LaunchDaemons/com.apple.screensharing.plist
   ```

5. exit と入力して Enter キーを押して、インスタンスとの接続を切断します。
6. コンピュータから、次の ssh コマンドを使用してインスタンスに接続します。前のセクションで示したオプションに加えて、ポート転送を有効にしローカルポート 5900 のすべてのトラフィックをインスタンスの ARD サーバーに転送する場合は、-L オプションを使用します。

   ```
   ssh -L 5900:localhost:5900 -i /path/key-pair-name.pem ec2-user@instance-public-dns-name
   ```

7. ローカルコンピュータから、ARD をサポートしている ARD クライアントまたは VNC クライアントを使用して、localhost:5900 に接続します。例えば、macOS で画面共有アプリケーションを次のように使用します。

a. [検索] を開いて、[移動] を選択してください。

b. [サーバーに接続] を選択してください。

c. [サーバーアドレス] フィールドに、`vnc://localhost:5900` と入力してください。

d. プロンプトに従って、**`ec2-user`** を使用して ec2-user アカウント用に作成したユーザー名とパスワードでログインします。

Mac インスタンスで macOS の画面解像度を変更する

ARD または ARD をサポートする VNC クライアントを使用して EC2 Mac インスタンスに接続したら、displayplacer などの公開されている macOS ツールやユーティリティのいずれかを使用して macOS 環境の画面解像度を変更できます。

displayplacer を使用して画面解像度を変更するには

1. displayplacer をインストールします。

```
[ec2-user ~]$ brew tap jakehilborn/jakehilborn && brew install displayplacer
```

2. 現在の画面情報と可能な画面解像度を表示します。

```
[ec2-user ~]$ displayplacer list
```

3. 希望の画面解像度を適用します。

```
[ec2-user ~]$ displayplacer "id:<screenID> res:<width>x<height> origin:(0,0)
 degree:0"
```

例えば、次のようになります。

```
RES="2560x1600"
displayplacer "id:69784AF1-CD7D-B79B-E5D4-60D937407F68 res:${RES} scaling:off
 origin:(0,0) degree:0"
```

Amazon EC2 Mac インスタンス上のオペレーティングシステムとソフトウェアの更新

次のトピックでは、Apple シリコン Mac インスタンス (Mac2、Mac2-m1ultra、Mac2-m2、Mac2-m2pro、Mac-m4、and Mac-m4pro) および x86 Mac インスタンス (Mac1) でオペレーティングシステムとソフトウェアを更新する方法について説明します。

> **Warning**
>
> ベータ版またはプレビュー版の macOS バージョンのインストールは、Apple シリコン Mac インスタンスでのみ可能です。Amazon EC2 は macOS のベータ版やプレビュー版をサポートしていないため、実稼働前の macOS バージョンに更新した後もインスタンスが機能し続けることは保証されません。
> ベータ版またはプレビュー版の macOS バージョンを Amazon EC2 にインストールすると、インスタンスの停止または終了時に、x86 Mac インスタンスが Amazon EC2 Mac 専有ホストのパフォーマンスを低下させるため、そのホストで新しいインスタンスを開始または起動できなくなります。

> **Note**
>
> AWS が公式 AMI をリリースする前にユーザーがインプレース macOS 更新を実行すると、その更新は選択したホストにのみ適用されます。他のホストがある場合、または新しいホストを起動する場合は、それらのホストでも同じ更新プロセスを実行する必要があります。macOS の各バージョンでは、基盤となる Apple Mac ハードウェアでファームウェアの最小バージョンが必要です。インプレース更新では、選択したホストのファームウェアのみが更新され、他の既存または新しいホストには反映されません。Amazon EC2 Mac 専有ホストと互換性がある macOS バージョンを確認するには、「Amazon EC2 Mac 専有ホストでサポートされている macOS バージョンを特定する」を参照してください。

Apple Silicon Mac インスタンスでソフトウェアを更新する

前提条件

ネットワークドライバー設定が更新されたため、ENA ドライバーバージョン 1.0.2 は macOS 13.3 以降と互換性がありません。ベータ版、プレビュー版、または実稼働版の macOS バージョン 13.3 以降をインストールする必要があり、最新の ENA ドライバーをインストールしていない場合は、次の手順を使用して新しいバージョンのドライバーをインストールします。

ENA ドライバーの新しいバージョンをインストールするには

1. ターミナルウィンドウで、SSH を使用して Apple Silicon Mac インスタンスに接続します。
2. Homebrew を更新し、次のコマンドを使用して、ENA アプリケーションを Applications ファイルにダウンロードします。

```
[ec2-user ~]$ brew update
```

```
[ec2-user ~]$ brew install amazon-ena-ethernet-dext
```

3. exit と入力して return キーを押して、インスタンスとの接続を切断します。
4. VNC クライアントを使用して ENA アプリケーションをアクティブ化します。
   a. インスタンスのグラフィカルユーザーインターフェイス (GUI) に接続する を使用して VNC クライアントを設定します。
   b. 画面共有アプリケーションを使用してインスタンスに接続したら、Applications フォルダーに移動して ENA アプリケーションを開きます。
   c. [Activate] を選択してください。
   d. ドライバーが正しくアクティブ化されたことを確認するには、ターミナルウィンドウで次のコマンドを実行します。コマンドの出力は、古いドライバが終了状態で、新しいドライバがアクティブ状態であることを示しています。

```
systemextensionsctl list;
```

   e. インスタンスを再起動すると、新しいドライバーのみが表示されます。

ソフトウェア更新を実行する

Apple Silicon Mac インスタンスでは、オペレーティングシステムのインプレースアップデートを実行するために数ステップの手順を実行する必要があります。これには、Amazon EBS ルートボリュームの所有権を EBS ルートボリューム管理ユーザーに委任することが含まれます。これを行うには、Amazon EC2 API を使用して自動的に行うか、もしくはインスタンスでコマンドを実行して手動で行うかのいずれかを選択できます。

Automated volume ownership delegation (Recommended)

考慮事項

- ボリューム所有権の委任タスクが完了するまでに 30～90 分かかる場合があります。この間、インスタンスは到達不能になります。
- 以下の macOS バージョンがサポートされています。
  - Mac2 | Mac2-m1ultra – macOS Ventura (バージョン 13.0 以降)
  - Mac2-m2 | Mac2-m2pro – macOS Ventura (バージョン 13.2 以降)
  - Mac-m4 | Mac-m4pro — macOS Sequoia (バージョン 15.6 以降)
- インスタンスにはブート可能なボリュームが 1 つだけあることが前提です。アタッチされた各ボリュームには、それぞれ管理ユーザーを 1 人追加できます。

ステップ 1: パスワードを設定し、EBS ルートボリューム管理ユーザーのセキュアトークンを有効にする

パスワードを設定し、Amazon EBS ルートボリューム管理ユーザー (ec2-user) のセキュアトークンを有効にする必要があります。

> **Note**
>
> パスワードとセキュアトークンは、GUI を使用して Apple シリコン Mac インスタンスに初めて接続するときに設定されます。以前に GUI を使用してインスタンスに接続したことがある場合は、これらのステップを実行する必要はありません。

1. SSH を使用してインスタンスに接続します。
2. ec2-user ユーザーのパスワードを設定します。

```
$ sudo /usr/bin/dscl . -passwd /Users/ec2-user
```

3. ec2-user ユーザーのセキュアトークンを有効にします。-oldPassword の場合、前のステップと同じパスワードを指定します。-newPassword の場合、別のパスワードを指定します。次のコマンドは、古いパスワードと新しいパスワードが .txt ファイルに保存されていることを前提としています。

```
$ sysadminctl -oldPassword `cat old_password.txt` -newPassword `cat
 new_password.txt`
```

4. セキュアトークンが有効になっていることを確認します。

```
$ sysadminctl -secureTokenStatus ec2-user
```

ステップ 2: Amazon EBS ルートボリュームの所有権を EBS ルートボリューム管理ユーザーに委任する

所有権を委任するには、ボリューム所有権の委任タスクを作成する必要があります。

1. create-delegate-mac-volume-ownership-task コマンドを使用してタスクを作成します。--instance-id の場合、インスタンスの ID を指定します。--mac-credentials の場合、次の認証情報を指定します。

   - 内部ディスク管理ユーザー
     - ユーザー名 – デフォルトの管理ユーザー (aws-managed-user) のみがサポートされ、デフォルトで使用されます。別の管理ユーザーを指定することはできません。
     - パスワード – aws-managed-user のデフォルトのパスワードを変更しなかった場合は、デフォルトのパスワード (空白) を指定します。それ以外の場合は、パスワードを指定します。
   - Amazon EBS ルートボリューム管理ユーザー
     - ユーザー名 – デフォルトの管理ユーザーを変更しなかった場合は、ec2-user を指定します。それ以外の場合は、管理ユーザーのユーザー名を指定します。
     - パスワード – 上記のステップ 1 でルートボリューム管理ユーザーに設定したパスワードを指定します。

```
aws ec2 create-delegate-mac-volume-ownership-task \
--instance-id i-1234567890abcdef0 \
--mac-credentials file://mac-credentials.json
```

以下は、前述の例で参照されている mac-credentials.json ファイルの内容です。

```
{
```

```
  "internalDiskPassword":"internal-disk-admin_password",
  "rootVolumeUsername":"root-volume-admin_username",
  "rootVolumepassword":"root-volume-admin_password"
}
```

2. ボリューム所有権の委任タスクが完了し、インスタンスが正常な状態に戻るまで待ちます。describe-mac-modification-tasks コマンドを使用します。--mac-modification-task-id には、前のステップのボリューム所有権の委任タスクの ID を指定します。

```
aws ec2 describe-mac-modification-tasks \
--mac-modification-task-id task-id
```

3. ボリューム所有権の委任タスクが完了したら、ステップ 3 に進みます。

ステップ 3: ソフトウェアを更新する

Amazon EBS ルートボリュームの所有権を委任したら、x86 Mac インスタンスでのソフトウェアの更新 (以下) で説明されているステップに従ってソフトウェアを更新します。

Manual volume ownership delegation

この手順を進めると、2 つのパスワードが作成されます。1 つのパスワードは Amazon EBS ルートボリューム管理ユーザー (ec2-user) 用で、もう 1 つのパスワードは内部ディスクの管理ユーザー (aws-managed-user) 用です。これらのパスワードは手順を進めるときに使用しますので、覚えておいてください。

> **Note**
>
> macOS Big Sur でこの手順を実行すると、macOS Big Sur 11.7.3 から macOS Big Sur 11.7.4 へのアップデートなど、マイナーアップデートのみしか実行できません。macOS Monterey 以降では、主要なソフトウェアアップデートを実行できます。

内部ディスクにアクセスするには

1. ローカルコンピュータのターミナルで、次のコマンドで SSH を使用して Apple Silicon Mac インスタンスに接続します。詳細については、SSH を使用したインスタンスへの接続を参照してください。

```
ssh -i /path/key-pair-name.pem ec2-user@instance-public-dns-name
```

2. 次のコマンドを使用して macOS スクリーン共有をインストールして起動します。

```
[ec2-user ~]$ sudo launchctl enable system/com.apple.screensharing
sudo launchctl load -w /System/Library/LaunchDaemons/
com.apple.screensharing.plist
```

3. 次のコマンドを実行して、ec2-user のパスワードを設定します。パスワードは後で使用するので覚えておいてください。

```
[ec2-user ~]$ sudo /usr/bin/dscl . -passwd /Users/ec2-user
```

4. exit と入力して return キーを押して、インスタンスとの接続を切断します。
5. ローカルコンピュータのターミナルで、次のコマンドを使用して VNC ポートへの SSH トンネルを使用してインスタンスに再接続します。

```
ssh -i /path/key-pair-name.pem -L 5900:localhost:5900 ec2-user@instance-public-dns-name
```

> **Note**
>
> 次の VNC 接続と GUI の手順が完了するまで、この SSH セッションを終了しないでください。インスタンスを再起動すると、接続は自動的に終了します。

6. ローカルコンピュータから、次の手順を使用して localhost:5900 に接続します。
   a. [検索] を開いて、[移動] を選択してください。
   b. [サーバーに接続] を選択してください。
   c. [サーバーアドレス] フィールドに、vnc://localhost:5900 と入力してください。
7. macOS ウィンドウで、ステップ 3 で作成したパスワードを使用して、ec2-user として Apple Silicon Mac インスタンスのリモートセッションに接続します。
8. 次のいずれかのオプションを使用して、InternalDisk という名前の内部ディスクにアクセスします。
   a. macOS Ventura 以上の場合: [システム設定] を開き、左側のペインで [一般] を選択し、ペインの右下で [起動ディスク] を選択してください。
   b. macOS Monterey 以下の場合: [システム環境設定] を開いて、[起動ディスク] を選択し、ウィンドウの左下にあるロックアイコンを選択してペインのロックを解除します。

> **トラブルシューティングのヒント**
>
> 内部ディスクをマウントする必要がある場合は、ターミナルで次のコマンドを実行します。
>
> ```
> APFSVolumeName="InternalDisk" ; SSDContainer=$(diskutil list | grep
>  "Physical Store disk0" -B 3 | grep "/dev/disk" | awk {'print $1'} ) ;
>  diskutil apfs addVolume $SSDContainer APFS $APFSVolumeName
> ```

9. InternalDisk という名前の内部ディスクを選択し、[再起動] を選択してください。メッセージが表示されたら、もう一度 [再起動] を選択してください。

> **Important**
>
> 内部ディスクの名前が InternalDisk ではなく Macintosh HD の場合は、専有ホストを更新できるようにインスタンスを停止して再起動する必要があります。詳細については、Amazon EC2 Mac インスタンスを停止または終了するを参照してください。

管理ユーザーに所有権を委任するには、次の手順に従います。SSH でインスタンスに再接続すると、特別な管理ユーザー aws-managed-userを使用して内部ディスクから起動されます。aws-managed-user 用の初期パスワードは空白なため、最初の接続時に上書きする必要があります。その後、ブートボリュームが変更されたため、手順を繰り返して macOS の画面共有をインストールして起動する必要があります。

Amazon EBS ボリュームの管理者に所有権を委任するには

1. ローカルコンピュータのターミナルで、次のコマンドを使用して Apple Silicon Mac インスタンスに接続します。

```
ssh -i /path/key-pair-name.pem aws-managed-user@instance-public-dns-name
```

2. WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED! の警告が表示されたら、以下のいずれかのコマンドを使用してこの問題を解決します。

   a. 次のコマンドを使用して、既知のホストを削除します。次に、前の手順を繰り返します。

```
rm ~/.ssh/known_hosts
```

b. 前の手順の SSH コマンドに、次の形式を追加します。

```
-o UserKnownHostsFile=/dev/null -o StrictHostKeyChecking=no
```

3. 次のコマンドを実行して、aws-managed-user のパスワードを設定します。aws-managed-user 初期パスワードは空白であるため、最初の接続時に上書きする必要があります。

a.
```
[aws-managed-user ~]$ sudo /usr/bin/dscl . -passwd /Users/aws-managed-user password
```

b. プロンプトが表示されたら、Permission denied. Please enter user's old password:、Enter キーを押します。

> **トラブルシューティングのヒント**
>
> passwd: DS error: eDSAuthFailed のエラーが発生した場合は、次のコマンドを使用します。
>
> ```
> [aws-managed-user ~]$ sudo passwd aws-managed-user
> ```

4. 次のコマンドを使用して macOS スクリーン共有をインストールして起動します。

```
[aws-managed-user ~]$ sudo launchctl enable system/com.apple.screensharing
sudo launchctl load -w /System/Library/LaunchDaemons/
com.apple.screensharing.plist
```

5. exit と入力して return キーを押して、インスタンスとの接続を切断します。
6. ローカルコンピュータのターミナルで、次のコマンドを使用して VNC ポートへの SSH トンネルを使用してインスタンスに再接続します。

```
ssh -i /path/key-pair-name.pem -L 5900:localhost:5900 aws-managed-user@instance-public-dns-name
```

7. ローカルコンピュータから、次の手順を使用して localhost:5900 に接続します。

a. [検索] を開いて、[移動] を選択してください。

b. [サーバーに接続] を選択してください。

c. [サーバーアドレス] フィールドに、`vnc://localhost:5900` と入力してください。

8. macOS ウィンドウで、ステップ 3 で作成したパスワードを使用して、`aws-managed-user` として Apple Silicon Mac インスタンスのリモートセッションに接続します。

> **Note**
>
> Apple ID でサインインするように求めるメッセージが表示されたら、[後でセットアップ] を選択してください。

9. Amazon EBS ボリュームには、次のいずれかのオプションを使用してアクセスします。

a. macOS Ventura 以降の場合: [システム設定] を開き、左側のペインで [一般] を選択し、ペインの右下で [起動ディスク] を選択してください。

b. macOS Monterey 以前の場合: [システム環境設定] を開き、[起動ディスク] を選択し、ウィンドウの左下にあるロックアイコンを使用してペインのロックを解除します。

> **Note**
>
> 再起動するまで、管理者パスワードの入力を求められたら、上記で設定した `aws-managed-user` 用のパスワードを使用してください。このパスワードは、`ec2-user` 用に設定したパスワードやインスタンスのデフォルトの管理者アカウントとは異なる場合があります。以下の手順では、インスタンスの管理者パスワードをいつ使用するかを指定します。

10. Amazon EBS ボリューム ([起動ディスク] ウィンドウの InternalDisk という名前が付いていないボリューム) を選択し、[再起動] を選択してください。

> **Note**
>
> Apple Silicon Mac インスタンスに複数の起動可能な Amazon EBS ボリュームがアタッチされている場合は、必ず各ボリュームにそれぞれ固有の名前を使用してください。

11. 再起動を確認してから、プロンプトが表示されたら [ユーザーの認可] を選択してください。

12. [このボリュームのユーザーを認可] ペインで、管理者ユーザー (デフォルトでは ec2-user) が選択されていることを確認し、[認可] を選択してください。
13. 前の手順の手順 3 で作成した ec2-user パスワードを入力し、[続行] を選択してください。
14. プロンプトが表示されたら、特別管理ユーザー aws-managed-userのパスワードを入力してください。
15. ローカルコンピュータからターミナルで、ec2-user ユーザー名で SSH を使用してインスタンスに再接続します。

    > **トラブルシューティングのヒント**
    >
    > WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED! の警告が表示されたら、次のコマンドを実行し、SSH を使用してインスタンスに再接続します。
    >
    > ```
    > rm ~/.ssh/known_hosts
    > ```

16. ソフトウェアアップデートを実行するには、x86 Mac インスタンスでのソフトウェアの更新の下にあるコマンドを使用します。

x86 Mac インスタンスでのソフトウェアの更新

x86 Mac インスタンスでは、softwareupdate コマンドを使用して、Apple からオペレーティングシステムの更新をインストールできます。

x86 Mac インスタンスで Apple からオペレーティングシステムの更新プログラムをインストールするには

1. 次のコマンドを使用して、利用可能な更新プログラムを含むパッケージを一覧表示します。

   ```
   [ec2-user ~]$ softwareupdate --list
   ```

2. すべての更新プログラムをインストールするか、特定の更新プログラムのみをインストールします。特定の更新プログラムをインストールするには、次のコマンドを使用します。

   ```
   [ec2-user ~]$ sudo softwareupdate --install label
   ```

   すべての更新プログラムをインストールするには、次のコマンドを使用します。

```
[ec2-user ~]$ sudo softwareupdate --install --all --restart
```

システム管理者は、AWS Systems Manager を使用することで、事前に承認されたオペレーティングシステムの更新を x86 Mac インスタンスにロールアウトできます。詳細については、AWS Systems Manager ユーザーガイドを参照してください。

Homebrew を使用して、EC2 macOS AMI にパッケージへの更新プログラムをインストールします。これにより、インスタンスでこのパッケージの最新バージョンを使用できます。また、Homebrew を使用して Amazon EC2 macOS に共通の macOS アプリケーションをインストールして実行することもできます。詳細については、Homwbrew ドキュメントを参照してください。

Homebrew を使用して更新プログラムをインストールするには

1. 次のコマンドを使用して Homwbrew を更新します。

   ```
   [ec2-user ~]$ brew update
   ```

2. 次のコマンドを使用して、利用可能な更新プログラムを含むパッケージを一覧表示します。

   ```
   [ec2-user ~]$ brew outdated
   ```

3. すべての更新プログラムをインストールするか、特定の更新プログラムのみをインストールします。特定の更新プログラムをインストールするには、次のコマンドを使用します。

   ```
   [ec2-user ~]$ brew upgrade package name
   ```

   すべての更新プログラムをインストールするには、次のコマンドを使用します。

   ```
   [ec2-user ~]$ brew upgrade
   ```

Mac インスタンスの EBS ボリュームのサイズを増やす

Mac インスタンスの Amazon EBS ボリュームのサイズを増やすことができます。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS Elastic Volumes」を参照してください。

ボリュームのサイズを大きくした後、APFS コンテナのサイズを以下のように大きくする必要があります。

使用可能なディスク容量を増やす

1. 再起動が必要かどうかを判断します。実行中の Mac インスタンスで既存の EBS ボリュームのサイズを変更した場合、新たなサイズを使用可能にするには、そのインスタンスを再起動する必要があります。起動中にディスク領域の変更が行われた場合、再起動は必要ありません。

   ディスクサイズに関する現在のステータスを表示します。

   ```
   [ec2-user ~]$  diskutil list external physical
   /dev/disk0 (external, physical):
      #:                       TYPE NAME                    SIZE       IDENTIFIER
      0:                 GUID_partition_scheme            *322.1 GB     disk0
      1:                 EFI EFI                           209.7 MB     disk0s1
      2:                 Apple_APFS Container disk2        321.9 GB     disk0s2
   ```

2. 以下のコマンドをコピーして貼り付けます。

   ```
   [ec2-user ~]$ PDISK=$(diskutil list physical external | head -n1 | cut -d" " -f1)
   APFSCONT=$(diskutil list physical external | grep "Apple_APFS" | tr -s " " | cut -
   d" " -f8)
   yes | sudo diskutil repairDisk $PDISK
   ```

3. 以下のコマンドをコピーして貼り付けます。

   ```
   [ec2-user ~]$ sudo diskutil apfs resizeContainer $APFSCONT 0
   ```

Amazon EC2 Mac インスタンスを停止または終了する

Mac インスタンスを停止すると、インスタンスは、stopping 状態に入るまでの約 15 分間は stopped 状態のままになります。

Mac インスタンスを停止または終了すると、Amazon EC2 は基盤となる専有ホスト上でスクラブワークフローを実行して、内部 SSD を消去し、永続的な NVRAM 変数をクリアし、最新のデバイスファームウェアに更新します。これにより、Mac インスタンスは、他の EC2 Nitro インスタンスと同じセキュリティとデータプライバシーを提供できます。また、最新の macOS AMI を実行することも可能です。スクラブワークフローの最中に、専有ホストは一時的に保留状態になります。x86 Mac インスタンスでは、スクラブワークフローが完了するまでに最大で 50 分かかる場合があります。Amazon EC2 がデバイスファームウェアを更新する必要がある場合、ワークフローの完了には最大 3 時間かかる可能性があります。Apple シリコン Mac インスタンスでは、スクラブワークフローが完了するまでに最大で 4.5 時間かかる場合があります。

スクラブワークフローが完了するまで、停止した Mac インスタンスを起動したり、新しい Mac インスタンスを起動したりすることはできません。完了の時点で、Dedicated Host は `available` 状態になります。

専有ホストが `pending` 状態になると、メータリングと課金が一時停止されます。スクラブワークフロー中は課金されません。

Mac インスタンス用の Dedicated Host を解放する

Mac インスタンスの使用が終了したら、専有ホストをリリースすることでクリーンアップできます。専有ホストをリリースする前に、Mac インスタンスを停止または終了する必要があります。割り当て期間が少なくとも 24 時間を超えるまで、ホストをリリースすることはできません。

専有ホストをリリースするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択し、[インスタンスの状態] をクリックしてから、[インスタンスの停止] または [インスタンスの終了] を選択してください。
4. ナビゲーションペインで [Dedicated Hosts] を選択してください。
5. 専有ホストを選択し、[アクション]、[ホストのリリース] の順に選択してください。
6. 確認を求めるメッセージが表示されたら、[リリース] を選択してください。

Amazon EC2 Mac インスタンスのシステム整合性保護を構成する

x86 Mac インスタンスと Apple シリコン Mac インスタンスのシステム整合性保護 (SIP) を設定できます。SIP は、不正なコード実行やシステムレベルの改変を防ぐのに役立つ重要な macOS セキュリティ機能です。詳細については、「About System Integrity Protection」を参照してください。

すべての SIP 設定を有効または無効にすることもできるし、あるいは特定の SIP 設定のみを選択的に有効または無効にすることもできます。SIP は、タスクの実行に必要な時間だけ一時的に無効にし、その後はできるだけ早く再度有効化することをお勧めします。SIP を無効のままにしておくと、インスタンスが悪意のあるコードに対して脆弱になる可能性があります。

SIP 設定は、Amazon EC2 Mac インスタンスがサポートされているすべての AWS リージョンでサポートされています。

トピック

- 考慮事項

- デフォルトの SIP 設定
- SIP 設定を確認する
- Apple シリコン Mac インスタンスの前提条件
- SIP 設定を構成する
- SIP 設定タスクのステータスを確認する

考慮事項

- 次の Amazon EC2 Mac インスタンスタイプと macOS バージョンがサポートされています。
  - Mac1 | Mac2 | Mac2-m1ultra – macOS Ventura (バージョン 13.0 以降)
  - Mac2-m2 | Mac2-m2pro – macOS Ventura (バージョン 13.2 以降)
  - Mac-m4 | Mac-m4pro — macOS Sequoia (バージョン 15.6 以降)

  > **Note**
  > ベータバージョンおよびプレビューバージョンの macOS はサポートされていません。

- カスタム SIP 設定を指定して、個々の SIP 設定を選択的に有効または無効にできます。カスタム設定を実装したら、インスタンスに接続して設定を確認し、要件が適切に実装され、意図したとおりに機能していることを確認すること。

  SIP 設定は、macOS の更新に伴って変更される可能性があります。macOS バージョンのアップグレード後は、セキュリティ設定のシームレスな互換性と適切な機能性を保証するため、カスタム SIP 設定の内容を改めて確認することをお勧めします。
- x86 Mac インスタンスの場合、SIP 設定はインスタンスレベルで適用されます。インスタンスにアタッチされたルートボリュームは、構成された SIP 設定を自動的に引き継ぎます。

  Apple シリコン Mac インスタンスの場合、SIP 設定はボリュームレベルで適用されます。そのため、インスタンスにアタッチされたルートボリュームは SIP 設定を引き継ぎません。別のルートボリュームをアタッチする場合は、SIP 設定を必要な状態に再構成する必要があります。
- SIP 設定タスクが完了するまでに最大 90 分かかる場合があります。SIP 設定タスクの進行中、インスタンスはアクセスできない状態となります。
- SIP 設定は、後でインスタンスから作成するスナップショットや AMI には反映されません。
- Apple シリコン Mac インスタンスにはブート可能なボリュームが 1 つだけあることが前提です。アタッチされた各ボリュームには、それぞれ管理者ユーザーを 1 人追加できます。

デフォルトの SIP 設定

次の表に、x86 Mac インスタンスと Apple シリコン Mac インスタンスのデフォルトの SIP 設定を示します。

	Apple シリコン Mac インスタンス	x86 Mac インスタンス
Apple 内部	有効	無効
ファイルシステムの保護	有効	無効
ベースシステム	有効	有効
デバッグの制限	有効	有効
Dtrace 制限	有効	有効
Kext 署名	有効	有効
Nvram の保護	有効	有効

SIP 設定を確認する

変更の前後に SIP 設定をチェックして、想定どおりに設定されているかどうか確認することをお勧めします。

Amazon EC2 Mac インスタンスの SIP 設定を確認するには

SSH を使用してインスタンスに接続し、コマンドラインで次のコマンドを実行します。

```
$ csrutil status
```

以下は出力の例です。

```
System Integrity Protection status: enabled.

Configuration:
    Apple Internal: enabled
    Kext Signing: disabled
```

```
    Filesystem Protections: enabled
    Debugging Restrictions: enabled
    DTrace Restrictions: enabled
    NVRAM Protections: enabled
    BaseSystem Verification: disabled
```

Apple シリコン Mac インスタンスの前提条件

Apple シリコン Mac インスタンスの SIP 設定を構成する前に、パスワードを設定し、Amazon EBS ルートボリューム管理ユーザー (ec2-user) のセキュアトークンを有効にする必要があります。

> **Note**
>
> パスワードとセキュアトークンは、GUI を使用して Apple シリコン Mac インスタンスに初めて接続するときに設定されます。以前に GUI を使用してインスタンスに接続したことがある場合、または x86 Mac インスタンスを使用している場合は、これらのステップを実行する必要はありません。

> **Note**
>
> SIP 設定の API コールで使用するには、macOS 認証に使用されるすべての macOS ユーザー名とパスワードが 4 ~ 16 文字である必要があります。

パスワードを設定し、EBS ルートボリューム管理ユーザーのセキュアトークンを有効にするには

1. SSH を使用してインスタンスに接続します。
2. ec2-user ユーザーのパスワードを設定します。

   ```
   $ sudo /usr/bin/dscl . -passwd /Users/ec2-user
   ```

3. ec2-user ユーザーのセキュアトークンを有効にします。-oldPassword の場合、前のステップと同じパスワードを指定します。-newPassword の場合、別のパスワードを指定します。次のコマンドは、古いパスワードと新しいパスワードが .txt ファイルに保存されていることを前提としています。

   ```
   $ sysadminctl -oldPassword `cat old_password.txt` -newPassword `cat
    new_password.txt`
   ```

4. セキュアトークンが有効になっていることを確認します。

```
$ sysadminctl -secureTokenStatus ec2-user
```

SIP 設定を構成する

インスタンスの SIP 設定を構成するときは、すべての SIP 設定を有効または無効にすることもできるし、あるいは特定の SIP 設定を選択的に有効または無効にするカスタム設定を実装することもできます。

> **Note**
>
> カスタム設定を実装したら、インスタンスに接続して設定を確認し、要件が適切に実装され、意図したとおりに機能していることを確認すること。
> SIP 設定は、macOS の更新に伴って変更される可能性があります。macOS バージョンのアップグレード後は、セキュリティ設定のシームレスな互換性と適切な機能性を保証するため、カスタム SIP 設定の内容を改めて確認することをお勧めします。

インスタンスの SIP 設定を構成するには、SIP 設定タスクを作成する必要があります。SIP 設定タスクでは、インスタンスの SIP 設定を指定します。

Apple シリコン Mac インスタンスの SIP 設定を作成するときは、次の認証情報を指定する必要があります。

- 内部ディスク管理ユーザー
  - ユーザー名 – デフォルトの管理ユーザー (aws-managed-user) のみがサポートされ、デフォルトで使用されます。別の管理ユーザーを指定することはできません。
  - パスワード – aws-managed-user のデフォルトのパスワードを変更しなかった場合は、デフォルトのパスワード (空白) を指定します。それ以外の場合は、パスワードを指定します。
- Amazon EBS ルートボリューム管理ユーザー
  - ユーザー名 – デフォルトの管理ユーザーを変更しなかった場合は、ec2-user を指定します。それ以外の場合は、管理ユーザーのユーザー名を指定します。
  - パスワード – パスワードは常に指定する必要があります。

SIP 設定タスクを作成するには、次の方法を使用します。

Console

コンソールを使用して SIP 設定タスクを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションパネルで [インスタンス] を選択し、Amazon EC2 Mac インスタンスを選択してください。
3. [セキュリティ] タブで、[Mac の変更、システム整合性保護の変更] を選択します。
4. すべての SIP 設定を有効にするには、[SIP を有効にする] を選択します。すべての SIP 設定を無効にするには、[SIP を有効にする] の選択を解除します。
5. 特定の SIP 設定を選択的に有効または無効にするカスタム設定を指定するには、[カスタム SIP 設定を指定] を選択し、有効にする SIP 設定を選択するか、無効にする SIP 設定の選択を解除します。
6. ルートボリュームユーザーと内部ディスク所有者の認証情報を指定します。
7. [SIP 変更タスクの作成] を選択します。

AWS CLI

AWS CLI を使用して SIP 設定タスクを作成するには

create-mac-system-integrity-protection-modification-task コマンドを使用します。

すべての SIP 設定を有効または無効にする

すべての SIP 設定を完全に有効または無効にするには、`--mac-system-integrity-protection-status` パラメータのみを使用します。

次のコマンド例では、すべての SIP 設定を有効にします。

```
aws ec2 create-mac-system-integrity-protection-modification-task \
--instance-id i-0abcdef9876543210 \
--mac-system-integrity-protection-status enabled \
--mac-credentials file://mac-credentials.json
```

カスタム SIP 設定を指定する

特定の SIP 設定を選択的に有効または無効にするカスタム SIP 設定を指定するには、`--mac-system-integrity-protection-status` および `--mac-system-integrity-`

protection-configuration パラメータを指定します。この場合、mac-system-integrity-protection-status により SIP ステータス全体を指定し、mac-system-integrity-protection-configuration により個々の SIP 設定を選択的に有効または無効にします。

次のコマンド例では、NvramProtections と FilesystemProtections を除くすべての SIP 設定を有効にする SIP 設定タスクを作成します。

```
aws ec2 create-mac-system-integrity-protection-modification-task \
--instance-id i-0abcdef9876543210 \
--mac-system-integrity-protection-status enabled \
--mac-system-integrity-protection-configuration
 "NvramProtections=disabled, FilesystemProtections=disabled" \
--mac-credentials file://mac-credentials.json
```

次のコマンド例では、DtraceRestrictions を除くすべての SIP 設定を無効にする SIP 設定タスクを作成します。

```
aws ec2 create-mac-system-integrity-protection-modification-task \
--instance-id i-0abcdef9876543210 \
--mac-system-integrity-protection-status disabled \
--mac-system-integrity-protection-configuration "DtraceRestrictions=enabled" \
--mac-credentials file://mac-credentials.json
```

mac-credentials.json ファイルの内容

以下は、前述の例で参照されている mac-credentials.json ファイルの内容です。

```
{
  "internalDiskPassword":"internal-disk-admin_password",
  "rootVolumeUsername":"root-volume-admin_username",
  "rootVolumepassword":"root-volume-admin_password"
}
```

SIP 設定タスクのステータスを確認する

SIP 設定タスクのステータスを確認するには、次のいずれかの方法を使用します。

Console

コンソールで SIP 設定タスクを表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションパネルで [インスタンス] を選択し、Amazon EC2 Mac インスタンスを選択してください。
3. [セキュリティ] タブで、[Mac 変更タスク] セクションまで下にスクロールします。

AWS CLI

AWS CLI を使用して SIP 設定タスクのステータスを確認するには

describe-mac-modification-tasks コマンドを使用します。

Amazon EC2 Mac 専有ホストでサポートされている macOS バージョンを特定する

Amazon EC2 Mac 専有ホストでサポートされている最新の macOS バージョンを表示できます。この機能を使用すると、専有ホストが任意の macOS バージョンでのインスタンスの起動をサポートできるかどうかを検証できます。

macOS の各バージョンで正常に起動するには、基盤となる Apple Mac でファームウェアの最小バージョンが必要です。割り当てられた Mac 専有ホストが長期間アイドル状態のままである場合、または長時間実行されているインスタンスがある場合、Apple Mac ファームウェアのバージョンが古くなる可能性があります。

最新の macOS バージョンを確実にサポートするために、割り当てられた Mac 専有ホストでインスタンスを停止または終了できます。これにより、ホストスクラブワークフローがトリガーされ、基盤となる Apple Mac のファームウェアが最新の macOS バージョンをサポートするように更新されます。インスタンスが長時間実行されている専有ホストは、実行中のインスタンスを停止または終了すると、自動的に更新されます。

スクラブワークフローの詳細については、「Amazon EC2 Mac インスタンスを停止または終了する」を参照してください。

Mac インスタンスの起動方法の詳細については、「AWS マネジメントコンソール または AWS CLI を使用して Mac インスタンスを起動する」を参照してください。

Amazon EC2 コンソールまたは AWS CLI を使用して、割り当てられた専有ホストでサポートされている最新の macOS バージョンに関する情報を表示できます。

Console

コンソールを使用して専有ホストのファームウェア情報を表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Dedicated Hosts] を選択してください。
3. [専有ホストの詳細] ページの [サポートされている最新の macOS バージョン] に、ホストがサポートできる最新の macOS バージョンが表示されます。

AWS CLI

AWS CLI を使用して専有ホストファームウェア情報を表示するには

`describe-mac-hosts` コマンドを使用して、`region` を適切な AWS リージョンに置き換えます。

```
$ aws ec2 describe-mac-hosts --region us-east-1
  {
      "MacHosts": [
          {
              "HostId": "h-07879acf49EXAMPLE",
              "MacOSLatestSupportedVersions": [
                  "14.3",
                  "13.6.4",
                  "12.7.3"
              ]
          }
      ]
  }
```

macOS AMI の通知へのサブスクライブ

新しい AMI のリリース時、あるいは BridgeOS の更新時に通知を受け取るには、Amazon SNS を使用して通知にサブスクライブします。

EC2 macOS AMI の詳細については、「Amazon EC2 macOS AMI リリースノート」を参照してください。

macOS AMI の通知にサブスクライブするには

1. Amazon SNS コンソールのhttps://console.aws.amazon.com/sns/v3/homeを開いてください。
2. ナビゲーションバーで、必要に応じて、リージョンを [米国東部 (バージニア北部)] に変更します。購読する SNS 通知がこのリージョンで作成されているため、このリージョンを使用する必要があります。
3. ナビゲーションペインで [Subscriptions] を選択してください。
4. [Create subscription] を選択してください。
5. [サブスクリプションの作成] ダイアログボックスで、次の操作を行います。

   a. [ARN のトピック] で、次の Amazon リソースネーム (ARN) のいずれかをコピーアンドペーストします。

      - **`arn:aws:sns:us-east-1:898855652048:amazon-ec2-macos-ami-updates`**
      - **`arn:aws:sns:us-east-1:898855652048:amazon-ec2-bridgeos-updates`**

   b. [プロトコル] で、以下のいずれかを選択してください。

      - E メール:

        [エンドポイント] では、通知を受信するために使用できる E メールアドレスを入力してください。サブスクリプションを作成した後、件名が「`AWS Notification - Subscription Confirmation`」とされた確認メッセージが送られてきます。このメールを開き、[サブスクリプションの確認] をクリックして受信登録を完了します。

      - SMS:

        エンドポイントに、通知を受信するために使用する E メールアドレスを入力してください。

      - AWS Lambda、Amazon SQS、Amazon Data Firehose (通知は JSON 形式で送信されます):

        [Endpoint] (エンドポイント) に、通知を受信するために使用する Lambda 関数、SQS キュー、または Firehose ストリームの ARN を入力してください。

   c. [Create subscription] を選択してください。

macOS AMI がリリースされるたびに、`amazon-ec2-macos-ami-updates` トピックのサブスクライバーに対し通知が送信されます。BridgeOS が変更されるたびに、`amazon-ec2-bridgeos-`

updates トピックのサブスクライバーに対し通知が送信されます。通知が不要になった場合は、次の手順で受信登録を解除します。

macOS AMI の通知のサブスクライブを解除するには

1. Amazon SNS コンソールのhttps://console.aws.amazon.com/sns/v3/homeを開いてください。
2. ナビゲーションバーで、必要に応じて、リージョンを [米国東部 (バージニア北部)] に変更します。SNS 通知はこのリージョンで作成されたため、このリージョンを使用する必要があります。
3. ナビゲーションペインで [Subscriptions] を選択してください。
4. サブスクリプションを選択し、[アクション]、[サブスクリプションの削除] を選択してください。確認のプロンプトが表示されたら、[削除] を選択してください。

AWS Systems Manager Parameter Store API を使用して macOS AMI ID を取得する

インスタンスを起動するときは、AMI を指定する必要があります。AMI は、AWS リージョン、オペレーティングシステム、プロセッサアーキテクチャに固有です。AWS Systems Manager Parameter Store API をクエリすることで、AWS リージョン 内のすべての macOS AMI を表示したり、最新の macOS AMI を取得したりできます。これらのパブリックパラメータを使用すると、macOS AMI ID を手動で検索する必要がなくなります。パブリックパラメータは x86 および ARM64 macOS AMI で使用でき、既存の AWS CloudFormation テンプレートと統合できます。

必要なアクセス許可

このアクションを実行するには、IAM プリンシパルに `ssm:GetParameter` API アクションを呼び出すアクセス許可が必要です。

AWS CLI を使用して現在の AWS リージョン内のすべての macOS AMI のリストを表示するには

次の get-parameters-by-path コマンドを使用して、現在のリージョン内のすべての macOS AMI のリストを表示します。

```
aws ssm get-parameters-by-path --path /aws/service/ec2-macos --recursive --query
 "Parameters[].Name"
```

AWS CLI を使用して最新のメジャー macOS AMI の AMI ID を取得するには

サブパラメータ `image_id` を入れた次の get-parameter コマンドを使用します。次の例で、`sonoma` は macOS がサポートされているメジャーバージョン、`x86_64_mac` はプロセッサ、`region-code` は最新の macOS AMI ID を取得したいサポートされている AWS リージョンに置き換えます。

```
aws ssm get-parameter --name /aws/service/ec2-macos/sonoma/x86_64_mac/latest/image_id
 --region region-code
```

詳細については、「AWS Systems Manager ユーザーガイド」の「macOS 用 AMI パブリックパラメータを呼び出す」を参照してください。

Amazon EC2 macOS AMI リリースノート

以下の情報は、EC2 macOS AMI にデフォルトで含まれているパッケージの詳細と、各 EC2 macOS AMI リリースの変更点をまとめたものです。

macOS AMI の通知をサブスクライブする方法については、macOS AMI の通知へのサブスクライブを参照してください。

Mac インスタンスでは、次のいずれかのオペレーティングシステムを実行できます。

- macOS Mojave (バージョン 10.14) (x86 Mac インスタンスのみ)
- macOS Catalina (バージョン 10.15) (x86 Mac インスタンスのみ)
- macOS Big Sur (バージョン 11) (x86 および M1 Mac インスタンス)
- macOS Monterey (バージョン 12) (x86 および M1 Mac インスタンス)
- macOS Ventura (バージョン 13) (すべての Mac インスタンス、M2 および M2 Pro Mac インスタンスは macOS Ventura バージョン 13.2 以降をサポート)
- macOS Sonoma (バージョン 14) (すべての Mac インスタンス)
- macOS Sequoia (バージョン 15) (すべての Mac インスタンス)

> **Note**
>
> M4 および M4 Pro Mac インスタンスは、macOS Sequoia バージョン 15.6 以降をサポートしています。

macOS Sequoia のローカルネットワークプライバシーポリシーを承認する

macOS Sequoia (バージョン 15) には、Amazon EC2 インスタンスメタデータサービス (IMDS) を含むローカル IP ベースのサービスのユーザーに影響を与える新しいローカルネットワークプライバシー機能があります。

> **Important**
>
> ローカル IP ベースのサービスに中断なくアクセスできるようにするには、次の手順を使用してローカルネットワークプライバシーポリシーを承認します。

ローカルネットワークプライバシーポリシーを承認するには

1. インスタンスのグラフィカルユーザーインターフェイス (GUI) に接続する.
2. 画面のプロンプトに従って、ローカルネットワークプライバシーポリシーを承認します。
3. ポリシーを承認したら、EC2 Mac インスタンスの AMI を作成します。詳細については、Amazon EBS-backed AMI を作成するを参照してください。

新しく作成された AMI から起動された EC2 Mac インスタンスは、ローカルネットワークプライバシーアクセス許可を保持します。

Amazon EC2 macOS AMI に含まれるデフォルトのパッケージ

以下の表は、EC2 macOS AMI にデフォルトで含まれているパッケージの概要をまとめたものです。

パッケージ	リリースノート
EC2 macOS Init	https://github.com/aws/ec2-macos-init/tags
EC2 macOS ユーティリティ	https://github.com/aws/ec2-macos-utils/tags
Amazon SSM Agent	https://github.com/aws/amazon-ssm-agent/releases
AWS Command Line Interface AWS CLIバージョン 2	https://raw.githubusercontent.com/aws/aws-cli/v2/CHANGELOG.rst

パッケージ	リリースノート
Xcode 用のコマンドラインツール	https://developer.apple.com/documentation/xcode-release-notes
Homebrew	https://github.com/Homebrew/brew/releases
EC2 Instance Connect	https://github.com/aws/aws-ec2-instance-connect-config/releases
Safari	https://developer.apple.com/documentation/safari-release-notes

Amazon EC2 macOS AMI の更新

次の表は、EC2 macOS AMI リリースに含まれる変更点をまとめたものです。すべての EC2 macOS AMI に適用される変更もあれば、これらの AMI のサブセットにのみ適用される変更もあります。

EC2 macOS AMI の更新

リリース	変更
2026 年 5 月 19 日	すべての AMI • `awscli` を 2.34.46 に更新 • Homebrew を 5.1.11 に更新 MacOS Sonoma 14.8.7 のリリース • MacOS Sonoma 14.8.7 のセキュリティコンテンツ • Safari を 26.5 に更新 • Safari 26.5 のセキュリティコンテンツ macOS Sequoia 15.7.7 のリリース • macOS Sequoia 15.7.7 のセキュリティコンテンツ • Safari を 26.5 に更新

リリース	変更
	• Safari 26.5 のセキュリティコンテンツ macOS Tahoe 26.5 のリリース • macOS Tahoe 26.5 のセキュリティコンテンツ • コマンドラインツールを 26.5 に更新
2026 年 4 月 20 日	macOS Tahoe 26.4.1 のリリース • macOS Tahoe 26.4.1 のセキュリティコンテンツ • コマンドラインツールを 26.4.1 に更新 • `awscli` を 2.34.32 に更新 • Homebrew を 5.1.7 に更新

リリース	変更

2026 年 4 月 16 日	すべての AMI
	• awscli を 2.34.27 に更新
	• ec2-macos-init を 1.5.15 に更新
	• Homebrew を 5.1.5 に更新
	
	macOS Sonoma 14.8.5 のリリース
	• macOS Sonoma 14.8.5 のセキュリティコンテンツ
	• Safari を 26.4 に更新
	  • Safari 26.4 のセキュリティコンテンツ
	
	macOS Sequoia 15.7.5 のリリース
	• macOS Sequoia 15.7.5 のセキュリティコンテンツ
	• Safari を 26.4 に更新
	  • Safari 26.4 のセキュリティコンテンツ
	
	macOS Tahoe 26.4 のリリース
	• macOS Tahoe 26.4 のセキュリティコンテンツ
	• コマンドラインツールを 26.4 に更新

2026 年 3 月 17 日	macOS Tahoe 26.3.1 のリリース • macOS Tahoe 26 のアップデートに関する最新情報 • コマンドラインツールを 26.3 に更新 • awscli を 2.34.10 に更新 • Homebrew を 5.1.0 に更新

リリース	変更

2026 年 3 月 3 日	すべての AMI
	• `awscli` を 2.33.31 に更新
	• Homebrew を 5.0.15 に更新
	
	macOS Sonoma 14.8.4 のリリース
	• macOS Sonoma 14.8.4 のセキュリティコンテンツ
	• Safari を 26.3 に更新
	  • Safari 26.3 のセキュリティコンテンツ
	
	macOS Sequoia 15.7.4 のリリース
	• macOS Sequoia 15.7.4 のセキュリティコンテンツ
	• Safari を 26.3 に更新
	  • Safari 26.3 のセキュリティコンテンツ
	
	macOS Tahoe 26.3 のリリース
	• macOS Tahoe 26.3 のセキュリティコンテンツ

リリース	変更

2025 年 12 月 26 日	すべての AMI
	• `awscli` を 2.32.19 に更新
	• `amazon-ssm-agent` を 3.3.3270.0 に更新
	• Homebrew を 5.0.6 に更新
	
	macOS Sonoma 14.8.3 のリリース
	• macOS Sonoma 14.8.3 のセキュリティコンテンツ
	• Safari を 26.2 に更新
	  • Safari 26.2 のセキュリティコンテンツ
	
	macOS Sequoia 15.7.3 のリリース
	• macOS Sequoia 15.7.3 のセキュリティコンテンツ
	• Safari を 26.2 に更新
	  • Safari 26.2 のセキュリティコンテンツ
	• コマンドラインツールを 26.2 に更新
	
	macOS Tahoe 26.2 のリリース
	• macOS Tahoe 26.2 のセキュリティコンテンツ
	• コマンドラインツールを 26.2 に更新

リリース	変更

2025 年 12 月 17 日	すべての AMI
	• `awscli` を 2.32.16 に更新
	• Homebrew を 5.0.5 に更新
	
	macOS Sonoma 14.8.2 のリリース
	• macOS Sonoma 14.8.2 のセキュリティコンテンツ
	• Safari を 26.1 に更新
	  • Safari 26.1 のセキュリティコンテンツ
	
	macOS Sequoia 15.7.2 のリリース
	• macOS Sequoia 15.7.2 のセキュリティコンテンツ
	• Safari を 26.1 に更新
	  • Safari 26.1 のセキュリティコンテンツ
	• コマンドラインツールを 26.1 に更新
	
	macOS Tahoe 26.1 のリリース
	• macOS Tahoe 26.1 のセキュリティコンテンツ
	• コマンドラインツールを 26.1 に更新

リリース	変更

2025 年 11 月 18 日	すべての AMI
	• awscli を 2.31.35 に更新
	• ec2-macos-init を 1.5.13 に更新
	• ec2-macos-utils を 1.0.7 に更新
	• Homebrew を 5.0.1 に更新
	
	macOS Sonoma 14.8.1 のリリース
	• macOS Sonoma 14.8.1 のセキュリティコンテンツ
	• Safari を 26.0.1 に更新
	  • Safari 26.0.1 のセキュリティコンテンツ
	
	macOS Sequoia 15.7.1 のリリース
	• macOS Sequoia 15.7.1 のセキュリティコンテンツ
	• Safari を 26.0.1 に更新
	  • Safari 18.6 のセキュリティコンテンツ
	• コマンドラインツールを 26.0 に更新
	
	macOS Tahoe 26.0.1 のリリース
	• macOS Tahoe 26.0.1 のセキュリティコンテンツ
	• コマンドラインツールを 26.0 に更新

リリース	変更
2025 年 9 月 4 日	すべての AMI • `awscli` を 2.28.19 に更新 • `ec2-instance-connect` を 2.0.0-5 に更新 • Homebrew を 4.6.7 に更新 macOS Ventura 13.7.8 のリリース • macOS Ventura 13.7.8 のセキュリティコンテンツ MacOS Sonoma 14.7.8 のリリース • MacOS Sonoma 14.7.8 のセキュリティコンテンツ macOS Sequoia 15.6.1 のリリース • macOS Sequoia 15.6.1 のセキュリティコンテンツ

リリース	変更

2025 年 8 月 5 日	すべての AMI
	• `awscli` が 2.28.0 に更新されました
	• Homebrew が 4.5.13 に更新されました
	
	macOS Ventura 13.7.7 のリリース
	• macOS Ventura 13.7.7 のセキュリティコンテンツ
	• Safari が 18.6 に更新されました
	  • Safari 18.6 のセキュリティコンテンツ
	
	MacOS Sonoma 14.7.7 のリリース
	• MacOS Sonoma 14.7.7 のセキュリティコンテンツ
	• Safari が 18.6 に更新されました
	  • Safari 18.6 のセキュリティコンテンツ
	
	MacOS Sequoia 15.6 のリリース
	• MacOS Sequoia 15.6 のセキュリティコンテンツ
	• AWS ENA イーサネットが 2.0.0 に更新されました

リリース	変更

2025 年 6 月 27 日	すべての AMI
	• `awscli` を 2.27.40 に更新
	• Homebrew を 4.5.7 に更新
	• x86_64 イメージを AWS ENA イーサネット dext に移行
	
	macOS Ventura 13.7.6 のリリース
	• macOS Ventura 13.7.6 のセキュリティコンテンツ
	• Safari を 18.5 に更新
	  • Safari 18.5 のセキュリティコンテンツ
	
	macOS Sonoma 14.7.6 のリリース
	• macOS Sonoma 14.7.6 のセキュリティコンテンツ
	• Safari を 18.5 に更新
	  • Safari 18.5 のセキュリティコンテンツ
	
	macOS Sequoia 15.5 のリリース
	• macOS Sequoia 15.5 のセキュリティコンテンツ
	• コマンドラインツールを 16.4 に更新
	• ENA AWS イーサネットを 1.0.9 に更新

リリース	変更

2025 年 5 月 21 日	すべての AMI
	• awscli を 2.27.7 に更新
	• ec2-macos-init を 1.5.11 に更新
	• ec2-macos-utils を 1.0.5 に更新
	• Homebrew を 4.5.1 に更新
	
	macOS Ventura 13.7.5 のリリース
	• macOS Ventura 13.7.5 のセキュリティコンテンツ
	• Safari を 18.4 に更新
	  • Safari 18.4 のセキュリティコンテンツ
	
	macOS Sonoma 14.7.5 のリリース
	• macOS Sonoma 14.7.5 のセキュリティコンテンツ
	• Safari を 18.4 に更新
	  • Safari 18.4 のセキュリティコンテンツ
	
	macOS Sequoia 15.4.1 のリリース
	• macOS Sequoia 15.4.1 のセキュリティコンテンツ
	• コマンドラインツールを 16.3 に更新

リリース	変更

2025 年 5 月 5 日	すべての AMI
	• awscli を 2.27.1 に更新
	• ec2-macos-init を 1.5.11 に更新
	• ec2-macos-system-monitor を 1.3.1 に更新
	• ec2-macos-utils を 1.0.5 に更新
	• Homebrew を 4.4.32 に更新
	
	macOS Ventura 13.7.5 のリリース
	• macOS Ventura 13.7.5 のセキュリティコンテンツ
	• Safari を 18.4 に更新
	  • Safari 18.4 のセキュリティコンテンツ
	
	macOS Sonoma 14.7.5 のリリース
	• macOS Sonoma 14.7.5 のセキュリティコンテンツ
	• Safari を 18.4 に更新
	  • Safari 18.4 のセキュリティコンテンツ
	
	macOS Sequoia 15.4.1 のリリース
	• macOS Sequoia 15.4.1 のセキュリティコンテンツ

リリース	変更
2025 年 3 月 18 日	すべての AMI • `awscli` を 2.24.2 に更新 • Homebrew を 4.4.20 に更新 macOS Sequoia 15.3.1 のリリース • macOS Sequoia 15.3.1 のセキュリティコンテンツ macOS Sonoma 14.7.4 のリリース • macOS Sonoma 14.7.4 のセキュリティコンテンツ • Safari を 18.3 に更新 macOS Ventura 13.7.4 のリリース • macOS Ventura 13.7.4 のセキュリティコンテンツ • Safari を 18.3 に更新

リリース	変更
2025 年 1 月 24 日	すべての AMI • `awscli` を 2.22.33 に更新 • Homebrew を 4.4.15 に更新 macOS Sequoia 15.2 のリリース • macOS Sequoia 15.2 のセキュリティコンテンツ • コマンドラインツールを 16.2 に更新 macOS Sonoma 14.7.2 のリリース • macOS Sonoma 14.7.2 のセキュリティコンテンツ • Safari を 18.2 に更新 • コマンドラインツールを 16.2 に更新 macOS Ventura 13.7.2 のリリース • macOS Ventura 13.7.2 のセキュリティコンテンツ • Safari を 18.2 に更新

リリース	変更
2024 年 12 月 20 日	すべての AMI • Homebrew を 4.4.8 に更新 • `aws-cli` を 2.22.5 に更新 • `amazon-ssm-agent` を 3.3.987.0 に更新 macOS Sequoia 15.1.1 のリリース • macOS Sequoia 15.1.1のセキュリティコンテンツ macOS Sonoma 14.7.1 のリリース • macOS Sonoma 14.7.1 のセキュリティコンテンツ • Safari を 18.1.1 に更新 macOS Ventura 13.7.1 のリリース • macOS Ventura 13.7.1 のセキュリティコンテンツ • Safari を 18.1.1 に更新

リリース	変更

2024 年 10 月 28 日	すべての AMI
	• Homebrew を 4.4.2 に更新
	• aws-cli を 2.18.13 に更新
	• amazon-ssm-agent を 3.3.987.0 に更新
	• ec2-macos-init を 1.5.10 に更新
	• ec2-macos-utils を 1.0.4 に更新
	
	macOS Sequoia 15.0 のリリース
	• macOS Sequoia 15 のセキュリティコンテンツ
	
	macOS Sonoma 14.7 のリリース
	• macOS Sonoma 14.7 のセキュリティコンテンツ
	• コマンドラインツールを 16.0 に更新
	• Safari を 18.0.1 に更新
	  • Safari 18 のセキュリティコンテンツ
	
	macOS Ventura 13.7 のリリース
	• macOS Ventura 13.7 のセキュリティコンテンツ
	• Safari を 18.0.1 に更新
	  • Safari 18 のセキュリティコンテンツ

リリース	変更

2024 年 8 月 20 日	すべての AMI
	• Homebrew を 4.3.14 に更新
	• `aws-cli` を 2.17.29 に更新
	
	macOS Sonoma 14.6.1 のリリース
	• 公開された CVE エントリはありません。
	
	macOS Ventura 13.6.9 のリリース
	• 公開された CVE エントリはありません。
	• Safari を 17.6 に更新
	  • Safari 17.6 のセキュリティコンテンツ
	
	macOS Monterey 12.7.6 のリリース
	• macOS Monterey 12.7.6 のセキュリティコンテンツ
	• Safari を 17.6 に更新
	  • Safari 17.6 のセキュリティコンテンツ

リリース	変更

2024 年 6 月 7 日	すべての AMI
	• Homebrew を 4.3.1-1 に更新
	• `aws-cli` を 2.15.56 に更新
	• `amazon-ssm-agent` を 3.3.380.0-1 に更新
	
	macOS Sonoma 14.5 のリリース
	• macOS Sonoma 14.5 のセキュリティコンテンツ
	
	macOS Ventura 13.6.7 のリリース
	• macOS Ventura 13.6.7 のセキュリティコンテンツ
	• Safari を 17.5 に更新
	  • Safari 17.5 のセキュリティコンテンツ
	
	macOS Monterey 12.7.5 のリリース
	• macOS Monterey 12.7.5 のセキュリティコンテンツ
	• Safari を 17.5 に更新
	  • Safari 17.5 のセキュリティコンテンツ

リリース	変更

2024 年 4 月 12 日	すべての AMI
	• Homebrew を 4.2.16-1 に更新
	• `aws-cli` を 2.15.36 に更新
	
	macOS Sonoma 14.4.1 のリリース
	• macOS Sonoma 14.4.1 のセキュリティコンテンツ
	
	macOS Ventura 13.6.6 のリリース
	• macOS Ventura 13.6.6 のセキュリティコンテンツ
	• Safari を 17.4.1 に更新
	  • Safari 17.4.1 のセキュリティコンテンツ
	
	macOS Monterey
	• Safari を 17.4.1 に更新
	  • Safari 17.4.1 のセキュリティコンテンツ

Amazon EBS 最適化インスタンスタイプ

Amazon EBS 最適化インスタンスは、最適化された設定スタックを使用し、Amazon EBS I/O 用に専用のキャパシティを追加で提供します。このように最適化することで、Amazon EBS I/O と、インスタンスからのその他のトラフィックとの間の競合を最小に抑え、EBS ボリュームの最高のパフォーマンスを実現します。

汎用 SSD (gp2 および gp3) ボリュームを EBS 最適化インスタンスにアタッチすると、1 年で 99% の期間、プロビジョンド IOPS パフォーマンスの少なくとも 90% のボリュームが提供されます。また、プロビジョンド IOPS SSD (`io1` および `io2`) ボリュームでは、1 年で 99.9% の期間、プロビジョンド IOPS パフォーマンスの少なくとも 90% のボリュームが提供されます。スループット最適化 HDD `st1`および クール HDD `sc1`は、1 年で 99% の期間、想定されるスループットパフォーマンスの少なくとも 90% のボリュームを提供します。毎時間、予測合計スループットの 99% 達成を目

標に、準拠しない期間はほぼ均一に分散されています。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームの種類」を参照してください。

一部のインスタンスタイプはデフォルトで EBS 最適化されており、これを有効にする必要はなく、無効にしようとしてもできません。他のインスタンスタイプではオプションで EBS 最適化がサポートされており、起動時または起動後に時間単位の追加料金で有効にできます。一部のインスタンスタイプでは EBS 最適化がサポートされていません。

インスタンスタイプの詳細な仕様および機能については、「Amazon EC2 インスタンスタイプガイド」を参照してください。

考慮事項

- インスタンスの EBS パフォーマンスは、インスタンスタイプのパフォーマンス制限、またはアタッチされたボリュームの合計パフォーマンスのうち、どちらか小さい方によって制限されます。EBS のパフォーマンスを最大化するには、インスタンスにアタッチされたボリュームが合計でインスタンスの最大パフォーマンスと同等かそれ以上のパフォーマンスを発揮する必要があります。例えば、`r6i.16xlarge` で `80,000` IOPS を実現するには、インスタンスにそれぞれ `16,000` IOPS でプロビジョニングされた少なくとも 5 つの gp2 ボリューム (5 ボリューム x `16,000` IOPS = `80,000` IOPS) が必要ですが、`80,000` IOPS でプロビジョニングされた 1 つの gp3 ボリュームでもかまいません。アプリケーションのニーズよりも多い専用 Amazon EBS スループットを提供するインスタンスタイプを選択することをお勧めします。このようなインスタンスを選択しない場合、Amazon EBS と Amazon EC2 間の接続がパフォーマンスのボトルネックになる可能性があります。
- インスタンスにアタッチできる Amazon EBS ボリュームの最大数はインスタンスのタイプとサイズによって異なります。詳細については「Amazon EC2 インスタンスの Amazon EBS ボリューム制限」を参照してください。
- 最大 IOPS とスループットの制限は相互に依存します。I/O サイズによっては、一方の制限に達する場合があり、全体的なパフォーマンスに影響する可能性があります。最適な結果を得るには、ワークロードを計画する際に両方の制限を考慮してください。

デフォルトで EBS 最適化

次のインスタンスタイプは、デフォルトで EBS 最適化されています。EBS 最適化を有効にする必要はなく、EBS 最適化を無効にしてもその効果はなくなりません。

インスタンス

- 汎用

- コンピューティングの最適化
- メモリ最適化
- ストレージの最適化
- 高速コンピューティング
- 高性能コンピューティング

汎用

> **Note**
>
> M8a、M8g、M8gd、M8i、M8id、M8i-flex、M9g、M9gd のインスタンスタイプは、設定可能な帯域幅の重み付けをサポートします。これらのインスタンスタイプを使用すると、ネットワークパフォーマンスまたは Amazon EBS パフォーマンスのいずれかでインスタンスの帯域幅を最適化できます。次の表は、これらのインスタンスタイプのデフォルトの Amazon EBS 帯域幅パフォーマンスを示しています。詳細については「EC2 インスタンスの帯域幅の重み設定」を参照してください。

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
a1.medium [1]	300	3500	37.50	437.50	2500	20000
a1.large [1]	525	3500	65.62	437.50	4000	20000
a1.xlarge [1]	800	3500	100.00	437.50	6000	20000
a1.2xlarge [1]	1750	3500	218.75	437.50	10000	20000
a1.4xlarge [2]	3500		437.5		20000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
a1.metal [2]	3500		437.5		20000	
m4.large [2]	450		56.25		3600	
m4.xlarge [2]	750		93.75		6000	
m4.2xlarge [2]	1,000		125.0		8000	
m4.4xlarge [2]	2000		250.0		16000	
m4.10xlarge [2]	4000		500.0		32000	
m4.16xlarge [2]	10000		1250.0		65000	
m5.large [1]	650	4750	81.25	593.75	3600	18750
m5.xlarge [1]	1150	4750	143.75	593.75	6000	18750
m5.2xlarge [1]	2300	4750	287.50	593.75	12000	18750
m5.4xlarge [2]	4750		593.75		18750	
m5.8xlarge [2]	6800		850.0		30000	
m5.12xlarge [2]	9500		1187.5		40000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m5.16xlarge [2]	13600		1700.0		60000	
m5.24xlarge [2]	19000		2375.0		80000	
m5.metal [2]	19000		2375.0		80000	
m5a.large [1]	650	2880	81.25	360.00	3600	16000
m5a.xlarge [1]	1085	2880	135.62	360.00	6000	16000
m5a.2xlarge [1]	1580	2880	197.50	360.00	8333	16000
m5a.4xlarge [2]	2880		360.0		16000	
m5a.8xlarge [2]	4750		593.75		20000	
m5a.12xlarge [2]	6780		847.5		30000	
m5a.16xlarge [2]	9500		1187.5		40000	
m5a.24xlarge [2]	13750		1718.75		60000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m5ad.large [1]	650	2880	81.25	360.00	3600	16000
m5ad.xlarge [1]	1085	2880	135.62	360.00	6000	16000
m5ad.2xlarge [1]	1580	2880	197.50	360.00	8333	16000
m5ad.4xlarge [2]	2880		360.0		16000	
m5ad.8xlarge [2]	4750		593.75		20000	
m5ad.12xlarge [2]	6780		847.5		30000	
m5ad.16xlarge [2]	9500		1187.5		40000	
m5ad.24xlarge [2]	13750		1718.75		60000	
m5d.large [1]	650	4750	81.25	593.75	3600	18750
m5d.xlarge [1]	1150	4750	143.75	593.75	6000	18750
m5d.2xlarge [1]	2300	4750	287.50	593.75	12000	18750

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m5d.4xlarge [2]	4750		593.75		18750	
m5d.8xlarge [2]	6800		850.0		30000	
m5d.12xlarge [2]	9500		1187.5		40000	
m5d.16xlarge [2]	13600		1700.0		60000	
m5d.24xlarge [2]	19000		2375.0		80000	
m5d.metal [2]	19000		2375.0		80000	
m5dn.large [1]	650	4750	81.25	593.75	3600	18750
m5dn.xlarge [1]	1150	4750	143.75	593.75	6000	18750
m5dn.2xlarge [1]	2300	4750	287.50	593.75	12000	18750
m5dn.4xlarge [2]	4750		593.75		18750	
m5dn.8xlarge [2]	6800		850.0		30000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m5dn.12xlarge [2]	9500		1187.5		40000	
m5dn.16xlarge [2]	13600		1700.0		60000	
m5dn.24xlarge [2]	19000		2375.0		80000	
m5dn.metal [2]	19000		2375.0		80000	
m5n.large [1]	650	4750	81.25	593.75	3600	18750
m5n.xlarge [1]	1150	4750	143.75	593.75	6000	18750
m5n.2xlarge [1]	2300	4750	287.50	593.75	12000	18750
m5n.4xlarge [2]	4750		593.75		18750	
m5n.8xlarge [2]	6800		850.0		30000	
m5n.12xlarge [2]	9500		1187.5		40000	
m5n.16xlarge [2]	13600		1700.0		60000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m5n.24xlarge [2]	19000		2375.0		80000	
m5n.metal [2]	19000		2375.0		80000	
m5zn.large [1]	800	3170	100.00	396.25	3333	13333
m5zn.xlarge [1]	1564	3170	195.50	396.25	6667	13333
m5zn.2xlarge [2]	3170		396.25		13333	
m5zn.3xlarge [2]	4750		593.75		20000	
m5zn.6xlarge [2]	9500		1187.5		40000	
m5zn.12xlarge [2]	19000		2375.0		80000	
m5zn.metal [2]	19000		2375.0		80000	
m6a.large [1]	650	10000	81.25	1250.00	3600	40000
m6a.xlarge [1]	1250	10000	156.25	1250.00	6000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m6a.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m6a.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m6a.8xlarge [2]	10000		1250.0		40000	
m6a.12xlarge [2]	15000		1875.0		60000	
m6a.16xlarge [2]	20000		2500.0		80000	
m6a.24xlarge [2]	30000		3750.0		120000	
m6a.32xlarge [2]	40000		5000.0		160000	
m6a.48xlarge [2]	40000		5000.0		240000	
m6a.metal [2]	40000		5000.0		240000	
m6g.medium [1]	315	4750	39.38	593.75	2500	20000
m6g.large [1]	630	4750	78.75	593.75	3600	20000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m6g.xlarge [1]	1188	4750	148.50	593.75	6000	20000
m6g.2xlarge [1]	2375	4750	296.88	593.75	12000	20000
m6g.4xlarge [2]	4750		593.75		20000	
m6g.8xlarge [2]	9500		1187.5		40000	
m6g.12xlarge [2]	14250		1781.25		50000	
m6g.16xlarge [2]	19000		2375.0		80000	
m6g.metal [2]	19000		2375.0		80000	
m6gd.medium [1]	315	4750	39.38	593.75	2500	20000
m6gd.large [1]	630	4750	78.75	593.75	3600	20000
m6gd.xlarge [1]	1188	4750	148.50	593.75	6000	20000
m6gd.2xlarge [1]	2375	4750	296.88	593.75	12000	20000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m6gd.4xlarge [2]	4750		593.75		20000	
m6gd.8xlarge [2]	9500		1187.5		40000	
m6gd.12xlarge [2]	14250		1781.25		50000	
m6gd.16xlarge [2]	19000		2375.0		80000	
m6gd.metal [2]	19000		2375.0		80000	
m6i.large [1]	650	10000	81.25	1250.00	3600	40000
m6i.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m6i.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m6i.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m6i.8xlarge [2]	10000		1250.0		40000	
m6i.12xlarge [2]	15000		1875.0		60000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m6i.16xlarge [2]	20000		2500.0		80000	
m6i.24xlarge [2]	30000		3750.0		120000	
m6i.32xlarge [2]	40000		5000.0		160000	
m6i.metal [2]	40000		5000.0		160000	
m6id.large [1]	650	10000	81.25	1250.00	3600	40000
m6id.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m6id.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m6id.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m6id.8xlarge [2]	10000		1250.0		40000	
m6id.12xlarge [2]	15000		1875.0		60000	
m6id.16xlarge [2]	20000		2500.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m6id.24xlarge [2]	30000		3750.0		120000	
m6id.32xlarge [2]	40000		5000.0		160000	
m6id.metal [2]	40000		5000.0		160000	
m6idn.large [1]	1562	25000	195.31	3125.00	6250	100000
m6idn.xlarge [1]	3125	25000	390.62	3125.00	12500	100000
m6idn.2xlarge [1]	6250	25000	781.25	3125.00	25000	100000
m6idn.4xlarge [1]	12500	25000	1562.50	3125.00	50000	100000
m6idn.8xlarge [2]	25000		3125.0		100000	
m6idn.12xlarge [2]	37500		4687.5		150000	
m6idn.16xlarge [2]	50000		6250.0		200000	
m6idn.24xlarge [2]	75000		9375.0		300000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m6idn.32xlarge [2]	100000		12500.0		400000	
m6idn.metal [2]	100000		12500.0		400000	
m6in.large [1]	1562	25000	195.31	3125.00	6250	100000
m6in.xlarge [1]	3125	25000	390.62	3125.00	12500	100000
m6in.2xlarge [1]	6250	25000	781.25	3125.00	25000	100000
m6in.4xlarge [1]	12500	25000	1562.50	3125.00	50000	100000
m6in.8xlarge [2]	25000		3125.0		100000	
m6in.12xlarge [2]	37500		4687.5		150000	
m6in.16xlarge [2]	50000		6250.0		200000	
m6in.24xlarge [2]	75000		9375.0		300000	
m6in.32xlarge [2]	100000		12500.0		400000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m6in.metal [2]	100000		12500.0		400000	
m7a.medium [1]	325	10000	40.62	1250.00	2500	40000
m7a.large [1]	650	10000	81.25	1250.00	3600	40000
m7a.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m7a.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m7a.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m7a.8xlarge [2]	10000		1250.0		40000	
m7a.12xlarge [2]	15000		1875.0		60000	
m7a.16xlarge [2]	20000		2500.0		80000	
m7a.24xlarge [2]	30000		3750.0		120000	
m7a.32xlarge [2]	40000		5000.0		160000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m7a.48xlarge [2]	40000		5000.0		240000	
m7a.metal-48xl [2]	40000		5000.0		240000	
m7g.medium [1]	315	10000	39.38	1250.00	2500	40000
m7g.large [1]	630	10000	78.75	1250.00	3600	40000
m7g.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m7g.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m7g.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m7g.8xlarge [2]	10000		1250.0		40000	
m7g.12xlarge [2]	15000		1875.0		60000	
m7g.16xlarge [2]	20000		2500.0		80000	
m7g.metal [2]	20000		2500.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m7gd.medium [1]	315	10000	39.38	1250.00	2500	40000
m7gd.large [1]	630	10000	78.75	1250.00	3600	40000
m7gd.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m7gd.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m7gd.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m7gd.8xlarge [2]	10000		1250.0		40000	
m7gd.12xlarge [2]	15000		1875.0		60000	
m7gd.16xlarge [2]	20000		2500.0		80000	
m7gd.metal [2]	20000		2500.0		80000	
m7i.large [1]	650	10000	81.25	1250.00	3600	40000
m7i.xlarge [1]	1250	10000	156.25	1250.00	6000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m7i.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m7i.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m7i.8xlarge [2]	10000		1250.0		40000	
m7i.12xlarge [2]	15000		1875.0		60000	
m7i.16xlarge [2]	20000		2500.0		80000	
m7i.24xlarge [2]	30000		3750.0		120000	
m7i.48xlarge [2]	40000		5000.0		240000	
m7i.metal-24xl [2]	30000		3750.0		120000	
m7i.metal-48xl [2]	40000		5000.0		240000	
m7i-flex.large [1]	312	10000	39.06	1250.00	2500	40000
m7i-flex.xlarge [1]	625	10000	78.12	1250.00	3600	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m7i-flex.2xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m7i-flex.4xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m7i-flex.8xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m7i-flex.12xlarge [1]	7500	15000	937.50	1875.00	30000	60000
m7i-flex.16xlarge [1]	10000	20000	1250.00	2500.00	40000	80000
m8a.medium [1]	325	10000	40.62	1250.00	2500	40000
m8a.large [1]	650	10000	81.25	1250.00	3600	40000
m8a.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m8a.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m8a.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m8a.8xlarge [2]	10000		1250.0		40000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m8a.12xlarge [2]	15000		1875.0		60000	
m8a.16xlarge [2]	20000		2500.0		80000	
m8a.24xlarge [2]	30000		3750.0		120000	
m8a.48xlarge [2]	60000		7500.0		240000	
m8a.metal-24xl [2]	30000		3750.0		120000	
m8a.metal-48xl [2]	60000		7500.0		240000	
m8azn.medium [1]	625	15000	78.12	1875.00	2500	60000
m8azn.large [1]	1250	15000	156.25	1875.00	5000	60000
m8azn.xlarge [1]	2500	15000	312.50	1875.00	10000	60000
m8azn.3xlarge [1]	7500	15000	937.50	1875.00	30000	60000
m8azn.6xlarge [2]	15000		1875.0		60000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m8azn.12xlarge [2]	30000		3750.0		120000	
m8azn.24xlarge [2]	60000		7500.0		240000	
m8azn.metal-12xl [2]	30000		3750.0		120000	
m8azn.metal-24xl [2]	60000		7500.0		240000	
m8g.medium [1]	315	10000	39.38	1250.00	2500	40000
m8g.large [1]	630	10000	78.75	1250.00	3600	40000
m8g.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m8g.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m8g.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m8g.8xlarge [2]	10000		1250.0		40000	
m8g.12xlarge [2]	15000		1875.0		60000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m8g.16xlarge [2]	20000		2500.0		80000	
m8g.24xlarge [2]	30000		3750.0		120000	
m8g.48xlarge [2]	40000		5000.0		240000	
m8g.metal-24xl [2]	30000		3750.0		120000	
m8g.metal-48xl [2]	40000		5000.0		240000	
m8gb.medium [1]	1562	25000	195.31	3125.00	7500	120000
m8gb.large [1]	3125	25000	390.62	3125.00	15000	120000
m8gb.xlarge [1]	6250	25000	781.25	3125.00	30000	120000
m8gb.2xlarge [1]	12500	25000	1562.50	3125.00	60000	120000
m8gb.4xlarge [2]	25000		3125.0		120000	
m8gb.8xlarge [2]	50000		6250.0		240000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m8gb.12xlarge [2]	75000		9375.0		360000	
m8gb.16xlarge [2]	100000		12500.0		480000	
m8gb.24xlarge [2]	150000		18750.0		720000	
m8gb.48xlarge [2]	300000		37500.0		1440000	
m8gb.metal-24xl [2]	150000		18750.0		720000	
m8gb.metal-48xl [2]	300000		37500.0		1440000	
m8gd.medium [1]	315	10000	39.38	1250.00	2500	40000
m8gd.large [1]	630	10000	78.75	1250.00	3600	40000
m8gd.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m8gd.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m8gd.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m8gd.8xlarge [2]	10000		1250.0		40000	
m8gd.12xlarge [2]	15000		1875.0		60000	
m8gd.16xlarge [2]	20000		2500.0		80000	
m8gd.24xlarge [2]	30000		3750.0		120000	
m8gd.48xlarge [2]	40000		5000.0		240000	
m8gd.metal-24xl [2]	30000		3750.0		120000	
m8gd.metal-48xl [2]	40000		5000.0		240000	
m8gn.medium [1]	760	10000	95.00	1250.00	2500	40000
m8gn.large [1]	1250	10000	156.25	1250.00	5000	40000
m8gn.xlarge [1]	2500	10000	312.50	1250.00	10000	40000
m8gn.2xlarge [1]	5000	10000	625.00	1250.00	20000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m8gn.4xlarge [2]	10000		1250.0		40000	
m8gn.8xlarge [2]	20000		2500.0		80000	
m8gn.12xlarge [2]	30000		3750.0		120000	
m8gn.16xlarge [2]	40000		5000.0		160000	
m8gn.24xlarge [2]	60000		7500.0		240000	
m8gn.48xlarge [2]	120000		15000.0		480000	
m8gn.metal-24xl [2]	60000		7500.0		240000	
m8gn.metal-48xl [2]	120000		15000.0		480000	
m8i.large [1]	650	10000	81.25	1250.00	3600	40000
m8i.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m8i.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m8i.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m8i.8xlarge [2]	10000		1250.0		40000	
m8i.12xlarge [2]	15000		1875.0		60000	
m8i.16xlarge [2]	20000		2500.0		80000	
m8i.24xlarge [2]	30000		3750.0		120000	
m8i.32xlarge [2]	40000		5000.0		160000	
m8i.48xlarge [2]	60000		7500.0		240000	
m8i.96xlarge [2]	80000		10000.0		480000	
m8i.metal-48xl [2]	60000		7500.0		240000	
m8i.metal-96xl [2]	80000		10000.0		480000	
m8id.large [1]	650	10000	81.25	1250.00	3600	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m8id.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m8id.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m8id.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m8id.8xlarge [2]	10000		1250.0		40000	
m8id.12xlarge [2]	15000		1875.0		60000	
m8id.16xlarge [2]	20000		2500.0		80000	
m8id.24xlarge [2]	30000		3750.0		120000	
m8id.32xlarge [2]	40000		5000.0		160000	
m8id.48xlarge [2]	60000		7500.0		240000	
m8id.96xlarge [2]	80000		10000.0		480000	
m8id.metal-48xl [2]	60000		7500.0		240000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m8id.metal-96xl [2]	80000		10000.0		480000	
m8i-flex.large [1]	315	10000	39.38	1250.00	2500	40000
m8i-flex.xlarge [1]	630	10000	78.75	1250.00	3600	40000
m8i-flex.2xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m8i-flex.4xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m8i-flex.8xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m8i-flex.12xlarge [1]	7500	15000	937.50	1875.00	30000	60000
m8i-flex.16xlarge [1]	10000	20000	1250.00	2500.00	40000	80000
m8in.large [1]	650	10000	81.25	1250.00	3600	40000
m8in.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m8in.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m8in.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m8in.8xlarge [2]	10000		1250.0		40000	
m8in.12xlarge [2]	15000		1875.0		60000	
m8in.16xlarge [2]	20000		2500.0		80000	
m8in.24xlarge [2]	30000		3750.0		120000	
m8in.32xlarge [2]	40000		5000.0		160000	
m8in.48xlarge [2]	60000		7500.0		240000	
m8in.96xlarge [2]	120000		15000.0		480000	
m8in.metal-48xl [2]	60000		7500.0		240000	
m8in.metal-96xl [2]	120000		15000.0		480000	
m8idn.large [1]	650	10000	81.25	1250.00	3600	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m8idn.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m8idn.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m8idn.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m8idn.8xlarge [2]	10000		1250.0		40000	
m8idn.12xlarge [2]	15000		1875.0		60000	
m8idn.16xlarge [2]	20000		2500.0		80000	
m8idn.24xlarge [2]	30000		3750.0		120000	
m8idn.32xlarge [2]	40000		5000.0		160000	
m8idn.48xlarge [2]	60000		7500.0		240000	
m8idn.96xlarge [2]	120000		15000.0		480000	
m8idn.metal-48xl [2]	60000		7500.0		240000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m8idn.metal-96xl [2]	120000		15000.0		480000	
m8ine.large [1]	650	10000	81.25	1250.00	3600	40000
m8ine.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m8ine.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m8ine.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m8ine.8xlarge [2]	10000		1250.0		40000	
m8ine.12xlarge [2]	15000		1875.0		60000	
m8ib.large [1]	1563	25000	195.38	3125.00	7500	120000
m8ib.xlarge [1]	3125	25000	390.62	3125.00	15000	120000
m8ib.2xlarge [1]	6250	25000	781.25	3125.00	30000	120000
m8ib.4xlarge [1]	12500	25000	1562.50	3125.00	60000	120000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m8ib.8xlarge [2]	25000		3125.0		120000	
m8ib.12xlarge [2]	37500		4687.5		180000	
m8ib.16xlarge [2]	50000		6250.0		240000	
m8ib.24xlarge [2]	75000		9375.0		360000	
m8ib.32xlarge [2]	100000		12500.0		480000	
m8ib.48xlarge [2]	150000		18750.0		720000	
m8ib.96xlarge [2]	300000		37500.0		1440000	
m8ib.metal-48xl [2]	150000		18750.0		720000	
m8ib.metal-96xl [2]	300000		37500.0		1440000	
m8idb.large [1]	1563	25000	195.38	3125.00	7500	120000
m8idb.xlarge [1]	3125	25000	390.62	3125.00	15000	120000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m8idb.2xlarge [1]	6250	25000	781.25	3125.00	30000	120000
m8idb.4xlarge [1]	12500	25000	1562.50	3125.00	60000	120000
m8idb.8xlarge [2]	25000		3125.0		120000	
m8idb.12xlarge [2]	37500		4687.5		180000	
m8idb.16xlarge [2]	50000		6250.0		240000	
m8idb.24xlarge [2]	75000		9375.0		360000	
m8idb32xlarge [2]	100000		12500.0		480000	
m8idb.48xlarge [2]	150000		18750.0		720000	
m8idb.96xlarge [2]	300000		37500.0		1440000	
m8idb.metal-48xl [2]	150000		18750.0		720000	
m8idb.metal-96xl [2]	300000		37500.0		1440000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m9g.medium [1]	380	12000	47.50	1500.00	2500	48000
m9g.large [1]	760	12000	95.00	1500.00	3600	48000
m9g.xlarge [1]	1500	12000	187.50	1500.00	6000	48000
m9g.2xlarge [1]	3000	12000	375.00	1500.00	12000	48000
m9g.4xlarge [1]	6000	12000	750.00	1500.00	24000	48000
m9g.8xlarge [2]	12000		1500.0		48000	
m9g.12xlarge [2]	18000		2250.0		72000	
m9g.16xlarge [2]	24000		3000.0		96000	
m9g.24xlarge [2]	36000		4500.0		144000	
m9g.48xlarge [2]	72000		9000.0		288000	
m9g.metal-48xl [2]	72000		9000.0		288000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m9gd.medium [1]	380	12000	47.50	1500.00	2500	48000
m9gd.large [1]	760	12000	95.00	1500.00	3600	48000
m9gd.xlarge [1]	1500	12000	187.50	1500.00	6000	48000
m9gd.2xlarge [1]	3000	12000	375.00	1500.00	12000	48000
m9gd.4xlarge [1]	6000	12000	750.00	1500.00	24000	48000
m9gd.8xlarge [2]	12000		1500.0		48000	
m9gd.12xlarge [2]	18000		2250.0		72000	
m9gd.16xlarge [2]	24000		3000.0		96000	
m9gd.24xlarge [2]	36000		4500.0		144000	
m9gd.48xlarge [2]	72000		9000.0		288000	
m9gd.metal-48xl [2]	72000		9000.0		288000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
mac1.metal [2]	14000		1750.0		80000	
mac2.metal [2]	10000		1250.0		55000	
mac2-m1ultra.metal [2]	10000		1250.0		55000	
mac2-m2.metal [2]	8000		1000.0		55000	
mac2-m2pro.metal [2]	8000		1000.0		55000	
mac-m4.metal [2]	8000		1000.0		55000	
mac-m4pro.metal [2]	8000		1000.0		55000	
mac-m4max.metal [2]	10000		1250.0		55000	
t3.nano [1]	43	2085	5.38	260.62	250	11800
t3.micro [1]	87	2085	10.88	260.62	500	11800
t3.small [1]	174	2085	21.75	260.62	1,000	11800

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
t3.medium [1]	347	2085	43.38	260.62	2000	11800
t3.large [1]	695	2780	86.88	347.50	4000	15700
t3.xlarge [1]	695	2780	86.88	347.50	4000	15700
t3.2xlarge [1]	695	2780	86.88	347.50	4000	15700
t3a.nano [1]	45	2085	5.62	260.62	250	11800
t3a.micro [1]	90	2085	11.25	260.62	500	11800
t3a.small [1]	175	2085	21.88	260.62	1,000	11800
t3a.medium [1]	350	2085	43.75	260.62	2000	11800
t3a.large [1]	695	2780	86.88	347.50	4000	15700
t3a.xlarge [1]	695	2780	86.88	347.50	4000	15700
t3a.2xlarge [1]	695	2780	86.88	347.50	4000	15700
t4g.nano [1]	43	2085	5.38	260.62	250	11800
t4g.micro [1]	87	2085	10.88	260.62	500	11800
t4g.small [1]	174	2085	21.75	260.62	1,000	11800
t4g.medium [1]	347	2085	43.38	260.62	2000	11800

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
t4g.large [1]	695	2780	86.88	347.50	4000	15700
t4g.xlarge [1]	695	2780	86.88	347.50	4000	15700
t4g.2xlarge [1]	695	2780	86.88	347.50	4000	15700

[1] これらのインスタンスは、最大パフォーマンスを 24 時間ごとに少なくとも 1 回は 30 分間維持することができ、その後はベースラインのパフォーマンスに戻ります。

[2] これらのインスタンスは、記載されているパフォーマンスを無期限に維持することができます。ワークロードで、最大パフォーマンスを 30 分以上維持する必要がある場合は、これらのインスタンスの中から 1 つ使用します。

コンピューティングの最適化

> **Note**
>
> C8a、C8g、C8gd、C8i、C8id、C8i-flex、C9g、C9gd のインスタンスタイプは、設定可能な帯域幅の重み付けをサポートします。これらのインスタンスタイプを使用すると、ネットワークパフォーマンスまたは Amazon EBS パフォーマンスのいずれかでインスタンスの帯域幅を最適化できます。次の表は、これらのインスタンスタイプのデフォルトの Amazon EBS 帯域幅パフォーマンスを示しています。詳細については「EC2 インスタンスの帯域幅の重み設定」を参照してください。

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c4.large [2]	500		62.5		4000	
c4.xlarge [2]	750		93.75		6000	
c4.2xlarge [2]	1,000		125.0		8000	
c4.4xlarge [2]	2000		250.0		16000	
c4.8xlarge [2]	4000		500.0		32000	
c5.large [1]	650	4750	81.25	593.75	4000	20000
c5.xlarge [1]	1150	4750	143.75	593.75	6000	20000
c5.2xlarge [1]	2300	4750	287.50	593.75	10000	20000
c5.4xlarge [2]	4750		593.75		20000	
c5.9xlarge [2]	9500		1187.5		40000	
c5.12xlarge [2]	9500		1187.5		40000	
c5.18xlarge [2]	19000		2375.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c5.24xlarge [2]	19000		2375.0		80000	
c5.metal [2]	19000		2375.0		80000	
c5a.large [1]	200	3170	25.00	396.25	800	13300
c5a.xlarge [1]	400	3170	50.00	396.25	1600	13300
c5a.2xlarge [1]	800	3170	100.00	396.25	3200	13300
c5a.4xlarge [1]	1580	3170	197.50	396.25	6600	13300
c5a.8xlarge [2]	3170		396.25		13300	
c5a.12xlarge [2]	4750		593.75		20000	
c5a.16xlarge [2]	6300		787.5		26700	
c5a.24xlarge [2]	9500		1187.5		40000	
c5ad.large [1]	200	3170	25.00	396.25	800	13300

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c5ad.xlarge [1]	400	3170	50.00	396.25	1600	13300
c5ad.2xlarge [1]	800	3170	100.00	396.25	3200	13300
c5ad.4xlarge [1]	1580	3170	197.50	396.25	6600	13300
c5ad.8xlarge [2]	3170		396.25		13300	
c5ad.12xlarge [2]	4750		593.75		20000	
c5ad.16xlarge [2]	6300		787.5		26700	
c5ad.24xlarge [2]	9500		1187.5		40000	
c5d.large [1]	650	4750	81.25	593.75	4000	20000
c5d.xlarge [1]	1150	4750	143.75	593.75	6000	20000
c5d.2xlarge [1]	2300	4750	287.50	593.75	10000	20000
c5d.4xlarge [2]	4750		593.75		20000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c5d.9xlarge [2]	9500		1187.5		40000	
c5d.12xlarge [2]	9500		1187.5		40000	
c5d.18xlarge [2]	19000		2375.0		80000	
c5d.24xlarge [2]	19000		2375.0		80000	
c5d.metal [2]	19000		2375.0		80000	
c5n.large [1]	650	4750	81.25	593.75	4000	20000
c5n.xlarge [1]	1150	4750	143.75	593.75	6000	20000
c5n.2xlarge [1]	2300	4750	287.50	593.75	10000	20000
c5n.4xlarge [2]	4750		593.75		20000	
c5n.9xlarge [2]	9500		1187.5		40000	
c5n.18xlarge [2]	19000		2375.0		80000	
c5n.metal [2]	19000		2375.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c6a.large [1]	650	10000	81.25	1250.00	3600	40000
c6a.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c6a.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
c6a.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c6a.8xlarge [2]	10000		1250.0		40000	
c6a.12xlarge [2]	15000		1875.0		60000	
c6a.16xlarge [2]	20000		2500.0		80000	
c6a.24xlarge [2]	30000		3750.0		120000	
c6a.32xlarge [2]	40000		5000.0		160000	
c6a.48xlarge [2]	40000		5000.0		240000	
c6a.metal [2]	40000		5000.0		240000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c6g.medium [1]	315	4750	39.38	593.75	2500	20000
c6g.large [1]	630	4750	78.75	593.75	3600	20000
c6g.xlarge [1]	1188	4750	148.50	593.75	6000	20000
c6g.2xlarge [1]	2375	4750	296.88	593.75	12000	20000
c6g.4xlarge [2]	4750		593.75		20000	
c6g.8xlarge [2]	9500		1187.5		40000	
c6g.12xlarge [2]	14250		1781.25		50000	
c6g.16xlarge [2]	19000		2375.0		80000	
c6g.metal [2]	19000		2375.0		80000	
c6gd.medium [1]	315	4750	39.38	593.75	2500	20000
c6gd.large [1]	630	4750	78.75	593.75	3600	20000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c6gd.xlarge [1]	1188	4750	148.50	593.75	6000	20000
c6gd.2xlarge [1]	2375	4750	296.88	593.75	12000	20000
c6gd.4xlarge [2]	4750		593.75		20000	
c6gd.8xlarge [2]	9500		1187.5		40000	
c6gd.12xlarge [2]	14250		1781.25		50000	
c6gd.16xlarge [2]	19000		2375.0		80000	
c6gd.metal [2]	19000		2375.0		80000	
c6gn.medium [1]	760	9500	95.00	1187.50	2500	40000
c6gn.large [1]	1235	9500	154.38	1187.50	5000	40000
c6gn.xlarge [1]	2375	9500	296.88	1187.50	10000	40000
c6gn.2xlarge [1]	4750	9500	593.75	1187.50	20000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c6gn.4xlarge [2]	9500		1187.5		40000	
c6gn.8xlarge [2]	19000		2375.0		80000	
c6gn.12xlarge [2]	28500		3562.5		120000	
c6gn.16xlarge [2]	38000		4750.0		160000	
c6i.large [1]	650	10000	81.25	1250.00	3600	40000
c6i.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c6i.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
c6i.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c6i.8xlarge [2]	10000		1250.0		40000	
c6i.12xlarge [2]	15000		1875.0		60000	
c6i.16xlarge [2]	20000		2500.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c6i.24xlarge [2]	30000		3750.0		120000	
c6i.32xlarge [2]	40000		5000.0		160000	
c6i.metal [2]	40000		5000.0		160000	
c6id.large [1]	650	10000	81.25	1250.00	3600	40000
c6id.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c6id.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
c6id.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c6id.8xlarge [2]	10000		1250.0		40000	
c6id.12xlarge [2]	15000		1875.0		60000	
c6id.16xlarge [2]	20000		2500.0		80000	
c6id.24xlarge [2]	30000		3750.0		120000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c6id.32xlarge [2]	40000		5000.0		160000	
c6id.metal [2]	40000		5000.0		160000	
c6in.large [1]	1562	25000	195.31	3125.00	6250	100000
c6in.xlarge [1]	3125	25000	390.62	3125.00	12500	100000
c6in.2xlarge [1]	6250	25000	781.25	3125.00	25000	100000
c6in.4xlarge [1]	12500	25000	1562.50	3125.00	50000	100000
c6in.8xlarge [2]	25000		3125.0		100000	
c6in.12xlarge [2]	37500		4687.5		150000	
c6in.16xlarge [2]	50000		6250.0		200000	
c6in.24xlarge [2]	75000		9375.0		300000	
c6in.32xlarge [2]	100000		12500.0		400000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c6in.metal [2]	100000		12500.0		400000	
c7a.medium [1]	325	10000	40.62	1250.00	2500	40000
c7a.large [1]	650	10000	81.25	1250.00	3600	40000
c7a.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c7a.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
c7a.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c7a.8xlarge [2]	10000		1250.0		40000	
c7a.12xlarge [2]	15000		1875.0		60000	
c7a.16xlarge [2]	20000		2500.0		80000	
c7a.24xlarge [2]	30000		3750.0		120000	
c7a.32xlarge [2]	40000		5000.0		160000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c7a.48xlarge [2]	40000		5000.0		240000	
c7a.metal-48xl [2]	40000		5000.0		240000	
c7g.medium [1]	315	10000	39.38	1250.00	2500	40000
c7g.large [1]	630	10000	78.75	1250.00	3600	40000
c7g.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c7g.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
c7g.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c7g.8xlarge [2]	10000		1250.0		40000	
c7g.12xlarge [2]	15000		1875.0		60000	
c7g.16xlarge [2]	20000		2500.0		80000	
c7g.metal [2]	20000		2500.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c7gd.medium [1]	315	10000	39.38	1250.00	2500	40000
c7gd.large [1]	630	10000	78.75	1250.00	3600	40000
c7gd.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c7gd.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
c7gd.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c7gd.8xlarge [2]	10000		1250.0		40000	
c7gd.12xlarge [2]	15000		1875.0		60000	
c7gd.16xlarge [2]	20000		2500.0		80000	
c7gd.metal [2]	20000		2500.0		80000	
c7gn.medium [1]	521	10000	65.12	1250.00	2083	40000
c7gn.large [1]	1042	10000	130.25	1250.00	4167	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c7gn.xlarge [1]	2083	10000	260.38	1250.00	8333	40000
c7gn.2xlarge [1]	4167	10000	520.88	1250.00	16667	40000
c7gn.4xlarge [1]	8333	10000	1041.62	1250.00	33333	40000
c7gn.8xlarge [1]	16667	20000	2083.38	2500.00	66667	80000
c7gn.12xlarge [1]	25000	30000	3125.00	3750.00	100000	120000
c7gn.16xlarge [1]	33333	40000	4166.62	5000.00	133333	160000
c7gn.metal [1]	33333	40000	4166.62	5000.00	133333	160000
c7i.large [1]	650	10000	81.25	1250.00	3600	40000
c7i.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c7i.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
c7i.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c7i.8xlarge [2]	10000		1250.0		40000	
c7i.12xlarge [2]	15000		1875.0		60000	
c7i.16xlarge [2]	20000		2500.0		80000	
c7i.24xlarge [2]	30000		3750.0		120000	
c7i.48xlarge [2]	40000		5000.0		240000	
c7i.metal-24xl [2]	30000		3750.0		120000	
c7i.metal-48xl [2]	40000		5000.0		240000	
c7i-flex.large [1]	312	10000	39.06	1250.00	2500	40000
c7i-flex.xlarge [1]	625	10000	78.12	1250.00	3600	40000
c7i-flex.2xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c7i-flex.4xlarge [1]	2500	10000	312.50	1250.00	12000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c7i-flex.8xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c7i-flex.12xlarge [1]	7500	15000	937.50	1875.00	30000	60000
c7i-flex.16xlarge [1]	10000	20000	1250.00	2500.00	40000	80000
c8a.medium [1]	325	10000	40.62	1250.00	2500	40000
c8a.large [1]	650	10000	81.25	1250.00	3600	40000
c8a.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c8a.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
c8a.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c8a.8xlarge [2]	10000		1250.0		40000	
c8a.12xlarge [2]	15000		1875.0		60000	
c8a.16xlarge [2]	20000		2500.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c8a.24xlarge [2]	30000		3750.0		120000	
c8a.48xlarge [2]	60000		7500.0		240000	
c8a.metal-24xl [2]	30000		3750.0		120000	
c8a.metal-48xl [2]	60000		7500.0		240000	
c8g.medium [1]	315	10000	39.38	1250.00	2500	40000
c8g.large [1]	630	10000	78.75	1250.00	3600	40000
c8g.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c8g.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
c8g.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c8g.8xlarge [2]	10000		1250.0		40000	
c8g.12xlarge [2]	15000		1875.0		60000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c8g.16xlarge [2]	20000		2500.0		80000	
c8g.24xlarge [2]	30000		3750.0		120000	
c8g.48xlarge [2]	40000		5000.0		240000	
c8g.metal-24xl [2]	30000		3750.0		120000	
c8g.metal-48xl [2]	40000		5000.0		240000	
c8gb.medium [1]	1562	25000	195.31	3125.00	7500	120000
c8gb.large [1]	3125	25000	390.62	3125.00	15000	120000
c8gb.xlarge [1]	6250	25000	781.25	3125.00	30000	120000
c8gb.2xlarge [1]	12500	25000	1562.50	3125.00	60000	120000
c8gb.4xlarge [2]	25000		3125.0		120000	
c8gb.8xlarge [2]	50000		6250.0		240000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c8gb.12xlarge [2]	75000		9375.0		360000	
c8gb.16xlarge [2]	100000		12500.0		480000	
c8gb.24xlarge [2]	150000		18750.0		720000	
c8gb.48xlarge [2]	300000		37500.0		1440000	
c8gb.metal-24xl [2]	150000		18750.0		720000	
c8gb.metal-48xl [2]	300000		37500.0		1440000	
c8gd.medium [1]	315	10000	39.38	1250.00	2500	40000
c8gd.large [1]	630	10000	78.75	1250.00	3600	40000
c8gd.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c8gd.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
c8gd.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c8gd.8xlarge [2]	10000		1250.0		40000	
c8gd.12xlarge [2]	15000		1875.0		60000	
c8gd.16xlarge [2]	20000		2500.0		80000	
c8gd.24xlarge [2]	30000		3750.0		120000	
c8gd.48xlarge [2]	40000		5000.0		240000	
c8gd.metal-24xl [2]	30000		3750.0		120000	
c8gd.metal-48xl [2]	40000		5000.0		240000	
c8gn.medium [1]	760	10000	95.00	1250.00	2500	40000
c8gn.large [1]	1250	10000	156.25	1250.00	5000	40000
c8gn.xlarge [1]	2500	10000	312.50	1250.00	10000	40000
c8gn.2xlarge [1]	5000	10000	625.00	1250.00	20000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c8gn.4xlarge [2]	10000		1250.0		40000	
c8gn.8xlarge [2]	20000		2500.0		80000	
c8gn.12xlarge [2]	30000		3750.0		120000	
c8gn.16xlarge [2]	40000		5000.0		160000	
c8gn.24xlarge [2]	60000		7500.0		240000	
c8gn.48xlarge [2]	120000		15000.0		480000	
c8gn.metal-24xl [2]	60000		7500.0		240000	
c8gn.metal-48xl [2]	120000		15000.0		480000	
c8i.large [1]	650	10000	81.25	1250.00	3600	40000
c8i.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c8i.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c8i.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c8i.8xlarge [2]	10000		1250.0		40000	
c8i.12xlarge [2]	15000		1875.0		60000	
c8i.16xlarge [2]	20000		2500.0		80000	
c8i.24xlarge [2]	30000		3750.0		120000	
c8i.32xlarge [2]	40000		5000.0		160000	
c8i.48xlarge [2]	60000		7500.0		240000	
c8i.96xlarge [2]	80000		10000.0		480000	
c8i.metal-48xl [2]	60000		7500.0		240000	
c8i.metal-96xl [2]	80000		10000.0		480000	
c8id.large [1]	650	10000	81.25	1250.00	3600	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c8id.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c8id.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
c8id.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c8id.8xlarge [2]	10000		1250.0		40000	
c8id.12xlarge [2]	15000		1875.0		60000	
c8id.16xlarge [2]	20000		2500.0		80000	
c8id.24xlarge [2]	30000		3750.0		120000	
c8id.32xlarge [2]	40000		5000.0		160000	
c8id.48xlarge [2]	60000		7500.0		240000	
c8id.96xlarge [2]	80000		10000.0		480000	
c8id.metal-48xl [2]	60000		7500.0		240000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c8id.metal-96xl [2]	80000		10000.0		480000	
c8i-flex.large [1]	315	10000	39.38	1250.00	2500	40000
c8i-flex.xlarge [1]	630	10000	78.75	1250.00	3600	40000
c8i-flex.2xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c8i-flex.4xlarge [1]	2500	10000	312.50	1250.00	12000	40000
c8i-flex.8xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c8i-flex.12xlarge [1]	7500	15000	937.50	1875.00	30000	60000
c8i-flex.16xlarge [1]	10000	20000	1250.00	2500.00	40000	80000
c8in.large [1]	650	10000	81.25	1250.00	3600	40000
c8in.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c8in.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c8in.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c8in.8xlarge [2]	10000		1250.0		40000	
c8in.12xlarge [2]	15000		1875.0		60000	
c8in.16xlarge [2]	20000		2500.0		80000	
c8in.24xlarge [2]	30000		3750.0		120000	
c8in.32xlarge [2]	40000		5000.0		160000	
c8in.48xlarge [2]	60000		7500.0		240000	
c8in.96xlarge [2]	120000		15000.0		480000	
c8in.metal-48xl [2]	60000		7500.0		240000	
c8in.metal-96xl [2]	120000		15000.0		480000	
c8ine.large [1]	650	10000	81.25	1250.00	3600	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c8ine.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c8ine.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
c8ine.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c8ine.8xlarge [2]	10000		1250.0		40000	
c8ine.12xlarge [2]	15000		1875.0		60000	
c8ib.large [1]	1563	25000	195.38	3125.00	7500	120000
c8ib.xlarge [1]	3125	25000	390.62	3125.00	15000	120000
c8ib.2xlarge [1]	6250	25000	781.25	3125.00	30000	120000
c8ib.4xlarge [1]	12500	25000	1562.50	3125.00	60000	120000
c8ib.8xlarge [2]	25000		3125.0		120000	
c8ib.12xlarge [2]	37500		4687.5		180000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c8ib.16xlarge [2]	50000		6250.0		240000	
c8ib.24xlarge [2]	75000		9375.0		360000	
c8ib.32xlarge [2]	100000		12500.0		480000	
c8ib.48xlarge [2]	150000		18750.0		720000	
c8ib.96xlarge [2]	300000		37500.0		1440000	
c8ib.metal-48xl [2]	150000		18750.0		720000	
c8ib.metal-96xl [2]	300000		37500.0		1440000	
c9g.medium [1]	380	12000	47.50	1500.00	2500	48000
c9g.large [1]	760	12000	95.00	1500.00	3600	48000
c9g.xlarge [1]	1500	12000	187.50	1500.00	6000	48000
c9g.2xlarge [1]	3000	12000	375.00	1500.00	12000	48000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c9g.4xlarge [1]	6000	12000	750.00	1500.00	24000	48000
c9g.8xlarge [2]	12000		1500.0		48000	
c9g.12xlarge [2]	18000		2250.0		72000	
c9g.16xlarge [2]	24000		3000.0		96000	
c9g.24xlarge [2]	36000		4500.0		144000	
c9g.48xlarge [2]	72000		9000.0		288000	
c9g.metal-48xl [2]	72000		9000.0		288000	
c9gd.medium [1]	380	12000	47.50	1500.00	2500	48000
c9gd.large [1]	760	12000	95.00	1500.00	3600	48000
c9gd.xlarge [1]	1500	12000	187.50	1500.00	6000	48000
c9gd.2xlarge [1]	3000	12000	375.00	1500.00	12000	48000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c9gd.4xlarge [1]	6000	12000	750.00	1500.00	24000	48000
c9gd.8xlarge [2]	12000		1500.0		48000	
c9gd.12xlarge [2]	18000		2250.0		72000	
c9gd.16xlarge [2]	24000		3000.0		96000	
c9gd.24xlarge [2]	36000		4500.0		144000	
c9gd.48xlarge [2]	72000		9000.0		288000	
c9gd.metal-48xl [2]	72000		9000.0		288000	

[1] これらのインスタンスは、最大パフォーマンスを 24 時間ごとに少なくとも 1 回は 30 分間維持することができ、その後はベースラインのパフォーマンスに戻ります。

[2] これらのインスタンスは、記載されているパフォーマンスを無期限に維持することができます。ワークロードで、最大パフォーマンスを 30 分以上維持する必要がある場合は、これらのインスタンスの中から 1 つ使用します。

メモリ最適化

> **Note**
>
> - R8a、R8g、R8gd、R8i、R8id、R8i-flex、X8g、X8aedz、X8i のインスタンスタイプは、設定可能な帯域幅の重み付けをサポートします。これらのインスタンスタイプを使用すると、ネットワークパフォーマンスまたは Amazon EBS パフォーマンスのいずれかでインスタンスの帯域幅を最適化できます。次の表は、これらのインスタンスタイプのデフォルトの Amazon EBS 帯域幅パフォーマンスを示しています。詳細については「EC2 インスタンスの帯域幅の重み設定」を参照してください。
> - U7i インスタンスで IOPS パフォーマンスを最大化するには、io2 BlockExpress ボリュームの使用をお勧めします。

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r4.large [2]	425		53.125		3000	
r4.xlarge [2]	850		106.25		6000	
r4.2xlarge [2]	1700		212.5		12000	
r4.4xlarge [2]	3500		437.5		18750	
r4.8xlarge [2]	7000		875.0		37500	
r4.16xlarge [2]	14000		1750.0		75000	
r5.large [1]	650	4750	81.25	593.75	3600	18750

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r5.xlarge [1]	1150	4750	143.75	593.75	6000	18750
r5.2xlarge [1]	2300	4750	287.50	593.75	12000	18750
r5.4xlarge [2]	4750		593.75		18750	
r5.8xlarge [2]	6800		850.0		30000	
r5.12xlarge [2]	9500		1187.5		40000	
r5.16xlarge [2]	13600		1700.0		60000	
r5.24xlarge [2]	19000		2375.0		80000	
r5.metal [2]	19000		2375.0		80000	
r5a.large [1]	650	2880	81.25	360.00	3600	16000
r5a.xlarge [1]	1085	2880	135.62	360.00	6000	16000
r5a.2xlarge [1]	1580	2880	197.50	360.00	8333	16000
r5a.4xlarge [2]	2880		360.0		16000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r5a.8xlarge [2]	4750		593.75		20000	
r5a.12xlarge [2]	6780		847.5		30000	
r5a.16xlarge [2]	9500		1187.5		40000	
r5a.24xlarge [2]	13570		1696.25		60000	
r5ad.large [1]	650	2880	81.25	360.00	3600	16000
r5ad.xlarge [1]	1085	2880	135.62	360.00	6000	16000
r5ad.2xlarge [1]	1580	2880	197.50	360.00	8333	16000
r5ad.4xlarge [2]	2880		360.0		16000	
r5ad.8xlarge [2]	4750		593.75		20000	
r5ad.12xlarge [2]	6780		847.5		30000	
r5ad.16xlarge [2]	9500		1187.5		40000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r5ad.24xlarge [2]	13570		1696.25		60000	
r5b.large [1]	1250	10000	156.25	1250.00	5417	43333
r5b.xlarge [1]	2500	10000	312.50	1250.00	10833	43333
r5b.2xlarge [1]	5000	10000	625.00	1250.00	21667	43333
r5b.4xlarge [2]	10000		1250.0		43333	
r5b.8xlarge [2]	20000		2500.0		86667	
r5b.12xlarge [2]	30000		3750.0		130000	
r5b.16xlarge [2]	40000		5000.0		173333	
r5b.24xlarge [2]	60000		7500.0		260000	
r5b.metal [2]	60000		7500.0		260000	
r5d.large [1]	650	4750	81.25	593.75	3600	18750
r5d.xlarge [1]	1150	4750	143.75	593.75	6000	18750

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r5d.2xlarge [1]	2300	4750	287.50	593.75	12000	18750
r5d.4xlarge [2]	4750		593.75		18750	
r5d.8xlarge [2]	6800		850.0		30000	
r5d.12xlarge [2]	9500		1187.5		40000	
r5d.16xlarge [2]	13600		1700.0		60000	
r5d.24xlarge [2]	19000		2375.0		80000	
r5d.metal [2]	19000		2375.0		80000	
r5dn.large [1]	650	4750	81.25	593.75	3600	18750
r5dn.xlarge [1]	1150	4750	143.75	593.75	6000	18750
r5dn.2xlarge [1]	2300	4750	287.50	593.75	12000	18750
r5dn.4xlarge [2]	4750		593.75		18750	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r5dn.8xlarge [2]	6800		850.0		30000	
r5dn.12xlarge [2]	9500		1187.5		40000	
r5dn.16xlarge [2]	13600		1700.0		60000	
r5dn.24xlarge [2]	19000		2375.0		80000	
r5dn.metal [2]	19000		2375.0		80000	
r5n.large [1]	650	4750	81.25	593.75	3600	18750
r5n.xlarge [1]	1150	4750	143.75	593.75	6000	18750
r5n.2xlarge [1]	2300	4750	287.50	593.75	12000	18750
r5n.4xlarge [2]	4750		593.75		18750	
r5n.8xlarge [2]	6800		850.0		30000	
r5n.12xlarge [2]	9500		1187.5		40000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r5n.16xlarge [2]	13600		1700.0		60000	
r5n.24xlarge [2]	19000		2375.0		80000	
r5n.metal [2]	19000		2375.0		80000	
r6a.large [1]	650	10000	81.25	1250.00	3600	40000
r6a.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
r6a.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
r6a.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r6a.8xlarge [2]	10000		1250.0		40000	
r6a.12xlarge [2]	15000		1875.0		60000	
r6a.16xlarge [2]	20000		2500.0		80000	
r6a.24xlarge [2]	30000		3750.0		120000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r6a.32xlarge [2]	40000		5000.0		160000	
r6a.48xlarge [2]	40000		5000.0		240000	
r6a.metal [2]	40000		5000.0		240000	
r6g.medium [1]	315	4750	39.38	593.75	2500	20000
r6g.large [1]	630	4750	78.75	593.75	3600	20000
r6g.xlarge [1]	1188	4750	148.50	593.75	6000	20000
r6g.2xlarge [1]	2375	4750	296.88	593.75	12000	20000
r6g.4xlarge [2]	4750		593.75		20000	
r6g.8xlarge [2]	9500		1187.5		40000	
r6g.12xlarge [2]	14250		1781.25		50000	
r6g.16xlarge [2]	19000		2375.0		80000	
r6g.metal [2]	19000		2375.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r6gd.medium [1]	315	4750	39.38	593.75	2500	20000
r6gd.large [1]	630	4750	78.75	593.75	3600	20000
r6gd.xlarge [1]	1188	4750	148.50	593.75	6000	20000
r6gd.2xlarge [1]	2375	4750	296.88	593.75	12000	20000
r6gd.4xlarge [2]	4750		593.75		20000	
r6gd.8xlarge [2]	9500		1187.5		40000	
r6gd.12xlarge [2]	14250		1781.25		50000	
r6gd.16xlarge [2]	19000		2375.0		80000	
r6gd.metal [2]	19000		2375.0		80000	
r6i.large [1]	650	10000	81.25	1250.00	3600	40000
r6i.xlarge [1]	1250	10000	156.25	1250.00	6000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r6i.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
r6i.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r6i.8xlarge [2]	10000		1250.0		40000	
r6i.12xlarge [2]	15000		1875.0		60000	
r6i.16xlarge [2]	20000		2500.0		80000	
r6i.24xlarge [2]	30000		3750.0		120000	
r6i.32xlarge [2]	40000		5000.0		160000	
r6i.metal [2]	40000		5000.0		160000	
r6id.large [1]	650	10000	81.25	1250.00	3600	40000
r6id.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
r6id.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r6id.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r6id.8xlarge [2]	10000		1250.0		40000	
r6id.12xlarge [2]	15000		1875.0		60000	
r6id.16xlarge [2]	20000		2500.0		80000	
r6id.24xlarge [2]	30000		3750.0		120000	
r6id.32xlarge [2]	40000		5000.0		160000	
r6id.metal [2]	40000		5000.0		160000	
r6idn.large [1]	1562	25000	195.31	3125.00	6250	100000
r6idn.xlarge [1]	3125	25000	390.62	3125.00	12500	100000
r6idn.2xlarge [1]	6250	25000	781.25	3125.00	25000	100000
r6idn.4xlarge [1]	12500	25000	1562.50	3125.00	50000	100000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r6idn.8xlarge [2]	25000		3125.0		100000	
r6idn.12xlarge [2]	37500		4687.5		150000	
r6idn.16xlarge [2]	50000		6250.0		200000	
r6idn.24xlarge [2]	75000		9375.0		300000	
r6idn.32xlarge [2]	100000		12500.0		400000	
r6idn.metal [2]	100000		12500.0		400000	
r6in.large [1]	1562	25000	195.31	3125.00	6250	100000
r6in.xlarge [1]	3125	25000	390.62	3125.00	12500	100000
r6in.2xlarge [1]	6250	25000	781.25	3125.00	25000	100000
r6in.4xlarge [1]	12500	25000	1562.50	3125.00	50000	100000
r6in.8xlarge [2]	25000		3125.0		100000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r6in.12xlarge [2]	37500		4687.5		150000	
r6in.16xlarge [2]	50000		6250.0		200000	
r6in.24xlarge [2]	75000		9375.0		300000	
r6in.32xlarge [2]	100000		12500.0		400000	
r6in.metal [2]	100000		12500.0		400000	
r7a.medium [1]	325	10000	40.62	1250.00	2500	40000
r7a.large [1]	650	10000	81.25	1250.00	3600	40000
r7a.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
r7a.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
r7a.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r7a.8xlarge [2]	10000		1250.0		40000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r7a.12xlarge [2]	15000		1875.0		60000	
r7a.16xlarge [2]	20000		2500.0		80000	
r7a.24xlarge [2]	30000		3750.0		120000	
r7a.32xlarge [2]	40000		5000.0		160000	
r7a.48xlarge [2]	40000		5000.0		240000	
r7a.metal-48xl [2]	40000		5000.0		240000	
r7g.medium [1]	315	10000	39.38	1250.00	2500	40000
r7g.large [1]	630	10000	78.75	1250.00	3600	40000
r7g.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
r7g.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
r7g.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r7g.8xlarge [2]	10000		1250.0		40000	
r7g.12xlarge [2]	15000		1875.0		60000	
r7g.16xlarge [2]	20000		2500.0		80000	
r7g.metal [2]	20000		2500.0		80000	
r7gd.medium [1]	315	10000	39.38	1250.00	2500	40000
r7gd.large [1]	630	10000	78.75	1250.00	3600	40000
r7gd.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
r7gd.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
r7gd.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r7gd.8xlarge [2]	10000		1250.0		40000	
r7gd.12xlarge [2]	15000		1875.0		60000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r7gd.16xlarge [2]	20000		2500.0		80000	
r7gd.metal [2]	20000		2500.0		80000	
r7i.large [1]	650	10000	81.25	1250.00	3600	40000
r7i.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
r7i.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
r7i.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r7i.8xlarge [2]	10000		1250.0		40000	
r7i.12xlarge [2]	15000		1875.0		60000	
r7i.16xlarge [2]	20000		2500.0		80000	
r7i.24xlarge [2]	30000		3750.0		120000	
r7i.48xlarge [2]	40000		5000.0		240000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r7i.metal-24xl [2]	30000		3750.0		120000	
r7i.metal-48xl [2]	40000		5000.0		240000	
r7iz.large [1]	792	10000	99.00	1250.00	3600	40000
r7iz.xlarge [1]	1584	10000	198.00	1250.00	6667	40000
r7iz.2xlarge [1]	3168	10000	396.00	1250.00	13333	40000
r7iz.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r7iz.8xlarge [2]	10000		1250.0		40000	
r7iz.12xlarge [2]	19000		2375.0		76000	
r7iz.16xlarge [2]	20000		2500.0		80000	
r7iz.32xlarge [2]	40000		5000.0		160000	
r7iz.metal-16xl [2]	20000		2500.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r7iz.metal-32xl [2]	40000		5000.0		160000	
r8a.medium [1]	325	10000	40.62	1250.00	2500	40000
r8a.large [1]	650	10000	81.25	1250.00	3600	40000
r8a.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
r8a.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
r8a.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r8a.8xlarge [2]	10000		1250.0		40000	
r8a.12xlarge [2]	15000		1875.0		60000	
r8a.16xlarge [2]	20000		2500.0		80000	
r8a.24xlarge [2]	30000		3750.0		120000	
r8a.48xlarge [2]	60000		7500.0		240000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r8a.metal-24xl [2]	30000		3750.0		120000	
r8a.metal-48xl [2]	60000		7500.0		240000	
r8g.medium [1]	315	10000	39.38	1250.00	2500	40000
r8g.large [1]	630	10000	78.75	1250.00	3600	40000
r8g.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
r8g.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
r8g.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r8g.8xlarge [2]	10000		1250.0		40000	
r8g.12xlarge [2]	15000		1875.0		60000	
r8g.16xlarge [2]	20000		2500.0		80000	
r8g.24xlarge [2]	30000		3750.0		120000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r8g.48xlarge [2]	40000		5000.0		240000	
r8g.metal-24xl [2]	30000		3750.0		120000	
r8g.metal-48xl [2]	40000		5000.0		240000	
r8gb.medium [1]	1562	25000	195.31	3125.00	7500	120000
r8gb.large [1]	3125	25000	390.62	3125.00	15000	120000
r8gb.xlarge [1]	6250	25000	781.25	3125.00	30000	120000
r8gb.2xlarge [1]	12500	25000	1562.50	3125.00	60000	120000
r8gb.4xlarge [2]	25000		3125.0		120000	
r8gb.8xlarge [2]	50000		6250.0		240000	
r8gb.12xlarge [2]	75000		9375.0		360000	
r8gb.16xlarge [2]	100000		12500.0		480000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r8gb.24xlarge [2]	150000		18750.0		720000	
r8gb.48xlarge [2]	300000		37500.0		1440000	
r8gb.metal-24xl [2]	150000		18750.0		720000	
r8gb.metal-48xl [2]	300000		37500.0		1440000	
r8gd.medium [1]	315	10000	39.38	1250.00	2500	40000
r8gd.large [1]	630	10000	78.75	1250.00	3600	40000
r8gd.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
r8gd.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
r8gd.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r8gd.8xlarge [2]	10000		1250.0		40000	
r8gd.12xlarge [2]	15000		1875.0		60000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r8gd.16xlarge [2]	20000		2500.0		80000	
r8gd.24xlarge [2]	30000		3750.0		120000	
r8gd.48xlarge [2]	40000		5000.0		240000	
r8gd.metal-24xl [2]	30000		3750.0		120000	
r8gd.metal-48xl [2]	40000		5000.0		240000	
r8gn.medium [1]	760	10000	95.00	1250.00	2500	40000
r8gn.large [1]	1250	10000	156.25	1250.00	5000	40000
r8gn.xlarge [1]	2500	10000	312.50	1250.00	10000	40000
r8gn.2xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r8gn.4xlarge [2]	10000		1250.0		40000	
r8gn.8xlarge [2]	20000		2500.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r8gn.12xlarge [2]	30000		3750.0		120000	
r8gn.16xlarge [2]	40000		5000.0		160000	
r8gn.24xlarge [2]	60000		7500.0		240000	
r8gn.48xlarge [2]	120000		15000.0		480000	
r8gn.metal-24xl [2]	60000		7500.0		240000	
r8gn.metal-48xl [2]	120000		15000.0		480000	
r8i.large [1]	650	10000	81.25	1250.00	3600	40000
r8i.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
r8i.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
r8i.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r8i.8xlarge [2]	10000		1250.0		40000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r8i.12xlarge [2]	15000		1875.0		60000	
r8i.16xlarge [2]	20000		2500.0		80000	
r8i.24xlarge [2]	30000		3750.0		120000	
r8i.32xlarge [2]	40000		5000.0		160000	
r8i.48xlarge [2]	60000		7500.0		240000	
r8i.96xlarge [2]	80000		10000.0		480000	
r8i.metal-48xl [2]	60000		7500.0		240000	
r8i.metal-96xl [2]	80000		10000.0		480000	
r8id.large [1]	650	10000	81.25	1250.00	3600	40000
r8id.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
r8id.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r8id.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r8id.8xlarge [2]	10000		1250.0		40000	
r8id.12xlarge [2]	15000		1875.0		60000	
r8id.16xlarge [2]	20000		2500.0		80000	
r8id.24xlarge [2]	30000		3750.0		120000	
r8id.32xlarge [2]	40000		5000.0		160000	
r8id.48xlarge [2]	60000		7500.0		240000	
r8id.96xlarge [2]	80000		10000.0		480000	
r8id.metal-48xl [2]	60000		7500.0		240000	
r8id.metal-96xl [2]	80000		10000.0		480000	
r8i-flex.large [1]	315	10000	39.38	1250.00	2500	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r8i-flex.xlarge [1]	630	10000	78.75	1250.00	3600	40000
r8i-flex.2xlarge [1]	1250	10000	156.25	1250.00	6000	40000
r8i-flex.4xlarge [1]	2500	10000	312.50	1250.00	12000	40000
r8i-flex.8xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r8i-flex.12xlarge [1]	7500	15000	937.50	1875.00	30000	60000
r8i-flex.16xlarge [1]	10000	20000	1250.00	2500.00	40000	80000
r8in.large [1]	650	10000	81.25	1250.00	3600	40000
r8in.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
r8in.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
r8in.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r8in.8xlarge [2]	10000		1250.0		40000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r8in.12xlarge [2]	15000		1875.0		60000	
r8in.16xlarge [2]	20000		2500.0		80000	
r8in.24xlarge [2]	30000		3750.0		120000	
r8in.32xlarge [2]	40000		5000.0		160000	
r8in.48xlarge [2]	60000		7500.0		240000	
r8in.96xlarge [2]	120000		15000.0		480000	
r8in.metal-48xl [2]	60000		7500.0		240000	
r8in.metal-96xl [2]	120000		15000.0		480000	
r8idn.large [1]	650	10000	81.25	1250.00	3600	40000
r8idn.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
r8idn.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r8idn.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r8idn.8xlarge [2]	10000		1250.0		40000	
r8idn.12xlarge [2]	15000		1875.0		60000	
r8idn.16xlarge [2]	20000		2500.0		80000	
r8idn.24xlarge [2]	30000		3750.0		120000	
r8idn.32xlarge [2]	40000		5000.0		160000	
r8idn.48xlarge [2]	60000		7500.0		240000	
r8idn.96xlarge [2]	120000		15000.0		480000	
r8idn.metal-48xl [2]	60000		7500.0		240000	
r8idn.metal-96xl [2]	120000		15000.0		480000	
r8ib.large [1]	1563	25000	195.38	3125.00	7500	120000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r8ib.xlarge [1]	3125	25000	390.62	3125.00	15000	120000
r8ib.2xlarge [1]	6250	25000	781.25	3125.00	30000	120000
r8ib.4xlarge [1]	12500	25000	1562.50	3125.00	60000	120000
r8ib.8xlarge [2]	25000		3125.0		120000	
r8ib.12xlarge [2]	37500		4687.5		180000	
r8ib.16xlarge [2]	50000		6250.0		240000	
r8ib.24xlarge [2]	75000		9375.0		360000	
r8ib.32xlarge [2]	100000		12500.0		480000	
r8ib.48xlarge [2]	150000		18750.0		720000	
r8ib.96xlarge [2]	300000		37500.0		1440000	
r8ib.metal-48xl [2]	150000		18750.0		720000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r8ib.metal-96xl [2]	300000		37500.0		1440000	
r8idb.large [1]	1563	25000	195.38	3125.00	7500	120000
r8idb.xlarge [1]	3125	25000	390.62	3125.00	15000	120000
r8idb.2xlarge [1]	6250	25000	781.25	3125.00	30000	120000
r8idb.4xlarge [1]	12500	25000	1562.50	3125.00	60000	120000
r8idb.8xlarge [2]	25000		3125.0		120000	
r8idb.12xlarge [2]	37500		4687.5		180000	
r8idb.16xlarge [2]	50000		6250.0		240000	
r8idb.24xlarge [2]	75000		9375.0		360000	
r8idb.32xlarge [2]	100000		12500.0		480000	
r8idb.48xlarge [2]	150000		18750.0		720000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r8idb.96xlarge [2]	300000		37500.0		1440000	
r8idb.metal-48xl [2]	150000		18750.0		720000	
r8idb.metal-96xl [2]	300000		37500.0		1440000	
u-3tb1.56xlarge [2]	19000		2375.0		80000	
u-6tb1.56xlarge [2]	38000		4750.0		160000	
u-6tb1.112xlarge [2]	38000		4750.0		160000	
u-6tb1.metal [2]	38000		4750.0		160000	
u-9tb1.112xlarge [2]	38000		4750.0		160000	
u-9tb1.metal [2]	38000		4750.0		160000	
u-12tb1.112xlarge [2]	38000		4750.0		160000	
u-12tb1.metal [2]	38000		4750.0		160000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
u-18tb1.112xlarge [2]	38000		4750.0		160000	
u-18tb1.metal [2]	38000		4750.0		160000	
u-24tb1.112xlarge [2]	38000		4750.0		160000	
u-24tb1.metal [2]	38000		4750.0		160000	
u7i-6tb.112xlarge [2]	100000		12500.0		560000	
u7i-8tb.112xlarge [2]	100000		12500.0		560000	
u7i-12tb.224xlarge [2]	100000		12500.0		560000	
u7in-16tb.224xlarge [2]	100000		12500.0		560000	
u7in-24tb.224xlarge [2]	100000		12500.0		560000	
u7in-32tb.224xlarge [2]	100000		12500.0		560000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
u7inh-32tb.480xlarge [2]	160000		20000.0		840000	
x1.16xlarge [2]	7000		875.0		40000	
x1.32xlarge [2]	14000		1750.0		80000	
x1e.xlarge [2]	500		62.5		3700	
x1e.2xlarge [2]	1,000		125.0		7400	
x1e.4xlarge [2]	1750		218.75		10000	
x1e.8xlarge [2]	3500		437.5		20000	
x1e.16xlarge [2]	7000		875.0		40000	
x1e.32xlarge [2]	14000		1750.0		80000	
x2gd.medium [1]	315	4750	39.38	593.75	2500	20000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
x2gd.large [1]	630	4750	78.75	593.75	3600	20000
x2gd.xlarge [1]	1188	4750	148.50	593.75	6000	20000
x2gd.2xlarge [1]	2375	4750	296.88	593.75	12000	20000
x2gd.4xlarge [2]	4750		593.75		20000	
x2gd.8xlarge [2]	9500		1187.5		40000	
x2gd.12xlarge [2]	14250		1781.25		60000	
x2gd.16xlarge [2]	19000		2375.0		80000	
x2gd.metal [2]	19000		2375.0		80000	
x2idn.16xlarge [2]	40000		5000.0		173333	
x2idn.24xlarge [2]	60000		7500.0		260000	
x2idn.32xlarge [2]	80000		10000.0		260000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
x2idn.metal [2]	80000		10000.0		260000	
x2iedn.xlarge [1]	2500	20000	312.50	2500.00	8125	65000
x2iedn.2xlarge [1]	5000	20000	625.00	2500.00	16250	65000
x2iedn.4xlarge [1]	10000	20000	1250.00	2500.00	32500	65000
x2iedn.8xlarge [2]	20000		2500.0		65000	
x2iedn.16xlarge [2]	40000		5000.0		130000	
x2iedn.24xlarge [2]	60000		7500.0		195000	
x2iedn.32xlarge [2]	80000		10000.0		260000	
x2iedn.metal [2]	80000		10000.0		260000	
x2iezn.2xlarge [2]	3170		396.25		13333	
x2iezn.4xlarge [2]	4750		593.75		20000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
x2iezn.6xlarge [2]	9500		1187.5		40000	
x2iezn.8xlarge [2]	12000		1500.0		55000	
x2iezn.12xlarge [2]	19000		2375.0		80000	
x2iezn.metal [2]	19000		2375.0		80000	
x8g.medium [1]	315	10000	39.38	1250.00	2500	40000
x8g.large [1]	630	10000	78.75	1250.00	3600	40000
x8g.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
x8g.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
x8g.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
x8g.8xlarge [2]	10000		1250.0		40000	
x8g.12xlarge [2]	15000		1875.0		60000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
x8g.16xlarge [2]	20000		2500.0		80000	
x8g.24xlarge [2]	30000		3750.0		120000	
x8g.48xlarge [2]	40000		5000.0		240000	
x8g.metal-24xl [2]	30000		3750.0		120000	
x8g.metal-48xl [2]	40000		5000.0		240000	
x8aedz.large [1]	1250	15000	156.25	1875.00	5000	60000
x8aedz.xlarge [1]	2500	15000	312.50	1875.00	10000	60000
x8aedz.3xlarge [1]	7500	15000	937.50	1875.00	30000	60000
x8aedz.6xlarge [2]	15000		1875.0		60000	
x8aedz.12xlarge [2]	30000		3750.0		120000	
x8aedz.24xlarge [2]	60000		7500.0		240000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
x8aedz.metal-12xl [2]	30000		3750.0		120000	
x8aedz.metal-24xl [2]	60000		7500.0		240000	
x8i.large [1]	650	10000	81.25	1250.00	3600	40000
x8i.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
x8i.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
x8i.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
x8i.8xlarge [2]	10000		1250.0		40000	
x8i.12xlarge [2]	15000		1875.0		60000	
x8i.16xlarge [2]	20000		2500.0		80000	
x8i.24xlarge [2]	30000		3750.0		120000	
x8i.32xlarge [2]	40000		5000.0		160000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
x8i.48xlarge [2]	60000		7500.0		240000	
x8i.64xlarge [2]	70000		8750.0		320000	
x8i.96xlarge [2]	80000		10000.0		480000	
x8i.metal-48xl [2]	60000		7500.0		240000	
x8i.metal-96xl [2]	80000		10000.0		480000	
z1d.large [1]	800	3170	100.00	396.25	3333	13333
z1d.xlarge [1]	1580	3170	197.50	396.25	6667	13333
z1d.2xlarge [2]	3170		396.25		13333	
z1d.3xlarge [2]	4750		593.75		20000	
z1d.6xlarge [2]	9500		1187.5		40000	
z1d.12xlarge [2]	19000		2375.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
z1d.metal [2]	19000		2375.0		80000	

[1] これらのインスタンスは、最大パフォーマンスを 24 時間ごとに少なくとも 1 回は 30 分間維持することができ、その後はベースラインのパフォーマンスに戻ります。

[2] これらのインスタンスは、記載されているパフォーマンスを無期限に維持することができます。ワークロードで、最大パフォーマンスを 30 分以上維持する必要がある場合は、これらのインスタンスの中から 1 つ使用します。

ストレージの最適化

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
d2.xlarge [2]	750		93.75		6000	
d2.2xlarge [2]	1,000		125.0		8000	
d2.4xlarge [2]	2000		250.0		16000	
d2.8xlarge [2]	4000		500.0		32000	
d3.xlarge [1]	850	2800	106.25	350.00	5000	15000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
d3.2xlarge [1]	1700	2800	212.50	350.00	10000	15000
d3.4xlarge [2]	2800		350.0		15000	
d3.8xlarge [2]	5000		625.0		30000	
d3en.xlarge [1]	850	2800	106.25	350.00	5000	15000
d3en.2xlarge [1]	1700	2800	212.50	350.00	10000	15000
d3en.4xlarge [2]	2800		350.0		15000	
d3en.6xlarge [2]	4000		500.0		25000	
d3en.8xlarge [2]	5000		625.0		30000	
d3en.12xlarge [2]	7000		875.0		40000	
h1.2xlarge [2]	1750		218.75		12000	
h1.4xlarge [2]	3500		437.5		20000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
h1.8xlarge [2]	7000		875.0		40000	
h1.16xlarge [2]	14000		1750.0		80000	
i3.large [2]	425		53.125		3000	
i3.xlarge [2]	850		106.25		6000	
i3.2xlarge [2]	1700		212.5		12000	
i3.4xlarge [2]	3500		437.5		16000	
i3.8xlarge [2]	7000		875.0		32500	
i3.16xlarge [2]	14000		1750.0		65000	
i3.metal [2]	19000		2375.0		80000	
i3en.large [1]	576	4750	72.10	593.75	3000	20000
i3en.xlarge [1]	1153	4750	144.20	593.75	6000	20000
i3en.2xlarge [1]	2307	4750	288.39	593.75	12000	20000
i3en.3xlarge [1]	3800	4750	475.00	593.75	15000	20000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
i3en.6xlarge [2]	4750		593.75		20000	
i3en.12xlarge [2]	9500		1187.5		40000	
i3en.24xlarge [2]	19000		2375.0		80000	
i3en.metal [2]	19000		2375.0		80000	
i4g.large [1]	625	10000	78.12	1250.00	2500	40000
i4g.xlarge [1]	1250	10000	156.25	1250.00	5000	40000
i4g.2xlarge [1]	2500	10000	312.50	1250.00	10000	40000
i4g.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
i4g.8xlarge [2]	10000		1250.0		40000	
i4g.16xlarge [2]	20000		2500.0		80000	
i4i.large [1]	625	10000	78.12	1250.00	2500	40000
i4i.xlarge [1]	1250	10000	156.25	1250.00	5000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
i4i.2xlarge [1]	2500	10000	312.50	1250.00	10000	40000
i4i.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
i4i.8xlarge [2]	10000		1250.0		40000	
i4i.12xlarge [2]	15000		1875.0		60000	
i4i.16xlarge [2]	20000		2500.0		80000	
i4i.24xlarge [2]	30000		3750.0		120000	
i4i.32xlarge [2]	40000		5000.0		160000	
i4i.metal [2]	40000		5000.0		160000	
i7i.large [1]	625	10000	78.12	1250.00	2500	40000
i7i.xlarge [1]	1250	10000	156.25	1250.00	5000	40000
i7i.2xlarge [1]	2500	10000	312.50	1250.00	10000	40000
i7i.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
i7i.8xlarge [2]	10000		1250.0		40000	
i7i.12xlarge [2]	15000		1875.0		60000	
i7i.16xlarge [2]	20000		2500.0		80000	
i7i.24xlarge [2]	30000		3750.0		120000	
i7i.48xlarge [2]	60000		7500.0		240000	
i7i.metal-24xl [2]	30000		3750.0		120000	
i7i.metal-48xl [2]	60000		7500.0		240000	
i7ie.large [1]	625	10000	78.12	1250.00	2500	40000
i7ie.xlarge [1]	1250	10000	156.25	1250.00	5000	40000
i7ie.2xlarge [1]	2500	10000	312.50	1250.00	10000	40000
i7ie.3xlarge [1]	3750	10000	468.75	1250.00	15000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
i7ie.6xlarge [1]	7500	10000	937.50	1250.00	30000	40000
i7ie.12xlarge [2]	15000		1875.0		60000	
i7ie.18xlarge [2]	22500		2812.5		90000	
i7ie.24xlarge [2]	30000		3750.0		120000	
i7ie.48xlarge [2]	60000		7500.0		240000	
i7ie.metal-24xl [2]	30000		3750.0		120000	
i7ie.metal-48xl [2]	60000		7500.0		240000	
i8g.large [1]	625	10000	78.12	1250.00	2500	40000
i8g.xlarge [1]	1250	10000	156.25	1250.00	5000	40000
i8g.2xlarge [1]	2500	10000	312.50	1250.00	10000	40000
i8g.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
i8g.8xlarge [2]	10000		1250.0		40000	
i8g.12xlarge [2]	15000		1875.0		60000	
i8g.16xlarge [2]	20000		2500.0		80000	
i8g.24xlarge [2]	30000		3750.0		120000	
i8g.48xlarge [2]	60000		7500.0		240000	
i8g.metal-24xl [2]	30000		3750.0		120000	
i8g.metal-48xl [2]	60000		7500.0		240000	
i8ge.large [1]	625	10000	78.12	1250.00	2500	40000
i8ge.xlarge [1]	1250	10000	156.25	1250.00	5000	40000
i8ge.2xlarge [1]	2500	10000	312.50	1250.00	10000	40000
i8ge.3xlarge [1]	3750	10000	468.75	1250.00	15000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
i8ge.6xlarge [1]	7500	10000	937.50	1250.00	30000	40000
i8ge.12xlarge [2]	15000		1875.0		60000	
i8ge.18xlarge [2]	22500		2812.5		90000	
i8ge.24xlarge [2]	30000		3750.0		120000	
i8ge.48xlarge [2]	60000		7500.0		240000	
i8ge.metal-24xl [2]	30000		3750.0		120000	
i8ge.metal-48xl [2]	60000		7500.0		240000	
im4gn.large [1]	1250	10000	156.25	1250.00	5000	40000
im4gn.xlarge [1]	2500	10000	312.50	1250.00	10000	40000
im4gn.2xlarge [1]	5000	10000	625.00	1250.00	20000	40000
im4gn.4xlarge [2]	10000		1250.0		40000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
im4gn.8xlarge [2]	20000		2500.0		80000	
im4gn.16xlarge [2]	40000		5000.0		160000	
is4gen.medium [1]	625	10000	78.12	1250.00	2500	40000
is4gen.large [1]	1250	10000	156.25	1250.00	5000	40000
is4gen.xlarge [1]	2500	10000	312.50	1250.00	10000	40000
is4gen.2xlarge [1]	5000	10000	625.00	1250.00	20000	40000
is4gen.4xlarge [2]	10000		1250.0		40000	
is4gen.8xlarge [2]	20000		2500.0		80000	

[1] これらのインスタンスは、最大パフォーマンスを 24 時間ごとに少なくとも 1 回は 30 分間維持することができ、その後はベースラインのパフォーマンスに戻ります。

[2] これらのインスタンスは、記載されているパフォーマンスを無期限に維持することができます。ワークロードで、最大パフォーマンスを 30 分以上維持する必要がある場合は、これらのインスタンスの中から 1 つ使用します。

高速コンピューティング

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
dl1.24xlarge [2]	19000		2375.0		80000	
dl2q.24xlarge [2]	19000		2375.0		80000	
f2.6xlarge [2]	7500		937.5		30000	
f2.12xlarge [2]	15000		1875.0		60000	
f2.48xlarge [2]	60000		7500.0		240000	
g3.4xlarge [2]	3500		437.5		20000	
g3.8xlarge [2]	7000		875.0		40000	
g3.16xlarge [2]	14000		1750.0		80000	
g4ad.xlarge [1]	400	3170	50.00	396.25	1700	13333
g4ad.2xlarge [1]	800	3170	100.00	396.25	3400	13333

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
g4ad.4xlarge [1]	1580	3170	197.50	396.25	6700	13333
g4ad.8xlarge [2]	3170		396.25		13333	
g4ad.16xlarge [2]	6300		787.5		26667	
g4dn.xlarge [1]	950	3500	118.75	437.50	3000	20000
g4dn.2xlarge [1]	1150	3500	143.75	437.50	6000	20000
g4dn.4xlarge [2]	4750		593.75		20000	
g4dn.8xlarge [2]	9500		1187.5		40000	
g4dn.12xlarge [2]	9500		1187.5		40000	
g4dn.16xlarge [2]	9500		1187.5		40000	
g4dn.metal [2]	19000		2375.0		80000	
g5.xlarge [1]	700	3500	87.50	437.50	3000	15000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
g5.2xlarge [1]	850	3500	106.25	437.50	3500	15000
g5.4xlarge [2]	4750		593.75		20000	
g5.8xlarge [2]	16000		2000.0		65000	
g5.12xlarge [2]	16000		2000.0		65000	
g5.16xlarge [2]	16000		2000.0		65000	
g5.24xlarge [2]	19000		2375.0		80000	
g5.48xlarge [2]	19000		2375.0		80000	
g5g.xlarge [1]	1188	4750	148.50	593.75	6000	20000
g5g.2xlarge [1]	2375	4750	296.88	593.75	12000	20000
g5g.4xlarge [2]	4750		593.75		20000	
g5g.8xlarge [2]	9500		1187.5		40000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
g5g.16xlarge [2]	19000		2375.0		80000	
g5g.metal [2]	19000		2375.0		80000	
g6.xlarge [1]	1,000	5000	125.00	625.00	4000	20000
g6.2xlarge [1]	2000	5000	250.00	625.00	8000	20000
g6.4xlarge [2]	8000		1000.0		32000	
g6.8xlarge [2]	16000		2000.0		64000	
g6.12xlarge [2]	20000		2500.0		80000	
g6.16xlarge [2]	20000		2500.0		80000	
g6.24xlarge [2]	30000		3750.0		120000	
g6.48xlarge [2]	60000		7500.0		240000	
g6e.xlarge [1]	1,000	5000	125.00	625.00	4000	20000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
g6e.2xlarge [1]	2000	5000	250.00	625.00	8000	20000
g6e.4xlarge [2]	8000		1000.0		32000	
g6e.8xlarge [2]	16000		2000.0		64000	
g6e.12xlarge [2]	20000		2500.0		80000	
g6e.16xlarge [2]	20000		2500.0		80000	
g6e.24xlarge [2]	30000		3750.0		120000	
g6e.48xlarge [2]	60000		7500.0		240000	
g6f.large [1]	936	5000	117.00	625.00	3750	20000
g6f.xlarge [1]	1,000	5000	125.00	625.00	4000	20000
g6f.2xlarge [1]	2000	5000	250.00	625.00	8000	20000
g6e.4xlarge [2]	6000		750.0		24000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
gr6.4xlarge [2]	8000		1000.0		32000	
gr6.8xlarge [2]	16000		2000.0		64000	
gr6f.4xlarge [2]	8000		1000.0		32000	
g7.2xlarge [1]	4000	8000	500.00	1000.00	16000	32000
g7.4xlarge [2]	8000		1000.0		32000	
g7.8xlarge [2]	16000		2000.0		64000	
g7.12xlarge [2]	20000		2500.0		80000	
g7.24xlarge [2]	40000		5000.0		160000	
g7.48xlarge [2]	80000		10000.0		320000	
g7e.2xlarge [1]	2000	5000	250.00	625.00	8000	20000
g7e.4xlarge [2]	8000		1000.0		32000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
g7e.8xlarge [2]	16000		2000.0		64000	
g7e.12xlarge [2]	25000		3125.0		100000	
g7e.24xlarge [2]	50000		6250.0		200000	
g7e.48xlarge [2]	100000		12500.0		400000	
inf1.xlarge [1]	1190	4750	148.75	593.75	4000	20000
inf1.2xlarge [1]	1190	4750	148.75	593.75	6000	20000
inf1.6xlarge [2]	4750		593.75		20000	
inf1.24xlarge [2]	19000		2375.0		80000	
inf2.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
inf2.8xlarge [2]	10000		1250.0		40000	
inf2.24xlarge [2]	30000		3750.0		120000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
inf2.48xlarge [2]	60000		7500.0		240000	
p3.2xlarge [2]	1750		218.75		10000	
p3.8xlarge [2]	7000		875.0		40000	
p3.16xlarge [2]	14000		1750.0		80000	
p3dn.24xlarge [2]	19000		2375.0		80000	
p4d.24xlarge [2]	19000		2375.0		80000	
p4de.24xlarge [2]	19000		2375.0		80000	
p5.4xlarge [2]	10000		1250.0		32500	
p5.48xlarge [2]	80000		10000.0		260000	
p5e.48xlarge [2]	80000		10000.0		260000	
p5en.48xlarge [2]	100000		12500.0		400000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
p6-b200.48xlarge [2]	100000		12500.0		400000	
p6-b300.48xlarge [2]	100000		12500.0		400000	
p6e-gb200.36xlarge [2]	60000		7500.0		240000	
trn1.2xlarge [1]	5000	20000	625.00	2500.00	16250	65000
trn1.32xlarge [2]	80000		10000.0		260000	
trn1n.32xlarge [2]	80000		10000.0		260000	
trn2.3xlarge [2]	5000		625.0		16250	
trn2.48xlarge [2]	80000		10000.0		260000	
trn2u.48xlarge [2]	80000		10000.0		260000	
vt1.3xlarge [1]	2375	4750	296.88	593.75	10000	20000
vt1.6xlarge [2]	4750		593.75		20000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
vt1.24xlarge [2]	19000		2375.0		80000	

[1] これらのインスタンスは、最大パフォーマンスを 24 時間ごとに少なくとも 1 回は 30 分間維持することができ、その後はベースラインのパフォーマンスに戻ります。

[2] これらのインスタンスは、記載されているパフォーマンスを無期限に維持することができます。ワークロードで、最大パフォーマンスを 30 分以上維持する必要がある場合は、これらのインスタンスの中から 1 つ使用します。

高性能コンピューティング

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
hpc6a.48xlarge [1]	87	2085	10.88	260.62	500	11000
hpc6id.32xlarge [1]	87	2085	10.88	260.62	500	11000
hpc7a.12xlarge [1]	87	2085	10.88	260.62	500	11000
hpc7a.24xlarge [1]	87	2085	10.88	260.62	500	11000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
hpc7a.48xlarge [1]	87	2085	10.88	260.62	500	11000
hpc7a.96xlarge [1]	87	2085	10.88	260.62	500	11000
hpc7g.4xlarge [1]	87	2085	10.88	260.62	500	11000
hpc7g.8xlarge [1]	87	2085	10.88	260.62	500	11000
hpc7g.16xlarge [1]	87	2085	10.88	260.62	500	11000
hpc8a.96xlarge [1]	87	2085	10.88	260.62	500	11000

[1] これらのインスタンスは、最大パフォーマンスを 24 時間ごとに少なくとも 1 回は 30 分間維持することができ、その後はベースラインのパフォーマンスに戻ります。

[2] これらのインスタンスは、記載されているパフォーマンスを無期限に維持することができます。ワークロードで、最大パフォーマンスを 30 分以上維持する必要がある場合は、これらのインスタンスの中から 1 つ使用します。

EBS 最適化をサポート

次のインスタンスタイプは EBS 最適化をサポートしていますが、EBS 最適化はデフォルトでは有効になっていません。説明されている EBS パフォーマンスのレベルを達成するには、起動時または起動後に、追加の時間料金で EBS 最適化を有効にする必要があります。

インスタンスサイズ	最大帯域幅 (Mbps)	最大スループット (MB/秒、128 KiB I/O)	最大 IOPS (16 KiB I/O)
c1.xlarge	1,000	125.0	8000
c3.xlarge	500	62.5	4000
c3.2xlarge	1,000	125.0	8000
c3.4xlarge	2000	250.0	16000
i2.xlarge	500	62.5	4000
i2.2xlarge	1,000	125.0	8000
i2.4xlarge	2000	250.0	16000
m1.large	500	62.5	4000
m1.xlarge	1,000	125.0	8000
m2.2xlarge	500	62.5	4000
m2.4xlarge	1,000	125.0	8000
m3.xlarge	500	62.5	4000
m3.2xlarge	1,000	125.0	8000
r3.xlarge	500	62.5	4000
r3.2xlarge	1,000	125.0	8000
r3.4xlarge	2000	250.0	16000

Note

`i2.8xlarge`、`c3.8xlarge`、および `r3.8xlarge` インスタンスには専用の EBS 帯域幅がないため、EBS 最適化を提供しません。これらのインスタンスでは、ネットワークトラフィッ

クと Amazon EBS トラフィックは同じ 10 ギガビットネットワークインターフェイスで共有されます。

Amazon EBS 最適化パフォーマンスを最大限に発揮する

インスタンスの EBS パフォーマンスは、インスタンスタイプのパフォーマンス制限、またはアタッチされたボリュームの合計パフォーマンスのうち、どちらか小さい方によって制限されます。EBS のパフォーマンスを最大化するには、インスタンスにアタッチされたボリュームが合計でインスタンスの最大パフォーマンスと同等かそれ以上のパフォーマンスを発揮する必要があります。例えば、`r6i.16xlarge` で `80,000` IOPS を実現するには、インスタンスにそれぞれ `16,000` IOPS でプロビジョニングされた少なくとも 5 つの gp2 ボリューム (5 ボリューム x `16,000` IOPS = `80,000` IOPS) が必要ですが、`80,000` IOPS でプロビジョニングされた 1 つの gp3 ボリュームでもかまいません。アプリケーションのニーズよりも多い専用 Amazon EBS スループットを提供するインスタンスタイプを選択することをお勧めします。このようなインスタンスを選択しない場合、Amazon EBS と Amazon EC2 間の接続がパフォーマンスのボトルネックになる可能性があります。

Important

設定可能な帯域幅の重み付けを使用する場合、インスタンスの EBS 帯域幅制限が変更される可能性があります。VPC-1 重み設定のインスタンスでは、ネットワーク帯域幅が増加するため、IOPS 制限の前に EBS 帯域幅制限に達するため、EBS ボリュームの IOPS が予想よりも低くなる可能性があります。これは、I/O サイズが大きいほど特に顕著です。選択した帯域幅の重み付けでパフォーマンス要件を満たしていることを確認するために、常に特定のワークロードをテストしてください。詳細については「EC2 インスタンスの帯域幅の重み設定」を参照してください。

`EBSIOBalance%` および `EBSByteBalance%` メトリクスを使用して、インスタンスのサイズが正しく設定されているかどうかを判断できます。これらのメトリクスを CloudWatch コンソールで表示して、指定したしきい値に基づいてトリガーされるアラームを設定することができます。これらのメトリクスは、割合 (%) で表されます。バランスの割合が常に低いインスタンスは、拡大する必要があります。バランスの割合が 100% を下回ることのないインスタンスは縮小する必要があります。詳細については「CloudWatch を使用したインスタンスのモニタリング」を参照してください。

ハイメモリインスタンスは大規模なインメモリデータベースを実行するよう設定されており、これにはクラウド内の SAP HANA インメモリデータベースの本番デプロイメントを含みます。EBS パ

フォーマンスを最大化するには、プロビジョニングされたパフォーマンスが同じで偶数の io1 または io2 ボリュームを持つハイメモリインスタンスを使用します。例えば、IOPS 負荷の高いワークロードの場合は、40,000 のプロビジョンド IOPS を持つ 4 つの io1 または io2 ボリュームを使用して、最大 160,000 のインスタンス IOPS を取得します。同様に、スループットの多いワークロードの場合は、48,000 のプロビジョンド IOPS を持つ 6 つの io1 または io2 ボリュームを使用して、最大 4,750 MB/秒のスループットを取得します。その他の推奨事項については、SAP HANA のストレージ構成を参照してください。

考慮事項

- 2020 年 2 月 26 日以降に起動された G4dn、I3en、Inf1、M5a、M5ad、R5a、R5ad、T3、T3a、および Z1d の各インスタンスは、最大の EBS 最適化パフォーマンスを提供します。2020 年 2 月 26 日より前に起動されたインスタンスのパフォーマンスを最大化するには、インスタンスを停止してから起動します。
- 2019 年 12 月 3 日以降に起動された C5、C5d、C5n、M5、M5d、M5n、M5dn、R5、R5d、R5n、R5dn、および P3dn の各インスタンスは、最大の EBS 最適化パフォーマンスを提供します。2019 年 12 月 3 日より前に起動されたインスタンスから最大のパフォーマンスを得るには、インスタンスを停止してから起動します。
- 2020 年 3 月 12 日以降に起動された u-6tb1.metal、u-9tb1.metal、および u-12tb1.metal の各インスタンスは、最大の EBS 最適化パフォーマンスを提供します。2020 年 3 月 12 日より前に開始されたこれらのタイプのインスタンスのパフォーマンスはそれより低い可能性があります。2020 年 3 月 12 日より前に起動されたインスタンスから最大限のパフォーマンスを得るには、アカウントチームに連絡して、インスタンスをアップグレードしてください (追加料金なし)。

EBS 最適化 EC2 インスタンスタイプを見つける

各リージョンの EBS 最適化をサポートするインスタンスタイプを表示できます。

Console

デフォルトで EBS 最適化されているインスタンスタイプを見つけるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンスタイプ]を選択します。
3. [EBS 最適化のサポート] = [デフォルト] でフィルタリングします。

4. (オプション) [設定] アイコンをクリックし、[EBS の最大 IOPS] や [EBS のベースライン IOPS] などの関連する列をオンにします。
5. (オプション) フィルターを追加して、対象の、特定のインスタンスタイプをさらに絞り込みます。

AWS CLI

デフォルトで EBS 最適化されているインスタンスタイプを見つけるには

次の describe-instances コマンドを使用します。

```
aws ec2 describe-instance-types \
--filters Name=ebs-info.ebs-optimized-support,Values=default  \
--query 'InstanceTypes[].{InstanceType:InstanceType,
 "MaxBandwidth(Mb/s)":EbsInfo.EbsOptimizedInfo.MaximumBandwidthInMbps,
 MaxIOPS:EbsInfo.EbsOptimizedInfo.MaximumIops, "MaxThroughput(MB/
s)":EbsInfo.EbsOptimizedInfo.MaximumThroughputInMBps}' \
--output=table
```

EBS 最適化をオプションでサポートするインスタンスタイプを見つけるには

次の describe-instances コマンドを使用します。

```
aws ec2 describe-instance-types \
--filters Name=ebs-info.ebs-optimized-support,Values=supported \
--query 'InstanceTypes[].{InstanceType:InstanceType,
 "MaxBandwidth(Mb/s)":EbsInfo.EbsOptimizedInfo.MaximumBandwidthInMbps,
 MaxIOPS:EbsInfo.EbsOptimizedInfo.MaximumIops, "MaxThroughput(MB/
s)":EbsInfo.EbsOptimizedInfo.MaximumThroughputInMBps}' \
--output=table
```

eu-west-1 の出力例を次に示します。

```
-------------------------------------------------------------------------
|                        DescribeInstanceTypes                          |
+--------------+----------------------+----------+-----------------------+
| InstanceType | MaxBandwidth(Mb/s)   | MaxIOPS  |  MaxThroughput(MB/s)  |
+--------------+----------------------+----------+-----------------------+
|  i2.2xlarge  |  1000                |  8000    |  125.0                |
|  m2.4xlarge  |  1000                |  8000    |  125.0                |
|  m2.2xlarge  |  500                 |  4000    |  62.5                 |
```

```
|  c1.xlarge   |  1000                |  8000    |  125.0                |
|  i2.xlarge   |  500                 |  4000    |  62.5                 |
|  m3.xlarge   |  500                 |  4000    |  62.5                 |
|  m1.xlarge   |  1000                |  8000    |  125.0                |
|  r3.4xlarge  |  2000                |  16000   |  250.0                |
|  r3.2xlarge  |  1000                |  8000    |  125.0                |
|  c3.xlarge   |  500                 |  4000    |  62.5                 |
|  m3.2xlarge  |  1000                |  8000    |  125.0                |
|  r3.xlarge   |  500                 |  4000    |  62.5                 |
|  i2.4xlarge  |  2000                |  16000   |  250.0                |
|  c3.4xlarge  |  2000                |  16000   |  250.0                |
|  c3.2xlarge  |  1000                |  8000    |  125.0                |
|  m1.large    |  500                 |  4000    |  62.5                 |
+--------------+----------------------+----------+-----------------------+
```

PowerShell

デフォルトで EBS 最適化されているインスタンスタイプを見つけるには

Get-EC2InstanceType コマンドレットを使用します。

```
Get-EC2InstanceType `
    -Filter @{Name="ebs-info.ebs-optimized-support"; Values="default"} | `
    Select InstanceType, `
        @{Name="MaxBandwidth(Mb/s)";
 Expression={($_.EbsInfo.EbsOptimizedInfo.MaximumBandwidthInMbps)}}, `
        @{Name="MaxIOPS"; Expression={($_.EbsInfo.EbsOptimizedInfo.MaximumIops)}}, `
        @{Name="MaxThroughput (MB/s)";
 Expression={($_.EbsInfo.EbsOptimizedInfo.MaximumThroughputInMBps)}}
```

EBS 最適化をオプションでサポートするインスタンスタイプを見つけるには

Get-EC2InstanceType コマンドレットを使用します。

```
Get-EC2InstanceType `
    -Filter @{Name="ebs-info.ebs-optimized-support"; Values="supported"} | `
    Select InstanceType, `
        @{Name="MaxBandwidth(Mb/s)";
 Expression={($_.EbsInfo.EbsOptimizedInfo.MaximumBandwidthInMbps)}}, `
        @{Name="MaxIOPS"; Expression={($_.EbsInfo.EbsOptimizedInfo.MaximumIops)}}, `
        @{Name="MaxThroughput (MB/s)";
 Expression={($_.EbsInfo.EbsOptimizedInfo.MaximumThroughputInMBps)}}
```

eu-west-1 の出力例を次に示します。

```
InstanceType MaxBandwidth(Mb/s) MaxIOPS MaxThroughput (MB/s)
------------ ------------------ ------- --------------------
m2.4xlarge                 1000    8000              125.000
i2.2xlarge                 1000    8000              125.000
c1.xlarge                  1000    8000              125.000
m2.2xlarge                  500    4000               62.500
r3.2xlarge                 1000    8000              125.000
m3.xlarge                   500    4000               62.500
r3.4xlarge                 2000   16000              250.000
m1.xlarge                  1000    8000              125.000
i2.xlarge                   500    4000               62.500
c3.xlarge                   500    4000               62.500
c3.4xlarge                 2000   16000              250.000
c3.2xlarge                 1000    8000              125.000
i2.4xlarge                 2000   16000              250.000
r3.xlarge                   500    4000               62.500
m3.2xlarge                 1000    8000              125.000
m1.large                    500    4000               62.500
```

Amazon EC2 インスタンスの EBS 最適化を有効にする

EBS 最適化を手動で有効にできるのは、オプションで EBS 最適化をサポートしている旧世代のインスタンスタイプのみです。これらのインスタンスタイプで EBS 最適化を有効にすると、時間単位の追加料金が発生します。

前提条件

- インスタンスタイプで EBS 最適化を有効にする必要があるかどうか確認します。詳細については「EBS 最適化をサポート」を参照してください。
- 起動後に EBS 最適化を有効にするには、インスタンスを停止する必要があります。

> **Warning**
>
> インスタンスストアボリューム上のデータは、インスタンスを停止すると失われます。このデータを保持するには、永続ストレージにバックアップしてください。

Console

起動時に Amazon EBS 最適化を有効にするには

[インスタンスを起動] ウィザードで、必要なインスタンスタイプを選択してください。[高度な詳細] セクションを展開し、[EBS 最適化インスタンス] で [有効にする] を選択してください。

選択したインスタンスタイプが Amazon EBS 最適化をサポートしていない場合、ドロップダウンは無効になっています。インスタンスタイプがデフォルトで Amazon EBS 最適化されている場合は、[有効にする] が既に選択されています。

起動後に Amazon EBS 最適化を有効にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Instances] を選択し、インスタンスを選択してください。
3. インスタンスを停止します。[アクション]、[インスタンスの状態]、[インスタンスの停止] の順に選択してください。
4. インスタンスが選択された状態で、[アクション]、[インスタンス設定]、[インスタンスタイプの変更] の順に選択してください。
5. [EBS 最適化] を選択し、[適用] を選択してください。

   インスタンスタイプがデフォルトで Amazon EBS 最適化されている場合、または Amazon ES 最適化をサポートしていない場合は、チェックボックスは無効になっています。
6. インスタンスを再起動します。[インスタンスの状態]、[インスタンスの開始] の順に選択してください。

AWS CLI

起動時に Amazon EBS 最適化を有効にするには

`--ebs-optimized` オプションで run-instances コマンドを使用します。

起動後に Amazon EBS 最適化を有効にするには

1. インスタンスが実行中の場合、stop-instances コマンドを使用して停止します。

   ```
   aws ec2 stop-instances --instance-ids i-1234567890abcdef0
   ```
2. `--ebs-optimized` オプションで modify-instance-attribute コマンドを使用して、EBS 最適化を有効にします。

```
aws ec2 modify-instance-attribute \
    --instance-id i-1234567890abcdef0 \
    --ebs-optimized
```

PowerShell

起動時に Amazon EBS 最適化を有効にするには

`-EbsOptimized` オプションで New-EC2Instance コマンドレットを使用します。

起動後に Amazon EBS 最適化を有効にするには

1. インスタンスが実行中の場合、Stop-EC2Instance コマンドを使用して停止します。

```
Stop-EC2Instance -InstanceId i-1234567890abcdef0
```

2. `-EbsOptimized` オプションで Edit-EC2InstanceAttribute コマンドレットを使用して EBS 最適化を有効にします。

```
Edit-EC2InstanceAttribute `
    -InstanceId i-1234567890abcdef0 `
    -EbsOptimized $true
```

Amazon EC2 インスタンスに関する CPU オプション

多くの Amazon EC2 インスタンスは単一の Intel Xeon CPU コアで同時に複数のスレッドを実行できる同時マルチスレッド (SMT) をサポートしています。各スレッドはインスタンスの仮想 CPU (vCPU) として表されます。インスタンスにはインスタンスタイプによって異なるデフォルト数の CPU コアがあります。例えば、`m5.xlarge` インスタンスタイプには 2 つの CPU コアがあり、デフォルトでは各コアごとに 2 つのスレッドの合計で 4 つの vCPU があります。—

ほとんどの場合、ワークロードに適したメモリと vCPU 数を組み合わせた Amazon EC2 インスタンスタイプがあります。ただし、特定のワークロードやビジネスニーズに合わせてインスタンスを最適化するために、インスタンスの起動中と起動後の両方で以下の CPU オプションを指定できます。

- CPU コア数: インスタンスの CPU コア数をカスタマイズできます。これによって、大量のメモリを使用するワークロード用に十分な RAM 量がありながら、少ない CPU コアのインスタンスのソフトウェアのライセンスコストを最適化することにつながります。

- コア別のスレッド: SMT を無効化するにはCPU コアごとに 1 つのスレッドを指定します。ハイパフォーマンスコンピューティング (HPC) のワークロードのような特定のワークロードでこれを使用できます。

考慮事項

- T2、C7a、M7a、R7a の各インスタンス、Apple シリコン Mac インスタンス、および AWS Graviton プロセッサベースのインスタンスのコアあたりのスレッド数は変更できません。
- 実行できるインスタンスの数は、使用するインスタンスタイプのデフォルトの vCPU に基づきます。インスタンスが消費する vCPU の計算方法は、CPU オプションを変更しても影響を受けません。

料金

CPU オプションの指定に対する追加料金はありません。ライセンス込みの Windows および SQL Server AMI から起動された EC2 インスタンスについては、CPU オプションをカスタマイズして EC2 の CPU 最適化機能を利用することで、インスタンスに設定されている vCPU の数に基づいてライセンス料金を支払うことができます。その他の EC2 インスタンスについては、デフォルトの CPU オプションで起動したインスタンスと同じ料金が請求されます。

内容

- Amazon EC2 インスタンスに関する CPU オプションを指定するためのルール
- Amazon EC2 インスタンスタイプでサポートされている CPU オプション
- Amazon EC2 インスタンスに関する CPU オプションを指定する
- Amazon EC2 インスタンスに関する CPU スレッドとコアを表示する
- ライセンス込みインスタンスの CPU を最適化する

Amazon EC2 インスタンスに関する CPU オプションを指定するためのルール

インスタンスで CPU オプションを指定するには次のルールに注意してください。

- ベアメタルインスタンスには CPU オプションを指定できません。
- インスタンスの起動中と起動後の両方で CPU オプションを指定できます。
- CPU オプションを構成する場合はリクエストで CPU コアの数とコアあたりのスレッド数の両方を指定する必要があります。リクエスト例についてはAmazon EC2 インスタンスに関する CPU オプションを指定するを参照してください。

- インスタンスの vCPU の数はコア別のスレッドで乗算した CPU コアの数です。vCPU のカスタム数を指定するにはインスタンスタイプで CPU およびコア別のスレッドの有効な数を指定する必要があります。インスタンスのデフォルト vCPU の数を超えることはできません。詳細については「Amazon EC2 インスタンスタイプでサポートされている CPU オプション」を参照してください。
- ハイパースレッディングとも呼ばれる同時マルチスレッド (SMT) を無効にするにはコア 1 個あたり 1 つのスレッドを指定します。
- コンソールで、既存のインスタンスのインスタンスタイプを変更すると、Amazon EC2 は、可能であれば既存のインスタンスから新しいインスタンスに CPU オプション設定を適用します。新しいインスタンスタイプがこれらの設定をサポートしない場合、CPU オプションは [デフォルトの CPU オプションを使用する] にリセットされます。このオプションでは、新しいインスタンスタイプのデフォルトの vCPU 数が使用されます。

  新しいインスタンスの設定を更新するには[インスタンスタイプの変更] ビューの [詳細] で [CPU オプションを指定する] を選択してください。
- 指定された CPU オプションはインスタンスの停止、開始あるいは再起動後にも保持されます。
- リザーブドインスタンスを使用する場合、同じ支払者アカウント内でライセンス込みの Windows AMI から起動されたインスタンスに CPU 最適化を設定しても割引が適用されない場合があります。vCPU ベースのライセンスコストを節約し、コンピューティングコストで同等の節約を実現する Savings Plans の使用をお勧めします。
- Windows および SQL Server のライセンス込みの AMI から起動されたインスタンスのライセンスコストを節約するには、少なくとも 4 個の vCPU を設定する必要があります。設定する vCPU が 4 個未満の場合は、デフォルト請求が適用されます。
- T3 インスタンスタイプでは、ライセンス込みのインスタンスに対する CPU 最適化がサポートされません。
- ハードウェア専有インスタンスタイプでは、ライセンス込みのインスタンスに対する CPU 最適化がサポートされません。

Amazon EC2 インスタンスタイプでサポートされている CPU オプション

次の表ではCPU オプションの指定をサポートしているインスタンスタイプを一覧表示しています。

内容

- 汎用インスタンス
- コンピュート最適化インスタンス

- メモリ最適化インスタンス
- ストレージ最適化インスタンス
- 高速コンピューティングインスタンス
- ハイパフォーマンスコンピューティングインスタンス

汎用インスタンス

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m2.xlarge	2	2	1	1、2	1
m2.2xlarge	4	4	1	1、2、3、4	1
m2.4xlarge	8	8	1	1、2、3、4、5、6、7、8	1
m3.large	2	1	2	1	1、2
m3.xlarge	4	2	2	1、2	1、2
m3.2xlarge	8	4	2	1、2、3、4	1、2
m4.large	2	1	2	1	1、2
m4.xlarge	4	2	2	1、2	1、2
m4.2xlarge	8	4	2	1、2、3、4	1、2
m4.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m4.10xlarge	40	20	2	2、4、6、8、10、12、14、16、18、20	1、2
m4.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
m5.large	2	1	2	1	1、2
m5.xlarge	4	2	2	2	1、2
m5.2xlarge	8	4	2	2、4	1、2
m5.4xlarge	16	8	2	2、4、6、8	1、2
m5.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
m5.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
m5.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m5.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
m5a.large	2	1	2	1	1、2
m5a.xlarge	4	2	2	2	1、2
m5a.2xlarge	8	4	2	2、4	1、2
m5a.4xlarge	16	8	2	2、4、6、8	1、2
m5a.8xlarge	32	16	2	4、6、8、10、12、14、16	1、2
m5a.12xlarge	48	24	2	6、12、18、24	1、2
m5a.16xlarge	64	32	2	8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
m5a.24xlarge	96	48	2	12、18、24、36、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m5ad.large	2	1	2	1	1、2
m5ad.xlarge	4	2	2	2	1、2
m5ad.2xlarge	8	4	2	2、4	1、2
m5ad.4xlarge	16	8	2	2、4、6、8	1、2
m5ad.8xlarge	32	16	2	4、6、8、10、12、14、16	1、2
m5ad.12xlarge	48	24	2	6、12、18、24	1、2
m5ad.16xlarge	64	32	2	8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
m5ad.24xlarge	96	48	2	12、18、24、36、48	1、2
m5d.large	2	1	2	1	1、2
m5d.xlarge	4	2	2	2	1、2
m5d.2xlarge	8	4	2	2、4	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m5d.4xlarge	16	8	2	2、4、6、8	1、2
m5d.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
m5d.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
m5d.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
m5d.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
m5dn.large	2	1	2	1	1、2
m5dn.xlarge	4	2	2	1、2	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m5dn.2xlarge	8	4	2	2、4	1、2
m5dn.4xlarge	16	8	2	2、4、6、8	1、2
m5dn.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
m5dn.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
m5dn.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
m5dn.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
m5n.large	2	1	2	1	1、2
m5n.xlarge	4	2	2	1、2	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m5n.2xlarge	8	4	2	2、4	1、2
m5n.4xlarge	16	8	2	2、4、6、8	1、2
m5n.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
m5n.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
m5n.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
m5n.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
m5zn.large	2	1	2	1	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m5zn.xlarge	4	2	2	1、2	1、2
m5zn.2xlarge	8	4	2	2、4	1、2
m5zn.3xlarge	12	6	2	2、4、6	1、2
m5zn.6xlarge	24	12	2	2、4、6、8、10、12	1、2
m5zn.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
m6a.large	2	1	2	1	1、2
m6a.xlarge	4	2	2	1、2	1、2
m6a.2xlarge	8	4	2	1、2、3、4	1、2
m6a.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
m6a.8xlarge	32	16	2	4、6、8、10、12、14、16	1、2
m6a.12xlarge	48	24	2	1、2、3、4、5、6、7、8、16、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6a.16xlarge	64	32	2	4、6、8、10、12、14、16、32	1、2
m6a.24xlarge	96	48	2	4、6、8、10、12、14、16、32、48	1、2
m6a.32xlarge	128	64	2	8、12、16、20、24、28、32、64	1、2
m6a.48xlarge	192	96	2	8、12、16、20、24、28、32、64、96	1、2
m6g.large	2	2	1	1、2	1
m6g.xlarge	4	4	1	1、2、3、4	1
m6g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
m6g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
m6g.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
m6gd.large	2	2	1	1、2	1
m6gd.xlarge	4	4	1	1、2、3、4	1
m6gd.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6gd.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
m6gd.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
m6gd.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6gd.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
m6i.large	2	1	2	1	1、2
m6i.xlarge	4	2	2	1、2	1、2
m6i.2xlarge	8	4	2	2、4	1、2
m6i.4xlarge	16	8	2	2、4、6、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6i.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
m6i.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
m6i.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
m6i.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6i.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
m6id.large	2	1	2	1	1、2
m6id.xlarge	4	2	2	1、2	1、2
m6id.2xlarge	8	4	2	2、4	1、2
m6id.4xlarge	16	8	2	2、4、6、8	1、2
m6id.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
m6id.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6id.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
m6id.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
m6id.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
m6idn.large	2	1	2	1	1、2
m6idn.xlarge	4	2	2	1、2	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6idn.2xlarge	8	4	2	1、2、3、4	1、2
m6idn.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
m6idn.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
m6idn.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
m6idn.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6idn.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
m6idn.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
m6in.large	2	1	2	1	1、2
m6in.xlarge	4	2	2	1、2	1、2
m6in.2xlarge	8	4	2	1、2、3、4	1、2
m6in.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6in.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
m6in.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
m6in.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
m6in.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6in.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
m7a.large	2	2	1	1、2	1
m7a.xlarge	4	4	1	1、2、3、4	1
m7a.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
m7a.4xlarge	16	16	1	1、2、4、6、8、10、12、14、16	1
m7a.8xlarge	32	32	1	1、2、3、4、8、12、16、20、24、28、32	1
m7a.12xlarge	48	48	1	1、2、3、4、5、6、12、18、24、30、36、42、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m7a.16xlarge	64	64	1	1、2、3、4、5、6、7、8、16、24、32、40、48、56、64	1
m7a.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、24、36、48、60、72、84、96	1
m7a.32xlarge	128	128	1	4、6、8、10、12、14、16、32、48、64、80、96、112、128	1
m7a.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、48、72、96、120、144、168、192	1
m7g.large	2	2	1	1、2	1
m7g.xlarge	4	4	1	1、2、3、4	1
m7g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m7g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
m7g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
m7g.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m7g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
m7gd.large	2	2	1	1、2	1
m7gd.xlarge	4	4	1	1、2、3、4	1
m7gd.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m7gd.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
m7gd.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
m7gd.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m7gd.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
m7i.large	2	1	2	1	1、2
m7i.xlarge	4	2	2	1、2	1、2
m7i.2xlarge	8	4	2	1、2、3、4	1、2
m7i.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m7i.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
m7i.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
m7i.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m7i.24xlarge	96	48	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1、2
m7i.48xlarge	192	96	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m7i-flex.large	2	1	2	1	1、2
m7i-flex.xlarge	4	2	2	1、2	1、2
m7i-flex.2xlarge	8	4	2	1、2、3、4	1、2
m7i-flex.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
m7i-flex.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
m7i-flex.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m7i-flex.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
m8a.medium	1	1	1	1	1
m8a.large	2	2	1	1、2	1
m8a.xlarge	4	4	1	1、2、3、4	1
m8a.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
m8a.4xlarge	16	16	1	1、2、4、6、8、10、12、14、16	1
m8a.8xlarge	32	32	1	1、2、3、4、8、12、16、20、24、28、32	1
m8a.12xlarge	48	48	1	1、2、3、4、5、6、12、18、24、30、36、42、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8a.16xlarge	64	64	1	1、2、3、4、5、6、7、8、16、24、32、40、48、56、64	1
m8a.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、24、36、48、60、72、84、96	1
m8a.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、48、72、96、120、144、168、192	1
m8azn.medium	1	1	1	1	1
m8azn.large	2	2	1	1、2	1
m8azn.xlarge	4	4	1	1、2、3、4	1
m8azn.3xlarge	12	12	1	1、2、4、6、8、10、12	1
m8azn.6xlarge	24	24	1	1、2、3、4、8、12、16、20、24	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8azn.12xlarge	48	48	1	1、2、3、4、5、6、7、8、16、24、32、40、48	1
m8azn.24xlarge	96	96	1	4、6、8、10、12、14、16、32、48、64、80、96	1
m8g.large	2	2	1	1、2	1
m8g.xlarge	4	4	1	1、2、3、4	1
m8g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
m8g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
m8g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8g.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8g.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8g.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96、98、100、102、104、106、108、110、112、114、116、118、120、122、124、126、128、130、132、134、136、138、140、142、144、146、148、150、152、154、156、158、160、162、164、166、168、170、172	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				、174、176、178、180、182、184、186、188、190、192	
m8gb.large	2	2	1	1、2	1
m8gb.xlarge	4	4	1	1、2、3、4	1
m8gb.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
m8gb.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
m8gb.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8gb.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8gb.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8gb.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8gb.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96、98、100、102、104、106、108、110、112、114、116、118、120、122、124、126、128、130、132、134、136、138、140、142、144、146、148、150、152、154、156、158、160、162、164、166、168、170、172	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				、174、176、178、180、182、184、186、188、190、192	
m8gd.large	2	2	1	1、2	1
m8gd.xlarge	4	4	1	1、2、3、4	1
m8gd.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
m8gd.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
m8gd.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8gd.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8gd.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8gd.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8gd.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96、98、100、102、104、106、108、110、112、114、116、118、120、122、124、126、128、130、132、134、136、138、140、142、144、146、148、150、152、154、156、158、160、162、164、166、168、170、172	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				、174、176、178、180、182、184、186、188、190、192	
m8gn.large	2	2	1	1、2	1
m8gn.xlarge	4	4	1	1、2、3、4	1
m8gn.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
m8gn.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
m8gn.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8gn.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8gn.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8gn.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8gn.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96、98、100、102、104、106、108、110、112、114、116、118、120、122、124、126、128、130、132、134、136、138、140、142、144、146、148、150、152、154、156、158、160、162、164、166、168、170、172	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				、174、176、178、180、182、184、186、188、190、192	
m8i.large	2	1	2	1	1、2
m8i.xlarge	4	2	2	1、2	1、2
m8i.2xlarge	8	4	2	1、2、3、4	1、2
m8i.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
m8i.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
m8i.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8i.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
m8i.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
m8i.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8i.48xlarge	192	96	2	6、9、12、15、18、21、24、27、30、33、36、39、42、45、48、51、54、57、60、63、66、69、72、75、78、81、84、87、90、93、96	1、2
m8i.96xlarge	384	192	2	12、18、24、30、36、42、48、54、60、66、72、78、84、90、96、102、108、114、120、126、132、138、144、150、156、162、168、174、180、186、192	1、2
m8id.large	2	1	2	1	1、2
m8id.xlarge	4	2	2	1、2	1、2
m8id.2xlarge	8	4	2	1、2、3、4	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8id.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
m8id.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
m8id.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
m8id.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8id.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
m8id.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
m8id.48xlarge	192	96	2	6、9、12、15、18、21、24、27、30、33、36、39、42、45、48、51、54、57、60、63、66、69、72、75、78、81、84、87、90、93、96	1、2

インスタンスタイプ	デフォルトvCPU	デフォルトのCPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8id.96xlarge	384	192	2	12、18、24、30、36、42、48、54、60、66、72、78、84、90、96、102、108、114、120、126、132、138、144、150、156、162、168、174、180、186、192	1、2
m8i-flex.large	2	1	2	1	1、2
m8i-flex.xlarge	4	2	2	1、2	1、2
m8i-flex.2xlarge	8	4	2	1、2、3、4	1、2
m8i-flex.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
m8i-flex.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8i-flex.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
m8i-flex.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
m8in.large	2	1	2	1	1、2
m8in.xlarge	4	2	2	1、2	1、2
m8in.2xlarge	8	4	2	1、2、3、4	1、2
m8in.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8in.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
m8in.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
m8in.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
m8in.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルトvCPU	デフォルトのCPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8in.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
m8in.48xlarge	192	96	2	6、9、12、15、18、21、24、27、30、33、36、39、42、45、48、51、54、57、60、63、66、69、72、75、78、81、84、87、90、93、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8in.96xlarge	384	192	2	12、18、24、30、36、42、48、54、60、66、72、78、84、90、96、102、108、114、120、126、132、138、144、150、156、162、168、174、180、186、192	1、2
m8idn.large	2	1	2	1	1、2
m8idn.xlarge	4	2	2	1、2	1、2
m8idn.2xlarge	8	4	2	1、2、3、4	1、2
m8idn.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
m8idn.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8idn.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
m8idn.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
m8idn.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8idn.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
m8idn.48xlarge	192	96	2	6、9、12、15、18、21、24、27、30、33、36、39、42、45、48、51、54、57、60、63、66、69、72、75、78、81、84、87、90、93、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8idn.96xlarge	384	192	2	12、18、24、30、36、42、48、54、60、66、72、78、84、90、96、102、108、114、120、126、132、138、144、150、156、162、168、174、180、186、192	1、2
m8ine.large	2	1	2	1	1、2
m8ine.xlarge	4	2	2	1、2	1、2
m8ine.2xlarge	8	4	2	1、2、3、4	1、2
m8ine.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
m8ine.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8ine.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
m8ib.large	2	1	2	1	1、2
m8ib.xlarge	4	2	2	1、2	1、2
m8ib.2xlarge	8	4	2	1、2、3、4	1、2
m8ib.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
m8ib.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
m8ib.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8ib.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
m8ib.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
m8ib.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8ib.48xlarge	192	96	2	6、9、12、15、18、21、24、27、30、33、36、39、42、45、48、51、54、57、60、63、66、69、72、75、78、81、84、87、90、93、96	1、2
m8ib.96xlarge	384	192	2	12、18、24、30、36、42、48、54、60、66、72、78、84、90、96、102、108、114、120、126、132、138、144、150、156、162、168、174、180、186、192	1、2
m8idb.large	2	1	2	1	1、2
m8idb.xlarge	4	2	2	1、2	1、2
m8idb.2xlarge	8	4	2	1、2、3、4	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8idb.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
m8idb.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
m8idb.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
m8idb.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8idb.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
m8idb.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
m8idb.48xlarge	192	96	2	6、9、12、15、18、21、24、27、30、33、36、39、42、45、48、51、54、57、60、63、66、69、72、75、78、81、84、87、90、93、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m8idb.96xlarge	384	192	2	12、18、24、30、36、42、48、54、60、66、72、78、84、90、96、102、108、114、120、126、132、138、144、150、156、162、168、174、180、186、192	1、2
m9g.large	2	2	1	1、2	1
m9g.xlarge	4	4	1	1、2、3、4	1
m9g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
m9g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m9g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
m9g.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m9g.16xlarge	64	64	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m9g.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m9g.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96、98、100、102、104、106、108、110、112、114、116、118、120、122、124、126、128、130、132、134、136、138、140、142、144、146、148、150、152、154、156、158、160、162、164、166、168、170、172	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				、174、176、178、180、182、184、186、188、190、192	
m9gd.large	2	2	1	1、2	1
m9gd.xlarge	4	4	1	1、2、3、4	1
m9gd.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
m9gd.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
m9gd.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m9gd.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1
m9gd.16xlarge	64	64	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m9gd.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m9gd.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96、98、100、102、104、106、108、110、112、114、116、118、120、122、124、126、128、130、132、134、136、138、140、142、144、146、148、150、152、154、156、158、160、162、164、166、168、170、172	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				、174、176、178、180、182、184、186、188、190、192	
t3.nano	2	1	2	1	1、2
t3.micro	2	1	2	1	1、2
t3.small	2	1	2	1	1、2
t3.medium	2	1	2	1	1、2
t3.large	2	1	2	1	1、2
t3.xlarge	4	2	2	2	1、2
t3.2xlarge	8	4	2	2、4	1、2
t3a.nano	2	1	2	1	1、2
t3a.micro	2	1	2	1	1、2
t3a.small	2	1	2	1	1、2
t3a.medium	2	1	2	1	1、2
t3a.large	2	1	2	1	1、2
t3a.xlarge	4	2	2	2	1、2
t3a.2xlarge	8	4	2	2、4	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
t4g.nano	2	2	1	1、2	1
t4g.micro	2	2	1	1、2	1
t4g.small	2	2	1	1、2	1
t4g.medium	2	2	1	1、2	1
t4g.large	2	2	1	1、2	1
t4g.xlarge	4	4	1	1、2、3、4	1
t4g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

コンピュート最適化インスタンス

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c3.large	2	1	2	1	1、2
c3.xlarge	4	2	2	1、2	1、2
c3.2xlarge	8	4	2	1、2、3、4	1、2
c3.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c3.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
c4.large	2	1	2	1	1、2
c4.xlarge	4	2	2	1、2	1、2
c4.2xlarge	8	4	2	1、2、3、4	1、2
c4.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
c4.8xlarge	36	18	2	2、4、6、8、10、12、14、16、18	1、2
c5.large	2	1	2	1	1、2
c5.xlarge	4	2	2	2	1、2
c5.2xlarge	8	4	2	2、4	1、2
c5.4xlarge	16	8	2	2、4、6、8	1、2
c5.9xlarge	36	18	2	2、4、6、8、10、12、14、16、18	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c5.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
c5.18xlarge	72	36	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36	1、2
c5.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
c5a.large	2	1	2	1	1、2
c5a.xlarge	4	2	2	1、2	1、2
c5a.2xlarge	8	4	2	1、2、3、4	1、2
c5a.4xlarge	16	8	2	1、2、3、4、8	1、2
c5a.8xlarge	32	16	2	1、2、3、4、8、12、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c5a.12xlarge	48	24	2	1、2、3、4、8、12、16、20、24	1、2
c5a.16xlarge	64	32	2	1、2、3、4、8、12、16、20、24、28、32	1、2
c5a.24xlarge	96	48	2	1、2、3、4、8、12、16、20、24、28、32、36、40、44、48	1、2
c5ad.large	2	1	2	1	1、2
c5ad.xlarge	4	2	2	1、2	1、2
c5ad.2xlarge	8	4	2	1、2、3、4	1、2
c5ad.4xlarge	16	8	2	1、2、3、4、8	1、2
c5ad.8xlarge	32	16	2	1、2、3、4、8、12、16	1、2
c5ad.12xlarge	48	24	2	1、2、3、4、8、12、16、20、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c5ad.16xlarge	64	32	2	1、2、3、4、8、12、16、20、24、28、32	1、2
c5ad.24xlarge	96	48	2	1、2、3、4、8、12、16、20、24、28、32、36、40、44、48	1、2
c5d.large	2	1	2	1	1、2
c5d.xlarge	4	2	2	2	1、2
c5d.2xlarge	8	4	2	2、4	1、2
c5d.4xlarge	16	8	2	2、4、6、8	1、2
c5d.9xlarge	36	18	2	2、4、6、8、10、12、14、16、18	1、2
c5d.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c5d.18xlarge	72	36	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36	1、2
c5d.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
c5n.large	2	1	2	1	1、2
c5n.xlarge	4	2	2	2	1、2
c5n.2xlarge	8	4	2	2、4	1、2
c5n.4xlarge	16	8	2	2、4、6、8	1、2
c5n.9xlarge	36	18	2	2、4、6、8、10、12、14、16、18	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c5n.18xlarge	72	36	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36	1、2
c6a.large	2	1	2	1	1、2
c6a.xlarge	4	2	2	1、2	1、2
c6a.2xlarge	8	4	2	1、2、3、4	1、2
c6a.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
c6a.8xlarge	32	16	2	4、6、8、10、12、14、16	1、2
c6a.12xlarge	48	24	2	1、2、3、4、5、6、7、8、16、24	1、2
c6a.16xlarge	64	32	2	4、6、8、10、12、14、16、32	1、2
c6a.24xlarge	96	48	2	4、6、8、10、12、14、16、32、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6a.32xlarge	128	64	2	8、12、16、20、24、28、32、64	1、2
c6a.48xlarge	192	96	2	8、12、16、20、24、28、32、64、96	1、2
c6g.large	2	2	1	1、2	1
c6g.xlarge	4	4	1	1、2、3、4	1
c6g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
c6g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
c6g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6g.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
c6gd.large	2	2	1	1、2	1
c6gd.xlarge	4	4	1	1、2、3、4	1
c6gd.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6gd.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
c6gd.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
c6gd.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6gd.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
c6gn.medium	1	1	1	1	1
c6gn.large	2	2	1	1、2	1
c6gn.xlarge	4	4	1	1、2、3、4	1
c6gn.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6gn.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
c6gn.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
c6gn.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6gn.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
c6i.large	2	1	2	1	1、2
c6i.xlarge	4	2	2	1、2	1、2
c6i.2xlarge	8	4	2	2、4	1、2
c6i.4xlarge	16	8	2	2、4、6、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6i.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
c6i.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
c6i.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
c6i.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6i.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
c6id.large	2	1	2	1	1、2
c6id.xlarge	4	2	2	1、2	1、2
c6id.2xlarge	8	4	2	2、4	1、2
c6id.4xlarge	16	8	2	2、4、6、8	1、2
c6id.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
c6id.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6id.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
c6id.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
c6id.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
c6in.large	2	1	2	1	1、2
c6in.xlarge	4	2	2	1、2	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6in.2xlarge	8	4	2	1、2、3、4	1、2
c6in.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
c6in.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
c6in.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
c6in.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6in.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
c6in.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
c7a.large	2	2	1	1、2	1
c7a.xlarge	4	4	1	1、2、3、4	1
c7a.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
c7a.4xlarge	16	16	1	1、2、4、6、8、10、12、14、16	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7a.8xlarge	32	32	1	1、2、3、4、8、12、16、20、24、28、32	1
c7a.12xlarge	48	48	1	1、2、3、4、5、6、12、18、24、30、36、42、48	1
c7a.16xlarge	64	64	1	1、2、3、4、5、6、7、8、16、24、32、40、48、56、64	1
c7a.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、24、36、48、60、72、84、96	1
c7a.32xlarge	128	128	1	4、6、8、10、12、14、16、32、48、64、80、96、112、128	1
c7a.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、48、72、96、120、144、168、192	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7g.large	2	2	1	1、2	1
c7g.xlarge	4	4	1	1、2、3、4	1
c7g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
c7g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
c7g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7g.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
c7gd.large	2	2	1	1、2	1
c7gd.xlarge	4	4	1	1、2、3、4	1
c7gd.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7gd.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
c7gd.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
c7gd.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7gd.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
c7gn.large	2	2	1	1、2	1
c7gn.xlarge	4	4	1	1、2、3、4	1
c7gn.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7gn.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
c7gn.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
c7gn.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7gn.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
c7i.large	2	1	2	1	1、2
c7i.xlarge	4	2	2	1、2	1、2
c7i.2xlarge	8	4	2	1、2、3、4	1、2
c7i.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7i.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
c7i.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
c7i.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7i.24xlarge	96	48	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1、2
c7i.48xlarge	192	96	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7i-flex.large	2	1	2	1	1、2
c7i-flex.xlarge	4	2	2	1、2	1、2
c7i-flex.2xlarge	8	4	2	1、2、3、4	1、2
c7i-flex.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
c7i-flex.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
c7i-flex.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7i-flex.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
c8a.medium	1	1	1	1	1
c8a.large	2	2	1	1、2	1
c8a.xlarge	4	4	1	1、2、3、4	1
c8a.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
c8a.4xlarge	16	16	1	1、2、4、6、8、10、12、14、16	1
c8a.8xlarge	32	32	1	1、2、3、4、8、12、16、20、24、28、32	1
c8a.12xlarge	48	48	1	1、2、3、4、5、6、12、18、24、30、36、42、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8a.16xlarge	64	64	1	1、2、3、4、5、6、7、8、16、24、32、40、48、56、64	1
c8a.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、24、36、48、60、72、84、96	1
c8a.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、48、72、96、120、144、168、192	1
c8g.large	2	2	1	1、2	1
c8g.xlarge	4	4	1	1、2、3、4	1
c8g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
c8g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
c8g.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8g.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8g.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96、98、100、102、104、106、108、110、112、114、116、118、120、122、124、126、128、130、132、134、136、138、140、142、144、146、148、150、152、154、156、158、160、162、164、166、168、170、172	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				、174、176、178、180、182、184、186、188、190、192	
c8gb.large	2	2	1	1、2	1
c8gb.xlarge	4	4	1	1、2、3、4	1
c8gb.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
c8gb.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
c8gb.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8gb.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8gb.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8gb.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96	1

インスタンスタイプ	デフォルトvCPU	デフォルトのCPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8gb.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96、98、100、102、104、106、108、110、112、114、116、118、120、122、124、126、128、130、132、134、136、138、140、142、144、146、148、150、152、154、156、158、160、162、164、166、168、170、172	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				、174、176、178、180、182、184、186、188、190、192	
c8gd.large	2	2	1	1、2	1
c8gd.xlarge	4	4	1	1、2、3、4	1
c8gd.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
c8gd.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
c8gd.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8gd.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8gd.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8gd.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8gd.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96、98、100、102、104、106、108、110、112、114、116、118、120、122、124、126、128、130、132、134、136、138、140、142、144、146、148、150、152、154、156、158、160、162、164、166、168、170、172	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				、174、176、178、180、182、184、186、188、190、192	
c8gn.large	2	2	1	1、2	1
c8gn.xlarge	4	4	1	1、2、3、4	1
c8gn.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
c8gn.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
c8gn.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8gn.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8gn.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8gn.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96	1

インスタンスタイプ	デフォルトvCPU	デフォルトのCPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8gn.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96、98、100、102、104、106、108、110、112、114、116、118、120、122、124、126、128、130、132、134、136、138、140、142、144、146、148、150、152、154、156、158、160、162、164、166、168、170、172	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				、174、176、178、180、182、184、186、188、190、192	
c8i.large	2	1	2	1	1、2
c8i.xlarge	4	2	2	1、2	1、2
c8i.2xlarge	8	4	2	1、2、3、4	1、2
c8i.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
c8i.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
c8i.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8i.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
c8i.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
c8i.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8i.48xlarge	192	96	2	6、9、12、15、18、21、24、27、30、33、36、39、42、45、48、51、54、57、60、63、66、69、72、75、78、81、84、87、90、93、96	1、2
c8i.96xlarge	384	192	2	12、18、24、30、36、42、48、54、60、66、72、78、84、90、96、102、108、114、120、126、132、138、144、150、156、162、168、174、180、186、192	1、2
c8id.large	2	1	2	1	1、2
c8id.xlarge	4	2	2	1、2	1、2
c8id.2xlarge	8	4	2	1、2、3、4	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8id.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
c8id.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
c8id.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
c8id.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8id.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
c8id.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
c8id.48xlarge	192	96	2	6、9、12、15、18、21、24、27、30、33、36、39、42、45、48、51、54、57、60、63、66、69、72、75、78、81、84、87、90、93、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8id.96xlarge	384	192	2	12、18、24、30、36、42、48、54、60、66、72、78、84、90、96、102、108、114、120、126、132、138、144、150、156、162、168、174、180、186、192	1、2
c8i-flex.large	2	1	2	1	1、2
c8i-flex.xlarge	4	2	2	1、2	1、2
c8i-flex.2xlarge	8	4	2	1、2、3、4	1、2
c8i-flex.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
c8i-flex.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8i-flex.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
c8i-flex.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
c8in.large	2	1	2	1	1、2
c8in.xlarge	4	2	2	1、2	1、2
c8in.2xlarge	8	4	2	1、2、3、4	1、2
c8in.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8in.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
c8in.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
c8in.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
c8in.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8in.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
c8in.48xlarge	192	96	2	6、9、12、15、18、21、24、27、30、33、36、39、42、45、48、51、54、57、60、63、66、69、72、75、78、81、84、87、90、93、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8in.96xlarge	384	192	2	12、18、24、30、36、42、48、54、60、66、72、78、84、90、96、102、108、114、120、126、132、138、144、150、156、162、168、174、180、186、192	1、2
c8ine.large	2	1	2	1	1、2
c8ine.xlarge	4	2	2	1、2	1、2
c8ine.2xlarge	8	4	2	1、2、3、4	1、2
c8ine.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
c8ine.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8ine.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
c8ib.large	2	1	2	1	1、2
c8ib.xlarge	4	2	2	1、2	1、2
c8ib.2xlarge	8	4	2	1、2、3、4	1、2
c8ib.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
c8ib.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
c8ib.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8ib.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
c8ib.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
c8ib.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2

インスタンスタイプ	デフォルトvCPU	デフォルトのCPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c8ib.48xlarge	192	96	2	6、9、12、15、18、21、24、27、30、33、36、39、42、45、48、51、54、57、60、63、66、69、72、75、78、81、84、87、90、93、96	1、2
c8ib.96xlarge	384	192	2	12、18、24、30、36、42、48、54、60、66、72、78、84、90、96、102、108、114、120、126、132、138、144、150、156、162、168、174、180、186、192	1、2
c9g.large	2	2	1	1、2	1
c9g.xlarge	4	4	1	1、2、3、4	1
c9g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c9g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
c9g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
c9g.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c9g.16xlarge	64	64	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c9g.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96	1

インスタンスタイプ	デフォルトvCPU	デフォルトのCPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c9g.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96、98、100、102、104、106、108、110、112、114、116、118、120、122、124、126、128、130、132、134、136、138、140、142、144、146、148、150、152、154、156、158、160、162、164、166、168、170、172	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				、174、176、178、180、182、184、186、188、190、192	
c9gd.large	2	2	1	1、2	1
c9gd.xlarge	4	4	1	1、2、3、4	1
c9gd.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
c9gd.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
c9gd.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c9gd.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1
c9gd.16xlarge	64	64	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c9gd.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c9gd.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96、98、100、102、104、106、108、110、112、114、116、118、120、122、124、126、128、130、132、134、136、138、140、142、144、146、148、150、152、154、156、158、160、162、164、166、168、170、172	1

インスタンスタイプ	デフォルトvCPU	デフォルトのCPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				、174、176、178、180、182、184、186、188、190、192	

メモリ最適化インスタンス

インスタンスタイプ	デフォルトvCPU	デフォルトのCPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r3.large	2	1	2	1	1、2
r3.xlarge	4	2	2	1、2	1、2
r3.2xlarge	8	4	2	1、2、3、4	1、2
r3.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
r3.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
r4.large	2	1	2	1	1、2
r4.xlarge	4	2	2	1、2	1、2
r4.2xlarge	8	4	2	1、2、3、4	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r4.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
r4.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
r4.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
r5.large	2	1	2	1	1、2
r5.xlarge	4	2	2	2	1、2
r5.2xlarge	8	4	2	2、4	1、2
r5.4xlarge	16	8	2	2、4、6、8	1、2
r5.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
r5.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r5.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
r5.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
r5a.large	2	1	2	1	1、2
r5a.xlarge	4	2	2	2	1、2
r5a.2xlarge	8	4	2	2、4	1、2
r5a.4xlarge	16	8	2	2、4、6、8	1、2
r5a.8xlarge	32	16	2	4、6、8、10、12、14、16	1、2
r5a.12xlarge	48	24	2	6、12、18、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r5a.16xlarge	64	32	2	8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
r5a.24xlarge	96	48	2	12、18、24、36、48	1、2
r5ad.large	2	1	2	1	1、2
r5ad.xlarge	4	2	2	2	1、2
r5ad.2xlarge	8	4	2	2、4	1、2
r5ad.4xlarge	16	8	2	2、4、6、8	1、2
r5ad.8xlarge	32	16	2	4、6、8、10、12、14、16	1、2
r5ad.12xlarge	48	24	2	6、12、18、24	1、2
r5ad.16xlarge	64	32	2	8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
r5ad.24xlarge	96	48	2	12、18、24、36、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r5b.large	2	1	2	1	1、2
r5b.xlarge	4	2	2	1、2	1、2
r5b.2xlarge	8	4	2	2、4	1、2
r5b.4xlarge	16	8	2	2、4、6、8	1、2
r5b.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
r5b.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
r5b.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
r5b.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r5d.large	2	1	2	1	1、2
r5d.xlarge	4	2	2	2	1、2
r5d.2xlarge	8	4	2	2、4	1、2
r5d.4xlarge	16	8	2	2、4、6、8	1、2
r5d.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
r5d.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
r5d.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
r5d.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r5dn.large	2	1	2	1	1、2
r5dn.xlarge	4	2	2	1、2	1、2
r5dn.2xlarge	8	4	2	2、4	1、2
r5dn.4xlarge	16	8	2	2、4、6、8	1、2
r5dn.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
r5dn.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
r5dn.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r5dn.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
r5n.large	2	1	2	1	1、2
r5n.xlarge	4	2	2	1、2	1、2
r5n.2xlarge	8	4	2	2、4	1、2
r5n.4xlarge	16	8	2	2、4、6、8	1、2
r5n.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
r5n.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
r5n.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r5n.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
r6a.large	2	1	2	1	1、2
r6a.xlarge	4	2	2	1、2	1、2
r6a.2xlarge	8	4	2	1、2、3、4	1、2
r6a.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
r6a.8xlarge	32	16	2	4、6、8、10、12、14、16	1、2
r6a.12xlarge	48	24	2	1、2、3、4、5、6、7、8、16、24	1、2
r6a.16xlarge	64	32	2	4、6、8、10、12、14、16、32	1、2
r6a.24xlarge	96	48	2	4、6、8、10、12、14、16、32、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6a.32xlarge	128	64	2	8、12、16、20、24、28、32、64	1、2
r6a.48xlarge	192	96	2	8、12、16、20、24、28、32、64、96	1、2
r6g.large	2	2	1	1、2	1
r6g.xlarge	4	4	1	1、2、3、4	1
r6g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
r6g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
r6g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6g.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
r6gd.large	2	2	1	1、2	1
r6gd.xlarge	4	4	1	1、2、3、4	1
r6gd.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6gd.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
r6gd.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
r6gd.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6gd.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
r6i.large	2	1	2	1	1、2
r6i.xlarge	4	2	2	1、2	1、2
r6i.2xlarge	8	4	2	2、4	1、2
r6i.4xlarge	16	8	2	2、4、6、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6i.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
r6i.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
r6i.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
r6i.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6i.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
r6id.large	2	1	2	1	1、2
r6id.xlarge	4	2	2	1、2	1、2
r6id.2xlarge	8	4	2	2、4	1、2
r6id.4xlarge	16	8	2	2、4、6、8	1、2
r6id.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
r6id.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6id.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
r6id.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
r6id.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
r6idn.large	2	1	2	1	1、2
r6idn.xlarge	4	2	2	1、2	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6idn.2xlarge	8	4	2	1、2、3、4	1、2
r6idn.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
r6idn.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
r6idn.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
r6idn.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6idn.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
r6idn.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
r6in.large	2	1	2	1	1、2
r6in.xlarge	4	2	2	1、2	1、2
r6in.2xlarge	8	4	2	1、2、3、4	1、2
r6in.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6in.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
r6in.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
r6in.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
r6in.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6in.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
r7a.large	2	2	1	1、2	1
r7a.xlarge	4	4	1	1、2、3、4	1
r7a.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
r7a.4xlarge	16	16	1	1、2、4、6、8、10、12、14、16	1
r7a.8xlarge	32	32	1	1、2、3、4、8、12、16、20、24、28、32	1
r7a.12xlarge	48	48	1	1、2、3、4、5、6、12、18、24、30、36、42、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r7a.16xlarge	64	64	1	1、2、3、4、5、6、7、8、16、24、32、40、48、56、64	1
r7a.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、24、36、48、60、72、84、96	1
r7a.32xlarge	128	128	1	4、6、8、10、12、14、16、32、48、64、80、96、112、128	1
r7a.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、48、72、96、120、144、168、192	1
r7g.large	2	2	1	1、2	1
r7g.xlarge	4	4	1	1、2、3、4	1
r7g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r7g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
r7g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
r7g.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r7g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
r7gd.large	2	2	1	1、2	1
r7gd.xlarge	4	4	1	1、2、3、4	1
r7gd.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r7gd.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
r7gd.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
r7gd.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r7gd.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
r7i.large	2	1	2	1	1、2
r7i.xlarge	4	2	2	1、2	1、2
r7i.2xlarge	8	4	2	1、2、3、4	1、2
r7i.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r7i.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
r7i.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
r7i.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r7i.24xlarge	96	48	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1、2
r7i.48xlarge	192	96	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r7iz.large	2	1	2	1	1、2
r7iz.xlarge	4	2	2	1、2	1、2
r7iz.2xlarge	8	4	2	1、2、3、4	1、2
r7iz.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
r7iz.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
r7iz.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r7iz.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
r7iz.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
r8a.medium	1	1	1	1	1
r8a.large	2	2	1	1、2	1
r8a.xlarge	4	4	1	1、2、3、4	1
r8a.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8a.4xlarge	16	16	1	1、2、4、6、8、10、12、14、16	1
r8a.8xlarge	32	32	1	1、2、3、4、8、12、16、20、24、28、32	1
r8a.12xlarge	48	48	1	1、2、3、4、5、6、12、18、24、30、36、42、48	1
r8a.16xlarge	64	64	1	1、2、3、4、5、6、7、8、16、24、32、40、48、56、64	1
r8a.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、24、36、48、60、72、84、96	1
r8a.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、48、72、96、120、144、168、192	1
r8g.large	2	2	1	1、2	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8g.xlarge	4	4	1	1、2、3、4	1
r8g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
r8g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
r8g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8g.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8g.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8g.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96、98、100、102、104、106、108、110、112、114、116、118、120、122、124、126、128、130、132、134、136、138、140、142、144、146、148、150、152、154、156、158、160、162、164、166、168、170、172	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				、174、176、178、180、182、184、186、188、190、192	
r8gb.large	2	2	1	1、2	1
r8gb.xlarge	4	4	1	1、2、3、4	1
r8gb.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
r8gb.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
r8gb.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8gb.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8gb.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8gb.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96	1

インスタンスタイプ	デフォルトvCPU	デフォルトのCPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8gb.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96、98、100、102、104、106、108、110、112、114、116、118、120、122、124、126、128、130、132、134、136、138、140、142、144、146、148、150、152、154、156、158、160、162、164、166、168、170、172	1

インスタンスタイプ	デフォルトvCPU	デフォルトのCPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				、174、176、178、180、182、184、186、188、190、192	
r8gd.large	2	2	1	1、2	1
r8gd.xlarge	4	4	1	1、2、3、4	1
r8gd.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
r8gd.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
r8gd.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8gd.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8gd.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8gd.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8gd.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96、98、100、102、104、106、108、110、112、114、116、118、120、122、124、126、128、130、132、134、136、138、140、142、144、146、148、150、152、154、156、158、160、162、164、166、168、170、172	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				、174、176、178、180、182、184、186、188、190、192	
r8gn.large	2	2	1	1、2	1
r8gn.xlarge	4	4	1	1、2、3、4	1
r8gn.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
r8gn.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
r8gn.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8gn.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8gn.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8gn.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96	1

インスタンスタイプ	デフォルトvCPU	デフォルトのCPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8gn.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96、98、100、102、104、106、108、110、112、114、116、118、120、122、124、126、128、130、132、134、136、138、140、142、144、146、148、150、152、154、156、158、160、162、164、166、168、170、172	1

インスタンスタイプ	デフォルトvCPU	デフォルトのCPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				、174、176、178、180、182、184、186、188、190、192	
r8i.large	2	1	2	1	1、2
r8i.xlarge	4	2	2	1、2	1、2
r8i.2xlarge	8	4	2	1、2、3、4	1、2
r8i.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
r8i.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
r8i.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8i.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
r8i.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
r8i.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8i.48xlarge	192	96	2	6、9、12、15、18、21、24、27、30、33、36、39、42、45、48、51、54、57、60、63、66、69、72、75、78、81、84、87、90、93、96	1、2
r8i.96xlarge	384	192	2	12、18、24、30、36、42、48、54、60、66、72、78、84、90、96、102、108、114、120、126、132、138、144、150、156、162、168、174、180、186、192	1、2
r8id.large	2	1	2	1	1、2
r8id.xlarge	4	2	2	1、2	1、2
r8id.2xlarge	8	4	2	1、2、3、4	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8id.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
r8id.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
r8id.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
r8id.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8id.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
r8id.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
r8id.48xlarge	192	96	2	6、9、12、15、18、21、24、27、30、33、36、39、42、45、48、51、54、57、60、63、66、69、72、75、78、81、84、87、90、93、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8id.96xlarge	384	192	2	12、18、24、30、36、42、48、54、60、66、72、78、84、90、96、102、108、114、120、126、132、138、144、150、156、162、168、174、180、186、192	1、2
r8i-flex.large	2	1	2	1	1、2
r8i-flex.xlarge	4	2	2	1、2	1、2
r8i-flex.2xlarge	8	4	2	1、2、3、4	1、2
r8i-flex.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
r8i-flex.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8i-flex.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
r8i-flex.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
r8in.large	2	1	2	1	1、2
r8in.xlarge	4	2	2	1、2	1、2
r8in.2xlarge	8	4	2	1、2、3、4	1、2
r8in.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8in.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
r8in.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
r8in.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
r8in.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8in.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
r8in.48xlarge	192	96	2	6、9、12、15、18、21、24、27、30、33、36、39、42、45、48、51、54、57、60、63、66、69、72、75、78、81、84、87、90、93、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8in.96xlarge	384	192	2	12、18、24、30、36、42、48、54、60、66、72、78、84、90、96、102、108、114、120、126、132、138、144、150、156、162、168、174、180、186、192	1、2
r8idn.large	2	1	2	1	1、2
r8idn.xlarge	4	2	2	1、2	1、2
r8idn.2xlarge	8	4	2	1、2、3、4	1、2
r8idn.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
r8idn.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8idn.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
r8idn.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
r8idn.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8idn.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
r8idn.48xlarge	192	96	2	6、9、12、15、18、21、24、27、30、33、36、39、42、45、48、51、54、57、60、63、66、69、72、75、78、81、84、87、90、93、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8idn.96xlarge	384	192	2	12、18、24、30、36、42、48、54、60、66、72、78、84、90、96、102、108、114、120、126、132、138、144、150、156、162、168、174、180、186、192	1、2
r8ib.large	2	1	2	1	1、2
r8ib.xlarge	4	2	2	1、2	1、2
r8ib.2xlarge	8	4	2	1、2、3、4	1、2
r8ib.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
r8ib.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8ib.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
r8ib.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
r8ib.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8ib.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
r8ib.48xlarge	192	96	2	6、9、12、15、18、21、24、27、30、33、36、39、42、45、48、51、54、57、60、63、66、69、72、75、78、81、84、87、90、93、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8ib.96xlarge	384	192	2	12、18、24、30、36、42、48、54、60、66、72、78、84、90、96、102、108、114、120、126、132、138、144、150、156、162、168、174、180、186、192	1、2
r8idb.large	2	1	2	1	1、2
r8idb.xlarge	4	2	2	1、2	1、2
r8idb.2xlarge	8	4	2	1、2、3、4	1、2
r8idb.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
r8idb.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8idb.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
r8idb.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
r8idb.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8idb.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
r8idb.48xlarge	192	96	2	6、9、12、15、18、21、24、27、30、33、36、39、42、45、48、51、54、57、60、63、66、69、72、75、78、81、84、87、90、93、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r8idb.96xlarge	384	192	2	12、18、24、30、36、42、48、54、60、66、72、78、84、90、96、102、108、114、120、126、132、138、144、150、156、162、168、174、180、186、192	1、2
u-3tb1.56xlarge	224	112	2	8、12、16、20、24、28、32、36、40、44、48、52、56、60、64、68、72、76、80、84、88、92、96、100、104、 108、112	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
u-6tb1.56xlarge	224	224	1	16、24、32、40、48、56、64、72、80、88、96、104、112、120、128、136、144、152、160、168、176、184、192、200、208、216、224	1
u-6tb1.112xlarge	448	224	2	16、24、32、40、48、56、64、72、80、88、96、104、112、120、128、136、144、152、160、168、176、184、192、200、208、216、224	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
u-9tb1.112xlarge	448	224	2	16、24、32、40、48、56、64、72、80、88、96、104、112、120、128、136、144、152、160、168、176、184、192、200、208、216、224	1、2
u-12tb1.112xlarge	448	224	2	16、24、32、40、48、56、64、72、80、88、96、104、112、120、128、136、144、152、160、168、176、184、192、200、208、216、224	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
u-18tb1.112xlarge	448	224	2	16、24、32、40、48、56、64、72、80、88、96、104、112、120、128、136、144、152、160、168、176、184、192、200、208、216、224	1、2
u-24tb1.112xlarge	448	224	2	16、24、32、40、48、56、64、72、80、88、96、104、112、120、128、136、144、152、160、168、176、184、192、200、208、216、224	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
u7i-6tb.112xlarge	448	224	2	8、12、16、20、24、28、32、36、40、44、48、52、56、60、64、68、72、76、80、84、88、92、96、100、104、108、112、116、120、124、128、132、136、140、144、148、152、156、160、164、168、172、176、180、184、188、192、196、200、204、208、212、216、220、224	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
u7i-8tb.112xlarge	448	224	2	8、12、16、20、24、28、32、36、40、44、48、52、56、60、64、68、72、76、80、84、88、92、96、100、104、108、112、116、120、124、128、132、136、140、144、148、152、156、160、164、168、172、176、180、184、188、192、196、200、204、208、212、216、220、224	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
u7i-12tb.224xlarge	896	448	2	16、24、32、40、48、56、64、72、80、88、96、104、112、120、128、136、144、152、160、168、176、184、192、200、208、216、224、232、240、248、256、264、272、280、288、296、304、312、320、328、336、344、352、360、368、376、384、392、400、408、416、424、432、440、448	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
u7in-16tb.224xlarge	896	448	2	16、24、32、40、48、56、64、72、80、88、96、104、112、120、128、136、144、152、160、168、176、184、192、200、208、216、224、232、240、248、256、264、272、280、288、296、304、312、320、328、336、344、352、360、368、376、384、392、400、408、416、424、432、440、448	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
u7in-24tb.224xlarge	896	448	2	16、24、32、40、48、56、64、72、80、88、96、104、112、120、128、136、144、152、160、168、176、184、192、200、208、216、224、232、240、248、256、264、272、280、288、296、304、312、320、328、336、344、352、360、368、376、384、392、400、408、416、424、432、440、448	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
u7in-32tb.224xlarge	896	448	2	16、24、32、40、48、56、64、72、80、88、96、104、112、120、128、136、144、152、160、168、176、184、192、200、208、216、224、232、240、248、256、264、272、280、288、296、304、312、320、328、336、344、352、360、368、376、384、392、400、408、416、424、432、440、448	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
u7inh-32tb.480xlarge	1920	960	2	32、48、64、80、96、112、128、144、160、176、192、208、224、240、256、272、288、304、320、336、352、368、384、400、416、432、448、464、480、496、512、528、544、560、576、592、608、624、640、656、672、688、704、720、736、752、768、784、800、816、832、848、864、880、896、912、928、944、960	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x1.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
x1.32xlarge	128	64	2	4、8、12、16、20、24、28、32、36、40、44、48、52、56、60、64	1、2
x1e.xlarge	4	2	2	1、2	1、2
x1e.2xlarge	8	4	2	1、2、3、4	1、2
x1e.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
x1e.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
x1e.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x1e.32xlarge	128	64	2	4、8、12、16、20、24、28、32、36、40、44、48、52、56、60、64	1、2
x2gd.large	2	2	1	1、2	1
x2gd.xlarge	4	4	1	1、2、3、4	1
x2gd.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
x2gd.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
x2gd.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x2gd.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x2gd.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
x2idn.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x2idn.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
x2idn.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
x2iedn.xlarge	4	2	2	1、2	1、2
x2iedn.2xlarge	8	4	2	2、4	1、2
x2iedn.4xlarge	16	8	2	2、4、6、8	1、2
x2iedn.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x2iedn.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
x2iedn.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
x2iedn.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
x2iezn.2xlarge	8	4	2	2、4	1、2
x2iezn.4xlarge	16	8	2	2、4、6、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x2iezn.6xlarge	24	12	2	2、4、6、8、10、12	1、2
x2iezn.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
x2iezn.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
x8g.large	2	2	1	1、2	1
x8g.xlarge	4	4	1	1、2、3、4	1
x8g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
x8g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x8g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
x8g.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x8g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x8g.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x8g.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96、98、100、102、104、106、108、110、112、114、116、118、120、122、124、126、128、130、132、134、136、138、140、142、144、146、148、150、152、154、156、158、160、162、164、166、168、170、172	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				、174、176、178、180、182、184、186、188、190、192	
x8aedz.large	2	2	1	1、2	1
x8aedz.xlarge	4	4	1	1、2、3、4	1
x8aedz.3xlarge	12	12	1	1、2、4、6、8、10、12	1
x8aedz.6xlarge	24	24	1	1、2、3、4、8、12、16、20、24	1
x8aedz.12xlarge	48	48	1	1、2、3、4、5、6、7、8、16、24、32、40、48	1
x8aedz.24xlarge	96	96	1	4、6、8、10、12、14、16、32、48、64、80、96	1
x8i.large	2	1	2	1	1、2
x8i.xlarge	4	2	2	1、2	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x8i.2xlarge	8	4	2	1、2、3、4	1、2
x8i.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
x8i.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
x8i.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
x8i.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x8i.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
x8i.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
x8i.48xlarge	192	96	2	6、9、12、15、18、21、24、27、30、33、36、39、42、45、48、51、54、57、60、63、66、69、72、75、78、81、84、87、90、93、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x8i.64xlarge	256	128	2	8、12、16、20、24、28、32、36、40、44、48、52、56、60、64、68、72、76、80、84、88、92、96、100、104、108、112、116、120、124、128	1、2
x8i.96xlarge	384	192	2	12、18、24、30、36、42、48、54、60、66、72、78、84、90、96、102、108、114、120、126、132、138、144、150、156、162、168、174、180、186、192	1、2
z1d.large	2	1	2	1	1、2
z1d.xlarge	4	2	2	1、2	1、2
z1d.2xlarge	8	4	2	2、4	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
z1d.3xlarge	12	6	2	2、4、6	1、2
z1d.6xlarge	24	12	2	2、4、6、8、10、12	1、2
z1d.12xlarge	48	24	2	4、6、8、10、12、14、16、18、20、22、24	1、2

ストレージ最適化インスタンス

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
d2.xlarge	4	2	2	1、2	1、2
d2.2xlarge	8	4	2	1、2、3、4	1、2
d2.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
d2.8xlarge	36	18	2	2、4、6、8、10、12、14、16、18	1、2
d3.xlarge	4	2	2	1、2	1、2
d3.2xlarge	8	4	2	2、4	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
d3.4xlarge	16	8	2	2、4、6、8	1、2
d3.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
d3en.xlarge	4	2	2	1、2	1、2
d3en.2xlarge	8	4	2	2、4	1、2
d3en.4xlarge	16	8	2	2、4、6、8	1、2
d3en.6xlarge	24	12	2	2、4、6、8、10、12	1、2
d3en.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
d3en.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
h1.2xlarge	8	4	2	1、2、3、4	1、2
h1.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
h1.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
h1.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
i2.xlarge	4	2	2	1、2	1、2
i2.2xlarge	8	4	2	1、2、3、4	1、2
i2.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
i2.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
i3.large	2	1	2	1	1、2
i3.xlarge	4	2	2	1、2	1、2
i3.2xlarge	8	4	2	1、2、3、4	1、2
i3.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i3.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
i3.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
i3en.large	2	1	2	1	1、2
i3en.xlarge	4	2	2	1、2	1、2
i3en.2xlarge	8	4	2	2、4	1、2
i3en.3xlarge	12	6	2	2、4、6	1、2
i3en.6xlarge	24	12	2	2、4、6、8、10、12	1、2
i3en.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i3en.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
i4g.large	2	2	1	1、2	1
i4g.xlarge	4	4	1	1、2、3、4	1
i4g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
i4g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
i4g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i4g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
i4i.large	2	1	2	1	1、2
i4i.xlarge	4	2	2	1、2	1、2
i4i.2xlarge	8	4	2	1、2、3、4	1、2
i4i.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i4i.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
i4i.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
i4i.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
i4i.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i4i.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
i7i.large	2	1	2	1	1、2
i7i.xlarge	4	2	2	1、2	1、2
i7i.2xlarge	8	4	2	1、2、3、4	1、2
i7i.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
i7i.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i7i.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
i7i.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i7i.24xlarge	96	48	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1、2
i7i.48xlarge	192	96	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i7ie.large	2	1	2	1	1、2
i7ie.xlarge	4	2	2	1、2	1、2
i7ie.2xlarge	8	4	2	1、2、3、4	1、2
i7ie.3xlarge	12	6	2	1、2、3、4、5、6	1、2
i7ie.6xlarge	24	12	2	1、2、3、4、5、6、7、8、9、10、11、12	1、2
i7ie.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i7ie.18xlarge	72	36	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36	1、2
i7ie.24xlarge	96	48	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i7ie.48xlarge	192	96	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96	1、2
i8g.large	2	2	1	1、2	1
i8g.xlarge	4	4	1	1、2、3、4	1
i8g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
i8g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i8g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
i8g.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i8g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i8g.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i8g.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96、98、100、102、104、106、108、110、112、114、116、118、120、122、124、126、128、130、132、134、136、138、140、142、144、146、148、150、152、154、156、158、160、162、164、166、168、170、172	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				、174、176、178、180、182、184、186、188、190、192	
i8ge.large	2	2	1	1、2	1
i8ge.xlarge	4	4	1	1、2、3、4	1
i8ge.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
i8ge.3xlarge	12	12	1	1、2、3、4、5、6、7、8、9、10、11、12	1
i8ge.6xlarge	24	24	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i8ge.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i8ge.18xlarge	72	72	1	1、2、3、4、5、 6、 7、8、 9、 10、11、 12、13、 14、15、 16、17、 18、19、 20、21、 22、23、 24、25、 26、27、 28、29、 30、31、 32、33、 34、35、 36、37、 38、39、 40、41、 42、43、 44、45、 46、47、 48、49、 50、51、 52、53、 54、55、 56、57、 58、59、 60、61、 62、63、 64、65、 66、67、 68、	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				69、 70、 71、 72	

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i8ge.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i8ge.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96、98、100、102、104、106、108、110、112、114、116、118、120、122、124、126、128、130、132、134、136、138、140、142、144、146、148、150、152、154、156、158、160、162、164、166、168、170、172	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				、174、176、178、180、182、184、186、188、190、192	
im4gn.large	2	2	1	1、2	1
im4gn.xlarge	4	4	1	1、2、3、4	1
im4gn.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
im4gn.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
im4gn.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
im4gn.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
is4gen.medium	1	1	1	1	1
is4gen.large	2	2	1	1、2	1
is4gen.xlarge	4	4	1	1、2、3、4	1
is4gen.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
is4gen.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
is4gen.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

高速コンピューティングインスタンス

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
dl1.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
dl2q.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
f2.6xlarge	24	12	2	1、2、3、6、9、12	1、2
f2.12xlarge	48	24	2	1、2、3、6、9、12、15、18、21、24	1、2
f2.48xlarge	192	96	2	4、6、8、10、12、24、36、48、60、72、84、96	1、2
g3.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
g3.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
g3.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
g4ad.xlarge	4	2	2	2	1、2
g4ad.2xlarge	8	4	2	2、4	1、2
g4ad.4xlarge	16	8	2	2、4、8	1、2
g4ad.8xlarge	32	16	2	2、4、8、16	1、2
g4ad.16xlarge	64	32	2	2、4、8、16、32	1、2
g4dn.xlarge	4	2	2	2	1、2
g4dn.2xlarge	8	4	2	2、4	1、2
g4dn.4xlarge	16	8	2	2、4、6、8	1、2
g4dn.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
g4dn.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
g4dn.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
g5g.xlarge	4	4	1	1、2、3、4	1
g5g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
g5g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
g5g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
g5g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
g6.xlarge	4	2	2	1、2	1、2
g6.2xlarge	8	4	2	1、2、3、4	1、2
g6.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
g6.8xlarge	32	16	2	1、2、4、6、8、10、12、14、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
g6.12xlarge	48	24	2	1、2、3、6、9、12、15、18、21、24	1、2
g6.16xlarge	64	32	2	1、2、3、4、8、12、16、20、24、28、32	1、2
g6.24xlarge	96	48	2	1、2、3、4、5、6、12、18、24、30、36、42、48	1、2
g6.48xlarge	192	96	2	4、6、8、10、12、24、36、48、60、72、84、96	1、2
g6e.xlarge	4	2	2	1、2	1、2
g6e.2xlarge	8	4	2	1、2、3、4	1、2
g6e.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
g6e.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
g6e.12xlarge	48	24	2	3、6、9、12、15、18、21、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
g6e.16xlarge	64	32	2	4、8、12、16、20、24、28、32	1、2
g6e.24xlarge	96	48	2	6、12、18、24、30、36、42、48	1、2
g6e.48xlarge	192	96	2	12、24、36、48、60、72、84、96	1、2
g6f.large	2	1	2	1	1、2
g6f.xlarge	4	2	2	1、2	1、2
g6f.2xlarge	8	4	2	1、2、3、4	1、2
g6f.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
gr6.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
gr6.8xlarge	32	16	2	1、2、4、6、8、10、12、14、16	1、2
gr6f.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
g7.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
g7.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
g7.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
g7.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
g7.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
g7.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96、98、100、102、104、106、108、110、112、114、116、118、120、122、124、126、128、130、132、134、136、138、140、142、144、146、148、150、152、154、156、158、160、162、164、166、168、170、172	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				、174、176、178、180、182、184、186、188、190、192	
g7e.2xlarge	8	4	2	1、2、3、4	1、2
g7e.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
g7e.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
g7e.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
g7e.24xlarge	96	48	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1、2
g7e.48xlarge	192	96	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
inf1.xlarge	4	2	2	2	1、2
inf1.2xlarge	8	4	2	2、4	1、2
inf1.6xlarge	24	12	2	2、4、6、8、10、12	1、2
inf1.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
inf2.xlarge	4	2	2	1、2	1、2
inf2.8xlarge	32	16	2	4、6、8、10、12、14、16	1、2
inf2.24xlarge	96	48	2	4、6、8、10、12、14、16、32、48	1、2
inf2.48xlarge	192	96	2	4、8、12、16、20、24、28、32、64、96	1、2
p3.2xlarge	8	4	2	1、2、3、4	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
p3.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
p3.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
p3dn.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
p4d.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
p4de.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
p5.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
p5.48xlarge	192	96	2	12、24、36、48、60、72、84、96	1、2
p5e.48xlarge	192	96	2	12、24、36、48、60、72、84、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
p5en.48xlarge	192	96	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
p6-b200.48xlarge	192	96	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
p6-b300.48xlarge	192	96	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96	1、2

インスタンスタイプ	デフォルトvCPU	デフォルトのCPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
p6e-gb200.36xlarge	144	144	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64、65、66、67、68、69、70、71、72、73、74、75、76、77、78、79、80、81、82、83、84、85、86、87、88、89、90、91、92、93、94、95、96、97、98、99、	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				100、101、102、103、104、105、106、107、108、109、110、111、112、113、114、115、116、117、118、119、120、121、122、123、124、125、126、127、128、129、130、131、132、133、134、135、136、137、138、139、140、141、142、143、144	
trn1.2xlarge	8	4	2	2、4	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
trn1.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
trn1n.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
trn2.3xlarge	12	6	2	1、2、3、4、5、6	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
trn2.48xlarge	192	96	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
trn2u.48xlarge	192	96	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96	1、2
vt1.3xlarge	12	6	2	6	1、2
vt1.6xlarge	24	12	2	6、12	1、2
vt1.24xlarge	96	48	2	6、12、48	1、2

ハイパフォーマンスコンピューティングインスタンス

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
hpc6id.32xlarge	64	64	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1
hpc8a.96xlarge	192	192	1	24、48、72、96、120、144、168、192	1

Amazon EC2 インスタンスに関する CPU オプションを指定する

インスタンスの起動中または起動後に CPU オプションを指定できます。

タスク

- 同時マルチスレッドを無効にする
- 起動時の vCPU のカスタム数の指定
- 起動テンプレートでの vCPU のカスタム数の指定
- EC2 インスタンスの CPU オプションを変更する

同時マルチスレッドを無効にする

ハイパースレッディングとも呼ばれる同時マルチスレッド (SMT) を無効にするにはコア 1 個あたり 1 つのスレッドを指定します。

Console

インスタンスの起動時に SMT を無効にするには

1. コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する の手順に従い、必要に応じてインスタンスを設定します。
2. [高度な詳細] を展開し、[CPU オプションの指定] チェックボックスをオンにします。
3. [Core count (コア数)] では必要な CPU コア数を選択してください。この例では`r5.4xlarge` インスタンスにデフォルトの CPU コア数を指定するには8 を選択してください。
4. SMT を無効にするには[コアごとのスレッド] で、[1] を選択してください。
5. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択してください。詳細については「コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する」を参照してください。

AWS CLI

インスタンスの起動時に SMT を無効にするには

run-instances AWS CLI コマンドを使用して、1 パラメータの `ThreadsPerCore` の `--cpu-options` の値を指定します。[`CoreCount`] ではCPU コア数を指定します。この例では`r7i.4xlarge` インスタンスにデフォルトの CPU コア数を指定するには8 の値を選択してください。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type r7i.4xlarge \
    --cpu-options "CoreCount=8,ThreadsPerCore=1" \
    --key-name my-key-pair
```

PowerShell

インスタンスの起動時に SMT を無効にするには

New-EC2Instance コマンドを使用して、`-CpuOptions` パラメータの `ThreadsPerCore` に 1 の値を指定します。[`CoreCount`] ではCPU コア数を指定します。この例では`r7i.4xlarge` インスタンスにデフォルトの CPU コア数を指定するには8 の値を選択してください。

```
New-EC2Instance `
```

```
    -ImageId 'ami-0abcdef1234567890' `
    -InstanceType 'r7i.4xlarge' `
    -CpuOptions @{CoreCount=8; ThreadsPerCore=1} `
    -KeyName 'my-key-pair'
```

> **Note**
>
> 既存のインスタンスの SMT を無効にするにはEC2 インスタンスの CPU オプションを変更する に示すプロセスに従い、コアごとに実行されるスレッドの数を 1 に変更します。

起動時の vCPU のカスタム数の指定

EC2 コンソールまたは AWS CLI からインスタンスを起動する時に、CPU コア数とコアあたりのスレッドの数をカスタマイズできます。このセクションの例では以下のデフォルト設定を持つ `r5.4xlarge` インスタンスタイプを使用します。

- CPU コア: 8
- コアあたりのスレッド数: 2

インスタンスはデフォルトでインスタンスタイプで使用できる vCPU の最大数で起動します。このインスタンスタイプでは合計 16 vCPU (それぞれ 2 つのスレッドを実行する 8 個のコア) です。インスタンスタイプの詳細については「メモリ最適化インスタンス」を参照してください。

次の例では4 つの vCPU で `r5.4xlarge` インスタンスを起動します。

Console

インスタンス起動中に vCPU のカスタム数を指定するには

1. コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する の手順に従い、必要に応じてインスタンスを設定します。
2. [高度な詳細] を展開し、[CPU オプションの指定] チェックボックスをオンにします。
3. 4 つの vCPU を取得するには次のように、2 つの CPU コアおよびコアごとに 2 つのスレッドを指定します。
   - [コアカウント] には 2 を選択してください。
   - [Threads per core (コアごとのスレッド)] には[2] を選択してください。

4. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択してください。詳細については「コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する」を参照してください。

AWS CLI

インスタンス起動中に vCPU のカスタム数を指定するには

run-instances AWS CLI コマンドを使用して、--cpu-options パラメータの CPU コア数およびスレッドの数を指定します。2 つの CPU コアおよびコアごとに 2 つのスレッドを指定すると、4 つの vCPU を取得できます。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type r7i.4xlarge \
    --cpu-options "CoreCount=2,ThreadsPerCore=2" \
    --key-name my-key-pair
```

また、4 つの CPU コアおよびコアごとに 1 つのスレッドを指定 (SMT を無効化) して、4 つの vCPU を取得することもできます。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type r7i.4xlarge \
    --cpu-options "CoreCount=4,ThreadsPerCore=1" \
    --key-name my-key-pair
```

PowerShell

インスタンス起動中に vCPU のカスタム数を指定するには

New-EC2Instance コマンドを使用し、-CpuOptions パラメータに CPU コア数とスレッド数を指定します。2 つの CPU コアおよびコアごとに 2 つのスレッドを指定すると、4 つの vCPU を取得できます。

```
New-EC2Instance `
    -ImageId 'ami-0abcdef1234567890' `
    -InstanceType 'r7i.4xlarge' `
    -CpuOptions @{CoreCount=2; ThreadsPerCore=2} `
```

```
    -KeyName 'my-key-pair'
```

また、4 つの CPU コアおよびコアごとに 1 つのスレッドを指定 (SMT を無効化) して、4 つの vCPU を取得することもできます。

```
New-EC2Instance `
    -ImageId 'ami-0abcdef1234567890' `
    -InstanceType 'r7i.4xlarge' `
    -CpuOptions @{CoreCount=4; ThreadsPerCore=1} `
    -KeyName 'my-key-pair'
```

起動テンプレートでの vCPU のカスタム数の指定

起動テンプレートでインスタンスの CPU コア数とコアごとのスレッドの数をカスタマイズできます。このセクションの例では以下のデフォルト設定を持つ `r5.4xlarge` インスタンスタイプを使用します。

- CPU コア: 8
- コアあたりのスレッド数: 2

インスタンスはデフォルトでインスタンスタイプで使用できる vCPU の最大数で起動します。このインスタンスタイプでは合計 16 vCPU (それぞれ 2 つのスレッドを実行する 8 個のコア) です。インスタンスタイプの詳細については「メモリ最適化インスタンス」を参照してください。

次の例ではvCPU が 4 つの `r5.4xlarge` インスタンスの設定を指定する起動テンプレートを作成します。

Console

起動テンプレートで vCPU のカスタム数を指定するには

1. 「パラメータを指定して起動テンプレートを作成する」の手順に従い、必要に応じて起動テンプレートを設定します。
2. [高度な詳細] を展開し、[CPU オプションの指定] チェックボックスをオンにします。
3. 4 つの vCPU を取得するには次のように、2 つの CPU コアおよびコアごとに 2 つのスレッドを指定します。

    - [コアカウント] には 2 を選択してください。

- [Threads per core (コアごとのスレッド)] には[2] を選択してください。

4. [Summary] (概要) パネルでインスタンスの設定を確認し、[Create launch template] (起動テンプレートの作成) を選択してください。詳細については「Amazon EC2 起動テンプレートにインスタンス起動パラメータを保存する」を参照してください。

AWS CLI

起動テンプレートで vCPU のカスタム数を指定するには

create-launch-template AWS CLI コマンドを使用して、CpuOptions パラメータで CPU コア数およびスレッドの数を指定します。2 つの CPU コアおよびコアごとに 2 つのスレッドを指定すると、4 つの vCPU を取得できます。

```
aws ec2 create-launch-template \
    --launch-template-name TemplateForCPUOptions \
    --version-description CPUOptionsVersion1 \
    --launch-template-data file://template-data.json
```

インスタンスを設定するための起動テンプレートデータ (CPU オプションを含む) を含む、JSON ファイルの例を以下に示します。

```
{
    "NetworkInterfaces": [{
        "AssociatePublicIpAddress": true,
        "DeviceIndex": 0,
        "Ipv6AddressCount": 1,
        "SubnetId": "subnet-0abcdef1234567890"
    }],
    "ImageId": "ami-0abcdef1234567890",
    "InstanceType": "r5.4xlarge",
    "TagSpecifications": [{
        "ResourceType": "instance",
        "Tags": [{
            "Key":"Name",
            "Value":"webserver"
        }]
    }],
    "CpuOptions": {
        "CoreCount":2,
        "ThreadsPerCore":2
    }
```

```
}
```

また、4 つの CPU コアおよびコアごとに 1 つのスレッドを指定 (SMT を無効化) して、4 つの vCPU を取得することもできます。

```
{
    "NetworkInterfaces": [{
        "AssociatePublicIpAddress": true,
        "DeviceIndex": 0,
        "Ipv6AddressCount": 1,
        "SubnetId": "subnet-0abcdef1234567890"
    }],
    "ImageId": "ami-0abcdef1234567890",
    "InstanceType": "r5.4xlarge",
    "TagSpecifications": [{
        "ResourceType": "instance",
        "Tags": [{
            "Key":"Name",
            "Value":"webserver"
        }]
    }],
    "CpuOptions": {
        "CoreCount":4,
        "ThreadsPerCore":1
    }
}
```

PowerShell

起動テンプレートで vCPU のカスタム数を指定するには

New-EC2LaunchTemplate を使用します。

```
New-EC2LaunchTemplate `
    -LaunchTemplateName 'TemplateForCPUOptions' `
    -VersionDescription 'CPUOptionsVersion1' `
    -LaunchTemplateData (Get-Content -Path 'template-data.json' | ConvertFrom-Json)
```

EC2 インスタンスの CPU オプションを変更する

時間の経過とともにニーズが変化し、既存のインスタンスの CPU オプションの設定を変更したい場合があります。インスタンスで実行される各スレッドは仮想 CPU (vCPU) と呼ばれます。Amazon

EC2 コンソール、AWS CLI、API、または SDK で、既存のインスタンスに対して実行される vCPU の数を変更できます。この変更を行う前に、インスタンスの状態を Stopped にする必要があります。

コンソールまたはコマンドラインの手順を表示するには環境に一致するタブを選択してください。API リクエストとレスポンスの情報については「Amazon EC2 API リファレンス」の「ModifyInstanceCpuOptions」の項目を参照してください。

Console

AWS マネジメントコンソールからインスタンスのアクティブな vCPU の数を変更するには次の手順に従います。

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左のナビゲーションペインで、[Instances] (インスタンス) をクリックしてください。これにより、現在の AWS リージョンに定義されているインスタンスのリストが開きます。
3. [インスタンス] の一覧から、インスタンスを選択してください。またはインスタンスリンクを選択して、インスタンスの詳細ページを開くこともできます。
4. インスタンスが実行中の場合はまず進む前に停止する必要があります。[インスタンスの状態] メニューから [インスタンスの停止] を選択してください。
5. vCPU 設定を変更するには、[アクション] メニューの [インスタンス設定] から [CPU オプションの変更] を選択してください。これにより、[CPU オプションの変更] ページが開きます。
6. インスタンスの設定を変更するには次のいずれかの CPU オプションを選択してください。

   [デフォルトの CPU オプションを使用する]

   このオプションは、インスタンスタイプのデフォルト vCPU 数にインスタンスをリセットします。デフォルトではすべての CPU コアのすべてのスレッドを実行します。

   [CPU オプションを指定する]

   このオプションはインスタンスで実行されている vCPU の数の設定を有効にします。
7. [CPU オプションを指定] を選択した場合、[アクティブ vCPU] フィールドが表示されます。
   - 最初のセレクターを使用して、各 CPU コアのスレッド数を設定します。同時マルチスレッドを無効にするには、1 を選択します。
   - 2 番目のセレクターを使用して、インスタンスで実行される CPU の数を設定します。

CPU オプションセレクタを変更すると、次のフィールドが動的に更新されます。

- [アクティブな vCPU]: 行った選択に基づいて、CPU コアの数にコアあたりのスレッドを乗算します。例えば、2 つのスレッドと 4 つのコアを選択した場合、8 つの vCPU となります。
- [vCPU の合計] : インスタンスタイプの vCPU の最大数。例えば、`m6i.4xlarge` インスタンスタイプの場合、これは 16 vCPU (それぞれ 2 つのスレッドを実行する 8 個のコア) です。

8. [変更] を選択して更新を適用します。

AWS CLI

AWS CLI からインスタンスのアクティブな vCPU の数を変更するには次の手順に従います。

modify-instance-cpu-options `--core-count` コマンドを使用して、 パラメータで実行される CPU コアの数と、`--threads-per-core` パラメータでコアごとに実行されるスレッドの数を指定します。

次の例は、指定されたインスタンスで 8 個の vCPU を実行するために `m6i.4xlarge` インスタンスタイプで行うことができる 2 つの設定を説明しています。このインスタンスタイプのデフォルトは 16 vCPU (それぞれ 2 つのスレッドを実行する 8 個のコア) です。

例 1: 4 つの CPU コアを 1 コアあたり 2 つのスレッドで実行し、合計 8 つの vCPU を実行します。

```
aws ec2 modify-instance-cpu-options \
    --instance-id i-1234567890abcdef0 \

    --core-count=4 \
    --threads-per-core=2
```

例 2: コアごとに実行されるスレッドの数を 1 に変更して、同時マルチスレッドを無効にします。結果の設定では合計 8 つの vCPU (コアあたり 1 つのスレッドで 8 つの CPU コア) も実行されます。

```
aws ec2 modify-instance-cpu-options \
    --instance-id 1234567890abcdef0 \
    --core-count=8 \
```

```
    --threads-per-core=1
```

PowerShell

インスタンスのアクティブな vCPU の数を変更するには

Edit-EC2InstanceCpuOption コマンドレットを使用し、-CoreCount パラメータで実行する CPU コアの数を指定し、ThreadsPerCore パラメータでコアごとに実行するスレッドの数を指定します。

例 1: 4 つの CPU コアを 1 コアあたり 2 つのスレッドで実行し、合計 8 つの vCPU を実行します。

```
Edit-EC2InstanceCpuOption `
    -InstanceId 'i-1234567890abcdef0' `
    -CoreCount 4 `
    -ThreadsPerCore 2
```

例 2: コアごとに実行されるスレッドの数を 1 に変更して、同時マルチスレッドを無効にします。結果の設定では合計 8 つの vCPU (コアあたり 1 つのスレッドで 8 つの CPU コア) も実行されます。

```
Edit-EC2InstanceCpuOption `
    -InstanceId 'i-1234567890abcdef0' `
    -CoreCount 8 `
    -ThreadsPerCore 1
```

Amazon EC2 インスタンスに関する CPU スレッドとコアを表示する

インスタンスを記述して、既存のインスタンスの CPU オプションを表示できます。

Console

インスタンスの CPU オプションを表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左ナビゲーションペインで [インスタンス] を選択し、インスタンスを選択してください。
3. [Details (詳細)] タブの [Host and placement group (ホストとプレイスメントグループ)] で、[Number of vCPUs (vCPU の数)] を見つけます。

AWS CLI

インスタンスの CPU オプションを表示するには

describe-instances コマンドを使用します。

```
aws ec2 describe-instances \
    --instance-ids i-1234567890abcdef0 \
    --query Reservations[].Instances[].CpuOptions
```

以下は出力の例です。CoreCount フィールドは、インスタンスのコア数を示しています。ThreadsPerCore フィールドはコア別のスレッド数を示します。

```
[
    {
        "CoreCount": 24,
        "ThreadsPerCore": 2
    },
]
```

PowerShell

インスタンスの CPU オプションを表示するには

Get-EC2Instance コマンドレットを使用します。

```
(Get-EC2Instance `
    -InstanceId 'i-1234567890abcdef0').Instances.CpuOptions
```

以下は出力の例です。

```
AmdSevSnp CoreCount ThreadsPerCore
--------- --------- --------------
             24        2
```

またはCPU 情報を表示するのに、インスタンスに接続し、次のいずれかのシステムツールを使用できます。

- Windows インスタンスでの Windows Task Manager
- Linux インスタンスでの lscpu コマンド

インスタンスの終了を含む、インスタンスにおける設定変更の記録、判断、監査、評価のために、AWS Config を使用できます。詳細については、「AWS Config デベロッパーガイド」の「AWS Configの使用開始」を参照してください。

ライセンス込みインスタンスの CPU を最適化する

Microsoft SQL Server などのワークロードでは、高レベルのメモリと IOPS が必要でも、vCPU 数は少なくしなければならないことがよくあります。AWS は、ほとんどのインフラストラクチャニーズに対応できる幅広いインスタンスタイプを提供しています。しかし、Windows や Microsoft SQL Server の vCPU ベースのライセンスコストを削減するには、同一のメモリ、ストレージ、ネットワーク仕様を維持しながら EC2 インスタンスで実行される vCPU の数をカスタマイズすることが可能です。このアプローチにより、ライセンス込みのワークロードと Bring-Your-Own-License (BYOL) ワークロードの両方において vCPU ベースのライセンスコストを節約できます。コンソールまたは Amazon EC2 API を使用すれば、インスタンスの起動時または変更時に CPU オプションを指定できます。手順については、「Amazon EC2 インスタンスに関する CPU オプションを指定する」を参照してください。

詳細については、SQL Server ワークロードに合わせて CPU を最適化するためのベストプラクティスに関するこのブログ記事を参照してください。

サポートされているライセンスタイプ

CPU 最適化は、ライセンス込みの AMI から起動されたインスタンスについて、以下のタイプのライセンス設定でアクティブ CPU の数に基づく請求をサポートします。ライセンスタイプの詳細については、「AMI 請求情報フィールド」を参照してください。

ライセンス込み AMI インスタンスの請求

ライセンス込み	使用オペレーション	vCPU 時間あたりの料金
Windows サーバー	RunInstances:0002	0.046 USD
Windows Server with SQL Server Enterprise	RunInstances:0102	0.421 USD
Windows Server with SQL Server Standard	RunInstances:0006	0.166 USD

ライセンス込み	使用オペレーション	vCPU 時間あたりの料金
Windows Server with SQL Server Web	RunInstances:0202	0.063 USD

サポートされる購入オプション

CPU 最適化は、ライセンス込みのインスタンスに対して次の購入オプションをサポートしています。

- オンデマンド
- Savings Plans

> **Warning**
>
> リザーブドインスタンスを使用する場合、同じ支払者アカウント内でライセンス込みのインスタンスに CPU 最適化を設定しても割引が適用されない場合があります。vCPU ベースのライセンスコストを節約し、コンピューティングコストで同等の節約を実現する Savings Plans の使用をお勧めします。
>
> 現行の請求エクスペリエンスを維持するためのオプトアウトリストに、2025 年 10 月 15 日より前に、同じインスタンスタイプで Windows および SQL Server に CPU 最適化とリザーブドインスタンスの両方を使用したアカウントが追加されました。CPU 最適化によるライセンスコストの節約を利用するには、AWS サポートセンターに連絡してオプトアウトリストからの削除を依頼してください。

CPU 最適化がライセンス料金を節約する仕組み

以下の例は、CPU 使用率の設定時に実現可能なコスト節約の説明に役立ちます。

例 1: デフォルト請求 ライセンス込みの Windows および SQL Server Enterprise AMI から起動された r7i.8xlarge インスタンスの例で、このインスタンスタイプのデフォルト CPU 設定である 32 個の vCPU が 100 時間実行されます (3,200 vCPU 時間)。

請求書には、使用料金とライセンス料金の両方が含まれる合計料金が記載された明細項目が 1 つ表示されています。

For illustration purposes only Amazon Web Services, Inc. charges by service Total services: 1 Description	Usage Quantity	Total charges $1,558.88 Amount in USD
Elastic Compute Cloud		$1,558.88
|- US East (N. Virginia)		$1,558.88
|- Amazon Elastic Compute Cloud running Windows and SQL Server Enterprise		$1,558.88
| |- $15.5888 per On Demand Windows with SQL Server Enterprise r7i.8xlarge Instance Hour	100 Hrs	$1,558.88

例 2: CPU 最適化請求 ライセンス込みの Windows および SQL Server Enterprise AMI から起動された r7i.8xlarge インスタンスの例です。ライセンス料金を節約するため、アクティブな CPU の数を 16 個の vCPU に削減しました。その後、新しい設定を使用してインスタンスが 100 時間実行されました。

請求書には、次の 2 つの明細項目が表示されています。

請求の説明: Elastic Compute Cloud

最初の明細項目には、100 時間実行された Windows および SQL Server インスタンスのベースラインコスト (211.68 USD) が表示されています。

請求の説明: Amazon EC2 Optimize CPU License Included Third Party Fees

2 番目の明細項目には、請求期間中にアクティブ化されていた vCPU の数に基づくライセンス料金 (673.60 USD) が表示されています。

For illustration purposes only Amazon Web Services, Inc. charges by service Total services: 2 Description	Usage Quantity	Total charges $885.28 Amount in USD
Elastic Compute Cloud		$211.68
|- US East (N. Virginia)		$211.68
|- Amazon Elastic Compute Cloud running Windows and SQL Server Enterprise		$211.68
| |- $2.1168 per On Demand Windows and SQL Server Enterprise without licenses r7i.8xlarge Instance Hr	100 Hrs	$211.68
Amazon EC2 Optimize CPU License Included Third Party Fees		$673.60
|- US East (N. Virginia)		$673.60
|- Amazon Elastic Compute Cloud running Windows and SQL Server Enterprise		$673.60
| |- $0.421 per vCPU-Hour of On Demand Windows with SQL Server licenses on r7i Instance	1,600 vCPU-Hrs	$673.60

例 3: Savings Plans を使用した CPU 最適化請求 ライセンス込みの Windows および SQL Server Enterprise AMI から起動された r7i.8xlarge インスタンスの例です。ライセンス料金を節約するため、アクティブな CPU の数を 16 個の vCPU に削減しました。その後、新しい設定を使用してインスタンスが 100 時間実行されました。

1.60 USD/時間 (四捨五入した金額) のコミットメントが設定された 1 年間の前払いなしの Compute Savings Plan により、Windows および SQL Server インスタンスのベースラインコストを削減する

追加のコスト節約が実現されています。Savings Plan コミットメントにより、r7i.8xlarge インスタンスの 100 時間分の使用全体に Savings Plans 料金 (1.53362 USD/時間) が適用されます。

請求書には、次の明細項目が表示されます。

請求の説明: Savings Plans for Compute usage

最初の明細項目は、100 時間分の使用全体に対する Savings Plan コミットメント (160.00 USD) が表示されています。

請求の説明: Elastic Compute Cloud

2 番目の明細項目には 2 つの仕訳が含まれています。最初の仕訳には、100 時間実行された Windows および SQL Server インスタンスの Savings Plan を使用しなかった場合のベースラインコスト (211.68 USD) が表示されています。2 番目の仕訳には、Compute Savings Plan が適用されたベースラインコスト全額 (-211.68 USD) が表示されており、この明細項目の正味コストをゼロにしています。

請求の説明: Amazon EC2 Optimize CPU License Included Third Party Fees

3 番目の明細項目には、請求期間中にアクティブ化されていた vCPU の数に基づくライセンス料金 (673.60 USD) が表示されています。

For illustration purposes only **Amazon Web Services, Inc. charges by service** **Total services: 3** **Description**	**Usage Quantity**	**Total charges** **$833.60** **Amount in USD**
Savings Plans for Compute usage		$160.00
|- Any		$160.00
|- Compute Savings Plans		$160.00
| |- 1 year No Upfront Compute Savings Plan	100 Hrs	$160.00
Elastic Compute Cloud		$0.00
|- US East (N. Virginia)		$0.00
|- Amazon Elastic Compute Cloud running Windows and SQL Server Enterprise		$0.00
| |- $2.1168 per On Demand Windows and SQL Server Enterprise without licenses r7i.8xlarge Instance Hr	100 Hrs	$211.68
| |- r7i.8xlarge Windows with SQL Server Enterprise without licenses instance usage covered by CSP	100 Hrs	($211.68)
Amazon EC2 Optimize CPU License Included Third Party Fees		$673.60
|- US East (N. Virginia)		$673.60
|- Amazon Elastic Compute Cloud running Windows and SQL Server Enterprise		$673.60
| |- $0.421 per vCPU-Hour of On Demand Windows with SQL Server licenses on r7i Instance	1,600 vCPU-Hrs	$673.60

Amazon EC2 インスタンスの AMD SEV-SNP

AMD Secure Encrypted Virtualization-Secure Nested Paging (AMD SEV-SNP) は以下の特性を持つ CPU 機能です。

- 認証 – AMD SEV-SNP を使用すると、インスタンスの状態と ID に加え、正規の AMD ハードウェアで実行されていることの検証にも使用できる暗号化手段を含む署名済みの認証レポートを取得できます。詳細については、「AMD SEV-SNP による Amazon EC2 インスタンスの認証」を参照してください。
- メモリの暗号化 — AMD EPYC (Milan)、AWS Graviton2、Intel Xeon Scalable (Ice Lake) プロセッサ以降の CPU ではインスタンスメモリは常に暗号化されます。

AMD SEV-SNP は、専有ホストまたは共有テナンシーで使用できます。

内容

- 概念と用語
- 要件
- 考慮事項
- 料金
- AMD SEV-SNP をサポートする EC2 インスタンスタイプの検索
- EC2 インスタンスの AMD SEV-SNP の有効化
- AMD SEV-SNP による Amazon EC2 インスタンスの認証

概念と用語

AMD SEV-SNP の使用を開始する前に、次の概念と用語を理解しておきます。

AMD SEV-SNP 認証レポート

AMD SEV-SNP 認証レポートはインスタンスが CPU に要求できる文書です。AMD SEV-SNP 認証レポートを使用して、インスタンスの状態と ID を検証し、認可された AMD 環境で実行されていることを確認できます。レポートには起動測定値が含まれます。起動測定とはインスタンスの初期起動状態の暗号化ハッシュであり、初期インスタンスのメモリ内容と vCPU の初期状態が含まれます。AMD SEV-SNP 認証レポートは、AMD の信頼ルートにチェーンバックする VCEK 署名 (専有ホスト) または VLEK 署名 (共有テナンシー) によって署名されます。

VCEK

バージョニング付きチップ認証鍵 (VCEK) は AMD が認定したチップごとの署名キーであり、専有ホストで AMD SEV-SNP 認証レポートに署名するために AMD CPU によって使用されます。VCEK 署名は AMD が提供する証明書を使用して検証できます。

VLEK

バージョニング付きロード済認証鍵 (VLEK) は AMD が認定したバージョン付き署名キーであり、共有テナンシーで AMD SEV-SNP 認証レポートに署名するために AMD CPU によって使用されます。VLEK 署名はAMD が提供する証明書を使用して検証できます。

OVMF バイナリ

オープン仮想マシンファームウェア (OVMF) はインスタンスに UEFI 環境を提供するために使用されるアーリーブートコードです。アーリーブートコードはAMI のコードが起動する前に実行されます。また、OVMF は AMI で提供されるブートローダーを見つけて実行します。詳細については「OVMF リポジトリ」を参照してください。

要件

AMD SEV-SNP の要件は、専有ホストまたは共有テナンシーのいずれかの使用によって異なります。

専有ホストの要件

専有ホストで AMD SEV-SNP を使用するには、次の要件を満たす必要があります。

- 以下のサポートされているインスタンスタイプのいずれかを使用します。
  - 汎用:: `m6a.large` | `m6a.xlarge` | `m6a.2xlarge` | `m6a.4xlarge` | `m6a.8xlarge`
  - コンピューティングの最適化:: `c6a.large` | `c6a.xlarge` | `c6a.2xlarge` | `c6a.4xlarge` | `c6a.8xlarge` | `c6a.12xlarge` | `c6a.16xlarge`
  - メモリの最適化:: `r6a.large` | `r6a.xlarge` | `r6a.2xlarge` | `r6a.4xlarge`
- AMD SEV-SNP が有効になっている専有ホストを割り当てます。詳細については、「AMD SEV-SNP を持つ専有ホストを割り当てる」を参照してください。
- 任意の商用 AWS リージョン でインスタンスを起動します。
- AMI は`uefi` または `uefi-preferred` ブートモードおよび。AMD SEV-SNP をサポートするオペレーティングシステムで使用してください。AWS についてはAMD SEV-SNP は AL2023、RHEL 9.3、SLES 15 SP4、および Ubuntu 23.04 以降でサポートされています。

共有テナンシーの要件

共有テナンシーで AMD SEV-SNP を使用するには、次の要件を満たす必要があります。

- 以下のサポートされているインスタンスタイプのいずれかを使用します。

  - 汎用:: `m6a.large | m6a.xlarge | m6a.2xlarge | m6a.4xlarge | m6a.8xlarge`
  - コンピューティングの最適化:: `c6a.large | c6a.xlarge | c6a.2xlarge | c6a.4xlarge | c6a.8xlarge | c6a.12xlarge | c6a.16xlarge`
  - メモリの最適化:: `r6a.large | r6a.xlarge | r6a.2xlarge | r6a.4xlarge`
- サポートされている AWS リージョン でインスタンスを起動します: 米国東部 (オハイオ) または欧州 (アイルランド)。
- AMI は`uefi` または `uefi-preferred` ブートモードおよび。AMD SEV-SNP をサポートするオペレーティングシステムで使用してください。AWS についてはAMD SEV-SNP は AL2023、RHEL 9.3、SLES 15 SP4、および Ubuntu 23.04 以降でサポートされています。

考慮事項

AMD SEV-SNP はインスタンスの起動時にのみ有効にすることができます。インスタンスの起動時に AMD SEV-SNP が有効になっている場合は次のルールが適用されます。

- 有効にした後で、AMD SEV-SNP を無効にすることはできません。インスタンスのライフサイクル全体を通して有効なままになります。
- インスタンスタイプの変更はAMD SEV-SNP をサポートする別のインスタンスタイプへのみ可能です。
- 休止と Nitro Enclaves はサポートされていません。

専有ホストに関するその他の考慮事項

専有ホストで AMD SEV-SNP を使用する場合、次のその他の考慮事項が適用されます。

- ファームウェアの更新は、ホストを割り当てるときに適用されます。最新のファームウェアを取得するには、既存のホストを解放して新しいホストを割り当てます。詳細については、「AMD SEV-SNP 専有ホストでファームウェアを更新する」を参照してください。
- AMD SEV-SNP がサポートされている専有ホストで AMD SEV-SNP が有効になっているかどうかを問わず、インスタンスを起動できます。
- ホストリソースグループと専有ホスト予約はサポートされています。

共有テナンシーに関するその他の考慮事項

共有テナンシーで AMD SEV-SNP を使用する場合、次のその他の考慮事項が適用されます。

- インスタンスの基盤となるホストがメンテナンスの予定になっている場合はイベントの 14 日前に予定されているイベント通知が届きます。インスタンスを新しいホストに移動するにはインスタンスを手動で停止または再起動する必要があります。

料金

専用ホスト

専有ホストで AMD SEV-SNP の使用には、追加料金はかかりません。標準の専有ホスト料金のみをお支払いいただきます。詳細については、「専有ホストの料金」を参照してください。

共有テナンシー

共有テナンシーで AMD SEV-SNP が有効になっている Amazon EC2 インスタンスを起動すると、選択したインスタンスタイプの時間単位のオンデマンド料金の 10% に相当する追加の時間単位の使用料が発生します。この AMD SEV-SNP 使用料はAmazon EC2 インスタンスの使用料とは別に請求されます。リザーブドインスタンス、Savings Plans、およびオペレーティングシステムの使用量はこの料金に影響しません。

共有テナンシーで AMD SEV-SNP が有効になっているスポットインスタンスを起動するように設定した場合、選択したインスタンスタイプの時間単位のオンデマンド料金の 10% に相当する追加の時間単位の使用料が発生します。配分戦略で価格を入力として使用する場合、スポットフリートにはこの追加料金は含まれず、スポット料金のみが使用されます。

AMD SEV-SNP をサポートする EC2 インスタンスタイプの検索

AMD SEV-SNP をサポートするインスタンスタイプを検索できます。Amazon EC2 コンソールでは、インスタンスタイプに対してこの情報は表示されません。

AWS CLI

AMD SEV-SNP をサポートするインスタンスタイプを検索するには

次の describe-instance-types コマンドを使用します。

```
aws ec2 describe-instance-types \
    --filters Name=processor-info.supported-features,Values=amd-sev-snp \
    --query 'InstanceTypes[*].[InstanceType]' \
    --output text | sort
```

以下は出力の例です。

```
c6a.12xlarge
c6a.16xlarge
c6a.2xlarge
c6a.4xlarge
c6a.8xlarge
c6a.large
c6a.xlarge
m6a.2xlarge
m6a.4xlarge
m6a.8xlarge
m6a.large
m6a.xlarge
r6a.2xlarge
r6a.4xlarge
r6a.large
r6a.xlarge
```

PowerShell

AMD SEV-SNP をサポートするインスタンスタイプを検索するには

Get-EC2InstanceType コマンドレットを使用します。

```
(Get-EC2InstanceType `
    -Filter @{Name="processor-info.supported-features"; Values="amd-sev-
snp"}).InstanceType.Value | Sort-Object
```

以下は出力の例です。

```
c6a.12xlarge
c6a.16xlarge
c6a.2xlarge
c6a.4xlarge
c6a.8xlarge
c6a.large
c6a.xlarge
m6a.2xlarge
m6a.4xlarge
m6a.8xlarge
m6a.large
m6a.xlarge
r6a.2xlarge
r6a.4xlarge
```

```
r6a.large
r6a.xlarge
```

EC2 インスタンスの AMD SEV-SNP の有効化

AMD SEV-SNP を有効にした状態でインスタンスを起動できます。起動後に AMD SEV-SNP を有効にすることはできません。

内容

- AMD SEV-SNP を持つ専有ホストを割り当てる
- AMD SEV-SNP 専有ホストでインスタンスを起動する
- AMD SEV-SNP 専有ホストでファームウェアを更新する
- 共有テナンシーで AMD SEV-SNP を持つインスタンスを起動する
- EC2 インスタンスで AMD SEV-SNP が有効になっているかどうかの確認

AMD SEV-SNP を持つ専有ホストを割り当てる

専有ホストで AMD SEV-SNP を持つインスタンスを起動する前に、AMD SEV-SNP サポートが有効になっている専有ホストを割り当てる必要があります。専有ホストをインスタンスファミリー別 (m6a、r6a、c6a など) に割り当てたら、ホスト上でそのファミリーのサポートされている任意のインスタンスタイプを実行できます。AMD SEV-SNP を持つ専有ホストは、任意の商用 AWS リージョン に割り当てることができます。

Console

AMD SEV-SNP が有効になっている専有ホストを割り当てるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Dedicated Hosts] を選択してください。
3. [専有ホストの割り当て] を選択します。
4. [AMD SEV-SNP] には、[有効化] を選択します。
5. 必要に応じて残りのホストを設定したら、[割り当て] を選択します。

AWS CLI

AMD SEV-SNP が有効になっている専有ホストを割り当てるには

--instance-type と --cpu-options パラメータを持つ allocate-hosts コマンドを使用します。

```
aws ec2 allocate-hosts \
    --instance-type "m6a.large" \
    --availability-zone "us-east-1a" \
    --auto-placement "off" \
    --cpu-options AmdSevSnp=enabled \
    --quantity 1
```

以下は出力の例です。

```
{
    "HostIds": [
        "h-0123456789abcdef0"
    ]
}
```

> **Note**
>
> AMD SEV-SNP が有効になっている専有ホストを割り当てると、AWS が最新のファームウェアを適用している間に、ホストは configuring 状態になります。数分から最大 2 時間かかる場合があります。configuring 状態のときに課金されます。

AMD SEV-SNP 専有ホストでインスタンスを起動する

専有ホストが設定されると、AMD SEV-SNP が有効になっているインスタンスを起動できます。

> **Note**
>
> AMD SEV-SNP で割り当てられた専有ホストで AMD SEV-SNP が有効になっていないインスタンスを起動することもできます。AMD SEV-SNP インスタンスと AMD SEV-SNP 以外のインスタンスは両方とも同じホストで同時に実行できます。

AWS CLI

専有ホストで AMD SEV-SNP を持つインスタンスを起動するには

--cpu-options と --placement オプションで run-instances コマンドを使用します。「*h-0123456789abcdef0*」を割り当てられた専有ホストの ID に置き換えます。

```
aws ec2 run-instances \
    --instance-type m6a.xlarge \
    --image-id ami-0123456789example \
    --cpu-options AmdSevSnp=enabled \
    --placement HostId=h-0123456789abcdef0
```

PowerShell

専有ホストで AMD SEV-SNP を持つインスタンスを起動するには

-CpuOption と -Placement_HostId パラメータで New-EC2Instance コマンドレットを使用します。

```
New-EC2Instance `
    -InstanceType m6a.xlarge `
    -ImageId ami-0123456789example `
    -CpuOption @{AmdSevSnp="enabled"} `
    -Placement_HostId h-0123456789abcdef0
```

AMD SEV-SNP 専有ホストでファームウェアを更新する

AMD SEV-SNP が有効になっている専有ホストを割り当てると、AWS は利用可能な最新のファームウェアをホストに適用します。ファームウェアの更新には、AMD のセキュリティパッチや機能改善が含まれる場合があります。

ファームウェアの更新を適用するには、新しいホスト (最新のファームウェアを受信) を割り当て、インスタンスを新しいホストに移動し、古いホストを解放します。

1. AMD SEV-SNP が有効になっている新しい専有ホストを割り当てます。新しいホストは、利用可能な最新のファームウェアを受信します。詳細については、「AMD SEV-SNP を持つ専有ホストを割り当てる」を参照してください。
2. 古い専有ホストで実行中のインスタンスを停止します。

```
aws ec2 stop-instances \
    --instance-ids i-0123456789example
```

3. 新しい専有ホストをターゲットするように、インスタンスの配置を変更します。modify-instance-placement コマンドを使用します。

```
aws ec2 modify-instance-placement \
    --instance-id i-0123456789example \
    --host-id h-09876543210abcdef
```

4. 新しい専有ホストでインスタンスを起動します。

```
aws ec2 start-instances \
    --instance-ids i-0123456789example
```

5. 古い専有ホストを解放します。release-hosts コマンドを使用します。

```
aws ec2 release-hosts \
    --host-ids h-0123456789abcdef0
```

> **Note**
>
> 専有ホストのメンテナンス通知は、専有ホストの他のメンテナンスイベントを受信する方法と同様に受信します。通知を受信したら、前の手順に従って、ファームウェアが更新された新しいホストに移動します。

共有テナンシーで AMD SEV-SNP を持つインスタンスを起動する

米国東部 (オハイオ) と欧州 (アイルランド) の共有テナンシーで AMD SEV-SNP を持つインスタンスを起動できます。

AWS CLI

共有テナンシーで AMD SEV-SNP を持つインスタンスを起動するには

`--cpu-options` オプションで run-instances コマンドを使用します。

```
aws ec2 run-instances \
    --instance-type m6a.xlarge \
    --image-id ami-0123456789example \
    --cpu-options AmdSevSnp=enabled
```

PowerShell

共有テナンシーで AMD SEV-SNP を持つインスタンスを起動するには

`-CpuOption` パラメータで New-EC2Instance コマンドレットを使用します。

```
New-EC2Instance `
    -InstanceType m6a.xlarge `
    -ImageId ami-0123456789example `
    -CpuOption @{AmdSevSnp="enabled"}
```

EC2 インスタンスで AMD SEV-SNP が有効になっているかどうかの確認

AMD SEV-SNP が有効になっているインスタンスを検索できます。Amazon EC2 コンソールにはこの情報は表示されません。

AWS CLI

インスタンスで AMD SEV-SNP が有効になっているかどうかを確認するには

describe-instances コマンドを使用します。

```
aws ec2 describe-instances \
    --instance-ids i-1234567890abcdef0 \
    --query Reservations[].Instances[].CpuOptions
```

以下は出力の例です。`AmdSevSnp` が `CpuOptions` に存在しない場合、AMD SEV-SNP は無効になります。

```
[
    {
        "AmdSevSnp": "enabled",
        "CoreCount": 1,
        "ThreadsPerCore": 2
    }
]
```

PowerShell

インスタンスで AMD SEV-SNP が有効になっているかどうかを確認するには

Get-EC2Instance コマンドレットを使用します。

```
(Get-EC2Instance `
    -InstanceId i-1234567890abcdef0).Instances.CpuOptions
```

以下は出力の例です。AmdSevSnp の値が存在しない場合、AMD SEV-SNP は無効になります。

```
AmdSevSnp CoreCount ThreadsPerCore
--------- --------- --------------
enabled   1         2
```

AWS CloudTrail

インスタンス起動要求の AWS CloudTrail イベントで、次のプロパティはインスタンスで AMD SEV-SNP が有効になっていることを示します。

```
"cpuOptions": {"AmdSevSnp": "enabled"}
```

AMD SEV-SNP による Amazon EC2 インスタンスの認証

認証はインスタンスがその状態と身元を証明できるようにするプロセスです。インスタンスで AMD SEV-SNP を有効にすると、基盤となるプロセッサに AMD SEV-SNP 認証レポートをリクエストできます。AMD SEV-SNP 認証レポートには初期ゲストメモリーの内容と vCPU の初期状態の起動測定と呼ばれる暗号化ハッシュが含まれています。認証レポートは、AMD の信頼ルートにチェーンバックする VCEK 署名 (専有ホスト) または VLEK 署名 (共有テナンシー) によって署名されます。認証レポートに含まれる起動測定値を使用して、インスタンスが正規の AMD 環境で実行されていることを確認し、インスタンスの起動に使用された初期ブートコードを検証できます。

前提条件

AMD SEV-SNP が有効になっているインスタンスを起動します。詳細については、「EC2 インスタンスの AMD SEV-SNP の有効化」を参照してください。

内容

- 専有ホストでのインスタンスの認証
- 共有テナンシーのインスタンスの認証

専有ホストでのインスタンスの認証

専有ホストは、一意のチップごとの署名キーであるバージョニング付きチップ認証鍵 (VCEK) を使用して、認証レポートに署名します。VCEK 証明書から AMD の信頼チェーンに戻るように、信頼チェーンを検証する必要があります。

> **Note**
>
> 現在サポートされているプロセッサーモデルは `milan` です。

前提条件

Git、Cargo、Perl がインストールされて、AMD SEV-SNP が有効になっている専有ホストで実行中のインスタンス。

ステップ 1: snpguest ユーティリティを構築する

このステップでは、認証レポートの生成、必要な証明書の取得、認証レポートの検証に使用する `snpguest` ユーティリティをインストールして構築します。

1. インスタンスに接続します。
2. 前提条件がインストールされていることを確認します。

```
$ perl --version; cargo --version; git --version
```

3. snpguest repository から `snpguest` ユーティリティを構築するには次のコマンドを実行します。

```
$ git clone https://github.com/virtee/snpguest.git
$ cd snpguest
$ cargo build -r
$ cd target/release
```

ステップ 2: 認証レポートを生成する

認証レポートのリクエストを生成します。`snpguest` ユーティリティはホストを介して AMD Secure Processor から認証レポートをリクエストし、バイナリファイルに書き込みます。次の例では、ランダムなリクエスト文字列を作成して、レポートを `report.bin` に保存します。

```
$ ./snpguest report report.bin request-file.txt --random
```

ステップ 3: VCEK 証明書チェーンを取得して検証する

認証レポートは、専有ホストの AMD チップ固有の VCEK によって署名されます。VCEK から AMD の信頼チェーンに戻るように、信頼チェーンを検証する必要があります。

1. AMD Key Distribution Service (KDS) から VCEK 証明書を取得します。証明書は、認証レポートのチップ ID と TCB バージョンによって識別されます。

   ```
   $ ./snpguest fetch vcek pem ./ ./report.bin --processor-model milan
   ```

2. 信頼チェーンを形成する AMD ルート証明書 (AMD ルートキー (ARK) と AMD SEV キー (ASK)) を取得します。

   ```
   $ ./snpguest fetch ca PEM ./ milan --endorser vcek
   ```

3. 証明書チェーンを確認します。

   ```
   $ ./snpguest verify certs ./
   ```

   以下は出力の例です。

   ```
   The AMD ARK was self-signed!
   The AMD ASK was signed by the AMD ARK!
   The VCEK was signed by the AMD ASK!
   ```

オプション: 透明性を強化するために、AMD から直接証明書をダウンロードして openssl を使用することで、チェーンを個別に検証できます。

```
$ curl --proto '=https' --tlsv1.2 -sSf https://kdsintf.amd.com/vcek/v1/Milan/cert_chain
 -o ./cert_chain.pem
$ openssl verify --CAfile ./cert_chain.pem vcek.pem
```

以下は出力の例です。

```
vcek.pem: OK
```

ステップ 4: 認証レポートの署名を検証する

認証レポートが VCEK 証明書によって署名されていることを確認します。レポートが正規の AMD ハードウェアで生成され、改ざんされていないことが確認されます。

```
$ ./snpguest verify attestation ./ report.bin
```

以下は出力の例です。

```
Reported TCB Boot Loader from certificate matches the attestation report.
Reported TCB TEE from certificate matches the attestation report.
Reported TCB SNP from certificate matches the attestation report.
Reported TCB Microcode from certificate matches the attestation report.
VEK signed the Attestation Report!
```

TCB (Trusted Computing Base) バージョンフィールドは、証明書のファームウェアバージョンが認証レポートで報告されているバージョンと一致することを確認します。最後の行では、VCEK によってレポートが署名されていることを確認できます。

共有テナンシーのインスタンスの認証

共有テナンシーインスタンスは、AMD が AWS に対して発行するバージョニング付きロード済承認鍵 (VLEK) を使用します。VLEK 証明書から AMD の信頼チェーンに戻るように、信頼チェーンを検証する必要があります。

ステップ 1: snpguest ユーティリティを構築する

このステップでは、認証レポートの生成、必要な証明書の取得、認証レポートの検証に使用する snpguest ユーティリティをインストールして構築します。

1. インスタンスに接続します。
2. snpguest repository から snpguest ユーティリティを構築するには次のコマンドを実行します。

   ```
   $ git clone https://github.com/virtee/snpguest.git
   $ cd snpguest
   $ cargo build -r
   $ cd target/release
   ```

ステップ 2: 認証レポートを生成する

認証レポートのリクエストを生成します。snpguest ユーティリティはホストから認証レポートをリクエストし、バイナリファイルに書き込みます。次の例では、ランダムなリクエスト文字列を作成して、レポートを report.bin に保存します。

```
$ ./snpguest report report.bin request-file.txt --random
```

ホストメモリから証明書をリクエストし、PEM ファイルとして保存します。

```
$ ./snpguest certificates PEM ./
```

ステップ 3: VLEK 証明書チェーンを取得して検証する

認証レポートは、AMD が AWS に対して発行する VLEK によって署名されます。VLEK から AMD の信頼チェーンに戻るように、信頼チェーンを検証する必要があります。

1. VLEK の信頼ルート証明書を公式の AMD Key Distribution Service からダウンロードして現在のディレクトリにアップロードします。

   ```
   $ sudo curl --proto '=https' --tlsv1.2 -sSf https://kdsintf.amd.com/vlek/v1/Milan/
   cert_chain -o ./cert_chain.pem
   ```

2. openssl を使用して、VLEK 証明書が AMD の信頼証明書ルートによって署名されていることを確認します。

   ```
   $ sudo openssl verify --CAfile ./cert_chain.pem vlek.pem
   ```

   以下は出力の例です。

   ```
   vlek.pem: OK
   ```

ステップ 4: 認証レポートの署名を検証する

認証レポートが VLEK 証明書によって署名されていることを確認します。レポートが正規の AMD ハードウェアで生成され、改ざんされていないことが確認されます。

```
$ ./snpguest verify attestation ./ report.bin
```

以下は出力の例です。

```
Reported TCB Boot Loader from certificate matches the attestation report.
Reported TCB TEE from certificate matches the attestation report.
Reported TCB SNP from certificate matches the attestation report.
Reported TCB Microcode from certificate matches the attestation report.
VEK signed the Attestation Report!
```

Amazon EC2 Linux インスタンスのプロセッサのステート制御

C ステートはアイドル時のコアのスリープレベルを制御します。C ステートはC0 (コアがアクティブで、命令を実行している最も浅い状態) から始まる番号が付けられ、C6 (コアの電源がオフになっている最も深いアイドル状態) まで移行します。

P ステートはコアに希望するパフォーマンス (CPU 周波数) を制御します。P ステートはP0 (コアが Intel Turbo Boost Technology を使用して可能であれば周波数を上げることができる最高パフォーマンスの設定) から始まる番号が付けられ、P1 (最大限のベースライン周波数をリクエストする P ステート) から P15 (最小限の周波数) まで移行します。

> **Note**
>
> AWS Graviton プロセッサには組み込みの省電力モードがあり、固定周波数で動作します。そのため、オペレーティングシステムが C ステートと P ステートを制御する機能は提供されていません。

C ステートと P ステート

次のインスタンスタイプにより、オペレーティングシステムが C ステートと P ステートを制御できるようになります。

- 汎用: `m4.10xlarge` | `m4.16xlarge`
- コンピューティング最適化: `c4.8xlarge`
- メモリ最適化: `r4.8xlarge` | `r4.16xlarge` | `x1.16xlarge` | `x1.32xlarge` | `x1e.8xlarge` | `x1e.16xlarge` | `x1e.32xlarge`
- ストレージ最適化: `d2.8xlarge` | `i3.8xlarge` | `i3.16xlarge` | `h1.8xlarge` | `h1.16xlarge`
- 高速コンピューティング `g3.16xlarge` | `p3.16xlarge`
- ベアメタル: Intel および AMD プロセッサを搭載したすべてのベアメタルインスタンス

C ステートのみ

次のインスタンスタイプにより、オペレーティングシステムが C ステートを制御できるようになります。

- 汎用: m5.12xlarge | m5.24xlarge | m5d.12xlarge | m5d.24xlarge | m5n.12xlarge | m5n.24xlarge | m5dn.12xlarge | m5dn.24xlarge | m5zn.6xlarge | m5zn.12xlarge | m6a.24xlarge | m6a.48xlarge | m6i.16xlarge | m6i.32xlarge | m6id.16xlarge | m6id.32xlarge | m6idn.16xlarge | m6in.16xlarge | m6in.32xlarge | m7a.medium | m7a.large | m7a.xlarge | m7a.2xlarge | m7a.4xlarge | m7a.8xlarge | m7a.12xlarge | m7a.16xlarge | m7a.24xlarge | m7a.32xlarge | m7a.48xlarge | m7i.large | m7i.xlarge | m7i.2xlarge | m7i.4xlarge | m7i.8xlarge | m7i.12xlarge | m7i.16xlarge | m7i.24xlarge | m7i.48xlarge | m8a.medium | m8a.large | m8a.xlarge | m8a.2xlarge | m8a.4xlarge | m8a.8xlarge | m8a.12xlarge | m8a.16xlarge | m8a.24xlarge | m8a.48xlarge | m8azn.medium | m8azn.large | m8azn.xlarge | m8azn.3xlarge | m8azn.6xlarge | m8azn.12xlarge | m8azn.24xlarge | m8i.large | m8i.xlarge | m8i.2xlarge | m8i.4xlarge | m8i.8xlarge | m8i.12xlarge | m8i.16xlarge | m8i.24xlarge | m8i.32xlarge | m8i.48xlarge | m8i.96xlarge | m8id.large | m8id.xlarge | m8id.2xlarge | m8id.4xlarge | m8id.8xlarge | m8id.12xlarge | m8id.16xlarge | m8id.24xlarge | m8id.32xlarge | m8id.48xlarge | m8id.96xlarge | m8in.large | m8in.xlarge | m8in.2xlarge | m8in.4xlarge | m8in.8xlarge | m8in.12xlarge | m8in.16xlarge | m8in.24xlarge | m8in.32xlarge | m8in.48xlarge | m8in.96xlarge | m8idn.large | m8idn.xlarge | m8idn.2xlarge | m8idn.4xlarge | m8idn.8xlarge | m8idn.12xlarge | m8idn.16xlarge | m8idn.24xlarge | m8idn.32xlarge | m8idn.48xlarge | m8idn.96xlarge | m8ib.large | m8ib.xlarge | m8ib.2xlarge | m8ib.4xlarge | m8ib.8xlarge | m8ib.12xlarge | m8ib.16xlarge | m8ib.24xlarge | m8ib.32xlarge | m8ib.48xlarge | m8ib.96xlarge | m8idb.large | m8idb.xlarge | m8idb.2xlarge | m8idb.4xlarge | m8idb.8xlarge | m8idb.12xlarge | m8idb.16xlarge | m8idb.24xlarge | m8idb.32xlarge | m8idb.48xlarge | m8idb.96xlarge | m8ine.large | m8ine.xlarge | m8ine.2xlarge | m8ine.4xlarge | m8ine.8xlarge | m8ine.12xlarge
- コンピューティング最適化: c5.9xlarge | c5.12xlarge | c5.18xlarge | c5.24xlarge | c5a.24xlarge | c5ad.24xlarge | c5d.9xlarge | c5d.12xlarge | c5d.18xlarge | c5d.24xlarge | c5n.9xlarge | c5n.18xlarge | c6a.24xlarge | c6a.32xlarge | c6a.48xlarge | c6i.16xlarge | c6i.32xlarge | c6id.24xlarge | c6id.32xlarge | c6in.32xlarge | c7a.medium | c7a.large | c7a.xlarge | c7a.2xlarge | c7a.4xlarge | c7a.8xlarge | c7a.12xlarge | c7a.16xlarge | c7a.24xlarge | c7a.32xlarge |

c7a.48xlarge | c7i.large | c7i.xlarge | c7i.2xlarge | c7i.4xlarge | c7i.8xlarge | c7i.12xlarge | c7i.16xlarge | c7i.24xlarge | c7i.48xlarge | c8a.medium | c8a.large | c8a.xlarge | c8a.2xlarge | c8a.4xlarge | c8a.8xlarge | c8a.12xlarge | c8a.16xlarge | c8a.24xlarge | c8a.48xlarge | c8i.large | c8i.xlarge | c8i.2xlarge | c8i.4xlarge | c8i.8xlarge | c8i.12xlarge | c8i.16xlarge | c8i.24xlarge | c8i.32xlarge | c8i.48xlarge | c8i.96xlarge | c8id.large | c8id.xlarge | c8id.2xlarge | c8id.4xlarge | c8id.8xlarge | c8id.12xlarge | c8id.16xlarge | c8id.24xlarge | c8id.32xlarge | c8id.48xlarge | c8id.96xlarge | c8in.large |c8in.xlarge | c8in.2xlarge | c8in.4xlarge |c8in.8xlarge | c8in.12xlarge | c8in.16xlarge | c8in.24xlarge | c8in.32xlarge | c8in.48xlarge |c8in.96xlarge | c8ib.large |c8ib.xlarge | c8ib.2xlarge | c8ib.4xlarge | c8ib.8xlarge | c8ib.12xlarge | c8ib.16xlarge |c8ib.24xlarge | c8ib.32xlarge | c8ib.48xlarge |c8ib.96xlarge | c8in.large |c8in.xlarge | c8in.2xlarge | c8in.4xlarge |c8in.8xlarge | c8in.12xlarge | x8i.large | x8i.xlarge | x8i.2xlarge | x8i.4xlarge | x8i.8xlarge | x8i.12xlarge | x8i.16xlarge | x8i.24xlarge | x8i.32xlarge | x8i.48xlarge | x8i.48xlarge | x8i.96xlarge

- メモリ最適化: r5.12xlarge | r5.24xlarge | r5b.12xlarge | r5d.12xlarge | r5d.24xlarge | r5n.12xlarge | r5n.24xlarge | r5dn.12xlarge | r5dn.24xlarge | r6a.24xlarge | r6a.48xlarge | r6i.16xlarge | r6i.32xlarge | r6id.16xlarge | r6id.32xlarge | r6in.16xlarge | r6in.32xlarge | r7a.medium | r7a.large | r7a.xlarge | r7a.2xlarge | r7a.4xlarge | r7a.8xlarge | r7a.12xlarge | r7a.16xlarge | r7a.24xlarge | r7a.32xlarge | r7a.48xlarge | r7i.large | r7i.xlarge | r7i.2xlarge | r7i.4xlarge | r7i.8xlarge | r7i.12xlarge | r7i.16xlarge | r7i.24xlarge | r7i.48xlarge | r7iz.large | r7iz.xlarge | r7iz.2xlarge | r7iz.4xlarge | r7iz.8xlarge | r7iz.12xlarge | r7iz.16xlarge | r7iz.32xlarge | r8a.medium | r8a.large | r8a.xlarge | r8a.2xlarge | r8a.4xlarge | r8a.8xlarge | r8a.12xlarge | r8a.16xlarge | r8a.24xlarge | r8a.48xlarge | r8i.large | r8i.xlarge | r8i.2xlarge | r8i.4xlarge | r8i.8xlarge | r8i.12xlarge | r8i.16xlarge | r8i.24xlarge | r8i.32xlarge | r8i.48xlarge | r8i.96xlarge | r8id.large | r8id.xlarge | r8id.2xlarge | r8id.4xlarge | r8id.8xlarge | r8id.12xlarge | r8id.16xlarge | r8id.24xlarge | r8id.32xlarge | r8id.48xlarge | r8id.96xlarge | r8in.large | r8in.xlarge | r8in.2xlarge | r8in.4xlarge | r8in.8xlarge | r8in.12xlarge | r8in.16xlarge | r8in.24xlarge | r8in.32xlarge | r8in.48xlarge | r8in.96xlarge | r8idn.large | r8idn.xlarge | r8idn.2xlarge | r8idn.4xlarge | r8idn.8xlarge | r8idn.12xlarge | r8idn.16xlarge | r8idn.24xlarge | r8idn.32xlarge | r8idn.48xlarge | r8idn.96xlarge | r8ib.large

| r8ib.xlarge | r8ib.2xlarge | r8ib.4xlarge | r8ib.8xlarge | r8ib.12xlarge | r8ib.16xlarge | r8ib.24xlarge | r8ib.32xlarge | r8ib.48xlarge | r8ib.96xlarge | r8idb.large | r8idb.xlarge | r8idb.2xlarge | r8idb.4xlarge | r8idb.8xlarge | r8idb.12xlarge | r8idb.16xlarge | r8idb.24xlarge | r8idb.32xlarge | r8idb.48xlarge | r8idb.96xlarge | u-3tb1.56xlarge | u-6tb1.56xlarge | u-6tb1.112xlarge | u-9tb1.112xlarge | u-12tb1.112xlarge | u-18tb1.112xlarge | u-24tb1.112xlarge | u7i-6tb.112xlarge | u7i-8tb.112xlarge | u7i-12tb.224xlarge | u7in-16tb.224xlarge | u7in-24tb.224xlarge | u7in-32tb.224xlarge | u7inh-32tb.480xlarge | x2idn.32xlarge | x2iedn.16xlarge | x2iedn.32xlarge | x2iezn.12xlarge | x8aedz.large | x8aedz.xlarge | x8aedz.3xlarge | x8aedz.6xlarge | x8aedz.12xlarge | x8aedz.24xlarge | z1d.6xlarge | z1d.12xlarge

- ストレージ最適化: d3en.12xlarge | dl1.24xlarge | i3en.12xlarge | i3en.24xlarge | i4i.16xlarge | i7i.large | i7i.xlarge | i7i.2xlarge | i7i.4xlarge | i7i.8xlarge | i7i.12xlarge | i7i.16xlarge | i7i.24xlarge | i7i.48xlarge | i7ie.large | i7ie.xlarge | i7ie.2xlarge | i7ie.3xlarge | i7ie.6xlarge | i7ie.12xlarge | i7ie.18xlarge | i7ie.24xlarge | i7ie.48xlarge | r5b.12xlarge | r5b.24xlarge
- 高速コンピューティング: dl1.24xlarge | f2.6xlarge | f2.12xlarge | f2.48xlarge | g5.24xlarge | g5.48xlarge | g6.24xlarge | g6.48xlarge | g6e.12xlarge | g6e.24xlarge | g6e.48xlarge | g7e.2xlarge | g7e.4xlarge | g7e.8xlarge | g7e.12xlarge | g7e.24xlarge | g7e.48xlarge | g7.2xlarge | g7.4xlarge | g7.8xlarge | g7.12xlarge | g7.24xlarge | g7.48xlarge | inf1.24xlarge | p3dn.24xlarge | p4d.24xlarge | p4de.24xlarge | p5.48xlarge | p5e.48xlarge | p5en.48xlarge | p6-b200.48xlarge | p6-b300.48xlarge | trn1.32xlarge | trn2.3xlarge | trn2.48xlarge | trn2a.3xlarge | trn2a.48xlarge | trn2n.3xlarge | trn2n.48xlarge | trn2p.48xlarge | trn2u.48xlarge | vt1.24xlarge
- ハイパフォーマンスコンピューティング: hpc7a.12xlarge | hpc7a.24xlarge | hpc7a.48xlarge | hpc7a.96xlarge | hpc8a.96xlarge

プロセッサのパフォーマンスの安定性を向上させたり、レイテンシーを減らしたり、インスタンスを特定のワークロード用に調整するために、C ステートまたは P ステートの設定を変更したいと思う場合があるかもしれません。デフォルトの C ステートおよび P ステートの設定はほとんどの作業負荷に対して最適なパフォーマンスを提供します。ただし、アプリケーションにおいて、より高いシングルコアまたはデュアルコアの周波数でレイテンシーを軽減したい場合、またはバースト的な Turbo Boost 周波数よりも低い周波数でより安定したパフォーマンスを維持することを優先する場合、これらのインスタンスで利用可能な C ステートまたは P ステートを試みることを考慮してください。

さまざまなプロセッサ設定および、Amazon Linux の設定の影響をモニタリングする方法については Amazon Linux 2 ユーザーガイドの「Amazon EC2 Amazon Linux インスタンスのプロセッサ状態制御」を参照してください。これらの手順は Amazon Linux を対象としており、Amazon Linux に適用されるものですが、バージョン 3.9 以降の Linux カーネルのある、他の Linux ディストリビューションでも使用できる可能性があります。他の Linux ディストリビューションやプロセッサのステート制御についてはシステム固有ドキュメントを参照してください。

Amazon EC2 マネージドインスタンス

Amazon EC2 マネージドインスタンスはEKS 自動モードによる Amazon EKS などの指定されたサービスプロバイダーによってプロビジョニングおよび管理される EC2 インスタンスです。マネージドインスタンスはAmazon EC2 でコンピューティングワークロードを実行するための簡単な方法を提供します。これにより、インスタンスの運用管理をサービスプロバイダーに委任できます。

管理されたインスタンスに導入される唯一の変更は委任コントロールです。技術仕様と請求は非管理された EC2 インスタンスと同じです。管理されたインスタンスを使用すると、サービスプロバイダーに制御を委任できるため、サービスプロバイダーの運用上の専門知識とベストプラクティスを活用できます。インスタンスが管理されている場合、サービスプロバイダーはインスタンスのプロビジョニング、ソフトウェアの設定、容量のスケーリング、インスタンスの障害と置換の処理、インスタンスの終了などのタスクを担当します。

マネージドインスタンスの設定を直接変更したり、終了したりすることはできません。サービスと特定のオペレーションはお客様とサービスプロバイダーの間の契約によって決まります。ただし、管理されたインスタンスのタグを追加、変更、または削除して、AWS環境内で分類できます。

内容

- 管理されたインスタンスの請求
- 管理されたインスタンスを特定する
- マネージドリソースの可視性設定
- インスタンスの使用を開始する

管理されたインスタンスの請求

Amazon EC2 マネージドインスタンスには非マネージド Amazon EC2 インスタンスと同じ基本料金と、サービスプロバイダーの別の料金が発生します。この追加料金はインスタンスを管理するサービスプロバイダーによって課金され、個別に請求されます。管理されたインスタンスの運用と保守のために提供されるサービスのコストをカバーします。

オンデマンドインスタンス、リザーブドインスタンス、スポットインスタンス、Savings Plans など、すべての Amazon EC2 購入オプションがマネージドインスタンスで使用できます。コンピューティングを EC2 から直接調達し、サービスプロバイダーに提供することで、アカウントに適用されている既存のリザーブドインスタンスまたは Savings Plans のメリットを享受できるため、利用可能な最も費用対効果の高いコンピューティングキャパシティーを使用できます。

例えば、Amazon EKS Auto Mode を使用する場合、基盤となるインスタンスの標準の EC2 インスタンス料金に加えて、ユーザーに代わってインスタンスを管理するための Amazon EKS からの追加料金を支払います。その後、Savings Plans にサインアップすると、EC2 インスタンス料金は Savings Plans によって削減されますが、Amazon EKS からの追加料金は変更されません。

管理されたインスタンスを特定する

管理されたインスタンスは管理された フィールドの 正 値によって識別されます。サービスプロバイダーはオペレータフィールド (コンソール) または `Principal`フィールド (CLI) で識別されます。

管理されたインスタンスを識別するには次の手順に従います。

Console

管理されたインスタンスを識別するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. 削除するインスタンスを選択してください。
4. 詳細タブ (チェックボックスを選択した場合) または概要エリア (インスタンス ID を選択した場合) で、管理対象フィールドを見つけます。
   - 正 の値は管理されたインスタンスを示します。
   - 誤 の値は非管理されたインスタンスを示します。
5. 管理された が 正 に設定されている場合、オペレータフィールドにはインスタンスの管理を担当するサービスプロバイダーを識別する値が表示されます。例えば、eks.amazonaws.com の値は Amazon EKS をサービスプロバイダーとして識別します。

AWS CLI

管理されたインスタンスを識別するには

describe-instances コマンドを使用して、インスタンス ID を指定します。

```
aws ec2 describe-instances \
    --instance-ids i-1234567890abcdef0 \
    --query Reservations[].Instances[].Operator
```

以下は出力の例です。Managed が true である場合、インスタンスはマネージドインスタンスであり、Principal が含まれています。プリンシパルはインスタンスを管理するサービスプロバイダーです。例えば、 の値は Amazon EKS をサービスプロバイダーとしてeks.amazonaws.com識別します。

```
[
    {
        "Managed": true,
        "Principal": "eks.amazonaws.com"
    }
]
```

マネージドインスタンスを見つけるには

describe-instances コマンドを使用して、operator.managed および フィルターを持つパラメータを指定します true。--query オプションには、マネージドインスタンスの ID のみが表示されます。

```
aws ec2 describe-instances \
    --filters "Name=operator.managed,Values=true" \
    --query Reservations[*].Instances[].InstanceId
```

PowerShell

管理されたインスタンスを識別するには

Get-EC2Instance コマンドレットを使用します。

```
(Get-EC2Instance -InstanceId i-1234567890abcdef0).Instances.Operator
```

以下は出力の例です。

```
Managed Principal
------- ---------
True    eks.amazonaws.com
```

マネージドインスタンスを見つけるには

Get-EC2Instance コマンドレットを使用します。この例では、マネージドインスタンスの ID のみが表示されます。

```
(Get-EC2Instance -Filter @{Name="operator.managed";
 Values="true"}).Instances.InstanceId
```

マネージドリソースの可視性設定

AWS サービスがユーザーに代わってプロビジョニングするリソースが Amazon EC2 コンソールビューおよび API リストオペレーションに表示されるかどうかを制御できます。

マネージドリソースの可視性とは

Amazon EKS、Amazon ECS、WorkSpaces Core、AWS Lambda などの AWS サービスは、アカウント内で Amazon EC2 インスタンスを直接プロビジョニングおよび運用します。これらのサービスは、スケーリング、OS パッチ、セキュリティ更新、ライフサイクル管理の責任を負います。結果として得られる Amazon EC2 インスタンス、Amazon EC2 起動テンプレート、Amazon EBS ボリューム、ネットワークインターフェイス (ENI) は、Amazon EC2 コンソールと API のカスタマーマネージドリソースとともに表示されます。マネージドリソースの可視性設定により、これらのマネージドリソースがリソースビューに表示されるかどうかを制御できます。

影響を受けるリソースタイプ

リソースタイプ	これらのリソースをプロビジョニングするサービス	説明
Amazon EC2 インスタンス	Amazon EKS ワーカーノード、Amazon ECS コンテナインスタンス、AWS Lambda 実行環境、Amazon WorkSpaces Core	可視性設定の影響を受けるプライマリリソースタイプ
Amazon EC2 起動テンプレート	Amazon EKS、Amazon ECS、Amazon WorkSpaces Core	マネージドサービスによって作成された起動テンプレート
Amazon EBS ボリューム	Amazon EKS、Amazon ECS、Amazon WorkSpaces Core	マネージドインスタンスにアタッチされたボリューム

リソースタイプ	これらのリソースをプロビジョニングするサービス	説明
ネットワークインターフェイス (ENI)	Amazon EKS、Amazon ECS、AWSLambda、Amazon WorkSpaces Core	マネージドワークロード用にプロビジョニングされたネットワークインターフェイス

> **Note**
>
> デフォルトでは、Amazon EC2 は、可視性設定が利用可能になる前にマネージドリソースを保有していなかったアカウントについて、それらのリソースを非表示にします。すでにマネージドリソースが存在するアカウントについては、Amazon EC2 は既存のワークフローを維持するため、可視性設定を 表示 に設定します。可視性設定は、作成時期にかかわらず、リージョン内のすべてのマネージドリソースに適用されます。可視性設定はいつでも変更できます。

可視性設定を構成する理由

可視性設定を構成すると、運用ツール全体でマネージドリソースがどのように表示されるかを調整できます。一般的ユースケースには以下が含まれます。

- コンプライアンスダッシュボードのリソース数をカスタマーマネージドリソースのみに減らすことで、ガバナンスを簡素化します。
- アカウント内のすべてのインスタンスで Amazon EC2 メトリクスを集約するオブザーバビリティツールのノイズを減らします。
- マネージドリソースを顧客の設定ミスとしてフラグ付けするクラウドセキュリティ体制管理 (CSPM) スキャナー (Qualys など) の誤検出を防止します。
- マネージドインスタンスでは、AWS が Amazon EC2 インスタンスの設定、パッチ適用、正常性に責任を持ちます。可視性を制御することで、エンドユーザーに責任共有モデルをより明確に表現できます。

> **Note**
>
> 可視性設定は、AWS コンソールビューと API リストオペレーションでのリソースの表示を制御します。請求、リソースオペレーション、実際のアクセス許可には影響しません。非表示のリソースは完全に動作し、請求可能です。

マネージドリソースの可視性を設定

Amazon EC2 コンソールまたは AWS CLI を使用して、マネージドリソースの表示設定を行うことができます。

Console

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、ダッシュボードを選択してください。
3. アカウント属性カードの設定で、マネージドリソースを選択します。
4. [Modify] (変更) を選択します。
5. デフォルトの可視性で、次のいずれかのオプションを選択します。
   - 非表示 (デフォルト) – すべてのマネージドリソースを非表示にします。
   - 表示 – すべてのマネージドリソースを表示します。
6. 可視性の変更を選択します。

AWS CLI

現在の可視性設定を取得

get-managed-resource-visibility コマンドを使用して、現在の可視性設定を取得します。

```
aws ec2 get-managed-resource-visibility
```

レスポンスの例:

```
{
    "visibility": {
        "defaultVisibility": "hidden"
    }
```

```
}
```

すべてのマネージドリソースを非表示にする

modify-managed-resource-visibility コマンドを使用して、演算子に関係なくすべてのマネージドリソースを非表示にします。

```
aws ec2 modify-managed-resource-visibility \
    --default-visibility "hidden"
```

非表示のマネージドリソースを検出

可視性をオフにしても、マネージドリソースにアクセスできます。次のメソッドは、オンデマンドでそれらを表示します。

1. サービス固有のコンソール: それぞれの AWS サービスコンソール (Amazon EKS コンソールなど) に移動して、そのサービス用にプロビジョニングされたインスタンスを表示します。サービスコンソールには、サービスがアカウントで管理するすべてのリソースの詳細が表示されます。
2. 直接 API クエリ: 特定の `instance-id` パラメータで `describe-instances` API を使用します。既知のインスタンス ID を持つ直接クエリは、可視性設定に関係なく結果を返します。可視性設定は、リストオペレーションとフィルターオペレーションにのみ影響します。`include-managed-resources` パラメータで `describe-instances` を使用して、マネージドインスタンスを検出することもできます。

> **Note**
>
> 同一の direct-query-by-ID 動作は、影響を受けるすべてのリソースタイプに適用されます。`describe-volumes`、`describe-launch-templates`、`describe-network-interfaces` を特定のリソース ID とともに使用して、それらのタイプの非表示のマネージドリソースにアクセスできます。

請求に関する考慮事項

マネージドリソースの可視性設定は、請求には影響しません。非表示のマネージドインスタンスは、アカウント内で実行され、ユーザーに代わってプロビジョニングされ、可視性設定に関係なく完全に請求可能なリソースであるため、引き続き請求データに表示されます。

非表示のリソースは、次の場所に表示されます。

- AWS の請求書
- AWS コストと使用状況レポート

> **Important**
>
> マネージドインスタンスはアカウントにプロビジョニングされ、コンピューティングリソースを消費します。コンソールビューから非表示にしても、コストは削減されません。マネージドインスタンスの料金の詳細については、サービス固有の請求ドキュメント (Amazon EKS 料金表、Amazon ECS 料金表など) を参照してください。

制限事項

- 可視性設定はアカウント全体に適用され、すべての IAM プリンシパルに均一に影響します。
- リソースタイプまたはリソースを作成したサービスによって、マネージドリソースを選択的に表示または非表示にすることはできません。たとえば、Lambda、Amazon ECS、Amazon WorkSpaces によって作成されたマネージドインスタンスを非表示にしながら、Amazon EKS によって作成されたマネージドインスタンスを表示するように選択することはできません。

インスタンスの使用を開始する

マネージドインスタンスの使用に関するガイダンスについては「Amazon EKS ユーザーガイド」の「EKS 自動モードでクラスターインフラストラクチャを自動化する」を参照してください。

ネストされた仮想化を使用して Amazon EC2 インスタンスでハイパーバイザーを実行する

ネストされた仮想化は、仮想 Amazon EC2 インスタンス内での Hyper-V や KVM などのハイパーバイザーの実行を可能にします。仮想 EC2 インスタンスはベアメタル以外のインスタンスです。この機能は、仮想 EC2 インスタンスにプロセッサレベルの仮想化サポートを追加することで仮想化の柔軟性を高めるため、インスタンス内で実行されているハイパーバイザーが仮想マシンを作成して管理できるようになります。

ネストされた仮想化は、特定のパフォーマンス要件と料金要件を満たす幅広い標準的な仮想 Amazon EC2 インスタンスタイプから選択できるため、開発ワークフローでの Docker Desktop、Windows Subsystem for Linux 2 (WSL2)、Android Studio エミュレータ、または QEMU などの開発ツールの実行時に役立ちます。

ネストされた仮想化の使用に追加料金はかかりません。

内容

- 仕組み
- 考慮事項
- ネストされた仮想化を有効にした新規インスタンスを起動する
- ネストされた仮想化を使用するように既存のインスタンスを設定する

仕組み

仮想 EC2 インスタンスは、Nitro Hypervisor を使用する物理ホストで実行されます。ネストされた仮想化をサポートするため、Nitro System はプロセッサ拡張機能 (Intel VT-x など) をインスタンスに渡して、ネストされた仮想マシンの実行を円滑化します。ネストされた仮想化のアーキテクチャは、物理的な AWS インフラストラクチャと Nitro Hypervisor (L0)、ハイパーバイザーを実行する EC2 インスタンス (L1)、そのインスタンス内で作成された 1 つ、または複数の仮想マシン (L2) の 3 つのレイヤーで構成されています。

考慮事項

ネストされた仮想化の使用を開始する前に、以下の点を考慮してください。

- サポートされているインスタンスタイプ – ネストされた仮想化は現在、C8i、M8i、R8i、C8id、R8id、M8id、C8i-flex、R8i-flex、M8i-flex、X8i、C7i、R7i、M7i、C7i-flex、M7i-flex、I7i インスタンスでサポートされています。
- サポートされるハイパーバイザー – 現在サポートされている L1 ハイパーバイザーは、KVM と Hyper-V です。
- Windows インスタンス – Windows インスタンスでネストされた仮想化が有効化されている場合:
  - Credential Guard – Virtual Secure Mode (VSM) は自動的に無効化されます。
  - ハイバネーション – インスタンスの休止と再開はサポートされません。
  - CPU 制限 – CPU 数が 192 個を超える Windows インスタンス (`m8i.96xl` など) ではサポートされません。

- セキュリティ上の責任 – EC2 インスタンスでネストされた仮想化を使用する場合は、AWS が「クラウドのセキュリティ」に対する責任を担うことで基盤となるインフラストラクチャを保護し、AWS Nitro System が提供する EC2 インスタンス間の強固な分離境界を維持します。お客様は「クラウド内のセキュリティ」に対する責任を担います。これには、EC2 インスタンス内のオペレーティングシステム、ハイパーバイザー、ネストされた仮想マシン、ゲストオペレーティングシステム、アプリケーション、およびデータのセキュア化が含まれます。
- パフォーマンス – ハードウェア仮想化拡張機能へのアクセスを必要とし、パフォーマンス依存型、または厳しいレイテンシー要件を持つワークロードの実行を希望するお客様の場合、AWS ではベアメタルインスタンスを評価することをお勧めしています。

ネストされた仮想化を有効にした新規インスタンスを起動する

新しいインスタンスを起動するときは、インスタンスでハイパーバイザーと仮想マシンを実行するためにネストされた仮想化を有効にできます。

前提条件

Amazon EC2 インスタンスを起動するために必要な IAM 許可が必要です。

Console

インスタンスの起動時にネストされた仮想化を有効にする

1. the section called “インスタンス起動ウィザードを使用した起動” の手順に従い、必要に応じてインスタンスを設定します。
2. サポートされているインスタンスタイプが選択されていることを確認します。
3. [高度な詳細] を開き、[ネストされた仮想化] で [有効] を選択します。
4. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択してください。

AWS CLI

ネストされた仮想化を有効にしたインスタンスを起動する

run-instances コマンドを使用します。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type r8i.4xlarge \
```

```
    --cpu-options "NestedVirtualization=enabled" \
    --key-name my-key-pair
```

PowerShell

ネストされた仮想化を有効にしたインスタンスを起動する

New-EC2Instance コマンドを使用します。

```
New-EC2Instance `
    -ImageId ami-0abcdef1234567890 `
    -InstanceType r8i.4xlarge `
    -CpuOption @{NestedVirtualization='enabled'} `
    -KeyName my-key-pair
```

ネストされた仮想化を使用するように既存のインスタンスを設定する

既存の Amazon EC2 インスタンスでネストされた仮想化を有効にできます。

前提条件

- インスタンスは stopped の状態である必要があります。
- インスタンスタイプがネストされた仮想化をサポートしている必要があります。

Console

既存のインスタンスでネストされた仮想化を有効にする

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンステーブルから変更するインスタンスを選択します。
4. [アクション]、[インスタンスの設定]、[CPU オプションの変更] を選択します。
5. [CPU オプションの変更] ページの [ネストされた仮想化] で、次のオプションのいずれかを選択します。
   - [有効] – インスタンスでネストされた仮想化を有効にします
   - [無効] – インスタンスでネストされた仮想化を無効にします
6. 変更を確認したら、[変更] を選択して新しい CPU オプションを適用します。

AWS CLI

既存のインスタンスでネストされた仮想化を有効にする

インスタンスを停止してから、modify-instance-cpu-options コマンドを使用します。

```
aws ec2 modify-instance-cpu-options \
    --instance-id i-1234567890abcdef0 \
    --core-count 4 \
    --threads-per-core 2 \
    --nested-virtualization enabled
```

PowerShell

既存のインスタンスでネストされた仮想化を有効にする

インスタンスを停止してから、Edit-EC2InstanceCpuOption コマンドを使用します。

```
Edit-EC2InstanceCpuOption `
    -InstanceId i-1234567890abcdef0 `
    -CoreCount 4 `
    -ThreadsPerCore 2 `
    -NestedVirtualization enabled
```

Amazon EC2 の請求および購入オプション

以下のオプションを使用して、Amazon EC2 のコストを最適化できます。

- オンデマンドインスタンス – 起動したインスタンスに対して秒単位でお支払いいただきます。
- Savings Plans – 1〜3 年の期間、1 時間 あたり USD 単位で一定の使用量を契約することにより、Amazon EC2 にかかるコストを削減します。
- リザーブドインスタンス – 1 ~ 3 年の期間、インスタンスタイプとリージョンを含むインスタンス設定を維持する契約により、Amazon EC2 にかかるコストを削減します。
- スポットインスタンス – 未使用の EC2 インスタンスをリクエストすることで、Amazon EC2にかかるコストを大幅に削減できます。
- 専用ホスト – インスタンスの実行のみを目的とした物理ホストに対してお支払いいただきます。ソケット単位、コア単位、または VM 単位で、ご使用中のソフトウェアライセンスを持ち込むことでコストを削減できます。

- ハードウェア専有インスタンス – シングルテナントハードウェアで実行されるインスタンスに対して、時間単位でお支払いいただきます。
- キャパシティー予約 – 特定のアベイラビリティーゾーンの EC2 インスタンスのキャパシティを予約できます。

特定のインスタンス設定を確約できないものの、使用量を確約できる場合はSavings Plans を購入するとオンデマンドインスタンスのコストを削減できます。キャパシティ予約が必要な場合は特定のアベイラビリティーゾーンのリザーブドインスタンスまたはキャパシティ予約を購入します。キャパシティブロックを使用すると GPU インスタンスのクラスターを予約できます。スポットインスタンスはアプリケーションを実行するタイミングに柔軟性がある場合や、アプリケーションを中断できる場合に費用効率の高い選択肢です。専用ホスト または Dedicated Instances (ハードウェア専有インスタンス) を使用すると、サーバーにバインドされた既存のソフトウェアライセンスを使用することにより、コンプライアンス要件に対応しながらコストを削減できます。

詳細についてはAmazon EC2 の料Amazon EC2 マネージドインスタンス 金表を参照してください。

Amazon EC2 のオンデマンドインスタンスの購入

オンデマンドインスタンスでは長期契約は必要なく、料金はコンピューティング性能に対して秒単位で発生します。そのインスタンスライフサイクルを完全に制御でき、いつ起動、停止、休止、開始、再起動、または終了するかを決定できます。

オンデマンドインスタンス を購入するときに要求される長期的なコミットメントはありません。ご利用の オンデマンドインスタンス が `running` 状態になっている秒数 (最低 60 秒) に対してのみお支払いいただきます。オンデマンドインスタンス が実行される秒数あたりの料金は固定で、Amazon EC2 の料金、オンデマンド料金ページに記載されています。

短期間、不規則なワークロードがあり中断できないアプリケーションには、オンデマンドインスタンス の使用をお勧めします。

オンデマンドインスタンスで料金を大幅に削減するには、AWS Savings Plans、スポットインスタンス、または Amazon EC2 のリザーブドインスタンスの概要 を使用してください。

目次

- オンデマンドインスタンスクォータ
  - オンデマンドインスタンス のクォータと使用量のモニタリング
  - クォータ引き上げをリクエストする
- オンデマンドインスタンスの料金を照会する

オンデマンドインスタンスクォータ

各リージョンごとに、AWS アカウント ごとに実行中のオンデマンド インスタンスの数に対するクォータがあります。オンデマンドインスタンスのクォータは、インスタンスタイプに関係なく、実行中のオンデマンドインスタンスで使用している仮想中央演算装置 (vCPU) の数で管理されます。各クォータタイプは、1 つ以上のインスタンスファミリーに対し、最大の vCPU 数を指定しています。

アカウントには、オンデマンドインスタンスの次のクォータが含まれます。状態が保留中、停止中、停止済み、休止済みのインスタンスは、オンデマンドインスタンスのクォータにはカウントされません。キャパシティ予約は、未使用であっても、オンデマンドインスタンスのクォータにカウントされます。

名前	デフォルト	引き上げ可能
オンデマンド DL インスタンスの実行	0	あり
オンデマンド F インスタンスの実行	0	あり
オンデマンド G および VT インスタンスの実行	0	あり
オンデマンドオール HPC インスタンスの実行	0	あり
オンデマンドハイメモリインスタンスの実行	0	あり
オンデマンド Inf インスタンスの実行	0	あり
オンデマンド P インスタンスの実行	0	あり
オンデマンド標準 (A、C、D、H、I、M、R、T、Z) インスタンスの実行	5	あり
オンデマンド Trn インスタンスの実行	0	あり
オンデマンド X インスタンスの実行	0	あり

さまざまなインスタンスファミリー、世代、およびサイズについては、「Amazon EC2 インスタンスタイプガイド」を参照してください。

vCPU の数が自分のアカウントでのクォータを超えていない限り、変化するアプリケーションのニーズに合わせて、任意の組み合わせでインスタンスタイプを起動できます。例えば、256 vCPU の

クォータがあるスタンダードインスタンスでは、32 個の m5.2xlarge インスタンス (32 x 8 vCPU) または 16 個の c5.4xlarge インスタンス (16 x 16 vCPU) を起動できます。詳細については、EC2 オンデマンドインスタンス の制限を参照してください。

タスク

- オンデマンドインスタンス のクォータと使用量のモニタリング
- クォータ引き上げをリクエストする

オンデマンドインスタンス のクォータと使用量のモニタリング

次の方法を使用して、各リージョンのオンデマンド インスタンス クォータを表示および管理できます。

Service Quotas コンソールを使用して現在のクォータを表示するには

1. Service Quotas コンソール (https://console.aws.amazon.com/servicequotas/home/services/ec2/quotas/) を開きます。
2. ナビゲーションバーから、リージョンを選択します。
3. フィルターフィールドに、**On-Demand** と入力します。
4. [適用されたクォータ値] 列には、アカウントの各オンデマンドインスタンスのクォータタイプの vCPU の最大数が表示されます。

AWS Trusted Advisor コンソールを使用して現在のクォータを表示するには

AWS Trusted Advisor コンソールのサービスの制限ページを開きます。

CloudWatch アラームを設定するには

Amazon CloudWatch のメトリクス統合では、クォータに対して EC2 の使用量をモニタリングできます。クォータに近づいたときに警告を発するようにアラームを設定することもできます。詳細については、「Service Quotas」ユーザーガイドのサービスクォータと Amazon CloudWatch アラームを参照してください。

クォータ引き上げをリクエストする

オンデマンドインスタンスの上限は、使用量に基づき Amazon EC2 によって自動的に引き上げられますが、必要であればクォータの引き上げをリクエストすることも可能です。例えば、現在のクォータで許可されているよりも多くのインスタンスを起動する場合は、前のセクション「Amazon EC2

の Service Quotas」に記載されている Service Quotas コンソールで説明したように、 を使用してクォータの増加を要求できます。

オンデマンドインスタンスの料金を照会する

Price List Service API または AWS Price List API を使用して、オンデマンドインスタンス の料金を照会できます。詳細については、AWSユーザーガイドのAWS Billing Price List API の使用を参照してください。

Amazon EC2 のリザーブドインスタンスの概要

> **Important**
>
> リザーブドインスタンスよりも Savings Plans をお勧めします。節約プランはリザーブドインスタンスと同様に、AWS コンピューティングコストを節約し、低価格 (オンデマンド料金から最大 72% オフ) を実現する最も簡単で柔軟な方法です。ただし、Savings Plans はリザーブドインスタンスとは異なります。リザーブドインスタンスでは特定のインスタンス構成にコミットしますが、Savings Plans ではニーズに最も合ったインスタンス構成を柔軟に使用できます。Savings Plans を使用する場合は1 時間 につき USD 単位で一定の使用量を守ることになります。詳細についてはAWS Savings Plans ユーザーガイドをご参照ください。

オンデマンドインスタンスの料金と比較して、リザーブドインスタンスでは Amazon EC2 の料金を大幅に節約することができます。リザーブドインスタンスは物理インスタンスではありませんが、請求の割引はアカウントでのオンデマンドインスタンスの使用に適用されます。請求割引のメリットを得るにはこれらのオンデマンドインスタンスはインスタンスタイプやリージョンなどの特定の属性に一致する必要があります。

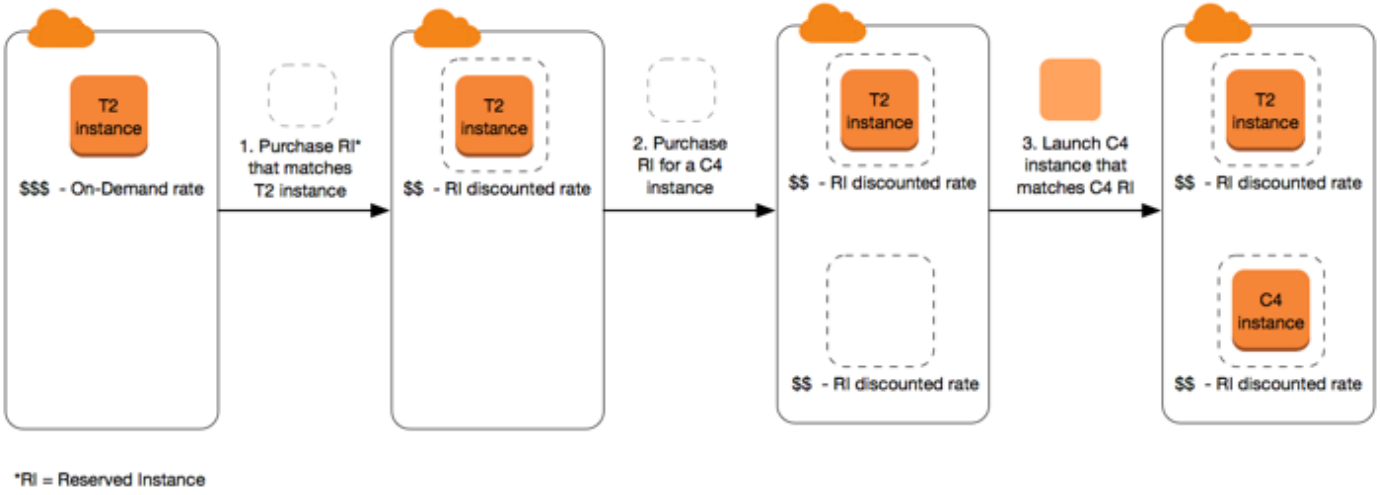
リザーブドインスタンスのトピック

- リザーブドインスタンスのシナリオ例
- リザーブドインスタンス 料金を決定するキー変数
- リージョンおよびゾーン リザーブドインスタンス (スコープ)
- リザーブドインスタンス のタイプ (提供しているクラス)
- リザーブドインスタンスの割引はどのように適用されるか
- お使いの リザーブドインスタンス の使用
- リザーブドインスタンスの課金の仕組み
- Amazon EC2 のリザーブドインスタンスを購入する

- リザーブドインスタンスマーケットプレイスで Amazon EC2 のリザーブドインスタンスを販売する
- リザーブドインスタンス の変更
- コンバーティブルリザーブドインスタンス の交換
- リザーブドインスタンスのクォータ

リザーブドインスタンスのシナリオ例

次の図に、リザーブドインスタンスの購入と使用の基本的なシナリオを示します。



このシナリオでは現在オンデマンドレートを支払っているアカウントの オンデマンドインスタンス (T2) を実行しています。実行しているインスタンスの属性を一致する リザーブドインスタンス を購入すると、料金上の利点が即時適用されます。次に、C4 インスタンスに リザーブドインスタンス を購入します。この リザーブドインスタンス に属性が一致するアカウントで実行しているインスタンスはありません。この最終ステップではC4 リザーブドインスタンス の属性と一致するインスタンスを起動すると、料金上の利点が独自適用されます。

リザーブドインスタンス 料金を決定するキー変数

リザーブドインスタンス 料金は次のキー変数によって決まります。

インスタンスの属性

リザーブドインスタンス にはその料金を決める 4 つのインスタンス属性があります。

- インスタンスタイプ: 例えば、`m4.large`。これはインスタンスファミリー (`m4` など) とインスタンスサイズ (`large` など) で構成されます。

- リージョン: リザーブドインスタンスが購入されているリージョン。
- テナンシー: インスタンスが共有 (デフォルト) または単一のテナンシー (専用) のハードウェアで実行されるかについて。詳細については「Amazon EC2 ハードウェア専有インスタンス」を参照してください。
- プラットフォーム: オペレーティング システム。例えば、Windows や Linux/Unix。詳細については「プラットフォームの選択」を参照してください。

コミットメント期間

1 年あるいは 3 年のコミットメントで リザーブドインスタンス を購入することができます。3 年のコミットメントには大幅な割引が提供されます。

- 1 年: 1 年は 31536000 秒 (365 日) として定義されます。
- 3 年: 3 年は 94608000 秒 (1095 日) として定義されます。

リザーブドインスタンス は自動的に更新されません。有効期限が切れても、引き続き EC2 インスタンスを使用できますが、オンデマンド価格が課金されます。上記の例ではT2 および C4 インスタンスを対象とする リザーブドインスタンス の期限が切れた場合、インスタンスが終了するまでオンデマンドレートの支払いに戻るか、あるいはインスタンスの属性に一致する新しい リザーブドインスタンス を購入します。

> **Important**
>
> リザーブドインスタンス を購入した後で購入をキャンセルすることはできません。ただし、ユーザーのニーズが変更した場合、リザーブドインスタンス を変更、交換、売却できる場合もあります。

支払いオプション

リザーブドインスタンス では次の支払いオプションが用意されています。

- すべて前払い: 期間の開始時に全額が支払われ、使用時間数に関係なく、残りの期間にその他のコストや追加時間課金は生じません。
- 一部前払い: 料金の一部を前払いする必要があり、期間内の残りの時間はリザーブドインスタンスが使用されたどうかにかかわらず、割引された時間料金で請求されます。

- 前払いなし: リザーブドインスタンス が使用されたどうかにかかわらず、期間内のすべての時間は割引時間料金での請求となります。前払い料金は必要ではありません。

> **Note**
>
> 前払いなしの リザーブドインスタンス は予約の全期間について毎月支払いを行う契約義務に基づいています。そのため、前払いなしの リザーブドインスタンス を購入するには請求履歴に問題がないことが必須となります。

一般的にはリザーブドインスタンス の前払い額を高く設定するほど、より多くの費用を節約できます。また、Reserved Instance Marketplace ではサードパーティーの販売者が提供する、より安価で期間の短いリザーブドインスタンス を見つけることもできます。詳細については「リザーブドインスタンスマーケットプレイスで Amazon EC2 のリザーブドインスタンスを販売する」を参照してください。

提供クラス

コンピューティングに変更が必要な場合、提供クラスによって、リザーブドインスタンス を変更または交換することができます。

- スタンダード: 最大の割引を提供しますが、変更のみを行うことができます。スタンダード リザーブドインスタンス は交換できません。
- コンバーティブル: スタンダード リザーブドインスタンス より少ない割引ですが、異なるインスタンス属性を使用する別のコンバーティブル リザーブドインスタンス と交換できます。コンバーティブル リザーブドインスタンス は変更することもできます。

詳細については「リザーブドインスタンス のタイプ (提供しているクラス)」を参照してください。

> **Important**
>
> リザーブドインスタンス を購入した後で購入をキャンセルすることはできません。ただし、ユーザーのニーズが変更した場合、リザーブドインスタンス を変更、交換、売却できる場合もあります。

詳細についてはAmazon EC2 リザーブドインスタンスの料金表ページを参照してください。

リージョンおよびゾーン リザーブドインスタンス (スコープ)

リザーブドインスタンス の購入時に リザーブドインスタンス のスコープを決定します。スコープはリージョンあるいはゾーンのいずれかになります。

- リージョナル: リージョン用に リザーブドインスタンス を購入する場合、これはリージョナル リザーブドインスタンス と呼ばれます。
- ゾーン: 特定のアベイラビリティーゾーン用に リザーブドインスタンス を購入する場合、これはゾーンリザーブドインスタンスと呼ばれます。

範囲は料金には影響しません。リージョンまたはゾーンごとの リザーブドインスタンス に同じ料金を支払います。リザーブドインスタンス の料金の詳細についてはリザーブドインスタンス 料金を決定するキー変数とAmazon EC2 リザーブドインスタンスの料金を参照してください。

リザーブドインスタンスのスコープを指定する方法の詳細については「RI の属性」、特に「アベイラビリティーゾーン」の箇条書きを参照してください。

リージョンとゾーンの リザーブドインスタンス の違い

次のテーブルではリージョン リザーブドインスタンス とゾーン リザーブドインスタンス の主な違いをいくつか示しています。

	リージョン リザーブドインスタンス	ゾーン リザーブドインスタンス
キャパシティーを予約する機能	リージョンの リザーブドインスタンス ではキャパシティーは予約されません。	ゾーンの リザーブドインスタンス では指定されたアベイラビリティーゾーンでキャパシティーが予約されます。
アベイラビリティーゾーンの柔軟性	指定するリージョン内のすべてのアベイラビリティーゾーンにおけるインスタンスの使用に対して、リザーブドインスタンス 割引が適用されます。	アベイラビリティーゾーンの柔軟性なし — リザーブドインスタンス 割引は指定したアベイラビリティー–ゾーン内のみのインスタンスの使用に対して適用されます。

	リージョン リザーブドインスタンス	ゾーン リザーブドインスタンス
インスタンスサイズの柔軟性	インスタンスファミリー内のインスタンスの使用に対して、サイズを問わず、リザーブドインスタンス 割引が適用されます。 Amazon Linux/Unix リザーブドインスタンス のデフォルトテナンシーのみでサポートされます。詳細については「正規化係数によって決定されたインスタンスサイズの柔軟性」を参照してください。	インスタンスサイズの柔軟性なし — リザーブドインスタンス 割引は指定されたインスタンスタイプとサイズにおけるインスタンスの使用に対してのみ適用されます。
購入をキューに入れる	リージョンリザーブドインスタンスの購入をキューに入れることができます。	ゾーンリザーブドインスタンスの購入をキューに入れることはできません。

詳細な説明と例についてはリザーブドインスタンスの割引はどのように適用されるかを参照してください。

リザーブドインスタンス のタイプ (提供しているクラス)

リザーブドインスタンス の提供クラスはスタンダードまたはコンバーチブルのいずれかです。スタンダード リザーブドインスタンス はコンバーチブル リザーブドインスタンス よりも大幅な割引が受けられますが、スタンダード リザーブドインスタンス を交換することはできません。コンバーティブル リザーブドインスタンス は交換できます。スタンダードおよびコンバーティブルのリザーブドインスタンスは変更可能です。

リザーブドインスタンス の設定は期間内の 1 つのインスタンスタイプ、プラットフォーム、スコープ、およびテナンシーで構成されています。コンピューティングに変更が必要な場合はリザーブドインスタンス を変更または交換できる可能性があります。

スタンダードとコンバーティブル リザーブドインスタンス の違い

以下に、スタンダードとコンバーティブル リザーブドインスタンス の違いを示します。

	スタンダード リザーブドインスタンス	Convertible Reserved Instance
リザーブドインスタンス の変更	一部の属性は変更できます。詳細については「リザーブドインスタンス の変更」を参照してください。	一部の属性は変更できます。詳細については「リザーブドインスタンス の変更」を参照してください。
リザーブドインスタンスの交換	交換できません。	期間内で、インスタンスファミリー、インスタンスタイプ、プラットフォーム、スコープやテナンシーなどの新しい属性の別の コンバーティブルリザーブドインスタンスに交換することができます。詳細については「コンバーティブルリザーブドインスタンス の交換」を参照してください。
Reserved Instance Marketplace での販売	Reserved Instance Marketplace で販売可能です。	Reserved Instance Marketplace では販売できません。
リザーブドインスタンスの Marketplace での購入	Reserved Instance Marketplace で購入可能です。	Reserved Instance Marketplace では購入できません。

リザーブドインスタンスの割引はどのように適用されるか

リザーブドインスタンスは物理インスタンスではありませんが、請求の割引はアカウントでのオンデマンドインスタンスの実行に適用されます。請求割引のメリットを得るにはオンデマンドインスタンスはリザーブドインスタンスの特定の仕様に一致する必要があります。

リザーブドインスタンスを購入し、リザーブドインスタンスの仕様に一致するオンデマンドインスタンスが既に実行されている場合、請求割引は直ちに自動的に適用されます。インスタンスを再起動す

る必要はありません。使用可能な実行中のオンデマンドインスタンスが存在しない場合、リザーブドインスタンスと同じ仕様でオンデマンドインスタンスを起動します。詳細については「お使いの リザーブドインスタンス の使用」を参照してください。

リザーブドインスタンスの提供クラス (スタンダードまたはコンバーティブル) は請求割引の適用には影響しません。

トピック

- ゾーン リザーブドインスタンス がどのように適用されるか
- リージョンリザーブドインスタンスがどのように適用されるか
- インスタンスサイズの柔軟性
- リザーブドインスタンス の適用例

ゾーン リザーブドインスタンス がどのように適用されるか

特定のアベイラビリティーゾーンで容量を予約するために購入されるリザーブドインスタンスはゾーンリザーブドインスタンスと呼ばれます。

- リザーブドインスタンスの割引はそのアベイラビリティーゾーンに一致したインスタンスの使用に適用されます。
- 実行中のインスタンスの属性 (テナンシー、プラットフォーム、アベイラビリティーゾーン、インスタンスタイプ、およびインスタンスサイズ) はリザーブドインスタンス の属性と一致する必要があります。

例えば、us-east-1a のアベイラビリティーゾーンで、デフォルトテナンシーの `c4.xlarge` Linux/Unix スタンダード リザーブドインスタンス を 2 つ購入すると、us-east-1a のアベイラビリティーゾーンで実行している 2 つまでのデフォルトテナンシーの `c4.xlarge` Linux/Unix インスタンスでリザーブドインスタンス 割引を利用できます。

リージョンリザーブドインスタンスがどのように適用されるか

リージョン用に購入されるリザーブドインスタンスはリージョンリザーブドインスタンスと呼ばれ、アベイラビリティーゾーンとインスタンスサイズの柔軟性を提供します。

- このリージョン内のすべてのアベイラビリティーゾーンにおけるインスタンスの使用に対して、リザーブドインスタンス 割引が適用されます。

- リザーブドインスタンスの割引はサイズに関係なく、インスタンスファミリー内のインスタンスの使用に適用されます。これは「インスタンスサイズの柔軟性」と呼ばれます。

インスタンスサイズの柔軟性

インスタンスサイズの柔軟性により、ファミリーが同一のインスタンスを使用した場合、リザーブドインスタンス割引が適用されます。リザーブドインスタンスはインスタンスファミリー内の最もサイズの小さいインスタンスからサイズの大きいインスタンスへ、正規化係数に基づいて適用されます。リザーブドインスタンス割引の適用例については「シナリオ 2: 正規化係数を使用した 1 つのアカウントのリザーブドインスタンス」を参照してください。

制限事項

- サポート: インスタンスサイズの柔軟性はリージョナルリザーブドインスタンスでのみサポートされています。
- 未サポート: 次のリザーブドインスタンスではインスタンスサイズの柔軟性はサポートされていません。
  - 特定のアベイラビリティーゾーン (ゾーンリザーブドインスタンス) 用に購入されたリザーブドインスタンス
  - G4ad、G4dn、G5、G5g、G6、G6e、G6f、Gr6、Gr6f、hpc7a、P5、Inf1、Inf2、u7i-6tb、および u7i-8tb インスタンス用のリザーブドインスタンス
  - リザーブドインスタンス for Windows Server、Windows Server with SQL Standard、Windows Server with SQL Server Enterprise、Windows Server with SQL Server Web、RHEL、SUSE Linux Enterprise Server
  - 専有テナントを使用する リザーブドインスタンス

正規化係数によって決定されたインスタンスサイズの柔軟性

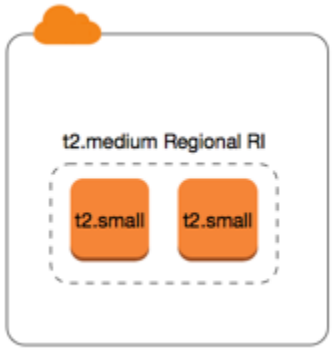
インスタンスサイズの柔軟性はインスタンスサイズの正規化係数によって決定されます。割引はリージョン内のアベイラビリティーゾーンで、予約したインスタンスサイズによって、同じインスタンスファミリーで実行中のインスタンスに完全または部分的に適用されます。一致する必要がある属性はインスタンスファミリー、テナンシー、プラットフォームのみです。

次の表はインスタンスファミリー内のさまざまなサイズおよび対応する正規化係数の一覧です。このスケールを使用して、リザーブドインスタンス の割引料金をインスタンスファミリーの正規化された使用に適用します。

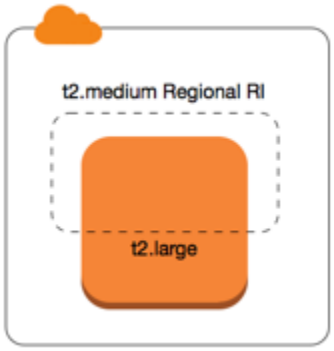
インスタンスサイズ	正規化係数
nano	0.25
micro	0.5
small	1
medium	2
large	4
xlarge	8
2xlarge	16
3xlarge	24
4xlarge	32
6xlarge	48
8xlarge	64
9xlarge	72
10xlarge	80
12xlarge	96
16xlarge	128
18xlarge	144
24xlarge	192
32xlarge	256
48xlarge	384
56xlarge	448

インスタンスサイズ	正規化係数
96xlarge	768
112xlarge	896

例えば、t2.medium インスタンスには 2 の正規化係数があります。US East (N. Virginia) で t2.medium デフォルトテナンシー Amazon Linux/Unix リザーブドインスタンス を購入し、このリージョンのアカウントで 2 つの t2.small インスタンスを実行している場合、料金上の利点はどちらのインスタンスにも完全に適用されます。



またはUS East (N. Virginia) リージョンのアカウントで実行している 1 つの t2.large インスタンスがある場合、料金上の利点はこのインスタンスの使用の 50% に適用されます。



正規化係数はリザーブドインスタンス の変更時にも適用されます。詳細については「リザーブドインスタンス の変更」を参照してください。

ベアメタルインスタンスの正規化係数

また、インスタンスサイズの柔軟性はインスタンスファミリー内のベアメタルインスタンスにも適用されます。ベアメタルインスタンスで共有するテナントを使用したリージョンの Amazon Linux/Unix

リザーブドインスタンス がある場合、同じインスタンスファミリー内で リザーブドインスタンス 割引の特典を受けることができます。逆の場合も同様です。同じファミリー内のインスタンスでベアメタルインスタンスとしてテナントを共有している、リージョン Amazon Linux/Unix リザーブドインスタンス がある場合、ベアメタルインスタンスで リザーブドインスタンス 割引の特典を受けることができます。

metal インスタンスサイズには単一の正規化係数がありません。ベアメタルインスタンスの正規化係数は同じインスタンスファミリー内の同等な仮想インスタンスサイズと同じです。例えば、1 つの i3.metal インスタンスには i3.16xlarge インスタンスと同じ正規化係数があります。

インスタンスサイズ	正規化係数
a1.metal	32
m5zn.metal | x2iezn.metal z1d.metal	96
c6g.metal | c6gd.metal | i3.metal | m6g.metal | m6gd.metal | r6g.metal | r6gd.metal | x2gd.metal	128
c5n.metal	144
c5.metal | c5d.metal | i3en.metal | m5.metal | m5d.metal | m5dn.metal | m5n.metal | r5.metal | r5b.metal | r5d.metal | r5dn.metal | r5n.metal	192
c6i.metal | c6id.metal | m6i.metal | m6id.metal | r6d.metal | r6id.metal	256
u-18tb1.metal | u-24tb1.metal	448
u-6tb1.metal | u-9tb1.metal | u-12tb1.metal	896

例えば、1 つの i3.metal インスタンスには 128 の正規化係数があります。US East (N. Virginia) で i3.metal デフォルトテナンシー Amazon Linux/Unix リザーブドインスタンス を購入した場合、料金上のメリットは次のようになります。

- そのリージョン内のアカウントで実行している 1 つの i3.16xlarge がある場合、料金上のメリットは i3.16xlarge インスタンス全体に適用されます (i3.16xlarge 正規化係数 = 128)。

- あるいはそのリージョン内のアカウントで実行している 2 つの i3.8xlarge がある場合、料金上のメリットは両方の i3.8xlarge インスタンス全体に適用されます (i3.8xlarge 正規化係数 = 64)。
- あるいはそのリージョン内のアカウントで実行している 4 つの i3.4xlarge がある場合、料金上のメリットは 4 つの i3.4xlarge インスタンス全体に適用されます (i3.4xlarge 正規化係数 = 32)。

逆の場合も同様です。例えば、US East (N. Virginia) で 2 つの i3.8xlarge デフォルトテナンシー Amazon Linux/Unix リザーブドインスタンス を購入し、そのリージョンで実行している 1 つの i3.metal インスタンスがある場合、料金上のメリットは i3.metal インスタンス全体に適用されます。

リザーブドインスタンス の適用例

リザーブドインスタンス の適用方法を以下のシナリオで示します。

- シナリオ 1: 単一アカウントの リザーブドインスタンス
- シナリオ 2: 正規化係数を使用した 1 つのアカウントのリザーブドインスタンス
- シナリオ 3: 連結アカウントのリージョンリザーブドインスタンス
- シナリオ 4: 連結アカウントのゾーンリザーブドインスタンス

シナリオ 1: 単一アカウントの リザーブドインスタンス

アカウント A で以下の オンデマンドインスタンス を実行しているとします。

- us-east-1a アベイラビリティーゾーンで 4 つのデフォルトテナンシーの m3.large Linux インスタンス
- us-east-1b アベイラビリティーゾーンで 2 つのデフォルトテナンシーの m4.xlarge Amazon Linux インスタンス
- us-east-1c アベイラビリティーゾーンで 1 つのデフォルトテナンシーの c4.xlarge Amazon Linux インスタンス

アカウント A で以下の リザーブドインスタンス を購入するとします。

- us-east-1a アベイラビリティーゾーンで 4 つのデフォルトテナンシーの m3.large Linux リザーブドインスタンス (キャパシティーの予約あり)

- us-east-1 リージョンで 4 つのデフォルトテナンシーの m4.large Amazon Linux リザーブドインスタンス
- us-east-1 リージョンで 1 つのデフォルトテナンシーの c4.large Amazon Linux リザーブドインスタンス

リザーブドインスタンス の利点は以下のように適用されます。

- 4 つの m3.large ゾーン リザーブドインスタンス の割引とキャパシティーの予約は属性 (インスタンスサイズ、リージョン、プラットフォーム、テナンシー) が一致する 4 つの m3.large インスタンスによって使用されます。
- m4.large リージョン リザーブドインスタンス はデフォルトテナンシーの Amazon Linux リザーブドインスタンス であるため、アベイラビリティーゾーンおよびインスタンスサイズの柔軟性を提供します。

  1 つの m4.large は4 つの正規化された単位/時間に相当します。

  4 つの m4.large リージョン リザーブドインスタンス を購入したので、合計で 16 の正規化された単位/時間 (4x4) に相当します。アカウント A で実行している 2 つの m4.xlarge インスタンスは16 の正規化された単位/時間 (2x8) に相当します。この場合、4 つの m4.large リージョナルリザーブドインスタンスによって、2 つの m4.xlarge インスタンスの使用に対する全面的な請求のメリットが提供されます。
- us-east-1 の c4.large リージョン リザーブドインスタンス はデフォルトテナンシーのリージョンの Amazon Linux リザーブドインスタンス であるため、アベイラビリティーゾーンおよびインスタンスサイズの柔軟性を提供し、c4.xlarge インスタンスに適用されます。c4.large インスタンスは 4 つの正規化された単位/時間に相当し、c4.xlarge インスタンスは 8 つの正規化された単位/時間に相当します。

  この場合、c4.large リージョン リザーブドインスタンス はc4.xlarge の使用の一部に対してメリットを提供します。これはこの c4.large リザーブドインスタンス は 4 つの正規化された単位/時間に相当しますが、c4.xlarge インスタンスは 8 つの正規化された単位/時間を必要とするためです。したがって、c4.large リザーブドインスタンス の請求割引はc4.xlarge の使用の 50% に適用されます。c4.xlarge の残りの使用はオンデマンド価格で課金されます。

シナリオ 2: 正規化係数を使用した 1 つのアカウントのリザーブドインスタンス

アカウント A で以下の オンデマンドインスタンス を実行しているとします。

- us-east-1a アベイラビリティーゾーンで 2 つのデフォルトテナンシーの m3.xlarge Amazon Linux インスタンス
- us-east-1b アベイラビリティーゾーンで 2 つのデフォルトテナンシーの m3.large Amazon Linux インスタンス

アカウント A で以下のリザーブドインスタンスを購入するとします。

- us-east-1 リージョンで 1 つのデフォルトテナンシーの m3.2xlarge Amazon Linux リザーブドインスタンス

リザーブドインスタンス の利点は以下のように適用されます。

- us-east-1 の m3.2xlarge リージョンリザーブドインスタンスはデフォルトテナンシーのリージョンの Amazon Linux リザーブドインスタンスであるため、アベイラビリティーゾーンおよびインスタンスサイズの柔軟性を提供します。これは最初に m3.large インスタンスに適用され、次に m3.xlarge インスタンスに適用されます (インスタンスファミリー内の最もサイズの小さいインスタンスからサイズの大きいインスタンスへ、正規化係数に基づいて適用されるため)。

  1 つの m3.large インスタンスは4 つの正規化された単位/時間に相当します。

  1 つの m3.xlarge インスタンスは8 つの正規化された単位/時間に相当します。

  1 つの m3.2xlarge インスタンスは16 の正規化された単位/時間に相当します。

  この利点は次のように適用されます。

  m3.2xlarge リージョナルリザーブドインスタンスは2 つの m3.large の使用に最大の利点をもたらします。これらのインスタンスは8 つの正規化された単位/時間に相当するためです。これにより、m3.xlarge インスタンスに適用される 8 つの正規化された単位/時間が残ります。

  残りの 8 つの正規化された単位/時間により、m3.2xlarge リージョナルリザーブドインスタンスは1 つの m3.xlarge の使用に最大の利点をもたらします。各 m3.xlarge インスタンスは8 つの正規化された単位/時間に相当するためです。m3.xlarge の残りの使用はオンデマンド価格で課金されます。

シナリオ 3: 連結アカウントのリージョンリザーブドインスタンス

リザーブドインスタンス は最初に購入アカウント内の使用に適用され、次に組織内の他のアカウントの該当する使用に適用されます。詳細については「リザーブドインスタンス および一括請求 (コンソリデーティッドビリング)」を参照してください。サイズの柔軟性を提供するリージョン リザーブドインスタンス の場合、インスタンスファミリー内のインスタンスサイズに関係なく、利点がインスタンスに適用されます。

アカウント A (購入しているアカウント) で以下の オンデマンドインスタンス を実行しているとします。

- us-east-1a アベイラビリティーゾーンで 2 つのデフォルトテナンシーの `m4.xlarge` Linux インスタンス
- us-east-1b アベイラビリティーゾーンで 1 つのデフォルトテナンシーの `m4.2xlarge` Linux インスタンス
- us-east-1a アベイラビリティーゾーンで 2 つのデフォルトテナンシーの `c4.xlarge` Linux インスタンス
- us-east-1b アベイラビリティーゾーンで 1 つのデフォルトテナンシーの `c4.2xlarge` Linux インスタンス

別のお客様はアカウント B— (連結アカウント) で以下の オンデマンドインスタンス を実行しています。

- us-east-1a アベイラビリティーゾーンで 2 つのデフォルトテナンシーの `m4.xlarge` Linux インスタンス

アカウント A で以下のリージョン リザーブドインスタンス を購入するとします。

- us-east-1 リージョンで 4 つのデフォルトテナンシーの `m4.xlarge` Linux リザーブドインスタンス
- us-east-1 リージョンで 2 つのデフォルトテナンシーの `c4.xlarge` Linux リザーブドインスタンス

リージョンリザーブドインスタンスの利点は以下のように適用されます。

- 4 つの `m4.xlarge` リザーブドインスタンス の割引はアカウント A の 2 つの `m4.xlarge` インスタンスと 1 つの `m4.2xlarge` インスタンスによって使用されます。3 つのインスタンスすべてに

おいて、属性 (インスタンスファミリー、リージョン、プラットフォーム、テナンシー) が一致しています。アカウント B (リンクされたアカウント) に リザーブドインスタンス にも一致する 2 つの m4.xlarge がある場合でも、この割引は購入したアカウント (アカウント A) 内のインスタンスにまず適用されます。この リザーブドインスタンス はリージョン リザーブドインスタンス であるため、キャパシティの予約はありません。

- c4.xlarge リザーブドインスタンス インスタンスよりもインスタンスサイズが小さいため、2 つの c4.xlarge リザーブドインスタンスの割引は2 つの c4.2xlarge インスタンスに適用されます。この リザーブドインスタンス はリージョン リザーブドインスタンス であるため、キャパシティの予約はありません。

シナリオ 4: 連結アカウントのゾーンリザーブドインスタンス

通常、アカウントで所有されている リザーブドインスタンス が、そのアカウントでの使用に最初に適用されます。ただし、組織の他のアカウントに特定のアベイラビリティーゾーン (ゾーン リザーブドインスタンス) の未使用の リザーブドインスタンス がある場合はこれらがアカウントで所有されているリージョン リザーブドインスタンス より先に適用されます。これはリザーブドインスタンスの使用率を最大限に高めて請求額を下げるための処置です。請求の目的では組織内のすべてのアカウントが 1 つのアカウントとして扱われます。次の例が、この説明に役立つ場合があります。

アカウント A (購入しているアカウント) で以下の オンデマンドインスタンス を実行しているとします。

- us-east-1a アベイラビリティーゾーンで 1 つのデフォルトテナンシーの m4.xlarge Linux インスタンス

お客様は別の連結アカウント B で以下の オンデマンドインスタンス を実行しています。

- us-east-1b アベイラビリティーゾーンで 1 つのデフォルトテナンシーの m4.xlarge Linux インスタンス

アカウント A で以下のリージョン リザーブドインスタンス を購入するとします。

- us-east-1 リージョンで 1 つのデフォルトテナンシーの m4.xlarge Linux リザーブドインスタンス

ユーザーが連結アカウント C で以下のゾーン リザーブドインスタンス も購入するとします。

- us-east-1a アベイラビリティーゾーンで 1 つのデフォルトテナンシーの m4.xlarge Linux リザーブドインスタンス

リザーブドインスタンス の利点は以下のように適用されます。

- アカウント C で所有されている m4.xlarge ゾーン リザーブドインスタンス の割引はアカウント A の m4.xlarge の使用に適用されます。
- アカウント A で所有されている m4.xlarge リージョン リザーブドインスタンス の割引はアカウント B の m4.xlarge の使用に適用されます。
- アカウント A で所有されているリージョン リザーブドインスタンス は最初にアカウント A での使用に適用されます。アカウント C で所有されているゾーン リザーブドインスタンス は使用されず、アカウント B での使用はオンデマンド価格で課金されます。

詳細については「AWS Cost and Usage Report」の「Understanding your reservations」を参照してください。

> **Note**
>
> ゾーンリザーブドインスタンスは所有アカウントにのみ容量を予約し、他の AWS アカウント と共有することはできません。他の AWS アカウント と容量を共有する必要がある場合はEC2 オンデマンドキャパシティ予約を使用してコンピューティングキャパシティを予約 を使用してください。

お使いの リザーブドインスタンス の使用

リザーブドインスタンス は仕様の一致する実行中の オンデマンドインスタンス に自動的に適用されます。リザーブドインスタンス の仕様と一致する実行中の オンデマンドインスタンス が存在しない場合、必要な仕様が搭載されるインスタンスを起動するまで、リザーブドインスタンス は未使用となります。

リザーブドインスタンスの料金上の利点を利用するためにオンデマンドインスタンスを起動する場合はオンデマンドインスタンスの設定時に以下の情報を指定してください。

プラットフォーム

リザーブドインスタンス のプラットフォーム (製品の説明) と一致するAmazon マシンイメージ (AMI) を指定する必要があります。例えば、Linux/UNIX をリザーブドインスタンスに指定する場合、Amazon Linux AMI または Ubuntu AMI からインスタンスを起動できます。

インスタンスタイプ

ゾーンリザーブドインスタンスを購入した場合はリザーブドインスタンスと同じインスタンスタイプを指定する必要があります (例: t3.large)。詳細については「ゾーン リザーブドインスタンス がどのように適用されるか」を参照してください。

リージョンリザーブドインスタンスを購入した場合はリザーブドインスタンスのインスタンスタイプと同じインスタンスファミリーからインスタンスタイプを指定する必要があります。例えば、t3.xlarge を指定した場合、リザーブドインスタンスの場合はT3 ファミリーからインスタンスを起動する必要がありますが、任意のサイズ (例: t3.medium) を指定できます。詳細については「リージョンリザーブドインスタンスがどのように適用されるか」を参照してください。

アベイラビリティーゾーン

特定のアベイラビリティーゾーンにゾーンごとの リザーブドインスタンス を購入する場合、同じアベイラビリティーゾーンでインスタンスを起動する必要があります。

リージョン リザーブドインスタンスを購入した場合、リザーブドインスタンスに指定したリージョンの任意のアベイラビリティーゾーンにインスタンスを起動することができます。

テナンシー

インスタンスのテナンシー (dedicated や shared) はリザーブドインスタンス のテナンシーが一致する必要があります。詳細については「Amazon EC2 ハードウェア専有インスタンス」を参照してください。

実行しているオンデマンドインスタンスにどのようにリザーブドインスタンスが適用されるかについての例はリザーブドインスタンスの割引はどのように適用されるかを参照してください。詳細についてはAmazon EC2 リザーブドインスタンスが想定通りに AWS 請求書に反映されないのはなぜですか？を参照してください。

リザーブドインスタンス割引を使用するオンデマンドインスタンスを起動するにはさまざまな方法を使用できます。様々な起動方法の詳細についてはAmazon EC2 インスタンスの起動を参照してください。Amazon EC2 Auto Scaling を使用してインスタンスを起動できます。詳細についてはAmazon EC2 Auto Scaling ユーザーガイドを参照してください。

リザーブドインスタンスの課金の仕組み

すべての リザーブドインスタンス の料金はオンデマンドインスタンスの料金から割引された額になります。リザーブドインスタンス では実際の使用に関係なく、全期間の料金をお支払いいただきます。リザーブドインスタンス に特定された支払いオプションによって、リザーブドインスタンス の支払い方法を前払い、一部前払い、月ごとから選択できます。

リザーブドインスタンス 期間が終了すると、EC2 インスタンスの使用についてオンデマンド価格が課金されます。リザーブドインスタンス の購入を最大 3 年先までキューに入れることができます。これにより、サービスを切れ目なく利用できます。詳細については「購入をキューに入れる」を参照してください。

AWS 無料利用枠 は新しい AWS アカウント でご利用いただけます。AWS 無料利用枠 を使用して Amazon EC2 インスタンスを実行している場合、購入したリザーブドインスタンスは標準の料金で課金されます。詳細については、「AWS 無料利用枠」を参照してください。

内容

- 使用料の請求
- 請求の表示
- リザーブドインスタンス および一括請求 (コンソリデーティッドビリング)
- リザーブドインスタンス 割引料金範囲

使用料の請求

リザーブドインスタンス は選択した期間内の 1 時間ごとに請求されます。インスタンスが実行中であるかどうかは関係しません。各時間は標準の 24 時間の時計の正時 (毎時ゼロ分ゼロ秒) に開始します。例えば、1:00:00 ~ 1:59:59 が 1 時間です。インスタンスステータスの詳細についてはAmazon EC2 インスタンスの状態変更を参照してください。

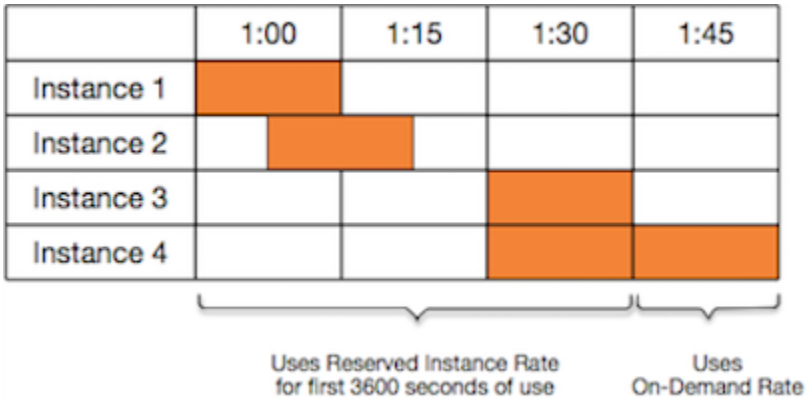
リザーブドインスタンス の料金上の特典は秒単位課金で実行中のインスタンスに適用されます。秒単位の請求はオープンソースの Linux ディストリビューション (Amazon Linux、Ubuntu など) を使用するインスタンスで使用できます。時間単位の請求はLinux の商用ディストリビューション (Red Hat Enterprise Linux、SUSE Linux Enterprise Server など) で使用できます。

リザーブドインスタンス の料金上の利点は1 時間当たり最大 3600 秒 (1 時間) のインスタンス使用に適用できます。複数のインスタンスを同時に実行できますが、リザーブドインスタンス 割引の特典を受けることができるのは 1 時間あたり合計 3600 秒までです。インスタンスの使用が 1 時間あたり 3600 秒を超えると、オンデマンドレートで課金されます。

例えば、1 つの m4.xlarge リザーブドインスタンス を購入し、4 つの m4.xlarge インスタンスを 1 時間同時に実行する場合、1 つのインスタンスには リザーブドインスタンス の 1 時間分の使用料が、他の 3 つのインスタンスにはオンデマンドの 3 時間分の使用料が課金されます。

一方、1 つの m4.xlarge リザーブドインスタンス を購入し、4 つの m4.xlarge インスタンスを同じ 1 時間内にそれぞれ 15 分 (900 秒) ずつ実行した場合、インスタンスの合計実行時間は 1 時間となり、リザーブドインスタンス の使用料が 1 時間分課金されるだけで、オンデマンドの使用料は課金されません。

複数の対象インスタンスが同時に実行されている場合、リザーブドインスタンスの課金特典は1 時間に最大 3600 秒まで、すべてのインスタンスに同時に適用され、その後、オンデマンド料金が適用されます。



Billing and Cost Management コンソール上の [Cost Explorer] はオンデマンドインスタンス の実行に対する節約を分析することができます。リザーブドインスタンス についてのよくある質問には表示価格の計算例があります。

AWS アカウントを解約すると、リソースのオンデマンド課金は停止します。ただし、アカウントにリザーブドインスタンス がある場合にはその期限が切れるまで続けて課金されます。

請求の表示

AWS Billing and Cost Management コンソールで、アカウントへの請求および料金を確認できます。

- [ダッシュボード] にはアカウント利用料の概要が表示されます。
- [請求書] ページの [明細] で、[Elastic Compute Cloud] セクションと リザーブドインスタンス の請求情報を取得するリージョンを展開します。

請求額をオンラインで表示することも、CSV ファイルとしてダウンロードすることもできます。

また、AWS のコストと使用状況レポートを使用して、リザーブドインスタンスの使用を追跡することも可能です。詳細についてはの Understanding your reservations.を参照してください。

リザーブドインスタンス および一括請求 (コンソリデーティッドビリング)

購入アカウントが、1 つの一括請求の支払いアカウントに請求される一連のアカウントの一部である場合、リザーブドインスタンス の料金面でのメリットを広範囲に利用できます。すべてのメンバーアカウントのインスタンス使用量が月次で支払人アカウントに集約されます。さまざまな役割を持つチームやグループがある企業にとっては特に便利です。したがって、請求書の計算には通常の リザーブドインスタンス のロジックが適用されます。詳細についてはAWS Organizations の一括請求を参照してください。

リザーブドインスタンス を購入したアカウントを解約した場合、そのリザーブドインスタンス が期限切れになるまで、支払いアカウントに対する請求が継続されます。アカウントは解約から 90 日後に完全に削除され、メンバーアカウントにはリザーブドインスタンスによる割引料金が適用されなくなります。

> **Note**
>
> ゾーンリザーブドインスタンスは所有アカウントにのみ容量を予約し、他の AWS アカウント と共有することはできません。他の AWS アカウント と容量を共有する必要がある場合はEC2 オンデマンドキャパシティ予約を使用してコンピューティングキャパシティを予約 を使用してください。

リザーブドインスタンス 割引料金範囲

割引料金範囲が適用されると、そのアカウントは以降、その範囲レベル内で行われる リザーブドインスタンス 購入の前払い料金およびインスタンス使用料に対して、自動的に割引を受けます。割引を受けるためにはリージョンの リザーブドインスタンス の表示価格が 500,000 USD 以上である必要があります。

以下のルールが適用されます。

- 料金範囲およびそれに関連する割引はAmazon EC2 スタンダード リザーブドインスタンス の購入にのみ適用されます。
- 料金範囲はSQL Server Standard、SQL Server Web、および SQL Server Enterprise を使用する Windows 用の リザーブドインスタンス には適用されません。
- 料金階層は、SQL Server を使用する Linux のリザーブドインスタンスには適用されません。
- 料金範囲の割引はAWS での購入にのみ適用されます。これはサードパーティーの リザーブドインスタンス の購入には適用されません。
- 料金範囲割引は現在、コンバーティブルリザーブドインスタンス の購入には適用されません。

トピック

- リザーブドインスタンス 料金割引の計算
- 割引範囲での購入
- 購入料金範囲
- 料金範囲の一括請求

リザーブドインスタンス 料金割引の計算

リージョンのすべての リザーブドインスタンス の合計表示価格を計算することによって、アカウントの料金範囲を決定できます。各予約の時間当たりの定期料金を期間の合計時間数に掛けて、購入時の割引されていない前払い料金 (固定料金とも呼ばれる) を加算します。表示価格は割引前料金 (一般料金) に基づいているため、従量制割引の適用を受けた場合または リザーブドインスタンス を購入した後の値下げ分は反映されません。

```
List value = fixed price + (undiscounted recurring hourly price * hours in term)
```

例えば、1 年間の一部前払い `t2.small` リザーブドインスタンス の場合、前払い料金は 60.00 ドル、時間料金は 0.007 ドルと仮定します。これにより、表示価格は 121.32 ドルとなります。

```
121.32 = 60.00 + (0.007 * 8760)
```

Amazon EC2 コンソールを使用して リザーブドインスタンス の固定料金を表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Reserved Instances] を選択してください。

3. [前払い料金] 列を表示するには右上にある設定
   ( )
   をクリックし、[前払い料金] をオンにして [確認] をクリックしてください。

リザーブドインスタンス の固定料金をコマンドラインを使用して表示するには

- describe-reserved-instances (AWS CLI)
- Get-EC2ReservedInstance (AWS Tools for Windows PowerShell)

割引範囲での購入

リザーブドインスタンス を購入すると、割引料金範囲に該当する部分のリザーブドインスタンスに対応する割引があれば、Amazon EC2 によって自動的に適用されます。特に何かを行う必要はなく、どの Amazon EC2 ツールを使用しても リザーブドインスタンス を購入できます。詳細については「Amazon EC2 のリザーブドインスタンスを購入する」を参照してください。

リージョンでのアクティブな リザーブドインスタンス の表示価格が割引料金範囲に該当した場合、そのリージョンでの リザーブドインスタンス は以降、すべての購入が割引料金で課金されます。リージョンの リザーブドインスタンス の 1 回の購入でしきい値を超える場合はそのご購入分のうち、割引範囲のしきい値を超える部分が割引になります。購入プロセス中に作成された一時的な リザーブドインスタンス ID の詳細については購入料金範囲を参照してください。

表示価格が割引範囲の金額を下回った場合は (一部の リザーブドインスタンス の有効期限が切れた場合など)、以降、そのリージョンで購入される リザーブドインスタンス には割引が適用されません。ただし、もともと割引料金範囲で購入された リザーブドインスタンス に対しては引き続き割引が適用されます。

リザーブドインスタンス を購入すると、次の 4 つのいずれかの状況になります。

- 割引なし — 1 つのリージョンでのリザーブドインスタンスの購入が、まだ割引しきい値より下である。
- 一部割引 — 1 つのリージョンでのリザーブドインスタンスの購入により、最初の割引範囲のしきい値を超える。割引なしが 1 つ以上の予約に適用され、割引料金が残りの予約に適用されます。
- 完全割引 — リージョン内の購入全体が 1 つの割引範囲に完全に含まれ、適切に割引されます。
- 2 つの割引料金 — 1 つのリージョンでのリザーブドインスタンスの購入が、下の割引範囲から上の割引範囲まで及ぶ。2 つの異なる料金が課金されます。1 つ以上の予約により低い割引率が適用され、残りの予約により高い割引率が適用されます。

購入料金範囲

割引料金範囲に到達した場合、その購入に対して複数のエントリが表示されます。通常の料金が課金される部分と、割引料金が適用されて課金される部分です。

リザーブドインスタンス サービスによって複数の リザーブドインスタンス ID が生成されます。これは割引が適用されない範囲と割引範囲、または複数の割引範囲に購入がまたがるためです。範囲内の予約のセットにはそれぞれ ID があります。この結果、購入 CLI コマンドまたは API アクションによって返される ID は新しい リザーブドインスタンス の実際の ID とは異なるものになります。

料金範囲の一括請求

一括請求アカウントはリージョン内のメンバーアカウントの表示価格を集計します。一括請求アカウントのすべてのアクティブな リザーブドインスタンス の表示価格が割引料金範囲に達すると、以降 (その一括請求アカウントの表示価格が割引価格範囲のしきい値を超えている限り)、一括請求アカウント内のアカウントで購入された リザーブドインスタンス には割引が適用されます。詳細については「リザーブドインスタンス および一括請求 (コンソリデーティッドビリング)」を参照してください。

Amazon EC2 のリザーブドインスタンスを購入する

Amazon EC2 のリザーブドインスタンスを購入するにはAmazon EC2 コンソール、コマンドラインツール、SDK を使用して、AWS とサードパーティー販売者からのリザーブドインスタンス製品を検索できます。その際、希望するものと完全に一致するものが見つかるまで、検索パラメータを調整してください。

購入する リザーブドインスタンス を検索する際、提供タイプの費用の見積もりが表示されます。購入手続きに進むと、AWS は自動的に購入価格に上限価格を指定します。リザーブドインスタンス の合計コストが、指定した金額を超えることはありません。

何らかの理由により料金が上がったり変更された場合、購入は完了しません。サードパーティー販売者のリザーブドインスタンスを Amazon EC2 リザーブドインスタンスマーケットプレイスから購入する際に、自分が選択したものと類似しているが前払い料金がより安い製品が存在する場合、AWS はその製品をその安い前払い料金で販売します。

購入を承認する前に、購入を検討している リザーブドインスタンス の詳細を点検して、すべてのパラメータが正しいことを確認してください。リザーブドインスタンス を購入した後は (販売者が Reserved Instance Marketplace のサードパーティーであっても、AWS であったとしても) その購入をキャンセルすることはできません。将来の日付での購入をキューに入れることができ、スケジュールされた時間の前であればキューに入れた購入をキャンセルすることができます。

購入後のリザーブドインスタンスを修正するにはユーザーに、アベイラビリティーゾーンを記述する機能などの適切な許可が付与されていることを確認します。詳細については「the section called “リザーブドインスタンス の操作”」(API) または「the section called “リザーブドインスタンス の操作”」(コンソール) を参照してください。

タスク

- プラットフォームの選択
- 購入をキューに入れる
- スタンダード リザーブドインスタンス の購入
- コンバーティブルリザーブドインスタンス の購入
- Reserved Instance Marketplace からの購入
- キューに入れた購入のキャンセル
- リザーブドインスタンス の更新

プラットフォームの選択

Amazon EC2 はリザーブドインスタンスで次のプラットフォームをサポートしています。

- Linux/UNIX
- Linux with SQL Server Standard
- Linux with SQL Server Web
- Linux with SQL Server Enterprise
- SUSE Linux
- Red Hat Enterprise Linux
- Red Hat Enterprise Linux with HA
- Windows
- Windows with SQL Server Standard
- Windows with SQL Server Web
- Windows with SQL Server Enterprise

考慮事項

- Red Hat Enterprise Linux、SUSE Linux、または Ubuntu Pro の既存のサブスクリプション (BYOS) を持ち込む場合は、Linux/Unix プラットフォーム用のサービスを選択する必要があります。

- リザーブドインスタンスは、macOS または Ubuntu Pro (EC2 サブスクリプションを含む、つまり BYOS ではない) を実行しているインスタンスではサポートされていません。オンデマンドインスタンスの料金で節約するには、macOS および Ubuntu Pro (EC2 サブスクリプションを含む) インスタンスと Savings Plans の併用をお勧めします。詳細については、「Savings Plans ユーザーガイド」を参照してください。

インスタンスが特定のリザーブドインスタンスで実行されるようにするには、リザーブドインスタンスのプラットフォームが、インスタンスの起動に使用される AMI のプラットフォームと一致している必要があります。Linux AMI の場合は、AMI プラットフォームが一般値の [Linux/UNIX] を使用しているか、[SUSE Linux] などのより具体的な値を使用しているかを確認することが重要です。

Console

AMI プラットフォームを確認するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. AMI を選択します。
4. [詳細] タブで、[プラットフォームの詳細] の値をメモします。

AWS CLI

AMI プラットフォームを確認するには

describe-images コマンドを使用して `PlatformDetails` の値を確認します。

```
aws ec2 describe-images \
    --image-id ami-0abcdef1234567890 \
    --query Images[*].PlatformDetails
```

以下は出力の例です。

```
[
    "Linux/UNIX"
]
```

PowerShell

AMI プラットフォームを確認するには

Get-EC2Image コマンドレットを使用して、PlatformDetails の値を確認します。

```
Get-EC2Image `
    -ImageId ami-0abcdef1234567890 | `
    Select PlatformDetails
```

以下は出力の例です。

```
PlatformDetails
---------------
Linux/UNIX
```

購入をキューに入れる

デフォルトではリザーブドインスタンス を購入すると、すぐに購入が行われます。別の方法として、将来の日時の購入予約をキューに入れることができます。例えば、既存の リザーブドインスタンス が期限切れになる頃の購入予約をキューに入れることができます。これにより、サービスを切れ目なく利用できます。

リザーブドインスタンス の購入予約をキューに入れる場合、リージョンは指定できますが、ゾーンを指定した リザーブドインスタンス の購入予約や、他の販売者からの リザーブドインスタンス の購入予約を行うことはできません。購入予約は 3 年先までキューに入れることができます。予約した日時に、デフォルトの支払い方法を使用して購入が実行されます。支払いが正常に行われると、支払い特典が適用されます。

Amazon EC2 コンソールでキューに入れた購入の日付を設定でき、その購入はこの日付の 00:00 UTC までキューに入った状態になります。キューに入っている購入に対して別の時間を指定するにはAWS SDK またはコマンドラインツールを使用します。

キューに入れた購入予約は Amazon EC2 コンソールで確認できます。キューに入れた購入予約のステータスは [queued] になります。キューに入れた購入予約は予約日の前にいつでもキャンセルできます。詳細についてはキューに入れた購入のキャンセルを参照してください。

スタンダード リザーブドインスタンス の購入

スタンダード リザーブドインスタンス を特定のアベイラビリティーゾーンで購入し、キャパシティーの予約ができます。またはキャパシティの予約を見送り、リージョンのスタンダード リザーブドインスタンス を購入することもできます。

購入の完了後、リザーブドインスタンス の仕様と一致するインスタンスをすでに実行している場合は支払い特典が即座に適用されます。インスタンスを再起動する必要はありません。適切な実行中のインスタンスが存在しない場合、インスタンスを起動して、リザーブドインスタンスに指定した条件と正確に一致していることを確認します。詳細については「お使いの リザーブドインスタンス の使用」を参照してください。

実行しているインスタンスにどのように リザーブドインスタンス が適用されるかについての例はリザーブドインスタンスの割引はどのように適用されるかを参照してください。

Console

スタンダードリザーブドインスタンスを購入するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [リザーブドインスタンス] を選択し、[リザーブドインスタンス の購入] を選択してください。
3. [提供クラス] で [標準] を選択し、標準 リザーブドインスタンス を表示します。
4. キャパシティーの予約を購入するには購入画面の右上で、 [キャパシティ予約のある提供タイプのみ表示] をオンにします。この設定をオンにすると、[アベイラビリティーゾーン] フィールドが表示されます。

   リージョンの リザーブドインスタンス を購入するにはこの設定をオフにします。この設定をオフにすると、[アベイラビリティーゾーン] フィールドが消えます。
5. 必要に応じて他の設定を選択してから、[検索] をクリックしてください。
6. 購入する リザーブドインスタンス ごとに、希望する数量を入力して、[カートに入れる] を選択してください。

   Reserved Instance Marketplace からスタンダードリザーブドインスタンスを購入するには検索結果の [販売者] 列から、[サードパーティ] を見つけます。[期間] 列には標準以外の期間が表示されます。詳細については「Reserved Instance Marketplace からの購入」を参照してください。
7. 選択した リザーブドインスタンス の概要を確認するには[カートを見る] を選択してください。
8. [注文日] が [Now (今すぐ注文)] になっている場合は[すべて注文] をクリックすれば即時購入が完了します。購入予約をキューに入れるには[Now] を選択して日付を選択してください。カート内の有効なサービスごとに別の日付を選択できます。購入は選択した日付の 00:00 UTC までキューに入れられます。

9. 注文を確定するには[すべて注文] をクリックしてください。

   注文時に、選択したインスタンスと同等でより安価なインスタンスがある場合、AWS はより安価なインスタンスを販売します。

10. [閉じる] を選択してください。

    注文のステータスは [State] 列に表示されます。注文が確定されると、[State] の値が [Payment-pending] から [Active] に変わります。リザーブドインスタンス が `Active` となっていれば、使用準備が完了しています。

    ステータスが `Retired` になると、AWS は支払いを受領していない場合があります。

AWS CLI

スタンダードリザーブドインスタンス を購入するには

1. describe-reserved-instances-offerings コマンドを使用して、利用できる リザーブドインスタンス を見つけます。スタンダードリザーブドインスタンスのみを返すには `standard` を `--offering-class` オプションに指定します。追加の条件を適用して、結果を絞り込むことができます。例えば、`Linux/UNIX` のデフォルトテナンシーのリージョナル `t2.large` リザーブドインスタンスを 1 年間の期間だけで購入するには、次のコマンドを使用します。

```
aws ec2 describe-reserved-instances-offerings \
    --instance-type t2.large \
    --offering-class standard \
    --product-description "Linux/UNIX" \
    --instance-tenancy default \
    --filters Name=duration,Values=31536000 \
              Name=scope,Values=Region
```

   Reserved Instance Marketplace からのみの リザーブドインスタンス を探すには`marketplace` フィルターを使用します。期間が 1 年間 あるいは 3 年間より短い場合があるため、リクエストに期間は指定しません。

```
aws ec2 describe-reserved-instances-offerings \
    --instance-type t2.large \
    --offering-class standard \
    --product-description "Linux/UNIX" \
    --instance-tenancy default \
```

```
    --filters Name=marketplace,Values=true
```

ニーズに合う リザーブドインスタンス が見つかったら、提供 ID を書き留めます。次に例を示します。

```
"ReservedInstancesOfferingId": "bec624df-a8cc-4aad-a72f-4f8abc34caf2"
```

2. purchase-reserved-instances-offering コマンドを使用して、リザーブドインスタンス を購入します。前のステップで取得した リザーブドインスタンス 提供 ID を指定し、予約するインスタンスの数を指定する必要があります。

```
aws ec2 purchase-reserved-instances-offering \
    --reserved-instances-offering-id bec624df-a8cc-4aad-a72f-4f8abc34caf2 \
    --instance-count 1
```

デフォルトでは購入は即座に実行されます。別の方法として、購入予約をキューに入れるには次のオプションを前の呼び出しに追加します。

```
--purchase-time "2020-12-01T00:00:00Z"
```

3. describe-reserved-instances コマンドを使用して、購入したリザーブドインスタンスのステータスを確認します。

```
aws ec2 describe-reserved-instances \
    --reserved-instances-ids b847fa93-e282-4f55-b59a-1342fec06327 \
    --query ReservedInstances[].State
```

PowerShell

スタンダードリザーブドインスタンス を購入するには

1. Get-EC2ReservedInstancesOffering コマンドレットを使用して、使用可能なリザーブドインスタンスを見つけます。スタンダード リザーブドインスタンス のみを返すには`standard` を `-OfferingClass` パラメータに指定します。追加の条件を適用して、結果を絞り込むことができます。例えば、Linux/UNIX のデフォルトテナンシーのリージョナル `t2.large` リザーブドインスタンスを 1 年間の期間だけで購入するには、次のコマンドを使用します。

```
Get-EC2ReservedInstancesOffering `
    -InstanceType "t2.large" `
```

```
    -OfferingClass "standard" `
    -ProductDescription "Linux/UNIX" `
    -InstanceTenancy "default" `
    -Filters @{Name="duration"; Values="31536000"} `
             @{Name="scope"; Values="Region"
```

Reserved Instance Marketplace からのみの リザーブドインスタンス を探すには`marketplace` フィルターを使用します。期間が 1 年間 あるいは 3 年間より短い場合があるため、リクエストに期間は指定しません。

```
Get-EC2ReservedInstancesOffering `
    -InstanceType t2.large `
    -OfferingClass "standard" `
    -ProductDescription "Linux/UNIX" `
    -InstanceTenancy default `
    -Filters @{Name="marketplace"; Values="true"}
```

ニーズに合う リザーブドインスタンス が見つかったら、提供 ID を書き留めます。例えば、次のようになります。

```
bec624df-a8cc-4aad-a72f-4f8abc34caf2
```

2. New-EC2ReservedInstance コマンドレットを使用して、リザーブドインスタンスを購入します。前のステップで取得した リザーブドインスタンス 提供 ID を指定し、予約するインスタンスの数を指定する必要があります。

```
New-EC2ReservedInstance `
    -ReservedInstancesOfferingId "bec624df-a8cc-4aad-a72f-4f8abc34caf2" `
    -InstanceCount 1
```

デフォルトでは購入は即座に実行されます。別の方法として、購入予約をキューに入れるには次のパラメータを前の呼び出しに追加します。

```
-PurchaseTime "2020-12-01T00:00:00Z"
```

3. Get-EC2ReservedInstance コマンドレットを使用して、リザーブドインスタンスのステータスを取得します。

```
Get-EC2ReservedInstance `
    -ReservedInstancesId b847fa93-e282-4f55-b59a-1342fec06327 | `
```

```
    Select State
```

コンバーティブルリザーブドインスタンス の購入

コンバーティブルリザーブドインスタンス を特定のアベイラビリティーゾーンで購入し、キャパシティーの予約ができます。またはキャパシティの予約を見送り、リージョン コンバーティブルリザーブドインスタンス を購入することもできます。

リザーブドインスタンス の仕様と一致するインスタンスをすでに実行している場合、料金上の利点は即時適用されます。インスタンスを再起動する必要はありません。適切な実行中のインスタンスが存在しない場合、インスタンスを起動して、リザーブドインスタンス に指定した条件と一致していることを確認します。詳細については「お使いの リザーブドインスタンス の使用」を参照してください。

実行しているインスタンスにどのように リザーブドインスタンス が適用されるかについての例はリザーブドインスタンスの割引はどのように適用されるかを参照してください。

Console

コンバーティブルリザーブドインスタンスを購入するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [リザーブドインスタンス] を選択し、[リザーブドインスタンス の購入] を選択してください。
3. [提供クラス] で [コンバーティブル] を選択し、コンバーティブルリザーブドインスタンス を表示します。
4. キャパシティーの予約を購入するには購入画面の右上で、 [キャパシティ予約のある提供タイプのみ表示] をオンにします。この設定をオンにすると、[アベイラビリティーゾーン] フィールドが表示されます。

   リージョンの リザーブドインスタンス を購入するにはこの設定をオフにします。この設定をオフにすると、[アベイラビリティーゾーン] フィールドが消えます。
5. 必要に応じて他の設定を選択してから、[Search] を選択してください。
6. 購入する コンバーティブルリザーブドインスタンス ごとに数量を入力して、[カートに入れる] を選択してください。
7. 選択内容の概要を表示するには[カートを見る] を選択してください。

8. [注文日] が [Now (今すぐ注文)] になっている場合は[すべて注文] をクリックすれば即時購入が完了します。購入予約をキューに入れるには[Now] を選択して日付を選択してください。カート内の有効なサービスごとに別の日付を選択できます。購入は選択した日付の 00:00 UTC までキューに入れられます。
9. 注文を確定するには[すべて注文] をクリックしてください。

   注文時に、選択したインスタンスと同等でより安価なインスタンスがある場合、AWS はより安価なインスタンスを販売します。
10. [閉じる] を選択してください。

    注文のステータスは [State] 列に表示されます。注文が確定されると、[State] の値が [Payment-pending] から [Active] に変わります。リザーブドインスタンス が Active となっていれば、使用準備が完了しています。

    ステータスが Retired になると、AWS は支払いを受領していない場合があります。

AWS CLI

コンバーティブルリザーブドインスタンスを購入するには

1. describe-reserved-instances-offerings コマンドを使用して、利用できる リザーブドインスタンス を見つけます。コンバーティブルリザーブドインスタンスだけを返すには convertible を --offering-class オプションに指定します。追加の条件を適用して、結果を絞り込むことができます。例えば、Linux/UNIX のデフォルトテナンシーのリージョナル t2.large リザーブドインスタンスを購入するには、次のコマンドを使用します。

   ```
   aws ec2 describe-reserved-instances-offerings \
       --instance-type t2.large \
       --offering-class convertible \
       --product-description "Linux/UNIX" \
       --instance-tenancy default \
       --filters Name=scope,Values=Region
   ```

   ニーズに合う リザーブドインスタンス が見つかったら、提供 ID を書き留めます。次に例を示します。

   ```
   "ReservedInstancesOfferingId": "bec624df-a8cc-4aad-a72f-4f8abc34caf2"
   ```

2. purchase-reserved-instances-offering コマンドを使用して、リザーブドインスタンス を購入します。前のステップで取得した リザーブドインスタンス 提供 ID を指定し、予約するインスタンスの数を指定する必要があります。

```
aws ec2 purchase-reserved-instances-offering \
    --reserved-instances-offering-id bec624df-a8cc-4aad-a72f-4f8abc34caf2 \
    --instance-count 1
```

デフォルトでは購入は即座に実行されます。別の方法として、購入予約をキューに入れるには次のオプションを前の呼び出しに追加します。

```
--purchase-time "2020-12-01T00:00:00Z"
```

3. describe-reserved-instances コマンドを使用して、購入したリザーブドインスタンスのステータスを確認します。

```
aws ec2 describe-reserved-instances \
    --reserved-instances-ids b847fa93-e282-4f55-b59a-1342fec06327 \
    --query ReservedInstances[].State
```

PowerShell

コンバーティブルリザーブドインスタンスを購入するには

1. Get-EC2ReservedInstancesOffering コマンドレットを使用して、使用可能なリザーブドインスタンスを見つけます。コンバーティブルリザーブドインスタンス だけを返すには`convertible` を `-OfferingClass` パラメータに指定します。追加の条件を適用して、結果を絞り込むことができます。例えば、`Linux/UNIX` のデフォルトテナンシーのリージョナル `t2.large` リザーブドインスタンスを購入するには、次のコマンドを使用します。

```
Get-EC2ReservedInstancesOffering `
    -InstanceType "t2.large" `
    -OfferingClass "convertible" `
    -ProductDescription "Linux/UNIX" `
    -InstanceTenancy "default" `
    -Filters @{Name="scope"; Values="Region"
```

ニーズに合う リザーブドインスタンス が見つかったら、提供 ID を書き留めます。例えば、次のようになります。

```
bec624df-a8cc-4aad-a72f-4f8abc34caf2
```

2. New-EC2ReservedInstance コマンドレットを使用して、リザーブドインスタンスを購入します。前のステップで取得したリザーブドインスタンス提供 ID を指定し、予約するインスタンスの数を指定する必要があります。

```
New-EC2ReservedInstance `
    -ReservedInstancesOfferingId "bec624df-a8cc-4aad-a72f-4f8abc34caf2" `
    -InstanceCount 1
```

デフォルトでは購入は即座に実行されます。別の方法として、購入予約をキューに入れるには次のパラメータを前の呼び出しに追加します。

```
-PurchaseTime "2020-12-01T00:00:00Z"
```

3. Get-EC2ReservedInstance コマンドレットを使用して、リザーブドインスタンスのステータスを取得します。

```
Get-EC2ReservedInstance `
    -ReservedInstancesId b847fa93-e282-4f55-b59a-1342fec06327 | `
    Select State
```

Reserved Instance Marketplace からの購入

Reserved Instance Marketplace から、サードパーティーの販売者が所有しており、必要がなくなったリザーブドインスタンスを購入できます。これを行うにはAmazon EC2 コンソールまたはコマンドラインツールを使用します。このプロセスはAWS から リザーブドインスタンス を購入する場合と似ています。詳細については「スタンダード リザーブドインスタンス の購入」を参照してください。

Reserved Instance Marketplace で購入したリザーブドインスタンスと、AWS から直接購入したリザーブドインスタンスとの間には以下のような違いがあります。

- 期間 – サードパーティー販売者から購入した リザーブドインスタンス は残り期間が完全な標準期間よりも短くなります。AWS の完全な標準期間は 1 年または 3 年間です。
- 前払い料金 – サードパーティーの リザーブドインスタンス はさまざまな前払い料金で販売されます。使用に対して、または定期的に支払う料金はリザーブドインスタンスを最初に AWS から購入した際に設定された料金と同じ金額です。

- リザーブドインスタンスのタイプ – Reserved Instance Marketplace から購入できるのはAmazon EC2 スタンダードリザーブドインスタンスのみです。コンバーティブルリザーブドインスタンス、Amazon RDS、および Amazon ElastiCache のリザーブドインスタンスはReserved Instance Marketplace では購入できません。

お客様に関する基本情報 (郵便番号や国情報など) は販売者と共有されます。

この情報を使用して、販売者は国に支払う必要な取引税 (売上税や付加価値税など) を計算し、支払いレポートとして提示します。まれに、AWS が販売者に E メールアドレスを提供する必要がある場合があります。これは販売者が、販売に関する質問があり、それに関して連絡できるようにするためです (例えば税務上の質問など)。

同様の理由で、AWS は購入者の請求書に販売者の正式名を記載します。税金または関連する理由で販売者の情報が必要な場合はサポート までお問い合わせください。

キューに入れた購入のキャンセル

購入予約は 3 年先までキューに入れることができます。キューに入れた購入予約は予約日の前にいつでもキャンセルできます。

Console

キューに入れた購入をキャンセルするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Reserved Instances] を選択してください。
3. 1 つまたは複数の リザーブドインスタンス を選択してください。
4. [アクション]、[キュー入りリザーブドインスタンスの削除] の順にクリックしてください。
5. 確認を求められたら、[削除] をクリックし、次に [閉じる] をクリックします。

AWS CLI

キューに入れた購入をキャンセルするには

delete-queued-reserved-instances コマンドを使用します。

```
aws ec2 delete-queued-reserved-instances \
    --reserved-instances-ids b847fa93-e282-4f55-b59a-1342fec06327
```

PowerShell

キューに入れた購入をキャンセルするには

Remove-EC2QueuedReservedInstance コマンドレットを使用します。

```
Remove-EC2QueuedReservedInstance `
    -ReservedInstancesId b847fa93-e282-4f55-b59a-1342fec06327
```

リザーブドインスタンス の更新

リザーブドインスタンス は有効期限が切れる前に更新できます。リザーブドインスタンス を更新すると、現在の リザーブドインスタンス が期限切れになるまで、同じ設定の リザーブドインスタンス の購入予約がキューに入れられます。

Amazon EC2 コンソールを使用してリザーブドインスタンスを更新する必要があります。

キューに入れた購入予約を使用してリザーブドインスタンスを更新するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Reserved Instances] を選択してください。
3. 更新するリザーブドインスタンスを選択してください。
4. [Actions (アクション)]、[Renew Reserved Instances (リザーブドインスタンスの更新)] の順に選択してください。
5. 注文を確定するには[すべて注文]、[閉じる] の順にクリックしてください。

リザーブドインスタンスマーケットプレイスで Amazon EC2 のリザーブドインスタンスを販売する

Amazon EC2 リザーブドインスタンスマーケットプレイスはAWS のお客様およびサードパーティー販売者が使用していないスタンダードリザーブドインスタンスを簡単に販売できるようにするプラットフォームです。これらのリザーブドインスタンスは契約期間と料金オプションが異なる場合があります。インスタンスを新しい AWS リージョンに移動する場合、別のインスタンスタイプに変更する場合、リザーブドインスタンスの契約期間が終了する前にプロジェクトが終了する場合、ビジネスニーズが変化した場合、容量が余った場合など、不要になったリザーブドインスタンスを販売することをお勧めします。

リザーブドインスタンスはReserved Instance Marketplace への出品直後から、購入希望者に表示されます。すべての リザーブドインスタンス は残り期間や時間料金別にグループ化されます。

サードパーティー販売者のリザーブドインスタンスを、リザーブドインスタンスマーケットプレイスから購入したいという購入者のリクエストに応えるため、AWS では指定されたグループ内で前払い料金が最も安いリザーブドインスタンスを最初に販売します。次に AWS は購入者の注文全体が満たされるまで、料金が低い順にリザーブドインスタンスを販売します。AWS はその後、トランザクションを処理しリザーブドインスタンスの所有権を購入者に移転します。

出品した リザーブドインスタンス は売れるまでお客様の所有です。販売後は予約済みのキャパシティーと割引使用料金は使用できません。インスタンスの使用が継続している間、AWS はリザーブドインスタンスが販売された時間から起算したオンデマンド料金を課金します。

Reserved Instance Marketplace で、使用していないリザーブドインスタンスを販売するには特定の要件基準を満たしている必要があります。

Reserved Instance Marketplace でのリザーブドインスタンスの購入の詳細についてはReserved Instance Marketplace からの購入を参照してください。

内容

- 制約と制限
- 販売者として登録する
- 支払い用の銀行口座
- 税金情報
- リザーブドインスタンス 料金設定
- リザーブドインスタンス のリスト
- リザーブドインスタンス の出品状態
- 出品のライフサイクル
- リザーブドインスタンス が売却された後
- 支払いを受け取る
- 購入者と共有する情報

制約と制限

使用していないリザーブドインスタンスを販売できるようになる前に、Reserved Instance Marketplace に、自分を販売者として登録する必要があります。詳細については販売者として登録する を参照してください。

リザーブドインスタンス の販売時に次の制約と制限が適用されます:

- Reserved Instance Marketplace で販売可能なのはAmazon EC2 スタンダードのリージョンとゾーンのリザーブドインスタンスのみです。
- Amazon EC2 コンバーティブルリザーブドインスタンスはリザーブドインスタンスマーケットプレイスでは販売できません。
- 他の AWS サービス (Amazon RDS や Amazon ElastiCache など) 向けのリザーブドインスタンスはReserved Instance Marketplace では販売できません。
- スタンダード リザーブドインスタンス の有効期間が 1 か月以上残っている必要があります。
- スタンダードリザーブドインスタンスはデフォルトで無効になっているリージョンでは販売することができません。
- Reserved Instance Marketplace で許容される最低販売価格は0.00 USD です。
- 前払いなし、一部前払い、または全額前払いのリザーブドインスタンスはアカウント内で 30 日間以上アクティブになっている限り、リザーブドインスタンス Marketplace で販売できます。さらに、リザーブドインスタンスに前払いがある場合はAWS が前払い料金を受領した後にのみ販売できます。
- ボリューム割引を使用して購入したリザーブドインスタンスを、リザーブドインスタンスマーケットプレイスで販売することはできません。
- Reserved Instance Marketplace に表示された出品内容は直接変更することはできません。ただし、最初に出品をキャンセルしてから、新しいパラメータで別の出品を作成することはできます。詳細についてはリザーブドインスタンス 料金設定 を参照してください。出品する前に リザーブドインスタンス を変更することもできます。詳細についてはリザーブドインスタンス の変更 を参照してください。
- AWS はリザーブドインスタンスマーケットプレイスで販売するスタンダードリザーブドインスタンスごとに、前払料金の総額の 12% をサービス料として課金します。前払い価格は販売者がスタンダード リザーブドインスタンス に課金する価格です。
- 販売者として登録する場合、指定する銀行には米国の住所が必要です。詳細についてはAWS Marketplace 販売者ガイドの有料製品の販売者の追加要件を参照してください。
- Amazon Web Services India Private Limited (AWS India) のお客様は、米国の銀行口座をお持ちであっても、EC2 リザーブドインスタンスマーケットプレイスに販売者として登録できません。また、EC2 リザーブドインスタンスマーケットプレイスでリザーブドインスタンスを出品または販売することもできません。詳細については「AWS アカウント と AWS India アカウントの違い」を参照してください。

- 登録販売者を Amazon Web Services India Private Limited (AWS India) に変更すると、EC2 リザーブドインスタンスマーケットプレイスから販売者として登録解除され、EC2 リザーブドインスタンスマーケットプレイスの既存のアクティブな出品はすべて削除されます。販売者ステータスを復元するには、アカウントの場所をインド以外の国に変更し、販売者登録手続きを再度完了する必要があります。

販売者として登録する

> **Note**
>
> アカウントを販売者として登録できるのはAWS アカウントのルートユーザー のみです。

Reserved Instance Marketplace で販売するには最初に自分を販売者として登録する必要があります ます。登録時には以下の情報を指定します。

- 銀行情報 – AWS は予約の販売時に集金された金額をお支払いするために、お客様の銀行情報を必要とします。住所が米国内の銀行でなければなりません。詳細については「支払い用の銀行口座」を参照してください。
- 税金情報 — 必要な税金報告義務を判断するために、販売者は必ず、税金情報の質問に回答する必要があります。詳細については「税金情報」を参照してください。

記入された販売者登録が AWS に受領されると、登録を確認する電子メールが送信され、Reserved Instance Marketplace での販売を開始できることが伝えられます。

支払い用の銀行口座

AWS はリザーブドインスタンスの販売時に集金された金額をお支払いするために、お客様の銀行情報を必要とします。住所が米国内の銀行でなければなりません。詳細についてはAWS Marketplace 販売者ガイドの有料製品の販売者の追加要件を参照してください。

支払い用のデフォルトの銀行口座を登録するには

1. [Reserved Instance Marketplace 販売者登録] ページを開き、AWS の認証情報を使用してサインインします。
2. [Manage Bank Account] ページで、支払いを受け取る銀行に関する以下の情報を提供します。
    - 銀行口座の名義

- 支店コード
- アカウント番号
- 銀行口座の種類

> **Note**
>
> 法人の銀行口座を使用する場合は口座に関する情報を FAX (1-206-765-3424) で送信するように指示されます。

登録後、指定された銀行口座がデフォルトとして設定され、銀行の確認待ちとなります。新しい銀行口座を確認するには最長で 2 週間かかります。この間は支払金を受け取ることができません。確立済みの口座の場合、支払い完了まで通常およそ 2 日かかります。

支払い用のデフォルトの銀行口座を変更するには

1. [Reserved Instance Marketplace 販売者登録] ページで、登録時に使用したアカウントでサインインします。
2. [Manage Bank Account] ページで、必要に応じて新規口座のアカウントを追加するか、デフォルトの銀行口座を変更します。

税金情報

リザーブドインスタンス の販売には取引関連の税金 (消費税または付加価値税など) がかかることがあります。取引関連の税金が適用されるかどうかについては税務部、法務部、財務部、または経理部に確認する必要があります。お客様は取引関連の税金を収集し、該当する税務署に納める役割を担います。

販売者登録手続きの一環として、販売者登録ポータルの Tax Interview を完了させる必要があります。このインタビューでは税金情報を収集し、IRS フォーム W-9、W-8BEN、または W-8BEN-E に入力してください。このフォームは必要な税金報告義務を明確にするために使用されます。

Tax Interview の一環として入力する税金情報は個人または法人であるか、また、米国人または非米国人 (米国企業または非米国企業) かによって異なる場合があります。Tax interview の記入を行う際は次に注意してください。

- AWS が提供する情報 (このトピックの情報を含む) は税金、法律、またはその他の専門的なアドバイスではありません。IRS のレポート要件がビジネスに及ぼす影響について知りたい場合、またはご質問がある場合は税金、法律、またはその他の専門家にお問い合わせください。
- IRS のレポート要件をできるだけ効率的に満たすにはTax interview の中で要求されたすべての質問に答え、情報を入力してください。
- 答えを確認します。綴りを間違ったり、誤った税金識別番号を入力したりしないようにします。これらのミスがあると、誤った税金フォームが生成されます。

税金に関する質問の回答と IRS 報告書のしきい値に基づいて、Amazon は Form 1099-K を提出する場合があります。 Amazon はお客様の納税金額がしきい値レベルに達した年の翌年の 1 月 31 日までに、フォーム 1099-K のコピーを郵送します。例えば、税金口座が 2018 年にしきい値に達した場合は2019 年の 1 月 31 日までにフォーム 1099-K が郵送されます。

IRS の要件とフォーム 1099-K の詳細についてはIRSのウェブサイトを参照してください。

リザーブドインスタンス 料金設定

リザーブドインスタンスの料金を設定するときは次の点を考慮してください。

- 前払い料金 – 前払い料金は販売するリザーブドインスタンスについて指定できる唯一の料金です。前払い料金とは、購入者がリザーブドインスタンスを購入する際に支払う一括払いの料金です。

  リザーブドインスタンス の価値は時間の経過に伴い低下するため、AWS ではデフォルトで、1 か月ごとに同じ割合で低下するような価格設定を行っています。ただし、予約を販売する時期に基づいて、異なる前払い価格を設定できます。例えば、リザーブドインスタンス の残りの期間が 9 か月の場合、顧客が残り 9 か月の リザーブドインスタンス を購入する場合、受領する額を指定できます。残りが 5 か月である別の価格を設定し、さらに残りが 1 か月の別の価格を設定することができます。

  Reserved Instance Marketplace で許容される最低販売価格は0.00 USD です。
- 制限 – リザーブドインスタンスの販売に関する次の制限は AWS アカウント の有効期間中に適用されます。これらは年間制限ではないため、引き上げることはできません。
  - リザーブドインスタンスの販売額は、最大 50,000 USD です。
  - 最大 5,000 のリザーブドインスタンスを販売可能です。
- [変更不可] – 出品内容を直接変更することはできません。ただし、最初に出品をキャンセルしてから、新しいパラメータで別の出品を作成することはできます。

- [キャンセル可能] – 出品はactive 状態であれば、いつでもキャンセルできます。既にマッチングされていたり、販売処理が行われている出品はキャンセルできません。出品したインスタンスの一部がマッチングされている場合にその出品をキャンセルすると、マッチングされていない残りのインスタンスが出品から削除されます。

リザーブドインスタンス のリスト

登録済みの販売者の場合、販売する リザーブドインスタンス を 1 つまたは複数選択できます。1 件のリストにすべてまとめて販売することも、個別に販売することもできます。さらに、インスタンスタイプ、プラットフォーム、スコープのすべての設定で リザーブドインスタンス を出品できます。

コンソールによって、提示価格が決定します。お客様の リザーブドインスタンス に一致するサービスを確認し、最低価格のサービスに合わせます。それ以外の場合、残り時間の リザーブドインスタンス のコストに基づいて提示価格を計算します。計算後の値が 1.01 USD 未満の場合、提示価格は 1.01 USD です。

出品をキャンセルする場合、その一部が既に売れているとき、売却済みの部分についてのキャンセルは無効です。出品されたうち、まだ売れていない部分のみが Reserved Instance Marketplace から削除されます。

Console

リザーブドインスタンスマーケットプレイス内のリザーブドインスタンスを一覧表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Reserved Instances] を選択してください。
3. 出品する リザーブドインスタンス を選択し、[Actions (アクション)]、[リザーブドインスタンス の出品] の順に選択してください。
4. [Configure Your リザーブドインスタンス Listing (リザーブドインスタンス出品の設定)] ページで、販売するインスタンス数および残り期間に対する前払い価格を該当列に設定します。[Months Remaining] 列の隣にある矢印を選択して、残りの有効期間における予約の価値の変化状況を確認します。
5. 上級ユーザーが価格をカスタマイズする場合は今後の月に異なる価格を入力できます。デフォルトの直線形の価格減少に戻すには[Reset] を選択してください。
6. 出品の設定が終了したら、[Continue] を選択してください。

7. [Confirm Your リザーブドインスタンス Listing (リザーブドインスタンス出品の確認)] ページに表示された出品詳細を確認し、問題がなければ [List Reserved Instance (リザーブドインスタンスの出品)] を選択してください。

出品したインスタンスを Amazon EC2 コンソールで表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Reserved Instances] を選択してください。
3. 出品した リザーブドインスタンス を選択し、ページ下部にある [My Listings (自分の出品)] タブを選択してください。

AWS CLI

Reserved Instance Marketplace 内のリザーブドインスタンスを管理するには

1. describe-reserved-instances コマンドを使用して、リザーブドインスタンス の一覧を取得します。出品するリザーブドインスタンスの ID をメモします。
2. create-reserved-instances-listing コマンドを使用します。リザーブドインスタンス の ID、インスタンスの数、価格体系を指定する必要があります。
3. ユーザーの出品を表示するにはdescribe-reserved-instances-listings コマンドを使用します。
4. 出品をキャンセルするには cancel-reserved-instances-listing コマンドを使用します。

PowerShell

Reserved Instance Marketplace 内のリザーブドインスタンスを管理するには

1. Get-EC2ReservedInstance コマンドレットを使用して、リザーブドインスタンスのリストを取得します。出品するリザーブドインスタンスの ID をメモします。
2. New-EC2ReservedInstancesListing コマンドレットを使用します。リザーブドインスタンスの ID、インスタンスの数、価格体系を指定する必要があります。
3. 出品を表示するには Get-EC2ReservedInstancesListing コマンドレットを使用します。
4. 出品をキャンセルするには Stop-EC2ReservedInstancesListing コマンドレットを使用します。

リザーブドインスタンス の出品状態

リザーブドインスタンス ページの [出品] タブの [出品状態] には出品の現在状況が表示されます。

[出品状態] にはReserved Instance Marketplace へのお客様の出品に関する情報が表示されます。これは[リザーブドインスタンス] ページの [State] 列に表示される状態情報とは異なります。この [State] 情報はお客様の予約に関するものです。

- [アクティブ] — 購入できます。
- [キャンセル済み] – 出品がキャンセルされ、Reserved Instance Marketplace での購入ができません。
- [closed (クローズ)] — リザーブドインスタンス は出品されていません。リザーブドインスタンス は出品が完了したため [`closed`] になっている可能性があります。

出品のライフサイクル

出品したすべてのインスタンスがマッチングされて売れると、[My Listings] タブに表示される [Total instance count] が [Sold] の下に表示された数と同じになります。出品で残っている [Available] インスタンスがなくなり、[Status] が [`closed`] になります。

出品の一部だけが売却された場合、AWS では出品されている リザーブドインスタンス を取り下げ、売却されなかった残りの リザーブドインスタンスと同数のリザーブドインスタンス を作成します。したがって、出品 ID とその ID の出品は販売中の予約が少なくなっていますがアクティブのままです。

この出品内の リザーブドインスタンス の売却は今後、この方法で行われます。出品されたすべての リザーブドインスタンス が売却されると、AWS はその出品を `closed` としてマークします。

例えば、出品数が 5 のリザーブドインスタンス ID 5ec28771-05ff-4b9b-aa31-9e57dexample の出品を作成するとします。

コンソールの [Reserved Instance] ページの [My Listings] タブに、次のように出品が表示されます。

リザーブドインスタンス の出品 ID 5ec28771-05ff-4b9b-aa31-9e57dexample

- Total reservation count = 5
- Sold = 0
- Available = 5
- Status = active

購入者が 2 つの予約を購入すると、3 つの予約が販売用に残ります。一部分が売れたため、AWS では引き続き販売される残りの予約に相当する、3 つの新しい予約が作成されます。

お客様の出品は[My Listings (自分の出品)] タブで次のように表示されます。

リザーブドインスタンス の出品 ID 5ec28771-05ff-4b9b-aa31-9e57dexample

- Total reservation count = 5
- Sold = 2
- Available = 3
- Status = active

出品をキャンセルする場合、その一部が既に売れているとき、売却済みの部分についてのキャンセルは無効です。出品されたうち、まだ売れていない部分のみがリザーブドインスタンス Marketplace から削除されます。

リザーブドインスタンス が売却された後

自分のリザーブドインスタンス が売却されると、AWS から E メールの通知が届きます。何らかのアクティビティがあった日ごとに、毎日のすべてのアクティビティをキャプチャした 1 通の E メール通知が送信されます。アクティビティには出品の作成または販売や、AWS によるアカウントへの資金の送金などがあります。

Console

リザーブドインスタンスの出品の状態を追跡するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションページで、[リザーブドインスタンス] を選択してください。
3. [出品] タブで、[出品状態] の値を確認します。また、期間、出品価格、および出品されているインスタンスの中で使用可能、保留中、売却済み、およびキャンセルされたものがいくつあるかの詳細といった情報がこのタブに示されます。

AWS CLI

リザーブドインスタンスの出品の状態を追跡するには

describe-reserved-instances-listings コマンドを適切なフィルタで使用して、リザーブドインスタンスの出品に関する情報を取得します。

```
aws ec2 describe-reserved-instances-listings
```

PowerShell

リザーブドインスタンスの出品の状態を追跡するには

Get-EC2ReservedInstancesListing コマンドレットを使用します。

```
Get-EC2ReservedInstancesListing
```

支払いを受け取る

AWS が購入者からの支払い金を受領するとすぐに、販売された リザーブドインスタンス に登録されている所有者アカウントの E メールアドレスに、メッセージが送信されます。

AWS は指定された銀行口座に、自動決済機関 (ACH) の電子送金を送信します。通常、この送金はリザーブドインスタンス の売却後 1 ~ 3 日の間に行われます。支払いは1 日に 1 回行われます。送金が行われると、支払い報告がメールで届きます。AWS が銀行からの検証結果を受領するまで、支払い金を受け取れないことに注意してください。これには最大 2 週間かかることがあります。

販売した リザーブドインスタンス はユーザーの リザーブドインスタンス の詳細に引き続き表示されます。

販売者はリザーブドインスタンスに対する現金の支払いを、振込によって直接各自の銀行口座で受け取ります。 AWSはReserved Instance Marketplace で販売するリザーブドインスタンスごとに、前払い料金の総額の 12% をサービス料として課金します。

購入者と共有する情報

Reserved Instance Marketplace で販売する場合、AWS は米国の規制に従って、お客様の正式な会社名を購入者向けのステートメントに表示します。さらに、請求書またはその他の税金関連の理由で、購入者から販売者に連絡したいとの要望が サポート にあった場合、購入者から販売者に直接連絡できるよう、AWS は必要に応じて販売者の E メールアドレスを購入者に提供する場合があります。

同様の理由で、販売者には購入者の郵便番号および国情報が支払いレポートによって提供されます。販売者として、この情報を、国に支払わなければならない取引税 (売上税や付加価値税など) に添付する必要がある場合があります。

AWS は税金に関する助言を行うことはできませんが、お客様が追加情報を必要としているとお客様の税務専門家が判断した場合はサポート にお問い合わせください。

リザーブドインスタンス の変更

ニーズが変化したときはスタンダードまたは コンバーティブルリザーブドインスタンス を変更し、引き続き料金上の利点を得られます。アベイラビリティー–ゾーン、インスタンスサイズ (同じインスタンスファミリーや世代で) やリザーブドインスタンスのスコープなどの属性を変更できます。

> **Note**
>
> また、別の構成で、別の コンバーティブルリザーブドインスタンス の コンバーティブルリザーブドインスタンス に交換することもできます。詳細については「コンバーティブルリザーブドインスタンス の交換」を参照してください。

すべての リザーブドインスタンス、またはそのサブセットを変更できます。元の リザーブドインスタンス を 2 つ以上の新しい リザーブドインスタンス に分割できます。例えば、us-east-1a に 10 のインスタンスを予約があり、そのうち 5 つのインスタンスを us-east-1b に移動する場合、結果的にこの変更は 2 つの新しい予約をリクエストします。us-east-1a での 5 つのインスタンス用の予約と us-east-1b での 5 つのインスタンス用の予約です。

また、2 つ以上の リザーブドインスタンス を単一の リザーブドインスタンス にマージすることもできます。例えば、それぞれに 1 つのインスタンスがある 4 つの t2.small リザーブドインスタンスがある場合、これらをマージして 1 つの t2.large リザーブドインスタンス を作成できます。詳細については「インスタンスサイズの変更のサポート」を参照してください。

変更後、リザーブドインスタンス の利点はリザーブドインスタンスの新しいパラメータと一致するインスタンスのみに適用されます。例えば、予約のアベイラビリティーゾーンを変更する場合、キャパシティーの予約と料金上の利点は新しいアベイラビリティーゾーン内のインスタンスの使用に対して自動的に適用されます。新しいパラメータに一致しないインスタンスは他に適用可能な予約がない場合、オンデマンド価格で課金されます。

変更リクエストが成功した場合。

- 変更後の予約がすぐに有効になり、変更リクエストが完了した時刻から、割引料金が新しいインスタンスに適用されます。例えば、午後 9 時 15 分に予約の変更が成功した場合、割引料金は午後 9 時 00 分から新しいインスタンスに移ります。変更された リザーブドインスタンス の発行日は describe-reserved-instances コマンドを使用して取得できます。
- 元の予約は終了します。その終了日は新しい予約の開始日であり、新しい予約の終了日は元の リザーブドインスタンス の終了日と同じです。有効期限のうち 16 か月が残っている 3 年の予約を正常に変更した場合、変更後の予約は 16 か月の予約であり、終了日は変更前の予約と同じです。

- 変更後の予約の固定価格は 0 USD であり、元の予約の固定価格ではありません。
- 変更後の予約の固定価格はアカウントに適用される割引料金範囲の計算に影響を与えません。割引範囲の計算は元の予約の固定価格に基づきます。

変更リクエストに失敗した場合、リザーブドインスタンス は元の設定を維持し、別の変更リクエストをすぐに利用できます。

変更に手数料は必要なく、新しく課金されたり、請求書が届いたりすることはありません。

予約の変更は必要に応じて何度でも行うことができますが、変更を送信後に保留中の変更リクエストを変更またはキャンセルすることはできません。変更が完了した後は必要に応じて別の変更リクエストを送信して、実行した変更をロールバックできます。

内容

- 変更の要件と制限
- インスタンスサイズの変更のサポート
- 変更リクエストの送信
- 変更リクエストのトラブルシューティング

変更の要件と制限

以下のように、これらの属性を変更できます。

変更可能な属性	サポートされているプラットフォーム	制約事項と考慮事項
同じリージョン内でアベイラビリティーゾーンを変更する	Linux と Windows	-
スコープをアベイラビリティーゾーンからリージョンに、またはその逆に変更する	Linux と Windows	ゾーンのリザーブドインスタンスはアベイラビリティーゾーンとそのアベイラビリティーゾーンのリザーブドキャパシティにスコープされます。スコープをアベイラビ

変更可能な属性	サポートされているプラットフォーム	制約事項と考慮事項
		リティーゾーンからリージョンに (つまり、ゾーンからリージョナルに) 変更すると、キャパシティ予約のメリットが失われます。 リージョンのリザーブドインスタンスのスコープはリージョンに限定されます。リザーブドインスタンスの割引はそのリージョンの任意のアベイラビリティーゾーンで実行されているインスタンスに適用できます。さらに、リザーブドインスタンスの割引は選択したインスタンスファミリのすべてのサイズのインスタンスの使用に適用されます。スコープをリージョンからアベイラビリティーゾーンに (つまり、リージョナルからゾーンに) 変更すると、アベイラビリティーゾーンの柔軟性とインスタンスサイズの柔軟性 (該当する場合) が失われます。 詳細については「リザーブドインスタンスの割引はどのように適用されるか」を参照してください。

変更可能な属性	サポートされているプラットフォーム	制約事項と考慮事項
同じインスタンスファミリーや世代でインスタンスサイズを変更する	Linux/UNIX のみ リザーブドインスタンス のインスタンスサイズの柔軟性は他のプラットフォーム (Linux with SQL Server Standard、Linux with SQL Server Web、Linux with SQL Server Enterprise、Red Hat Enterprise Linux、SUSE Linux、Windows、Windows with SQL Standard、Windows with SQL Server Enterprise、および Windows with SQL Server Web) では利用できません。	予約ではデフォルトのテナンシーを使用する必要があります。使用できる他のサイズがないため、一部のインスタンスファミリーはサポートされません。詳細については「インスタンスサイズの変更のサポート」を参照してください。

要件

変更リクエストは新しい設定 (該当する場合) に対して十分なリザーブドインスタンス容量があり、以下の条件が満たされている場合に Amazon EC2 で処理されます。

- 購入と同時期またはその前に リザーブドインスタンス を変更できないこと
- リザーブドインスタンス がアクティブであること
- 保留中の変更リクエストがないこと
- リザーブドインスタンスは Reserved Instance Marketplace に出品されていないこと
- 元の予約のインスタンスサイズのフットプリントと新しい設定が一致している必要があります。詳細については「インスタンスサイズの変更のサポート」を参照してください。
- 元の リザーブドインスタンス はすべてスタンダード リザーブドインスタンス あるいはすべて コンバーティブルリザーブドインスタンス であり、両方のタイプが混ざっていないこと
- スタンダード リザーブドインスタンス の場合、元の リザーブドインスタンス は同じ時間内に期限切れとなること

- インスタンス サイズを変更するには、リザーブドインスタンスはインスタンスサイズの柔軟性をサポートしている必要があります。インスタンスサイズの柔軟性をサポートしていないリザーブドインスタンスのリストについては「インスタンスサイズの柔軟性」を参照してください。

インスタンスサイズの変更のサポート

次の要件が満たされている場合はリザーブドインスタンス のインスタンスサイズを変更できます。

要件

- プラットフォームが Linux/UNIX であること。
- 同じインスタンスファミリー (T などの文字で示される) と世代 (2 などの数字で示される) 内の別のインスタンスサイズを選択する必要があります。

  例えば、リザーブドインスタンスはどちらも同じ T2 ファミリーおよび世代に属しているため、リザーブドインスタンスを `t2.small` から `t2.large` に変更できます。ただし、どちらの例でも、ターゲットインスタンスのファミリーと世代が元のリザーブドインスタンスと同じではないため、リザーブドインスタンスを T2 から M2 に、または T2 から T3 に変更することはできません。
- リザーブドインスタンスがインスタンスサイズの柔軟性をサポートしている場合にのみ、リザーブドインスタンスのインスタンスサイズを変更できます。インスタンスサイズの柔軟性をサポートしていないリザーブドインスタンスのリストについては「インスタンスサイズの柔軟性」を参照してください。
- `t1.micro` には 1 つのサイズしかないため、`t1.micro` インスタンスのリザーブドインスタンスのインスタンスサイズは変更できません。
- 元のと新しい リザーブドインスタンス はインスタンスサイズのフットプリントが同じであること。

内容

- インスタンスサイズのフットプリント
- ベアメタルインスタンスの正規化係数

インスタンスサイズのフットプリント

各 リザーブドインスタンス にはインスタンスサイズのフットプリントがあり、これはインスタンスサイズの正規化係数と予約に含まれるインスタンスの数によって決まります。リザーブドインスタン

ス のインスタンスサイズを変更する場合、新しい設定のフットプリントは元の設定のフットプリントと一致する必要があります。一致しないと、変更リクエストは処理されません。

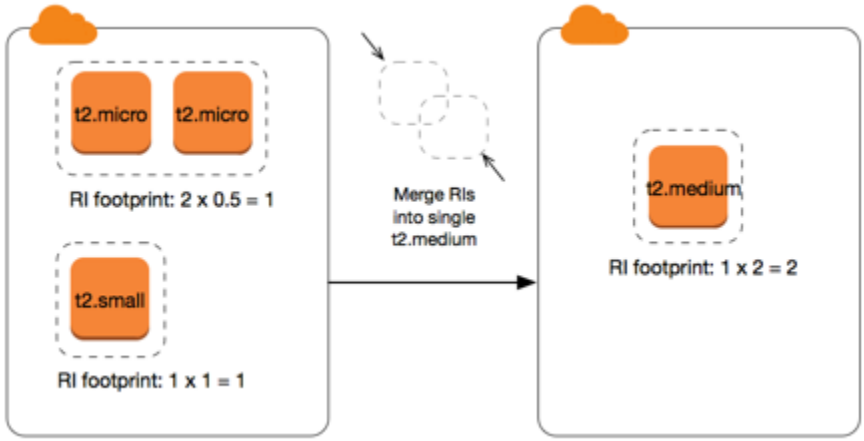
リザーブドインスタンス でインスタンスサイズのフットプリントを計算するにはインスタンスの数に正規化係数を掛けます。Amazon EC2 コンソールでは正規化係数はユニットで測定されます。次の表に、インスタンスファミリーのインスタンスサイズの正規化係数を示します。例えば、`t2.medium` の正規化係数は 2 であるため、4 つの `t2.medium` インスタンスの予約は 8 ユニットのフットプリントになります。

インスタンスサイズ	正規化係数
nano	0.25
micro	0.5
small	1
medium	2
large	4
xlarge	8
2xlarge	16
3xlarge	24
4xlarge	32
6xlarge	48
8xlarge	64
9xlarge	72
10xlarge	80
12xlarge	96
16xlarge	128

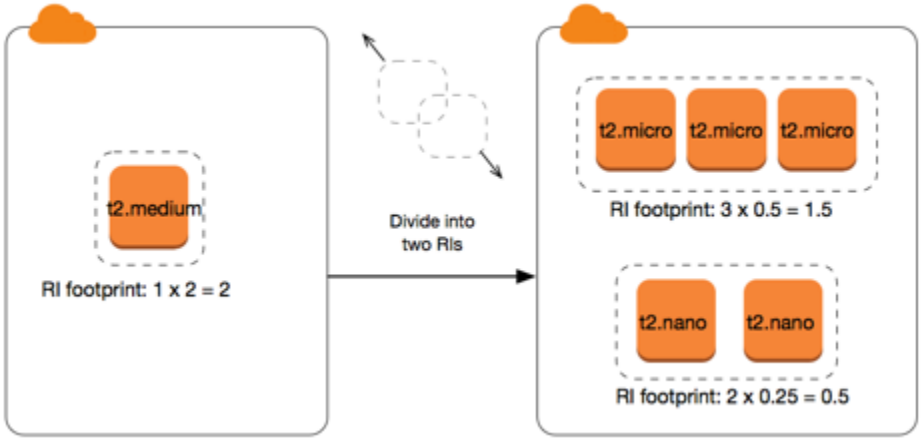
インスタンスサイズ	正規化係数
18xlarge	144
24xlarge	192
32xlarge	256
48xlarge	384
56xlarge	448
96xlarge	768
112xlarge	896

予約のインスタンスサイズのフットプリントが同じである場合は同じインスタンスファミリー内で、予約を異なるインスタンスサイズとして割り当てることができます。例えば、1 つの t2.large (1 @ 4 ユニット) インスタンスの予約は 4 つの t2.small (4 @ 1 ユニット) インスタンスに分割できます。同様に、4 つの t2.small インスタンスの予約は 1 つの t2.large インスタンスにまとめることができます。ただし、2 つの t2.small インスタンスの予約を 1 つの t2.large インスタンスに変更することはできません。新しい予約のフットプリント (4 ユニット) が元の予約のフットプリント (2 ユニット) より大きいためです。

次の例では2 つの t2.micro インスタンスの予約 (1 ユニット) と 1 つの t2.small インスタンスの予約 (1 ユニット) があります。両方の予約を 1 つの t2.medium インスタンスの予約 (2 ユニット) に結合すると、新しい予約のフットプリントと結合後の予約のフットプリントは等しくなります。



また、予約を変更して 2 つ以上の予約に分割することもできます。次の例では1 つの t2.medium インスタンスの予約 (2 ユニット) があります。この予約を 2 つの予約に分割できます。t2.nano インスタンス 2 個の予約 (0.5 ユニット) と、t2.micro インスタンス 3 個の予約 (1.5 ユニット) です。



ベアメタルインスタンスの正規化係数

同じインスタンスファミリー内の他のサイズを使用して、metal インスタンスの予約を変更できます。同様に、同じインスタンスファミリー内の metal サイズを使用して、ベアメタルインスタンス以外のインスタンスの予約を変更できます。通常、ベアメタルインスタンスのサイズは同じインスタンスファミリー内の最大のインスタンスサイズと同じです。例えば、i3.metal インスタンスは i3.16xlarge インスタンスと同じサイズであるため、正規化係数が同じになります。

次の表はベアメタルインスタンスを持つインスタンスファミリーのベアメタルインスタンスサイズの正規化係数を示しています。metal インスタンスの正規化係数は他のインスタンスサイズとは異なり、インスタンスファミリーによって決まります。

インスタンスサイズ	正規化係数
a1.metal	32
m5zn.metal | x2iezn.metal z1d.metal	96
c6g.metal | c6gd.metal | i3.metal | m6g.metal | m6gd.metal | r6g.metal | r6gd.metal | x2gd.metal	128
c5n.metal	144

インスタンスサイズ	正規化係数
c5.metal | c5d.metal | i3en.metal | m5.metal | m5d.metal | m5dn.metal | m5n.metal | r5.metal | r5b.metal | r5d.metal | r5dn.metal | r5n.metal	192
c6i.metal | c6id.metal | m6i.metal | m6id.metal | r6d.metal | r6id.metal	256
u-18tb1.metal | u-24tb1.metal	448
u-6tb1.metal | u-9tb1.metal | u-12tb1.metal	896

例えば、1 つの i3.metal インスタンスには 128 の正規化係数があります。i3.metal デフォルトテナンシー Amazon Linux/Unix リザーブドインスタンス を購入する場合、次のように予約を分割できます。

- i3.16xlarge は i3.metal インスタンスと同じサイズであるため、その正規化係数は 128 (128/1) です。1 つの i3.metal インスタンスの予約は1 つの i3.16xlarge インスタンス内で変更できます。
- i3.8xlarge は i3.metal インスタンスの半分のサイズであるため、その正規化係数は 64 (128/2) です。1 つの i3.metal インスタンスの予約は2 つの i3.8xlarge インスタンスに分割できます。
- i3.4xlarge は i3.metal インスタンスの 4 分の 1 のサイズであるため、その正規化係数は 32 (128/4) です。1 つの i3.metal インスタンスの予約は4 つの i3.4xlarge インスタンスに分割できます。

変更リクエストの送信

リザーブドインスタンスを変更する前に、適用される制限を必ず確認してください。インスタンスのサイズを変更する前に、変更する元の予約の合計インスタンスサイズフットプリントを計算し、その結果が新しい設定の合計インスタンスサイズフットプリントと一致することを確認してください。

Console

リザーブドインスタンスを変更するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

2. [リザーブドインスタンス] ページで、変更する リザーブドインスタンス を 1 つ以上選択し、[アクション]、[リザーブドインスタンスの変更] の順に選択してください。

   リザーブドインスタンス がアクティブ状態ではない場合、または変更できない場合は[リザーブドインスタンス の変更] が無効となります。

3. 変更テーブルの最初のエントリには選択した リザーブドインスタンス の属性と、その下部に少なくとも 1 つのターゲット設定が表示されます。[単位] 列には全インスタンスサイズのフットプリントが表示されます。追加する各新規設定で 追加 を選択してください。必要に応じて各構成の属性を変更します。

   - [スコープ]: 設定の適用先が 1 つのアベイラビリティーゾーンまたはリージョン全体のどちらであるかを選択してください。
   - [アベイラビリティーゾーン]: 必要なアベイラビリティーゾーンを選択してください。リージョン リザーブドインスタンス には適用されません。
   - [インスタンスタイプ]: 必要なインスタンスタイプを選択してください。組み合わせた設定は元の設定のインスタンスサイズのフットプリントと等しくなければなりません。
   - [カウント]: インスタンス数を指定します。リザーブドインスタンス を複数の設定に分割するにはカウントを減らし、[追加] を選択して、追加する設定のカウントを指定します。例えば、カウントが 10 の設定が 1 つある場合、そのカウントを 6 に変更し、カウントが 4 の設定を別に追加できます。このプロセスでは新しい リザーブドインスタンス がアクティブになった後で、元の リザーブドインスタンス を終了させます。

4. [続行] をクリックしてください。
5. ターゲット設定の指定を完了し変更内容を確認したい場合は[変更を送信] を選択してください。
6. 変更リクエストのステータスはリザーブドインスタンス 画面の [状態] 列で確認できます。有効な状態には以下のものがあります。

   - アクティブ (変更の保留) —元の リザーブドインスタンス の移行状態
   - リタイア (変更の保留) — 新しい リザーブドインスタンス を作成中の元の リザーブドインスタンス の移行状態
   - リタイア — リザーブドインスタンス は正常に変更され、置き換えられました
   - アクティブ — 次のいずれかを選択してください。
     - 正常な変更リクエストにより新しい リザーブドインスタンス が作成されました
     - 変更リクエストが失敗したため、元の リザーブドインスタンス です

AWS CLI

リザーブドインスタンスを変更するには

modify-reserved-instances コマンドを使用します。JSON ファイルで設定の詳細を指定できます。

```
aws ec2 modify-reserved-instances \
    --reserved-instances-ids b847fa93-e282-4f55-b59a-1342f5bd7c02 \
    --target-configurations file://configuration.json
```

変更リクエストのステータスを取得するには

describe-reserved-instances-modifications コマンドを使用します。ステータスは、processing、fulfilled、または failed です。

```
aws ec2 describe-reserved-instances-modifications \
    --reserved-instances-modification-ids rimod-d3ed4335-b1d3-4de6-ab31-0f13aaf46687
 \
    --query ReservedInstancesModifications[].Status
```

PowerShell

リザーブドインスタンスを変更するには

Edit-EC2ReservedInstance コマンドレットを使用します。タイプ Amazon.EC2.Model.ReservedInstancesConfiguration のオブジェクトで設定の詳細を指定できます。

```
Edit-EC2ReservedInstance `
    -ReservedInstancesId b847fa93-e282-4f55-b59a-1342f5bd7c02 `
    -TargetConfiguration $configuration
```

変更リクエストのステータスを取得するには

Get-EC2ReservedInstancesModification コマンドレットを使用します。ステータスは、processing、fulfilled、または failed です。

```
Get-EC2ReservedInstancesModification `
    -ReservedInstancesModificationId rimod-d3ed4335-b1d3-4de6-ab31-0f13aaf46687 | `
```

```
    Select Status
```

変更リクエストのトラブルシューティング

リクエストしたターゲット設定が一意であれば、リクエストが処理されるメッセージを受信します。この時点ではAmazon EC2 は変更リクエストのパラメータが有効であることのみを確認しています。まだ、処理中に容量が利用できないために変更リクエストが失敗する可能性があります。

場合によって、確認の代わりに変更リクエストが不完全または失敗したことを示すメッセージが表示されることがあります。メッセージの情報を参考にして、別の変更リクエストを再送信します。リクエストを送信する前に、適用される制約を必ず確認してください。

選択された リザーブドインスタンス に変更できないものがあります

Amazon EC2 は変更できない リザーブドインスタンス を示します。このようなメッセージを受け取ったら、Amazon EC2 コンソールの [リザーブドインスタンス] ページ リザーブドインスタンス についての詳細情報を確認します。

変更リクエストの処理中にエラーが発生しました

送信した リザーブドインスタンス 変更リクエストをすべて処理できません。変更している予約の数によってはメッセージが異なる場合があります。

Amazon EC2 は変更リクエストを処理できない理由を示します。例えば、変更している リザーブドインスタンス の 1 つ以上のサブセットに同じターゲット設定 (アベイラビリティーゾーンとプラットフォームの組み合わせ) を指定したような場合です。予約のインスタンス詳細が一致し、変更対象のすべてのサブセットのターゲット設定が一意であることを確認して、変更リクエストの再送信を試みます。

コンバーティブルリザーブドインスタンス の交換

また、インスタンスファミリー、オペレーティングシステム、およびテナンシーを含む別の構成で、1 つ以上の別の コンバーティブルリザーブドインスタンス の コンバーティブルリザーブドインスタンス に交換することもできます。新しいコンバーティブルリザーブドインスタンスが交換するコンバーティブルリザーブドインスタンスと同等あるいはそれ以上の値である限り、交換の実行回数に制限はありません。

コンバーティブルリザーブドインスタンスを交換する場合、現在の予約のインスタンスの数は新しいコンバーティブルリザーブドインスタンスにおいて、同じ設定値かそれ以上のインスタンス数と交換

されます。Amazon EC2 は交換の結果として受け取ることができるリザーブドインスタンスの数を計算します。

スタンダード リザーブドインスタンス は交換できませんが、変更することはできます。詳細については「リザーブドインスタンス の変更」を参照してください。

内容

- コンバーティブルリザーブドインスタンス 交換の要件
- コンバーティブルリザーブドインスタンス の交換の計算
- コンバーティブルリザーブドインスタンス のマージ
- コンバーティブルリザーブドインスタンス の一部の交換
- 交換リクエストの送信

コンバーティブルリザーブドインスタンス 交換の要件

以下の条件を満たしている場合に、Amazon EC2 では交換リクエストが処理されます。コンバーティブルリザーブドインスタンス は次のとおりである必要があります:

- アクティブ
- 保留中の以前の交換リクエストがないこと
- 有効期限が切れるまで、少なくとも 24 時間残っていること

以下のルールが適用されます。

- コンバーティブルリザーブドインスタンスは、その時点で AWS によって提供されている他のコンバーティブルリザーブドインスタンスに交換する必要があります。
- コンバーティブルリザーブドインスタンス は特定のリージョンと関連付けられ、予約の期間中は固定されます。コンバーティブルリザーブドインスタンスを別のリージョンのコンバーティブルリザーブドインスタンスに交換することはできません。
- ゾーンコンバーティブルリザーブドインスタンスを交換するには、AWS の該当リージョンに新しいインスタンスタイプに対応できる十分な容量が必要です。
- 1 つの コンバーティブルリザーブドインスタンス で 1 つ以上の コンバーティブルリザーブドインスタンス を一度に交換することができます。
- コンバーティブルリザーブドインスタンス の一部を交換するには2 つ以上の予約に変更して、1 つ以上の予約を新しい コンバーティブルリザーブドインスタンス に交換することができます。詳細

については「コンバーティブルリザーブドインスタンス の一部の交換」を参照してください。リザーブドインスタンス の変更の詳細についてはリザーブドインスタンス の変更を参照してください。

- 全額前払い コンバーティブルリザーブドインスタンス は一部前払い コンバーティブルリザーブドインスタンス に交換できます。その逆も可能です。

> **Note**
>
> 交換に必要な前払いの合計 (調整額) が 0.00 USD 未満の場合、その額が 0.00 USD 以上になるだけのコンバーティブルリザーブドインスタンス のインスタンス数が、AWS によって自動的に提供されます。

> **Note**
>
> 新しいコンバーティブルリザーブドインスタンス の合計料金 (前払い料金 + 時間料金 x 残り時間数) が交換前のコンバーティブルリザーブドインスタンスの合計料金未満の場合、交換前のコンバーティブルリザーブドインスタンスの合計料金以上になるように、AWS によって自動的にコンバーティブルリザーブドインスタンスのインスタンス数が提供されます。

- より有利な料金を利用するために、前払いなし コンバーティブルリザーブドインスタンス を全額前払いまたは一部前払い コンバーティブルリザーブドインスタンス に交換できます。
- 全額前払いまたは一部前払いのコンバーティブルリザーブドインスタンスを、前払いなしコンバーティブルリザーブドインスタンスに交換することはできません。
- 前払いなし コンバーティブルリザーブドインスタンス を別の前払いなし コンバーティブルリザーブドインスタンス に交換できます。ただし、新しい コンバーティブルリザーブドインスタンス の時間料金が交換元の コンバーティブルリザーブドインスタンス の時間料金以上である場合に限ります。

> **Note**
>
> 新しいコンバーティブルリザーブドインスタンスの合計料金 (時間料金 x 残り時間数) が交換前のコンバーティブルリザーブドインスタンスの合計料金未満の場合、交換前のコンバーティブルリザーブドインスタンスの合計料金以上になるように、AWS によって自動的にコンバーティブルリザーブドインスタンスのインスタンス数が提供されます。

- 有効期限の異なる複数の コンバーティブルリザーブドインスタンス を交換すると、新しい コンバーティブルリザーブドインスタンス の有効期限は将来の最も遠い日付になります。
- 1 つの コンバーティブルリザーブドインスタンス を交換する場合は新しい コンバーティブルリザーブドインスタンス と同じ期間 (1 年または 3 年) が必要です。異なる期間を持つ複数の コンバーティブルリザーブドインスタンス をマージすると、新しい コンバーティブルリザーブドインスタンス の期間は 3 年となります。詳細については「コンバーティブルリザーブドインスタンスのマージ」を参照してください。
- Amazon EC2 がコンバーティブルリザーブドインスタンスを交換すると、関連する予約が取り消され、終了日が新しいリザベーションに転送されます。交換後、Amazon EC2 は古いリザベーションの終了日と新しいリザベーションの開始日の両方を交換日と同じ日付に設定します。例えば、有効期限のうち 16 か月が残っている 3 年の予約を交換する場合、新しい予約は 16 か月のリザベーションであり、終了日は交換したコンバーティブルリザーブドインスタンスのリザベーションと同じ日付です。

コンバーティブルリザーブドインスタンス の交換の計算

コンバーティブルリザーブドインスタンス の交換は無料です。ただし、前払い額を按分計算した結果、所有していたコンバーティブルリザーブドインスタンスと交換して受け取る新しいコンバーティブルリザーブドインスタンスに差額があれば、その清算額を支払う必要がある場合があります。

それぞれの コンバーティブルリザーブドインスタンス に定価があります。交換元と交換先の コンバーティブルリザーブドインスタンス で定価が比較され、交換の結果として得られるコンバーティブルリザーブドインスタンスの数が決まります。

例えば、定価 35 USD の 1 つの コンバーティブルリザーブドインスタンス を定価 10 USD の新しいインスタンスタイプに交換するとします。

```
$35/$10 = 3.5
```

コンバーティブルリザーブドインスタンス を 10 USD の 3 つの コンバーティブルリザーブドインスタンス に交換できます。半個単位で購入することはできないため、余り分を補うにはコンバーティブルリザーブドインスタンス を追加購入する必要があります。

```
3.5 = 3 whole Convertible Reserved Instances + 1 additional Convertible Reserved
 Instance
```

4 つ目の コンバーティブルリザーブドインスタンス で、終了日が他の 3 つのものと同じものだとすると、一部前払いまたは全額前払いの コンバーティブルリザーブドインスタンス を交換する場合、

その 4 つ目のリザーブドインスタンスの料金を差額として支払います。交換元の コンバーティブルリザーブドインスタンス の前払い額のうち 500 USD が残っており、交換先のコンバーティブルリザーブドインスタンスの料金が按分計算で 600 USD になるとすると、100 USD が請求されます。

```
$600 prorated upfront cost of new reservations - $500 remaining upfront cost of old
 reservations = $100 difference
```

コンバーティブルリザーブドインスタンス のマージ

2 つ以上のコンバーティブルリザーブドインスタンスをマージする場合、新しいコンバーティブルリザーブドインスタンスの期限は元のコンバーティブルリザーブドインスタンスの期限がすべて同じであればその期限、そうでなければ元のコンバーティブルリザーブドインスタンスの最も遅い期限になります。新しい コンバーティブルリザーブドインスタンス の有効期間は将来の最も長い有効期間となります。

例えば、アカウントに以下の コンバーティブルリザーブドインスタンス があるとします。

Reserved Instance ID	用語	有効期限日
aaaa1111	1年	2018-12-31
bbbb2222	1年	2018-07-31
cccc3333	3年	2018-06-30
dddd4444	3年	2019-12-31

- aaaa1111 と bbbb2222 をマージして、それらを 1 年の コンバーティブルリザーブドインスタンス と交換できます。それらを 3 年の コンバーティブルリザーブドインスタンス に交換することはできません。新しい コンバーティブルリザーブドインスタンス 有効期限は 2018-12-31 です。
- bbbb2222 と cccc3333 をマージして、それらを 3 年の コンバーティブルリザーブドインスタンス と交換できます。それらを 1 年の コンバーティブルリザーブドインスタンス に交換することはできません。新しい コンバーティブルリザーブドインスタンス 有効期限は 2018-07-31 です。
- cccc3333 と dddd4444 をマージして、それらを 3 年の コンバーティブルリザーブドインスタンス と交換できます。それらを 1 年の コンバーティブルリザーブドインスタンス に交換することはできません。新しい コンバーティブルリザーブドインスタンス 有効期限は 2019-12-31 です。

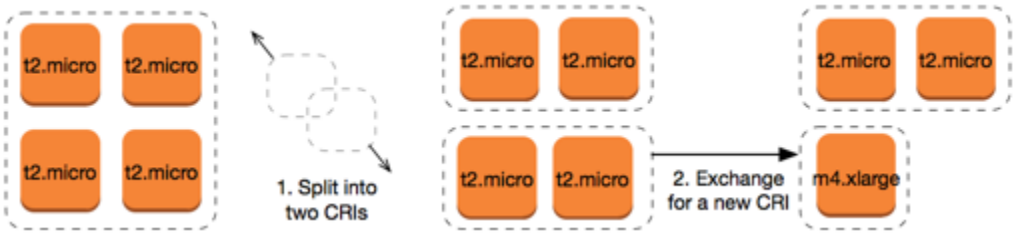
コンバーティブルリザーブドインスタンス の一部の交換

変更プロセスを使用して、コンバーティブルリザーブドインスタンス をより小さい予約に分割し、新しい予約のうちの 1 つ以上を新しい コンバーティブルリザーブドインスタンス と交換することができます。次の例はそれを行う方法を示しています。

Example例: 複数のインスタンスを持つ コンバーティブルリザーブドインスタンス

この例ではこの例では予約に 4 つのインスタンスがある t2.micro コンバーティブルリザーブドインスタンス があります。t2.micro インスタンスに対して 2 つの m4.xlarge インスタンスを交換するには:

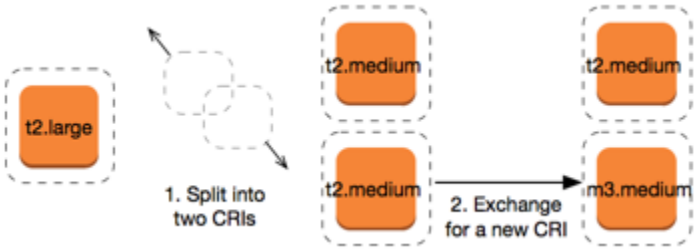
1. t2.micro コンバーティブルリザーブドインスタンス を変更するにはそれぞれ 2 つの t2.micro コンバーティブルリザーブドインスタンス に分割します。
2. 新しい t2.micro コンバーティブルリザーブドインスタンス のいずれかを m4.xlarge コンバーティブルリザーブドインスタンス と交換します。



Example例: 1 つのインスタンスを持つ コンバーティブルリザーブドインスタンス

この例ではt2.large コンバーティブルリザーブドインスタンス があります。小さな t2.medium インスタンスと m3.medium インスタンスに変更するには:

1. t2.large コンバーティブルリザーブドインスタンス を変更するには2 つの t2.medium コンバーティブルリザーブドインスタンス に分割します。1 つの t2.large インスタンスに対して、2 つの t2.medium インスタンスと同じインスタンスサイズのフットプリントが含まれます。
2. 新しい t2.medium コンバーティブルリザーブドインスタンス のいずれかを m3.medium コンバーティブルリザーブドインスタンス と交換します。



詳細については[インスタンスサイズの変更のサポート]および[交換リクエストの送信]を参照してください。

交換リクエストの送信

コンバーティブルリザーブドインスタンスは交換できます。交換されたリザーブドインスタンスは廃止されます。

Console

コンバーティブルリザーブドインスタンスを交換するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [リザーブドインスタンス] を選択し、交換する コンバーティブルリザーブドインスタンス を選び、[アクション]、[リザーブドインスタンス の交換] の順に選択してください。
3. 対象となる設定の属性を選択し、[提供タイプの検索] をクリックしてください。
4. 新しい コンバーティブルリザーブドインスタンス を選択してください。画面の下部に、交換のために受け取った リザーブドインスタンスの番号と追加コストが表示されます。
5. 必要に応じて コンバーティブルリザーブドインスタンス を選択したら、[確認] をクリックしてください。
6. [交換]、[閉じる] の順にクリックしてください。

AWS CLI

コンバーティブルリザーブドインスタンスを交換するには

1. describe-reserved-instances-offerings コマンドを使用して、ニーズを満たす新しいコンバーティブルリザーブドインスタンスを見つけます。
2. get-reserved-instances-exchange-quote コマンドを使用して、交換の見積りを取得します。これには、交換で受け取るリザーブドインスタンスの数と交換時の差額の起案額が含まれています。

3. accept-reserved-instances-exchange-quote コマンドを使用して、交換を実行します。

PowerShell

コンバーティブルリザーブドインスタンスを交換するには

1. Get-EC2ReservedInstancesOffering コマンドレットを使用して、ニーズを満たす新しいコンバーティブルリザーブドインスタンスを見つけます。
2. GetEC2-ReservedInstancesExchangeQuote コマンドレットを使用して、交換の見積りを取得します。これには、交換で受け取るリザーブドインスタンスの数と交換時の差額の起案額が含まれています。
3. Approve-EC2ReservedInstancesExchangeQuote コマンドレットを使用して、交換を実行します

リザーブドインスタンスのクォータ

毎月新しいリザーブドインスタンスを購入できます。毎月購入できる新しいリザーブドインスタンスの数は次のように 1 か月ごとのクォータによって決まります。

クォータの説明	デフォルトのクォータ
新しいリージョンレベルのリザーブドインスタンス	リージョンあたり 20/月
新しいゾーンレベルのリザーブドインスタンス	アベイラビリティーゾーンあたり 20/月

例えば、3 つのアベイラビリティーゾーンがあるリージョンではデフォルトのクォータは 1 か月あたり 80 個の新規リザーブドインスタンスとなります。これは次のように計算されます。

- リージョンの 20 個のリージョンレベルのリザーブドインスタンス
- さらに 60 個のゾーンレベルのリザーブドインスタンス (3 つのアベイラビリティーゾーン用に 20 個ずつ)

`running` 状態のインスタンスはクォータにカウントされます。`pending`、`stopping`、`stopped`、および `hibernated` 状態のインスタンスはクォータにはカウントされません。

購入したリザーブドインスタンスの数を表示する

購入するリザーブドインスタンスの数は[Instance count] (インスタンス数) フィールド (コンソール) または `InstanceCount` パラメータ (AWS CLI) によって示されます。新しいリザーブドインスタンスを購入すると、クォータはインスタンスの総数に照らして測定されます。例えば、インスタンス数が 10 個のリザーブドインスタンス設定を 1 つ購入した場合、その購入はクォータに対して 1 ではなく 10 としてカウントされます。

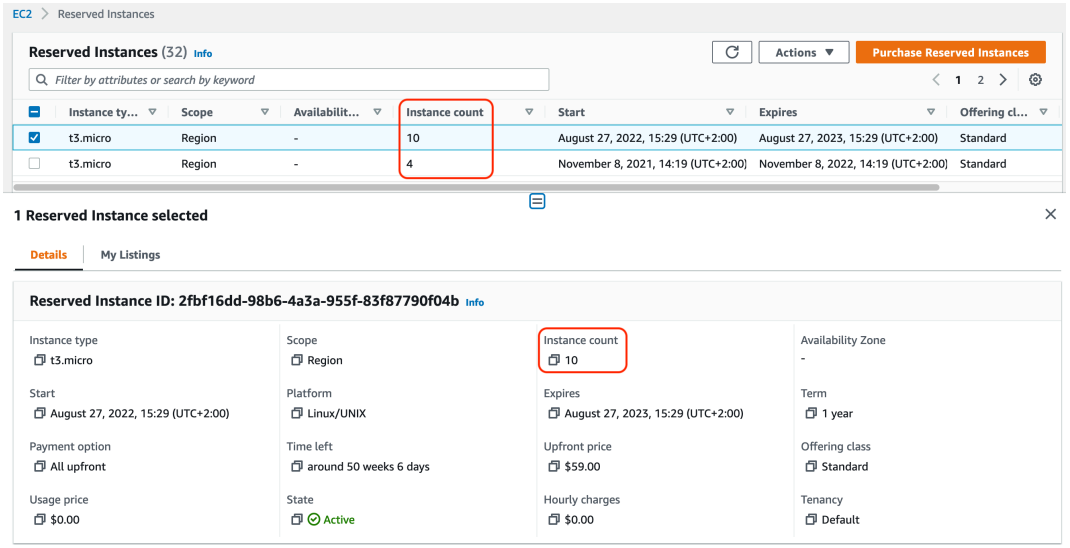
Amazon EC2 または AWS CLI を使用して、購入したリザーブドインスタンスの数を確認できます。

Console

購入したリザーブドインスタンスの数を表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Reserved Instances] を選択してください。
3. テーブルからリザーブドインスタンス設定を選択し、[Instance count] (インスタンス数) フィールドを確認します。

   次のスクリーンショット内の選択されている行は`t3.micro` インスタンスタイプの単一のリザーブドインスタンス設定を表しています。テーブルビューの [Instance count] (インスタンス数) 列と詳細ビューの [Instance count] (インスタンス数) フィールド (スクリーンショット参照) はこの設定に 10 個のリザーブドインスタンスがあることを示しています。



AWS CLI

購入したリザーブドインスタンスの数を表示するには

describe-reserved-instances CLI コマンドを使用して、リザーブドインスタンス設定の ID を指定します。

```
aws ec2 describe-reserved-instances \
    --reserved-instances-ids a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --output table
```

以下は出力の例です。InstanceCount フィールドはこの設定用に 10 個のリザーブドインスタンスがあることを示しています。

```
-------------------------------------------------------------------
|                    DescribeReservedInstances                    |
+-----------------------------------------------------------------+
||                       ReservedInstances                       ||
|+----------------------+----------------------------------------+|
||  CurrencyCode        |  USD                                   ||
||  Duration            |  31536000                              ||
||  End                 |  2023-08-27T13:29:44+00:00             ||
||  FixedPrice          |  59.0                                  ||
||  InstanceCount       |  10                                    ||
||  InstanceTenancy     |  default                               ||
||  InstanceType        |  t3.micro                              ||
||  OfferingClass       |  standard                              ||
||  OfferingType        |  All Upfront                           ||
||  ProductDescription  |  Linux/UNIX                            ||
||  ReservedInstancesId |  a1b2c3d4-5678-90ab-cdef-EXAMPLE11111  ||
||  Scope               |  Region                                ||
||  Start               |  2022-08-27T13:29:45.938000+00:00      ||
||  State               |  active                                ||
||  UsagePrice          |  0.0                                   ||
|+----------------------+----------------------------------------+|
|||                      RecurringCharges                       |||
||+---------------------------------+--------------------------+||
|||  Amount                         |  0.0                     |||
|||  Frequency                      |  Hourly                  |||
||+---------------------------------+--------------------------+||
```

PowerShell

購入したリザーブドインスタンスの数を表示するには

Get-EC2ReservedInstance コマンドレットを使用して、リザーブドインスタンス設定の ID を指定します。

```
Get-EC2ReservedInstance -ReservedInstancesId a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

以下は出力の例です。InstanceCount フィールドはこの設定用に 10 個のリザーブドインスタンスがあることを示しています。

```
AvailabilityZone    :
CurrencyCode        : USD
Duration            : 31536000
End                 : 1/12/2017 8:57:08 PM
FixedPrice          : 0
InstanceCount       : 10
InstanceTenancy     : default
InstanceType        : t3.medium
OfferingClass       : standard
OfferingType        : All Upfront
ProductDescription  : Windows
RecurringCharges    : {}
ReservedInstancesId : a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
Scope               : Region
Start               : 10/12/2016 4:00:00 PM
State               : active
Tags                : {}
UsagePrice          : 0
```

考慮事項

リージョン リザーブドインスタンス ではオンデマンドインスタンス の実行に割引が適用されます。デフォルトの オンデマンドインスタンス の制限は 20 です。リージョン オンデマンドインスタンス を購入すると、リザーブドインスタンス の実行制限を超えることはできません。例えば、すでに オンデマンドインスタンス を 20 回実行していて、20 のリージョン リザーブドインスタンス を購入した場合、20 のリージョン リザーブドインスタンス には 20 回の オンデマンドインスタンス の実行に割引が適用されます。さらに多くのリージョン リザーブドインスタンス を購入した場合はオンデマンドインスタンス の制限に達しているため、さらにインスタンスを起動することはできません。

リージョン リザーブドインスタンス を購入する前に、オンデマンドインスタンス の制限数が所有する予定のリージョン リザーブドインスタンス の数に一致するかそれを超えることを確認してください。必要に応じて、さらにリージョン リザーブドインスタンス を購入する前に オンデマンドインスタンス の制限数の増加を依頼してください。

ゾーンレベルのリザーブドインスタンス (特定のアベイラビリティーゾーンで購入されたリザーブドインスタンス) はキャパシティ予約と割引を提供します。ゾーン リザーブドインスタンス を購入することで、実行中の オンデマンドインスタンス の制限を超えることができます。例えば、すでに 20 の オンデマンドインスタンス を実行していて、20 のゾーン リザーブドインスタンス を購入した場合はゾーン リザーブドインスタンス の仕様に一致する 20 の オンデマンドインスタンス をさらに起動して、合計 40 実行インスタンスを実行できます。

リザーブドインスタンスのクォータを表示してクォータの引き上げをリクエストする

Amazon EC2 コンソールでクォータ情報を確認できます。クォータの引き上げをリクエストすることもできます。詳細については現在の制限を表示するにはおよび引き上げのリクエストを参照してください。

スポットインスタンス

スポットインスタンスは休止中の EC2 キャパシティーを使用するインスタンスで、オンデマンド価格より低料金で利用できます。スポットインスタンスでは未使用の EC2 インスタンスを大幅な割引価格でリクエストできるため、Amazon EC2 のコストを大幅に削減できます スポットインスタンスの時間料金はスポット料金と呼ばれます。各アベイラビリティーゾーンにおける各インスタンスタイプのスポット料金はAmazon EC2 によって設定され、スポットインスタンス の長期的な需給に基づいて徐々に調整されます。スポットインスタンスはキャパシティが利用可能なときに、いつでも実行されます。

スポットインスタンス はアプリケーションを実行する時間に柔軟性がある場合や、アプリケーションを中断できる場合に、費用効率の高い選択肢です。例えば、スポットインスタンス はデータ分析、バッチジョブ、バックグラウンド処理、およびオプションタスクに適しています。詳細についてはAmazon EC2スポットインスタンスを参照してください。

EC2 インスタンスのさまざまな購入オプションの比較については「Amazon EC2 の請求および購入オプション」を参照してください。

概念

スポットインスタンスを使用するときは事前に以下の概念を理解しておく必要があります。

- スポットキャパシティープール – インスタンスタイプ (`m5.large` など) とアベイラビリティーゾーンが同一で、使用されていない EC2 インスタンスのセットです。
- スポット料金 – スポットインスタンスの現在の料金です (時間あたり)。
- スポットインスタンスリクエスト – スポットインスタンスに対するリクエストです。キャパシティーが利用可能になると、Amazon EC2 がリクエストを実行します。スポットインスタンスリ

クエストにはワンタイムと永続の2種類があります。リクエストに関連付けられたスポットインスタンスが中断された後、Amazon EC2 は永続的スポットインスタンスリクエストを自動的に再送信します。

- EC2 インスタンスの再調整に関する推奨事項 – Amazon EC2 は、スポットインスタンスにおいて中断のリスクが高まったことをユーザーに通知するために、インスタンスの再調整に関する推奨事項シグナルを送信します。このシグナルにより、スポットインスタンスで中断 2 分前の通知が発信されていなくても、ユーザーは既存の、または新しいスポットインスタンスについて、前もってワークロードを再調整することができます。
- スポットインスタンスの中断 – Amazon EC2 が容量を戻してもらう必要がある場合にはAmazon EC2 はスポットインスタンスを終了、停止、または休止状態にします。Amazon EC2 はスポットインスタンスが中断される 2 分前に、そのインスタンスに対し中断を警告するための通知を送信します。

スポットインスタンスとオンデマンドインスタンスの違い

次の表はスポットインスタンスとオンデマンドインスタンスの主な違いをまとめたものです。

	Spot Instances	On-Demand Instances
作成時刻	スポットインスタンスリクエストがアクティブであり、利用可能なキャパシティーがある場合に限り即時に起動できます。	手動で起動リクエストを実行し、容量が利用可能である場合に限り、即時に起動できます。
使用可能な容量	利用可能なキャパシティ - がない場合、スポットインスタンスリクエストはキャパシティ - が利用可能になるまで継続して自動的に起動リクエストを実行します。	起動リクエストを行うときに容量が利用可能でない場合は容量不足エラー (ICE) が表示されます。
時間料金	スポットインスタンスの 1 時間単位の使用料金は長期的な需要と供給に基づいて変化します。	オンデマンドインスタンス の時間単位の使用料金は固定です。

	Spot Instances	On-Demand Instances
再調整に関する推奨事項	実行中のスポットインスタンスにおいて中断のリスクが高まった場合に、Amazon EC2 はそのインスタンスに対してシグナルを発します。	お客様はいつオンデマンドインスタンスが中断 (停止、休止、または終了) されるかを決定します。
インスタンスの中断	ユーザーはAmazon EBS-backed スポットインスタンスを停止および開始することができます。さらに、キャパシティーが利用できなくなった場合、Amazon EC2 は個々のスポットインスタンスを中断することができます。	お客様はいつオンデマンドインスタンスが中断 (停止、休止、または終了) されるかを決定します。

料金と削減額

スポットインスタンス はスポット料金で課金されます。これは Amazon EC2 によって設定され、スポットインスタンス の長期供給と需要に基づいて徐々に調整されます。スポットインスタンス はお客様が自らスポットインスタンスを終了するか、容量が使用できなくなるか、スケールイン時に Amazon EC2 Auto Scaling グループのインスタンスが削除されるまで実行されます。

ユーザー または Amazon EC2 が実行中のスポットインスタンスを中断した場合、使用しているオペレーティングシステムおよび中断したユーザーに応じて、使用した秒数または時間数の料金が請求されます (料金が発生しない場合もあります)。詳細については「中断された スポットインスタンス の請求」を参照してください。

スポットインスタンスは Savings Plans の対象外です。Savings Plans をお持ちの場合、スポットインスタンスの使用によって既に取得している割引に対する追加の割引は提供されません。さらに、スポットインスタンスへの支出ではCompute Savings Plans のコミットメントは適用されません。

料金の表示

AWS リージョン およびインスタンスタイプごとに、現在の (5 分ごとに更新される) 最低スポット料金を確認するにはAmazon EC2 スポットインスタンスの料金ページを参照してください。

過去 3 か月間のスポット価格の履歴を表示するにはAmazon EC2 コンソールを使用するか、describe-spot-price-history コマンド () を使用します。詳細については「スポットインスタンスの料金履歴の表示」を参照してください。

AWS アカウント ごとに、個々のアベイラビリティーゾーンがコードにマッピングされます。したがって、アカウント間で同じアベイラビリティーゾーンコード (例えば、us-west-2a) に対して結果が異なる場合があります。

削減額の表示

スポットインスタンスを 1 つのスポットフリートまたはすべてのスポットインスタンスに対して使用することで得られる節約額を確認できます。過去 1 時間または過去 3 日間の削減状況を表示でき、vCPU 時間あたりの平均コストとメモリ (GiB) 時間あたりの平均コストも確認できます。削減額が予想されますが、使用状況に対する請求の調整が含まれていないため、実際の削減額と異なる場合があります。削減額情報の表示の詳細についてはスポットインスタンス 購入による削減額を参照してください。

請求書の表示

請求書にはサービスの使用量に関する詳細が記載されています。詳細についてはAWS Billing ユーザーガイドの請求書の表示を参照してください。

Amazon EC2 スポットのベストプラクティス

Amazon EC2 ではスポットインスタンスを利用して AWS クラウド にある予備の EC2 コンピューティングキャパシティにアクセスし、オンデマンド料金に比べて最大 90% のコストを節約できます。オンデマンドインスタンスとスポットインスタンスの違いはAmazon EC2 がキャパシティを再要求する必要がある場合に、Amazon EC2 がスポットインスタンスを中断できることだけです。中断の際には2 分前に通知があります。スポットインスタンスで快適なエクスペリエンスを得るにはスポットインスタンスを使用するためのベストプラクティスを理解して適用することが重要です。

スポットインスタンス はステートレスかつフォールトトレラントで、柔軟性の高いアプリケーションに適しています。例えば、 スポットインスタンス はビッグデータ、コンテナ化されたワークロード、CI/CD、ステートレスウェブサーバー、ハイパフォーマンスコンピューティング (HPC)、レンダリングワークロードに適しています。

実行中、 スポットインスタンス は オンデマンドインスタンス とまったく同じ動作をします。ただし、スポットはワークロードが完了するまで十分な期間、実行中のインスタンスが動作し続けることを保証するものではありません。また、スポットは必要としているインスタンスをすぐに取得できること、またはリクエストした総容量がいつでも取得できることを保証するものではありません。さらに、スポットインスタンスの可用性は需要と供給によって変化し、将来のパフォーマンスが過去の実績により保証されるものではないため、スポットインスタンスの容量や発生する中断は時間の経過とともに変化する可能性があります。

スポットインスタンス は柔軟性がない、ステートフル、フォールトイントレラント、またはインスタンスノード間で緊密に結合されているワークロードには適していません。また、ターゲットキャパシティが完全に使用できない期間が時折あることが許容されないワークロードにはスポットインスタンスは推奨されません。スポットのベストプラクティスに従ってインスタンスタイプとアベイラビリティーゾーンに柔軟性を持たせることで、高可用性を実現できますが、オンデマンドインスタンスの需要が急増するとスポットインスタンスのワークロードが中断される可能性があるため、容量が使用可能になる保証はありません。

こうしたワークロードにスポットインスタンスを使用したり、中断や停止期間を処理するためにオンデマンドインスタンスへのフェールオーバーを試みたりしないよう、強く勧告します。オンデマンドインスタンスにフェイルオーバーすると、他のスポットインスタンスの中断が誤って発生する可能性があります。さらに、インスタンスタイプとアベイラビリティーゾーンの組み合わせのスポットインスタンスが中断された場合、同じ組み合わせでオンデマンドインスタンスを取得することが困難になる可能性があります。

スポットの使用に慣れている場合でも、スポットインスタンスを使い始めたばかりの場合でも、スポットインスタンスの中断や可用性に関する問題が発生している場合にはスポットサービスを最大限に活用できるよう、これらのベストプラクティスに従うことをお勧めします。

スポットを利用するうえでのベストプラクティス

- 中断に備えて個々のインスタンスを準備する
- インスタンスタイプとアベイラビリティーゾーンについて柔軟に対応する
- 属性ベースのインスタンスタイプの選択を使用する
- スポットプレースメントスコアを使用して最適なリージョンとアベイラビリティーゾーンを特定する
- EC2 Auto Scaling グループまたは EC2 フリートを使用して総容量を管理する
- 価格と容量を最適化する配分戦略を使用する
- 統合された AWS のサービスを使用して スポットインスタンス を管理する
- 使用すべき最適なスポットリクエスト方法はどれですか?

中断に備えて個々のインスタンスを準備する

スポットインスタンスの中断を適切に処理する最善の方法は耐障害性のあるアプリケーションを設計することです。これを実現するためにはEC2 インスタンスの再調整に関する推奨事項、ならびにスポットインスタンスの中断通知を利用できます。

EC2 インスタンスの再調整に関するレコメンデーションはスポットインスタンスで中断のリスクが高まった場合に通知するためのシグナルです。ユーザーはこのシグナルにより、スポットインスタンスの中断 2 分前の通知が届いていない段階で、事前にスポットインスタンスの管理を行えます。ワークロードを、中断のリスクが高くない新規または既存の スポットインスタンス に再調整することができます。このシグナルはAuto Scaling グループと EC2 フリートの容量の再分散機能を使うことで簡単に利用できます。

スポットインスタンスの中断通知はAmazon EC2 がスポットインスタンスを中断する 2 分前に発行される警告です。ワークロードに「時間の面での柔軟性」がある場合は中断が発生した際にそれを終了するのではなく、停止または休止状態になるようにスポットインスタンスを設定することができます。Amazon EC2 は中断が発生したスポットインスタンスを自動的に停止または休止状態にし、使用可能な容量が確保できた際にはそのインスタンスを自動的に再開します。

再調整に関する推奨事項と中断通知をキャプチャし、ワークロードの進行状況のチェックポイントをトリガーするか、中断を適切に処理するルールを Amazon EventBridge で作成することをお勧めします。詳細については再調整に関する推奨事項シグナルのモニタリングを参照してください。イベントルールの作成および使用方法の詳細な例については Taking Advantage of Amazon EC2 スポットインスタンスInterruption Noticesを参照してください。

詳細については EC2 インスタンスの再調整に関する推奨事項および スポットインスタンスの中断。 を参照してください。

インスタンスタイプとアベイラビリティーゾーンについて柔軟に対応する

スポットキャパシティープールは同じインスタンスタイプ (`m5.large` など) とアベイラビリティーゾーン (us-east-1a など) を持つ、未使用の EC2 インスタンスのセットです。どのインスタンスタイプをリクエストし、どのアベイラビリティーゾーンでワークロードをデプロイするか柔軟に対応することで、スポットが必要な量のコンピューティング性能を見つけ、割り当てられる可能性が高くなります。例えば、c4、m5、m4 ファミリのラージを使用してもよいのであれば、 `c5.large` を指定する必要はないということです。

具体的なニーズに応じて、コンピューティング要件を満たすためにどのインスタンスタイプを使用できるか評価できます。ワークロードを垂直にスケーリングできる場合はより大きいインスタンスタイプ (vCPU とメモリが多い) をリクエストに含めてください。水平にしかスケーリングできない場合はオンデマンドの顧客からの需要が少ない、旧世代のインスタンスタイプを含めることをお勧めします。

一般的に、ワークロードごとに少なくとも 10 種類のインスタンスタイプに柔軟に対応できれば十分です。さらに、すべてのアベイラビリティーゾーンが VPC で使用するように設定され、ワークロード用に選択されていることを確認してください。

属性ベースのインスタンスタイプの選択を使用する

属性ベースのインスタンスタイプの選択では実行するワークロードに応じて vCPU、メモリ、ストレージなどのインスタンス属性を指定できます。EC2 Auto Scaling または EC2 フリートが、指定の属性に一致するインスタンスを自動的に特定して起動します。これで、特定のインスタンスタイプを手動で選択する手間が省けます。手動で選択するには各インスタンスタイプが提供するサービスについて深い知識が求められます。

また、属性ベースのインスタンスタイプの選択を使用した場合はインスタンスタイプが新規にリリースされたら自動的に使用できるようになります。これにより、ますます幅広いスポットインスタンスキャパシティにシームレスにアクセスできるようになります。

属性ベースのインスタンスタイプの選択は実行先のインスタンスタイプに柔軟に対応できるワークロードとフレームワークに理想的です。例えば、ハイパフォーマンスコンピューティング (HPC) やビッグデータワークロードなどです。

詳細については「Amazon EC2 Auto Scaling ユーザーガイド」の「Create mixed instances group using attribute-based instance type selection」およびこのガイドの「EC2 フリートまたはスポットフリートのインスタンスタイプを選択するための属性を指定する」を参照してください。

スポットプレースメントスコアを使用して最適なリージョンとアベイラビリティーゾーンを特定する

スポットインスタンスは未使用の EC2 キャパシティであり、このキャパシティは EC2 の需要と供給に基づいて変動します。このため、特定の時間に特定の場所で必要になるスポットキャパシティをいつでも正確に取得できるとは限りません。この予測不能性を軽減するために、スポットプレースメントスコア機能を使用できます。これはスポットキャパシティのニーズを満たせるだけの十分なキャパシティがある可能性が高いリージョンまたはアベイラビリティーゾーンを推奨する機能です。そうしたロケーションで先にスポットインスタンスを起動しておく必要はありません。

スポットプレースメントスコアはインスタンスタイプとそのタイプで使用可能なリージョンやアベイラビリティーゾーンに柔軟に対応できるワークロードで使用するのに最適です。そのために必要なのは必要とするスポットキャパシティ、インスタンスタイプ要件、およびリージョンやアベイラビリティーゾーンの推奨が必要かどうかを指定することだけです。これらを指定すると、リージョンまたはアベイラビリティーゾーンごとに 1 ~ 10 のスコアが返されます。スコアはリクエストしたスポットキャパシティを目的のロケーションに正常にプロビジョニングできる可能性を示します。スコア 10 はスポットリクエストが正常に処理される可能性が非常に高いことを示します。

キャパシティは時間の経過と共に変動するため、スポットプレースメントスコアはあくまでもその時点での推奨であることに留意してください。キャパシティの可用性を保証するものでも、中断のリスクを予測するものでもありません。

スポットプレースメントスコア機能はAmazon EC2 コンソール、AWS CLI、または SDK で使用できます。詳細については「スポットプレイスメントスコア」を参照してください。

EC2 Auto Scaling グループまたは EC2 フリートを使用して総容量を管理する

スポットを使用すると、個々のインスタンスの観点からではなく、総容量 (vCPUs、メモリ、ストレージ、またはネットワークスループットなどの単位) の観点から検討することが可能になります。Auto Scaling グループと EC2 フリートはターゲットキャパシティの起動および維持のために使用できます。これにより、中断されたり手動で終了されたりしたリソースを置き換えるリソースを自動的に要求できます。Auto Scaling グループまたは EC2 フリートを設定する際にはアプリケーションのニーズに基づいてインスタンスタイプとターゲットキャパシティを指定するだけで済みます。詳細についてはAmazon EC2 Auto Scaling ユーザーガイド の Auto Scaling グループおよびこのユーザーガイドの EC2 フリートの作成 をご参照ください。

価格と容量を最適化する配分戦略を使用する

Auto Scaling グループの配分戦略を使えば、予備容量を持つスポットキャパシティープールを手動で探す必要なく、ターゲット容量をプロビジョニングできます。最も安い価格で最も利用性の高いスポットキャパシティープールからインスタンスが自動的にプロビジョニングされる、`price-capacity-optimized` 戦略を使用することをお勧めします。また、EC2 フリートの `price-capacity-optimized` 配分戦略も活用できます。最適な容量を持つプールからスポットインスタンス容量が供給されるため、使用しているスポットインスタンスが再要求される可能性は低くなります。配分戦略の詳細についてはこのユーザーガイドのワークロードの中断コストが高い場合およびAmazon EC2 Auto Scaling ユーザーガイドのスポットインスタンスを参照してください。

統合された AWS のサービスを使用して スポットインスタンス を管理する

他の AWS のサービスは個々のインスタンスやフリートを管理する必要なく、全体的なコンピューティングコストを削減できるよう、スポットと統合されています。該当するワークロードの場合、Amazon EMR、Amazon Elastic Container Service、AWS Batch、Amazon Elastic Kubernetes Service、Amazon SageMaker AI、AWS Elastic Beanstalk、Amazon GameLift Servers の各ソリューションを検討することをお勧めします。これらのサービスでのスポットベストプラクティスの詳細についてはAmazon EC2 スポットインスタンス Workshops Website を参照してください。

使用すべき最適なスポットリクエスト方法はどれですか?

次の表を使用して、スポットインスタンスをリクエストする際に使用する API を決定します。

API	どのようなときに使うか?	ユースケース	この API を使うべきか?
CreateAutoScalingGroup	• 単一の構成または混合の構成を持つ、複数のインスタンスが必要です。 • 構成可能な API を使用してライフサイクル管理を自動化するのがよいでしょう。	必要な数のインスタンスを維持しながら、インスタンスのライフサイクルを管理する Auto Scaling グループを作成します。指定した最小値と最大限度の間の水平スケーリング (インスタンスの追加) をサポートします。	はい
CreateFleet	• 単一の構成または混合の構成を持つ、複数のインスタンスが必要です。 • インスタンスのライフサイクルを自己管理するのがよいでしょう。 • 自動スケーリングが必要ない場合は`instant` タイプフリートの使用をお勧めします。	インスタンスタイプ別、AMI 別、アベイラビリティーゾーン別、またはサブネット別で異なる、複数の起動条件を指定し、オンデマンドインスタンスとスポットインスタンス両方のフリートを 1 回のリクエストで作成します。スポットインスタンスの割り当てストラテジーのデフォルトはユニットあたりの `lowest-price` ですが、`price-capacity-optimized` 、`capacity-optimized` または	はい - 自動スケーリングを必要としない場合は `instant` モード。

API	どのようなときに使うか?	ユースケース	この API を使うべきか?
		`diversified` に変更可能です。	
RunInstances	• 既に RunInstances API を使用してオンデマンドインスタンスを起動しているので、単一のパラメータを変更することで、スポットインスタンスの起動に変更するとよいでしょう。 • 異なるインスタンスタイプを持つ複数のインスタンスは必要ありません。	AMI と 1 つのインスタンスタイプを使用して、指定した数のインスタンスを起動します。	いいえ - RunInstances は1 回のリクエストで複数のインスタンスタイプを許可しないため。

API	どのようなときに使うか?	ユースケース	この API を使うべきか?
RequestSpotFleet	• RequestSpotFleet API は計画投資のないレガシー API であるため、使用はお勧めしません。 • インスタンスのライフサイクルを管理する場合はCreateFleet API を使用します。 • インスタンスのライフサイクルを管理したくない場合はCreateAutoScalingGroup API を使用します。	使用しません。RequestSpotFleet は計画投資のないレガシー API です。	いいえ
RequestSpotInstances	• RequestSpotInstances API は計画投資のないレガシー API であるため、使用はお勧めしません。	使用しません。RequestSpotInstances は計画投資のないレガシー API です。	いいえ

スポットインスタンス のしくみ

スポットインスタンスを起動するにはユーザーがスポットインスタンスリクエストを作成します。またはAmazon EC2 が自動的にスポットインスタンスリクエストを作成することもできます。スポットインスタンスはスポットインスタンスリクエストが受理されると起動します。

スポットインスタンスはいくつかの異なるサービスを使用して起動できます。詳細についてはAmazon EC2 スポットインスタンスの開始方法を参照してください。このユーザーガイドではEC2 を使用してスポットインスタンスを起動する方法について説明します。

- スポットインスタンスリクエストはAmazon EC2 コンソールの インスタンス起動ウィザード または run-instances コマンドを使用して作成できます。詳細については「スポットインスタンスを管理する」を参照してください。
- EC2 フリートを作成して、必要な数のスポットインスタンスを指定することができます。Amazon EC2 はEC2 フリートで指定されているすべてのスポットインスタンスについて、ユーザーに代わってスポットインスタンスリクエストを作成します。詳細についてはEC2 フリートの作成を参照してください。
- スポットフリートリクエストを作成し、必要な数のスポットインスタンスを指定することができます。Amazon EC2 はスポットフリートリクエストで指定されたスポットインスタンスごとに、ユーザーに代わってスポットインスタンスリクエストを作成します。詳細については、「スポットフリートを作成する」を参照してください。

> **Note**
>
> スポットフリートは、計画された投資なしでレガシー API を使用します。代わりに EC2 フリートまたは Auto Scaling グループを使用することをお勧めします。詳細については、「フリートにどの方法を使用するのが最適か」を参照してください。

空きキャパシティがある場合、スポットインスタンスが起動します。スポットインスタンスはユーザーにより停止または終了されるか、Amazon EC2 により中断 (スポットインスタンスの中断と呼ばれます) されるまで実行されます。Amazon EC2 はスポットインスタンスを中断する際に、そのインスタンスを停止、終了、または休止状態にすることができます。

スポットインスタンスを使用する場合には中断に備えておく必要があります。スポットインスタンスの需要が増加した場合や、スポットインスタンスの供給が減少した場合、Amazon EC2 がスポットインスタンスを中断する可能性があります。Amazon EC2 によりスポットインスタンスが中断される際にはスポットインスタンスの中断通知が送信されます。それによりインスタンスに対し

て、Amazon EC2 による中断が発生する 2 分前の警告が提供されます。スポットインスタンス の削除保護を有効にすることはできません。詳細については「スポットインスタンスの中断。」を参照してください。

内容

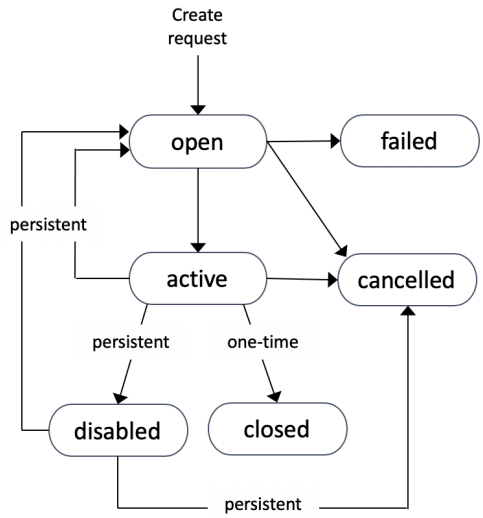
- スポットインスタンスリクエストの状態
- 起動グループでの スポットインスタンス の起動
- アベイラビリティーゾーングループでの スポットインスタンス の起動
- VPC での スポットインスタンス の起動
- バーストパフォーマンスインスタンスを起動する
- シングルテナントハードウェアで起動する

スポットインスタンスリクエストの状態

スポットインスタンスリクエストは次に示すいずれかの状態を取ります。

- `open` – リクエストは受理されるまで待機状態です。
- `active` – リクエストは受理されており、関連付けられたスポットインスタンスが存在します。
- `failed` – リクエストの 1 つ以上のパラメータが正しくありません。
- `closed` – スポットインスタンスは中断または終了されました。
- `disabled` – スポットインスタンスがユーザーにより停止されました。
- `cancelled` – このリクエストはユーザーによりキャンセルされたか、リクエストの有効期限が切れました。

次の図は リクエストの状態の遷移を示しています。遷移はリクエストのタイプ (ワンタイムまたは永続) によって異なります。



ワンタイムスポットインスタンスリクエストはAmazon EC2 がスポットインスタンスを起動するか、リクエストの有効期限が切れるか、またはユーザーがリクエストをキャンセルするまでアクティブ状態を維持します。利用できるキャパシティがない場合、スポットインスタンスは終了し、スポットインスタンスのリクエストは終了します。

永続スポットインスタンスリクエストはリクエストが受理された後も、リクエストの有効期限が切れるかユーザーによりキャンセルされるまで、アクティブ状態を維持します。キャパシティを利用できない場合はスポットインスタンスが中断されます。インスタンスが中断された後に、キャパシティが再び利用可能になると、スポットインスタンスが開始 (停止している場合)、あるいは再開 (休止状態の場合) されます。スポットインスタンスは停止して、キャパシティを利用できるようになったとき再び開始することができます。スポットインスタンスが (停止状態にあるか実行状態にあるかに関係なく) 終了した場合にはスポットインスタンスリクエストが再び開かれ、Amazon EC2 により新しいスポットインスタンスが起動されます。詳細についてはスポットインスタンスを停止する、スポットインスタンスを開始する、およびスポットインスタンスを終了するを参照してください。

スポットインスタンスリクエストの状態と、起動済みのスポットインスタンスのステータスを追跡することができます。詳細については「スポットインスタンスリクエストのステータスを取得する」を参照してください。

起動グループでの スポットインスタンス の起動

スポットインスタンスリクエストで起動グループを指定することによって、一連のスポットインスタンスのすべてが起動可能な場合にのみ、それらを起動するよう、Amazon EC2 に指示することができます。また、スポットサービスで、起動グループ内のインスタンスのいずれかを終了する必要がある場合、すべてのインスタンスを終了することが必要となります。ただし、お客様が起動グループ内

の 1 つ以上のインスタンスを終了する場合、Amazon EC2 は起動グループ内のその他のインスタンスを終了しません。

このオプションは便利な場合もありますが、この制約を追加することによって、スポットインスタンスリクエストが受理される可能性は低くなるので、スポットインスタンスが終了される可能性が高まります。例えば、起動グループに複数のアベイラビリティーゾーンのインスタンスが含まれるとします。これらのアベイラビリティーゾーンのいずれかのキャパシティーが減少して使用できなくなった場合、Amazon EC2 は起動グループのすべてのインスタンスを終了します。

以前に成功したリクエストと同じ (既存の) 起動グループを指定することで、新たに正常なスポットインスタンスリクエストを作成する場合には新しいインスタンスがこの起動グループに追加されます。したがって、この起動グループ内のインスタンスが終了されると、起動グループ内のすべてのインスタンスが終了します。これには最初のリクエストと 2 番目リクエストによって起動されたすべてのインスタンスが含まれます。

アベイラビリティーゾーングループでの スポットインスタンス の起動

スポットインスタンスリクエストでアベイラビリティーゾーングループを指定し、そのアベイラビリティーゾーン内で一連のスポットインスタンスを起動するように Amazon EC2 に指示します。Amazon EC2 はアベイラビリティーゾーングループのすべてのインスタンスを同時に中断する必要はありません。Amazon EC2 がアベイラビリティーゾーングループ内のいずれかのインスタンスを中断する場合、他のインスタンスはそのまま実行されます。

このオプションは便利な場合もありますが、この制約を追加することによって、スポットインスタンスリクエストが受理される可能性は低くなります。

アベイラビリティーゾーングループを指定したものの、スポットインスタンスリクエストでアベイラビリティーゾーンを指定していない場合の結果は使用するネットワークによって異なります。

デフォルト VPC

Amazon EC2 は指定されたサブネットのアベイラビリティーゾーンを使用します。サブネットを指定しなかった場合はアベイラビリティーゾーンとそのデフォルトのサブネットが選択されますが、最低価格のゾーンではない可能性があります。アベイラビリティーゾーンのデフォルトのサブネットを削除した場合は別のサブネットを指定する必要があります。

デフォルトではない VPC

Amazon EC2 は指定されたサブネットのアベイラビリティーゾーンを使用します。

VPC での スポットインスタンス の起動

スポットインスタンス のサブネットを指定するのと同じ方法で、オンデマンドインスタンス のサブネットを指定します。

- [デフォルトの VPC] 特定の低価格のアベイラビリティーゾーンでスポットインスタンスを起動したい場合には対応するサブネットをスポットインスタンスリクエスト内で指定する必要があります。サブネットを指定しなかった場合、Amazon EC2 によってサブネットが選択されますが、このサブネットのアベイラビリティーゾーンのスポット料金は最低ではない可能性があります。
- [デフォルト以外の VPC] スポットインスタンスのサブネットを指定する必要があります。

バーストパフォーマンスインスタンスを起動する

T インスタンスタイプはバーストパフォーマンスインスタンスです。バーストパフォーマンスインスタンスタイプを使用してスポットインスタンスを起動し、CPU クレジットを蓄積するアイドル時間なしでバーストパフォーマンススポットインスタンスをすぐに短時間使用する場合は支払いコストが高くなるのを避けるために、インスタンスを標準モードで起動することをお勧めします。バーストパフォーマンス スポットインスタンス を Unlimited モード で起動し、すぐに CPU をバーストさせると、余分なクレジットがバーストに消費されます。インスタンスを短時間使用する場合、インスタンスは余分なクレジットに見合うだけの CPU クレジットを蓄積する時間がないため、インスタンスの終了時に余分なクレジットに対して課金されます。

Unlimited モードがバーストパフォーマンス スポットインスタンス に適しているのはバースト用の CPU クレジットが蓄積されるまで、そのインスタンスが十分に長く実行される場合のみです。それ以外の場合は余分なクレジットを支払う必要があるため、バーストパフォーマンス スポットインスタンス は他のインスタンスよりも、使用コストが高くなります。詳細については「Unlimited モードと固定 CPU を使用する場合」を参照してください。

T2 インスタンスは標準モードで設定すると、起動クレジットを取得します。T2 インスタンスは起動クレジットを取得できる唯一のバーストパフォーマンスインスタンスです。起動クレジットはインスタンスを構成するために十分なコンピューティングリソースを提供し、T2 インスタンスの初期起動を効率的に実現することを意図しています。T2 インスタンスの起動を繰り返して新しい起動クレジットにアクセスすることは許可されていません。CPU が持続的に必要な場合、(一定期間のアイドリングにより) クレジットを獲得して T2 スポットインスタンス の Unlimited モード を使用するか、専用 CPU を搭載したインスタンスタイプを使用します。

シングルテナントハードウェアで起動する

スポットインスタンスはシングルテナントのハードウェア上で実行できます。ハードウェア専有 スポットインスタンス は他の AWS アカウントに属するインスタンスからは物理的に分離されます。詳細については「Amazon EC2 ハードウェア専有インスタンス」および「Amazon EC2 ハードウェア専有インスタンス」を参照してください。

ハードウェア専有スポットインスタンスを使用するには次のいずれかを実行します。

- スポットインスタンスリクエストを作成する際に、`dedicated` のテナンシーを指定します。詳細についてはスポットインスタンスを管理するを参照してください。
- `dedicated` のインスタンステナンシーを持つ VPC 内で、スポットインスタンスをリクエストします。詳細については「デフォルトのテナンシーで VPC に専有インスタンスを起動する」を参照してください。インスタンステナンシーが `default` である VPC ではテナンシーが `dedicated` であるスポットインスタンスをリクエストすることはできません。

T インスタンスを除くすべてのインスタンスファミリーが、Dedicated スポットインスタンス をサポートしています。対象となるインスタンスファミリーにおいて、最大のインスタンスサイズまたはメタルサイズのみが、Dedicated スポットインスタンス をサポートします。

スポットインスタンスの料金履歴の表示

スポットインスタンス料金は Amazon EC2 で設定され、スポットインスタンス容量に対する長期の需給傾向に基づいて緩やかに調整されます。

スポットリクエストが受理されると、オンデマンド料金を超えない現在のスポット料金で、スポットインスタンス が起動されます。インスタンスタイプ、オペレーティングシステム、アベイラビリティーゾーンでフィルタリングして、過去 90 日間のスポット料金履歴を表示できます。

最新のスポットインスタンス料金についてはAmazon EC2 スポットインスタンスの料金を参照してください。

Console

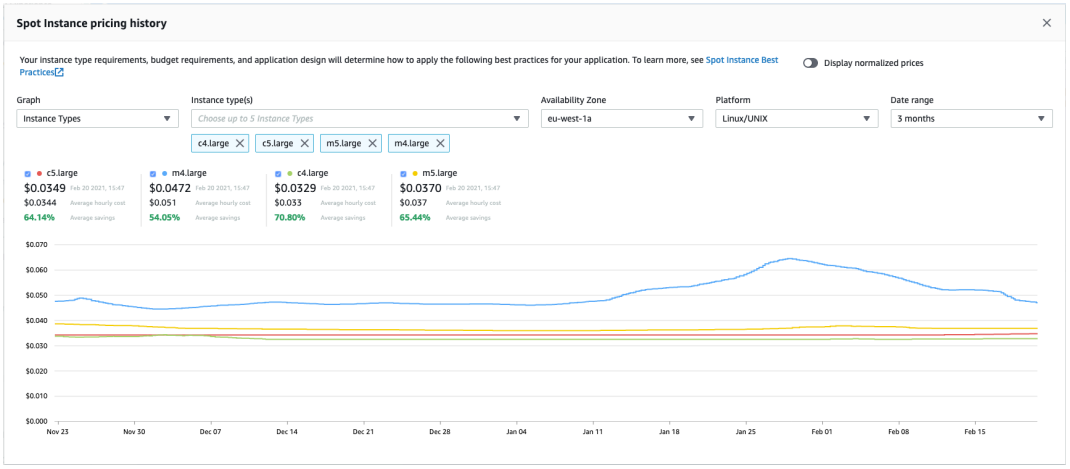
スポット料金の履歴を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. [料金設定履歴] を選択してください。

4. [グラフ] で、料金履歴を[アベイラビリティーゾーン] 別に比較するか、または[インスタンスタイプ] 別に比較するかを選択してください。

   - [アベイラビリティーゾーン] を選択した場合は、料金履歴を表示する [インスタンスタイプ]、オペレーティングシステム ([プラットフォーム])、および [日付範囲] を指定します。
   - [インスタンスタイプ] を選択した場合は、料金履歴を表示する [インスタンスタイプ] (最大 5 つ)、[アベイラビリティーゾーン]、オペレーティングシステム ([プラットフォーム])、および [日付範囲] を指定します。

   次のスクリーンショットは異なるインスタンスタイプでの料金比較を示しています。



5. マウスのカーソル (ポインタ) をグラフ上に移動させると、選択した日付範囲の特定の時刻の料金が表示されます。料金はグラフの上にある情報ブロックに表示されます。一番上の行に表示される料金は特定の日付の料金を示します。2 行目に表示される料金は選択した日付範囲での平均料金です。
6. vCPU あたりの料金を表示するには[正規化された料金を表示] をオンにします。インスタンスタイプの料金を表示するには[正規化された料金を表示] をオフにします。

AWS CLI

スポット料金の履歴を表示するには

次の describe-spot-price-history コマンドを使用します。

```
aws ec2 describe-spot-price-history \
    --instance-types c6i.xlarge \
    --product-descriptions "Linux/UNIX" \
    --start-time 2025-04-01T00:00:00 \
```

```
    --end-time 2025-04-02T00:00:0
```

PowerShell

スポット料金の履歴を表示するには

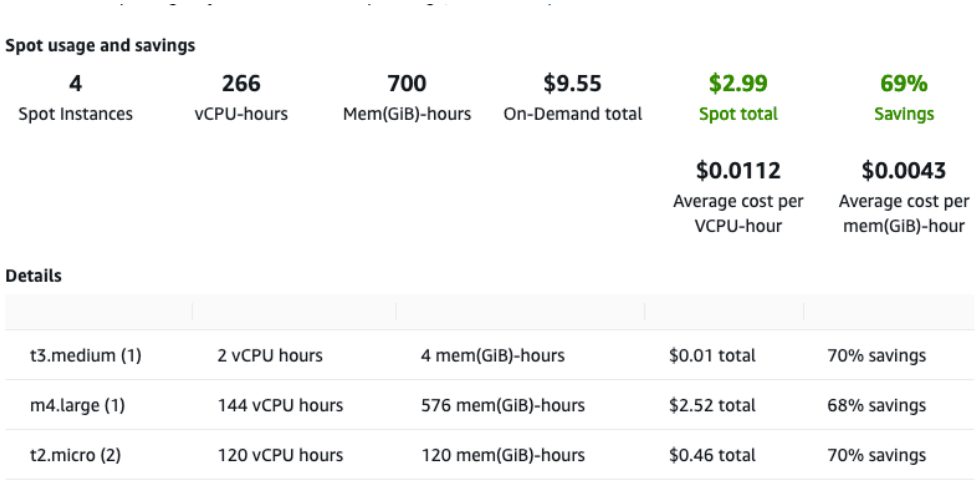
Get-EC2SpotPriceHistory コマンドレットを使用します。

```
Get-EC2SpotPriceHistory `
    -InstanceType c6i.xlarge `
    -ProductDescription "Linux/UNIX" `
    -UtcStartTime 2025-04-01T00:00:00 `
    -UtcEndTime 2025-04-02T00:00:0
```

スポットインスタンス 購入による削減額

フリートあたりレベルの スポットインスタンス またはすべての実行中の スポットインスタンス に関する使用状況と削減額の情報を表示できます。フリートあたりのレベルでは使用状況と削減額の情報にフリートが起動および終了するすべてのインスタンスが含まれます。この情報は過去 1 時間または過去 3 日間から表示できます。

次の [削減額] セクションのスクリーンショットではスポットフリートでのスポットの使用状況、ならびに削減額の情報を示しています。



表示できる使用状況と削減額の情報は次のとおりです。

- スポットインスタンス – スポットフリートによって起動および終了されたスポットインスタンスの数。削減額の要約を表示した場合、その数字は実行中のすべての スポットインスタンス を表します。
- vCPU-hours – 選択した時間枠ですべての スポットインスタンス で使用される vCPU 時間数。
- Mem(GiB)-hours – 選択した時間枠ですべての スポットインスタンス で使用される GiB 時間数。
- On-Demand total – これらのインスタンスを オンデマンドインスタンス として起動した場合、選択した時間枠で支払った合計金額。
- Spot total – 選択した時間枠で支払う合計金額。
- Savings – オンデマンド価格を支払わないことで節約される割合。
- Average cost per vCPU-hour – 選択した時間枠ですべての スポットインスタンス で vCPU を使用する 1 時間あたりの平均コスト。次の式で計算されます: Average cost per vCPU-hour = Spot total / vCPU-hours
- Average cost per mem(GiB)-hour – 選択した時間枠ですべての スポットインスタンス で GiB を使用する 1 時間あたりの平均コスト。次の式で計算されます: Average cost per mem(GiB)-hour = Spot total / Mem(GiB)-hours
- 詳細 テーブル – スポットフリートを構成するさまざまなインスタンスタイプ (括弧内はインスタンスタイプあたりのインスタンス数です)。削減額の要約を表示した場合、その数字は実行中のすべての スポットインスタンス から成ります。

削減額情報はAmazon EC2 コンソールからのみ表示できます。

> **Note**
>
> スポットフリートは、計画された投資なしでレガシー API を使用します。代わりに EC2 フリートまたは Auto Scaling グループを使用することをお勧めします。詳細については、「フリートにどの方法を使用するのが最適か」を参照してください。

スポットフリートの割引情報を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. スポットフリートリクエストの ID を選択し、[削減額] セクションまでスクロールします。

またはスポットフリートリクエスト ID の横にあるチェックボックスをオンにし、[削減] タブを選択してください。

4. デフォルトでは過去 3 日間の使用状況と削減額の情報が表示されます。[last hour] または [last three days] を選択できます。1 時間未満前に起動された スポットフリート の場合はその時間の削減見込み額が表示されます。

実行中のすべてのスポットインスタンスの削減額情報を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. [削減の概要] をクリックしてください。

スポットインスタンスリクエストを作成する

スポットインスタンスを使用するには希望するインスタンス数、インスタンスタイプ、アベイラビリティーゾーンを含む、スポットインスタンスリクエストを作成します。キャパシティが利用可能になると、Amazon EC2 がすぐにリクエストを受理します。それ以外の場合、Amazon EC2 はリクエストが受理できるようになるか、お客様がリクエストをキャンセルするまで待機します。

オンデマンドインスタンスを起動するのと同じ方法で、Amazon EC2 コンソールの インスタンス起動ウィザード または run-instances マンドを使用してスポットインスタンスをリクエストできます。このメソッドは以下の理由でのみ推奨されます。

- すでに インスタンスの起動ウィザード または run-instances コマンドを使用してオンデマンドインスタンスを起動しており、単一のパラメータを変更することでスポットインスタンスの起動に変更したいだけです。
- 異なるインスタンスタイプを持つ複数のインスタンスは必要ありません。

複数のインスタンスタイプを指定することはできず、同じリクエストでスポットインスタンスとオンデマンドインスタンスを起動することはできないため、このメソッドは通常、スポットインスタンスの起動にはお勧めしません。複数のインスタンスタイプを持つスポットインスタンスとオンデマンドインスタンスを含む フリート の起動を含む、スポットインスタンスを起動するための推奨される方法については「使用すべき最適なスポットリクエスト方法はどれですか?」を参照してください。

一度に複数のスポットインスタンスをリクエストした場合、 Amazon EC2 により個別のスポットインスタンスに対するリクエストが作成されるので、各リクエストのステータスを単独で追跡すること

が可能です。スポットインスタンスリクエストの追跡については[スポットインスタンスリクエストのステータスを取得する]を参照してください。

Console

スポットインスタンスリクエストを作成する

ステップ 1～9 はオンデマンドインスタンスの起動に使用するステップと同じです。ステップ 10 で、スポットインスタンスリクエストを設定します。

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 画面上部のナビゲーションバーで、リージョンを選択してください。
3. Amazon EC2 コンソールダッシュボードで、[インスタンスを起動] を選択してください。
4. (オプション) [Name and tags] (名前とタグ) で、インスタンスに名前を付け、スポットインスタンス要求、インスタンス、ボリューム、および Elastic Graphics にタグを付けることができます。タグの詳細については[Amazon EC2 リソースのタグ付け]を参照してください。

   a. [Name] (名前) に、インスタンスのわかりやすい名前を入力してください。

      インスタンス名はタグで、キーは [Name] (名前)、値は指定した名前です。名前を指定しない場合はインスタンスをその ID で識別できます。ID はインスタンスの起動時に自動的に生成されます。

   b. スポットインスタンスリクエスト、インスタンス、ボリューム、および Elastic Graphics にタグを付けするには[Add additional tags] (タグを追加) を選択してください。[Add tag] (タグを追加) を選択し、キーと値を入力し、タグ付けするリソースタイプを選択してください。追加するタグごとに [Add tag] (タグの追加) を選択してください。

5. [Application and OS Images (Amazon マシンイメージ)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で、インスタンスのオペレーティングシステム (OS) を選択してから、AMI を選択してください。詳細については「[アプリケーションと OS イメージ (Amazon マシンイメージ)]」を参照してください。
6. [Instance type] (インスタンスタイプ) で、インスタンスのハードウェア設定とサイズの要件を満たすインスタンスタイプを選択してください。詳細については「[インスタンスタイプ]」を参照してください。
7. [Key pair (login)] (キーペア (ログイン)) で、既存のキーペアを選択するか、[Create new key pair] (新しいキーペアを作成) を選択して新しいキーペアを作成します。詳細については「[Amazon EC2 のキーペアと Amazon EC2 インスタンス]」を参照してください。

> **Important**
>
> [Proceed without key pair] (キーペアなしで進む) オプションを選択した場合 (非推奨)、ユーザーが別の方法でログインすることを許可するように設定された AMI を選択した場合でなければ、インスタンスに接続できなくなります。

8. [Network settings] (ネットワーク設定) で、デフォルト設定を使用するか、[Edit] (編集) を選択して必要に応じてネットワーク設定を構成します。

   セキュリティグループはネットワーク設定の一部を形成し、インスタンスのファイアウォールルールを定義します。このルールではどの着信ネットワークトラフィックをインスタンスに配信するかを指定します。

   詳細については「ネットワーク設定」を参照してください。

9. 選択した AMI にはルートデバイスボリュームを含む、1 つまたは複数のストレージボリュームが含まれます。[Configure storage] (ストレージの設定) で、[Add new volume] (新しいボリュームの追加) を選択して、インスタンスに接続する追加のボリュームを指定できます。詳細については「ストレージの設定」を参照してください。

10. [Advanced details] (高度な設定) で、スポットインスタンスリクエストを次のように設定します。

    a. [購入オプション] で、[スポットインスタンスのリクエスト] チェックボックスをオンにします。

    b. スポットインスタンスリクエストのデフォルト設定を維持するか、[Customize] (カスタマイズ) (右側) を選択して、スポットインスタンスリクエストのカスタム設定を指定できます。

       [Customize] (カスタマイズ) を選択すると、次のフィールドが表示されます。

       i. [Maximum price] (最大価格): スポット価格でスポットインスタンスをリクエストするか、オンデマンド価格を上限とするか、支払う金額の最大額を指定できます。

          > **Warning**
          >
          > 最大料金を指定すると、[No maximum price] (最大料金なし) を選択した場合よりもインスタンスが頻繁に中断されます。

上限の料金を指定する場合は、0.001 USD 以上にする必要があります。0.001 USD 未満の値を指定すると、起動が失敗します。

- [No maximum price] (最大価格なし): スポットインスタンスは現在のスポット価格で起動します。価格はオンデマンド価格を超えることはありません。(推奨)
- [Set your maximum price (per instance/hour)] (最大価格を設定 (インスタンス / 時間あたり): 支払う意思のある最大金額を指定できます。
  - 現在のスポット価格よりも低い最大価格を指定すると、スポットインスタンスは起動しません。
  - 現在のスポット料金よりも高い最大料金を指定すると、スポットインスタンスが起動し、現在のスポット料金で請求されます。スポットインスタンスの実行後、スポット価格が最大価格を超えると、Amazon EC2 がスポットインスタンスを中断します。
  - 指定した上限料金にかかわらず、常に現在のスポット料金が請求されます。

  スポット料金の傾向を確認するにはスポットインスタンスの料金履歴の表示を参照してください。

ii. [Request type] (リクエストタイプ): 選択したスポットインスタンスリクエストタイプによって、スポットインスタンスが中断された場合に何が発生するかが決まります。

- [One-time] (ワンタイム): Amazon EC2 はスポットインスタンスに対して 1 回限りのリクエストを送信します。スポットインスタンスが中断された場合、リクエストは再送信されません。
- [Persistent request] (永続リクエスト): Amazon EC2 はスポットインスタンスに対して永続リクエストを送信します。スポットインスタンスが中断された場合、要求は再送信され、中断されたスポットインスタンスを補充します。

値を指定しない場合、デフォルトは1回限りのリクエストです。

iii. [有効期間終了]: 永続的なスポットインスタンスリクエストの有効期限。

このフィールドは1 回限りのリクエストではサポートされていません。ワンタイムリクエストは、リクエストのすべてのインスタンスが起動するか、ユーザーがリクエストをキャンセルするまで有効です。

- [No request expiry date] (リクエストの有効期限なし): リクエストはキャンセルされるまで有効です。
- [Set your request expiry date] (リクエストの有効期限を設定する): 永続的なリクエストは指定した日付まで、またはキャンセルするまで有効です。

iv. [Interruption behavior] (中断動作): 選択した動作によって、スポットインスタンスが中断されたときに何が起こるかが決まります。

- 永続的なリクエストの場合、有効な値は [Stop] (停止) と [Hibernate] (休止) です。インスタンスが停止すると、EBS ボリュームストレージの料金が適用されます。

> **Note**
>
> スポットインスタンスはオンデマンドインスタンスと同じ休止機能を使用するようになりました。休止を有効にするにはここで [休止] を選択するか、インスタンス起動ウィザードの下部に表示される [停止 – 休止動作] フィールドから [有効化] を選択してください。休止の前提条件については「EC2 インスタンスの休止の前提条件」を参照してください。

- ワンタイムリクエストの場合、[Terminate] (終了) のみが有効です。

値を指定していない場合、デフォルトは [Terminate] (終了) になり、これは永続的なスポットインスタンスリクエストには無効です。デフォルトのままにして永続的なスポットインスタンスリクエストを起動しようとすると、エラーが発生します。

詳細については「スポットインスタンスの中断の動作」を参照してください。

11. [合計mary] (概要) パネルの [Number of instances] (インスタンス数) に、起動するインスタンス数を入力してください。

> **Note**
>
> Amazon EC2 が、スポットインスタンスごとに個別のリクエストを作成します。

12. [合計mary] (概要) パネルで、インスタンスの詳細を確認し、必要な変更を加えます。スポットインスタンスリクエストを送信した後はリクエストのパラメータを変更することはできません。[合計mary] (概要) パネルでリンクを選択すると、インスタンスの起動ウィザードのセクションに直接移動できます。詳細については「概要」を参照してください。

13. インスタンスを起動する準備ができたら、[Launch instance] (インスタンスの起動) を選択してください。

    インスタンスが起動しないか、状態が `terminated` ではなくすぐに `running` になる場合はAmazon EC2 インスタンスの起動に関する問題のトラブルシューティングを参照してください。

AWS CLI

run-instances を使用してスポットインスタンスリクエストを作成するには

run-instances コマンドを使用し、`--instance-market-options` パラメータでスポットインスタンスのオプションを次のように指定します。

```
--instance-market-options file://spot-options.json
```

JSON ファイルで指定するデータ構造は、次のとおりです。`ValidUntil`、および `InstanceInterruptionBehavior`、を指定することもできます。データ構造でフィールドを指定しないと、デフォルト値が使用されます。

次のサンプルでは`persistent` リクエストを作成します。

```
{
  "MarketType": "spot",
  "SpotOptions": {
    "SpotInstanceType": "persistent"
  }
}
```

request-spot-instances を使用してスポットインスタンスリクエストを作成するには

> **Note**
>
> request-spot-instances コマンドを使用してスポットインスタンスをリクエストすることは強くお勧めしません。これは計画された投資がないレガシー API であるためです。詳細については「使用すべき最適なスポットリクエスト方法はどれですか?」を参照してください。

ワンタイムリクエストを作成するにはrequest-spot-instances コマンドを使用します。

```
aws ec2 request-spot-instances \
    --instance-count 5 \
    --type "one-time" \
    --launch-specification file://specification.json
```

永続リクエストを作成するにはrequest-spot-instances を使用します。

```
aws ec2 request-spot-instances \
    --instance-count 5 \
    --type "persistent" \
    --launch-specification file://specification.json
```

以下のコマンドで使用する起動仕様ファイルの例についてはスポットインスタンスリクエストでの起動仕様の例を参照してください。起動仕様ファイルをスポットリクエストコンソールからダウンロードする場合は代わりに request-spot-fleet コマンドを使用する必要があります (スポットリクエストコンソールはスポットフリートを使用してスポットインスタンスリクエストを指定します)。

PowerShell

スポットインスタンスリクエストを作成する

New-EC2Instance コマンドレットを使用し、-InstanceMarketOption パラメータを使ってスポットインスタンスオプションを指定します。

```
-InstanceMarketOptions $marketOptions
```

スポットインスタンスオプションのデータ構造を、次のように作成します。

```
$spotOptions = New-Object Amazon.EC2.Model.SpotMarketOptions
$spotOptions.SpotInstanceType="persistent"
$marketOptions = New-Object Amazon.EC2.Model.InstanceMarketOptionsRequest
$marketOptions.MarketType = "spot"
$marketOptions.SpotOptions = $spotOptions
```

スポットインスタンスリクエストでの起動仕様の例

以下に、スポットインスタンスリクエストを作成するための request-spot-instances コマンドで使用できる起動設定の例を示します。詳細については「スポットインスタンスを管理する」を参照してください。

> **Important**
>
> request-spot-instances コマンドを使用してスポットインスタンスをリクエストすることは強くお勧めしません。これは計画された投資がないレガシー API であるためです。詳細については「使用すべき最適なスポットリクエスト方法はどれですか?」を参照してください。

例

- 例 1: スポットインスタンス の起動
- 例 2: 指定したアベイラビリティーゾーンで スポットインスタンス を起動する
- 例 3: 指定したサブネットで スポットインスタンス を起動する
- 例 4: ハードウェア専有スポットインスタンスを起動する

例 1: スポットインスタンス の起動

以下の例にはアベイラビリティーゾーンやサブネットは指定していません。Amazon EC2 によって自動的にアベイラビリティーゾーンが選択されます。Amazon EC2 は選択したアベイラビリティーゾーンのデフォルトのサブネットでインスタンスを起動します。

```
{
  "ImageId": "ami-0abcdef1234567890",
  "KeyName": "my-key-pair",
  "SecurityGroupIds": [ "sg-1a2b3c4d5e6f7g8h9" ],
  "InstanceType": "m5.medium",
  "IamInstanceProfile": {
      "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
}
```

例 2: 指定したアベイラビリティーゾーンで スポットインスタンス を起動する

以下の例にはアベイラビリティーゾーンが含まれています。Amazon EC2 は指定したアベイラビリティーゾーンのデフォルトのサブネットでインスタンスを起動します。

```
{
  "ImageId": "ami-0abcdef1234567890",
  "KeyName": "my-key-pair",
  "SecurityGroupIds": [ "sg-1a2b3c4d5e6f7g8h9" ],
```

```
  "InstanceType": "m5.medium",
  "Placement": {
    "AvailabilityZone": "us-west-2a"
  },
  "IamInstanceProfile": {
      "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
}
```

例 3: 指定したサブネットで スポットインスタンス を起動する

次の例にはサブネットが含まれます。Amazon EC2 は指定されたサブネットでインスタンスを起動します。デフォルト以外の VPC である場合、インスタンスにはデフォルトでパブリック IPv4 アドレスは割り当てられません。

```
{
  "ImageId": "ami-0abcdef1234567890",
  "SecurityGroupIds": [ "sg-1a2b3c4d5e6f7g8h9" ],
  "InstanceType": "m5.medium",
  "SubnetId": "subnet-1a2b3c4d",
  "IamInstanceProfile": {
      "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
}
```

デフォルト以外の VPC である場合、インスタンスにパブリック IPv4 アドレスを割り当てるには以下の例に示しているように AssociatePublicIpAddress フィールドを指定します。ネットワークインターフェイスの指定時には上記のコードブロックに示している SubnetId および SecurityGroupIds フィールドではなく、ネットワークインターフェイスを使用して、サブネット ID およびセキュリティグループ ID を含める必要があります。

```
{
  "ImageId": "ami-0abcdef1234567890",
  "KeyName": "my-key-pair",
  "InstanceType": "m5.medium",
  "NetworkInterfaces": [
    {
      "DeviceIndex": 0,
      "SubnetId": "subnet-1a2b3c4d5e6f7g8h9",
      "Groups": [ "sg-1a2b3c4d5e6f7g8h9" ],
      "AssociatePublicIpAddress": true
```

```
    }
  ],
  "IamInstanceProfile": {
      "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
}
```

例 4: ハードウェア専有スポットインスタンスを起動する

次の例ではdedicated のテナンシーを使用するスポットインスタンスをリクエストしています。ハードウェア専用スポットインスタンスはVPC 内で起動される必要があります。

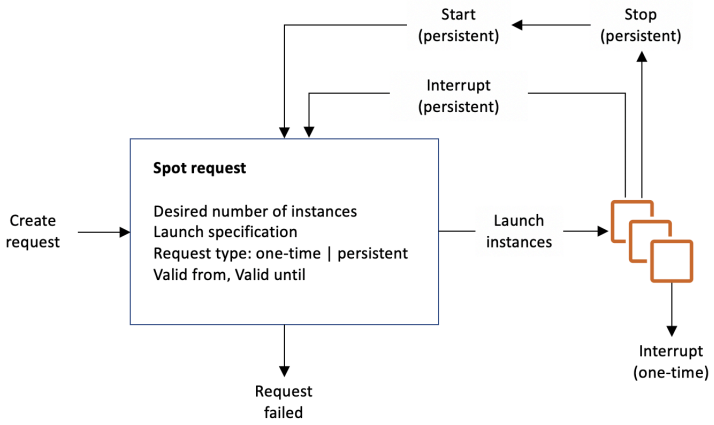
```
{
  "ImageId": "ami-0abcdef1234567890",
  "KeyName": "my-key-pair",
  "SecurityGroupIds": [ "sg-1a2b3c4d5e6f7g8h9" ],
  "InstanceType": "c5.8xlarge",
  "SubnetId": "subnet-1a2b3c4d5e6f7g8h9",
  "Placement": {
    "Tenancy": "dedicated"
  }
}
```

スポットインスタンスリクエストのステータスを取得する

スポットインスタンスリクエストを追跡し、スポットインスタンスの使用を計画するにはAmazon EC2 によって提供されるリクエストステータスを使用します。例えば、リクエストステータスによって、スポットリクエストがまだ受理されていない理由や、スポットリクエストの受理を妨げている制約の一覧を確認できます。

このプロセスの各ステップ (スポットリクエストのライフサイクルとも呼ばれる) では特定のイベントによって後続のリクエスト状態が決まります。

次の図にスポットインスタンスリクエストが動作する様子を示します。Amazon EC2がスポットインスタンスを中断した場合、あるいはユーザーがスポットインスタンスを停止した場合に、リクエストが再度開かれるかどうかはリクエストタイプ (ワンタイムまたは永続) によって決定されることに注意してください 。リクエストが永続リクエストの場合、スポットインスタンスの中断後、リクエストが再度開かれます。リクエストが永続的で、スポットインスタンスがユーザーにより停止された場合、リクエストはスポットインスタンスが開始されるまでは開かれません。



内容

- リクエストステータス情報の取得
- スポットリクエストコード
- EC2 スポットインスタンスリクエストのフルフィルメントイベント
- スポットリクエストの状態変化

リクエストステータス情報の取得

スポットインスタンスリクエストのステータス情報を取得できます。

Console

リクエストステータスの情報を取得するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択し、スポットリクエストを選択してください。
3. ステータスを確認するには[説明] タブの [ステータス] フィールドをチェックします。

AWS CLI

リクエストステータスの情報を取得するには

次の describe-spot-instance-requests コマンドを使用します。

```
aws ec2 describe-spot-instance-requests --spot-instance-request-
ids sir-0e54a519c9EXAMPLE
```

PowerShell

リクエストステータスの情報を取得するには

Get-EC2SpotInstanceRequest コマンドレットを使用します。

```
Get-EC2SpotInstanceRequest -SpotInstanceRequestId sir-0e54a519c9EXAMPLE
```

スポットリクエストコード

スポットリクエストステータス情報はステータスコード、更新時刻、およびステータスメッセージで構成されます。同時に、リクエスト入札ステータス情報はスポットリクエストの処理を決定する場合にも役に立ちます。

スポットリクエストステータスコードは次のとおりです。

`az-group-constraint`

Amazon EC2 は同じアベイラビリティーゾーンでお客様が要求したインスタンスをすべて起動できるとは限りません。

`bad-parameters`

スポットリクエストの 1 つ以上のパラメータが有効ではありません (例えば、指定した AMI が存在していません)。ステータスメッセージによって、どのパラメータが無効かを確認できます。

`canceled-before-fulfillment`

スポットリクエストが受理される前にユーザーがスポットリクエストをキャンセルしました。

`capacity-not-available`

要求したインスタンスに使用できる十分な容量が存在しません。

`constraint-not-fulfillable`

1 つ以上の制約条件が有効ではないため、スポットリクエストを受理できません (例えば、アベイラビリティーゾーンが存在していません)。ステータスメッセージによって、どの制約条件が無効かを確認できます。

`fulfilled`

スポットリクエストは `active` で、Amazon EC2 は スポットインスタンス を起動しています。

`instance-stopped-by-price`

スポット料金が上限価格を超えたため、インスタンスは停止しました。

`instance-stopped-by-user`

ユーザーがインスタンスを停止したか、インスタンスからシャットダウンコマンドを実行したために、インスタンスが停止されました。

`instance-stopped-no-capacity`

EC2 の容量管理のニーズにより、インスタンスが停止されました。

`instance-terminated-by-price`

スポット料金が上限価格を超えたため、インスタンスは削除されました。リクエストが永続入札の場合、プロセスが再開され、リクエストが評価保留となります。

`instance-terminated-by-schedule`

スポットインスタンスはスケジュールされた期間の最後に終了されました。

`instance-terminated-by-service`

インスタンスが停止状態から削除されました。

`instance-terminated-by-user`、または `spot-instance-terminated-by-user`

受理済みのスポットインスタンスを終了させたので、(永続リクエストでない限り) リクエストは `closed` 状態になり 、インスタンスは `terminated` 状態になます。

`instance-terminated-launch-group-constraint`

起動グループ内のインスタンスの 1 つ以上が終了したため、起動グループの制約条件が満たされなくなりました。

`instance-terminated-no-capacity`

標準的な容量管理プロセスにより、インスタンスは終了しました。

`launch-group-constraint`

Amazon EC2 はお客様が同時に要求したインスタンスをすべて起動できるわけではありません。同じ起動グループ内のインスタンスはすべて、同時に起動されて同時に終了します。

`limit-exceeded`

EBS ボリューム数または合計ボリュームストレージの上限を超えました。詳細については「Amazon EBS ユーザーガイド」の「Quotas for Amazon EBS」を参照してください。

`marked-for-stop`

スポットインスタンスは停止中としてマーキングされます。

`marked-for-termination`

スポットインスタンスに終了のためのマークが付けられています。

`not-scheduled-yet`

スポットリクエストはスケジュール設定された日付になるまで評価されません。

`pending-evaluation`

スポットインスタンスリクエストの作成後、システムがリクエストのパラメータを評価中はそのリクエストは `pending-evaluation` 状態となります。

`pending-fulfillment`

Amazon EC2 は スポットインスタンス をプロビジョニングしようとしています。

`placement-group-constraint`

現時点でスポットインスタンスをプレイスメントグループに追加できないため、まだスポットリクエストを受理することができません。

`price-too-low`

上限料金がスポット料金を下回っているため、リクエストを受理できません。この場合、インスタンスは起動されず、リクエストは `open` のままになります。

`request-canceled-and-instance-running`

スポットインスタンス がまだ実行されている間に、リクエストをキャンセルしました。リクエストは `cancelled` ですが、インスタンスは `running` のままです。

`schedule-expired`

スポットリクエストは指定された日付までに受理されなかったため、有効期限切れとなりました。

`system-error`

予期しないシステムエラーが発生しました。これが反復性の問題である場合はAWS サポート にお問い合わせください。

EC2 スポットインスタンスリクエストのフルフィルメントイベント

スポットインスタンスリクエストが受理されると、Amazon EC2 は EC2 スポットインスタンスリクエストのフルフィルメントイベントを Amazon EventBridge に送信します。Lambda 関数の呼び出しや Amazon SNS トピックへの通知など、このイベントが発生するたびにアクションを実行するルールを作成できます。

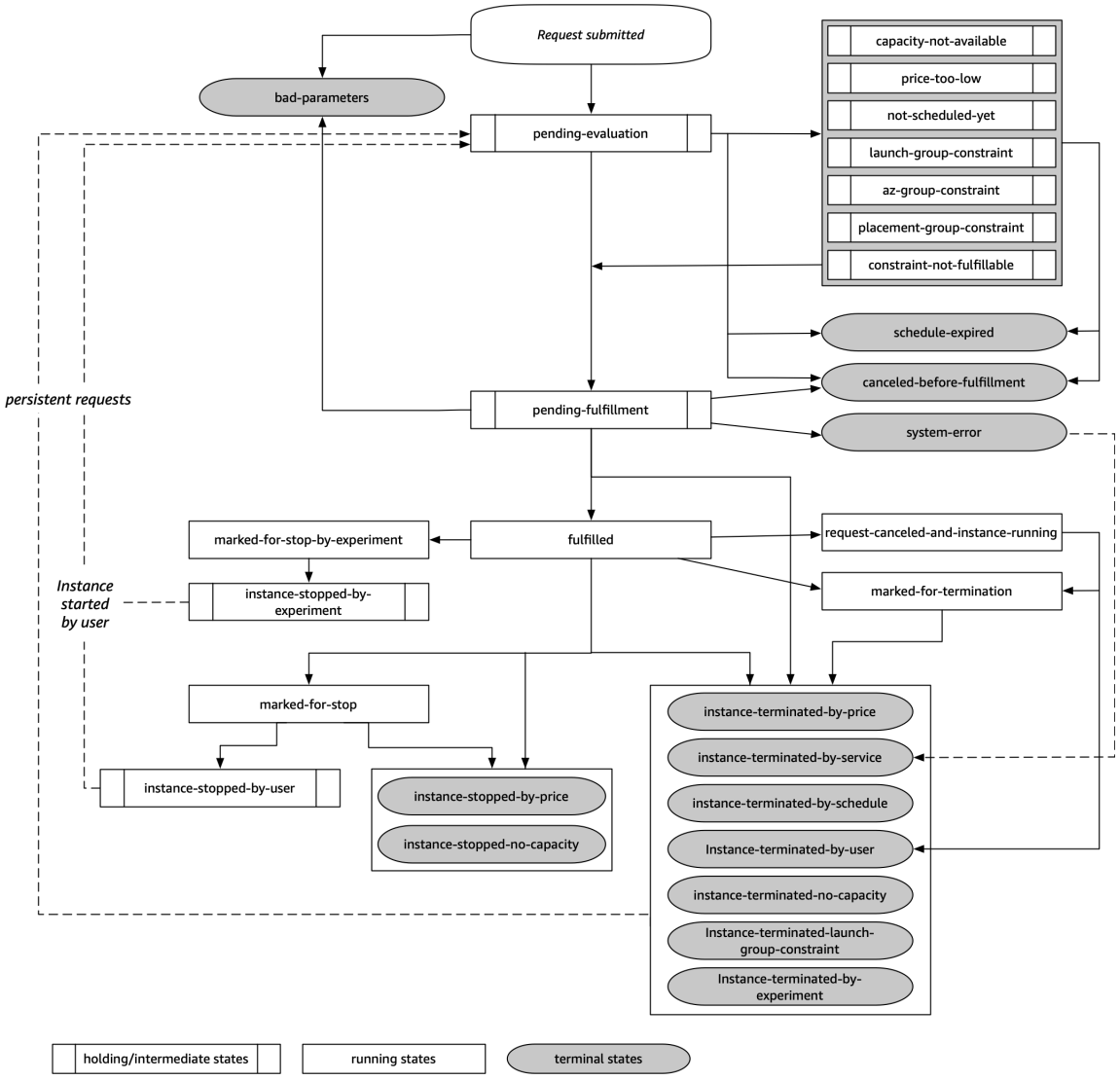
以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
    "id": "01234567-1234-0123-1234-012345678901",
    "detail-type": "EC2 Spot Instance Request Fulfillment",
    "source": "aws.ec2",
    "account": "123456789012",
    "time": "yyyy-mm-ddThh:mm:ssZ",
    "region": "us-east-2",
    "resources": ["arn:aws:ec2:us-east-2:123456789012:instance/i-1234567890abcdef0"],
    "detail": {
        "spot-instance-request-id": "sir-0e54a519c9EXAMPLE",
        "instance-id": "i-1234567890abcdef0"
    }
}
```

詳細については「Amazon EventBridge ユーザーガイド」を参照してください。

スポットリクエストの状態変化

次の図は申請から終了まで、スポットリクエストがライフサイクル全体を通してたどり得る経路を示しています。各ステップはノードとして表現され、各ノードのステータスコードはスポットリクエストおよびスポットインスタンスのステータスを示します。



評価保留

スポットインスタンスリクエストを作成すると、リクエストパラメータのいずれかが無効な (bad-parameters) 場合 を除き、そのリクエストは pending-evaluation 状態になります。

ステータスコード	リクエストの状態	インスタンスの状態
pending-evaluation	open	該当しない
bad-parameters	closed	該当しない

保持

1 つ以上のリクエストによる制約が有効であるが、まだ満足することができない場合や、容量が十分ではない場合、リクエストは制約が満たされるまで待機する保持状態になります。リクエストのオプションはリクエストが受理される可能性に影響します。例えば、キャパシティがない場合、キャパシティが利用可能になるまでリクエストは保留状態になります。アベイラビリティーゾーングループを指定する場合、アベイラビリティーゾーンの制約が満たされるまで、リクエストは保持状態になります。

いずれかのアベイラビリティーゾーンが停止した場合、他のアベイラビリティーゾーンのスポットインスタンスリクエストで使用可能な予備の EC2 容量が、影響を受ける可能性があります。

ステータスコード	リクエストの状態	インスタンスの状態
capacity-not-available	open	該当しない
price-too-low	open	該当しない
not-scheduled-yet	open	該当しない
launch-group-constraint	open	該当しない
az-group-constraint	open	該当しない
placement-group-constraint	open	該当しない
constraint-not-fulfillable	open	該当しない

評価保留/受理終了

特定の期間のみ有効なスポットインスタンスリクエストを作成し、そのリクエストが受理保留段階に到達する前に有効期間が経過した場合、 そのリクエストは terminal 状態になることがあります。これはお客様がリクエストをキャンセルした場合、またはシステムエラーが発生した場合にも発生する場合があります。

ステータスコード	リクエストの状態	インスタンスの状態
schedule-expired	cancelled	該当しない
canceled-before-fulfillment [1]	cancelled	該当しない
bad-parameters	failed	該当しない
system-error	closed	該当しない

[1] リクエストをキャンセルする場合。

受理保留

指定した制約条件 (もしあれば) が満たされると、スポットリクエストは pending-fulfillment ステータスになります。

この時点で、Amazon EC2 は要求されたインスタンスを提供するよう準備します。この段階でプロセスが停止した場合はスポットインスタンスが起動される前に、ユーザーがリクエストをキャンセルしたことが原因である可能性があります。または予期しないシステムエラーが発生したことが原因である可能性もあります。

ステータスコード	リクエストの状態	インスタンスの状態
pending-fulfillment	open	該当しない

受理済み

スポットインスタンスの仕様がすべて満たされると、スポットリクエストが受理されます。Amazon EC2 がスポットインスタンスを起動しますが、これには数分かかる場合があります。中断状態にあるスポットインスタンスが、休止または停止された場合、リクエストが再度受理できるようになるかキャンセルされるまで同じ状態が維持されます。

ステータスコード	リクエストの状態	インスタンスの状態
fulfilled	active	pending → running
fulfilled	active	stopped → running

スポットインスタンスを停止すると、そのインスタンスを再起動できるようになるか、リクエストがキャンセルされるまで、スポットリクエストは marked-for-stop または instance-stopped-by-user 状態になります。

ステータスコード	リクエストの状態	インスタンスの状態
marked-for-stop	active	stopping
instance-stopped-by-user [1]	disabled または cancelled [2]	stopped

[1] スポットインスタンスを停止するか、そのインスタンスからシャットダウンコマンドを実行すると、インスタンスは instance-stopped-by-user 状態になります。インスタンスを停止した後はインスタンスを再起動できるようになります。再起動時に、スポットインスタンスリクエストは pending-evaluation 状態に戻り、制約事項が満たされると Amazon EC2 によって新しいスポットインスタンスが起動されます。

[2] スポットインスタンスを停止して、リクエストをキャンセルしていない場合にはスポットリクエストの状態は disabled になります。スポットインスタンスが停止しており、リクエストの有効期限が切れている場合、リクエストの状態は cancelled になります。

受理済み終了

インスタンスタイプで使用可能なキャパシティがあり、お客様がインスタンスを終了しない限り、スポットインスタンスの実行は続行されます。Amazon EC2 でスポットインスタンスを終了する必要がある場合、スポットリクエストは終了状態になります。リクエストはお客様がスポットリクエストをキャンセルした場合や、スポットインスタンス を終了した場合も、終了状態になります。

ステータスコード	リクエストの状態	インスタンスの状態
request-canceled-and-instance-running	cancelled	running
marked-for-stop	active	running
marked-for-termination	active	running
instance-stopped-by-price	disabled	stopped
instance-stopped-by-user	disabled	stopped
instance-stopped-no-capacity	disabled	stopped
instance-terminated-by-price	closed (ワンタイム)、open (永続)	terminated
instance-terminated-by-schedule	closed	terminated
instance-terminated-by-service	cancelled	terminated
instance-terminated-by-user	closed または cancelled [1]	terminated
instance-terminated-no-capacity	closed (ワンタイム)、open (永続)	running †
instance-terminated-no-capacity	closed (ワンタイム)、open (永続)	terminated

ステータスコード	リクエストの状態	インスタンスの状態
instance-terminated-launch-group-constraint	closed (ワンタイム)、open (永続)	terminated

[1] インスタンスを終了したが、リクエストをキャンセルしていない場合、リクエストの状態は closed になります。インスタンスを終了し、リクエストをキャンセルする場合、リクエストの状態は cancelled になります。スポットリクエストをキャンセルする前にスポットインスタンスを終了した場合でも、そのスポットインスタンスの終了が Amazon EC2 によって検出されるまでに遅延が生じることがあります。この場合、リクエストの状態は closed または cancelled となります。

† Amazon EC2 が容量を戻す必要がある場合にスポットインスタンスに割り込み、かつ、インスタンスが割り込み時に終了するように設定されている場合、ステータスはすぐに instance-terminated-no-capacity に設定されます (marked-for-termination には設定されていません)。ただし、インスタンスはインスタンスがスポットインスタンスの中断通知を受信した 2 分間を反映して、2 分間 running 状態のままになります。2 分後、インスタンスの状態は terminated に設定されます。

中断実験

AWS Fault Injection Service を使用してスポットインスタンスの中断を開始すると、スポットインスタンス上のアプリケーションがどのように応答するかをテストできます。AWS FIS がスポットインスタンスを停止すると、スポットリクエストは marked-for-stop-by-experiment 状態になり、次に instance-stopped-by-experiment 状態になります。AWS FIS によってスポットインスタンスが終了した場合、スポットリクエストは instance-terminated-by-experiment 状態になります。詳細については「the section called “中断させる”」を参照してください。

ステータスコード	リクエストの状態	インスタンスの状態
marked-for-stop-by-experiment	active	running
instance-stopped-by-experiment	disabled	stopped

ステータスコード	リクエストの状態	インスタンスの状態
instance-terminated-by-experiment	closed	terminated

永続リクエスト

スポットリクエストが永続リクエストであり、関連するスポットインスタンスが (ユーザーまたは Amazon EC2 によって) 終了された場合にはそのリクエストは pending-evaluation 状態に戻るので、制約事項が満たされた後に Amazon EC2 は新しいスポットインスタンスを起動できます。

スポットインスタンスリクエストをタグ付けする

スポットインスタンスリクエストを分類および管理しやすくするため、カスタムメタデータでタグ付けすることができます。タグはスポットインスタンスリクエストの作成時、またはその後に割り当てることができます。Amazon EC2 コンソールまたはコマンドラインツールを使用してタグを割り当てることができます。

スポットインスタンスリクエストにタグ付けを行っても、そのスポットインスタンスリクエストによって起動されたインスタンスやボリュームには自動的なタグ付けは行われません。スポットインスタンスリクエストによって起動されたインスタンスやボリュームには明示的にタグを付ける必要があります。スポットインスタンスおよびボリュームへのタグの割り当ては起動時または起動後に行うことができます。

タグの仕組みの詳細については Amazon EC2 リソースのタグ付け を参照してください。

内容

- 前提条件
- 新しいスポットインスタンスリクエストにタグを付ける
- 既存のスポットインスタンスリクエストにタグ付けをする
- スポットインスタンスリクエストのタグを表示する

前提条件

リソースにタグ付けする許可をユーザーに付与します。IAM ポリシーとサンプルポリシーの詳細については 例: リソースのタグ付け を参照してください。

作成する IAM ポリシーはスポットインスタンスリクエストの作成に使用する方法によって決まります。

- インスタンスの起動ウィザードまたは `run-instances` を使用して スポットインスタンス をリクエストする場合はTo grant a user the permission to tag resources when using the launch instance wizard or run-instancesを参照してください。
- スポットインスタンスをリクエストするために `request-spot-instances`コマンドを使用する場合はTo grant a user the permission to tag resources when using request-spot-instancesを参照してください。

インスタンス起動ウィザードまたは run-instances を使用する場合にリソースにタグを付けるための許可をユーザーに付与するには

以下を含む IAM ポリシーを作成します。

- `ec2:RunInstances` アクション。これにより、インスタンスを起動するための許可がユーザーに付与されます。
- `Resource` で、`spot-instances-request` を指定します。これによりユーザーはスポットインスタンスを要求するためのスポットインスタンスリクエストを作成できるようになります。
- `ec2:CreateTags` アクション。これにより、タグを作成する許可がユーザーに付与されます。
- `Resource` で、`*` を指定します。これにより、ユーザーはインスタンスの起動時に作成されるすべてのリソースにタグを付けることを許可されます。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowLaunchInstances",
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1::image/*",
                "arn:aws:ec2:us-east-1:*:subnet/*",
```

```
                "arn:aws:ec2:us-east-1:*:network-interface/*",
                "arn:aws:ec2:us-east-1:*:security-group/*",
                "arn:aws:ec2:us-east-1:*:key-pair/*",
                "arn:aws:ec2:us-east-1:*:volume/*",
                "arn:aws:ec2:us-east-1:*:instance/*",
                "arn:aws:ec2:us-east-1:*:spot-instances-request/*"
            ]
        },
        {
            "Sid": "TagSpotInstanceRequests",
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "*"
        }
    ]
}
```

RunInstances アクションを使用してスポットインスタンスリクエストを作成し、その際、スポットインスタンスリクエストにタグを付ける場合にはAmazon EC2 が RunInstances ステートメント内で spot-instances-request リソースをどのように評価するのかについて、注意を払う必要があります。IAM ポリシーで、次のように評価が行われます。

- スポットインスタンスリクエストの作成時にタグを付けない場合、Amazon EC2 は RunInstances ステートメント内の spot-instances-request リソースを評価しません。
- スポットインスタンスリクエストの作成時にタグを付けると、 RunInstances ステートメント内の spot-instances-request リソースが、Amazon EC2 により評価されます。

したがって、spot-instances-request リソースの場合、次のルールが IAM ポリシーに適用されます。

- RunInstances を使用してスポットインスタンスリクエストを作成し、その際リクエストにタグを付けない場合はspot-instances-request リソースを明示的に許可しなくても、その呼び出しは成功します。
- RunInstances を使用してスポットインスタンスリクエストを作成する際に、そのリクエストにタグを付ける場合にはRunInstances の許可ステートメントに spot-instances-request リソースを含める必要があります。これがない場合は呼び出しが失敗します。
- RunInstances を使用してスポットインスタンスリクエストを作成する際に、そのリクエストにタグを付ける場合は許可ステートメント CreateTags で spot-instances-request リソースを指

定するか、そこに * ワイルドカードを含める必要があります。これがない場合は呼び出しが失敗します。

IAM ポリシー (スポットインスタンスリクエストでサポートされていないポリシーを含む) の例については[スポットインスタンス の操作]を参照してください。

request-spot-instances を使用する場合でリソースにタグを付けるための許可をユーザーに付与するには

以下を含む IAM ポリシーを作成します。

- `ec2:RequestSpotInstances` アクション。これにより、スポットインスタンスリクエストを作成する許可がユーザーに付与されます。
- `ec2:CreateTags` アクション。これにより、タグを作成する許可がユーザーに付与されます。
- `Resource` で、`spot-instances-request` を指定します。これにより、ユーザーはスポットインスタンスリクエストにのみタグを付けることが許可されます。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "TagSpotInstanceRequest",
            "Effect": "Allow",
            "Action": [
                "ec2:RequestSpotInstances",
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:us-east-1:111122223333:spot-instances-
request/*"
        }
    ]
}
```

新しいスポットインスタンスリクエストにタグを付ける

AWS CLI および PowerShell の例では、スポットインスタンスリクエストを次のように設定します。

- ResourceType で、spot-instances-request を指定します。別の値を指定すると、スポットインスタンスリクエストは失敗します。
- Tags で、キーと値のペアを指定します。キーと値のペアは複数指定できます。

Console

新しいスポットインスタンスリクエストにタグを付けるには

1. スポットインスタンスを管理するの手順に従います。
2. タグを追加するには[タグの追加] ページで [タグの追加] をクリックし、タグのキーと値を入力してください。追加するタグごとに [別のタグを追加] をクリックしてください。

   1 つのタグを、スポットインスタンスリクエスト、スポットインスタンス、およびボリュームに対し同時にタグ付けすることができます。3 つすべてにタグを付けるには[インスタンス]、[ボリューム]、[スポットインスタンスリクエスト] をそれぞれ選択してください。1 つまたは 2 つにのみタグを付けるにはタグを付けるリソースを選択し、他のリソースを選択していないことを確認します。
3. 必須フィールドにすべて入力してスポットインスタンスリクエストを作成した後、[起動] を選択してください。詳細については、「スポットインスタンスを管理する」を参照してください。

AWS CLI

新しいスポットインスタンスリクエストにタグを付けるには

request-spot-instances コマンドを --tag-specification オプションと共に使用します。

このタグ仕様は、スポットインスタンスリクエストに Environment=Production と Cost-Center=123 の 2 つのタグを追加します。

```
aws ec2 request-spot-instances \
    --instance-count 5 \
    --type "one-time" \
    --launch-specification file://specification.json \
    --tag-specification 'ResourceType=spot-instances-
request,Tags=[{Key=Environment,Value=Production},{Key=Cost-Center,Value=123}]'
```

PowerShell

新しいスポットインスタンスリクエストにタグを付けるには

Request-EC2SpotInstance コマンドレットを -TagSpecification パラメータと共に使用します。

```
-TagSpecification $tagspec
```

タグ仕様は次のように定義されます。スポットインスタンスリクエストに Environment=Production と Cost-Center=123 の 2 つのタグを追加します。

```
$tag1 = @{Key="Environment"; Value="Production"}
$tag2 = @{Key="Cost-Center"; Value="123"}
$tagspec = New-Object Amazon.EC2.Model.TagSpecification
$tagspec.ResourceType = "spot-instances-request"
$tagspec.Tags = @($tag1,$tag2)
```

既存のスポットインスタンスリクエストにタグ付けをする

Console

既存のスポットインスタンスリクエストにタグ付けをするには

スポットインスタンスリクエストの作成後に、コンソールを使用してそのリクエストにタグを追加できます。

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. スポットインスタンスリクエストを選択してください。
4. [Tags (タグ)] タブを選択してから、[タグの作成] を選択してください。

コンソールを使用して既存のスポットインスタンスにタグを付けるには

スポットインスタンスリクエストによってスポットインスタンスを起動した後で、コンソールを使用して、そのインスタンスにタグを追加できます。詳細については、「コンソールを使用してタグを追加する」を参照してください。

AWS CLI

既存のスポットインスタンスリクエストまたはスポットインスタンスにタグを付けるには

create-tags コマンドを使用して、既存のリソースにタグを付けます。次の例では、既存のスポットインスタンスリクエストとスポットインスタンスに、purpose=test のタグを付けています。

```
aws ec2 create-tags \
    --resources sir-0e54a519c9EXAMPLE i-1234567890abcdef0 \
    --tags Key=purpose,Value=test
```

PowerShell

既存のスポットインスタンスリクエストまたはスポットインスタンスにタグを付けるには

New-EC2Tag コマンドレットを使用します。次の例では、既存のスポットインスタンスリクエストとスポットインスタンスに、タグ purpose=test を追加しています。

```
New-EC2Tag `
    -Resource sir-0e54a519c9EXAMPLE, i-1234567890abcdef0 `
    -Tag @{Key="purpose"; Value="test"}
```

スポットインスタンスリクエストのタグを表示する

Console

スポットインスタンスリクエストのタグを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. スポットインスタンスリクエストを選択してから、[タグ] タブを選択してください。

AWS CLI

スポットインスタンスリクエストのタグを詳細表示するには

describe-spot-instance-requests コマンドを使用して、指定したスポットインスタンスリクエストの設定を表示します。この内容にはリクエストに指定されたタグがすべて含まれます。

```
aws ec2 describe-spot-instance-requests \
```

```
    --spot-instance-request-ids sir-0e54a519c9EXAMPLE \
    --query "SpotInstanceRequests[*].Tags"
```

以下は出力の例です。

```
[
    [
        {
            "Key": "Environment",
            "Value": "Production"
        },
        {
            "Key": "Department",
            "Value": "101"
        }
    ]
]
```

PowerShell

スポットインスタンスリクエストのタグを詳細表示するには

Get-EC2SpotInstanceRequest コマンドレットを使用します。

```
(Get-EC2SpotInstanceRequest `
    -SpotInstanceRequestId sir-0e54a519c9EXAMPLE).Tags
```

以下は出力の例です。

```
Key         Value
---         -----
Environment Production
Department  101
```

スポットインスタンスリクエストをキャンセルする

スポットインスタンスリクエストが不要になった場合にはそれをキャンセルすることができます。open、active、または disabled のスポットインスタンスリクエストのみキャンセルできます。

- スポットインスタンスリクエストがまだ受理されておらず、インスタンスが起動されていない段階ではそのリクエストは open 状態にあります。
- スポットインスタンスリクエストが受理され、スポットインスタンスの起動が完了している場合、そのスポットインスタンスリクエストは active 状態になります。
- ユーザーがスポットインスタンスを停止した場合、スポットインスタンスリクエストは disabled 状態になります。

スポットインスタンスリクエストの状態が active で、関連付けられたスポットインスタンスが実行されている場合、そのリクエストをキャンセルしても、関連するインスタンスは終了しません。スポットインスタンスの終了の詳細については[スポットインスタンスを終了する]を参照してください。。

Console

スポットインスタンスリクエストをキャンセルするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. スポットインスタンスリクエストを選択してください。
4. [アクション]、[リクエストのキャンセル] の順にクリックしてください。
5. (オプション) 関連付けられたスポットインスタンスを使い終わったら、スポットインスタンス を終了できます。[スポットリクエストのキャンセル] ダイアログボックスで、[インスタンスの終了]、[確認] の順にクリックしてください。

AWS CLI

スポットインスタンスリクエストをキャンセルするには

次の cancel-spot-instance-requests コマンドを使用します。

```
aws ec2 cancel-spot-instance-requests --spot-instance-request-
ids sir-0e54a519c9EXAMPLE
```

PowerShell

スポットインスタンスリクエストをキャンセルするには

Stop-EC2SpotInstanceRequest コマンドレットを使用します。

```
Stop-EC2SpotInstanceRequest -SpotInstanceRequestId sir-0e54a519c9EXAMPLE
```

スポットインスタンスを管理する

Amazon EC2 はキャパシティが利用可能であるときにスポットインスタンスを起動します。スポットインスタンスは中断されるか、ユーザーにより終了されるまで実行されます。

内容

- スポットインスタンスの検索
- 特定のリクエストで起動したインスタンスを検索する
- スポットインスタンスを停止する
- スポットインスタンスを開始する
- スポットインスタンスを終了する

スポットインスタンスの検索

スポットインスタンスはオンデマンドインスタンスとともに、コンソールの [インスタンス] ページに表示されます。以下の手順で、スポットインスタンスを検索します。

Console

スポットインスタンスを検索するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. すべてのスポットインスタンスを検索するには検索ペインで [インスタンスライフサイクル=スポット] を選択してください。
4. インスタンスがスポットインスタンスであることを確認するにはインスタンスを選択し、[詳細] タブを選択し、[ライフサイクル] の値を確認します。スポットインスタンスの値は `spot` で、オンデマンドインスタンスの値は `normal` です。

AWS CLI

スポットインスタンスを検索するには

次の describe-instances コマンドを使用します。

```
aws ec2 describe-instances --filters "Name=instance-lifecycle,Values=spot"
```

インスタンスがスポットインスタンスであるかどうかを確認するには

次の describe-instances コマンドを使用します。

```
aws ec2 describe-instances \
    --instance-ids i-1234567890abcdef0 \
    --query "Reservations[*].Instances[*].InstanceLifecycle" \
    --output text
```

出力が spot の場合、そのインスタンスはスポットインスタンスです。何も出力されない場合、インスタンスはオンデマンドインスタンスです。

PowerShell

スポットインスタンスを検索するには

Get-EC2Instance コマンドレットを使用します。

```
Get-EC2Instance -Filter @{Name="instance-lifecycle"; Values="spot"}
```

インスタンスがスポットインスタンスであるかどうかを確認するには

Get-EC2Instance コマンドレットを使用します。

```
(Get-EC2Instance -InstanceId i-1234567890abcdef0).Instances.InstanceLifecycle
```

出力が Spot の場合、そのインスタンスはスポットインスタンスです。何も出力されない場合、インスタンスはオンデマンドインスタンスです。

特定のリクエストで起動したインスタンスを検索する

次の手順に従って、特定のスポットインスタンスまたはスポットフリートリクエストから起動されたスポットインスタンスを検索します。

Console

リクエストのスポットインスタンスを検索するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインで、[Spot Requests] を選択してください。リストにはスポットインスタンスリクエストとスポットフリートリクエストの両方が含まれます。
3. スポットインスタンスリクエストが受理された場合、[キャパシティ] がスポットインスタンスの ID となります。スポットフリートの場合、[容量] はリクエストされた容量のうち受理された量を示します。スポットフリートのインスタンスの ID を表示するには拡張矢印を選択するか、フリートを選択した上で [インスタンス] を選択してください。
4. スポットフリートの場合、[キャパシティ] はリクエストされた容量のうち受理された量を示します。スポットフリート内のインスタンスの ID を表示するにはフリート ID を選択して詳細ページを開き、[インスタンス] ペインを見つけます。

AWS CLI

リクエストのスポットインスタンスを検索するには

次の describe-spot-instance-requests コマンドを使用します。

```
aws ec2 describe-spot-instance-requests \
    --spot-instance-request-ids sir-0e54a519c9EXAMPLE \
    --query "SpotInstanceRequests[*].{ID:InstanceId}"
```

出力例を次に示します。

```
[
    {
        "ID": "i-1234567890abcdef0"
    },
    {
        "ID": "i-0598c7d356eba48d7"
    }
]
```

PowerShell

リクエストのスポットインスタンスを検索するには

Get-EC2SpotInstanceRequest コマンドレットを使用します。

```
(Get-EC2SpotInstanceRequest -SpotInstanceRequestId sir-0e54a519c9EXAMPLE).InstanceId
```

スポットインスタンスを停止する

スポットインスタンスを今すぐ必要とはしないが、Amazon EBS ボリューム内に保持されているデータを失うことなく、後でこれを再起動する必要がある場合は、これを停止できます。スポットインスタンスを停止する手順はオンデマンドインスタンスを停止する手順と似ています。

> **Note**
>
> スポットインスタンスが停止している間、そのインスタンスの属性の一部は変更可能ですが、インスタンスタイプを変更することはできません。
> 停止しているスポットインスタンスの使用料またはデータ転送料は課金されませんが、Amazon EBS ボリュームのストレージに対しては課金されます。

制限事項

- スポットインスタンスを停止できるのはそのインスタンスが、`persistent` なスポットインスタンスリクエストから起動された場合だけです。
- 関連するスポットインスタンスリクエストがキャンセルされている場合はスポットインスタンスを停止することはできません。スポットインスタンスリクエストがキャンセルされた場合はスポットインスタンスを終了することのみ可能です。
- フリート、起動グループ、またはアベイラビリティーゾーングループの一部であるスポットインスタンスは停止できません。

Console

スポットインスタンスを停止するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. スポットインスタンスを選択してください。スポットインスタンスのインスタンス ID を保存しなかった場合は「the section called “スポットインスタンスの検索”」を参照してください。
4. [Instance state (インスタンスの状態)]、[Stop instance (インスタンスの停止)] の順に選択してください。
5. 確認を求められたら、[Stop] を選択してください。

AWS CLI

スポットインスタンスを停止するには

スポットインスタンスを手動で停止するにはstop-instances コマンドを使用します。

```
aws ec2 stop-instances --instance-ids i-1234567890abcdef0
```

PowerShell

スポットインスタンスを停止するには

Stop-EC2Instance コマンドレットを使用します。

```
Stop-EC2Instance -InstanceId i-1234567890abcdef0
```

スポットインスタンスを開始する

以前に停止したスポットインスタンスは開始することができます。

前提条件

スポットインスタンスは次の場合にのみ開始できます。

- スポットインスタンスを手動で停止している。
- スポットインスタンスが EBS-backed インスタンスである。
- スポットインスタンスに使用可能な容量がある。
- スポット料金が上限価格より低くなっている。

制限事項

- フリート、起動グループ、またはアベイラビリティーゾーングループの一部であるスポットインスタンスを開始することはできません。

スポットインスタンスを開始する手順はオンデマンドインスタンスを開始する手順と似ています。

Console

スポットインスタンスを開始するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. スポットインスタンスを選択してください。スポットインスタンスのインスタンス ID を保存しなかった場合は「the section called “スポットインスタンスの検索”」を参照してください。
4. [Instance state (インスタンスの状態)]、[Start instance (インスタンスの開始)] の順に選択してください。

AWS CLI

スポットインスタンスを開始するには

スポットインスタンスを手動で開始するにはstart-instances コマンドを使用します。

```
aws ec2 start-instances --instance-ids i-1234567890abcdef0
```

PowerShell

スポットインスタンスを開始するには

Start-EC2Instance コマンドレットを使用します。

```
Start-EC2Instance -InstanceId i-1234567890abcdef0
```

スポットインスタンスを終了する

> **Warning**
> インスタンスの終了は恒久的で、元に戻すことができません。
> インスタンスを終了すると、そのインスタンスには接続できなくなり、復元することもできません。アタッチされている Amazon EBS ボリュームで、終了時に削除されるように設定されているものも、すべて恒久的に削除され、復元できません。インスタンスストアボリュームに保存されているデータは、すべて恒久的に失われます。詳細については「インスタンスの終了方法」を参照してください。

インスタンスを終了する前に、終了後も保持する必要があるすべてのデータを永続的ストレージにバックアップしたことを確認してください。

永続スポットインスタンスリクエストによって起動された実行中または停止中のスポットインスタンスを終了すると、そのスポットインスタンスリクエストの状態は open に遷移し、新たなスポットインスタンスを起動できるようになります。新しいスポットインスタンスが起動されないようにするにはまずスポットインスタンスリクエストをキャンセルする必要があります。

スポットインスタンスを実行させている active スポットインスタンスリクエストをキャンセルしても、実行中のスポットインスタンスは自動的に終了されません。スポットインスタンスは手動で終了する必要があります。

停止中のスポットインスタンスを持つ disabled スポットインスタンスリクエストをキャンセルした場合、この停止中のスポットインスタンスはAmazon EC2 スポットサービスによって自動的に終了されます。スポットインスタンスリクエストをキャンセルしてから、スポットサービスがスポットインスタンスを終了するまでの間に、短い遅延が生じることがあります。

詳細については、「スポットインスタンスリクエストをキャンセルする」を参照してください。

Console

スポットインスタンスを手動で終了するには

1. インスタンスを終了する前に、終了時に Amazon EBS ボリュームが削除されることと、必要なデータすべてをインスタンスストアボリュームから永続的ストレージ (Amazon EBS や Amazon S3 など) にコピーしていることを確認して、データが失われないことを確認します。
2. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
3. ナビゲーションペインで、[インスタンス] を選択してください。
4. スポットインスタンスを選択してください。スポットインスタンスのインスタンス ID を保存しなかった場合は「the section called “スポットインスタンスの検索”」を参照してください。
5. [インスタンスの状態]、[インスタンスの終了 (削除)] の順に選択してください。
6. 確認を求めるメッセージが表示されたら、[終了 (削除)] を選択してください。

AWS CLI

スポットインスタンスを手動で終了するには

スポットインスタンスを手動で終了するにはterminate-instances コマンドを使用します。

```
aws ec2 terminate-instances --instance-ids i-1234567890abcdef0 i-0598c7d356eba48d7
```

PowerShell

スポットインスタンスを手動で終了するには

Remove-EC2Instance コマンドレットを使用します。

```
Remove-EC2Instance -InstanceId i-1234567890abcdef0
```

スポットインスタンスの中断。

Amazon EC2 で再びキャパシティーが必要になったときは予備の EC2 キャパシティーで スポットインスタンス を起動してキャパシティーを戻すのと引き換えに、大幅な割引を受けることができます。Amazon EC2 がスポットインスタンスを再要求した場合、このイベントをスポットインスタンスの中断と呼びます。

スポットインスタンス に対する需要は刻一刻と大幅に変化する可能性があります。また、スポットインスタンス の可用性も利用可能な未使用の EC2 インスタンスの数に応じて大きく変化する可能性があります。スポットインスタンスが中断される可能性は常に存在します。Amazon EC2 が スポットインスタンス を中断する場合、次のような理由が考えられます。

容量

Amazon EC2 は必要なときにスポットインスタンスを中断できます。EC2 は主に容量を再利用するためにインスタンスを再利用しますが、ホストのメンテナンスやハードウェアの使用停止などの他の理由でも発生する可能性があります。

価格

スポット料金が、上限料金を超えています。

上限料金はスポットリクエストで指定できます。ただし、上限料金を指定すると、指定しなかった場合に比べ、インスタンスの中断が増えます。

制約

スポットリクエストに、起動グループやアベイラビリティーゾーングループなどの制約を含んでいて、その制約条件が満たされなくなった場合にはスポットインスタンスはグループとして終了されます。

Amazon EC2 によりスポットインスタンスが中断される際にはスポットリクエストの作成時に指定した中断動作に応じて、インスタンスが終了、停止、または休止されます。

内容

- スポットインスタンスの中断の動作
- スポットインスタンスの中断に対する準備
- スポットインスタンスを中断させる
- スポットインスタンスの中断通知
- 中断した スポットインスタンス の検索
- Amazon EC2 がスポットインスタンスを終了しているかどうかを判別する
- 中断された スポットインスタンス の請求

スポットインスタンスの中断の動作

スポットリクエストの作成時に中断動作を指定できます。指定できる中断動作は以下のとおりです。

- 停止
- 休止
- 終了

デフォルトの動作ではAmazon EC2 は中断時にスポットインスタンスを終了します。

スポットインスタンス の中断の停止

中断時に Amazon EC2 がスポットインスタンスを停止するように指定できます。スポットインスタンスのリクエストのタイプは `persistent` である必要があります。スポットインスタンスリクエストで起動グループを指定することはできません。EC2 フリートまたはスポットフリートの場合、リクエストタイプは `maintain` である必要があります。

考慮事項

- 中断され停止したスポットインスタンスを再起動できるのは Amazon EC2 だけです。
- `persistent` スポットインスタンスリクエストで起動されたスポットインスタンスについては停止したインスタンスと同じアベイラビリティーゾーンと同じインスタンスタイプで利用可能な容量がある場合に、Amazon EC2 がその停止したインスタンスを再起動することができます (同じ起動仕様を使用する必要があります)。
- スポットインスタンスが停止している間、そのインスタンスの属性の一部は変更可能ですが、インスタンスタイプを変更することはできません。デタッチまたは削除された EBS ボリュームはスポットインスタンスが開始した際にアタッチされません。ユーザーがルートボリュームをデタッチし、Amazon EC2 がスポットインスタンスの開始を試みると、そのインスタンスは開始に失敗し、停止したインスタンスが Amazon EC2 により終了されます。
- ユーザーは停止中のスポットインスタンスを終了できます。
- ユーザーがスポットインスタンスリクエスト、EC2 フリート、またはスポットフリートをキャンセルすると、Amazon EC2 はそれらに関連付けられていて停止中のスポットインスタンスを終了します。
- 中断されたスポットインスタンスの停止中は維持されている EBS ボリュームに対してのみ課金されます。EC2 フリートおよびスポットフリートでは停止中のインスタンスの数が多い場合、アカウント内の EBS ボリューム数の上限を超えることがあります。スポットインスタンスが中断されたときの料金の詳細については「中断された スポットインスタンス の請求」を参照してください。
- インスタンスを停止することの影響について理解しておいてください。インスタンスが停止している場合に何が行われるかの詳細については「インスタンスの状態の違い」を参照してください。

中断した スポットインスタンス の休止

中断時に Amazon EC2 がスポットインスタンスを休止するように指定できます。詳細については「Amazon EC2 インスタンスの休止」を参照してください。

Amazon EC2 ではオンデマンドインスタンスで現在利用できるのと同じ休止状態をスポットインスタンスでも提供するようになりました。サポートの範囲が広がり、スポットインスタンスの休止では新たに以下がサポートされています。

- さらに多くの AMI をサポート
- さらに多くのインスタンスファミリーをサポート
- ユーザー起動の休止

中断したスポットインスタンスの終了

Amazon EC2 によりスポットインスタンスが中断される場合は停止や休止などの別の中断動作を指定しない限り、デフォルトでインスタンスが終了します。詳細については「Amazon EC2 インスタンスを終了する」を参照してください。

スポットインスタンスの中断に対する準備

スポットインスタンス に対する需要は刻一刻と大幅に変化する可能性があります。また、スポットインスタンス の可用性も利用可能な未使用の EC2 インスタンスの数に応じて大きく変化する可能性があります。スポットインスタンスが中断される可能性は常に存在します。したがって、アプリケーションでスポットインスタンスの中断に対して準備する必要があります。

スポットインスタンスの中断に備えて、以下のベストプラクティスに従うことをお勧めします。

- Auto Scaling グループを使用してスポットリクエストを作成します。スポットインスタンスが中断された場合、Auto Scaling グループは代替インスタンスを自動的に起動します。詳細については「Amazon EC2 Auto Scaling ユーザーガイド」の「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。
- 必要なソフトウェア設定を含む Amazon マシンイメージ (AMI) を使用することにより、リクエストが受理されたらすぐにインスタンスを実行できるように、準備が完了していることを確認します。また、ユーザーデータを使用して起動時にコマンドを実行することもできます。
- インスタンスストアボリュームのデータはインスタンスの停止または終了に伴って失われます。インスタンスストアボリュームの重要なデータを、Amazon S3、Amazon EBS、または Amazon DynamoDB などのより永続的なストレージにバックアップします。
- スポットインスタンスの終了の影響を受けない場所に、定期的に重要なデータを保存します。例えば、Amazon S3、Amazon EBS、または DynamoDB を使用できます。
- 作業を頻繁に保存できるように、作業を (Grid、Hadoop、キューベースのアーキテクチャを使用して) 細かいタスクに分割するか、チェックポイントを使用します。
- Amazon EC2 はスポットインスタンスでの中断のリスクが高まった場合に、再調整に関する推奨事項シグナルをそのインスタンスに対し送信します。再調整に関する推奨事項シグナルを利用すると、スポットインスタンス中断 2 分前の通知を待つことなく、スポットインスタンスの中断を事前に管理することができます。詳細については EC2 インスタンスの再調整に関する推奨事項を参照してください。
- スポットインスタンスのステータスをモニタリングするにはスポットインスタンス中断 2 分前の通知を使用します。詳細については スポットインスタンスの中断通知を参照してください。

- ではこの警告はできるだけ早く提供するよう努めていますが、警告が提供される前にスポットインスタンスが中断されることもあり得ます。再調整に関する推奨事項シグナルと中断通知をモニタリングしている場合でも、予期しないインスタンスの終了をアプリケーションが適切に処理できることを確認します。オンデマンドインスタンスを使用してアプリケーションを実行し、オンデマンドインスタンスを自分で終了することでこれを確認できます。
- AWS Fault Injection Service で制御された故障注入実験を実行し、スポットインスタンスが中断されたときのアプリケーションの応答をテストします。詳細については「AWS Fault Injection Service ユーザーガイド」の「チュートリアル: AWS FIS を使用してスポットインスタンスの中断をテストする」を参照してください。

スポットインスタンスを中断させる

Amazon EC2 コンソールでスポットインスタンスリクエストまたはスポットフリートリクエストを選択してスポットインスタンスの中断を実行すると、スポットインスタンス上のアプリケーションでの中断に関する処理をテストできます。スポットインスタンスの中断を開始すると、最初にそのスポットインスタンスの中断が 2 分後に行われることが Amazon EC2 から通知され、2 分経過後にインスタンスが中断されます。

スポットインスタンスの中断を処理するための、基盤となるサービスは AWS Fault Injection Service (AWS FIS) です。AWS FIS の詳細については「AWS Fault Injection Service」を参照してください。

> **Note**
>
> 中断動作は`terminate`、`stop`、および `hibernate` です。中断動作に対し `hibernate` を設定してスポットインスタンスの中断を開始すると、休止プロセスがすぐに開始されます。

スポットインスタンスの中断はすべての AWS リージョン (アジアパシフィック (ジャカルタ)、アジアパシフィック (大阪)、中国 (北京)、中国 (寧夏)、および中東 (UAE) を除く) で利用することができます。

内容

- スポットインスタンスを中断させる
- スポットインスタンスの中断を検証する
- クォータ

スポットインスタンスを中断させる

EC2 コンソールを使用すると、スポットインスタンスの中断をすばやく開始できます。スポットインスタンスリクエストを選択すると、1 つのスポットインスタンスの中断を開始できます。スポットフリートリクエストを選択すると、複数のスポットインスタンスの中断を一度に開始できます。

より高度な実験によりスポットインスタンスの中断をテストするにはAWS FIS コンソールで独自の実験を作成します。

EC2 コンソールを使用してスポットインスタンスリクエストで 1 つのスポットインスタンスの中断を開始するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Spot Requests] (スポットリクエスト) を選択してください。
3. スポットインスタンスリクエストを選択した後、[Actions] (アクション)、[Initiate interruption] (中断を開始) の順に選択してください。複数のスポットインスタンスリクエストを選択して中断を開始することはできません。
4. [Initiate Spot Instance interruption] (スポットインスタンスの中断を開始する) ダイアログボックスにある、[Service access] (サービスアクセス) で、デフォルトのロールか、既存のロールを選択してください。既存のロールを選択するには[既存のサービスロールを使用] を選択した後、[IAM ロール] で使用するロールを選択してください。
5. スポットインスタンスの中断を開始する準備ができたら、[Initiate interruption] (中断を開始) を選択してください。

EC2 コンソールを使用して、スポットフリートリクエストで 1 つまたは複数のスポットインスタンスの中断を開始するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Spot Requests] (スポットリクエスト) を選択してください。
3. スポットフリートリクエストを選択した後、[アクション]、[中断を開始] の順に選択してください。複数のスポットフリートリクエストを選択して中断を開始することはできません。
4. [スポットインスタンスの数を指定] ダイアログボックスの [中断するインスタンスの数] に、中断するスポットインスタンスの数を入力し、[確認] を選択してください。

> **Note**
>
> この数はフリート内のスポットインスタンス数や、1 回の実験で AWS FIS が中断できるスポットインスタンス数のクォータを超えることはできません。

5. [Initiate Spot Instance interruption] (スポットインスタンスの中断を開始する) ダイアログボックスにある、[Service access] (サービスアクセス) で、デフォルトのロールか、既存のロールを選択してください。既存のロールを選択するには[既存のサービスロールを使用] を選択した後、[IAM ロール] で使用するロールを選択してください。
6. スポットインスタンスの中断を開始する準備ができたら、[Initiate interruption] (中断を開始) を選択してください。

AWS FIS コンソールを使用して、スポットインスタンスの中断をテストするためのより高度な実験を作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Spot Requests] (スポットリクエスト) を選択してください。
3. [Actions] (アクション)、[Create advanced experiments] (高度な実験を作成) の順に選択してください。

   AWS FIS コンソールが開きます。詳細については「AWS Fault Injection Service ユーザーガイド」の「Tutorial: Test Spot Instance interruptions using AWS FIS」を参照してください。

スポットインスタンスの中断を検証する

中断を開始すると、以下のことが発生します。

- 対象のスポットインスタンスに対し、インスタンスの再調整に関する推奨事項が送信されます。
- AWS FIS がインスタンスを中断する 2 分前に、スポットインスタンスの中断の通知が発行されます。
- 2 分経過後に、スポットインスタンスが中断されます。
- AWS FIS によって停止されたスポットインスタンスはユーザーにより再起動されるまで停止状態を維持します。

中断を開始した後に、インスタンスが中断されていることを検証するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[スポットリクエスト] を開いてから、別のブラウザタブまたはウィンドウで[インスタンス] を開きます。
3. [スポットリクエスト] で、スポットインスタンスリクエストまたはスポットフリートリクエストを選択してください。初期ステータスはfulfilled です。インスタンスが中断されると、その中断の動作に応じて、以下のようにステータスが変化します。
   - terminate – ステータスが instance-terminated-by-experiment に変わります。
   - stop – ステータスが marked-for-stop-by-experiment に変わり、その後 instance-stopped-by-experiment に変わります。
4. インスタンスで、スポットインスタンスを選択してください。初期ステータスはRunning です。ユーザーがスポットインスタンスの中断通知を受け取り、2 分が経過すると、その中断の動作に応じて、以下のようにステータスが変化します。
   - stop – ステータスが Stopping に変わり、その後 Stopped に変わります。
   - terminate – ステータスが Shutting-down に変わり、その後 Terminated に変わります。

クォータ

AWS アカウント には1 回の実験で AWS FIS が中断できるスポットインスタンス数について、以下のデフォルトクォータがあります。

名前	デフォルト	引き上げ可能	説明
aws:ec2:send-spot-instance-interruptions のターゲット SpotInstances	サポートされている各リージョン： 5	あり	タグを使用してターゲットを特定するときに、aws:ec2:send-spot-instance-interruptions がターゲットにできるスポットインスタンスの実験ごとの最大数。

クォータの引き上げをリクエストできます。詳細については「Service Quotas ユーザーガイド」の「クォータの引き上げのリクエスト」を参照してください。

AWS FIS のクォータをすべて表示するにはService Quotas コンソールを開きます。ナビゲーションペインで、[AWS services] を選択し、AWS Fault Injection Service を選択してください。「AWS Fault Injection Service ユーザーガイド」で AWS Fault Injection Service のクォータをすべて確認することもできます。

スポットインスタンスの中断通知

スポットインスタンスの中断通知はAmazon EC2 がスポットインスタンスを停止または終了する 2 分前に発行される警告です。中断動作として休止状態を指定した場合は中断通知が表示されますが、休止状態プロセスはすぐに開始されるため、2 分間の警告は表示されません。

スポットインスタンスの中断を適切に処理する最善の方法は耐障害性のあるアプリケーションを設計することです。これを実現するにはスポットインスタンスの中断通知を活用します。中断通知は 5 秒ごとに確認することをお勧めします。

この中断通知は、EventBridge イベントとして、およびスポットインスタンス上のインスタンスメタデータの項目として提供されます。中断通知はベストエフォートベースで出力されます。

EC2 Spot Instance Interruption Warning イベント

Amazon EC2 がスポットインスタンスを中断しようとすると、実際の中断が起こる 2 分前にイベントが発生します (休止の場合は即時的にその状態に移行するため、中断通知は発行されますが 2 分前には提供されず、このイベントの対象にはなりません)。このイベントは Amazon EventBridge で検出できます。EventBridge イベントの詳細については「Amazon EventBridge ユーザーガイド」を参照してください。イベントルールの作成および使用方法の詳細な例についてはTaking Advantage of Amazon EC2 スポットインスタンスInterruption Noticesを参照してください。

以下に、スポットインスタンスでの中断イベントの例を示します。`instance-action` の可能な値は `hibernate`、`stop`、`terminate` です。

```
{
    "version": "0",
    "id": "12345678-1234-1234-1234-123456789012",
    "detail-type": "EC2 Spot Instance Interruption Warning",
    "source": "aws.ec2",
    "account": "123456789012",
    "time": "yyyy-mm-ddThh:mm:ssZ",
    "region": "us-east-2",
    "resources": ["arn:aws:ec2:us-east-2a:instance/i-1234567890abcdef0"],
```

```
    "detail": {
        "instance-id": "i-1234567890abcdef0",
        "instance-action": "action"
    }
}
```

> **Note**
>
> スポットインスタンスでの中断イベントの ARN 形式は `arn:aws:ec2:availability-zone:instance/instance-id` です。この形式はEC2 リソース ARN 形式とは異なります。

instance-action

`instance-action` 項目はアクションおよびアクションのおよその発生時刻 (UTC) を指定します。

Amazon EC2 が、スポットインスタンスを停止または終了のためにマークした場合、インスタンスメタデータ内に `instance-action` 項目が含まれるようになります。そうでない場合、これは存在しません。次のように、インスタンスメタデータサービスバージョン 2 (IMDSv2) を使用して、`instance-action` を取得できます。

Linux

```
TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
    && curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/spot/instance-action
```

Windows

```
[string]$token = Invoke-RestMethod `
    -Headers @{"X-aws-ec2-metadata-token-ttl-seconds" = "21600"} `
    -Method PUT -Uri http://169.254.169.254/latest/meta-data/spot/instance-action
```

次の出力例ではこのインスタンスの停止時刻を示します。

```
{"action": "stop", "time": "2017-09-18T08:22:00Z"}
```

次の出力例ではこのインスタンスの終了時刻を示します。

```
{"action": "terminate", "time": "2017-09-18T08:22:00Z"}
```

Amazon EC2 がインスタンスを停止または終了する準備をしていない場合や、お客様が自分でインスタンスを終了した場合、instance-action はインスタンスメタデータ内に存在せず、取得しようとした場合、HTTP 404 エラーが出力されます。

termination-time

termination-time 項目はインスタンスがシャットダウン信号を受信するだいたいの時刻 (UTC) を指定します。

> **Note**
>
> この項目は下位互換性のために維持されています。代わりに instance-action を使用してください。

スポットインスタンスに、Amazon EC2 によって (中断動作が terminate に設定されているスポットインスタンスの中断により、または永続的なスポットインスタンスリクエストのキャンセルにより) 終了のマークが付けられた場合、termination-time の項目はインスタンスのメタデータに含まれています。そうでない場合、これは存在しません。次のように、IMDSv2 を使用して termination-time を取得できます。

Linux

```
TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"`
if curl -H "X-aws-ec2-metadata-token: $TOKEN" -s http://169.254.169.254/latest/meta-data/spot/termination-time | grep -q .*T.*Z; then echo termination_scheduled; fi
```

Windows

```
[string]$token = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token-ttl-seconds" = "21600"} -Method PUT -Uri http://169.254.169.254/latest/meta-data/spot/termination-time
```

以下は出力の例です。

```
2015-01-05T18:02:00Z
```

Amazon EC2 がインスタンスを終了する準備をしていない場合 (スポットインスタンスの中断がない、または中断動作が stop または hibernate に設定されているため)、またはユーザーがスポットインスタンスを終了した場合にはtermination-time 項目はインスタンスメタデータ内に存在しないか (この場合、HTTP 404 エラーが出力されます)、時刻値以外の値が含まれます。

Amazon EC2 がインスタンスの終了に失敗した場合はリクエストステータスが fulfilled に設定されます。termination-time 値は元のおよその時刻のまま (過去の時刻になっていますが)、インスタンスのメタデータに残ります。

中断した スポットインスタンス の検索

EC2 インスタンスを記述すると、結果にはスポットインスタンスが含まれます。スポットインスタンスのインスタンスライフサイクルは spot です。スポットインスタンスのインスタンス状態は設定した中断動作に応じて stopped または terminated のいずれかになります。休止状態のスポットインスタンスの場合、インスタンスの状態は stopped です。

中断の理由の詳細についてはスポットリクエストのステータスコードを確認してください。詳細については、「the section called “スポットインスタンスリクエストのステータスを取得する”」を参照してください。

Console

中断したスポットインスタンスを検索するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. 次のフィルターを適用してください:[インスタンスライフサイクル=スポット]。
4. 設定した中断動作に応じて、[インスタンス状態=停止] または [インスタンス状態=終了] フィルターを適用します。
5. スポットインスタンスごとに、[詳細] タブの [インスタンスの詳細] で、[状態遷移メッセージ] を探します。次のコードはスポットインスタンスが中断されたことを示します。

   - Server.SpotInstanceShutdown
   - Server.SpotInstanceTermination

AWS CLI

中断したスポットインスタンスを見つけるには

describe-instances コマンドを --filters オプションで使用します。出力にインスタンス ID のみをリストするには --query オプションを含めます。

中断動作がスポットインスタンスを終了することである場合は、次の例を使用します。

```
aws ec2 describe-instances \
    --filters Name=instance-lifecycle,Values=spot \
              Name=instance-state-name,Values=terminated \
              Name=state-reason-code,Values=Server.SpotInstanceTermination \
    --query "Reservations[*].Instances[*].InstanceId"
```

中断動作がスポットインスタンスを停止することである場合は、次の例を使用します。

```
aws ec2 describe-instances \
    --filters Name=instance-lifecycle,Values=spot \
              Name=instance-state-name,Values=stopped \
              Name=state-reason-code,Values=Server.SpotInstanceShutdown \
    --query "Reservations[*].Instances[*].InstanceId"
```

PowerShell

中断したスポットインスタンスを見つけるには

Get-EC2Instance コマンドレットを使用します。

中断動作がスポットインスタンスを終了することである場合は、次の例を使用します。

```
(Get-EC2Instance `
    -Filter @{Key="instance-lifecycle"; Values="spot"} `
            @{Key="instance-state-name"; Values="terminated"} `
            @{Key="state-reason-code";
 Values="Server.SpotInstanceTermination"}).Instances.InstanceId
```

中断動作がスポットインスタンスを停止することである場合は、次の例を使用します。

```
(Get-EC2Instance `
    -Filter @{Key="instance-lifecycle"; Values="spot"} `
```

```
            @{Key="instance-state-name"; Values="stopped"} `
            @{Key="state-reason-code";
Values="Server.SpotInstanceTermination"}).Instances.InstanceId
```

Amazon EC2 がスポットインスタンスを終了しているかどうかを判別する

スポットインスタンスはAmazon EC2 がスポットインスタンスの中断に応じて終了するまで、またはユーザーが自分で終了するまで実行されます。詳細については「the section called “中断動作”」を参照してください。

スポットインスタンスが終了された後、AWS CloudTrail を使用して、Amazon EC2 がそのスポットインスタンスを終了したかどうかを確認できます。CloudTrail ログに `BidEvictedEvent` が含まれている場合、Amazon EC2 がスポットインスタンスを終了したことを示します。代わりに `TerminateInstances` イベントが含まれている場合はユーザーがスポットインスタンスを終了したことを示します。

またはAmazon EC2 がスポットインスタンスを中断することを示す通知を受け取りたい場合は Amazon EventBridge を使用して EC2 スポットインスタンスの中断警告イベントに応答します。

CloudTrail で BidEvictedEvent イベントを表示するには

1. CloudTrail コンソールの https://console.aws.amazon.com/cloudtrail/ を開いてください。
2. ナビゲーションペインで [Event history (イベント履歴)] を選択してください。
3. フィルターのリストから [イベント名] を選択し、右側のフィルターフィールドに「**`BidEvictedEvent`**」と入力してください。
4. (オプション) 時間範囲を選択してください。
5. リストが空でない場合は結果のエントリから [BidEvictedEvent] を選択して詳細ページを開きます。スポットインスタンスに関する情報はスポットインスタンスの ID を含めて、[イベントレコード] ペインで確認できます。イベントレコードの例を以下に示します。

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "accountId": "123456789012",
        "invokedBy": "ec2.amazonaws.com"
    },
    "eventTime": "2016-08-16T22:30:00Z",
    "eventSource": "ec2.amazonaws.com",
    "userAgent": "ec2.amazonaws.com",
```

```
    "sourceIPAddress": "ec2.amazonaws.com",
    "eventName": "BidEvictedEvent",
    "awsRegion": "us-east-2",
    "eventID": "d27a6096-807b-4bd0-8c20-a33a83375054",
    "eventType": "AwsServiceEvent",
    "recipientAccountId": "123456789012",
    "RequestParameters": null,
    "ResponseElements": null,
    "serviceEventDetails": {
        "instanceIdSet": [
          "i-1eb2ac8eEXAMPLE"
        ]
    }
}
```

6. BidEvictedEvent イベントのエントリが見つからない場合はイベント名として「**TerminateInstances**」と入力してください。TerminateInstances のイベントレコードの詳細については「the section called “Amazon EC2 API イベントの例”」を参照してください。

中断された スポットインスタンス の請求

スポットインスタンスが中断された場合、インスタンスと EBS ボリュームの使用量に対して課金され、その他に次のような料金が発生する可能性があります。

インスタンスの使用状況

スポットインスタンスを中断するユーザー	オペレーティングシステム	最初の 1 時間で中断	最初の 1 時間後の任意の時間に中断
ユーザー自らスポットインスタンスを停止または終了した場合	Windows および Linux (SUSE は除く)	使用された時間 (秒) の請求	使用された時間 (秒) の請求
	SUSE	使用時間が 1 時間未満の場合でも、1 時間分の料金を請求	使用された 1 時間分 (中断された時間が 1 時間未満の場合も 1時間分) を請求
Amazon EC2 がスポットインスタンスを中断した場合	Windows および Linux (SUSE は除く)	料金は発生しない	使用された時間 (秒) の請求
	SUSE	料金は発生しない	

スポットインスタンスを中断するユーザー	オペレーティングシステム	最初の 1 時間で中断	最初の 1 時間後の任意の時間に中断
			使用された 1 時間分は請求されるが、中断された時間が 1 時間未満の場合は請求されない

EBS ボリュームの使用状況

中断されたスポットインスタンスの停止中は維持されている EBS ボリュームに対してのみ課金されます。

EC2 フリートおよびスポットフリートでは停止中のインスタンスの数が多い場合、アカウント内の EBS ボリューム数の上限を超えることがあります。

EC2 インスタンスの再調整に関する推奨事項

EC2 インスタンスの再調整に関するレコメンデーションはスポットインスタンスで中断のリスクが高まった場合に通知するためのシグナルです。シグナルはスポットインスタンス中断 2 分前の通知よりも早く到着するので、スポットインスタンスを事前に管理するタイミングを知ることができます。ワークロードを、中断のリスクが高くない新規または既存の スポットインスタンス に再調整することができます。

2 分間のスポットインスタンス中断通知の前に、Amazon EC2 が再調整に関する推奨事項シグナルを送信することは必ずしも可能ではありません。したがって、再調整に関する推奨事項シグナルは2 分間の中断通知とともに到着する可能性があります。

再調整に関する推奨事項はEventBridge イベントとして、およびスポットインスタンス上のインスタンスメタデータの項目として使用できます。イベントはベストエフォートベースで発生します。

> **Note**
>
> 再調整に関する推奨事項は2020 年 11 月 5 日 00:00 UTC 以降に起動される スポットインスタンス のみをサポートしています。

内容

- 実行できるアクションの再調整
- 再調整に関する推奨事項シグナルのモニタリング
- 再調整に関する推奨事項シグナルを使用するサービス

実行できるアクションの再調整

以下に、実行可能な再調整アクションをいくつか挙げます。

適切なシャットダウン

スポットインスタンスの再調整に関する推奨事項シグナルを受信すると、インスタンスのシャットダウン手順を開始できます。この際、インスタンスを停止する前に各プロセスの完了を確認する必要が生じることがあります。例えば、システムログまたはアプリケーションログを Amazon Simple Storage Service (Amazon S3) にアップロードしたり、Amazon SQS ワーカーをシャットダウンしたり、ドメインネームシステム (DNS) からの登録解除を完了したりできます。また、作業内容を外部ストレージに保存し、後で再開することもできます。

新しい作業がスケジュールされるのを防止

スポットインスタンスの再調整に関する推奨事項シグナルを受信すると、スケジュールされた作業が完了するまでインスタンスの使用を継続しながら、新しい作業がインスタンス上でスケジュールされることを回避できます。

新しい代替インスタンスを積極的に起動

Auto Scaling グループ、EC2 フリート、または スポットフリート を設定しておくと、再調整に関する推奨事項シグナルが送信された場合に、代替のスポットインスタンスを自動的に起動させることができます。詳細については「Amazon EC2 Auto Scaling ユーザーガイド」の「キャパシティーの再調整を使用して Amazon EC2 スポットの中断に対処する」およびこのユーザーガイドの「EC2 フリートとスポットフリートでキャパシティの再調整を使用して、リスクのあるスポットインスタンスを置き換える」を参照してください。

再調整に関する推奨事項シグナルのモニタリング

再調整に関する推奨事項シグナルをモニタリングして、それが発されたときに、前のセクションで指定したアクションを実行できるようにすることができます。再調整に関する推奨事項シグナルは Amazon EventBridge (旧称 Amazon CloudWatch Events) に送信されるイベントとして、およびスポットインスタンス上のインスタンスメタデータとしての使用が可能です。

再調整に関する推奨事項シグナルをモニタリングする:

- Amazon EventBridge の使用
- インスタンスメタデータの使用

Amazon EventBridge の使用

再調整に関する推奨事項シグナルがスポットインスタンスに対して発信されると、そのシグナルのイベントが Amazon EventBridge に送信されます。EventBridge がルールで定義されているパターンに一致するイベントパターンを検出すると、EventBridge はルールで指定されているターゲットを呼び出します。

次に、再調整に関する推奨事項シグナルのイベント例を示します。

```
{
    "version": "0",
    "id": "12345678-1234-1234-1234-123456789012",
    "detail-type": "EC2 Instance Rebalance Recommendation",
    "source": "aws.ec2",
    "account": "123456789012",
    "time": "yyyy-mm-ddThh:mm:ssZ",
    "region": "us-east-2",
    "resources": ["arn:aws:ec2:us-east-2:123456789012:instance/i-1234567890abcdef0"],
    "detail": {
        "instance-id": "i-1234567890abcdef0"
    }
}
```

次のフィールドはルールで定義されているイベントパターンになります。

`"detail-type": "EC2 Instance Rebalance Recommendation"`

イベントが再調整に関する推奨事項イベントであることを特定します

`"source": "aws.ec2"`

イベントが Amazon EC2 からのものであることを特定します

EventBridge ルールを作成します

EventBridge ルールを作成し、イベントパターンがルールに一致したときに実行するアクションを自動化できます。

次の例ではAmazon EC2 が再調整に関する推奨事項シグナルを発するたびに、E メール、テキストメッセージ、またはモバイルプッシュ通知を送信する EventBridge ルールを作成します。シグナルは `EC2 Instance Rebalance Recommendation` イベントとして発され、ルールによって定義されたアクションがトリガーされます。

EventBridge ルールを作成する前に、E メール、テキストメッセージ、またはモバイルプッシュ通知用の Amazon SNS トピックを作成する必要があります。

再調整に関する推奨事項イベントの EventBridge ルールを作成するには

1. Amazon EventBridge コンソールの https://console.aws.amazon.com/events/ を開いてください。
2. [Create rule] を選択してください。
3. [Define rule detail] (詳細の定義) で、次の操作を行います。
   a. ルールの [Name (名前)] を入力し、必要に応じて説明を入力してください。

      ルールには同じリージョン内および同じイベントバス上の別のルールと同じ名前を付けることはできません。
   b. [イベントバス] として、[デフォルト] を選択してください。アカウント内の AWS のサービスで生成されたイベントは常に、そのアカウントのデフォルトのイベントバスに送られます。
   c. [ルールタイプ] で、[イベントパターンを持つルール] を選択してください。
   d. [Next] を選択してください。
4. [Build event pattern] (イベントパターンの作成) で、次の操作を行います。
   a. [Event source] (イベントソース) で、[AWS events or EventBridge partner events] (イベントまたは EventBridge パートナーイベント) を選択してください。
   b. この例では [Event pattern] (イベントパターン) で、`EC2 Instance Rebalance Recommendation` イベントと一致するように次のイベントパターンを指定してから、[Save] (保存) を選択してください。

```
{
 "source": ["aws.ec2"],
 "detail-type": ["EC2 Instance Rebalance Recommendation"]
}
```

イベントパターンを追加するには以下のように [Event pattern form] (イベントパターンフォーム) を選択してテンプレートを使用するか、[Custom pattern (JSON editor)] (カスタムパターン (JSON エディター)) を選択して独自のパターンを指定します。

i. テンプレートを使用してイベントパターンを作成するには以下の操作を行います。

A. [Event pattern form] (イベントパターンフォーム) を選択してください。

B. [イベントソース] で、[AWS のサービス] を選択してください。

C. [AWS Service] (サービス) で、[EC2 Spot Fleet] (EC2 スポットフリート) を選択してください。

D. [Event type] (イベントタイプ) で、[EC2 Instance Rebalance Recommendation] (EC2 インスタンスのリバランスに関するレコメンデーション) を選択してください。

E. テンプレートをカスタマイズするには[Edit pattern] (パターンを編集) を選択した上で、この例のイベントパターンに合わせた変更を行います。

ii. (代替案) 以下の操作を行って、カスタムイベントパターンを指定します。

A. [Custom pattern (JSON editor)] (カスタムパターン (JSON エディター)) を選択してください。

B. [Event pattern] (イベントパターン) ボックスに、この例のイベントパターンを追加します。

c. [Next] を選択してください。

5. [Select target(s)] (ターゲットを選択) で、以下の操作を行います。

a. [ターゲットタイプ] で、[AWS のサービス] を選択してください。

b. イベントの発生時に E メール、テキストメッセージ、またはモバイルプッシュ通知を送信するために、[Select a target] (ターゲットを選択) で、[SNS topic] (SNS トピック) を選択してください。

c. [Topic (トピック)] で、既存のトピックを選択してください。先ず、Amazon SNS コンソールを使用して Amazon SNS トピックを作成する必要があります。詳細については、Amazon Simple Notification Service デベロッパーガイド のAmazon SNS を使用した Application-to-Person (A2P) メッセージングを参照してください。

d. (オプション) [Additional settings] (追加設定) で、その他の設定を行うこともできます。詳細については、「Amazon EventBridge ユーザーガイド」の「イベントに反応する Amazon EventBridge ルールの作成」(ステップ 16) を参照してください。

e. [Next] を選択してください。

6. (オプション) 必要な場合は、[Tags] (タグ) で 1 つ以上のタグを作成したルールに割り当て、[Next] (次へ) を選択してください。
7. [Review and create] (確認して作成) で、以下の操作を行います。

   a. ルールの詳細を確認し、必要な場合は変更を行います。

   b. ルールの作成を選択してください。

詳細については「Amazon EventBridge ユーザーガイド」の「Amazon EventBridge ルール」と「Amazon EventBridge イベントパターン」を参照してください。

インスタンスメタデータの使用

インスタンスメタデータカテゴリ events/recommendations/rebalance はスポットインスタンスに対して再調整に関する推奨事項シグナルが発されたおおよその時間 (UTC) を示します。

再調整に関する推奨事項シグナルを 5 秒ごとに確認し、再調整に関する推奨事項に基づいて行動する機会を見逃さないようにすることをお勧めします。

スポットインスタンスが受信した、再調整に関する推奨事項ではそのシグナルが発行された時刻がインスタンスメタデータに含まれています。信号が発された時間は以下のように取得できます。

IMDSv2

Linux

次のコマンドを Linux インスタンスから実行します。

IMDSv2

```
TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
    && curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/events/recommendations/rebalance
```

Windows

次のコマンドレットを Windows インスタンスから実行します。

```
[string]$token = Invoke-RestMethod `
```

```
    -Headers @{"X-aws-ec2-metadata-token-ttl-seconds" = "21600"} `
    -Method PUT -Uri http://169.254.169.254/latest/api/token
```

```
Invoke-RestMethod `
    -Headers @{"X-aws-ec2-metadata-token" = $token} `
    -Method GET -Uri http://169.254.169.254/latest/meta-data/events/recommendations/
rebalance
```

IMDSv1

Linux

次のコマンドを Linux インスタンスから実行します。

```
curl http://169.254.169.254/latest/meta-data/events/recommendations/rebalance
```

Windows

次のコマンドレットを Windows インスタンスから実行します。

```
Invoke-RestMethod -Uri http://169.254.169.254/latest/meta-data/events/
recommendations/rebalance
```

次に、再調整に関する推奨事項シグナルがスポットインスタンスに対して送信された時刻を、UTC で表示する出力例を示します。

```
{"noticeTime": "2020-10-27T08:22:00Z"}
```

インスタンスに対してシグナルが送信されていない場合、`events/recommendations/rebalance` は存在せず、取得しようとすると HTTP 404 エラーが表示されます。

再調整に関する推奨事項シグナルを使用するサービス

Amazon EC2 Auto Scaling、EC2 フリート、およびスポットフリートにおいて再調整に関する推奨事項シグナルを使用することで、ワークロードの可用性を維持しやすくなります。実行中のインスタンスがスポットインスタンス中断 2 分前の通知を受信していない段階から、事前にフリートを新しいスポットインスタンスで強化することができます。このようなサービスではスポットインスタンスの可用性に影響する変更を事前にモニタリングし、対応させることができます。詳細については次を参照してください:

- 「Amazon EC2 Auto Scaling ユーザーガイド」の「キャパシティーの再調整を使用して Amazon EC2 スポットの中断に対処する」
- このユーザーガイドの EC2 フリートとスポットフリートのトピックの EC2 フリートとスポットフリートでキャパシティの再調整を使用して、リスクのあるスポットインスタンスを置き換える

スポットプレイスメントスコア

スポットプレイスメントスコア機能ではスポットの容量要件に基づいて AWS リージョンやアベイラビリティーゾーンを推奨することができます。スポット容量は変動し、必要な容量が常に得られるかどうかはわかりません。スポットプレイスメントスコアはリージョンまたはアベイラビリティーゾーンでスポットリクエストが成功する可能性を示します。

> **Note**
>
> スポットプレイスメントスコアは利用可能な容量や中断のリスクに関して、いかなる保証も提供しません。スポットプレイスメントスコアはレコメンデーションとしてのみ機能します。

ユースケース

スポットプレイスメントスコア機能は次の場合に使用できます。

- 現在のリージョンでの容量ニーズの増加または使用可能な容量の減少に応じて、必要に応じて別のリージョンでスポットコンピューティング性能を再配置してスケールします。
- 単一アベイラビリティーゾーンのワークロードを実行する最適なアベイラビリティーゾーンを特定します。
- 将来のスポット容量のニーズをシミュレートして、スポットベースのワークロードの拡張に最適なリージョンを選択できるようにします。
- スポットキャパシティのニーズを満たす、最適なインスタンスタイプの組み合わせを特定するには。

内容

- 制限
- コスト
- スポットプレイスメントスコアの仕組み

- スポットプレイスメントスコアに必要なアクセス許可
- スポットプレイスメントスコアを計算する

制限

- ターゲット容量制限 – スポットプレイスメントスコアのターゲット容量制限は潜在的な使用量の増加を考慮しながら、最近のスポット使用量に基づきます。最近のスポット使用がない場合はスポットリクエストの制限に合わせてデフォルトの低い制限が提供されます。
- リクエスト設定の制限 – スポットプレイスメントスコア機能の意図された使用に関連しないパターンを検出した場合、24 時間以内に新しいリクエスト設定の数を制限できます。上限に達した場合はすでに使用したリクエスト設定を再試行できますが、次の 24 時間まで新しいリクエスト設定を指定することはできません。
- インスタンスタイプの最小数 – インスタンスタイプを指定する場合は少なくとも 3 つの異なるインスタンスタイプを指定する必要があります。指定しないと、Amazon EC2 は低いスポットプレイスメントスコアを返します。同様に、インスタンス属性を指定する場合は少なくとも 3 つの異なるインスタンスタイプを解決する必要があります。インスタンスタイプは異なる名前を持つ場合、異なると見なされます。例えば、m5.8xlarge、m5a.8xlarge、およびm5.12xlarge はすべて異なると見なされます。

コスト

スポットプレイスメントスコア機能は追加料金なしで使用できます。

スポットプレイスメントスコアの仕組み

スポットプレイスメントスコア機能を使用する場合はまずスポットインスタンスのコンピューティング要件を指定します。その後、Amazon EC2 はスポットリクエストが成功する可能性が高い上位 10 リージョン、を返します。各リージョンまたはアベイラビリティーゾーンは1～10 のスケールで採点されます。10 はスポットリクエストが成功する可能性が高いことを示し、1 はスポットリクエストが成功する可能性が低いことを示します。

スポットプレイスメントスコア機能を使用するには次のステップに従います。

- ステップ 1: スポット要件を指定する
- ステップ 2: スポットプレイスメントスコアレスポンスのフィルタリング
- ステップ 3: レコメンデーションを確認する
- ステップ 4: レコメンデーションを使用する

ステップ 1: スポット要件を指定する

まず、希望するターゲットスポット容量とコンピューティング要件を次のように指定します。

1. ターゲットスポット容量を指定し、オプションでターゲット容量単位を指定します。

   目的のターゲットスポット容量はインスタンスまたは vCPUs の数、または MiB のメモリ量の観点から指定できます。vCPU 数またはメモリ量でターゲット容量を指定するにはターゲット容量単位を `vcpu` または `memory-mib` のように指定する必要があります。それ以外の場合、デフォルトはインスタンス数になります。

   vCPU の数またはメモリ量の観点からターゲット容量を指定することで、総容量を数えるときにこれらの単位を使用できます。例えば、異なるサイズのインスタンスを組み合わせて使用する場合はターゲット容量を vCPU の総数として指定できます。スポットプレイスメントスコア機能はvCPU の数でリクエスト内の各インスタンスタイプを考慮し、ターゲット容量を合計するときに、インスタンスの総数ではなく vCPU の総数をカウントします。

   例えば、合計ターゲット容量を 30 に指定し、インスタンスタイプリストが c5.xlarge (4 vCPU)、m5.2xlarge (8 vCPU)、および r5.large (2 vCPU) で構成されているとします。合計 30 個の vCPU を実現するには2 個のc5.xlarge (2*4 vCPU)、2 m5.2xlarge (2*8 vCPU)、3 r5.large (3*2 vCPU) を混在させることができます。
2. インスタンスタイプまたはインスタンス属性を指定します。

   使用するインスタンスタイプを指定するか、コンピューティング要件に必要なインスタンス属性を指定して、それらの属性を持つインスタンスタイプを Amazon EC2 に識別させることができます。これは属性ベースのインスタンスタイプの選択と呼ばれます。

   同じスポットプレイスメントスコアリクエストではインスタンスタイプとインスタンス属性の両方を指定することはできません。

   インスタンスタイプを指定する場合は少なくとも 3 つの異なるインスタンスタイプを指定する必要があります。指定しないと、Amazon EC2 は低いスポットプレイスメントスコアを返します。同様に、インスタンス属性を指定する場合は少なくとも 3 つの異なるインスタンスタイプを解決する必要があります。

スポット要件を指定するさまざまな方法の例については設定例を参照してください。

ステップ 2: スポットプレイスメントスコアレスポンスのフィルタリング

Amazon EC2 はリージョンまたはアベイラビリティーゾーンごとにスポットプレイスメントスコアを計算し、スポットリクエストが成功する可能性のある上位 10 のリージョンまたは上位 10 のアベイラビリティーゾーンのいずれかを返します。デフォルトではスコアリングされたリージョンのリストが返されます。すべてのスポット容量を単一のアベイラビリティーゾーンに起動する場合はスコアリングされたアベイラビリティーゾーンのリストをリクエストすると便利です。

リージョンフィルターを指定して、レスポンスで返されるリージョンを絞り込むことができます。

リージョンフィルターとスコア付きアベイラビリティーゾーンのリクエストを組み合わせることができます。このようにして、スコアリングされたアベイラビリティーゾーンはフィルターしたリージョンに限定されます。リージョン内の最高スコアのアベイラビリティーゾーンを検索するにはそのリージョンのみを指定すると、そのリージョン内のすべてのアベイラビリティーゾーンのスコアリストが返されます。

ステップ 3: レコメンデーションを確認する

各リージョンまたはアベイラビリティーゾーンのスポットプレイスメントスコアはターゲット容量、インスタンスタイプの構成、過去および現在のスポット使用傾向、およびリクエストの時間に基づいて計算されます。スポット容量は絶えず変動するため、同じスポット配置スコアのリクエストは異なる時間に計算されたときに異なるスコアを生成する可能性があります。

リージョンとアベイラビリティーゾーンは1 ~ 10 のスケールで採点されます。スコアが 10 の場合はスポットリクエストが成功する可能性が高いが、保証されていないことを示します。スコア 1 はスポットリクエストがまったく成功する可能性がないことを示します。異なるリージョンまたはアベイラビリティーゾーンで同じスコアが返される場合があります。

低スコアが返された場合はコンピューティング要件を編集してスコアを再計算できます。また、同じコンピューティング要件についてスポットプレイスメントスコアのレコメンデーションを 1 日の異なる時間にリクエストすることもできます。

ステップ 4: レコメンデーションを使用する

スポットプレイスメントスコアはスポットリクエストの構成がスポットプレイスメントスコアの構成と全く同じであり (ターゲット容量、ターゲット容量単位、インスタンスタイプまたはインスタンス属性)、`capacity-optimized` 配分戦略を使用するように構成されている場合にのみ意味を持ちます。それ以外の場合、使用可能なスポット容量が得られる可能性はスコアと一致しません。

スポットプレイスメントスコアはガイドラインとして機能し、スポットリクエストの完全または一部が満たされることを保証するスコアはありませんが、次の情報を使用して最良の結果を得ることができます。

- 同じ設定を使用する— スポットプレイスメントスコアはAuto Scaling グループ、EC2 フリート、またはスポットフリートのスポットリクエスト設定 (ターゲット容量、ターゲット容量単位、インスタンスタイプまたはインスタンス属性) がスポットプレイスメントスコアを取得するために入力した内容と同じである場合にのみ関連します。

  スポットプレイスメントスコアリクエストで属性ベースのインスタンスタイプの選択を使用した場合、属性ベースのインスタンスタイプの選択を使用して Auto Scaling グループ、EC2 フリート、またはスポットフリートを設定できます。詳細については「Create mixed instances group using attribute-based instance type selection」と「EC2 フリートまたはスポットフリートのインスタンスタイプを選択するための属性を指定する」を参照してください。

  > **Note**
  >
  > vCPU の数またはメモリ量の観点からターゲット容量を指定し、スポットプレイスメントスコア設定でインスタンスタイプを指定した場合は現在 Auto Scaling グループ、EC2 フリート、またはスポットフリートでこの設定を作成できないことに注意してください。代わりに、インスタンスの重み付けは `WeightedCapacity` パラメータを使用して手動で行います。

- **`capacity-optimized`** 配分戦略を使用する – いずれのスコアも、スポット容量のリクエストが成功するためにはフリートのリクエストがすべてのアベイラビリティーゾーン (リージョン間の容量をリクエストする場合) または単一のアベイラビリティーゾーン (1 つのアベイラビリティーゾーンで容量をリクエストする場合) と `capacity-optimized` スポット配分戦略を使用するように設定されていることを前提としています。`lowest-price` のような他の配分戦略を用いた場合、利用可能なスポット容量を得られる可能性はスコアと一致しません。
- すぐにスコアに基づいて行動する – スポットプレイスメントスコアのレコメンデーションはリクエスト時の利用可能なスポット容量を反映したものであり、スポット容量の変動により、同じ構成でも異なる時期に計算すると異なるスコアになることがあります。スコアが 10 の場合、スポット容量リクエストが成功する可能性が高い (保証はされません) ことを意味しますが、最良の結果を得るにはすぐにスコアに基づいて行動することをお勧めします。また、容量リクエストを試行するたびに新しいスコアを取得することをお勧めします。

スポットプレイスメントスコアに必要なアクセス許可

デフォルトではIAM アイデンティティ (ユーザー、ロール、またはグループ) にはthe section called “スポットプレイスメントスコア”を使用するためのアクセス許可がありません。IAM アイデンティティがスポットプレイスメントスコアを使用できるようにするにはec2:GetSpotPlacementScores EC2 API アクションを使用するためのアクセス許可を付与する IAM ポリシーを作成する必要があります。次に、アクセスを必要とする IAM アイデンティティにポリシーをアタッチします。

すべてのリソースの ec2:GetSpotPlacementScores EC2 API アクションのアクセス権限を付与する IAM ポリシーの例を次に示します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:GetSpotPlacementScores",
            "Resource": "*"
        }
    ]
}
```

IAM ポリシーの編集の詳細についてはIAM ユーザーガイドのIAM ポリシーの編集を参照してください。

アクセス権限を付与するにはユーザー、グループ、またはロールにアクセス許可を追加します。

- AWS IAM アイデンティティセンター のユーザーとグループ:

  アクセス許可セットを作成します。「AWS IAM アイデンティティセンター ユーザーガイド」の「アクセス許可セットを作成する」の手順に従ってください。
- IAM 内で、ID プロバイダーによって管理されているユーザー:

  ID フェデレーションのロールを作成します。詳細については IAM ユーザーガイド の サードパーティー ID プロバイダー (フェデレーション) 用のロールを作成する を参照してください。
- IAM ユーザー:

- ユーザーが担当できるロールを作成します。手順については IAM ユーザーガイド の IAM ユーザーのロールの作成 を参照してください。
- (お奨めできない方法) ポリシーをユーザーに直接アタッチするか、ユーザーをユーザーグループに追加します。詳細については「IAM ユーザーガイド」の「ユーザー (コンソール) へのアクセス権限の追加」を参照してください。

スポットプレイスメントスコアを計算する

スポットプレイスメントスコアはターゲット容量とコンピューティング要件に基づいて計算できます。詳細については「the section called “スポットプレイスメントスコアの仕組み”」を参照してください。

必要な アクセス許可

必要なアクセス許可を持っていることを確認してください。詳細については「the section called “必要なアクセス許可”」を参照してください。

オプション

- インスタンス属性を使用して計算する
- インスタンスタイプを使用して計算する
- AWS CLI を使用して計算する

自動化ソリューションが必要な場合は このユーザーガイドの手動ステップに従うのではなく、スポットプレースメントスコアトラッカーダッシュボードを構築します。スコアを自動的にキャプチャして Amazon CloudWatch に保存できます。詳細については「Guidance for Building a Spot Placement Score Tracker Dashboard on AWS」を参照してください。

インスタンス属性を使用して計算する

インスタンス属性を指定してスポットプレイスメントスコアを計算するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. [スポットインスタンスのリクエスト] の横にある下矢印を選択してから、[スポットプレイスメントスコアの計算] を選択してください。
4. [Enter requirements] (要件の入力) を選択してください。

5. [Target capacity] (ターゲット容量) には希望する容量をインスタンス数や vCPU 数、またはメモリ量 (MiB) で入力してください。
6. [Instance type requirements] (インスタンスタイプの要件) ではコンピューティング要件を指定し、Amazon EC2 にこれらの要件に最適なインスタンスタイプを識別させるために、[Specify instance attributes that match your compute requirements] (コンピューティング要件に一致するインスタンス属性を指定) を選択してください。
7. [vCPUs] に、希望する vCPU の最小数と最大数を入力してください。制限なしを指定するには、[No minimum] (最小値なし)、[No maximum] (最大値なし)、または両方を選択してください。
8. [Memory (GiB)] (メモリ (GiB)) に、希望する最小値と最大値を入力してください。制限なしを指定するには[No minimum] (最小値なし)、[No maximum] (最大値なし)、または両方を選択してください。
9. [CPU architecture] (CPU アーキテクチャ) では必要なインスタンスアーキテクチャを選択してください。
10. (オプション) [Additional instance attributes] (その他のインスタンス属性) ではオプションで 1 つ以上の属性を指定して、コンピューティング要件をより詳細に表現できます。追加の属性はリクエストにさらに制約を追加します。追加の属性は省略できます。省略すると、デフォルト値が使用されます。各属性およびそのデフォルト値の説明については、Amazon EC2 コマンドラインリファレンスの get-spot-placement-scores.を参照してください。
11. (オプション) 指定した属性を持つインスタンスタイプを表示するには[Preview matching instance types] (一致するインスタンスタイプをプレビューする) を展開します。インスタンスタイプを配置評価に使用しないようにするにはインスタンスを選択し、[Exclude selected instance types] (選択されたインスタンスタイプを除外する)を選択してください。
12. [Load placement scores] (プレイスメントスコアのロード) を選択し、結果を確認します。
13. (オプション) 特定のリージョンのスポットプレイスメントスコアを表示するには[Regions to evaluate] (評価するリージョン) で、評価するリージョンを選択し、[Calculate placement scores] (プレイスメントスコアの計算) を選択してください。
14. (オプション) 表示されたリージョンにアベイラビリティーゾーンのスポット配置スコアを表示するには[アベイラビリティーゾーンごとの配置スコアを表示] のチェックボックスをオンにします。スコアリングされたアベイラビリティーゾーンのリストはすべてのスポット容量を 1 つのアベイラビリティーゾーンで起動する場合に便利です。
15. (オプション) コンピューティング要件を編集して新しいプレイスメントスコアを取得するには[Edit] (編集) を選択し、必要な調整を行った後、[Calculate placement scores] (プレイスメントスコアの計算) を選択してください。

インスタンスタイプを使用して計算する

インスタンスタイプを指定してスポットプレイスメントスコアを計算するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. [スポットインスタンスのリクエスト] の横にある下矢印を選択してから、[スポットプレイスメントスコアの計算] を選択してください。
4. [Enter requirements] (要件の入力) を選択してください。
5. [Target capacity] (ターゲット容量) には希望する容量をインスタンス数や vCPU 数、またはメモリ量 (MiB) で入力してください。
6. [Instance type requirements] (インスタンスタイプの要件) では使用するインスタンスタイプを指定するため、[Manually select instance types] (手動でインスタンスタイプを選択する) を選択してください。
7. [Select instance types] (インスタンスタイプを選択) を選択し、使用するインスタンスタイプを選択してから[Select] (選択) を選択してください。インスタンスタイプをすばやく検索するにはフィルターバーを使用して、異なるプロパティでインスタンスタイプをフィルタリングできます。
8. [Load placement scores] (プレイスメントスコアのロード) を選択し、結果を確認します。
9. (オプション) 特定のリージョンのスポットプレイスメントスコアを表示するには[Regions to evaluate] (評価するリージョン) で、評価するリージョンを選択し、[Calculate placement scores] (プレイスメントスコアの計算) を選択してください。
10. (オプション) 表示されたリージョンにアベイラビリティーゾーンのスポット配置スコアを表示するには[アベイラビリティーゾーンごとの配置スコアを表示] のチェックボックスをオンにします。スコアリングされたアベイラビリティーゾーンのリストはすべてのスポット容量を 1 つのアベイラビリティーゾーンで起動する場合に便利です。
11. (オプション) インスタンスタイプのリストを編集して新しいプレイスメントスコアを取得するには[Edit] (編集) を選択し、必要な調整を行ってから [Calculate placement scores] (プレイスメントスコアの計算) を選択してください。

AWS CLI を使用して計算する

スポットプレイスメントスコアを計算するには

1. (オプション) スポットプレイスメントスコアの設定で指定可能なすべてのパラメータを生成するには get-spot-placement-scores コマンドと --generate-cli-skeleton パラメータを使用します。

```
aws ec2 get-spot-placement-scores \
    --region us-east-1 \
    --generate-cli-skeleton
```

以下は出力の例です。

```
{
    "InstanceTypes": [
        ""
    ],
    "TargetCapacity": 0,
    "TargetCapacityUnitType": "vcpu",
    "SingleAvailabilityZone": true,
    "RegionNames": [
        ""
    ],
    "InstanceRequirementsWithMetadata": {
        "ArchitectureTypes": [
            "x86_64_mac"
        ],
        "VirtualizationTypes": [
            "hvm"
        ],
        "InstanceRequirements": {
            "VCpuCount": {
                "Min": 0,
                "Max": 0
            },
            "MemoryMiB": {
                "Min": 0,
                "Max": 0
            },
            "CpuManufacturers": [
                "amd"
```

```
            ],
            "MemoryGiBPerVCpu": {
                "Min": 0.0,
                "Max": 0.0
            },
            "ExcludedInstanceTypes": [
                ""
            ],
            "InstanceGenerations": [
                "previous"
            ],
            "SpotMaxPricePercentageOverLowestPrice": 0,
            "OnDemandMaxPricePercentageOverLowestPrice": 0,
            "BareMetal": "excluded",
            "BurstablePerformance": "excluded",
            "RequireHibernateSupport": true,
            "NetworkInterfaceCount": {
                "Min": 0,
                "Max": 0
            },
            "LocalStorage": "included",
            "LocalStorageTypes": [
                "hdd"
            ],
            "TotalLocalStorageGB": {
                "Min": 0.0,
                "Max": 0.0
            },
            "BaselineEbsBandwidthMbps": {
                "Min": 0,
                "Max": 0
            },
            "AcceleratorTypes": [
                "fpga"
            ],
            "AcceleratorCount": {
                "Min": 0,
                "Max": 0
            },
            "AcceleratorManufacturers": [
                "amd"
            ],
            "AcceleratorNames": [
                "vu9p"
```

```
            ],
            "AcceleratorTotalMemoryMiB": {
                "Min": 0,
                "Max": 0
            }
        }
    },
    "DryRun": true,
    "MaxResults": 0,
    "NextToken": ""
}
```

2. 前のステップの出力を使用して JSON 設定ファイルを作成し、次のように設定します。

   a. TargetCapacity には希望するスポット容量をインスタンス数や vCPU 数、またはメモリ量 (MiB) で入力してください。

   b. TargetCapacityUnitType に、ターゲット容量の単位を入力してください。このパラメータを省略すると、デフォルトで units になります。

      有効な値:units (インスタンス数に変換されます) | vcpu | memory-mib

   c. スコアリングされたアベイラビリティーゾーンのリストを返すレスポンスのため、SingleAvailabilityZone に true を指定します。スコアリングされたアベイラビリティーゾーンのリストはすべてのスポット容量を 1 つのアベイラビリティーゾーンで起動する場合に便利です。このパラメータを省略すると、デフォルトで false となり、レスポンスはスコアリングされたリージョンのリストを返します。

   d. (オプション) RegionNames で、フィルターとして使用するリージョンを指定します。リージョンコードを指定する必要があります (例: us-east-1)。

      リージョンフィルターを使用すると、レスポンスは指定したリージョンのみを返します。true で SingleAvailabilityZone を指定した場合は指定したリージョンのアベイラビリティーゾーンのみを返します。

   e. 同じ設定に InstanceTypes または InstanceRequirements を含めることができますが、両方を含めることはできません。

      JSON 設定で、次のいずれかを指定します。

      - インスタンスタイプのリストを指定するにはInstanceTypes パラメータでインスタンスタイプを指定します。少なくとも 3 つの異なるインスタンスタイプを指定します。1 つまたは 2 つのインスタンスタイプのみを指定した場合、スポットプレイスメントスコア

は低スコアを返します。インスタンスタイプのリストについてはAmazon EC2 インスタンスタイプ を参照してください。

- Amazon EC2 がこれらの属性に一致するインスタンスタイプを識別するように、インスタンスの属性を指定するにはInstanceRequirements 構造内にある属性を指定します。

  VCpuCount、MemoryMiB および CpuManufacturers の値を指定する必要があります。その他の属性は省略できます。省略すると、デフォルト値が使用されます。各属性およびそのデフォルト値の説明についてはAmazon EC2 コマンドラインリファレンスの get-spot-placement-scores を参照してください。

設定例については設定例を参照してください。

3. JSON ファイルで指定した条件のスポットプレイスメントスコアを取得するにはget-spot-placement-scores コマンドを使用し、--cli-input-json パラメータで JSON ファイルの名前とパスを指定します。

```
aws ec2 get-spot-placement-scores \
    --region us-east-1 \
    --cli-input-json file://file_name.json
```

SingleAvailabilityZone が false に設定されているか、省略されている場合の出力例 (省略されている場合のデフォルトは false) – リージョンのスコアリングされたリストが返されます。

```
"SpotPlacementScores": [
    {
        "Region": "us-east-1",
        "Score": 7
    },
    {
        "Region": "us-west-1",
        "Score": 5
    },
   ...
```

SingleAvailabilityZone が true に設定されている場合の出力例 – アベイラビリティーゾーンのスコアリストが返されます。

```
"SpotPlacementScores": [
    {
        "Region": "us-east-1",
        "AvailabilityZoneId": "use1-az1",
        "Score": 8
    },
    {
        "Region": "us-east-1",
        "AvailabilityZoneId": "usw2-az3",
        "Score": 6
    },
   ...
```

設定例

AWS CLI を使用する場合、次の設定例を使用できます。

設定例

- 例: インスタンスタイプとターゲット容量の指定
- 例: メモリの観点からインスタンスタイプとターゲット容量を指定する
- 例: 属性ベースのインスタンスタイプ選択の属性を指定する
- 例: 属性ベースのインスタンスタイプ選択の属性を指定し、アベイラビリティーゾーンのスコアリストを返す

例: インスタンスタイプとターゲット容量の指定

次の設定例では3 つの異なるインスタンスタイプと 500 スポットインスタンスのターゲットスポット容量を指定します。

```
{
    "InstanceTypes": [
        "m5.4xlarge",
        "r5.2xlarge",
        "m4.4xlarge"
    ],
    "TargetCapacity": 500
}
```

例: メモリの観点からインスタンスタイプとターゲット容量を指定する

次の設定例では3 つの異なるインスタンスタイプとターゲットスポット容量 500,000 MiB のメモリを指定します。この場合、起動するスポットインスタンスの数は合計 500,000 MiB のメモリを提供する必要があります。

```
{
    "InstanceTypes": [
        "m5.4xlarge",
        "r5.2xlarge",
        "m4.4xlarge"
    ],
    "TargetCapacity": 500000,
    "TargetCapacityUnitType": "memory-mib"
}
```

例: 属性ベースのインスタンスタイプ選択の属性を指定する

次の設定例は属性ベースのインスタンスタイプ選択用に設定され、その後に設定例の説明が記載されています。

```
{
    "TargetCapacity": 5000,
    "TargetCapacityUnitType": "vcpu",
    "InstanceRequirementsWithMetadata": {
        "ArchitectureTypes": ["arm64"],
        "VirtualizationTypes": ["hvm"],
        "InstanceRequirements": {
            "VCpuCount": {
                "Min": 1,
                "Max": 12
            },
            "MemoryMiB": {
                "Min": 512
            }
        }
    }
}
```

InstanceRequirementsWithMetadata

属性ベースのインスタンスタイプの選択を使用するには設定に InstanceRequirementsWithMetadata 構造を含め、スポットインスタンスに必要な属性を指定します。

前の例で、次の必須インスタンス属性を指定しています。

- ArchitectureTypes – インスタンスタイプのアーキテクチャタイプは arm64 である必要があります。
- VirtualizationTypes – インスタンスタイプの仮想化タイプは hvm である必要があります。
- VCpuCount – インスタンスタイプには1 個以上、最大 12 個の vCPU が必要です。
- MemoryMiB – インスタンスタイプには最低 512 MiB のメモリが必要です。Max パラメータを省略した場合、上限がないことを示します。

指定できるオプションの属性は他にもいくつかあります。属性のリストについては「 コマンドラインリファレンス」の get-spot-placement-scores を参照してください。

TargetCapacityUnitType

TargetCapacityUnitType パラメータはターゲット容量の単位を指定します。この例ではターゲット容量が5000、ターゲット容量単位のタイプが vcpu となっており、合わせて 5000 vCPUの希望ターゲット容量が指定されており、起動するスポットインスタンスの数で合計 5000 vCPU を提供する必要があります。

例: 属性ベースのインスタンスタイプ選択の属性を指定し、アベイラビリティーゾーンのスコアリストを返す

次の設定例は属性ベースのインスタンスタイプ選択用に設定されています。"SingleAvailabilityZone": true を指定した場合、レスポンスはスコアリングされたアベイラビリティーゾーンのリストを返します。

```
{
    "TargetCapacity": 1000,
    "TargetCapacityUnitType": "vcpu",
    "SingleAvailabilityZone": true,
    "InstanceRequirementsWithMetadata": {
        "ArchitectureTypes": ["arm64"],
        "VirtualizationTypes": ["hvm"],
        "InstanceRequirements": {
            "VCpuCount": {
                "Min": 1,
```

```
                "Max": 12
            },
            "MemoryMiB": {
                "Min": 512
            }
        }
    }
}
```

スポットインスタンスのデータフィードを使用してスポットインスタンスのコストを追跡する

スポットインスタンスの料金について理解しやすくするため、Amazon EC2 ではスポットインスタンスの使用状況と料金の詳細を、データフィードにより提供しています。このデータフィードはデータフィードを購読するときに指定する Amazon S3 バケットに送信されます。

データフィードファイルは通常 1 時間に 1 回バケットに到達します。スポットインスタンス実行が一定の時間に満たない場合、その時間のデータフィードファイルは送信されません。

通常、スポットインスタンスの使用時間は 1 時間ごとに 1 つのデータファイルでカバーされます。これらのファイルはユーザーのバケットに配信される前に圧縮 (gzip) されます。ファイルが大きい場合 (ある時間に関するファイルの内容が、圧縮前に 50 MB を超える場合など) はAmazon EC2 は指定した時間の使用状況に関する情報を複数のファイルに書き込みます。

> **Note**
>
> 1 つの AWS アカウントにつき 1 つのスポットインスタンスデータフィードのみを作成できます。

スポットインスタンスのデータフィードはAWS中国 (北京)、中国 (寧夏)、AWSGovCloud (米国) 以外のすべてのリージョンおよびデフォルトでは無効になっているリージョンでサポートされています。

内容

- データフィードのファイル名と形式
- Amazon S3 バケットの要件
- スポットインスタンスのデータフィードの購読
- データフィード内のデータを表示する

- スポットインスタンスのデータフィードを削除する

データフィードのファイル名と形式

スポットインスタンスのデータフィードのファイル名には(UTC の日付と時刻を使用しながら)次のような形式が使用されます。

```
bucket-name.s3.amazonaws.com/optional-prefix/aws-account-id.YYYY-MM-DD-HH.n.unique-id.gz
```

例えば、バケット名が **amzn-s3-demo-bucket** で、プレフィクスが **my-prefix** である場合、ファイル名は次のようになります。

```
amzn-s3-demo-bucket.s3.amazonaws.com/my-prefix/111122223333.2023-12-09-07.001.b959dbc6.gz
```

バケット名の詳細については「Amazon S3 ユーザーガイド」の「Bucket naming rules」を参照してください。

スポットインスタンスのデータフィードファイルはタブ区切りです。データファイルの各行はそれぞれ 1 時間のインスタンス使用に相当し、次の表に示すフィールドを含みます。

フィールド	説明
Timestamp	そのインスタンス使用量に対して請求される価格を決定するために使用されるタイムスタンプ。
UsageType	請求の対象となっている使用タイプおよびインスタンスタイプ。m1.small スポットインスタンス ではこのフィールドは SpotUsage に設定されます。他のすべてのインスタンスタイプではこのフィールドは SpotUsage:{instance-type} に設定されます。例えば、SpotUsage:c1.medium。
Operation	請求の対象となっている製品。Linux スポットインスタンス の場合、このフィールドは RunInstances に設定されます。Windows スポットインスタンス の場合、このフィールドは RunInstances:0002 に設定されます。スポット使用状況は利用可能ゾーンに従ってグループ化されます。

フィールド	説明
InstanceID	このインスタンスの使用量情報を生成したスポットインスタンスの ID。
MyBidID	このインスタンスの使用量情報を生成したスポットインスタンスリクエストの ID。
MyMaxPrice	このスポットリクエストに指定された上限価格。
MarketPrice	Timestamp フィールドに指定された時刻のスポット料金。
Charge	このインスタンス使用量に請求される価格。
Version	データフィードバージョン。可能性のあるバージョンは 1.0 です。

Amazon S3 バケットの要件

データフィードの購読時に、データフィードファイルを格納する Amazon S3 バケットを指定する必要があります。

データフィード用の Amazon S3 バケットを選択する前に、以下の点を考慮します。

- バケットに対する FULL_CONTROL アクセス権限が必要です。バケット所有者にはデフォルトでこの権限があります。それ以外の場合、バケット所有者は AWS アカウントにこのアクセス権限を付与する必要があります。
- データフィードを購読すると、これらのアクセス権限を使用してバケット ACL が更新され、AWS データフィードアカウントに FULL_CONTROL アクセス権限が付与されます。AWS データフィードアカウントはデータフィードファイルをバケットに書き込みます。アカウントに必要なアクセス許可がない場合、データフィードファイルをバケットに書き込むことはできません。詳細については「Amazon CloudWatch Logs ユーザーガイド」の「Amazon S3 に送信された口グ」を参照してください。

  ACL を更新して AWS データフィードアカウントのアクセス権限を削除すると、データフィードファイルをバケットに書き込むことができなくなります。データフィードファイルを受け取るにはデータフィードを再購読する必要があります。

- 各データフィードファイルには独自の ACL があります (バケットの ACL とは別です)。バケット所有者にはデータファイルに対して FULL_CONTROL のアクセス許可があります。AWS データフィードアカウントには読み書きのアクセス権限があります。
- データフィードの購読を削除しても、Amazon EC2 はAWS データフィードアカウントでの、バケットまたはデータファイルに対する読み書きのアクセス許可を削除しません。これらのアクセス許可は自分で削除する必要があります。
- AWS Key Management Service (SSE-KMS) に保存されている AWS KMS キーによるサーバー側の暗号化を使用して Amazon S3 バケットを暗号化する場合はカスタマーマネージド型キーを使用する必要があります。詳細については「Amazon CloudWatch Logs ユーザーガイド」の「Simple Storage Service (Amazon S3) バケットのサーバー側の暗号化」を参照してください。

スポットインスタンスのデータフィードの購読

スポットインスタンスのデータフィードはいつでも購読できます。Amazon EC2 コンソールを使用してこのタスクを完了することはできません。

バケットに十分なアクセス許可がないというエラーが表示された場合は「Troubleshoot the data feed for Spot Instances」の記事でトラブルシューティングに関する情報を参照してください。。

AWS CLI

データフィードを購読するには

create-spot-datafeed-subscription コマンドを使用します。

プレフィックスを使用してバケットを指定するには、次の例を使用します。

```
aws ec2 create-spot-datafeed-subscription \
    --bucket amzn-s3-demo-bucket \
    --prefix my-prefix
```

プレフィックスを使用せずにバケットを指定するには、次の例を使用します。

```
aws ec2 create-spot-datafeed-subscription \
    --bucket amzn-s3-demo-bucket
```

PowerShell

データフィードを購読するには

New-EC2SpotDatafeedSubscription コマンドレットを使用します。

プレフィックスを使用してバケットを指定するには、次の例を使用します。

```
New-EC2SpotDatafeedSubscription `
    -Bucket amzn-s3-demo-bucket `
    -Prefix my-prefix
```

プレフィックスを使用せずにバケットを指定するには、次の例を使用します。

```
New-EC2SpotDatafeedSubscription `
    -Bucket amzn-s3-demo-bucket
```

データフィード内のデータを表示する

AWS マネジメントコンソールで AWS CloudShell を開きます。次の s3 sync コマンドを使用して、データフィードの S3 バケットから .gz ファイルを取得し、指定したフォルダに保存します。

```
aws s3 sync s3://amzn-s3-demo-bucket ./data-feed
```

.gz ファイルの内容を表示するにはS3 バケットの内容を保存したフォルダに移動します。

```
cd data-feed
```

ls コマンドを使用してファイルの名前を表示します。zcat コマンドをファイルの名前と共に使用すると、圧縮ファイルの内容が表示されます。以下にサンプルコマンドを示します。

```
zcat  111122223333.2023-12-09-07.001.b959dbc6.gz
```

以下は出力の例です。

```
#Version: 1.0
#Fields: Timestamp UsageType Operation InstanceID MyBidID MyMaxPrice MarketPrice Charge
 Version
2023-12-09 07:13:47 UTC USE2-SpotUsage:c7a.medium       RunInstances:SV050
 i-0c3e0c0b046e050df     sir-pwq6nmfp    0.0510000000 USD        0.0142000000 USD
  0.0142000000 USD        1
```

スポットインスタンスのデータフィードを削除する

スポットインスタンスのデータフィードが終了したら、データフィードを削除できます。

AWS CLI

データフィードを削除するには

delete-spot-datafeed-subscription コマンドを使用します。

```
aws ec2 delete-spot-datafeed-subscription
```

PowerShell

データフィードを削除するには

Remove-EC2SpotDatafeedSubscription コマンドレットを使用します。

```
Remove-EC2SpotDatafeedSubscription
```

スポットインスタンスリクエスト向けのサービスにリンクされたロール

Amazon EC2 はユーザーに代わって AWS の他のサービスを呼び出すために必要なアクセス許可のために、サービスにリンクされたロールを使用します。サービスにリンクされたロールはAWS のサービス に直接リンクされた一意のタイプの IAM ロールです。サービスにリンクされたロールはAWS のサービスにアクセス許可を委任するためのセキュアな方法を提供します。これはリンクされたサービスのみが、サービスにリンクされたロールを引き受けることができるためです。詳細については「IAM ユーザーガイド」の「Service-linked roles」を参照してください。

Amazon EC2 はAWSServiceRoleForEC2Spot という、サービスにリンクされたロールを使用して、ユーザーの代わりに スポットインスタンス を起動して管理します。

AWSServiceRoleForEC2Spot によって付与されるアクセス許可

Amazon EC2 はAWSServiceRoleForEC2Spot という、サービスにリンクされたロールを使用して、次のアクションを実行します。

- `ec2:DescribeInstances` – スポットインスタンスの記述
- `ec2:StopInstances` – スポットインスタンスの停止

- `ec2:StartInstances` – スポットインスタンスの開始

サービスにリンクされたロールの作成

ほとんどの状況ではサービスにリンクされたロールを手動で作成する必要はありません。Amazon EC2 はユーザーがコンソールを使用して初めてスポットインスタンスをリクエストした際に、サービスにリンクされたロール AWSServiceRoleForEC2Spot を作成します。

Amazon EC2 がこのサービスにリンクされたロールのサポートを開始した 2017 年 10 月よりも前に、ユーザーがアクティブなスポットインスタンスリクエストを行っている場合はAmazon EC2 により AWSServiceRoleForEC2Spot ロールが AWS アカウントに作成されています。詳細についてはIAM ユーザーガイドのアカウントに新しいロールが表示されるを参照してください。

AWS CLI または API を使用してスポットインスタンスをリクエストするにはまずこのロールが存在していることを確認する必要があります。

コンソールを使用して AWSServiceRoleForEC2Spot を作成するには

1. IAM コンソールの https://console.aws.amazon.com/iam/ を開いてください。
2. ナビゲーションペインで Roles (ロール) を選択してください。
3. [ロールの作成] を選択してください。
4. [Select type of trusted entity (信頼されたエンティティのタイプを選択)] ページで、[EC2]、[EC2 - Spot Instances (EC2 - スポットインスタンス)]、[Next: Permissions (次の手順: アクセス許可)] の順に選択してください。
5. 次のページで、[次へ: 確認] を選択してください。
6. [確認] ページで、[ロールの作成] を選択してください。

AWS CLI を使用して AWSServiceRoleForEC2Spot を作成するには

次のように、create-service-linked-role コマンドを使用します。

```
aws iam create-service-linked-role --aws-service-name spot.amazonaws.com
```

スポットインスタンス を使用する必要がなくなった場合は[AWSServiceRoleForEC2Spot] ロールを削除することをお勧めします。このロールがアカウントから削除された後で、Amazon EC2 をリクエストすると、スポットインスタンス はロールを再度作成します。

暗号化された AMI および EBS スナップショット用のカスタマーマネージド型キーへのアクセス権限の付与

スポットインスタンスのために暗号化された AMI または暗号化された Amazon EBS スナップショットを指定しており、カスタマーマネージド型キーを暗号化に使用する場合はAmazon EC2 がユーザーに代わってスポットインスタンスを起動できるようにするために、カスタマーマネージド型キーを使用する許可を AWSServiceRoleForEC2Spot ロールにより付与する必要があります。これを行うには次の手順で示すように、カスタマーマネージド型キーに対し付与を追加する必要があります。

アクセス権限を設定するときは付与がキーポリシーの代わりになります。詳細についてはAWS Key Management Service デベロッパーガイドの付与の使用とAWS KMS でのキーポリシーの使用を参照してください。

AWSServiceRoleForEC2Spot ロールにカスタマーマネージド型キーを使用する許可を付与するには

- create-grant コマンドを使用してカスタマーマネージド型キーに付与を追加し、プリンシパル (サービスにリンクされたロールの AWSServiceRoleForEC2Spot) を指定します。このプリンシパルには付与が許可するオペレーションを実行するためのアクセス許可が含まれています。カスタマーマネージド型キーはkey-id パラメータと、そのカスタマーマネージド型キーの ARN により指定します。プリンシパルを指定するにはgrantee-principal パラメータとサービスにリンクされたロール AWSServiceRoleForEC2Spot の ARN を使用します。

```
aws kms create-grant \
    --region us-east-1 \
    --key-id arn:aws:kms:us-
east-1:444455556666:key/1234abcd-12ab-34cd-56ef-1234567890ab \
    --grantee-principal arn:aws:iam::111122223333:role/aws-service-role/
spot.amazonaws.com/AWSServiceRoleForEC2Spot \
    --operations "Decrypt" "Encrypt" "GenerateDataKey"
 "GenerateDataKeyWithoutPlaintext" "CreateGrant" "DescribeKey" "ReEncryptFrom"
 "ReEncryptTo"
```

Spot Instance クォータ

実行中および要求されたスポットインスタンスの数、そして保留中のスポット インスタンスの数には各リージョンにつき AWS アカウント ごとに割り当てがあります。保留中のスポットインスタンスリクエストが受理されると、実行中のインスタンスがクォータにカウントされるため、リクエストはクォータにカウントされなくなります。

スポットインスタンスのクォータは仮想中央演算装置 (vCPU) の数について管理されます。この数は実行中のスポットインスタンスが使用中であるか、未処理のスポットインスタンスリクエストの受理が保留中であるため、後に使用されるかにより決定されます。ユーザーがスポットインスタンスを終了しており、かつスポットインスタンスリクエストをキャンセルしていない場合、Amazon EC2 がスポットインスタンスの終了を検出してリクエストを閉じるまで、リクエストはスポットインスタンスでの vCPU のクォータ数についてカウントされます。

スポットインスタンスには以下のクォータタイプが用意されています。

名前	デフォルト	引き上げ可能
オール DL スポットインスタンスリクエスト	0	はい
すべての F スポットインスタンスリクエスト	0	はい
オール G および VT スポットインスタンスリクエスト	0	はい
すべての Inf スポットインスタンスリクエスト	0	はい
すべての P スポットインスタンスリクエスト	0	はい
すべての標準 (A、C、D、H、I、M、R、T、Z) スポットインスタンスリクエスト	5	はい
すべての Trn スポットインスタンスリクエスト	0	はい
すべての X スポットインスタンスリクエスト	0	はい

Amazon EC2 は使用量に基づいてスポットインスタンスのクオータを自動的に調整しますが、必要に応じてクォータの引き上げをリクエストすることができます。例えば、現在のクォータで許可されているよりも多くの スポットインスタンス を起動する場合はクォータの引き上げをリクエストできます。また、スポットインスタンスリクエストを送信した後にエラー `Max spot instance count exceeded` を受け取ったとしても、クォータの引き上げをリクエストできます。クォータの増加を要求するには Amazon EC2 の Service Quotas で説明されている Service Quotas コンソールを使用します。

変化するアプリケーションのニーズに合わせて、任意の組み合わせのインスタンスタイプを起動できます。例えば、オールスタンダードスポットインスタンスリクエストのクォータが 256 vCPU の場

合、32 m5.2xlarge 個のスポットインスタンス (32 x 8 vCPU) または16 c5.4xlarge 個のスポットインスタンス (16 x 16 vCPU) をリクエストできます。

Amazon CloudWatch のメトリクス統合ではクォータに対して EC2 の使用量をモニタリングできます。クォータに近づいたときに警告を発するようにアラームを設定することもできます。詳細については、「Service Quotas ユーザーガイド」の「Service Quotas and Amazon CloudWatch alarms」およびを参照してください。

Amazon EC2 専有ホスト

Amazon EC2 Dedicated Host は完全にお客様専用の物理サーバーです。オプションで、インスタンス容量を他の AWS アカウントと共有することもできます。詳細については「クロスアカウント Amazon EC2 専有ホストの共有」を参照してください。

専有ホストは、インスタンスの配置を可視化および制御し、ホストアフィニティをサポートします。つまり、特定のホストでインスタンスを起動して実行でき、インスタンスが特定のホストでのみ実行されるようにできます。詳細については「Amazon EC2 専有ホストの自動配置とホストアフィニティ」を参照してください。

専有ホストは、包括的な Bring-Your-Own-License (BYOL) サポートを提供します。これにより、Windows Server、SQL Server、SUSE Linux Enterprise Server、Red Hat Enterprise Linux、または VM、ソケット、または物理コアにバインドされているその他のソフトウェアライセンスを含む、既存のソケット単位、コア単位、または VM 単位のソフトウェアライセンスをライセンス条項に従って使用できます。

インスタンスを専用ハードウェアで実行する必要があるが、インスタンスの配置を可視化または制御する必要はなく、ソケット単位またはコア単位のソフトウェアライセンスを使用する必要がない場合は、代わりにハードウェア専有インスタンスを使用することを検討できます。ハードウェア専有インスタンスと専有ホストのどちらを使用しても、専用の物理サーバーに Amazon EC2 インスタンスを起動することができます。ハードウェア専有インスタンス と Dedicated Hosts のインスタンスの間に、パフォーマンス、セキュリティ、または物理的な違いはありません。ただし、これらにはいくつかの重要な違いがあります。次のテーブルでは、Dedicated Hosts とハードウェア専有インスタンスの主な違いをいくつか紹介します。

	Dedicated Host	ハードウェア専有インスタンス
専用物理サーバー	お客様専用のインスタンス容量を持つ物理サーバー。	単一の顧客アカウント専用の物理サーバー。

	Dedicated Host	ハードウェア専有インスタンス
インスタンス容量の共有	インスタンス容量を他のアカウントと共有できます。	サポートされていません
請求	ホストごとの請求	インスタンスごとの請求
ソケット、コア、ホスト ID の可視性	ソケットと物理コアの数が見える	可視性なし
ホストおよびインスタンスアフィニティ	インスタンスを同じ物理サーバーに徐々にデプロイし続けることができる	サポート外
ターゲットを絞ったインスタンスの配置	インスタンスを物理サーバーに配置する方法についての可視性と制御が高い	サポート外
インスタンスの自動復旧	サポート対象。詳細については「Amazon EC2 専有ホストリカバリ」を参照してください。	サポート対象
Bring-Your-Own-License (BYOL)	サポート	部分的なサポート*
キャパシティ予約	サポート外	サポート

* ソフトウェアアシュアランスによるライセンスモビリティを使用する Microsoft SQL Server、および Windows Virtual Desktop Access (VDA) ライセンスを、ハードウェア専有インスタンスで使用することが可能です。

専有インスタンスの詳細については、「Amazon EC2 ハードウェア専有インスタンス」を参照してください。

Dedicated Hosts の制約事項

Dedicated Hosts を割り当てる際は、次の制限と制約に注意してください。

- 専有ホストで RHEL および SUSE Linux を実行するには、独自の AMI を使用する必要があります。AWS が提供している、あるいは AWS Marketplace から入手が可能な RHEL および SUSE Linux の AMI は、専有ホストでは使用できません。独自の AMI を作成する方法の詳細については、「独自のソフトウェアライセンスを Amazon EC2 専有ホストで使用する」を参照してください。

  この制限は、ハイメモリインスタンス (`u-6tb1.metal`、`u-9tb1.metal`、`u-12tb1.metal`、`u-18tb1.metal`、および `u-24tb1.metal`) に割り当てられたホストには適用されません。AWS によって提供される、または AWS Marketplace で利用できる RHEL および SUSE Linux AMI は、これらのホストで使用できます。
- インスタンスファミリーごとの Dedicated Hosts の実行数には、リージョンごとに AWS アカウントあたりの上限があります。クォータは実行中のインスタンスにのみ適用されます。インスタンスが保留中、停止処理中、停止済みの場合、クォータにはカウントされません。アカウントのクォータを確認する、または引き上げをリクエストするには、Service Quotas コンソールを使用してください。
- Auto Scaling グループは、ホストリソースグループを指定する起動テンプレートを使用する場合にサポートされます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「詳細設定を使用して起動テンプレートを作成する」を参照してください。
- Amazon RDS インスタンスはサポートされません。
- AWS 無料利用枠は Dedicated Hosts ではご使用になれません。
- インスタンスのプレイスメント制御は、Dedicated Hosts でのインスタンスの起動管理を表します。専有ホストをプレイスメントグループで起動することはできません。
- 仮想インスタンスタイプにホストを割り当てる場合、ホストの割り当て後にインスタンスタイプを `.metal` インスタンスタイプに変更することはできません。例えば、`m5.large` インスタンスタイプにホストを割り当てた場合、インスタンスタイプを `m5.metal` に変更することはできません。

  同様に、`.metal` インスタンスタイプにホストを割り当てる場合、ホストの割り当て後にインスタンスタイプを仮想インスタンスタイプに変更することはできません。例えば、`m5.metal` インスタンスタイプにホストを割り当てた場合、インスタンスタイプを `m5.large` に変更することはできません。

内容

- Amazon EC2 専有ホストの料金および請求
- Amazon EC2 専有ホストインスタンスのキャパシティ設定
- Amazon EC2 専有ホスト上のバースト可能な T3 インスタンス
- 独自のソフトウェアライセンスを Amazon EC2 専有ホストで使用する
- Amazon EC2 専有ホストの自動配置とホストアフィニティ
- アカウントで使用するために Amazon EC2 専有ホストを割り当てる
- Amazon EC2 専有ホストで Amazon EC2 インスタンスを起動する
- ホストリソースグループに Amazon EC2 インスタンスを起動する
- 既存の Amazon EC2 専有ホストの自動配置設定を変更する
- 既存の Amazon EC2 専有ホストでサポートされているインスタンスタイプを変更する
- Amazon EC2 インスタンスの Amazon EC2 専有ホストのテナンシーとアフィニティを変更する
- Amazon EC2 専有ホストを解放する
- Nitro ベースの Amazon EC2 Dedicated Hosts に移行する
- 専有ホストの予約を購入して、専有ホストの請求で割引を受ける
- クロスアカウント Amazon EC2 専有ホストの共有
- AWS Outposts の Amazon EC2 専有ホスト
- Amazon EC2 専有ホストリカバリ
- Amazon EC2 専有ホストのホストメンテナンス
- Amazon EC2 専有ホストの状態をモニタリングする
- AWS Config を使用して、Amazon EC2 専有ホストでの設定の変更を追跡する

Amazon EC2 専有ホストの料金および請求

Dedicated Host の料金は支払いオプションごとに異なります。

支払いオプション

- オンデマンド Dedicated Hosts
- Dedicated Host Reservations
- Savings Plans
- Dedicated Hosts での Windows サーバーの料金

オンデマンド Dedicated Hosts

アカウントに Dedicated Host を割り当てると、自動的にオンデマンド請求がアクティブになります。

Dedicated Host のオンデマンド価格は、インスタンスファミリーとリージョンによって異なります。起動するインスタンスの数量やサイズに関係なく、アクティブな Dedicated Host に対して 1 秒あたり (最低 60 秒) の料金が発生します。オンデマンド料金の詳細については、Amazon EC2 Dedicated Hosts オンデマンド料金を参照してください。

オンデマンド専有ホストはいつでもリリースして、料金の発生を止めることができます。Dedicated Host の解放の詳細については、Amazon EC2 専有ホストを解放するを参照してください。

Dedicated Host Reservations

Dedicated Host の予約 では、オンデマンド Dedicated Hosts の実行と比べて請求の割引が得られます。予約は、3 つの支払いオプションで利用できます。

- 前払いなし — 前払いなしの予約では、期間内の Dedicated Host の使用に対して割引があり、前払い料金は必要ありません。1 年および 3 年契約で利用できます。前払いなしの予約での 3 年契約は、一部のインスタンスファミリーのみでサポートされます。
- 一部前払い — 予約の一部を前払いする必要があり、期間内の残りの時間は割引された時間料金で請求されます。1 年および 3 年契約で利用できます。
- 全額前払い — 実質的に最低価格で利用できます。1 年および 3 年契約で利用でき、期間中のすべてのコストが含まれます。それ以外の料金は発生しません。

予約を購入するには、アカウントでアクティブな Dedicated Hosts が必要です。各予約では、単一のアベイラビリティーゾーンで同じインスタンスファミリーをサポートしている、複数のホストに対応できます。予約は、インスタンスサイズではなくホストのインスタンスファミリーに適用されます。インスタンスサイズが異なる 3 つの Dedicated Hosts (`m4.xlarge`、`m4.medium`、および `m4.large`) がある場合、1 つの `m4` 予約をこれらすべての Dedicated Hosts に関連付けることができます。予約のインスタンスファミリーとアベイラビリティーゾーンは、関連付ける Dedicated Hosts のインスタンスファミリーとアベイラビリティーゾーンに一致させる必要があります。

予約が専有ホストに関連付けられている場合、専有ホストは予約期間が終了するまでリリースできません。

予約の料金の詳細については、Amazon EC2 Dedicated Hosts 料金表を参照してください。

Savings Plans

Savings Plans は、オンデマンドインスタンスと比べて大幅に料金を節約できる柔軟な料金モデルです。Savings Plans は、1～3 年間は時間あたりの USD 建て料金で一定量の使用を継続するという確約を条件とする割引プランです。これによって、特定のDedicated Hostを使用することをコミットせずに、ニーズに最も適したDedicated Hostsを使用して、コストを削減し続ける柔軟性が得られます。詳細については、AWS Savings Plans ユーザーガイドをご参照ください。

> **Note**
>
> Savings Plans は、`u-6tb1.metal`、`u-9tb1.metal`、`u-12tb1.metal`、`u-18tb1.metal`、および `u-24tb1.metal` の Dedicated Hosts ではサポートされていません。

Dedicated Hosts での Windows サーバーの料金

Microsoft のライセンス条項に抵触しなければ、お手元にすでにある Windows Server や SQL Server のライセンスを Dedicated Hosts に移すことができます。ご自分のライセンスを使用する場合、追加のソフトウェア使用料は発生しません。

さらに、Amazon が提供する Windows Server AMI を使用して、最新バージョンの Windows Server を Dedicated Hosts で実行できます。これは、Dedicated Hosts 上で実行できる既存の SQL Server のライセンスは持っているが、SQL Server ワークロードを実行するために Windows Server を必要としている場合に、広く見られることです。Amazon が提供する Windows Server AMI のサポートは、現行のインスタンスタイプにのみ適用されます。詳細については、Amazon EC2 Dedicated Hosts の料金を参照してください。

Amazon EC2 専有ホストインスタンスのキャパシティ設定

Dedicated Hosts はさまざまな構成 (物理コア、ソケット、vCPU) をサポートしているため、さまざまなファミリーやサイズのインスタンスを実行できます。

アカウントに 専有ホスト を割り当てる場合、単一のインスタンスタイプ、または同じインスタンスファミリー内の複数のインスタンスタイプをサポートする構成を選択できます。ホストで実行できるインスタンスの数は、選択した設定によって異なります。

内容

- 単一インスタンスタイプのサポート
- 複数のインスタンスタイプのサポート

単一インスタンスタイプのサポート

1 つのインスタンスタイプのみをサポートする専有ホストを割り当てることができます。この設定では、専有ホストで起動するすべてのインスタンスは、ホストを割り当てるときに指定する同じインスタンスタイプである必要があります。

例えば、m5.4xlarge インスタンスタイプのみをサポートするホストを割り当てることができます。この場合、そのホスト上で実行できるインスタンスは m5.4xlarge のみです。

ホスト上で起動できるインスタンスの数は、ホストが提供する物理コアの数と、指定されたインスタンスタイプによって消費されるコア数によって異なります。例えば、m5.4xlarge インスタンスにホストを割り当てると、ホストは 48 個の物理コアを提供し、各 m5.4xlarge インスタンスは 8 個の物理コアを消費します。つまり、そのホストでは最大 6 つのインスタンスを起動できます (48 物理コア/インスタンスあたり 8 コア = 6 インスタンス)。

複数のインスタンスタイプのサポート

同じインスタンスファミリー内の複数のインスタンスタイプをサポートする専有ホストを割り当てることができます。これにより、同じインスタンス ファミリー内にあり、ホストに十分なインスタンスキャパシティがある限り、同じホスト上で異なるインスタンスタイプを実行できます。

例えば、R5 インスタンス ファミリー内のさまざまなインスタンスタイプをサポートするホストを割り当てることができます。この場合は、そのホスト上で、ホストの物理コア容量内で、r5.large、r5.xlarge、r5.2xlarge、r5.4xlarge などの R5 インスタンスタイプの特定の組み合わせを起動できます。

次のインスタンスファミリーは、複数のインスタンスタイプをサポートする Dedicated Hosts をサポートしています。

- 一般的な用途: A1 | M5 | M5n | M6i | M7i | T3
- コンピューティングの最適化: C5 | C5n | C6i | C7i
- メモリ最適化: R5 | R5b | R5n | R6i | R7i | R7iz | X2iedn

ホストで実行できるインスタンスの数は、ホストが提供する物理コア数、およびホスト上で実行する各インスタンスタイプによって消費されるコア数によって異なります。例えば、48 個の物理コアを提供する R5 ホストを割り当て、2 つの r5.2xlarge インスタンス (4 コア x 2 インスタンス) と 3 つの r5.4xlarge インスタンス (8 コア x 3 インスタンス) を実行するとします。これらのインスタンスは合計 32 コアを消費します。残りの 16 コア内であれば、R5 インスタンスの特定の組み合わせを実行できます。

ただし、各インスタンスタイプで実行可能な上限のインスタンス数は、インスタンスファミリーごとに異なります。例えば、R5 専有ホスト は、32 個の物理コアを使用する r5.8xlarge インスタンスを最大 2 個までサポートします。さらに、ホストのコア容量を満たすために、より小さいタイプの R5 インスタンスを追加して使用できます。各インスタンスファミリーでサポートされているインスタンスタイプの数については、「Amazon EC2 Dedicated Hosts の料金」を参照してください。

次の表に、インスタンスタイプの組み合わせの例を示します。

インスタンスファミリー	インスタンスタイプの組合せ例
R5	• 具体例 1: 4 x r5.4xlarge + 4 x r5.2xlarge • 具体例 2: 1 x r5.12xlarge + 1 x r5.4xlarge + 1 x r5.2xlarge + 5 x r5.xlarge + 2 x r5.large
C5	• 具体例 1: 1 x c5.9xlarge + 2 x c5.4xlarge + 1 x c5.xlarge • 具体例 2: 4 x c5.4xlarge + 1 x c5.xlarge + 2 x c5.large
M5	• 具体例 1: 4 x m5.4xlarge + 4 x m5.2xlarge • 具体例 2: 1 x m5.12xlarge + 1 x m5.4xlarge + 1 x m5.2xlarge + 5 x m5.xlarge + 2 x m5.large

考慮事項

複数のインスタンスタイプをサポートする Dedicated Hosts を使用する場合は、次の点に注意してください。

- 同一のホスト上で複数のインスタンスタイプを使用することは、同じインスタンスファミリー内でのみ可能です。

- インスタンスタイプを混在させる場合は、ホストの使用率を最大化するために、最初に大きなインスタンスタイプを起動してから小さなインスタンスタイプを起動することをお勧めします。
  - 専有ホスト上のインスタンスタイプの組み合わせと起動順序によっては、ホストの使用率を最大化することが物理的に不可能な場合があります。ホストでインスタンスタイプを混在させる場合、一部の容量がホストで使用可能であっても使用できない場合があります。例えば、r5n ホストで 16 個の vCPU が使用可能な場合、r5n.4xlarge が 16 個の vCPU で実行されていても、ホストで 4xlarge インスタンスを起動できないことがあります。

> **Note**
>
> A1 専有ホストを複数のインスタンスタイプに対して有効にすると、そのホストでは `a1.xlarge` インスタンスと `a1.2xlarge` インスタンスを組み合わせたもののみを起動できます。そのホストで `a1.medium` インスタンスまたは `a1.large` インスタンスを起動すると、そのホストで続けて起動できるのは、同じインスタンスタイプのみに制限されます。1 つの `a1.4xlarge` インスタンスが、そのホストのすべての容量を消費します。`a1.medium` インスタンスまたは `a1.large` インスタンスのいずれかに、ホストが必要な場合は、それらのインスタンスタイプに個別のホストを割り当てることが推奨されます。

Amazon EC2 専有ホスト上のバースト可能な T3 インスタンス

Dedicated Hosts は、バースト可能なパフォーマンス T3 インスタンスをサポート T3 インスタンスは、適格な BYOL ライセンスソフトウェアを専用ハードウェアで使用するための費用対効果の高い方法を提供します。T3 インスタンスの vCPU フットプリントが小さいため、ワークロードを少数のホストに統合し、コアごとのライセンスの使用率を最大化できます。

T3 Dedicated Hosts は、CPU 使用率が低 ~ 中程度の BYOL ソフトウェアを実行するのに最適です。Windows Server、Windows デスクトップ、SQL Server、SUSE Enterprise Linux Server、Red Hat Enterprise Linux および Oracle データベースなどの、ソケット単位、コア単位または VM 単位が含まれます。T3 Dedicated Hosts に適したワークロードの例としては、小中規模のデータベース、仮想デスクトップ、開発/テスト環境、コードリポジトリ、製品プロトタイプなどがあります。T3 Dedicated Hosts は、CPU 使用率が高いワークロードや、相関する CPU バーストが同時に発生するワークロードには推奨されません。

Dedicated Hosts 上の T3 インスタンスは、共有テナンシーハードウェア上の T3 インスタンスと同じクレジットモデルを使用します。ただし、それらは`standard`クレジットモードのみです。`unlimited`クレジットモードはサポートしていません。`standard` モードでは、専有ホスト上

の T3 インスタンスは、共有テナンシーハードウェア上のバースト可能なインスタンスと同じ方法でクレジットを獲得、消費、および蓄積します。バーストパフォーマンスインスタンスは、ベースラインレベルの CPU パフォーマンスを提供しながら、必要に応じてバーストする機能を備えています。ベースラインより上にバーストする場合、インスタンスは CPU クレジット残高に蓄積されたクレジットを消費します。発生したクレジットが枯渇すると、CPU 使用率はベースラインレベルまで低下します。`standard`モードの詳細については、スタンダードのバーストパフォーマンスインスタンスの仕組みを参照してください。

T3 Dedicated Hosts は、Amazon EC2 Dedicated Hosts が提供するすべての機能をサポートします。これには、1 つのホスト上の複数のインスタンスサイズ、ホストリソースグループ、および BYOL が含まれます。

サポートされる T3 インスタンスのサイズと構成

T3 Dedicated Hosts は、ベースライン CPU パフォーマンスと、必要に応じてより高いレベルまでバーストする機能を提供することにより、ホストの CPU リソースを共有する汎用バースト可能な T3 インスタンスを実行します。これにより、48 個のコアを持つ T3 Dedicated Host は、ホストあたり最大 192 個のインスタンスをサポートできます。ホストのリソースを効率的に利用し、最高のインスタンスパフォーマンスを提供するために、Amazon EC2 インスタンス配置アルゴリズムは、ホスト上で起動できるサポートされているインスタンス数とインスタンスサイズの組み合わせを自動的に計算します。

T3 Dedicated Host は、同じホストで複数のインスタンスタイプをサポートします。Dedicated Hosts では、すべての T3 インスタンスがサポートされています。ホストの CPU 制限まで、T3 インスタンスのさまざまな組み合わせを実行できます。

次の表は、サポートされているインスタンスタイプのリストと、各インスタンスタイプのパフォーマンスのサマリー、および起動可能な各サイズのインスタンスの最大数を示しています。

インスタンスタイプ	vCPU	メモリ (GiB)	vCPU あたりのベースライン CPU 使用率	ネットワークバースト帯域幅 (Gbps)	Amazon EBS バースト帯域幅 (Mbps)	専有ホストあたりの最大インスタンス数
t3.na	2	0.5	5%	5	最大 2,085	192
t3.m	2	1	10%	5	最大 2,085	192
t3.sn	2	2	20%	5	最大 2,085	192
t3.m	2	4	20%	5	最大 2,085	192
t3.la	2	8	30%	5	2,780	96
t3.xl	4	16	40%	5	2,780	48
t3.2x e	8	32	40%	5	2,780	24

T3 Dedicated Hosts の CPU 使用率の監視

`DedicatedHostCPUUtilization`Amazon CloudWatch メトリックスを使用して、専有ホストの vCPU 使用率を監視します。メトリクスは、`EC2`名前空間および`Per-Host-Metrics`ディメンションで利用可能です。詳細については「専用ホストメトリクス」を参照してください。

独自のソフトウェアライセンスを Amazon EC2 専有ホストで使用する

Dedicated Hosts を利用すると、ソフトウェアライセンスを、既存のソケット単位、コア単位、または VM 単位で使用できます。自分のライセンスを使用する場合、お客様は自分のライセンスを管理する責任があります。ただし、Amazon EC2 ではインスタンスアフィニティやターゲットを絞ったプレイスメントなど、ライセンスのコンプライアンスを維持するための機能を利用できます。

自分のボリュームライセンスマシンのイメージを Amazon EC2 で使用するための一般的な手順を以下に示します。

1. マシンイメージの使用を制御するライセンス条件が、仮想化クラウド環境での使用を許可していることを確認します。Microsoft ライセンスの詳細については、「Amazon Web Services と Microsoft ライセンス」を参照してください。
2. マシンイメージを Amazon EC2 内で使用できることを確認したら、VM Import/Export を使用してインポートします。マシンイメージをインポートする方法については、VM Import/Export ユーザーガイドを参照してください。
3. マシンイメージをインポートしたら、自分のアカウント内のアクティブな Dedicated Hosts で、そのイメージからインスタンスを起動できます。
4. オペレーティングシステムによっては、これらのインスタンスを実行する際、自分の KMS サーバー (Windows Server や Windows SQL Server など) で、これらのインスタンスをアクティブにすることが必要になる場合があります。インポートした Windows AMI を、Amazon Windows KMS サーバーに対して有効化することはできません。

> **Note**
>
> イメージが AWS 内でどのように使用されているかを追跡するには、AWS Config でホストの記録を有効にします。AWS Config では、Dedicated Hosts での設定の変更を記録したり、その出力をライセンスレポートのデータソースとして使用したりすることができます。詳細については「AWS Config を使用して、Amazon EC2 専有ホストでの設定の変更を追跡する」を参照してください。

Amazon EC2 専有ホストの自動配置とホストアフィニティ

Dedicated Hostsの配置制御は、インスタンスおよびホストの両レベルで行われます。

自動配置

自動配置はホストレベルで設定されます。自動配置を使用すると、起動するインスタンスについて、特定のホストで起動されるようにするか、設定が合致する任意のホストで起動されるようにするかを管理できます。

専有ホストの自動配置が無効になっている場合は、一意のホスト ID を指定するホストテナンシーインスタンス起動のみが受け入れられます。これは、新しい Dedicated Hosts に対する既定の設定です。

専有ホストの自動配置が有効になっている場合は、インスタンスタイプ設定が一致するすべてのターゲット未指定のホストテナンシーインスタンス起動が受け入れられます。

インスタンスの起動時に、テナンシーを設定する必要があります。特定の HostId を指定せずに Dedicated Host でインスタンスを起動すると、自動配置が有効で、インスタンスタイプが一致するすべての Dedicated Host でインスタンスを起動できます。

ホストのアフィニティ

ホストのアフィニティは、インスタンスレベルで設定します。また、インスタンスと Dedicated Host の間に関係を作成します。

アフィニティが Host に設定されている場合は、特定のホストで起動したインスタンスが停止しても、常に同じホストで再開されます。これは、ターゲットを絞った起動にもターゲットを絞らない起動にも適用されます。

アフィニティがDefault に設定されているときにインスタンスを停止して再起動する場合は、使用可能な任意のホスト上で再起動できます。ただし、最後に実行した Dedicated Host 上でベストエフォートベースでの再起動を試みます。

アカウントで使用するために Amazon EC2 専有ホストを割り当てる

専有ホストの使用を開始するには、まずアカウントに割り当てる必要があります。Dedicated Host を割り当てると、ご自身のアカウントにおいてDedicated Hostの容量がすぐに使用できるようになり、Dedicated Hostにおけるインスタンスの起動を開始できます。

アカウントに 専有ホスト を割り当てる場合、単一のインスタンスタイプ、または同じインスタンスファミリー内の複数のインスタンスタイプをサポートする構成を選択できます。ホスト上で実行できるインスタンスの数は、選択した構成によって異なります。詳細については、「Amazon EC2 専有ホストインスタンスのキャパシティ設定」を参照してください。

Console

Dedicated Host を割り当てるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Dedicated Hosts]、[Dedicated Host の割り当て] の順に選択してください。
3. [インスタンスファミリー] については、Dedicated Host向けのインスタンスファミリーを選びます。

4. Dedicated Host について、選択したインスタンスファミリー内にある複数のインスタンスをサポートするか、特定のインスタンスタイプのみサポートするかを指定します。次のいずれかを行ってください。

   - 選択したインスタンスファミリー内の複数のインスタンスタイプをサポートするようにDedicated Hostを設定するには、[複数のインスタンスタイプをサポートする]、[有効化] の順に選択してください。この選択を行うと、Dedicated Host において、同一インスタンスファミリー内の異なるインスタンスサイズを起動できるようになります。例えば、m5インスタンスファミリーとこのオプションを選択すると、Dedicated Hostにおいてm5.xlargeインスタンスとm5.4xlargeインスタンスを起動できます。
   - 選択したインスタンスファミリーの 1 つのインスタンスタイプをサポートするように Dedicated Host を設定するには、[Support multiple instance types (複数のインスタンスをサポートする)] をオフにして、[インスタンスタイプ] でサポートするインスタンスタイプを選択してください。これにより、Dedicated Host で 1 つのインスタンスタイプを起動できます。例えば、このオプションを選択し、m5.4xlargeをサポート対象インスタンスタイプとして指定すると、専有ホストにおいてはm5.4xlargeインスタンスに限り起動できます。

5. [アベイラビリティーゾーン] については、専有ホストを割り当てるアベイラビリティーゾーンを選択してください。
6. インスタンスタイプが一致する、ターゲットを絞らないインスタンスの起動を受け入れることを Dedicated Host に許可するには、[Instance auto-placement (インスタンスの自動プレイスメント)] で、[Enable (有効)] を選択してください。自動配置の詳細については、Amazon EC2 専有ホストの自動配置とホストアフィニティを参照してください。
7. Dedicated Host のホスト復旧を有効にするには、[Host recovery (ホスト復旧)]、[有効化] の順に選択してください。詳細については「Amazon EC2 専有ホストリカバリ」を参照してください。
8. [数量] については、割り当てるDedicated Hostsの数を入力してください。
9. (オプション) [新しいタグの追加] をクリックし、タグキーとタグ値を入力してください。
10. [Allocate] を選択してください。

AWS CLI

Dedicated Host を割り当てるには

allocate-hosts コマンドを使用します。次の例では、m5 アベイラビリティーゾーンで us-east-1a インスタンスファミリー内の複数のインスタンスタイプをサポートしている専有ホストを割り当てます。また、ホスト復旧を有効にし、自動配置を無効にします。

```
aws ec2 allocate-hosts \
    --instance-family "m5" \
    --availability-zone "us-east-1a" \
    --auto-placement "off" \
    --host-recovery "on" \
    --quantity 1
```

次の例では、指定されたアベイラビリティーゾーンでターゲット未指定のインスタンス起動をサポートする専有ホストを割り当て、ホスト復旧を有効にし、自動配置を有効にします。

```
aws ec2 allocate-hosts \
    --instance-type "m5.large" \
    --availability-zone "eu-west-1a" \
    --auto-placement "on" \
    --host-recovery "on" \
    --quantity 1
```

PowerShell

Dedicated Host を割り当てるには

New-EC2Host コマンドレットを使用します。次の例では、m5 アベイラビリティーゾーンで us-east-1a インスタンスファミリー内の複数のインスタンスタイプをサポートしている専有ホストを割り当てます。ホストでもホスト復旧が有効になっており、自動配置は無効になっています。

```
New-EC2Host `
    -InstanceFamily m5 `
    -AvailabilityZone us-east-1a `
    -AutoPlacement Off `
    -HostRecovery On `
    -Quantity 1
```

次の例では、指定されたアベイラビリティーゾーンでターゲット未指定のインスタンス起動をサポートする専有ホストを割り当て、ホスト復旧を有効にします。

```
New-EC2Host `
```

```
    -InstanceType m5.large `
    -AvailabilityZone eu-west-1a `
    -AutoPlacement On `
    -HostRecovery On `
    -Quantity 1
```

Amazon EC2 専有ホストで Amazon EC2 インスタンスを起動する

Dedicated Host を割り当てたら、そのホストにインスタンスを起動できます。起動するインスタンスタイプに使用できる十分な容量を持つアクティブなDedicated Hostsがない場合には、hostテナンシーでインスタンスを起動できません。

考慮事項

- Amazon EC2 によって提供されている SQL Server、SUSE、RHEL AMI を Dedicated Hosts で使用することはできません。
- 複数のインスタンスサイズをサポートする Dedicated Hosts については、大きいインスタンスサイズから始め、必要に応じて残りのインスタンスキャパシティを小さいインスタンスサイズで埋めることをお勧めします。
- インスタンスを起動する前に、制限事項を確認してください。詳細については「Dedicated Hosts の制約事項」を参照してください。

Console

Dedicated Hosts ページから特定の Dedicated Host でインスタンスを起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Dedicated Hosts] を選択してください。
3. [Dedicated Hosts] ページでホストを選択した後、[Actions] (アクション)、[Launch Instance(s) onto host] (インスタンスをホストで起動) の順に選択してください。
4. [Application and OS Images] (アプリケーションと OS イメージ) セクションにあるリストから、使用する AMI を選択してください。
5. [Instance type] (インスタンスタイプ) セクションで、起動するインスタンスタイプを選択してください。

> **Note**
>
> Dedicated Hostが単一のインスタンスタイプのみサポートしている場合、デフォルトでは、サポートされているインスタンスタイプが選択され、変更できません。Dedicated Host が複数のインスタンスタイプをサポートしている場合は、Dedicated Host の使用可能なインスタンスキャパシティーに基づいて、サポートされているインスタンスファミリー内のインスタンスタイプを選択する必要があります。大きいインスタンスサイズから始め、必要に応じて残りのインスタンス容量を小さいインスタンスサイズで埋めることをお勧めします。

6. [Key pair] (キーペア) セクションで、インスタンスに関連付けるキーペアを選択してください。
7. [高度な詳細] セクションにある [テナンシーのアフィニティ] で、次のいずれかを選択します:
   - [オフ] - ホストアフィニティが無効になります。インスタンスは指定されたホストで起動されますが、停止されると同じ専有ホストで再開されない場合があります。
   - 専有ホスト ID - ホストアフィニティが有効になっています。停止したインスタンスは、常にこの指定ホストで再開されます (キャパシティがある場合)。ホストに空き容量がない場合はインスタンスを再起動できません。異なるホストとのアフィニティを確立する必要があります。

   アフィニティの詳細については、Amazon EC2 専有ホストの自動配置とホストアフィニティを参照してください。

   > **Note**
   >
   > [Tenancy (テナンシー)] オプションと [Host (ホスト)] オプションは、選択したホストに基づき、事前に設定されています。

8. 必要に応じて、残りのインスタンスオプションを設定します。詳細については「Amazon EC2 インスタンス設定パラメータのリファレンス」を参照してください。
9. [インスタンスを起動] を選択してください。

インスタンス起動ウィザードを使用してインスタンスを Dedicated Host で起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

2. ナビゲーションペインで、[Instances] (インスタンス)、[Launch instance] (インスタンスを起動) の順に選択してください。
3. [Application and OS Images] (アプリケーションと OS イメージ) セクションにあるリストから、使用する AMI を選択します。
4. [Instance type] (インスタンスタイプ) セクションで、起動するインスタンスタイプを選択してください。
5. [Key pair] (キーペア) セクションで、インスタンスに関連付けるキーペアを選択してください。
6. [Advanced] (高度な情報) セクションで以下を実行します。

   a. [Tenancy] (テナンシー) で、[Dedicated Host] (専有ホスト) を選択してください。

   b. [Target host by] (ターゲットホスト) で、[Host ID] (ホスト ID) を選択してください。

   c. [Target host ID] (ターゲットホスト ID) で、インスタンスを起動するホストを選択してください。

   d. [テナンシーのアフィニティ] で、次のいずれかを選択します:

   - [オフ] - ホストアフィニティが無効になります。インスタンスは指定されたホストで起動されますが、停止されると同じ専有ホストで再開されない場合があります。
   - 専有ホスト ID - ホストアフィニティが有効になっています。停止したインスタンスは、常にこの指定ホストで再開されます (キャパシティがある場合)。ホストに空き容量がない場合はインスタンスを再起動できません。異なるホストとのアフィニティを確立する必要があります。

   アフィニティの詳細については、Amazon EC2 専有ホストの自動配置とホストアフィニティを参照してください。
7. 必要に応じて、残りのインスタンスオプションを設定します。詳細については「Amazon EC2 インスタンス設定パラメータのリファレンス」を参照してください。
8. [インスタンスを起動] を選択してください。

AWS CLI

Dedicated Host でインスタンスを起動するには

run-instances コマンドを使用し、`--placement` オプションでインスタンスのアフィニティ、テナンシー、ホストを指定します。

ホストアフィニティを使用して特定の専有ホストで起動するには (停止した場合、インスタンスは常に同じホストで再起動する):

```
--placement Affinity=host,Tenancy=host,HostId=h-07879acf49EXAMPLE
```

ホストアフィニティなしで特定の専有ホストで起動するには (インスタンスは使用可能な任意のホストで再起動できる):

```
--placement Tenancy=host,HostId=h-07879acf49EXAMPLE
```

自動配置が有効になっていてインスタンスタイプが一致する、利用可能な専有ホストで起動するには:

```
--placement Tenancy=host
```

PowerShell

Dedicated Host でインスタンスを起動するには

New-EC2Instance コマンドレットを使用し、-Placement パラメータでインスタンスのアフィニティ、テナンシー、およびホストを指定します。

ホストアフィニティを使用して特定の専有ホストで起動するには (停止した場合、インスタンスは常に同じホストで再起動する):

```
-Placement_Affinity host `
-Placement_Tenancy host `
-Placement_HostId h-07879acf49EXAMPLE
```

ホストアフィニティなしで特定の専有ホストで起動するには (インスタンスは使用可能な任意のホストで再起動できる):

```
-Placement_Tenancy host `
-Placement_HostId h-07879acf49EXAMPLE
```

自動配置が有効になっていてインスタンスタイプが一致する、利用可能な専有ホストで起動するには:

```
-Placement_Tenancy host
```

ホストリソースグループに Amazon EC2 インスタンスを起動する

Dedicated Hosts は AWS License Manager にも統合されています。License Manager では、ホストリソースグループを作成できます。ホストリソースグループは、1 つのエンティティとして管理される Dedicated Hosts コレクションです。ホストリソースグループを作成する場合は、Dedicated Hosts のホスト管理設定 (自動割り当てや自動リリースなど) を指定します。これにより、これらのホストを手動で割り当てて管理することなく、Dedicated Hosts にインスタンスを作成できます。詳細については、AWS License Manager ユーザーガイドの ホスト Resource Groupsを参照してください。

インスタンスの作成先のホストリソースグループ内のいずれかの Dedicated Host にインスタンス用の空き容量がある場合、Amazon EC2 はそのホストにインスタンスを作成します。ホストリソースグループ内のいずれのホストにもインスタンス用の空き容量がない場合、Amazon EC2 はホストリソースグループ内に新しいホストを自動的に割り当て、そのホストにインスタンスを作成します。詳細については、AWS License Manager ユーザーガイドの ホスト Resource Groupsを参照してください。

要件と制限

- コアベースまたはソケットベースのライセンス設定を AMI に関連付ける必要があります。
- Dedicated Hosts の Amazon EC2 で提供されている SQL Server、SUSE、または RHEL AMI を使用することはできません。
- ホスト ID を選択して特定のホストをターゲットにすることはできません。また、ホストリソースグループにインスタンスを作成するときに、インスタンスのアフィニティを有効にすることはできません。

Console

ホストリソースグループにインスタンスを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Instances] (インスタンス)、[Launch instance] (インスタンスを起動) の順に選択してください。
3. [Application and OS Images] (アプリケーションと OS イメージ) セクションにあるリストから、使用する AMI を選択します。
4. [Instance type] (インスタンスタイプ) セクションで、起動するインスタンスタイプを選択してください。

5. [Key pair] (キーペア) セクションで、インスタンスに関連付けるキーペアを選択してください。
6. [Advanced] (高度な情報) セクションで以下を実行します。
   a. [Tenancy] (テナンシー) で、[Dedicated Host] (専有ホスト) を選択してください。
   b. [Target host by] (ターゲットホスト) で、[Host resource group] (ホストリソースグループ) を選択してください。
   c. [Tenancy host resource group] (テナンシーのホストリソースグループ) で、インスタンスを起動するホストリソースグループを選択してください。
   d. [Tenancy affinity] (テナンシーのアフィニティ) で、以下のいずれかを実行します。
      - Off を選択– インスタンスは指定されたホストで起動されますが、停止された後に、以前と同一の Dedicated Host で再開される保証はありません。
      - Dedicated Host ID を選択 – インスタンスは停止後も、常に同じ特定のホストで再開されます。

      アフィニティの詳細については、Amazon EC2 専有ホストの自動配置とホストアフィニティを参照してください。
7. 必要に応じて、残りのインスタンスオプションを設定します。詳細については「Amazon EC2 インスタンス設定パラメータのリファレンス」を参照してください。
8. [インスタンスを起動] を選択してください。

AWS CLI

ホストリソースグループにインスタンスを作成するには

run-instances コマンドを使用します。`--placement` オプションで、テナンシーを省略し、ホストリソースグループの ARN を指定します。

```
--placement HostResourceGroupArn=arn:aws:resource-groups:us-
east-2:123456789012:group/my-resource-group
```

PowerShell

ホストリソースグループにインスタンスを作成するには

New-EC2Instance コマンドレットを使用します。-Placement パラメータで、テナンシーを省略し、ホストリソースグループの ARN を指定します。

```
-Placement_HostResourceGroupArn arn:aws:resource-groups:us-
east-2:123456789012:group/my-resource-group
```

既存の Amazon EC2 専有ホストの自動配置設定を変更する

専有ホストを AWS アカウントに割り当てると、その自動配置設定を変更できるようになります。

Console

Dedicated Host の自動配置を変更するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Dedicated Hosts] を選択してください。
3. ホストを選択し、[アクション]、[ホストの変更] の順に選択してください。
4. [インスタンスの自動配置] で、[有効化] を選択して自動配置を有効にするか、[有効化] をオフにして自動配置を無効にします。詳細については「Amazon EC2 専有ホストの自動配置とホストアフィニティ」を参照してください。
5. [Save] を選択してください。

AWS CLI

Dedicated Host の自動配置を変更するには

modify-hosts コマンドを使用します。

```
aws ec2 modify-hosts \
    --auto-placement on \
    --host-ids h-012a3456b7890cdef
```

PowerShell

Dedicated Host の自動配置を変更するには

Edit-EC2Host コマンドレットを使用します。

```
Edit-EC2Host `
    -AutoPlacement 1 `
    -HostId h-012a3456b7890cdef
```

既存の Amazon EC2 専有ホストでサポートされているインスタンスタイプを変更する

お客様は、Dedicated Hostを修正することで、このホストがサポートするインスタンスタイプを変更できます。単一のインスタンスタイプのみサポートしているホストの場合には、インスタンスファミリー内にある複数のインスタンスタイプをサポートするように修正できます。同様に、複数のインスタンスタイプをサポートしているホストの場合には、単一のインスタンスタイプのみをサポートするように修正できます。

複数のインスタンスタイプをサポートするようにDedicated Hostを修正するには、初めに、該当ホスト上で実行中のインスタンスをすべて停止する必要があります。修正は約 10 分で完了します。修正の実行中には、Dedicated Hostがpending状態に移行します。pending状態にあるDedicated Hostにおいて停止中のインスタンスを開始したり、新たなインスタンスを起動したりすることはできません。

複数のインスタンスタイプをサポートしているDedicated Hostを 1 つのインスタンスタイプのみをサポートするように変更するには、ホストで実行中のインスタンスがないか、実行中のインスタンスがホストでサポートするインスタンスタイプである必要があります。具体例を挙げると、m5インスタンスファミリー内にある複数のインスタンスタイプをサポートしているホストを、m5.largeインスタンスのみをサポートするように修正するには、Dedicated Hostが、いかなるインスタンスも実行していない状態であるか、m5.large実行中のインスタンスのみの状態でなければなりません。

仮想インスタンスタイプにホストを割り当てる場合、ホストの割り当て後にインスタンスタイプを .metal インスタンスタイプに変更することはできません。例えば、m5.large インスタンスタイプにホストを割り当てた場合、インスタンスタイプを m5.metal に変更することはできません。同様に、.metal インスタンスタイプにホストを割り当てる場合、ホストの割り当て後にインスタンスタイプを仮想インスタンスタイプに変更することはできません。例えば、m5.metal インスタンスタイプにホストを割り当てた場合、インスタンスタイプを m5.large に変更することはできません。

Console

Dedicated Host のサポートされているインスタンスタイプを変更するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Dedicated Hosts] (専有ホスト) を選択してください。

3. 変更する Dedicated Host を選択し、[アクション]、[ホストの変更] の順に選択してください。
4. Dedicated Host の現在の設定に応じて、次のいずれかの操作を実行します。

   - Dedicated Host が特定のインスタンスタイプを現在サポートしている場合は、[複数のインスタンスタイプをサポートする] は有効にならず、現在サポートされているインスタンスタイプが [インスタンスタイプ] に表示されます。現在のインスタンスファミリー内にある複数のタイプをサポートするようにホストを変更するには、[複数のインスタンスタイプをサポートする] の [有効化] を選択してください。

     複数のインスタンスタイプをサポートするようにホストを修正するには、初めに、該当ホスト上で実行されているすべてのインスタンスを停止する必要があります。
   - Dedicated Host がインスタンスファミリー内の複数のインスタンスタイプを現在サポートしている場合は、[複数のインスタンスタイプをサポートする] の [有効] が選択されています。特定のインスタンスタイプをサポートするようにホストを変更するには、[複数のインスタンスタイプをサポートする] で、[有効化] をオフにし、[インスタンスタイプ] で、サポートする特定のインスタンスタイプを選択してください。

     Dedicated Host がサポートするインスタンスファミリーを変更することはできません。
5. [Save] を選択してください。

AWS CLI

Dedicated Host のサポートされているインスタンスタイプを変更するには

modify-hosts コマンドを使用します。

以下の例では、m5 インスタンスファミリー内にある複数のインスタンスタイプをサポートするように、専有ホストを修正します。

```
aws ec2 modify-hosts \
    --instance-family m5 \
    --host-ids h-012a3456b7890cdef
```

以下の例では、m5.xlarge インスタンスのみをサポートするように専有ホストを修正します。

```
aws ec2 modify-hosts \
    --instance-type m5.xlarge \
```

```
    --instance-family --host-ids h-012a3456b7890cdef
```

PowerShell

Dedicated Host のサポートされているインスタンスタイプを変更するには

Edit-EC2Host コマンドレットを使用します。

以下の例では、m5 インスタンスファミリー内にある複数のインスタンスタイプをサポートするように、専有ホストを修正します。

```
Edit-EC2Host `
    -InstanceFamily m5 `
    -HostId h-012a3456b7890cdef
```

以下の例では、m5.xlarge インスタンスのみをサポートするように専有ホストを修正します。

```
Edit-EC2Host `
    -InstanceType m5.xlarge `
    -HostId h-012a3456b7890cdef
```

Amazon EC2 インスタンスの Amazon EC2 専有ホストのテナンシーとアフィニティを変更する

インスタンスのテナンシーは、インスタンスの起動後に変更できます。インスタンスのアフィニティを変更して、特定のホストをターゲットにしたり、アカウント内の属性が一致する使用可能な専有ホストで起動できるようにしたりすることもできます。インスタンスのテナンシーまたはアフィニティを修正するには、そのインスタンスをstopped状態にする必要があります。

インスタンスのオペレーティングシステムの詳細、および SQL Server がインストールされているかどうかによって、サポートされる変換が影響されます。インスタンスで使用できるテナンシー変換パスの詳細については、「License Manager ユーザーガイド」の「Tenancy conversion」を参照してください。

> **Note**
>
> T3 インスタンスの場合、host のテナンシーを使用するには専有ホストでインスタンスを起動する必要があります。T3 インスタンスの場合、テナンシーを host から dedicated また

は `default` に変更することはできません。これらのサポートされていないテナンシー変更のいずれかを試みると、エラーコード `InvalidRequest` が発生します。

Console

インスタンスのテナンシーまたはアフィニティを変更するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. [Instances (インスタンス)] を選択し、変更するインスタンスを選択してください。
3. [Instance state (インスタンスの状態)]、[Stop (停止)] の順に選択してください。
4. 選択したインスタンスについて、[アクション]、[インスタンス設定]、[インスタンスの配置の変更] を選択してください。
5. [インスタンスの配置の変更] ページで、次の設定を行います。

   - [Tenancy] — 次のいずれかを選択してください。
     - [専用ハードウェアインスタンスの実行] — インスタンスを ハードウェア専有インスタンス として起動します。詳細については「Amazon EC2 ハードウェア専有インスタンス」を参照してください。
     - [Launch the instance on a Dedicated Host] — 設定可能なアフィニティを使用してインスタンスを Dedicated Host で起動します。
   - [Affinity] — 次のいずれかを選択してください。
     - [This instance can run on any one of my hosts] — インスタンスは、そのインスタンスタイプをサポートするアカウントの利用可能な Dedicated Host で起動されます。
     - [This instance can only run on the selected host] — インスタンスは、[Target Host] (ターゲットホスト) として選択された Dedicated Host でのみ実行できます。
   - [Target Host] (ターゲットホスト) – インスタンスを実行させるための、Dedicated Host を選択してください。ターゲットホストが表示されない場合は、アカウントに利用可能な、互換性のある Dedicated Hosts がない可能性があります。

   詳細については「Amazon EC2 専有ホストの自動配置とホストアフィニティ」を参照してください。
6. [Save] を選択してください。

AWS CLI

インスタンスのテナンシーまたはアフィニティを変更するには

modify-instance-placement コマンドを使用します。次の例では、指定したインスタンスのアフィニティを default から host に変更し、インスタンスがアフィニティを持つ対象の Dedicated Host を指定します。

```
aws ec2 modify-instance-placement \
    --instance-id i-1234567890abcdef0 \
    --affinity host \
    --tenancy host \
    --host-id h-012a3456b7890cdef
```

PowerShell

インスタンスのテナンシーまたはアフィニティを変更するには

Edit-EC2InstancePlacement コマンドレットを使用します。次の例では、指定したインスタンスのアフィニティを default から host に変更し、インスタンスがアフィニティを持つ対象の Dedicated Host を指定します。

```
Edit-EC2InstancePlacement `
    -InstanceId i-1234567890abcdef0 `
    -Affinity host `
    -Tenancy host `
    -HostId h-012a3456b7890cdef
```

Amazon EC2 専有ホストを解放する

ある専有ホストが不要になった場合は、そのホストで実行されているインスタンスを停止し、別のホストで起動するように指定してから、ホストをリリースできます。

ホストをリリースする前に、専有ホストで実行中のインスタンスを停止する必要があります。これらのインスタンスはアカウントの他の Dedicated Hosts に移行し、引き続き使用することができます。これらのステップは、オンデマンド Dedicated Hosts にのみ適用されます。

Console

専有ホストをリリースするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Dedicated Hosts] を選択してください。
3. [Dedicated Hosts] ページでリリースする専有ホストを選択してください。
4. [アクション]、[ホストのリリース] の順に選択してください。
5. [リリース] を選択して確定します。

AWS CLI

専有ホストをリリースするには

release-hosts マンドを使用します。

```
aws ec2 release-hosts --host-ids h-012a3456b7890cdef
```

PowerShell

専有ホストをリリースするには

Remove-EC2Host コマンドレットを使用します。

```
Remove-EC2Host -HostId h-012a3456b7890cdef
```

専有ホストをリリースすると、同じホストまたはホスト ID を再使用できなくなり、該当ホストのオンデマンド料金請求が停止します。Dedicated Host の状態は `released` に変わり、このホストではインスタンスを起動できなくなります。

> **Note**
>
> 最近 Dedicated Hosts をリリースした場合、制限に加算されなくなるまでに少し時間がかかることがあります。それまでは、新しい Dedicated Hosts を割り当てようとすると `LimitExceeded` エラーが発生する場合があります。このエラーが発生した場合は、数分後に新しいホストを再び割り当ててみてください。

停止したインスタンスはまだ使用可能であり、[Instances] ページに表示されます。その [host] テナンシー設定も維持されています。

Nitro ベースの Amazon EC2 Dedicated Hosts に移行する

Nitro System は、AWS が構築した、高パフォーマンス、高可用性、高度なセキュリティを実現するハードウェアとソフトウェアコンポーネントのコレクションです。ニトロ ベースの 専有ホスト は、Xen ベースの 専有ホスト と比較して、価格パフォーマンスが向上します。アカウントに Xen ベースの 専有ホスト がある場合は、ワークロードを ニトロ ベースの 専有ホスト に移行することをお勧めします。詳細については、AWS Nitro System をご参照ください。

Xen ベースの 専有ホスト から ニトロ ベースの 専有ホスト に移行するには、専有ホスト の Xen ベースのインスタンスを ニトロ ベースのインスタンスタイプに移行し、新しい ニトロ ベースの 専有ホスト を割り当ててから、移行した ニトロ ベースのインスタンスを新しい ニトロ ベースの 専有ホスト に移行する必要があります。

このトピックでは、Xen ベースの 専有ホスト から ニトロ ベースの 専有ホスト に移行するための詳細な手順について説明します。

移行手順

- ステップ 1: Xen ベースの専有ホストを特定する
- ステップ 2: Xen ベースのインスタンスを ニトロ ベースのインスタンスタイプに移行する
- ステップ 3: ニトロ ベースの専有ホストを割り当てる
- ステップ 4: 移行したインスタンスを新しい ニトロ ベースの 専有ホスト に移動する
- ステップ 5: 未使用の Xen ベースの 専有ホスト をリリースする

ステップ 1: Xen ベースの専有ホストを特定する

次の 専有ホスト は Xen ベースであり、ニトロ ベースの 専有ホスト に移行できます。

- 凡用: M3 | M4
- コンピューティング最適化| C3 | C4
- メモリ最適化: R3 | R4 | X1 | X1e
- ストレージ最適化: D2 | H1 | I2 | I3
- 高速コンピューティング: G3 | P3

アカウントに Xen ベースの 専有ホスト があるかどうかを確認するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Dedicated Hosts] を選択してください。
3. 検索フィールドで、インスタンスファミリーフィルターを使用して上記の Xen ベースの 専有ホスト を検索します。例えば、インスタンスファミリー = m3 です。

ステップ 2: Xen ベースのインスタンスを ニトロ ベースのインスタンスタイプに移行する

Xen ベースの 専有ホスト で実行されるインスタンスも Xen ベースです。これらのインスタンスを ニトロ ベースの専有ホストに移動する前に、これらのインスタンスを ニトロ ベースのインスタンスタイプに移行する必要があります。

> **Important**
>
> インスタンスの移行を開始する前に、データをバックアップすることをお勧めします。詳細については、「Amazon EC2 インスタンスからマルチボリューム Amazon EBS スナップショットを作成する」を参照してください。

Xen ベースの 専有ホスト で実行されているインスタンスを検索するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Dedicated Hosts] を選択してください。
3. 移行する Xen ベースのホストを選択し、実行中のインスタンスタブを選択してください。タブには、選択したホストで実行されているすべてのインスタンスが一覧表示されます。

Linux インスタンスを移行するには、Amazon EC2 インスタンスタイプの変更.を参照してください。

Windows インスタンスを移行するには、EC2 Windows インスタンスを Nitro ベースのインスタンスタイプに移行するを参照してください。

> **Note**
>
> インスタンスは、移行先の ニトロ ベースの 専有ホスト と一致するインスタンスタイプに移行してください。例えば、M7i 専有ホストに移行する場合は、インスタンスを M7i インスタンスタイプに移行してください。

ステップ 3: ニトロ ベースの専有ホストを割り当てる

サポートされている ニトロ ベースの 専有ホスト を検索するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンスタイプ] を選択してください。
3. 以下のフィルタを使用できます。
   - ハイパーバイザー = ニトロ
   - 専有ホスト サポート = 正

適切な ニトロ ベースのインスタンスタイプを見つけたら、新しい 専有ホスト を割り当てます。

ステップ 4: 移行したインスタンスを新しい ニトロ ベースの 専有ホスト に移動する

ニトロ ベースの 専有ホスト を割り当てて `available`状態になったら、以前に ニトロ ベースのインスタンスタイプに移行したインスタンスを新しい 専有ホスト に移動できます。

インスタンスを新しい ニトロ ベースの 専有ホスト に移動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションパネルのインスタンス。
3. 移行したインスタンスを選択し、「アクション」、、「インスタンス設定」、、「インスタンス配置の変更」 の順に選択します。
4. ターゲット専有ホストで、新しい Nitro ベースの専有ホストを選択し、保存を選択してください。
5. インスタンスを再起動します。インスタンスを選択し、[Instance state (インスタンスの状態)]、[Start instance (インスタンス始める)] の順に選択してください。

ステップ 5: 未使用の Xen ベースの 専有ホスト をリリースする

ワークロードを Xen ベースの 専有ホスト から新しい ニトロ ベースの 専有ホスト に移行したら、不要になった Xen ベースの 専有ホスト をリリースできます。

専有ホストの予約を購入して、専有ホストの請求で割引を受ける

専有ホストの予約は、オンデマンド専有ホストの料金と比較して最大 70% の割引を提供します。専有ホストの予約を購入するには、アカウントでアクティブな専有ホストが割り当てられている必要があります。詳細については「Dedicated Host Reservations」を参照してください。

Console

予約を購入するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. [Dedicated Hosts]、[Dedicated Host の予約]、[Dedicated Host の予約 の購入] の順に選択してください。
3. [サービスを検索] 画面で、次の操作を行います。
   a. [インスタンスファミリー] で、専有ホスト予約を購入しようとしている専有ホストの、インスタンスファミリーを選択してください。
   b. [支払いオプション] で、希望の支払いオプションを選択し、設定します。
4. [Next] を選択してください。
5. 専有ホスト予約に関連付ける専有ホストを選択し、[次へ] をクリックしてください。
6. (オプション) 専有ホスト予約にタグを割り当てます。
7. 注文を確認し、[購入] をクリックしてください。

AWS CLI

予約を購入するには

1. describe-host-reservation-offerings コマンドを使用して、ニーズに合った利用可能なオファリングを一覧表示します。次の例では、m4 インスタンスファミリー内のインスタンスをサポートし、契約期間が 1 年のオファリングを一覧表示します。

   期間は秒単位で指定されます。1 年契約は 31,536,000 秒で、3 年契約は 94,608,000 秒です。

```
aws ec2 describe-host-reservation-offerings \
    --filter Name=instance-family,Values=m4 \
    --max-duration 31536000
```

コマンドは、条件に合ったオファリングのリストを返します。購入するオファリングの ID を書き留めます。

2. purchase-host-reservationコマンドを使用してオファリングを購入し、前のステップで書き留めた `offeringId` を指定します。次の例では、指定された予約を購入して、AWS アカウントに割り当て済みの特定の Dedicated Host に関連付けます。さらに、キーが `purpose` で値が `production` のタグを、購入した予約に対し適用します。

```
aws ec2 purchase-host-reservation \
    --offering-id hro-03f707bf363b6b324 \
    --host-id-set h-013abcd2a00cbd123 \
    --tag-specifications 'ResourceType=host-
reservation,Tags={Key=purpose,Value=production}'
```

PowerShell

予約を購入するには

1. Get-EC2HostReservationOffering コマンドレットを使用して、ニーズに合った利用可能なオファリングを一覧表示します。以下の例では、`m5` インスタンスファミリーでインスタンスをサポートし、1 年契約を持っているオファーをリストします。

   期間は秒単位で指定されます。1 年契約は 31,536,000 秒で、3 年契約は 94,608,000 秒です。

```
$filter = @{Name="instance-family"; Values="m5"}
Get-EC2HostReservationOffering `
    -Filter $filter `
    -MaxDuration 31536000
```

コマンドは、条件に合ったオファリングのリストを返します。購入するオファリングの ID を書き留めます。

2. New-EC2HostReservation コマンドレットを使用してオファリングを購入し、前のステップで書き留めた ID を指定します。次の例では、指定した予約を購入し、それを AWS アカウントに割り当て済みの特定の Dedicated Host と関連付けます。

```
New-EC2HostReservation `
    -OfferingId hro-03f707bf363b6b324 `
    -HostIdSet h-013abcd2a00cbd123
```

クロスアカウント Amazon EC2 専有ホストの共有

Dedicated Host の共有を使用すると、Dedicated Host の所有者は Dedicated Host を他の AWS アカウントと共有したり、AWS 組織内で共有したりできます。これにより、Dedicated Hosts の作成と管理を一元的に行い、複数の AWS アカウント間や AWS 組織内で共有することが可能になります。

このモデルでは、Dedicated Host を所有する AWS アカウント (所有者) が、Dedicated Host を他の AWS アカウント (コンシューマー) と共有します。コンシューマーは、各自のアカウントに割り当てた Dedicated Hosts にインスタンスを作成する場合と同じように、共有している Dedicated Hosts にインスタンスを作成できます。所有者は、Dedicated Host およびそこに作成したインスタンスの管理に責任を負います。所有者は、コンシューマーが共有 Dedicated Hosts に作成したインスタンスを変更することはできません。コンシューマーは、自己が共有している Dedicated Hosts に作成したインスタンスの管理に責任を負います。コンシューマーは、他のコンシューマーまたは Dedicated Host 所有者が所有するインスタンスを表示または変更することはできません。また、自己が共有している Dedicated Hosts を変更することはできません。

Dedicated Host 所有者が Dedicated Host を共有できる相手は次のとおりです。

- AWS の組織内または組織外の特定の AWS アカウント
- AWS 組織内の組織単位
- AWS 組織全体

内容

- Dedicated Hosts を共有するための前提条件
- Dedicated Host の共有に関する制限事項
- 関連サービス
- アベイラビリティーゾーン間での共有
- 共有 Dedicated Host のアクセス許可

- 請求と使用量測定
- Dedicated Host の制限
- ホストの復旧と Dedicated Host の共有
- 複数の AWS アカウントで Amazon EC2 専有ホストを共有する
- 他の AWS アカウントと共有されている専有ホストの共有を解除する
- AWS アカウント内の共有されている Amazon EC2 専有ホストを表示する

Dedicated Hosts を共有するための前提条件

- Dedicated Host を共有するには、それを自分の AWS アカウント内で所有している必要があります。既に共有している Dedicated Host を共有することはできません。
- AWS 組織や AWS 組織内の組織単位との間で Dedicated Host を共有するには、AWS Organizations で共有を有効にする必要があります。詳細については、AWS Organizations ユーザーガイドのAWS RAM で共有を有効化するを参照してください。

Dedicated Host の共有に関する制限事項

`u-6tb1.metal`、`u-9tb1.metal`、`u-12tb1.metal`、`u-18tb1.metal`、および `u-24tb1.metal` のインスタンスタイプに割り当てられた Dedicated Hosts は共有できません

関連サービス

AWS Resource Access Manager

Dedicated Host の共有は AWS Resource Access Manager AWS RAMと統合されています。AWS RAM は、任意の AWS アカウントに対して、あるいは AWS 経由で AWS Organizations リソースを共有するためのサービスです。AWS RAM を使用すると、リソース共有を作成することで、自身が所有するリソースを共有できます。リソース共有では、共有対象のリソースと、共有先となるコンシューマーを指定します。コンシューマーには、個人の AWS アカウントや、AWS Organizations 内の組織単位または組織全体を指定できます。

AWS RAM の詳細については、AWS RAM ユーザーガイドを参照してください。

アベイラビリティーゾーン間での共有

リソースがリージョンの複数のアベイラビリティーゾーンに分散されるようにするために、アベイラビリティーゾーンは各 アカウントの名前に個別にマッピングされます。このため、アカウントが異なると、アベイラビリティーゾーンの命名方法が異なる場合があります。例えば、`us-east-1a` ア

カウントのアベイラビリティーゾーン AWS の場所は、別の us-east-1a アカウントのアベイラビリティーゾーン AWS の場所と異なる可能性があります。

自己のアカウントを基準にして Dedicated Hosts の場所を特定するには、アベイラビリティーゾーン ID (AZ ID) を使用する必要があります。アベイラビリティーゾーン ID は、すべての AWS アカウントにわたって各アベイラビリティーゾーンを一意に識別する ID です。例えば、use1-az1 は us-east-1 リージョンのアベイラビリティーゾーン ID であり、すべての AWS アカウントで同じ場所を示します。

アカウントのアベイラビリティーゾーンのアベイラビリティーゾーン ID を表示するには

1. https://console.aws.amazon.com/ram/home で AWS RAM コンソールを開きます。
2. 現在のリージョンのアベイラビリティーゾーン ID は、画面の右側にある [お客様の AZ ID] パネルに表示されます。

共有 Dedicated Host のアクセス許可

所有者のアクセス許可

所有者は、共有 Dedicated Hosts およびそこに作成したインスタンスの管理に責任を負います。所有者は、コンシューマーによって作成されたインスタンスも含めて、共有 Dedicated Host で実行されているすべてのインスタンスを表示できます。ただし、所有者は、コンシューマーによって作成された実行中のインスタンスに対してアクションを実行することはできません。

コンシューマーのアクセス許可

コンシューマーは、共有 Dedicated Host に作成したインスタンスの管理に責任を負います。コンシューマーは、共有 Dedicated Host を一切変更できません。また、他のコンシューマーや Dedicated Host 所有者が作成したインスタンスを表示または変更することもできません。

請求と使用量測定

Dedicated Hosts の共有に追加料金はかかりません。

所有者は、自己が共有する Dedicated Hosts に対して課金されます。コンシューマーは、共有 Dedicated Hosts に作成したインスタンスに対して課金されません。

Dedicated Host の予約 は、共有 Dedicated Hosts に対して引き続き請求割引を提供します。Dedicated Host 所有者のみが、自己が所有する共有 Dedicated Hosts 用の Dedicated Host の予約 を購入できます。

Dedicated Host の制限

共有 Dedicated Hosts は、所有者の Dedicated Hosts 制限に対してのみカウントされます。共有 Dedicated Hosts は、コンシューマーの Dedicated Hosts 制限に対してはカウントされません。同様に、コンシューマーが共有 Dedicated Hosts に作成するインスタンスは、コンシューマーのインスタンス制限に対してカウントされません。

ホストの復旧と Dedicated Host の共有

ホストの復旧は、Dedicated Host の 所有者とその共有相手のコンシューマーによって作成されたインスタンスを復旧します。代替 Dedicated Host は所有者のアカウントに割り当てられます。元の Dedicated Host と同じリソース共有に追加され、同じコンシューマと共有されます。

詳細については「Amazon EC2 専有ホストリカバリ」を参照してください。

複数の AWS アカウントで Amazon EC2 専有ホストを共有する

所有者が Dedicated Host を共有すると、コンシューマーはそのホストにインスタンスを作成できます。コンシューマーは、共有ホストに空き容量がある限り、そこに必要なだけのインスタンスを作成できます。

> **Important**
>
> Dedicated Hosts で BYOL ライセンスを共有するための適切なライセンス権限があることを確認する責任があります。

自動配置を有効にして Dedicated Host を共有する場合は、意図しない形で Dedicated Host が使用されないよう、次の点に注意してください。

- コンシューマーが Dedicated Host テナンシーを使用してインスタンスを作成する場合、自己のアカウントで所有している Dedicated Host に空き容量がないと、インスタンスは自動的に共有 Dedicated Host に作成されます。

Dedicated Host を共有するには、それをリソース共有に追加する必要があります。リソース共有とは、自身のリソースを AWS RAM アカウント間で共有するための AWS リソースです。リソース共有では、共有対象のリソースと、共有先のコンシューマーを指定します。Dedicated Host は、既存のリソースに追加することも、新しいリソース共有に追加することもできます。

自分が AWS Organizations 内の組織のメンバーであり、所属する組織で共有が有効化されている場合には、自分の組織内のコンシューマーに対し、共有された Dedicated Host に対するアクセス許可を自動的に付与することができます。それ以外の場合、コンシューマーはリソース共有への参加の招待を受け取り、その招待を受け入れた後で、共有 Dedicated Host へのアクセス許可が付与されます。

> **Note**
>
> Dedicated Host を共有した場合、コンシューマーがそれにアクセスできるまでに数分かかることがあります。

Console

Amazon EC2 コンソールを使用して自己所有の Dedicated Host を共有するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Dedicated Hosts] を選択してください。
3. 共有する Dedicated Host を選択し、[アクション]、[ホストの共有] の順にクリックしてください。
4. Dedicated Host の追加先のリソース共有を選択し、[ホストの共有] をクリックしてください。

   コンシューマーから共有ホストにアクセスできるまでに、数分かかることがあります。

AWS RAM コンソールを使用して、自分が所有する Dedicated Hosts を共有するには

「AWS RAM ユーザーガイド」の「リソース共有の作成」を参照してください。

AWS CLI

自分が所有する専有ホストを共有するには

create-resource-share コマンドを使用します。

```
aws ram create-resource-share \
    --name my-resource-share \
    --resource-arns arn:aws:ec2:us-east-2:123456789012:dedicated-
host/h-07879acf49EXAMPLE
```

PowerShell

自分が所有する専有ホストを共有するには

New-RAMResourceShare コマンドレットを使用します。

```
New-RAMResourceShare `
    -Name my-resource-share `
    -ResourceArn arn:aws:ec2:us-east-2:123456789012:dedicated-
host/h-07879acf49EXAMPLE
```

他の AWS アカウントと共有されている専有ホストの共有を解除する

Dedicated Host 所有者は、共有 Dedicated Host をいつでも共有解除できます。共有 Dedicated Host を共有解除する場合、以下のルールが適用されます。

- Dedicated Host を共有しているコンシューマーは、そこに新しいインスタンスを作成できなくなります。
- 共有解除時に Dedicated Host で実行されていたコンシューマー所有のインスタンスは、引き続き実行されますが、リタイアが予定されます。コンシューマーは、インスタンスのリタイア通知を受け取り、2 週間以内に通知に対処します。ただし、リタイア通知期間内に Dedicated Host がコンシューマーに再共有されると、インスタンスのリタイアはキャンセルされます。

自己所有の共有 Dedicated Host を共有解除するには、それをリソース共有から削除する必要があります。

Console

自分が所有する共有されている専有ホストの共有を解除するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Dedicated Hosts] を選択してください。
3. 共有解除する Dedicated Host を選択し、[共有] タブをクリックしてください。
4. [共有] タブに、Dedicated Host の追加先のリソース共有が一覧表示されます。Dedicated Host を削除する対象のリソース共有を選択し、[リソース共有からホストを削除] をクリックしてください。

AWS RAM コンソールを使用して、自分が所有する共有済みの Dedicated Hosts で共有を解除するには

「AWS RAM ユーザーガイド」の「Update a resource share」を参照してください。

AWS CLI

自分が所有する共有されている専有ホストの共有を解除するには

disassociate-resource-share コマンドを使用します。

```
aws ram disassociate-resource-share \
    --resource-share-arn arn:aws:ram:us-east-2:123456789012:resource-share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE \
 --resource-arns arn:aws:ec2:us-east-2:123456789012:dedicated-host/h-07879acf49EXAMPLE
```

PowerShell

自分が所有する共有されている専有ホストの共有を解除するには

Disconnect-RAMResourceShare コマンドレットを使用します。

```
Disconnect-RAMResourceShare `
    -ResourceShareArn "arn:aws:ram:us-east-2:123456789012:resource-share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE" `
    -ResourceArn "arn:aws:ec2:us-east-2:123456789012:dedicated-host/h-07879acf49EXAMPLE"
```

AWS アカウント内の共有されている Amazon EC2 専有ホストを表示する

他のアカウントと共有している専有ホストと、自分と共有されている専有ホストを表示できます。専有ホストの所有者は、コンシューマーによって作成されたインスタンスも含めて、ホストで実行されているすべてのインスタンスを表示できます。専有ホストが共有されている場合、共有されているホスト上に起動したインスタンスのみが表示され、他のコンシューマーによって起動されたインスタンスは表示されません。

所有者とコンシューマーは、次のいずれかの方法を使用して共有 Dedicated Hosts を特定できます。

Console

共有されている専有ホストを特定するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Dedicated Hosts] を選択してください。この画面には、自己が所有している Dedicated Hosts と共有している Dedicated Hosts が一覧表示されます。
3. [所有者] 列には、Dedicated Host の所有者の AWS アカウント ID が表示されます。
4. ホスト上で実行されているインスタンスを表示するには、[インスタンス] タブを選択してください。

AWS CLI

共有されている専有ホストを特定するには

describe-hosts コマンドを使用します。このコマンドは、自己が所有している Dedicated Hosts と共有している Dedicated Hosts を返します。`Owner` の値は、専有ホストの所有者のアカウント ID です。この `Instances` リストでは、ホストで実行されているインスタンスを示しています。

```
aws ec2 describe-hosts --filter "Name=state,Values=available"
```

PowerShell

共有されている専有ホストを特定するには

Get-EC2host コマンドレットを使用します。このコマンドレットは、ユーザーが所有する専有ホストと、ユーザーと共有されている専有ホストを返します。レスポンスの `Owner` の値は、専有ホストの所有者のアカウント ID です。この `Instances` リストでは、ホストで実行されているインスタンスを示しています。

```
Get-EC2Host -Filter @{Name="state"; Values="available"}
```

AWS Outposts の Amazon EC2 専有ホスト

AWS Outposts は、AWS のインフラストラクチャ、サービス、API、ツールをユーザーのオンプレミスまで拡張するフル管理されたサービスです。AWS Outposts は、AWS 管理インフラストラクチャへのローカルアクセスを提供することにより、AWS リージョンと同じプログラミングインターフェイスを使用してオンプレミスでアプリケーションを構築および実行できると同時に、ローカルコン

ピューティングおよびストレージリソースを使用して、レイテンシとローカルデータ処理のニーズを低減します。

Outpost とはお客様のサイトにデプロイされる AWS のコンピューティングおよびストレージキャパシティーのプールです。AWS はAWS リージョンの一部としてこのキャパシティーを運営、監視、管理します。

アカウントで所有している Outposts に Dedicated Hosts を割り当てることができます。これにより、専用の物理サーバーを必要とする既存のソフトウェアライセンスとワークロードを AWS Outposts に簡単に持ち込むことができます。Outpost の特定のハードウェアアセットをターゲットにして、ワークロード間のレイテンシーを最小限に抑えることもできます。

Dedicated Hosts を使用すると、Amazon EC2 で適格なソフトウェアライセンスを使用できるため、独自のライセンスを使用する場合の柔軟性と費用対効果が得られます。仮想マシン、ソケット、または物理コアにバインドされている他のソフトウェアライセンスも、ライセンス条項に従って、Dedicated Hosts で使用できます。Outpost は常に BYOL ワークロードに適格なシングルテナント環境でしたが、Dedicated Hosts を使用すると、Outpost の展開全体ではなく、必要なライセンスを単一のホストにデプロイできます。

さらに、Outpostで Dedicated Hosts を使用すると、インスタンスタイプのデプロイの柔軟性が高まり、インスタンスの配置をより細かく制御できます。インスタンスの起動に特定のホストをターゲットにして、ホストアフィニティを使用して、インスタンスが常にそのホストで実行されるようにするか、自動配置を使用して、設定と使用可能な容量が一致する使用可能なホストにインスタンスを起動できます。

目次

- 前提条件
- サポートされている機能
- 考慮事項
- AWS Outposts で Amazon EC2 専有ホストを割り当てる

前提条件

Outpost は、自分のサイトにインストールする必要があります。詳細については、AWS Outposts ユーザーガイドのOutpost を作成し、Outpost 容量を注文するを参照してください。

サポートされている機能

- 次のインスタンスファミリーがサポートされています。

  - 汎用: M5 | M5d | M7i | M8i
  - コンピューティング最適化: C5 | C5d | C7i | C8i
  - メモリ最適化: R5 | R5d | R7i | R8i
  - ストレージ最適化: I3en
  - 高速コンピューティング: G4dn
- Outposts での Dedicated Hosts は、複数のインスタンスサイズをサポートするように設定できます。複数のインスタンスサイズに対するサポートは、次のインスタンスファミリーで利用できます。
  - 汎用: M5 | M5d | M7i
  - コンピューティング最適化: C5 | C5d | C7i
  - メモリ最適化: R5 | R5d | R7i

  詳細については「Amazon EC2 専有ホストインスタンスのキャパシティ設定」を参照してください。
- Outposts での Dedicated Hosts は、自動配置とターゲットインスタンスの起動をサポートします。詳細については「Amazon EC2 専有ホストの自動配置とホストアフィニティ」を参照してください。
- Outposts での Dedicated Hosts は、ホストアフィニティをサポートします。詳細については「Amazon EC2 専有ホストの自動配置とホストアフィニティ」を参照してください。
- Outposts での Dedicated Hosts は、AWS RAM との共有をサポートしています。詳細については「クロスアカウント Amazon EC2 専有ホストの共有」を参照してください。

考慮事項

- 専有ホスト予約は Outpost ではサポートされていません。
- ホストリソースグループと AWS License Manager は、Outposts ではサポートされていません。
- Outposts 上の Dedicated Hosts は、バースト可能な T3 インスタンスをサポートしていません。
- Outposts の Dedicated Hosts は、ホストの回復をサポートしていません。
- Outposts の専有ホストテナンシーを使用するインスタンスでは、簡易自動復旧はサポートされていません。

AWS Outposts で Amazon EC2 専有ホストを割り当てる

AWS リージョンの Dedicated Hosts の場合と同じ方法で、Outpost に Dedicated Hosts を割り当てて使用します。

前提条件

Outpost にサブネットを作成します。詳細については、AWS Outposts ユーザーガイドのサブネットの作成を参照してください。

Outpost に専有ホストを割り当てるには、次のいずれかの方法を使用します。

Console

AWS Outposts コンソールを使用して Outpost に専有ホストを割り当てるには

1. AWS Outposts コンソールの https://console.aws.amazon.com/outposts/ を開いてください。
2. ナビゲーションペインで、[Outpost] を選択してください。Outpost を選択し、[Actions] (アクション)、[Allocate Dedicated Host] (専有ホストの割り当て) を選択してください。
3. 必要に応じて専有ホストを設定します。詳細については「アカウントで使用するために Amazon EC2 専有ホストを割り当てる」を参照してください。

   > **Note**
   >
   > [アベイラビリティーゾーン] と [Outpost ARN]には、選択した Outpost のアベイラビリティーゾーンと ARN をあらかじめ組み込んでおく必要があります。

4. [割り当て] を選択してください。

Amazon EC2 コンソールを使用して Outpost に専有ホストを割り当てるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインにおいて、[Dedicated Hosts] (専有ホスト) を選択してから、[Allocate Dedicated Host] (専有ホスト割り当て) を選択してください。
3. [アベイラビリティーゾーン] には、Outpost に関連付けられているアベイラビリティーゾーンを選択してください。
4. Outpost ARN には、Outpost の ARN を入力してください。
5. Outpost の特定のハードウェアアセットをターゲットにするには、[Outpost の特定のハードウェアアセットをターゲットにする] で [有効] を選択してください。ターゲットにする各

ハードウェアアセットについて、[アセット ID を追加] を選択し、ハードウェアアセットの ID を入力してください。

> **Note**
>
> [数量] に指定する値は、指定するアセット ID の数と等しくなければなりません。例えば、3 つのアセット ID を指定する場合、数量も 3 でなければなりません。

6. 必要に応じて、残りの専有ホストを設定します。詳細については「アカウントで使用するために Amazon EC2 専有ホストを割り当てる」を参照してください。
7. [割り当て] を選択してください。

AWS CLI

Outpost に専有ホストを割り当てるには

allocate-hosts コマンドを使用します。--availability-zone には、Outpost に関連付けられているアベイラビリティーゾーンを指定します。--outpost-arn には Outpost の ARN を指定します。オプションで、ターゲットにする Outpost ハードウェアアセットの ID を --asset-ids に指定します。

```
aws ec2 allocate-hosts \
    --availability-zone "us-east-1a" \
    --outpost-arn "arn:aws:outposts:us-east-1a:111122223333:outpost/
op-4fe3dc21baEXAMPLE" \
    --asset-ids asset_id \
    --instance-family "m5" \
    --auto-placement "off" \
    --quantity 1
```

PowerShell

Outpost に専有ホストを割り当てるには

New-EC2Host コマンドレットを使用します。Outpost に関連付けられているアベイラビリティーゾーンを指定します。オプションで、ターゲットにする Outpost ハードウェアアセットの ID を -AssetId に指定します。

```
New-EC2Host `
    -AvailabilityZone "us-east-1a" `
```

```
    -OutpostArn "arn:aws:outposts:us-east-1a:111122223333:outpost/
op-4fe3dc21baEXAMPLE" `
    -AssetId asset_id `
    -InstanceFamily "m5" `
    -AutoPlacement "off" `
    -Quantity 1
```

Outpost 上の専有ホストでインスタンスを起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Dedicated Hosts] を選択してください。前のステップで割り当てた専有ホストを選択し、[Actions] (アクション)、[Launch instance onto host] (ホストへのインスタンスの起動) を選択してください。
3. 必要に応じてインスタンスを設定してから、インスタンスを起動します。詳細については「Amazon EC2 専有ホストで Amazon EC2 インスタンスを起動する」を参照してください。

Amazon EC2 専有ホストリカバリ

Dedicated Hosts 自動リカバリでは、Dedicated Host で特定の問題が検出されると、インスタンスが新しい代替ホストで自動的に再起動されます。システム電源やネットワーク接続に関する予期せぬ Dedicated Hosts の障害が発生した場合に、手動による介入の必要性を減らし、運用の負担を軽減します。その他の Dedicated Hosts の問題は、手動での復旧が必要となります。

内容

- Amazon EC2 専有ホスト復旧の仕組み
- サポートされるインスタンスタイプ
- 料金
- Amazon EC2 専有ホスト復旧を管理する
- Amazon EC2 専有ホストのホストリカバリ設定を表示する
- Amazon EC2 専有ホストリカバリでサポートされていないインスタンスを手動で復旧する

Amazon EC2 専有ホスト復旧の仕組み

Dedicated Hosts とホスト Resource Groups のリカバリープロセスは、Dedicated Hosts の可用性を評価し、根本的なシステム障害を検出するために、ホストレベルのヘルスチェックを使用します。Dedicated Hosts の自動回復が可能かどうかは、Dedicated Hosts の障害の種類によって決まり

ます。ホストレベルのヘルスチェックが失敗する場合、原因として以下のような問題が考えられます。

- ネットワーク接続の喪失
- システム電源の喪失
- 物理ホストのハードウェアまたはソフトウェアの問題

> **Important**
> ホストのリタイアが予定されている場合、専有ホストの自動復旧は発生しません。

Dedicated Hosts 自動リカバリ

Dedicated Host でシステム電源やネットワーク接続の障害が検出されると、Dedicated Host の自動リカバリが開始され、Amazon EC2 は自動的に元のDedicated Host と同じアベイラビリティーゾーンに代替の Dedicated Host を割り当てます。代替 Dedicated Host は新しいホスト ID を受け取りますが、元の Dedicated Host と同じ以下の属性を保持します。

- アベイラビリティーゾーン
- インスタンスタイプ
- タグ
- 自動プレイスメントの設定
- 予約する

代替の Dedicated Hosts が割り当てられると、インスタンスは代替 Dedicated Hosts に復旧されます。復旧されたインスタンスは、元のインスタンスと同じ以下の属性を保持します。

- インスタンス ID
- プライベート IP アドレス
- Elastic IP アドレス
- EBS ボリュームアタッチメント
- すべてのインスタンスメタデータ

また、組み込まれている AWS License Manager との統合により、ライセンスの追跡と管理が自動的に行われます。

> **Note**
>
> AWS License Manager との統合は、AWS License Manager を利用できるリージョンでのみサポートされます。

インスタンスと障害が発生した Dedicated Host との間にホストのアフィニティがある場合、復旧したインスタンスは代替 Dedicated Host との間にホストのアフィニティを確立します。

すべてのインスタンスが代替専有ホストに復旧されると、障害が発生した専有ホストがリリースされて、代替専有ホストが使用可能になります。

ホスト復旧が開始されると、AWS アカウントの所有者に対してメールまたは AWS Health Dashboard イベントで通知されます。ホスト復旧が正常に完了すると、2 つ目の通知が送信されます。

AWS License Manager を使用してライセンスを追跡している場合は、その AWS License Manager により、ライセンス設定の制限に応じて新しいライセンスが代替 Dedicated Hosts に割り当てられます。ホスト復旧の結果としてライセンス設定のハード制限を超える場合、復旧プロセスは許可されず、Amazon SNS 通知を介してホスト復旧の失敗が通知されます (AWS License Manager の通知設定が構成されている場合)。ホスト復旧の結果としてライセンス設定のソフト制限を超える場合は、復旧の続行が許可され、Amazon SNS 通知を介して制限の超過が通知されます。詳細については、AWS License Manager ユーザーガイドのライセンス設定の使用およびLicense Manager の設定を参照してください。

ホスト復旧の状態

Dedicated Host の障害が検出されると、障害が発生した Dedicated Host は `under-assessment` 状態になり、すべてのインスタンスは `impaired` 状態になります。障害が起きているDedicated Hostが`under-assessment`状態の間は、このホストでインスタンスを起動できません。

代替 Dedicated Host が割り当てられると、この代替ホストは `pending` 状態になります。ホスト復旧プロセスが完了するまでは、この状態に留まります。代替Dedicated Hostが`pending`状態の間は、このホストでインスタンスを起動できません。代替 Dedicated Host に復旧されたインスタンスは、復旧プロセス中、`impaired` 状態に留まります。

ホスト復旧が完了すると、代替 Dedicated Host は `available` 状態になり、復旧されたインスタンスは `running` 状態に戻ります。代替 Dedicated Host が `available` 状態になると、このホストでインスタンスを起動できます。障害が発生した元の専有ホストは完全にリリースされ、`released-permanent-failure` 状態になります。

障害が発生した専有ホストにホスト復旧をサポートしていないインスタンス (インスタンスストアのルートボリュームを備えたインスタンスなど) がある場合、専有ホストはリリースされません。代わりに、そのホストはリタイアとしてマークされ、`permanent-failure` 状態になります。

Dedicated Hosts の自動リカバリがないシナリオ

ホストのリタイアが予定されている場合、専有ホストの自動復旧は発生しません Amazon CloudWatch Events に関する使用停止通知が AWS Health Dashboard に送信され、AWS アカウント所有者のメールアドレスには Dedicated Hosts の障害に関するメッセージが届きます。指定された期間内にリタイア通知に記載されている修復手順に従って、リタイアするホストのインスタンスを手動で復旧します。

停止したインスタンスは代替 Dedicated Host に復旧されません。障害が発生した Dedicated Host を対象とする、停止したインスタンスを起動しようとすると、インスタンスの起動は失敗します。停止したインスタンスを変更して別の Dedicated Host を対象とするか、停止したインスタンスと一致する設定を持ち、自動配置が有効になっている Dedicated Host で起動することをお勧めします。

インスタンスストレージのあるインスタンスは代替 Dedicated Host に復旧されません。是正措置として、障害が発生した Dedicated Host はリタイアとしてマークされ、ホスト復旧の完了後にリタイア通知が送信されます。リタイア通知に記載されている是正措置に従い、指定された期間内に障害が発生した Dedicated Host の残りのインスタンスを手動で復旧します。

サポートされるインスタンスタイプ

ホスト復旧は以下のインスタンスファミリーでサポートされています。

- 汎用: A1 | M3 | M4 | M5 | M5n | M5zn | M6a | M6g | M6i | T3 | Mac1 | Mac2 | Mac2-m1ultra | Mac2-m2 | Mac2-m2pro | M6in | M7a | M7g | M7i | M8a | M8g | M8gb | M8gn | M8i | M8in | M8ib | M9g | Mac-m4 | Mac-m4pro | Mac-m4max | Mac-m3ultra
- コンピューティング最適化: C3 | C4 | C5 | C5n | C6a | C6g | C6i | C6gn | C6in | C7a | C7g | C7gn | C7i | C8a | C8g | C8gb | C8gn | C8i | C8in | C8ib | C9g
- メモリ最適化: R3 | R4 | R5 | R5b | R5n | R6g | R6i | U-6tb1 | U-9tb1 | U-12tb1 | U-18tb1 | U-24tb1 | X1 | X1e | X2iezn | R6a | R6in | R7a | R7g | R7i | R7iz | R8a | R8g | R8gb | R8gn | R8i | R8in | R8ib | X8g | X8i | U7i-6tb | U7i-8tb | U7i-12tb | U7in-16tb | U7in-24tb | U7in-32tb

- 高速コンピューティング: Inf1 | G3 | G5g | P3

サポートされていないインスタンスを復旧するには、Amazon EC2 専有ホストリカバリでサポートされていないインスタンスを手動で復旧するを参照してください。

> **Note**
>
> サポートされているメタルインスタンスタイプの Dedicated Hosts 自動リカバリは、非メタルインスタンスタイプよりも検出および復旧に時間がかかります。

料金

ホスト復旧の使用に伴う追加の料金はありません。通常の Dedicated Host 料金が適用されます。詳細については、Amazon EC2 Dedicated Hosts 料金を参照してください。

ホスト復旧が開始されると同時に、障害が発生した Dedicated Host には課金されなくなります。代替の専有ホストに対する課金は、専有ホストが `available` 状態になった後でのみ開始されます。

障害が発生した Dedicated Host の課金にオンデマンド料金が使用されていた場合は、代替の Dedicated Host の課金にもオンデマンド料金が使用されます。障害が発生した Dedicated Host でアクティブになっていた Dedicated Host の予約 は、代替の Dedicated Host に転送されます。

Amazon EC2 専有ホスト復旧を管理する

Dedicated Hosts 自動リカバリでは、Dedicated Host で特定の問題が検出されると、インスタンスが新しい代替ホストで自動的に再起動されます。ホストの復旧は、専有ホストの割り当て時または割り当て後に有効にすることができます。

ホストを割り当てるときにホストの復旧を有効にするには、次の手順に従います。

Console

割り当て時にホストの復旧を有効にするには

Amazon EC2 コンソールを使用して専有ホストを割り当てる場合、[ホストの復旧] で、[有効] を選択してください。詳細については「アカウントで使用するために Amazon EC2 専有ホストを割り当てる」を参照してください。

AWS CLI

割り当て時にホストの復旧を有効にするには

allocate-hosts コマンドを使用します。

```
aws ec2 allocate-hosts \
    --instance-type m5.large \
    --availability-zone eu-west-1a \
    --auto-placement on \
    --host-recovery on \
    --quantity 1
```

PowerShell

割り当て時にホストの復旧を有効にするには

New-EC2Host コマンドレットを使用します。

```
New-EC2Host `
    -InstanceType m5.large `
    -AvailabilityZone eu-west-1a `
    -AutoPlacement on `
    -HostRecovery on `
    -Quantity 1
```

専有ホストのホストの復旧を管理するには、次の手順に従います。

Console

ホストの復旧を割り当て後に管理するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Dedicated Hosts] を選択してください。
3. 専有ホストを選択してください。
4. [アクション]、[ホストを変更] の順に選択してください。
5. [ホストの復旧] で、[有効] を選択するかクリアします。
6. [Save] を選択してください。

AWS CLI

ホストの復旧を割り当て後に有効にするには

modify-hosts コマンドを使用します。

```
aws ec2 modify-hosts \
    --host-recovery on \
    --host-ids h-012a3456b7890cdef
```

ホストの復旧を割り当て後に無効にするには

modify-hosts コマンドを使用し、host-recovery パラメータの値を off に指定します。

```
aws ec2 modify-hosts \
    --host-recovery off \
    --host-ids h-012a3456b7890cdef
```

PowerShell

ホストの復旧を割り当て後に有効にするには

Edit-host コマンドレットを使用します。

```
Edit-EC2Host `
    -HostRecovery on `
    -HostId h-012a3456b7890cdef
```

ホストの復旧を割り当て後に無効にするには

Edit-EC2Host コマンドレットを使用します。

```
Edit-EC2Host `
    -HostRecovery off `
    -HostId h-012a3456b7890cdef
```

Amazon EC2 専有ホストのホストリカバリ設定を表示する

Dedicated Host のホスト復旧の設定はいつでも表示できます。

Console

専有ホストのホスト復旧設定を閲覧するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

2. ナビゲーションペインで [Dedicated Hosts] を選択してください。
3. Dedicated Host を選択し、[Description (説明)] タブの [Host Recovery (ホスト復旧)] フィールドを確認します。

AWS CLI

専有ホストのホスト復旧設定を閲覧するには

describe-hosts コマンドを使用します。

```
aws ec2 describe-hosts \
    --host-ids h-012a3456b7890cdef \
    --query Hosts[].HostRecovery
```

以下は出力の例です。

```
on
```

PowerShell

専有ホストのホスト復旧設定を閲覧するには

Get-EC2Host コマンドレットを使用します。

```
(Get-EC2Host -HostId h-012a3456b7890cdef).Hosts | Select HostRecovery
```

以下は出力の例です。

```
HostRecovery
------------
on
```

Amazon EC2 専有ホストリカバリでサポートされていないインスタンスを手動で復旧する

ホスト復旧は、インスタンスストアボリュームを使用するインスタンスの復旧をサポートしていません。自動的に復旧されないインスタンスがある場合は、以下の手順に従って、これらのインスタンスを手動で復旧します。

> **Warning**
>
> インスタンスストアボリュームのデータは、インスタンスの停止、休止、または終了に伴って失われます。これにはEBS ルートボリュームを備えたインスタンスにアタッチされたインスタンスストアボリュームも含まれます。インスタンスストアボリュームのデータを保護するには、インスタンスが停止または終了する前に、データを永続的ストレージにバックアップします。

EBS-backed インスタンスの手動復旧

自動的に復旧されない EBS-backed インスタンスの場合は、インスタンスを手動で停止または終了させて、新しい Dedicated Host に復旧することをお勧めします。インスタンスの停止や、インスタンスの停止に伴うインスタンス設定の変更の詳細については、Amazon EC2 インスタンスの停止と開始を参照してください。

インスタンスストアのルートボリュームを備えたインスタンスの手動復旧

自動的に復旧されない、インスタンスストアのルートボリュームを備えたインスタンスの場合は、以下の操作を行うことをお勧めします。

1. 新しい Dedicated Host で、最新の AMI から代替インスタンスを起動します。
2. すべての必要なデータを代替インスタンスに移行させます。
3. 障害が発生した Dedicated Host で元のインスタンスを終了します。

Amazon EC2 専有ホストのホストメンテナンス

ホストのメンテナンスによって、まれに専有ホストのパフォーマンスが低下することがあります。その場合は、ホストで実行中のインスタンスが、正常な代替専有ホストに自動的に移行します。これにより、ワークロードのダウンタイムが最小限に抑えられ、専有ホストの管理を簡素化することができます。ホストのメンテナンスは、Amazon EC2 の計画的かつ日常的なメンテナンスのためにも行われます。

Amazon EC2 は、次の 2 つのタイプのホストメンテナンスをサポートしています。

- ライブ移行ホストメンテナンス - インスタンスは 24 時間以内に代替ホストに自動的に移行されます。いったん停止してから再起動するという操作は必要ありません。
- 再起動ベースのホストメンテナンス - インスタンスの再起動スケジュールイベントがスケジュールされ、イベントが発生するとインスタンスは自動的に停止して代替ホストで再起動されます。

内容

- ホストメンテナンスとホスト復旧
- 考慮事項
- 関連サービス
- 料金
- Amazon EC2 専有ホストのホストメンテナンスの仕組み
- Amazon EC2 専有ホストのホストメンテナンス設定を構成する

ホストメンテナンスとホスト復旧

次の表は、ホストメンテナンスとホスト復旧の主な違いを示しています。

	ホスト復旧	ホストのメンテナンス
インスタンスの到達可能性	到達不能	到達可能
専有ホストの状態	`under-assessment`	`permanent-failure`
リソースグループをホストする	サポート	サポートされていません

ホスト復旧の詳細については、「ホスト復旧」を参照してください。

考慮事項

- ホストメンテナンスは、中国リージョンと AWS GovCloud (US) Regionsを除くすべての AWS リージョンで使用できます。
- ホストのメンテナンスは、AWS Outposts、AWS ローカルゾーンと AWS Wavelength ゾーンではサポートされていません。
- ホストリソースグループ内に既に含まれているホストについては、ホストメンテナンスをオンまたはオフにすることはできません。ホストリソースグループに追加されたホストは、そのホストメンテナンス設定を保持します。詳細については、「ホストリソースグループ」を参照してください。
- ホストメンテナンスは、次のインスタンスタイプではサポートされていません: C1、C3、D2、I2、M1、M2、M3、R3、X1。これらには、インスタンスストアのルートボリュームがあるためです。

関連サービス

専有ホストと AWS License Manager との統合 - Amazon EC2 Dedicated Hosts 全体でライセンスを追跡します (AWS License Manager が利用可能なリージョンでのみサポートされます)。詳細については、「AWS License Manager ユーザーガイド」を参照してください。

新しい専有ホストには AWS アカウント で十分なライセンスが必要です。スケジュールされたイベントの完了後にパフォーマンスが低下したホストが解放されると、そのホストに関連付けられたライセンスが解放されます。

料金

ホストメンテナンスの使用に伴う追加の料金はありません。通常の専有ホスト料金が適用されます。詳細については、Amazon EC2 Dedicated Hosts 料金を参照してください。

ホストメンテナンスが開始されると同時に、デグレードした専有ホストには課金されなくなります。代替の専有ホストに対する課金は、専有ホストが `available` 状態になった後でのみ開始されます。

デグレードした専有ホストの課金にオンデマンド料金が使用されていた場合は、代替の専有ホストの課金にもオンデマンド料金が使用されます。デグレードした専有ホストでアクティブになっていた専有ホストの予約は、新しい専有ホストに転送されます。

Amazon EC2 専有ホストのホストメンテナンスの仕組み

ホストメンテナンスが有効になっている専有ホストでパフォーマンスの低下が検出されると、アカウント内の代替専有ホストが自動的に割り当てられます。代替 Dedicated Host は新しいホスト ID を受け取りますが、元の Dedicated Host と同じ以下の属性を保持します。

- 自動プレイスメントの設定
- アベイラビリティーゾーン
- 専有ホスト予約の関連付け
- ホストのアフィニティ
- ホストのメンテナンス設定
- ホストの復旧設定
- インスタンスタイプ
- タグ

代替ホストが割り当てられると、インスタンスに応じてライブ移行ホストメンテナンスまたは再起動ベースのホストメンテナンスを使用してインスタンスが移行されます。

パフォーマンスが低下したホストで動作中のインスタンスがなくなると、そのホストはアカウントから完全に解放されます。

ライブ移行ホストメンテナンス

ライブ移行ホストメンテナンスを必要とするインスタンスは、24 時間以内に代替ホストに自動的に移行します。いったん停止してから再起動するという操作は必要ありません。移行されたインスタンスは、次のような既存の属性を保持します。

- [インスタンス ID]
- インスタンスメタデータ
- Amazon EBS ボリュームアタッチメント
- Elastic IP アドレスとプライベート IP アドレス
- メモリ、CPU、およびネットワークの状態

インスタンスのサイズがある程度大きくなると、移行中にパフォーマンスがわずかに低下することがあります。

インスタンスが代替ホストに自動的に移行すると、E メールと AWS Health ダッシュボードの通知がお手元に届きます。通知には、パフォーマンスが低下したホストと代替ホストの ID、ライブ移行ホストメンテナンスを使用して自動的に移行したインスタンスに関する情報、移行せずに残ったインスタンスに関する情報、が記載されています。

再起動ベースのホストメンテナンス

インスタンスで再起動ベースのホストメンテナンスが必要になると、通知日付から 14 日間、インスタンスの再起動スケジュールイベントがスケジュールされます。スケジュールされたイベントが始まるまでは、パフォーマンスが低下した専有ホスト上のインスタンスにこれまで通りアクセスできます。

元のイベント日時の 7 日以内の日付であれば、再起動イベントを再スケジュールできます。詳細については「EC2 インスタンスのスケジュールされたイベントを再スケジュールする」を参照してください。

Amazon EC2 は、代替ホスト上でこうしたインスタンスのためにキャパシティを自動的にリザーブします。こうしたリザーブドキャパシティでインスタンスを実行することはできません。

Amazon EC2 コンソールには、リザーブドキャパシティが使用済みキャパシティとして表示されます。インスタンスは、デグレードしたホストと代替ホストの両方で実行されているように見える場合

があります。ただし、インスタンスは、停止するか、代替ホストのリザーブドキャパシティに移行するまで、デグレードしたホストでのみ引き続き実行されます。

スケジュールされたイベントの日時になると、インスタンスは自動的に停止し、代替ホストのリザーブドキャパシティで再起動されます。移行されたインスタンスは、次のような既存の属性を保持します。

- [インスタンス ID]
- インスタンスメタデータ
- Amazon EBS ボリュームアタッチメント
- Elastic IP アドレスとプライベート IP アドレス

ただし、移行中にいったん停止してから再起動されるため、インスタンスはメモリ、CPU、ネットワークの状態を保持しません。

また、スケジュールされたイベントが始まる前にこれらのインスタンスをいつでも手動で停止し再起動してから、代替ホストまたは別のホストに移行できます。場合によっては、別のホストで再起動するようにインスタンスのホストアフィニティを変更する必要があります。スケジュールされたイベントが始まる前にインスタンスを停止すると、代替ホスト上のリザーブドキャパシティが解放されて使用可能になります。

ホストメンテナンスの状態

ホストのパフォーマンスが低下すると、ホストの状態が `permanent-failure` になります。状態が `permanent-failure` の専有ホストではインスタンスを起動できません。

代替ホストが割り当てられると、ライブ移行ホストメンテナンスをサポートするインスタンスがパフォーマンスの低下したホストから自動的に移行し、スケジュールされたイベントが残りのインスタンスに対してスケジュールされるまで、代替ホストは `pending` 状態のままになります。これらのタスクが完了すると、代替ホストの状態は `available` に変わります。

代替ホストの状態が `available` になったら、アカウント内の他のホストと同じように使用することができます。ただし、代替ホストのインスタンスキャパシティの一部が、再起動ベースのホスト移行を必要とするインスタンス用にリザーブされます。このリザーブドキャパシティで新しいインスタンスを起動することはできません。

パフォーマンスが低下したホストに動作中のインスタンスがなくなると、ホストは `released`, `permanent-failure` 状態になってアカウントから完全に解放されます。ホストとそのリソースは、その後しばらくコンソールに表示され続けます。

自動移行

一部のインスタンスは、代替ホストに自動的には移行できません。

EBS-backed ルートボリュームを備えたインスタンス

こうしたインスタンスの場合、通知日付から 28 日間、インスタンスの停止イベントがスケジュールされます。スケジュールされたイベントの日時になると、インスタンスは停止します。インスタンスを手動で停止して代替ホストまたは別のホストで再起動することをお勧めします。場合によっては、別のホストで再起動するようにインスタンスのホストアフィニティを変更する必要があります。

インスタンスストアのルートボリュームを持つインスタンス

こうしたインスタンスの場合、通知日付から 28 日間、インスタンスの廃止イベントがスケジュールされます。スケジュールされたイベントの日時になると、インスタンスは完全に終了します。スケジュールされたイベントが始まる前に、代替ホスト上で代替インスタンスを手動で開始して、必要なデータを代替インスタンスに移行することをお勧めします。

次のインスタンスには、インスタンスストアのルートボリュームがあります:
C1、C3、D2、I2、M1、M2、M3、R3、X1。

スケジュールされたイベントが始まるまでは、パフォーマンスが低下した専有ホスト上のインスタンスにこれまで通りアクセスできます。

Amazon EC2 専有ホストのホストメンテナンス設定を構成する

ホストメンテナンスを有効にすると、スケジュールされたメンテナンスイベント中に、専有ホストで実行されているインスタンスが新しい専有ホストに自動的に復旧されるようになります。

ホストのメンテナンスを無効にすると、デグレードしたホストをエビクションし、28 日以内にインスタンスを別のホストに手動で移行するよう求めるメール通知が届きます。専有ホストを予約している場合は、代替ホストが割り当てられます。28 日後、デグレードしたホストで実行されていたインスタンスは終了し、そのホストは自動的にリリースされます。

Console

専有ホストのホストメンテナンスを有効にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Dedicated Hosts] を選択してください。
3. [専有ホスト]、[アクション]、[ホストの変更] の順に選択してください。
4. [ホストのメンテナンス] フィールドで [オン] を選択してください。

専有ホストのホストメンテナンスを無効にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Dedicated Hosts] を選択してください。
3. [専有ホスト]、[アクション]、[ホストの変更] の順に選択してください。
4. [ホストのメンテナンス] フィールドで [オフ] を選択してください。

AWS CLI

専有ホストのホストメンテナンスを有効にするには

modify-hosts コマンドを使用します。

```
aws ec2 modify-hosts \
    --host-maintenance on \
    --host-ids h-0d123456bbf78910d
```

専有ホストのホストメンテナンスを無効にするには

modify-hosts コマンドを使用します。

```
aws ec2 modify-hosts \
    --host-maintenance off \
    --host-ids h-0d123456bbf78910d
```

PowerShell

専有ホストのホストメンテナンスを有効にするには

Edit-EC2Host コマンドレットを使用します。

```
Edit-EC2Host `
    -HostMaintenance on `
    -HostId h-0d123456bbf78910d
```

専有ホストのホストメンテナンスを無効にするには

Edit-EC2Host コマンドレットを使用します。

```
Edit-EC2Host `
    -HostMaintenance off `
```

```
    -HostId h-0d123456bbf78910d
```

Amazon EC2 専有ホストの状態をモニタリングする

Amazon EC2 は、Dedicated Hosts の状態を絶えずモニタリングします。更新は Amazon EC2 コンソールで伝達されます。Dedicated Host に関する情報は、次の方法を使用して表示できます。

Console

Dedicated Host の状態を表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Dedicated Hosts] を選択してください。
3. リストで Dedicated Host を見つけ、[State] 列で値を確認します。

AWS CLI

Dedicated Host の状態を表示するには

describe-hosts コマンドを使用します。

```
aws ec2 describe-hosts --host-id h-012a3456b7890cdef
```

PowerShell

Dedicated Host の状態を表示するには

Get-EC2Host コマンドレットを使用します。

```
Get-EC2Host -HostId h-012a3456b7890cdef
```

以下の表では、表示される可能性のある Dedicated Host の状態について説明します。

状態	説明
available	AWS は Dedicated Host で問題を検出しませんでした。予定されているメンテナンスまたは修復はありません。この専有ホストでインスタンスを起動できます。

状態	説明
released	専有ホストがリリースされました。ホスト ID は使用中ではありません。リリース済みのホストは再使用できません。
under-assessment	AWS は Dedicated Host の潜在的な問題を調査しています。アクションを実行する必要がある場合は、AWS マネジメントコンソールまたは E メールで通知されます。この状態のDedicated Hostではインスタンスを起動できません。
pending	専有ホストは新しいインスタンスの起動には使用できません。この状態は、複数のインスタンスタイプをサポートするように修正されている状態か、ホスト復旧実行中の状態です。
permanent-failure	回復不可能な障害が検出されました。インスタンスおよび E メールで削除通知が届きます。インスタンスは引き続き実行する場合があります。この状態にある Dedicated Host 上のすべてのインスタンスを停止または終了すると、AWS はホストを使用停止にします。AWS はこの状態のインスタンスを再起動しません。この状態のDedicated Hostsではインスタンスを起動できません。
released-permanent-failure	AWS は、障害が発生してインスタンスが実行されていない Dedicated Hosts を完全にリリースします。Dedicated Host ID も使用できなくなります。

AWS Config を使用して、Amazon EC2 専有ホストでの設定の変更を追跡する

AWS Config を使用すると、Dedicated Hosts の設定変更や、Dedicated Hosts 上で起動、停止、終了されたインスタンスの設定変更を記録できます。そして、AWS Config でキャプチャされた情報をライセンスレポートのデータソースとして使用することができます。

AWS Config は、Dedicated Hosts やインスタンスの設定情報を個別に記録し、関係を利用してそれぞれの設定情報をペアにします。3 つのレポート条件があります。

- AWS Config の記録ステータス – [オン] のとき、AWS Config は 1 つ以上の AWS リソースタイプを記録中です。記録の対象には、Dedicated Hosts や ハードウェア専有インスタンス も含まれます。ライセンスレポートに必要な情報をキャプチャするには、次のフィールドによって Host とインスタンスが記録されていることを確認します。

- Host recording status — [Enabled] の場合は、Dedicated Hosts の設定情報が記録されます。
- インスタンスの記録ステータス — [Enabled (有効)] の場合は、ハードウェア専有インスタンス の設定情報が記録されます。

これら 3 つの条件のいずれかが無効になっている場合、[Config 記録の編集] ボタン内のアイコンは赤です。このツールのメリットをすべて引き出すために、3 つの記録方法すべてを有効にしてください。3 つすべてが有効なとき、アイコンは緑です。設定を編集するには、[Config 記録の編集] を選択してください。AWS Config コンソールに [Set up AWS Config] ページが表示され、そこで AWS Config を設定し、ホスト、インスタンス、およびその他のサポートされるリソースタイプの記録を開始できます。詳細については、『AWS Config デベロッパーガイド』のコンソールを使用した AWS Config のセットアップを参照してください。

> **Note**
>
> AWS Config はリソースを検出 (数分かかる場合があります) して、記録します。

AWS Config がホストおよびインスタンスへの設定変更の記録を開始した後、ユーザーが割り当てたかリリースしたホストと、起動、停止、または終了したインスタンスの設定履歴を取得できます。例えば、Dedicated Host の設定履歴の任意の時点で、そのホストのソケット数とコア数と共に、そのホストで起動されているインスタンスの数を調べることができます。それらのインスタンスについても、対応する Amazon マシンイメージ (AMI) の ID を調べることができます。これらの情報を使用して、ソケット単位またはコア単位でライセンスが与えられているサーバーバインドソフトウェアのライセンスに関するレポートを作成できます。

設定履歴は以下のいずれかの方法で閲覧できます。

- AWS Config コンソールを使用する。記録されたリソースごとに、設定の詳細の履歴を提供するタイムラインページを表示することができます。このページを表示するには、[Dedicated Hosts] ページの [設定タイムライン] 列にあるグレーのアイコンを選択してください。詳細については、『AWS Config デベロッパーガイド』のAWS Config コンソールでの設定詳細の表示を参照してください。
- AWS CLI コマンドを実行する。まず、list-discovered-resources コマンドを使用して、すべてのホストとインスタンスのリストを取得できます。次に、get-resource-config-history コマンドを使用して、特定の時間間隔でホストまたはインスタンスの設定の詳細を取得できます。
- アプリケーションで AWS Config API を使用する。まず、ListDiscoveredResources アクションを使用して、すべてのホストとインスタンスのリストを取得できます。次

に、GetResourceConfigHistory アクションを使用して、特定の時間間隔でホストまたはインスタンスの設定の詳細を取得できます。

例えば、AWS Config から Dedicated Hosts のリストを取得するには、次のような CLI コマンドを実行します。

```
aws configservice list-discovered-resources --resource-type AWS::EC2::Host
```

AWS Config から Dedicated Hosts の設定履歴を取得するには、次のような CLI コマンドを実行します。

```
aws configservice get-resource-config-history \
    --resource-type AWS::EC2::Instance \
    --resource-id i-1234567890abcdef0
```

コンソールを使用して AWS Config の設定を管理するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. [Dedicated Hosts] ページで、[Config 記録の編集] を選択してください。
3. AWS Config コンソールで、次の手順に従って記録をオンにします。詳細については、コンソールを使用した AWS Config のセットアップを参照してください。

詳細については、AWS Config コンソールでの設定詳細の表示を参照してください。

コマンドラインまたは API を使用して AWS Config をアクティブ化するには

- AWS CLI: AWS CLI デベロッパーガイドの設定詳細の表示 (AWS Config)
- Amazon EC2 API: GetResourceConfigHistory

Amazon EC2 ハードウェア専有インスタンス

デフォルトでは、EC2 インスタンスは共有テナンシーハードウェアで実行されます。つまり、複数の AWS アカウントが同じ物理ハードウェアを共有する可能性があります。

ハードウェア専有インスタンスとは、単一の AWS アカウント専用のハードウェア上で動作する EC2 インスタンスです。つまり、ハードウェア専有インスタンスは、それらのアカウントが単一の支払者アカウントにリンクされている場合でも、他の AWS アカウント に属するインスタンスから

ホストハードウェアレベルで物理的に分離されています。ただし、ハードウェア専有インスタンスは、同じ AWS アカウント に属する、ハードウェア専有インスタンスではない他のインスタンスとはハードウェアを共有できます。

ハードウェア専有インスタンスは、インスタンスの配置を可視化したり制御したりせず、ホストアフィニティもサポートしません。ハードウェア専有インスタンスを停止して起動すると、同じホストで実行されない場合があります。同様に、インスタンスを起動または実行する特定のホストをターゲットにすることはできません。さらに、ハードウェア専有インスタンスでは、Bring-Your-Own-License (BYOL) のサポートが制限されています。

インスタンス配置の可視性と制御、およびより包括的な BYOL サポートが必要な場合は、代わりに専有ホストの使用を検討してください。ハードウェア専有インスタンスと専有ホストのどちらを使用しても、専用の物理サーバーに Amazon EC2 インスタンスを起動することができます。ハードウェア専有インスタンス と Dedicated Hosts のインスタンスの間に、パフォーマンス、セキュリティ、または物理的な違いはありません。ただし、これらにはいくつかの重要な違いがあります。次のテーブルでは、Dedicated Hosts とハードウェア専有インスタンスの主な違いをいくつか紹介します。

	Dedicated Host	Dedicated Instance
専用物理サーバー	お客様専用のインスタンス容量を持つ物理サーバー。	単一の顧客アカウント専用の物理サーバー。
インスタンス容量の共有	インスタンス容量を他のアカウントと共有できます。	サポートされていません
請求	ホストごとの請求	インスタンスごとの請求
ソケット、コア、ホスト ID の可視性	ソケットと物理コアの数が見える	可視性なし
ホストおよびインスタンスアフィニティ	インスタンスを同じ物理サーバーに徐々にデプロイし続けることができる	サポート外
ターゲットを絞ったインスタンスの配置	インスタンスを物理サーバーに配置する方法についての可視性と制御が高い	サポート外

	Dedicated Host	Dedicated Instance
インスタンスの自動復旧	サポート対象。詳細については、Amazon EC2 専有ホストリカバリを参照してください。	サポート対象
Bring-Your-Own-License (BYOL)	サポート	部分的なサポート*
キャパシティ予約	サポート外	サポート

* ソフトウェアアシュアランスによるライセンスモビリティを使用する Microsoft SQL Server、および Windows Virtual Desktop Access (VDA) ライセンスを、ハードウェア専有インスタンスで使用することが可能です。

詳細については、「Amazon EC2 専有ホスト」を参照してください。

内容

- ハードウェア専有インスタンス の基本
- サポートされている機能
- ハードウェア専有インスタンス の制限事項
- ハードウェア専有インスタンス の料金表
- デフォルトのテナンシーで VPC に専有インスタンスを起動する
- EC2 インスタンスのテナンシーを変更する
- VPC のインスタンステナンシーを変更する

ハードウェア専有インスタンス の基本

VPC は、`default`または`dedicated`のテナンシーを持つことができます。デフォルトでは、VPC `default` には `default` テナンシーがあり、テナンシー VPC `default` に起動されたインスタンスにはテナンシーがあります。ハードウェア専有インスタンス は次の手順を実行します。

- テナント属性が dedicated VPC を作成すると、VPC のすべてのインスタンスが専有インスタンスとして実行されます。詳細については、「デフォルトのテナンシーで VPC に専有インスタンスを起動する」を参照してください。
- テナンシー default の VPC を作成し、インスタンスを専用インスタンスとして実行するテナンシー dedicated を手動で指定します。詳細については、「デフォルトのテナンシーで VPC に専有インスタンスを起動する」を参照してください。

サポートされている機能

ハードウェア専有インスタンスは、以下の機能とAWSサービスの統合をサポートしています:

機能

- 予約インスタンス
- Auto Scaling
- 自動復旧
- ハードウェア専有スポットインスタンス
- バーストパフォーマンスインスタンス

予約インスタンス

ハードウェア専有インスタンスのキャパシティを予約するには、専有リザーブドインスタンスまたはキャパシティ予約を購入します。詳細については、「Amazon EC2 のリザーブドインスタンスの概要」および「EC2 オンデマンドキャパシティ予約を使用してコンピューティングキャパシティを予約」を参照してください。

専有リザーブドインスタンス を購入すると、格安の料金で専用インスタンスを起動する容量を利用できます。使用料の割引は、専用テナントでインスタンスを起動する場合にのみ適用されます。デフォルトテナンシーで リザーブドインスタンス を購入する場合、これは default テナンシーがある実行中のインスタンスにのみ適用され、dedicated テナンシーがある実行中のインスタンスには適用されません。

さらに、リザーブドインスタンス の購入後に変更プロセスを使用してそのテナンシーを変更することはできません。ただし、新しい コンバーティブルリザーブドインスタンス の コンバーティブルリザーブドインスタンス を別のテナンシーと交換することはできます。

Auto Scaling

Amazon EC2 Auto Scaling を使用して ハードウェア専有インスタンス を起動できます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「詳細設定を使用して起動テンプレートを作成する」を参照してください。

自動復旧

基盤ハードウェアの障害、またはAWS による修復を必要とする問題によって正常に機能しなくなった場合のために、ハードウェア専有インスタンス に対し自動復旧を設定できます。詳細については、インスタンスの自動復旧を参照してください。

ハードウェア専有スポットインスタンス

スポットインスタンスのリクエストを作成するとき、dedicated のテナントを指定することにより、ハードウェア専有スポットインスタンスを実行できます。詳細については、シングルテナントハードウェアで起動するを参照してください。

バーストパフォーマンスインスタンス

the section called “バーストパフォーマンスインスタンス” では、専有テナントハードウェアで実行することの利点を活用できます。T3 ハードウェア専有インスタンスは、デフォルトで Unlimited モードで起動します。また、ベースラインレベルの CPU パフォーマンスを提供し、ワークロードの必要に応じてより高い CPU レベルにバーストできます。T3 ベースラインパフォーマンスとバースト機能は、CPU クレジットによって管理されます。T3 インスタンスタイプはバーストであるため、最適なパフォーマンスを得るために T3 インスタンスで専用ハードウェアの CPU リソースをどのように使用しているかをモニタリングすることをお勧めします。T3 ハードウェア専有インスタンスは、お客様のワークロードが多様で CPU がランダムな動作を示すが、平均的な CPU 使用量が適切なベースライン使用量以下である場合に向いています。詳細については、the section called “主要なコンセプト”を参照してください。

Amazon EC2 には、パフォーマンスの変動を特定して修正するためのシステムが用意されています。ただし、CPU 使用パターンが相関する複数の T3 ハードウェア専有インスタンスを起動すると、依然として短期的な変動が発生する可能性があります。これらのより要求の厳しいワークロードや相関関係のあるワークロードについては、T3 ハードウェア専有インスタンスではなく、M5 または M5a ハードウェア専有インスタンスを使用することをお勧めします。

ハードウェア専有インスタンス の制限事項

ハードウェア専有インスタンスを使用するときは、以下の点を常に考慮する必要があります。

- 一部の AWS のサービスまたは機能は、インスタンスのテナント属性が dedicated に設定されている VPC ではサポートされていません。そのほかにも制限事項があるかどうかを確認するには、個別のサービスのドキュメントを参照してください。
- 一部の種類のインスタンスは、インスタンスのテナント属性が dedicated に設定されている VPC では起動できません。サポートされているインスタンスの種類の詳細については、Amazon EC2 ハードウェア専有インスタンスを参照してください。
- Amazon EBSでバックアップされたハードウェア専有インスタンスを起動する場合、シングルテナントのハードウェアで EBS ボリュームは実行できません。

ハードウェア専有インスタンス の料金表

ハードウェア専有インスタンス の料金表は、オンデマンドインスタンスの料金表と異なります。詳細については、「Amazon EC2 ハードウェア専有インスタンス」を参照してください。

デフォルトのテナンシーで VPC に専有インスタンスを起動する

VPC を作成するときにインスタンスのテナント属性を指定できます。インスタンスのテナンシーが dedicated である VPC 内にインスタンスを起動すると、そのインスタンスは専用ハードウェア上で専有インスタンスとして実行されます。

テナント属性として host を使用したインスタンスの作成の詳細については、「Amazon EC2 専有ホストで Amazon EC2 インスタンスを起動する」を参照してください。

VPC テナンシーオプションの詳細については、「Amazon VPC ユーザーガイド」の「VPC を作成する」を参照してください。

要件

- サポートされているインスタンスタイプを選択します。詳細については、Amazon EC2 ハードウェア専有インスタンスを参照してください。

Console

デフォルトテナンシー VPC にハードウェア専有インスタンスを起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Instances] (インスタンス)、[Launch instance] (インスタンスを起動) の順に選択してください。

3. [Application and OS Images] (アプリケーションと OS イメージ) セクションにあるリストから、使用する AMI を選択します。
4. [Instance type] (インスタンスタイプ) セクションで、起動するインスタンスタイプを選択してください。
5. [Key pair] (キーペア) セクションで、インスタンスに関連付けるキーペアを選択してください。
6. [Advanced details] (高度な詳細) セクションにある [Tenancy] (テナンシー) で、[Dedicated] (ハードウェア専有) を選択します。
7. 必要に応じて、残りのインスタンスオプションを設定します。詳細については、「Amazon EC2 インスタンス設定パラメータのリファレンス」を参照してください。
8. [インスタンスを起動] を選択してください。

AWS CLI

起動中にインスタンスのテナンシーオプションを設定するには

run-instances コマンドを使用し、`--placement` オプションに `Tenancy` を含めます。

```
--placement Tenancy=dedicated
```

PowerShell

起動中にインスタンスのテナンシーオプションを設定するには

`-Placement_Tenancy` パラメータで New-EC2Instance コマンドレットを使用します。

```
-Placement_Tenancy dedicated
```

EC2 インスタンスのテナンシーを変更する

インスタンスの起動後に停止されたテナンシーを変更できます。加えた変更は、次回のインスタンス起動時に有効になります。

または、仮想プライベートクラウド (VPC) のテナンシーを変更することもできます。詳細については、「the section called “VPC のテナント属性の変更”」を参照してください。

制限

- AWS マネジメントコンソールを使ってインスタンスのテナンシーを変更することはできません。
- インスタンスは stopped の状態である必要があります。
- インスタンスのオペレーティングシステムの詳細、および SQL Server がインストールされているかどうかによって、サポートされる変換が影響されます。インスタンスで使用できるテナンシー変換パスの詳細については、「License Manager ユーザーガイド」の「テナンシー変換」を参照してください。
- T3 インスタンスの場合、host のテナンシーを使用するには専有ホストでインスタンスを起動する必要があります。テナンシーを host から dedicated または default に変更することはできません。これらのサポートされていないテナンシー変更のいずれかを試みると、エラーコード InvalidRequest が発生します。

AWS CLI

インスタンスのテナンシー値を変更するには

modify-instance-placement コマンドを使用します。

```
aws ec2 modify-instance-placement \
    --instance-id i-1234567890abcdef0 \
    --tenancy dedicated
```

PowerShell

インスタンスのテナンシー値を変更するには

Edit-EC2InstancePlacement コマンドレットを使用します。

```
Edit-EC2InstancePlacement `
    -InstanceId i-1234567890abcdef0 `
    -Tenancy Dedicated
```

VPC のインスタンステナンシーを変更する

仮想プライベートクラウド (VPC) の作成後に VPC のインスタンステナンシーを dedicated から default に変更することができます。VPC のインスタンステナンシーを変更しても、VPC 内の既

存のインスタンスのテナンシーには影響が及びません。次回 VPC でインスタンスを起動すると、インスタンス起動時に指定していなければ、テナンシーが default になります。

あるいは、特定のインスタンスのテナンシーを変更することもできます。詳細については、「the section called “インスタンスのテナンシーの変更”」を参照してください。

制限

- VPC の作成後に、VPC のインスタンステナンシーを default から dedicated に変更することはできません。
- AWS マネジメントコンソール を使用して VPC インスタンスのテナントを変更することはできません。

AWS CLI

VPC のインスタンステナンシー属性を変更するには

modify-vpc-tenancy コマンドを使用します。サポートされているテナンシー値は default のみです。

```
aws ec2 modify-vpc-tenancy \
    --vpc-id vpc-1234567890abcdef0 \
    --instance-tenancy default
```

PowerShell

VPC のインスタンステナンシー属性を変更するには

Edit-EC2VpcTenancy コマンドレットを使用します。サポートされているテナンシー値は Default のみです。

```
Edit-EC2VpcTenancy `
    -VpcId vpc-1234567890abcdef0 `
    -InstanceTenancy Default
```

機械学習用のオンデマンドキャパシティ予約とキャパシティブロック

キャパシティ予約を使用すると、特定のアベイラビリティーゾーンで、Amazon EC2 インスタンスの計算能力を予約できます。キャパシティ予約には 2 種類あり、対応するユースケースが異なります。

キャパシティ予約の種類

- オンデマンドキャパシティ予約
- 機械学習用のキャパシティブロック

オンデマンドキャパシティ予約の一般的なユースケースは以下のとおりです。

- スケーリングのイベント — ビジネスクリティカルなイベントの前にオンデマンドキャパシティ予約を作成しておくと、必要なときに確実にスケールすることができます。
- 規制要件とディザスタリカバリ — 高可用性に関する規制要件を満たしたり、ディザスタリカバリ用に別のアベイラビリティーゾーンまたはリージョンにキャパシティを予約したりするときは、オンデマンドキャパシティ予約を使用します。
- 未使用のキャパシティの共有 – 中断可能なキャパシティ予約を使用して、アカウント内の他のワークロードで未使用のキャパシティを利用できるようにしつつ、必要に応じて再利用するための制御を維持します。

ML 用のキャパシティブロックの一般的なユースケースは以下のとおりです。

- 機械学習 (ML) モデルトレーニングと微調整 — ML モデルトレーニングと微調整を完了するために予約した GPU インスタンスに、中断なしにアクセスできます。
- ML 実験とプロトタイプ — GPU インスタンスを必要とする実験の実行およびプロトタイプの構築を短期間で行えます。

オンデマンドキャパシティ予約の使用時期

キャパシティの要件が厳格で、現在または将来のビジネスクリティカルなワークロードにキャパシティ保証が必要な場合、オンデマンドキャパシティ予約を使用します。オンデマンドキャパシティ予約を使用すると、予約した Amazon EC2 キャパシティに必要な限りアクセスすることができます。

機械学習用のキャパシティブロックを使用する時期

機械学習用のキャパシティブロックは、将来の一定期間、GPU インスタンスに中断なしにアクセスできるようにする必要がある場合に使用します。キャパシティブロックは、ML モデルのトレーニングや微調整、短期間の試験実行、将来、推論の需要が一時的に急増した場合の対応、に最適です。キャパシティブロックを使用すると、特定の日に GPU リソースにアクセスして確実に ML ワークロードを実行することができます。

EC2 オンデマンドキャパシティ予約を使用してコンピューティングキャパシティを予約

Amazon EC2 キャパシティ予約を使用すると、任意の期間で特定のアベイラビリティーゾーンに Amazon EC2 インスタンスのコンピューティングキャパシティを予約できます。一定レベルの長期または短期のキャパシティ保証が必要な現在または今後のビジネスクリティカルなワークロードに対し、厳格なキャパシティ要件がある場合、必要な限りにいつでも Amazon EC2 キャパシティにアクセスできるようにするため、キャパシティ予約を作成することをお勧めします。

キャパシティ予約はいつでも作成でき、開始時期を選択できます。すぐに使用できるキャパシティ予約をリクエストするか、将来の日付のキャパシティ予約をリクエストできます。

- すぐに使用できるキャパシティ予約をリクエストする場合、キャパシティ予約はすぐに使用可能で期間のコミットメントはありません。キャパシティ予約はいつでも変更できます。また、いつでもキャンセルしてリザーブドキャパシティを解放することで請求の発生を停止できます。
- 将来の日付のキャパシティ予約をリクエストする場合は、キャパシティが必要なタイミングと、キャパシティの維持をコミットする期間を指定します。指定された将来の日付になるとキャパシティ予約が使用可能になり、請求が開始されます。コミットメント期間中、インスタンス数を少なくすることやコミットメント期間を最初のコミットメントより短くすることはできません。キャパシティ予約はキャンセルできますが、キャンセルのタイミングによってはキャンセル料金が適用される場合があります。コミットメント期間が経過すると、請求なしでキャパシティ予約を変更することやキャンセルすることはできます。

キャパシティ予約を使用できるのは、属性が一致するインスタンスのみです。デフォルトでは、キャパシティ予約は、新しいインスタンスと、一致する属性 (インスタンスタイプ、プラットフォーム、アベイラビリティーゾーン、テナンシー) を持つ実行中のインスタンスを自動的に一致させます。つまり、一致する属性を持つインスタンスがキャパシティ予約で自動的に実行されます。ただし、特定のワークロードに対して キャパシティ予約 を指定することもできます。これにより、リザーブドキャパシティで実行できるインスタンスを明示的に制御できます。インスタンスがキャパシティ予約またはキャパシティ予約リソースグループでのみ実行されるように指定することもできます。

> **Important**
>
> 将来の日付のキャパシティ予約は、増分インスタンスを起動してカバーできるようにするためのものであり、既存の実行中のインスタンスをカバーするものではありません。既存の実行中のインスタンスをカバーする必要がある場合、代わりにすぐに開始されるキャパシティ予約を使用してください。

インスタンスタイプ、プラットフォーム、アベイラビリティーゾーン、テナンシーなど、一致する属性を持つサポートされているすべての Amazon EC2 インスタンスは、キャパシティ予約で実行できます。Amazon EC2 インスタンスは、ユーザー (非マネージドインスタンス) またはユーザーに代わって AWS サービス (マネージドインスタンス) によって起動できます。これは、一致する属性を持つ実行中のインスタンスと自動的に一致するオープンキャパシティ予約に特に当てはまります。例えば、次のサービスによってユーザーに代わって起動されたマネージドインスタンスは、作成および管理するキャパシティ予約で実行できます。

- Amazon EC2 Auto Scaling
- Amazon ECS
- Amazon EKS
- Amazon EMR
- Amazon SageMaker AI
- AWS Batch
- AWS Elastic Beanstalk
- AWS ParallelCluster
- AWS Parallel Computing Service (AWS PCS)

内容

- Amazon EC2 キャパシティ予約の概念
- キャパシティ予約、リザーブドインスタンス、Savings Plans 間の違い
- サポートされているプラットフォーム
- クォータ
- 制限事項
- キャパシティ予約の料金と請求
- キャパシティ予約の作成
- キャパシティ予約の状態を表示する
- 既存のキャパシティ予約へのインスタンスの起動
- アクティブなキャパシティ予約を変更する
- インスタンスのキャパシティ予約設定を変更する
- キャパシティ予約間でキャパシティを移動する

- 既存のキャパシティ予約からキャパシティを分割
- キャパシティ予約のキャンセル
- プレイスメントグループにキャパシティ予約を使用する
- キャパシティ予約グループ
- ローカルゾーンでのキャパシティ予約
- Wavelength Zone 内のキャパシティ予約
- AWS Outposts でのキャパシティ予約
- 共有キャパシティ予約
- キャパシティー予約フリート
- CloudWatch メトリクスを使用してキャパシティ予約の使用状況をモニタリングする
- キャパシティ予約の低使用率をモニタリングする
- 将来の日付のキャパシティ予約に関する状態の変化をモニタリングする
- 中断可能なキャパシティ予約

Amazon EC2 キャパシティ予約の概念

キャパシティ予約には次の主な概念が適用されます。

トピック

- 開始日時
- 終了日時
- コミットメント期間
- 将来の日付のキャパシティ予約に対する評価
- キャパシティ予約の属性
- インスタンスの一致条件

開始日時

開始日および時刻は、キャパシティ予約が利用可能になるタイミングを定義します。キャパシティ予約はすぐに開始することもできますし、将来の日付から開始することもできます。

- キャパシティ予約をすぐに開始する場合、リザーブドキャパシティは作成後にすぐに使用でき、キャパシティ予約がアクティブ状態になるとすぐに請求が開始されます。期間コミットメントは必

要はありません。キャパシティ予約は、必要に応じていつでも変更を行いニーズを満たすことができ、いつでもキャンセルしてキャパシティを解放し、課金を停止できます。

- 将来の日付のキャパシティ予約を開始する場合は、キャパシティが必要なタイミングと、キャパシティの維持をコミットする期間を指定します。指定された将来の日付にキャパシティ予約が使用可能になり、キャパシティ予約がアクティブ状態になると請求が開始されます。コミットメント期間は、アカウントにキャパシティ予約がプロビジョニングされるとすぐに開始されます。この期間中、インスタンス数をコミットされたインスタンス数より少なくすることやコミットメント期間より前の終了日を選択することはできません。キャパシティ予約はキャンセルできますが、キャンセルのタイミングによってはキャンセル料金が適用される場合があります。詳細については、「キャパシティ予約のキャンセル」を参照してください。コミットメント期間が経過した後は、キャパシティ予約を変更すること、およびキャンセルしてリザーブドキャパシティを解放して課金を停止することができます。

終了日時

終了日付および時刻は、キャパシティ予約が終了してリザーブドキャパシティがアカウントから解放されるタイミングを定義します。キャパシティ予約は特定の日付および時刻に自動的に終了するように設定するか、手動でキャンセルするまで無期限にアクティブにしておくように設定することもできます。

キャパシティ予約が自動的に終了するように設定した場合、キャパシティ予約は指定した時刻から 1 時間以内に期限が切れます。例えば、5/31/2019, 13:30:55 を指定した場合、キャパシティ予約は 5/31/2019 の 13:30:55 と 14:30:55 の間に終了することが保証されます。

予約が終了すると、リザーブドキャパシティはアカウントから解放され、インスタンスをキャパシティ予約にターゲットできなくなります。リザーブドキャパシティーで実行されているインスタンスは、中断されずに引き続き実行されます。キャパシティ予約をターゲットにしているインスタンスが停止される場合、キャパシティ予約のターゲット設定を削除するか、別のキャパシティ予約をターゲットするように設定するまで再開できません。詳細については「インスタンスのキャパシティ予約設定を変更する」を参照してください。

コミットメント期間

コミットメント期間は、将来の日付のキャパシティ予約にのみ適用されます。

コミットメント期間は、アカウントがプロビジョニングされた後に将来の日付のキャパシティ予約がアクティブ状態になるようにコミットする最小期間です。将来の日付のキャパシティ予約はコミットメント期間よりも長く維持できますが、短くすることはできません。コミットメント期間中には、次の内容が適用されます。

- キャパシティ予約はコミットメント期間中にキャンセルできますが、キャンセル料金が適用される場合があります。詳細については、「キャンセル料金」を参照してください。
- コミットされたインスタンス数を下回る数に減らすことはできませんが、増やすことができます。
- コミットメント期間内の日付または時刻に自動的に終了するようにキャパシティ予約を設定することはできません。コミットメント期間中に終了日付および時刻を延長できます。

Amazon EC2 は指定したコミットメント期間を使用し、リクエストをサポートできるかどうかを評価します。最小コミットメント期間は、ほとんどのインスタンスタイプでは 14 日間ですが、U および X インスタンスでは 12 週間です。リクエストを評価する際、Amazon EC2 はより短いコミットメント期間をサポートできると判断する場合があります。その場合、Amazon EC2 は短いコミットメント期間に設定した将来の日付のキャパシティ予約をスケジュールします。つまり、最初にリクエストした期間より短い期間に、アカウントにキャパシティ予約を保持することをコミットすることになります。

将来の日付のキャパシティ予約に対する評価

将来の日付のキャパシティ予約をリクエストすると、Amazon EC2 はリクエストを評価し、キャパシティの可用性および指定したコミットメント期間に基づいてサポートできるかどうかを判断します。通常、評価は 5 日以内に完了します。Amazon EC2 はリクエストを評価する際、次のような複数の要因を考慮します。

- 予測されたキャパシティ供給
- コミットメント期間
- 開始日を基準にしてキャパシティ予約をリクエストするタイミング
- リクエストのサイズ

将来の日付のキャパシティ予約は、5 ~ 120 日前にリクエストできます。お客様のリクエストをより適切にサポートするために、少なくとも 56 日 (8 週間) 前にリクエストすることをお勧めします。最小コミットメント期間は、ほとんどのインスタンスタイプでは 14 日間ですが、U および X インスタンスでは 12 週間です。最小インスタンス数は 32 vCPU です。

キャパシティ予約は、リクエストの評価中に `assessing` 状態のままになります。

リクエストをサポートできる場合、キャパシティ予約は `scheduled` 状態になり、リクエストされた日付および時刻に配信がスケジュールされます。キャパシティ予約が `scheduled` 状態である間、合計インスタンス数は 0 のままになりす。スケジュールされたキャパシティ予約が `active` になり、リクエストされた日付に使用可能になります。

リクエストをサポートできない場合、キャパシティ予約は unsupported 状態になります。サポートされていないキャパシティ予約は配信されません。

将来の日付のキャパシティ予約は、assessing、scheduled、または active 状態のときにキャンセルできます。キャンセル料金が適用される場合があります。詳細については、「キャパシティ予約のキャンセル」を参照してください。

将来の日付のキャパシティ予約を作成するには、「将来の日付のキャパシティ予約の作成」を参照してください。

キャパシティ予約の属性

キャパシティ予約を作成するとき、次の属性を指定する必要があります。

- アベイラビリティーゾーン
- インスタンスタイプ
- プラットフォーム (オペレーティングシステムタイプ)
- テナンシー (default または dedicated)

これらの属性に一致するインスタンスのみが、キャパシティ予約で起動または実行できます。

インスタンスの一致条件

インスタンスの一致条件またはインスタンス資格により、キャパシティ予約で起動および実行できるインスタンスが決まります。キャパシティ予約には、次のいずれかの一致条件を含めることができます。

- オープン — キャパシティ予約は、一致する属性 (インスタンスタイプ、プラットフォーム、アベイラビリティーゾーン) を持つすべてのインスタンスを自動的に一致させます。一致する属性を持つ新規および既存のインスタンスは、追加の設定を指定せずにキャパシティ予約で自動的に実行します。
- ターゲット済み — キャパシティ予約は一致する属性 (インスタンスタイプ、プラットフォーム、アベイラビリティーゾーン) を持ち、そのキャパシティ予約を明示的にターゲットするインスタンスのみを受け入れます。インスタンスは、キャパシティ予約がリザーブドキャパシティで起動または実行されるように具体的にターゲットする必要があります。リザーブドキャパシティで実行できるインスタンスを明示的に制御でき、意図しないリザーブドキャパシティの使用を回避できます。

将来の日付のキャパシティ予約をリクエストするとき、ターゲットした一致条件のみを指定できます。キャパシティ予約によって配信されるキャパシティが、配信時に任意の実行中のインスタンスまたはリザーブドキャパシティに対して増分または追加されるようにします。キャパシティ予約がアカウントでアクティブになったら、必要に応じてインスタンスの一致条件を変更して開くことができます。ただし、一致するインスタンスはすべてキャパシティ予約で自動的に実行されることに注意してください。これにより、意図しないキャパシティ使用が発生し、リクエストされたインスタンス数全体に対して新しいインスタンスを起動できなくなる可能性があります。

キャパシティ予約、リザーブドインスタンス、Savings Plans 間の違い

以下の表では、キャパシティ予約、リザーブドインスタンス、Savings Plans 間の主な違いを示しています。

	Capacity Reservations	ゾーン リザーブドインスタンス	リージョン リザーブドインスタンス	Savings Plans
用語	即時使用のキャパシティ予約にはコミットメントは必要ありません。必要に応じて作成、変更、キャンセルすることができます。 将来の日付のキャパシティ予約を使用すると、アカウントでキャパシティを維持することをコミットするコミットメント期間を指定します。キャパシティ予約はコミットメント期間中にキャンセルできますが、キャンセル料金が	固定の 1 年または 3 年のコミットメントが必要です。		

	Capacity Reservations	ゾーン リザーブドインスタンス	リージョン リザーブドインスタンス	Savings Plans
	適用される場合があります。			
キャパシティーの利点	特定のアベイラビリティーゾーンで予約されるキャパシティー。		予約されたキャパシティーがありません。	
請求割引	請求割引がありません。	請求割引を提供します。		
インスタンスの制限	リージョンごとのオンデマンドインスタンス 制限が適用されます。	デフォルトは、アベイラビリティーゾーンごとに 20 です。制限の引き上げをリクエストできます。	デフォルトは、リージョンごとに 20 です。制限の引き上げをリクエストできます。	無制限。

† キャパシティー予約と、Savings Plans またはリージョンリでのザーブドインスタンスを組み合わせて、割引を受けることができます。

詳細については、以下を参照してください。

- Amazon EC2 のリザーブドインスタンスの概要
- Savings Plans ユーザーガイド

サポートされているプラットフォーム

自分のインスタンスに適合するキャパシティー予約を、適切なプラットフォームで作成する必要があります。キャパシティ予約は、以下の `platform` の値をサポートしています。

- Linux/UNIX
- Linux with SQL Server Standard
- Linux with SQL Server Web

- Linux with SQL Server Enterprise
- SUSE Linux
- Red Hat Enterprise Linux
- RHEL with SQL Server Standard
- RHEL with SQL Server Enterprise
- RHEL with SQL Server Web
- RHEL with HA
- RHEL with HA および SQL Server Standard
- RHEL with HA および SQL Server Enterprise
- Ubuntu Pro
- Windows
- SQL Server と Windows
- Windows with SQL Server Web
- Windows with SQL Server Standard
- Windows with SQL Server Enterprise

インスタンスが特定のキャパシティ予約で実行されるようにするには、キャパシティ予約のプラットフォームが、インスタンスの起動に使用される AMI のプラットフォームと一致している必要があります。Linux AMI の場合は、AMI プラットフォームが一般値の [Linux/UNIX] を使用しているか、[SUSE Linux] などのより具体的な値を使用しているかを確認することが重要です。

Console

AMI プラットフォームを確認するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [AMI] を選択してください。
3. AMI を選択します。
4. [詳細] タブで、[プラットフォームの詳細] の値をメモします。

AWS CLI

AMI プラットフォームを確認するには

describe-images コマンドを使用して PlatformDetails の値を確認します。

```
aws ec2 describe-images \
    --image-ids ami-0abcdef1234567890 \
    --query Images[*].PlatformDetails
```

以下は出力の例です。

```
[
    "Linux/UNIX"
]
```

PowerShell

AMI プラットフォームを確認するには

Get-EC2Image コマンドレットを使用して、PlatformDetails の値を確認します。

```
Get-EC2Image `
    -ImageId ami-0abcdef1234567890 | `
    Select PlatformDetails
```

以下は出力の例です。

```
PlatformDetails
---------------
Linux/UNIX
```

クォータ

キャパシティの予約が許可されているインスタンスの数は、アカウントのオンデマンドインスタンスのクォータに基づいています。クォータに到達しない限り、既に実行されているインスタンスの数を差し引いた任意の数のインスタンスのキャパシティを予約できます。

assessing、scheduled、pending、active、delayed 状態のキャパシティ予約は、オンデマンドインスタンスのクォータにカウントされます。

制限事項

キャパシティ予約 を作成する前に、次の制限と制約に注意してください。

- アクティブで未使用のキャパシティ予約 は、オンデマンドインスタンス の制限の対象としてカウントされます。
- AWS アカウント間でキャパシティ予約を譲渡することはできません ただし、キャパシティー予約を他の AWS アカウントと共有することはできます。詳細については「共有キャパシティ予約」を参照してください。
- ゾーンリザーブドインスタンスの請求割引はキャパシティには適用されません。
- クラスタープレイスメントグループでキャパシティ予約を作成できます。スプレッドおよびパーティションプレイスメントグループはサポートされません。
- キャパシティ予約は Dedicated Hosts と供に使用することはできません キャパシティ予約は専有インスタンスで使用できます。
- [Windows インスタンス] キャパシティ予約は Bring-Your-Own-License (BYOL) と供に使用することはできません。
- [Red Hat インスタンス] キャパシティ予約は Bring-Your-Own-License (BYOL) と共に使用できます。
- キャパシティ予約では、休止状態のインスタンスを再開した場合でも、それが元の状態に復帰することを保証していません。
- 最小 32 vCPU のインスタンス数に対し、将来の日付のキャパシティ予約をリクエストできます。例えば、`m5.xlarge` インスタンスに将来の日付のキャパシティ予約をリクエストした場合、少なくとも 8 個のインスタンス (8 * m5.xlarge = 32 vCPU) をリクエストする必要があります。
- 将来の日付のキャパシティ予約は、C、G、I、M、R、T、U、X のファミリーのインスタンスタイプにリクエストできます。

キャパシティ予約の料金と請求

このセクションのトピックでは、キャパシティ予約の料金と請求の概要を説明します。

トピック

- 料金
- 請求
- 請求割引
- コストと使用状況レポート 2.0 に表示されるキャンセル料金
- 請求の表示

料金

インスタンスをリザーブドキャパシティで実行しているかどうかにかかわらず、キャパシティ予約にはオンデマンドと同等の料金 (ハードウェア専有インスタンスの地域別追加料金が適用される場合はこの料金を含む) が請求されます。予約を使用しない場合、この予約は Amazon EC2 請求書に未使用予約として記載されます。予約の属性に一致するインスタンスを実行するときは、そのインスタンスの料金のみを支払い、予約に料金はかかりません。前払い、または追加の料金はありません。

例えば、20 個の `m4.large` Linux インスタンスに対してキャパシティ予約を作成し、同じアベイラビリティーゾーンで 15 個の `m4.large` Linux インスタンスを実行すると、15 個のアクティブインスタンスと予約されている 5 個の未使用のインスタンス分が課金されます。

Savings Plans とリージョンのリザーブドインスタンスの請求割引がキャパシティー予約に適用されます。詳細については、請求割引を参照してください。

詳細については、Amazon EC2 の料金表を参照してください。

請求

アカウントでキャパシティ予約がプロビジョニングされるとすぐに課金が開始されます。以後、アカウントでキャパシティ予約がプロビジョニングされている間、継続して課金が発生します。将来の日付のキャパシティ予約の場合、リクエストされた将来の日付にキャパシティ予約がアカウントにプロビジョニングされたときにのみ請求が開始されることを意味します。

キャパシティ予約は、秒単位で課金されます。つまり、1 時間に満たない分に対して課金されます。例えば、24 時間 15 分の間、アカウント内でキャパシティ予約がプロビジョニングされたままである場合は、`24.25` 予約時間が課金されます。

次の例は、キャパシティ予約の請求方法を示しています。キャパシティ予約は 1 つの `m4.large` Linux インスタンスに対して作成され、オンデマンド料金は 1 時間あたり 0.10 USD です。この例では、キャパシティ予約はアカウントで 5 時間プロビジョニングされます。キャパシティ予約は最初の 1 時間は使用されないため、`m4.large` インスタンスタイプの標準オンデマンド料金で未使用の 1 時間分の料金が請求されます。2~5 時間目は、キャパシティ予約は `m4.large` インスタンスによって占有されます。この間、キャパシティ予約に料金は発生せず、代わりにそれを占有している `m4.large` インスタンスに対してアカウントが請求されます。6 時間目にはキャパシティ予約がキャンセルされ、`m4.large` インスタンスはリザーブドキャパシティー外で通常どおりに実行されます。その時間は、`m4.large` インスタンスタイプのオンデマンド料金で請求されます。

Hour	1	2	3	4	5	6	Total cost
Unused Capacity Reservation	$0.10	$0.00	$0.00	$0.00	$0.00	$0.00	**$0.10**
On-demand Instance Usage	$0.00	$0.10	$0.10	$0.10	$0.10	$0.10	**$0.50**
Hourly cost	**$0.10**	**$0.10**	**$0.10**	**$0.10**	**$0.10**	**$0.10**	**$0.60**

請求割引

Savings Plans とリージョンのリザーブドインスタンスの請求割引がキャパシティー予約に適用されます。AWS は、属性が一致するキャパシティー予約に対しこれらの割引を自動的に適用します。キャパシティ予約がインスタンスによって使用されると、割引がインスタンスに適用されます。割引は、未使用のキャパシティ予約を対象とする前に、インスタンスの使用に優先的に適用されます。

ゾーンリザーブドインスタンスの請求割引はキャパシティ予約には適用されません。

詳細については、以下を参照してください。

- Amazon EC2 のリザーブドインスタンスの概要
- Savings Plans ユーザーガイド
- 請求と購入のオプション

コストと使用状況レポート 2.0 に表示されるキャンセル料金

将来の日付のキャパシティ予約をキャンセルし、キャンセル料金が適用される場合、料金は AWS コストと使用状況レポート (CUR) 2.0 に表示されます。CUR 2.0 のキャパシティ予約列の詳細については、「Capacity reservation columns」を参照してください。次の表は、キャパシティ予約の明細項目がキャンセル前とキャンセル後にどのように表示されるかを示しています。

アクティブな予約の CUR 2.0 値

capacity_reservation_capacity_reservation_status	line_item_usage_type	表示
`Reserved`	予約する	リザーブドキャパシティの合計
`Used`	BoxUsage	起動したインスタンス
`Unused`	UnusedBox	アイドル状態の容量

キャンセル後の CUR 2.0 の値

capacity_reservation_capacity_reservation_status	line_item_usage_type	表示
Cancelling	予約する	キャンセルされた容量
Cancelling	UnusedBox	キャンセル料金

請求の表示

アカウントの請求と料金は、AWS Billing and Cost Management コンソールで確認できます。

- [ダッシュボード] には、アカウント利用料の概要が表示されます。
- [請求書] ページの [明細] で、[Elastic Compute Cloud] セクションとリージョンを展開して、キャパシティ予約の請求情報を取得します。

請求額をオンラインで表示することも、CSV ファイルとしてダウンロードすることもできます。詳細については、 キャパシティ予約 明細項目を参照してください。

キャパシティ予約の作成

いつでもキャパシティ予約を作成し、特定のアベイラビリティーゾーンで利用可能なコンピューティングキャパシティを確保できます。キャパシティ予約はすぐに開始することもできますし、将来の日付から開始することもできます。キャパシティ予約が `active` 状態になった後にのみ、キャパシティを利用できます。

> **Note**
>
> open インスタンスの一致条件を使用してキャパシティ予約を作成し、キャパシティ予約がアクティブになった時点で一致する属性を持つインスタンスを実行している場合、それらのインスタンスはリザーブドキャパシティで自動的に実行されます。これを回避するには、`targeted` インスタンスの一致条件を使用します。詳細については「インスタンスの一致条件」を参照してください。

次のいずれかが当てはまる場合、キャパシティ予約を作成するリクエストは失敗する可能性があります。

- Amazon EC2 には、リクエストに対応する十分なキャパシティ - がありません。時間をおいてからもう一度試すか、別のアベイラビリティーゾーンを試すか、リクエストを小さくしてみてください。インスタンスタイプとサイズに応じてアプリケーションに柔軟性がある場合は、別のインスタンス属性を試してみてください。
- リクエストされた数量は、選択したインスタンスファミリーに対するオンデマンドインスタンスの上限を超えています。インスタンスファミリーに対するオンデマンドインスタンスの上限を上げて、もう一度試してください。詳細については「オンデマンドインスタンスクォータ」を参照してください。

トピック

- 即時使用のキャパシティ予約の作成
- 将来の日付のキャパシティ予約の作成

即時使用のキャパシティ予約の作成

即時使用のキャパシティ予約を作成します。

Console

キャパシティ予約を作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. [キャパシティ予約]、[作成キャパシティ予約] の順に選択してください。
3. [インスタンスの詳細] セクションで次のように設定します。
   a. [インスタンスタイプ] — キャパシティを予約するインスタンスタイプ。
   b. [プラットフォーム] — インスタンスのオペレーティングシステム。詳細については「サポートされているプラットフォーム」を参照してください。
   c. [アベイラビリティーゾーン] — キャパシティーを予約するアベイラビリティーゾーン。
   d. [テナンシー] — リザーブドキャパシティに使用するテナンシーのタイプ。共有ハードウェアにキャパシティを予約する場合は [デフォルト] を選択するか、アカウント専用のハードウェアのキャパシティを予約する場合は [専用] を選択してください。
   e. (オプション) [プレイスメントグループ ARN] — キャパシティ予約を作成するクラスタープレイスメントグループの ARN。詳細については「プレイスメントグループにキャパシティ予約を使用する」を参照してください。

f. [合計インスタンス数] — キャパシティを予約するインスタンスの数。選択したインスタンスタイプに残りのオンデマンドインスタンスのクオータを超える数量を指定した場合、リクエストは失敗します。

4. [Reservation details (予約の詳細)] セクションで次のように設定します。

   a. [キャパシティ予約の開始] — [すぐに]選択してください。

   b. [キャパシティ予約の終了] — 次のいずれかのオプションを選択してください。

      - [手動] – 明示的にキャンセルするまでキャパシティを予約できます。
      - [特定の時間] — 指定された日付および時刻にキャパシティ予約を自動的にキャンセルします。

   c. [インスタンス資格] — 次のいずれかのオプションを選択してください。

      - [オープン] – (デフォルト) キャパシティ予約は、一致する属性 (インスタンスタイプ、プラットフォーム、アベイラビリティーゾーン、テナンシー) を持つすべてのインスタンスに一致します。一致する属性を持つインスタンスを起動すると、そのインスタンスはリザーブドキャパシティーに自動的に配置されます。
      - [ターゲット済み] – キャパシティ予約は、一致する属性 (インスタンスタイプ、プラットフォーム、アベイラビリティーゾーン、テナンシー) を持って明示的に予約をターゲットするインスタンスのみを受け入れます。

5. [作成] を選択します。

AWS CLI

キャパシティ予約を作成するには

create-capacity-reservation コマンドを使用します。

```
aws ec2 create-capacity-reservation \
    --availability-zone az_name \
    --instance-type instance_type \
    --instance-count number_of_instances \
    --instance-platform operating_system \
    --instance-match-criteria open|targeted
```

PowerShell

キャパシティ予約を作成するには

Add-EC2CapacityReservation コマンドレットを使用します。

```
Add-EC2CapacityReservation `
    -AvailabilityZone az_name `
    -InstanceType instance_type `
    -InstanceCount number_of_instances `
    -InstancePlatform operating_system `
    -InstanceMatchCriterion open|targeted
```

将来の日付のキャパシティ予約の作成

将来の日付および時刻にリザーブドキャパシティを利用できるようにする必要がある場合、将来の日付のキャパシティ予約をリクエストします。

将来の日付のキャパシティ予約をリクエストすると、リクエストは評価を受けてサポート可能かどうかを判断します。詳細については「将来の日付のキャパシティ予約に対する評価」を参照してください。

考慮事項

- 将来の日付のキャパシティ予約は、C、G、I、M、R、T、U、X のファミリーのインスタンスタイプにリクエストできます。
- 最小 32 vCPU のインスタンス数に対し、将来の日付のキャパシティ予約をリクエストできます。例えば、m5.xlarge インスタンスに将来の日付のキャパシティ予約をリクエストした場合、少なくとも 8 個のインスタンス (8 * m5.xlarge = 32 vCPU) にキャパシティをリクエストする必要があります。
- 将来の日付のキャパシティ予約は、5 ~ 120 日前にリクエストできます。サポート性を向上させるには、少なくとも 56 日 (8 週間) 前にリクエストすることをお勧めします。
- C、G、I、M、R、T インスタンスの最小コミットメント期間は 14 日間です。U および X インスタンスの場合、最小コミットメント期間は 12 週間です。

Console

キャパシティ予約を作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. [キャパシティ予約]、[作成キャパシティ予約] の順に選択してください。

3. [インスタンスの詳細] セクションで次のように設定します。

   a. [インスタンスタイプ] — キャパシティを予約するインスタンスタイプ。

   b. [プラットフォーム] — インスタンスのオペレーティングシステム。詳細については「サポートされているプラットフォーム」を参照してください。

   c. [アベイラビリティーゾーン] — キャパシティーを予約するアベイラビリティーゾーン。

   d. [テナンシー] — リザーブドキャパシティに使用するテナンシーのタイプ。共有ハードウェアにキャパシティを予約する場合は [デフォルト] を選択するか、アカウント専用のハードウェアのキャパシティを予約する場合は [専用] を選択してください。

   e. [合計インスタンス数] — キャパシティを予約するインスタンスの数。選択したインスタンスタイプに残りのオンデマンドインスタンスのクオータを超える数量を指定した場合、リクエストは失敗します。

4. [Reservation details (予約の詳細)] セクションで次のように設定します。

   a. [キャパシティ予約の開始] — [特定の時刻] を選択してください。

   b. [開始日] – キャパシティ予約を使用可能にする必要がある日付および時刻を指定します。詳細については「開始日時」を参照してください。

   c. [コミットメント期間] — キャパシティ予約が配信された後に維持することをコミットする最小期間を指定します。詳細については「コミットメント期間」を参照してください。

   d. [キャパシティ予約の終了] — 次のいずれかのオプションを選択してください。

      - [キャンセルするタイミング] — 明示的にキャンセルするまでキャパシティを予約します。
      - [特定の時間] — 指定された日付および時刻にキャパシティ予約を自動的にキャンセルします。

5. [作成] を選択します。

AWS CLI

キャパシティ予約を作成するには

create-capacity-reservation コマンドを使用します。

```
aws ec2 create-capacity-reservation \
    --availability-zone az_name \
    --instance-type instance_type \
```

```
    --instance-count number_of_instances \
    --instance-platform operating_system \
    --instance-match-criteria targeted \
    --delivery-preference incremental \
    --commitment-duration commitment_in_seconds \
    --start-date YYYY-MMDDThh:mm:ss.sssZ
```

PowerShell

キャパシティ予約を作成するには

Add-EC2CapacityReservation コマンドレットを使用します。

```
Add-EC2CapacityReservation `
    -AvailabilityZone az_name `
    -InstanceType instance_type `
    -InstanceCount number_of_instances `
    -InstancePlatform operating_system `
    -InstanceMatchCriterion targeted `
    -DeliveryPreference incremental `
    -CommitmentDuration commitment_in_seconds `
    -StartDate  YYYY-MMDDThh:mm:ss.sssZ
```

キャパシティ予約の状態を表示する

Amazon EC2 は、キャパシティ予約の状態を常にモニタリングしています。

最終的な整合性モデルとそれに続く Amazon EC2 API により、キャパシティ予約を作成した後、キャパシティ予約の状態が `active` であることが反映されるまでに最大 5 分かかる場合があります。この間、キャパシティ予約が `pending` 状態のままになる可能性があります。ただし、キャパシティ予約が既に使用可能になっている可能性があり、その場合は、キャパシティ予約でのインスタンスの起動の試みが成功します。

Console

キャパシティ予約を表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[キャパシティ予約] を選択してください。
3. キャパシティ予約を選択します。

AWS CLI

キャパシティ予約を記述するには

describe-capacity-reservations コマンドを使用します。

例えば、次のコマンドは、すべてのキャパシティ予約について説明します。

```
aws ec2 describe-capacity-reservations
```

以下は出力の例です。

```
{
    "CapacityReservations": [
        {
            "CapacityReservationId": "cr-1234abcd56EXAMPLE",
            "EndDateType": "unlimited",
            "AvailabilityZone": "eu-west-1a",
            "InstanceMatchCriteria": "open",
            "Tags": [],
            "EphemeralStorage": false,
            "CreateDate": "2019-08-16T09:03:18.000Z",
            "AvailableInstanceCount": 1,
            "InstancePlatform": "Linux/UNIX",
            "TotalInstanceCount": 1,
            "State": "active",
            "Tenancy": "default",
            "EbsOptimized": true,
            "InstanceType": "a1.medium",
            "PlacementGroupArn": "arn:aws:ec2:us-east-1:123456789012:placement-
group/MyPG"
        },
        {
            "CapacityReservationId": "cr-abcdEXAMPLE9876ef",
            "EndDateType": "unlimited",
            "AvailabilityZone": "eu-west-1a",
            "InstanceMatchCriteria": "open",
            "Tags": [],
            "EphemeralStorage": false,
            "CreateDate": "2019-08-07T11:34:19.000Z",
            "AvailableInstanceCount": 3,
            "InstancePlatform": "Linux/UNIX",
            "TotalInstanceCount": 3,
```

```
            "State": "cancelled",
            "Tenancy": "default",
            "EbsOptimized": true,
            "InstanceType": "m5.large"
        }
    ]
}
```

PowerShell

キャパシティ予約を記述するには

Get-EC2CapacityReservation コマンドレットを使用します。

```
Get-EC2CapacityReservation `
    -CapacityReservationId cr-1234abcd56EXAMPLE
```

キャパシティ予約の状態

キャパシティ予約には次の状態があります。

状態	説明
active	キャパシティ - を使用できます。
expired	キャパシティ予約は、予約リクエストで指定された日付と時刻に自動的に有効期限が切れました。リザーブドキャパシティーも使用できなくなります。
cancelled	キャパシティ予約はキャンセルされました。リザーブドキャパシティーも使用できなくなります。
pending	キャパシティ予約リクエストは正常に処理されましたが、キャパシティのプロビジョニングはまだ保留中です。
failed	キャパシティ予約リクエストは失敗しました。有効でないリクエストパラメータ、キャパシティー制約、またはインスタンス制限の制約のため、リクエストが失敗する可能性があります。失敗したリクエストを 60 分間表示できます。

状態	説明
scheduled	(将来の日付のキャパシティ予約のみ) 将来の日付のキャパシティ予約リクエストが承認され、キャパシティ予約はリクエストされた開始日に配信される予定です。
assessing	(将来の日付のキャパシティ予約のみ) Amazon EC2 は将来の日付のキャパシティ予約におけるリクエストを評価しています。詳細については「将来の日付のキャパシティ予約に対する評価」を参照してください。
delayed	(将来の日付のキャパシティ予約のみ) Amazon EC2 では、リクエストされた将来の日付のキャパシティ予約におけるプロビジョニングに遅延が発生しました。Amazon EC2 は、リクエストされた開始日付と時刻までにリクエストされたキャパシティを配信できません。
unsupported	(将来の日付のキャパシティ予約のみ) キャパシティの制約により、Amazon EC2 は将来の日付のキャパシティ予約リクエストをサポートできません。サポートされていないリクエストは 30 日間表示できます。キャパシティ予約は配信されません。
cancelling	(将来の日付のキャパシティ予約のみ) キャパシティ予約のキャンセル処理が行われています。キャパシティは解放されましたが、料金は削減されたコミットメント期間の終了まで継続されます。キャパシティ予約は、コミットメントが終了すると cancelled に移行します。詳細については、「キャンセル料金」を参照してください。

既存のキャパシティ予約へのインスタンスの起動

キャパシティ予約でインスタンスを起動できるのは、次の場合のみです。

- 一致する属性 (インスタンスタイプ、プラットフォーム、アベイラビリティーゾーン、テナンシー) があります
- 十分なキャパシティがあります
- active 状態です

インスタンスを起動するときに、インスタンスを任意の open キャパシティ予約 に起動するか、特定のキャパシティ予約に起動するか、またはキャパシティ予約のグループに起動するかを指定できます。

または、一致する属性と使用可能な容量を持つ open キャパシティ予約がある場合でも、キャパシティーの予約 でインスタンスを実行しないように設定できます。

キャパシティ予約にインスタンスを起動すると、起動されたインスタンスの数だけ使用可能なキャパシティーが減少します。例えば、3 つのインスタンスを起動すると、キャパシティ予約の使用可能なキャパシティは 3 つ減少します。

Console

既存のキャパシティ予約でインスタンスを起動するには

1. 手順に従ってインスタンスを起動しますが、次のステップを完了してプレイスメントグループとキャパシティ予約の設定を指定するまでインスタンスを起動しないでください。
2. [高度な詳細] を展開し、以下の操作を行います。

   a. [プレイスメントグループ] で、インスタンスを起動するクラスタープレイスメントグループを選択してください。

   b. [Capacity Reservation] (キャパシティ予約) で、キャパシティ予約の設定に応じて、次のいずれかのオプションを選択してください。

      - [なし] — インスタンスがキャパシティ予約に起動しないようにします。インスタンスはオンデマンド型キャパシティーで実行されます。
      - [開く] — 選択したインスタンスの数に対して一致する属性と十分なキャパシティのあるキャパシティ予約にインスタンスを起動します。十分なキャパシティを持つ、一致するキャパシティ予約がない場合は、インスタンスはオンデマンドのキャパシティーを使用します。
      - [キャパシティ予約の指定] — 選択したキャパシティ予約にインスタンスを起動します。選択されたこの キャパシティ予約に選択したインスタンスの数に対して十分なキャパシティがない場合、インスタンスの起動に失敗します。
      - [キャパシティ予約リソースグループの指定] — 選択したキャパシティ予約グループで一致する属性および利用可能なキャパシティを持つ任意のキャパシティ予約にインスタンスを起動します。選択したグループに、一致する属性と使用可能な容量を持つキャパシティ予約がない場合、インスタンスはオンデマンド型キャパシティに起動します。

- [キャパシティ予約の指定] — キャパシティ予約にインスタンスを起動します。キャパシティ予約 ID が指定されていない場合、インスタンスはオープンキャパシティ予約で起動します。キャパシティがない場合、インスタンスは起動に失敗します。
- [キャパシティ予約リソースグループのみを指定] – キャパシティ予約リソースグループのキャパシティ予約でインスタンスを起動します。キャパシティ予約リソースグループ ARN が指定されていない場合、インスタンスはオープンキャパシティ予約で起動します。キャパシティがない場合、インスタンスは起動に失敗します。

3. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択してください。

AWS CLI

既存のキャパシティ予約でインスタンスを起動するには

run-instances コマンドを使用して `--capacity-reservation-specification` オプションを指定します。

次の例では、属性と使用可能なキャパシティが一致する任意の開いているキャパシティ予約でインスタンスを起動します。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --count 1 \
    --instance-type t2.micro \
    --key-name my-key-pair \
    --subnet-id subnet-0abcdef1234567890 \
    --capacity-reservation-specification CapacityReservationPreference=open
```

次の例では、インスタンスを `targeted` のキャパシティ予約に起動します。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --count 1 \
    --instance-type t2.micro \
    --key-name my-key-pair \
    --subnet-id subnet-0abcdef1234567890 \
    --capacity-reservation-specification \
        CapacityReservationTarget={CapacityReservationId=cr-1234abcd56EXAMPLE}
```

次の例では、インスタンスを指定されたキャパシティ予約グループに起動します。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --count 1 \
    --instance-type t2.micro \
    --key-name my-key-pair \
    --subnet-id subnet-0abcdef1234567890 \
    --capacity-reservation-specification \

 CapacityReservationTarget={CapacityReservationResourceGroupArn=arn:aws:resource-
groups:us-west-2:123456789012:group/my-cr-group}
```

次の例では、インスタンスを 1 つのキャパシティ予約にのみ起動します。キャパシティ予約 ID が指定されていないため、インスタンスは、一致する属性および使用可能なキャパシティを持つ任意のオープンキャパシティ予約で起動します。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --count 1 \
    --instance-type t2.micro \
    --key-name my-key-pair \
    --subnet-id subnet-0abcdef1234567890 \
    --capacity-reservation-specification \
        CapacityReservationPreference=capacity-reservations-only
```

次の例では、インスタンスを特定のキャパシティ予約のみで起動します。指定されたキャパシティ予約でキャパシティがない場合、インスタンスは起動に失敗します。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --count 1 \
    --instance-type t2.micro \
    --key-name my-key-pair \
    --subnet-id subnet-0abcdef1234567890 \
    --capacity-reservation-specification \
        CapacityReservationPreference=capacity-reservations-only \
        CapacityReservationTarget={CapacityReservationId=cr-1234abcd56EXAMPLE}
```

PowerShell

既存のキャパシティ予約でインスタンスを起動するには

New-EC2Instance コマンドレットを使用します。

次の例では、属性と使用可能なキャパシティが一致する任意の開いているキャパシティ予約でインスタンスを起動します。

```
New-EC2Instance `
    -ImageId ami-0abcdef1234567890 `
    -InstanceType t2.micro `
    -KeyName "my-key-pair" `
    -SubnetId subnet-0abcdef1234567890 `
    -CapacityReservationSpecification_CapacityReservationPreference "open"
```

次の例では、インスタンスを targeted のキャパシティ予約に起動します。

```
New-EC2Instance `
    -ImageId ami-0abcdef1234567890 `
    -InstanceType t2.micro `
    -KeyName "my-key-pair" `
    -SubnetId subnet-0abcdef1234567890 `
    -CapacityReservationTarget_CapacityReservationId cr-1234abcd56EXAMPLE
```

次の例では、インスタンスを指定されたキャパシティ予約グループに起動します。

```
New-EC2Instance `
    -ImageId ami-0abcdef1234567890 `
    -InstanceType t2.micro `
    -KeyName "my-key-pair" `
    -SubnetId subnet-0abcdef1234567890 `
    -CapacityReservationTarget_CapacityReservationResourceGroupArn `
        "arn:aws:resource-groups:us-west-2:123456789012:group/my-cr-group"
```

次の例では、インスタンスを 1 つのキャパシティ予約にのみ起動します。キャパシティ予約 ID が指定されていないため、インスタンスは、一致する属性および使用可能なキャパシティを持つ任意のオープンキャパシティ予約で起動します。

```
New-EC2Instance `
    -ImageId ami-0abcdef1234567890 `
    -InstanceType t2.micro `
    -KeyName "my-key-pair" `
    -SubnetId subnet-0abcdef1234567890 `
    -CapacityReservationSpecification_CapacityReservationPreference "capacity-
reservations-only"
```

次の例では、インスタンスを特定のキャパシティ予約のみで起動します。指定されたキャパシティ予約でキャパシティがない場合、インスタンスは起動に失敗します。

```
New-EC2Instance `
    -ImageId ami-0abcdef1234567890 `
    -InstanceType t2.micro `
    -KeyName "my-key-pair" `
    -SubnetId subnet-0abcdef1234567890 `
    -CapacityReservationSpecification_CapacityReservationPreference "capacity-
reservations-only" `
    -CapacityReservationTarget_CapacityReservationId cr-1234abcd56EXAMPLE
```

アクティブなキャパシティ予約を変更する

キャパシティを必要とするワークロードに適さない既存のキャパシティ予約がある場合は、インスタンス数、インスタンスの適格性 (open または targeted)、終了時間 (At specific time または Manually) を変更できます。選択したインスタンスタイプの残りのオンデマンドインスタンス制限を超える新しいインスタンス数を指定すると、その更新は失敗します。

許可される変更は、キャパシティ予約の状態によって異なります。

- assessing または scheduled 状態 — タグのみ変更できます。
- pending 状態 — キャパシティ予約はいかなる方法で変更できません。
- active 状態であってもコミットメント期間内 – インスタンス数をコミットされたインスタンス数より少なくしたり、コミットされた期間より前の終了日を設定したりすることはできません。その他の変更はすべて許可されます。
- コミットメント期間または経過したコミットメント期間がない active 状態 — すべての変更が許可されます。
- expired、cancelled、unsupported、failed 状態 — キャパシティ予約はいかなる方法で変更できません。

考慮事項

- 作成後にインスタンスタイプ、プラットフォーム、アベイラビリティーゾーン、テナンシーを変更することはできません。これらの属性を変更する必要がある場合は、予約をキャンセルし、必要な属性を持つ新しいものを作成することをお勧めします。

- インスタンスの適格性を targeted から open に変更することで既存のキャパシティ予約を変更した場合、キャパシティ予約の属性に一致し、CapacityReservationPreference パラメータが open に設定されており、かつキャパシティ予約でまだ実行されていない実行中のインスタンスは、変更されたキャパシティ予約を自動的に使用します。
- インスタンスの適格性を変更するには、インスタンスが予約内で実行されている場合、Amazon EC2 はインスタンスの適格性を変更できないため、キャパシティ予約が完全にアイドル状態 (使用量ゼロ) である必要があります。

Console

キャパシティ予約を変更するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. [キャパシティ予約] を選択し、キャパシティ予約を選択して、次に [Edit (編集)] を選択してください。
3. 必要に応じて [合計キャパシティ]、[キャパシティ予約の終了]、または [インスタンスの適格性] オプションを変更し、[保存] を選択してください。

AWS CLI

キャパシティ予約を変更するには

modify-capacity-reservation コマンドを使用します。次の例では、8 つのインスタンスのキャパシティを予約するために、指定したキャパシティ予約を変更します。

```
aws ec2 modify-capacity-reservation \
    --capacity-reservation-id cr-1234567890abcdef0 \
    --instance-count 8
```

PowerShell

キャパシティ予約を変更するには

Edit-EC2CapacityReservation コマンドレットを使用します。次の例では、8 つのインスタンスのキャパシティを予約するために、指定したキャパシティ予約を変更します。

```
Edit-EC2CapacityReservation `
    -CapacityReservationId cr-1234567890abcdef0 `
```

```
    -InstanceCount 8
```

インスタンスのキャパシティ予約設定を変更する

停止したインスタンスの次のキャパシティ予約設定は、いつでも変更できます。

- 一致する属性 (インスタンスタイプ、プラットフォーム、アベイラビリティーゾーン、テナンシー) と使用可能なキャパシティを持つ任意のキャパシティ予約で起動します。
- 特定のキャパシティ予約でインスタンスを起動します。
- キャパシティー予約グループ内で、属性が一致し、キャパシティーが使用可能な、いずれかのキャパシティー予約を起動します。
- インスタンスがキャパシティ予約で起動しないようにします。

Console

インスタンスのキャパシティ予約設定を変更するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. [インスタンス] を選択し、変更するインスタンスを選択してください。インスタンスをまだ停止していない場合は、停止します。
3. [アクション]、[インスタンス設定]、[キャパシティ予約の変更] の順に選択してください。
4. [キャパシティ予約] で、以下のいずれかのオプションを選択してください。

   - [開く] — 選択したインスタンスの数に対して一致する属性と十分なキャパシティのあるキャパシティ予約にインスタンスを起動します。十分なキャパシティを持つ、一致するキャパシティ予約がない場合は、インスタンスはオンデマンドのキャパシティーを使用します。
   - [なし] — インスタンスがキャパシティ予約に起動しないようにします。インスタンスはオンデマンド型キャパシティーで実行されます。
   - [キャパシティ予約の指定] — 選択したキャパシティ予約にインスタンスを起動します。選択されたこの キャパシティ予約に選択したインスタンスの数に対して十分なキャパシティがない場合、インスタンスの起動に失敗します。
   - [キャパシティ予約グループの指定] — 選択したキャパシティ予約グループで一致する属性および利用可能なキャパシティを持つ任意のキャパシティ予約でインスタンスを起動します。選択したグループに、一致する属性と使用可能な容量を持つキャパシティ予約がない場合、インスタンスはオンデマンド型キャパシティに起動します。

- [キャパシティ予約の指定] — キャパシティ予約にインスタンスを起動します。キャパシティ予約 ID が指定されていない場合、インスタンスはオープンキャパシティ予約で起動します。キャパシティがない場合、インスタンスは起動に失敗します。
- [キャパシティ予約リソースグループのみを指定] – キャパシティ予約リソースグループのキャパシティ予約でインスタンスを起動します。キャパシティ予約リソースグループ ARN が指定されていない場合、インスタンスはオープンキャパシティ予約で起動します。キャパシティがない場合、インスタンスは起動に失敗します。

AWS CLI

インスタンスのキャパシティ予約設定を変更するには

modify-instance-capacity-reservation-attributes コマンドを使用します。

次の例では、キャパシティ予約設定を none に変更します。

```
aws ec2 modify-instance-capacity-reservation-attributes \
    --instance-id i-1234567890abcdef0 \
    --capacity-reservation-specification CapacityReservationPreference=none
```

次の例では、特定のキャパシティ予約をターゲットにします。

```
aws ec2 modify-instance-capacity-reservation-attributes \
    --instance-id i-1234567890abcdef0 \
    --capacity-reservation-specification \
    CapacityReservationTarget={CapacityReservationId=cr-1234567890abcdef0}
```

次の例では、ターゲットを特定のキャパシティ予約グループに変更します。

```
aws ec2 modify-instance-capacity-reservation-attributes \
    --instance-id i-1234567890abcdef0 \
    --capacity-reservation-specification \

 CapacityReservationTarget={CapacityReservationResourceGroupArn=arn:aws:resource-
groups:us-west-2:123456789012:group/my-cr-group}
```

次の例では、キャパシティ予約設定を capacity-reservation-only に変更します。キャパシティ予約が指定されていないため、インスタンスは、一致する属性および使用可能なキャパシティを持つ任意のオープンキャパシティ予約で起動します。

```
aws ec2 modify-instance-capacity-reservation-attributes \
    --instance-id i-1234567890abcdef0 \
    --capacity-reservation-specification CapacityReservationPreference=capacity-
reservation-only
```

次の例では、キャパシティ予約設定を capacity-reservation-only に変更し、ターゲットを特定のキャパシティ予約に変更します。指定されたキャパシティ予約でキャパシティがない場合、インスタンスは起動に失敗します。

```
aws ec2 modify-instance-capacity-reservation-attributes \
    --instance-id i-1234567890abcdef0 \
    --capacity-reservation-specification \
        CapacityReservationPreference=capacity-reservation-only \
        CapacityReservationTarget={CapacityReservationId=cr-1234567890abcdef0}
```

PowerShell

インスタンスのキャパシティ予約設定を変更するには

Edit-EC2InstanceCapacityReservationAttribute コマンドレットを使用します。

次の例では、キャパシティ予約設定を none に変更します。

```
Edit-EC2InstanceCapacityReservationAttribute `
    -InstanceId i-1234567890abcdef0 `
    -CapacityReservationSpecification_CapacityReservationPreference "none"
```

次の例では、特定のキャパシティ予約をターゲットにします。

```
Edit-EC2InstanceCapacityReservationAttribute `
    -InstanceId i-1234567890abcdef0 `
    -CapacityReservationTarget_CapacityReservationId cr-1234567890abcdef0
```

次の例では、ターゲットを特定のキャパシティ予約グループに変更します。

```
Edit-EC2InstanceCapacityReservationAttribute `
    -InstanceId i-1234567890abcdef0 `
    -CapacityReservationTarget_CapacityReservationResourceGroupArn `
        "arn:aws:resource-groups:us-west-2:123456789012:group/my-cr-group"
```

次の例では、キャパシティ予約設定を capacity-reservation-only に変更します。キャパシティ予約が指定されていないため、インスタンスは、一致する属性および使用可能なキャパシティを持つ任意のオープンキャパシティ予約で起動します。

```
Edit-EC2InstanceCapacityReservationAttribute `
    -InstanceId i-1234567890abcdef0 `
    -CapacityReservationSpecification_CapacityReservationPreference "capacity-
reservation-only"
```

次の例では、キャパシティ予約設定を capacity-reservation-only に変更し、ターゲットを特定のキャパシティ予約に変更します。指定されたキャパシティ予約でキャパシティがない場合、インスタンスは起動に失敗します。

```
Edit-EC2InstanceCapacityReservationAttribute `
    -InstanceId i-1234567890abcdef0 `
    -CapacityReservationSpecification_CapacityReservationPreference "capacity-
reservation-only" `
    -CapacityReservationTarget_CapacityReservationId cr-1234567890abcdef0
```

キャパシティ予約間でキャパシティを移動する

キャパシティをキャパシティ予約間で移動し、必要に応じてリザーブドコンピューティングリソースを再配分できます。例えば、使用量が増加している予約に追加のキャパシティが必要で、別の予約に利用可能なキャパシティがある場合、2 つの予約間でキャパシティを再割り当てできます。

キャパシティを移動するための前提条件

前提条件として、2 つのキャパシティ予約は次の要件を満たしている必要があります。

- 両方の予約ともアクティブ状態である必要があります。
- 両方の予約とも AWS アカウント で所有されている必要があります。異なる AWS アカウント が所有する予約間でキャパシティを移動することはできません。
- 両方の予約には、次の内容が同じである必要があります。
  - インスタンスタイプ
  - プラットフォーム
  - アベイラビリティーゾーン
  - テナンシー
  - 配置グループ

- 終了時間

移動先キャパシティ予約インスタンス資格 (open または targeted) およびタグは、移動元の予約と一致する必要はありません。移動元の予約のキャパシティが減少し、移動先の予約のキャパシティが増加していることを除き、両方の予約の設定は変わりません。

移動するインスタンスの数を指定するとき、デフォルトで利用可能なキャパシティが最初に移動され、次に適格な実行中のインスタンス (予約で使用されているキャパシティ) が移動されます。例えば、使用されている 5 インスタンスおよび利用可能な 3 インスタンスがある予約から 4 インスタンスを移動した場合、利用可能な 3 インスタンスおよび使用されている 1 インスタンスが移動されます。

> **Note**
>
> 利用可能なキャパシティを超える移動する数量を指定するとにより、予約から使用されているキャパシティを移動すると、open 状態でキャパシティ予約仕様で起動されたインスタンスのみが移動されます。

考慮事項

キャパシティを予約間で移動するとき、次の考慮事項が適用されます。

- 使用されているキャパシティは、同じアカウントのセットで共有されている open インスタンス資格を持つキャパシティ予約間でのみ移動できます。
- 使用されているキャパシティを移動すると、的確なインスタンスがランダムに選択されます。移動する実行中のインスタンスを指定することはできません。移動量を満たすために十分な数の適格なインスタンスが見つからない場合、移動操作は失敗します。
- ソース予約からすべてのキャパシティを移動した場合、キャパシティ予約は自動的にキャンセルされます。
- 将来の日付のキャパシティ予約 — コミットメント期間中に将来の日付のキャパシティ予約にキャパシティを移動することはできません。

> **Note**
>
> キャパシティブロックからキャパシティの移動はサポートされていません。

キャパシティを移動

キャパシティを移動元キャパシティ予約から移動先キャパシティ予約に移動することができます。

Console

キャパシティを移動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[キャパシティ予約] を選択してください。
3. 移動可能なキャパシティがあるオンデマンドキャパシティ予約 ID を選択してください。
4. [アクション] の [容量の管理] で、[移動] を選択してください。
5. [容量の移動] ページの [移動先キャパシティ予約] で、リストから予約を選択してください。
6. [移動する数量] で、スライダーを使用するか、移動元キャパシティ予約から移動先キャパシティ予約に移動するインスタンスの数を入力してください。
7. 概要を確認し、準備ができたら、[移動] を選択してください。

AWS CLI

キャパシティを移動するには

move-capacity-reservation-instances コマンドを使用します。次の例では、指定した移動元キャパシティ予約から指定した移動先キャパシティ予約に 10 個のインスタンスを移動します。

```
aws ec2 move-capacity-reservation-instances \
    --source-capacity-reservation-id cr-1234567890abcdef0 \
    --destination-capacity-reservation-id cr-021345abcdef56789 \
    --instance-count 10
```

PowerShell

キャパシティを移動するには

Move-EC2CapacityReservationInstance コマンドレットを使用します。次の例では、指定した移動元キャパシティ予約から指定した移動先キャパシティ予約に 10 個のインスタンスを移動します。

```
Move-EC2CapacityReservationInstance `
    -SourceCapacityReservationId cr-1234567890abcdef0 `
```

```
    -DestinationCapacityReservationId cr-021345abcdef56789 `
    -InstanceCount 10
```

既存のキャパシティ予約からキャパシティを分割

既存のキャパシティ予約からキャパシティを分割し、新規の予約を作成できます。キャパシティを分割することにより、元の予約の一部を特定のワークロードに割り当てるか、別の AWS アカウントと共有します。例えば、キャパシティ予約を別のアカウントと部分的に共有するには、キャパシティの一部を分割して小さいサイズのキャパシティ予約を作成できます。この小さいサイズのキャパシティ予約は、「AWS Resource Access Manager」を使用している他のアカウントと共有できます。

既存のキャパシティ予約からキャパシティを分割すると、新規のキャパシティ予約が自動的に作成されます。分割されたインスタンスの数による合計キャパシティの減少を除き、既存の予約は変更されません。既存のキャパシティ予約で実行されているインスタンスは影響を受けません。既存の予約は、1 つの新規のキャパシティ予約のみに分割できます。

新しいキャパシティ予約は、タグを除き、既存のキャパシティ予約と同じ設定になります。デフォルトでは、新しいキャパシティ予約にはタグがありません。分割オペレーション中に新しいタグを指定できます。新しいキャパシティ予約は、必要に応じて作成後に変更することもできます。

分割するインスタンスの数を指定すると、デフォルトで利用可能なキャパシティが最初に分割され、次に適格な実行中のインスタンス (予約で使用されているキャパシティ) が分割されます。例えば、使用されている 5 インスタンスおよび利用可能な 3 インスタンスがあるキャパシティ予約から 4 インスタンスを分割した場合、利用可能な 3 インスタンスおよび使用されている 1 インスタンスが新規の予約に分割されます。

キャパシティを分割する前提条件

前提条件として、キャパシティ予約は次の要件を満たしている必要があります。

- ソース予約はアクティブ状態である必要があります。
- ソース予約は AWS アカウント が所有されている必要があります。

> **Note**
>
> 利用可能なキャパシティを超える分割する数量を指定し、使用されているキャパシティを予約から分割すると、open としてキャパシティ予約仕様で起動したインスタンスのみが分割されます。

考慮事項

キャパシティをある予約から新規の予約に分割するとき、次の考慮事項が適用されます。

- 使用されているキャパシティは、いかなるアカウントと共有されていない「オープン」のインスタンス資格を持つキャパシティ予約に対してのみ分割できます。
- 使用されているキャパシティを分割すると、適格なインスタンスがランダムに選択されます。分割する実行中のインスタンスを指定することはできません。分割数量を満たすために十分な数の適格なインスタンスが見つからない場合、分割操作は失敗します。
- 既存の予約から分割するインスタンスの最大数は、予約のサイズから 1 を引いた数値です。例えば、予約の合計キャパシティが 5 インスタンスの場合、最大 4 インスタンスを新規の予約に分割できます。
- 将来の日付のキャパシティ予約 – コミットメント期間中に将来の日付のキャパシティ予約にキャパシティを分割することはできません。
- リソースグループ – 既存のキャパシティ予約がリソースグループに属している場合、新しいキャパシティ予約はリソースグループに自動的には追加されません。必要に応じて、作成後に新しいキャパシティ予約をリソースグループに追加できます。
- 共有 — 既存のキャパシティ予約がコンシューマーアカウントと共有されている場合、新しいキャパシティ予約はコンシューマーアカウントと自動的には共有されません。必要に応じて、作成後に新しいキャパシティ予約を共有できます。
- クラスタープレイスメントグループ – 既存のキャパシティ予約がクラスタープレイスメントグループに属している場合、新しいキャパシティ予約は同じクラスタープレイスメントグループに作成されます。

> **Note**
>
> キャパシティブロックからのキャパシティの分割はサポートされていません。

タグを使用してキャパシティ予約を分割するためのアクセスを制御する

タグを使用して Amazon EC2 リソースへのアクセスを制御できます。これには、既存のキャパシティ予約からキャパシティを分割して新規のキャパシティ予約を作成することも含まれます。詳細については、「IAM ユーザーガイド」の「タグを使用して AWS リソースへのアクセスの制御」を参照してください。

タグを使用してキャパシティ予約を分割するためのアクセスを制御するには、IAM ポリシーが分割元キャパシティ予約と新しく作成されたキャパシティ予約の両方に対して評価されるため、ポリシーステートメントでリソースタグとリクエストタグの両方を指定する必要があります。次のポリシーの例には、分割元キャパシティ予約用のタグ Owner=ExampleDepartment1 が付いた ec2:ResourceTag 条件キーと新しく作成したキャパシティ予約用のタグ stack=production が付いた ec2:RequestTag 条件キーが含まれています。

```
{
  "Statement": [
    {
      "Sid": "AllowSourceCapacityReservation",
      "Effect": "Allow",
      "Action": "ec2:CreateCapacityReservationBySplitting",
      "Resource": "arn:aws:ec2:us-east-1:111122223333:capacity-reservation/
cr-1234567890abcdef0",
      "Condition": {
        "StringEquals": {
          "ec2:ResourceTag/Owner": "ExampleDepartment1"
        }
      }
    },
    {
      "Sid": "AllowNewlyCreatedCapacityReservation",
      "Effect": "Allow",
      "Action": ["ec2:CreateCapacityReservationBySplitting", "ec2:CreateTags"],
      "Resource": "arn:aws:ec2:us-east-1:111122223333:capacity-reservation/*",
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/stack": "production"
        }
      }
    }
  ]
}
```

キャパシティ分割

既存のキャパシティ予約からキャパシティを分割し、新しいキャパシティ予約を作成できます。

Console

キャパシティを分割するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[キャパシティ予約] を選択してください。
3. 分割可能なキャパシティを持つオンデマンドキャパシティ予約 ID を選択してください。
4. [アクション] の [容量の管理] で、[分割] を選択してください。
5. [キャパシティ予約の分割] ページの [分割する数量] で、スライダーを使用するか、現在の予約から分割するインスタンスの数を入力してください。
6. (オプション) 新しいキャパシティ予約のタグを追加します。
7. 概要を確認し、準備ができたら [分割] を選択してください。

AWS CLI

キャパシティを分割するには

`create-capacity-reservation-by-splitting` コマンドを使用します。次の例では、指定したキャパシティ予約から 10 個のインスタンスを分割して、新しいキャパシティ予約を作成します。

```
aws ec2 create-capacity-reservation-by-splitting \
    --source-capacity-reservation-id cr-1234567890abdef0 \
    --instance-count 10
```

PowerShell

キャパシティを分割するには

New-EC2CapacityReservationBySplitting コマンドレットを使用します。次の例では、指定したキャパシティ予約から 10 個のインスタンスを分割して、新しいキャパシティ予約を作成します。

```
New-EC2CapacityReservationBySplitting `
    -SourceCapacityReservationId cr-1234567890abdef0 `
    -InstanceCount 10
```

キャパシティ予約のキャンセル

次の状態のキャパシティ予約をキャンセルできます。

- assessing
- scheduled
- active
- delayed

> **Note**
>
> キャパシティブロックを変更またはキャンセルすることはできません。詳細については、「機械学習用のキャパシティブロック」を参照してください。

キャパシティ予約が active でコミットメント期間または経過したコミットメント期間がない場合、キャンセル料金は発生しません。

キャパシティ予約が assessing または delayed の場合、キャンセル料金は発生しません。

キャパシティ予約が scheduled または active でコミットメント期間が残っている場合は、キャンセル料金が適用される場合があります。詳細については、「キャンセル料金」を参照してください。

実行中のインスタンスがあるキャパシティ予約をキャンセルした場合、インスタンスは、標準のオンデマンドインスタンス料金または割引料金 (一致する Savings Plans またはリージョンのリザーブドインスタンスがある場合) で、キャパシティ予約外で正常に動作し続けます。

キャパシティ予約をキャンセルすると、それをターゲットとするインスタンスは起動できなくなります。これらのインスタンスを異なる キャパシティ予約をターゲットに設定するように変更し、一致する属性と十分なキャパシティでオープンなキャパシティ予約に起動するか、キャパシティ予約への起動を回避します。詳細については、「インスタンスのキャパシティ予約設定を変更する」を参照してください。

トピック

- キャンセル料金
- キャンセル見積り
- キャパシティ予約のキャンセル

キャンセル料金

開始日から 8 週間以内、または提供後のコミットメント期間中に将来の日付のキャパシティ予約をキャンセルすると、コミットメント期間の割合に対して課金されます。キャパシティはキャンセルするとすぐに解放されますが、予約に対しては短縮されたコミットメント期間の終了までオンデマンド料金で引き続き請求が発生します。一致する Savings Plans またはリージョンのリザーブドインスタンスがある場合、割引料金が自動的に適用されます。

料金を計算するために、Amazon EC2 はコミットメント期間にキャンセルレートを掛けます。開始日に相対的に早くキャンセルするほど、レートは低くなります。次の表にキャンセルレートを示します。

キャンセルレート

キャンセルするタイミング	最低キャンセルレート
元の開始日の 8 週間以上前	料金は発生しない
元の開始日の 2 ~ 8 週間前	コミットメント全体の 25%
元の開始日から 2 週間以内	コミットメント全体の 50%
提供後、コミットメント期間中	残りのコミットメントの 50%

この表は、キャンセル料金に関する一般的なガイダンスを示しています。キャンセル見積りは、正確なキャンセル料金期間を決定します。

例えば、14 日間のコミットメント期間がある 70 インスタンスの`c8g.xlarge` キャパシティ予約があり、元の開始日から 2 週間以内にキャンセルした場合 (50% のキャンセルレート)、7 日間のコミットメントに対して課金されます。この 7 日間では、70 個のインスタンスに対してオンデマンド料金がかかります。

キャンセル料金が発生する場合:

- キャパシティ予約は `cancelling` 状態になります。
- キャパシティ予約を変更、移動、共有、分割することはできません。
- キャパシティ予約は、コンソールと `DescribeCapacityReservations` レスポンスに引き続き表示されます。
- キャパシティ予約は、コミットメントが終了すると `cancelled` に移行します。

キャンセル料金が終了するときを確認するには、describe-capacity-reservations コマンドを使用し、レスポンスの CommitmentInfo.CommitmentEndDate フィールドを確認します。コミットメントが終了し、キャパシティ予約が cancelled に遷移すると、請求は停止します。キャンセル料金が AWS コストと使用状況レポート 2.0 にどのように表示されるかについては、「コストと使用状況レポート 2.0 に表示されるキャンセル料金」を参照してください。

キャパシティ予約のコミットメント期間が残っている場合、インスタンスが実行している間はキャンセルできません。まず実行中のインスタンスを停止または終了する必要があります。これにより、キャンセル料金とオンデマンドインスタンス料金の両方が同時に発生することを防止できます。

キャンセル料金を支払ってキャパシティ予約をキャンセルするには、まずキャンセル見積りを生成し、条件に同意する必要があります。詳細については、「キャンセル見積り」を参照してください。

キャンセル見積り

キャンセルでキャンセル料金が発生する場合は、続行する前にキャンセル見積りを確認して承諾する必要があります。見積りには、以下の詳細を含む正確なキャンセル条件が記載されています。

- キャパシティ予約の現在の設定。
- キャンセル後にキャパシティ予約が遷移する状態 (cancelling)
- 料金が適用されるコミット済みインスタンス数
- 請求期間
- 請求終了日 (コミットメントが終了し、キャパシティ予約が cancelled に移行したとき)

条件を確認したら、見積りを承諾してキャンセルを確認します。

Amazon EC2 コンソールでは、キャンセルを開始すると見積りが自動的に生成されます。条件を確認し、承諾を直接確認できます。AWS CLI または API では、CreateCapacityReservationCancellationQuote を使用して見積りを生成し、見積り ID を CancelCapacityReservation に渡します。

キャンセル見積りは 24 時間有効です。24 時間が経過した後は、新しい見積りを生成する必要があります。既存のキャンセル見積りを表示するには、DescribeCapacityReservationCancellationQuotes を使用します。

> **Important**
>
> キャンセルは元に戻せません。キャンセル見積りを受け入れてキャパシティ予約をキャンセルした後にキャンセルを元に戻すことはできません。

キャパシティ予約のキャンセル

キャパシティ予約をキャンセルするには、次の手順に沿って操作します。

Console

キャパシティ予約をキャンセルするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. [キャパシティ予約] を選択し、キャンセルするキャパシティ予約を選択してください。
3. [予約をキャンセル] を選択します。
4. キャンセル見積りが必要な場合は、請求期間と請求終了日を含むキャンセル条件を確認してください。「**confirm**」と入力して条件を受け入れます。
5. [予約をキャンセル] を選択します。

AWS CLI

キャパシティ予約をキャンセルするには

cancel-capacity-reservation コマンドを使用します。

```
aws ec2 cancel-capacity-reservation \
    --capacity-reservation-id cr-1234567890abcdef0
```

キャンセル見積りが必要な場合は、まず create-capacity-reservation-cancellation-quote を使用して見積りを生成し、次に見積り ID を cancel コマンドに渡します。

```
aws ec2 create-capacity-reservation-cancellation-quote \
    --capacity-reservation-id cr-1234567890abcdef0
```

レスポンス の `CancellationTerms` (`ChargeCommitmentDurationHours` と `ChargeEndDate` を含む) を確認します。次に、見積り ID でキャンセルします。

```
aws ec2 cancel-capacity-reservation \
    --capacity-reservation-id cr-1234567890abcdef0 \
    --quote-id crcq-1a2b3c4d5e6f7g8h9 \
    --apply-cancellation-charges commitment-wind-down
```

PowerShell

キャパシティ予約をキャンセルするには

Remove-EC2CapacityReservation コマンドレットを使用します。

```
Remove-EC2CapacityReservation `
    -CapacityReservationId cr-1234567890abcdef0
```

キャンセル見積りが必要な場合は、まず New-EC2CapacityReservationCancellationQuote を使用して見積りを生成し、次に見積り ID を cancel コマンドレットに渡します。

```
New-EC2CapacityReservationCancellationQuote `
    -CapacityReservationId cr-1234567890abcdef0
```

レスポンス内のキャンセル条件を確認します。次に、見積り ID でキャンセルします。

```
Remove-EC2CapacityReservation `
    -CapacityReservationId cr-1234567890abcdef0 `
    -QuoteId crcq-1a2b3c4d5e6f7g8h9 `
    -ApplyCancellationCharge "commitment-wind-down"
```

プレイスメントグループにキャパシティ予約を使用する

プレイスメントグループでキャパシティ予約を作成して、ワークロードの Amazon EC2 コンピューティングキャパシティを予約できます。

プレイスメントグループでキャパシティ予約を作成すると、必要なときと必要な期間中に、プレイスメントグループでコンピューティングキャパシティにアクセスできるようになります。必要に応じて再びスケールアップできるように、キャパシティの利用を維持しながらスケールダウンできます。

プレイスメントグループでキャパシティ予約を作成したら、他の AWS アカウントと共有できます。詳細については、「プレイスメントグループでキャパシティ予約の共有」を参照してください。

サポートされている配置戦略

次の戦略を使用するプレイスメントグループでキャパシティ予約を作成できます。

- クラスター
- 精度時間

> **Note**
>
> スプレッドとパーティションプレイスメントグループは、キャパシティ予約をサポートしていません。

サポートされているインスタンスタイプ、利用可能なリージョン、配置戦略にの詳細ついては、「Amazon EC2 インスタンスのプレイスメントグループ」を参照してください。

トピック

- 制限事項
- プレイスメントグループでキャパシティ予約を使用する
- プレイスメントグループでキャパシティ予約の共有

制限事項

プレイスメントグループでキャパシティ予約を作成する場合、以下の点をご配慮ください。

- 既存のキャパシティ予約がプレイスメントグループにない場合は、キャパシティ予約を変更してプレイスメントグループ内でキャパシティを予約することはできません。プレイスメントグループでキャパシティを予約するには、プレイスメントグループでキャパシティ予約を作成する必要があります。
- プレイスメントグループでキャパシティ予約を作成した後、プレイスメントグループ外のキャパシティを予約するように変更することはできません。
- プレイスメントグループの既存のキャパシティ予約を変更するか、プレイスメントグループに追加のキャパシティ予約を作成して、プレイスメントグループのリザーブドキャパシティを増やすことができます。ただし、容量不足エラーが発生する可能性が高くなります。
- キャパシティ予約は、所有しているプレイスメントグループからのみ共有できます。キャパシティ予約は、所有していないプレイスメントグループから共有することはできません。

- `active` なキャパシティ予約を持つプレイスメントグループは削除できません。プレイスメントグループのキャパシティ予約をすべて削除する前に、キャンセルする必要があります。

プレイスメントグループでキャパシティ予約を使用する

プレイスメントグループにキャパシティ予約を使用するには、次のステップを実行してください。

> **Note**
>
> 既存のプレイスメントグループでキャパシティ予約を作成する場合、ステップ 1 をスキップします。ステップ 2 と 3 では、既存のプレイスメントグループの ARN を指定します。

タスク

- ステップ 1: (条件付き) キャパシティ予約を使用するプレイスメントグループを作成する
- ステップ 2: プレイスメントグループでキャパシティ予約を作成する
- ステップ 3: プレイスメントグループのキャパシティ予約にインスタンスを起動する

ステップ 1: (条件付き) キャパシティ予約を使用するプレイスメントグループを作成する

新規のプレイスメントグループを作成する必要がある場合にのみ、このステップを実行します。既存のプレイスメントグループを使用するには、このステップをスキップします。ステップ 2 と 3 では、プレイスメントグループの ARN を使用します。

Console

プレイスメントグループを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Placement Groups] (プレイスメントグループ)、[Create placement group] (プレイスメントグループの作成) の順に選択してください。
3. [Name] (名前) で、プレイスメントグループのわかりやすい名前を指定します。
4. [配置戦略] には、配置戦略を選択します。
5. [グループの作成] を選択してください。
6. [プレイスメントグループ] テーブルの [グループ ARN] 列で、作成したプレイスメントグループの ARN をメモします。これは次のステップで必要になります。

AWS CLI

プレイスメントグループを作成するには

create-placement-group コマンドを使用します。

```
aws ec2 create-placement-group \
    --group-name MyPG \
    --strategy strategy
```

出力で返されるプレイスメントグループ ARN は次のステップで必要となるので、メモしておいてください。

PowerShell

プレイスメントグループを作成するには

New-EC2PlacementGroup コマンドレットを使用します。

```
New-EC2PlacementGroup `
    -GroupName my-placement-group `
    -Strategy strategy
```

出力で返されるプレイスメントグループ ARN は次のステップで必要となるので、メモしておいてください。

ステップ 2: プレイスメントグループでキャパシティ予約を作成する

キャパシティ予約を作成するときと同じ方法で、プレイスメントグループでキャパシティ予約を作成します。ただし、キャパシティ予約の作成先となるプレイスメントグループの ARN も指定する必要があります。

考慮事項

- 指定したプレイスメントグループは `available` 状態になっている必要があります。プレイスメントグループが `pending`、`deleting`、`deleted` の状態になっている場合、リクエストは失敗します。
- キャパシティ予約とプレイスメントグループは、同じアベイラビリティーゾーンに存在する必要があります。キャパシティ予約を作成するリクエストで、プレイスメントグループのアベイラビリティーゾーンとは異なるアベイラビリティーゾーンが指定されている場合、リクエストは失敗します。

- キャパシティ予約は、プレイスメントグループでサポートされているインスタンスタイプに対してのみ作成できます。サポートされていないインスタンスタイプを指定すると、リクエストは失敗します。
- プレイスメントグループで open キャパシティ予約を作成し、一致する属性 (プレイスメントグループ ARN、インスタンスタイプ、アベイラビリティーゾーン、プラットフォーム、テナンシー) を持つ既存の実行中のインスタンスがある場合、これらのインスタンスはキャパシティ予約で自動的に実行されます。
- 次のいずれかが当てはまる場合、キャパシティ予約を作成するリクエストは失敗する可能性があります。
  - Amazon EC2 には、リクエストに対応する十分なキャパシティ - がありません。時間をおいてからもう一度試すか、別のアベイラビリティーゾーンを試すか、キャパシティ - を小さくしてみてください。インスタンスタイプとサイズに応じてワークロードに柔軟性がある場合は、別のインスタンス属性を試してみてください。
  - リクエストされた数量は、選択したインスタンスファミリーに対するオンデマンドインスタンスの上限を超えています。インスタンスファミリーに対するオンデマンドインスタンスの上限を上げて、もう一度試してください。詳細については「オンデマンドインスタンスクォータ」を参照してください。

Console

キャパシティ予約を作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. [キャパシティ予約]、[作成キャパシティ予約] の順に選択してください。
3. [キャパシティ予約を作成] ページで、必要に応じてインスタンスタイプ、プラットフォーム、アベイラビリティーゾーン、テナンシー、数量、および終了日を指定します。
4. [プレイスメントグループ] には、キャパシティ予約の作成先となるプレイスメントグループの ARN を選択します。
5. [作成] を選択します。

詳細については「キャパシティ予約の作成」を参照してください。

AWS CLI

キャパシティ予約を作成するには

create-capacity-reservation コマンドを使用します。--placement-group-arn には、キャパシティ予約の作成先となるプレイスメントグループの ARN を指定します。

```
aws ec2 create-capacity-reservation \
    --instance-type instance_type \
    --instance-platform platform \
    --availability-zone-id az_id \
    --instance-count quantity \
    --placement-group-arn "placement_group_arn"
```

PowerShell

キャパシティ予約を作成するには

Add-EC2CapacityReservation コマンドレットを使用します。-PlacementGroupArn には、キャパシティ予約の作成先となるプレイスメントグループの ARN を指定します。

```
Add-EC2CapacityReservation `
    -InstanceType instance_type `
    -InstancePlatform platform `
    -AvailabilityZoneId az_id `
    -InstanceCount quantity `
    -PlacementGroupArn "placement_group_arn"
```

ステップ 3: プレイスメントグループのキャパシティ予約にインスタンスを起動する

次のいずれかの方法により、プレイスメントグループにあるキャパシティ予約にインスタンスを起動できます。

- インスタンスの起動先となるプレイスメントグループの ARN の指定 – プレイスメントグループの ARN を指定すると、Amazon EC2 はこのプレイスメントグループにインスタンスを起動します。次のいずれかの方法を使用します。
  - *open* の指定 — インスタンスの起動リクエストでキャパシティ予約を指定する必要はありません。インスタンスに、指定したプレイスメントグループのキャパシティ予約に一致する属性 (プレイスメントグループ ARN、インスタンスタイプ、アベイラビリティーゾーン、プラットフォーム、テナンシー) がある場合、インスタンスはキャパシティ予約で自動的に実行されます。

  - キャパシティ予約の指定 – キャパシティ予約がターゲットされたインスタンスの起動のみを承諾した場合、リクエストのプレイスメントグループに加えて、ターゲットのキャパシティ予約を指定する必要があります。
  - キャパシティ予約グループの指定 – 詳細については、「Using Capacity Reservation in placement groups with a Capacity Reservation group」を参照してください。
- キャパシティ予約グループのみの指定 – 詳細については、「Using Capacity Reservation in placement groups with a Capacity Reservation group」を参照してください。
- キャパシティ予約のみの指定 – プレイスメントグループのキャパシティ予約にインスタンスを起動できます。

> **Note**
>
> キャパシティ予約のみまたはキャパシティ予約グループのみを指定してインスタンスを起動すると、インスタンスはプレイスメントグループで作成されたキャパシティ予約に起動されますが、インスタンスはプレイスメントグループには直接アタッチされません。

Console

既存のキャパシティ予約でインスタンスを起動するには

1. 手順に従ってインスタンスを起動しますが、次のステップを完了してプレイスメントグループとキャパシティ予約の設定を指定するまでインスタンスを起動しないでください。
2. [高度な詳細] を展開し、以下の操作を行います。
   a. [プレイスメントグループ] には、インスタンスの起動先となるプレイスメントグループを選択します。
   b. [Capacity Reservation] (キャパシティ予約) で、キャパシティ予約の設定に応じて、次のいずれかのオプションを選択してください。
      - [開く] – 一致する属性と十分なキャパシティを持つプレイスメントグループの open キャパシティ予約にインスタンスを起動する方法。
      - [ID 別のターゲット] — ターゲットインスタンスの起動のみを受け入れるキャパシティ予約でインスタンスを起動します。
      - [グループ別のターゲット] — 選択したキャパシティ予約グループ内で一致する属性と使用可能なキャパシティを持つ任意のキャパシティ予約にインスタンスを起動します。

3. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択してください。詳細については「コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する」を参照してください。

AWS CLI

既存のキャパシティ予約でインスタンスを起動するには

run-instances コマンドを使用します。特定のキャパシティ予約またはキャパシティ予約グループをターゲットにする必要がある場合は、--capacity-reservation-specification パラメータを指定します。--placement で、GroupName パラメータを指定し、前のステップで作成したプレイスメントグループの名前を指定します。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --count quantity \
    --instance-type instance_type \
    --key-name key_pair_name \
    --subnet-id subnet-0abcdef1234567890 \
    --capacity-reservation-specification
 CapacityReservationTarget={CapacityReservationId=capacity_reservation_id} \
    --placement "GroupName=placement_group_name"
```

PowerShell

既存のキャパシティ予約でインスタンスを起動するには

New-EC2Instance コマンドレットを使用します。-Placement で、GroupName パラメータを指定し、前のステップで作成したプレイスメントグループの名前を指定します。

```
New-EC2Instance `
    -ImageId ami-0abcdef1234567890 `
    -InstanceType instance_type `
    -KeyName key_pair_name `
    -SubnetId subnet-0abcdef1234567890 `
    -CapacityReservationTarget_CapacityReservationId capacity_reservation_id `
    -Placement_GroupName placement_group_name
```

プレイスメントグループでキャパシティ予約の共有

キャパシティ予約のみを共有するか、キャパシティ予約と作成先のプレイスメントグループを両方とも共有することで、プレイスメントグループでキャパシティ予約を共有できます。

キャパシティ予約のみを共有することで、そのキャパシティ予約に対してのみコンシューマーアカウントにアクセスを許可します。コンシューマーアカウントには、キャパシティ予約が作成されるプレイスメントグループの表示またはアクセスはできません。コンシューマーアカウントによるアクセスをきめ細かく制御できます。コンシューマーアカウントには、プレイスメントグループに関する情報 (ARN を含む) が表示されません。

プレイスメントグループとキャパシティ予約を共有すると、プレイスメントグループはコンシューマーアカウントに表示されて、アクセスできるようになります。インスタンスを起動したり、独自のキャパシティ予約を作成したりすることができます。

詳細については、以下のリソースを参照してください。

- プレイスメントグループのキャパシティ予約にインスタンスを起動する
- 共有キャパシティ予約
- 共有プレイスメントグループ

キャパシティ予約グループ

AWS Resource Groups を使用して、Resource Groupsと呼ばれるキャパシティー予約の論理コレクションを作成できます。リソースグループは、すべて同じ AWS リージョンにある AWS リソースの論理的なグループです。リソースグループの詳細については、『AWS Resource Groups ユーザーガイド』の「 リソースグループとは」を参照してください。

自分のアカウントで所有するキャパシティ予約および他の AWS アカウントから共有を受けているキャパシティ予約は、1 つのリソースグループに含めることができます。また、異なる属性 (インスタンスタイプ、プラットフォーム、アベイラビリティーゾーン、テナンシー) を持つキャパシティ予約も 1 つのリソースグループに入れることができます。

キャパシティ予約のリソースグループを作成すると、個別のキャパシティ予約ではなく、キャパシティ予約のグループをインスタンスのターゲットにできます。キャパシティ予約のグループをターゲットとするインスタンスは、一致する属性 (インスタンスタイプ、プラットフォーム、アベイラビリティーゾーン、テナンシー) と使用可能なキャパシティを持つグループ内のキャパシティ予約と一致します。一致する属性と使用可能な容量を持つキャパシティ予約がグループにない場合、インスタンスはオンデマンド型キャパシティを使用して実行されます。一致するキャパシティ予約が後の段階

でターゲットグループに追加されると、インスタンスは自動的にマッチングされ、リザーブドキャパシティに移動されます。

グループでキャパシティ予約の意図しない使用を防ぐには、キャパシティ予約を明示的にターゲットとするインスタンスだけを受け入れるように、グループのキャパシティ予約を設定します。これを行うには、Amazon EC2 コンソールを使用してキャパシティ予約を作成するときに、[インスタンスの適格性] を [この予約を指定するインスタンスのみ] に設定します。AWS CLI を使用する場合は、キャパシティ予約の作成時に `--instance-match-criteria targeted` を指定します。これにより、グループまたはグループ内のキャパシティ予約を明示的にターゲットとするインスタンスのみが、グループ内で実行できるようになります。

実行中のインスタンスがある間にグループの キャパシティ予約がキャンセルまたは期限切れになった場合、インスタンスは、一致する属性と使用可能な容量を持つグループ内の別のキャパシティ予約に自動的に移動されます。一致する属性と使用可能な容量を持つキャパシティ予約がグループに残っていない場合、インスタンスはオンデマンド型キャパシティで実行されます。一致するキャパシティ予約が後の段階でターゲットグループに追加されると、インスタンスは自動的にリザーブドキャパシティに移動されます。

タスク

- キャパシティ予約グループを作成する
- キャパシティ予約をグループに追加するには
- グループからキャパシティ予約を削除する
- キャパシティ予約グループを削除する
- キャパシティ予約グループを持つプレイスメントグループでのキャパシティ予約の使用

キャパシティ予約グループを作成する

次の例を使用すると、次のリクエストパラメータを使用してキャパシティ予約のリソースグループを作成できます。

- `AWS::EC2::CapacityReservationPool` – リソースグループがインスタンス起動の対象となるようにします。
- `AWS::ResourceGroups::Generic` で `allowed-resource-types` を `AWS::EC2::CapacityReservation` に設定 – リソースグループがキャパシティ予約のみを受け入れるようにします。

グループを作成した後に、そのグループにキャパシティ予約を追加できます。

AWS CLI

キャパシティ予約のグループを作成するには

create-group AWS CLI コマンドを使用します。

```
aws resource-groups create-group \
    --name MyCRGroup \
    --configuration \
        '{"Type": "AWS::EC2::CapacityReservationPool"}' \
        '{"Type": "AWS::ResourceGroups::Generic", "Parameters": [{"Name": "allowed-
resource-types", "Values": ["AWS::EC2::CapacityReservation"]}]}'
```

PowerShell

キャパシティ予約のグループを作成するには

New-RGGroup コマンドレットを使用します。

```
New-RGGroup `
    -Name MyCRGroup `
    -Configuration `
        @{"Type"="AWS::EC2::CapacityReserationPool"} `
        @{"Type"="AWS::ResourceGroups::Generic"; "Parameters"=@{"allowed-resource-
types"=@{"Values"="AWS::EC2::CapacityReservations"}}}
```

キャパシティ予約をグループに追加するには

共有されているキャパシティ予約をグループに追加し、そのキャパシティ予約の共有が解除されると、そのキャパシティ予約はグループから自動的に削除されます。

AWS CLI

キャパシティ予約をグループに追加するには

group-resources コマンドを使用します。

次の例では、指定したグループに 2 つのキャパシティ予約を追加します。

```
aws resource-groups group-resources \
    --group MyCRGroup \
    --resource-arns \
```

```
        arn:aws:ec2:sa-east-1:123456789012:capacity-reservation/cr-1234567890abcdef1
 \
        arn:aws:ec2:sa-east-1:123456789012:capacity-reservation/cr-54321abcdef567890
```

PowerShell

キャパシティ予約をグループに追加するには

Add-RGResource コマンドレットを使用します。

次の例では、指定したグループに 2 つのキャパシティ予約を追加します。

```
Add-RGResource `
    -Group MyCRGroup `
    -ResourceArn `
        "arn:aws:ec2:sa-east-1:123456789012:capacity-
reservation/cr-1234567890abcdef1", `
        "arn:aws:ec2:sa-east-1:123456789012:capacity-
reservation/cr-54321abcdef567890"
```

グループからキャパシティ予約を削除する

次の例を使用して、グループからキャパシティ予約を削除できます。

AWS CLI

グループからキャパシティ予約を削除するには

ungroup-resources コマンドを使用します。

次の例では、指定したグループから 2 つのキャパシティ予約を削除します。

```
aws resource-groups ungroup-resources \
    --group MyCRGroup \
    --resource-arns \
        arn:aws:ec2:sa-east-1:123456789012:capacity-reservation/cr-0e154d26a16094dd
 \
        arn:aws:ec2:sa-east-1:123456789012:capacity-reservation/cr-54321abcdef567890
```

PowerShell

グループからキャパシティ予約を削除するには

Remove-RGResource コマンドレットを使用します。

次の例では、指定したグループから 2 つの キャパシティ予約を削除します。

```
Remove-RGResource `
    -Group MyCRGroup `
    -ResourceArn `
        "arn:aws:ec2:sa-east-1:123456789012:capacity-
reservation/cr-0e154d26a16094dd", `
        "arn:aws:ec2:sa-east-1:123456789012:capacity-
reservation/cr-54321abcdef567890"
```

キャパシティ予約グループを削除する

次の例を使用して、キャパシティ予約グループを削除できます。

AWS CLI

グループを削除するには

delete-group コマンドを使用します。

```
aws resource-groups delete-group --group MyCRGroup
```

PowerShell

グループを削除するには

Remove-RGGroup コマンドレットを使用します。

```
Remove-RGGroup -GroupName MyCRGroup
```

キャパシティ予約グループを持つプレイスメントグループでのキャパシティ予約の使用

Amazon EC2 では、キャパシティ予約グループを持つプレイスメントグループでキャパシティ予約を使用するさまざまな起動方法が用意されています。ワークロードの要件に基づいてキャパシティ予約グループをターゲットにするため、次のいずれかの方法を選択できます。

- プレイスメントグループとキャパシティ予約グループの ARN の指定 – 選択したキャパシティ予約グループで一致する属性と利用可能なキャパシティを持つ利用可能なキャパシティ予約が使用され

ます。選択したグループに、一致する属性と使用可能な容量を持つキャパシティ予約がない場合、インスタンスはオンデマンド型キャパシティに起動します。

> **Note**
>
> この方法でインスタンスを起動すると、インスタンスは指定されたプレイスメントグループに配置されます。

- キャパシティ予約グループのみの指定 – キャパシティ予約グループのみを指定することで、キャパシティ予約グループ内で使用可能なキャパシティがすべて使用されます。インスタンスの起動中にキャパシティは次の順序で使用されます。
  - プレイスメントグループに属しないキャパシティ予約。
  - キャパシティ予約グループ内の任意のプレイスメントグループでのキャパシティ予約。
  - グループに一致する属性と利用可能なキャパシティを持つキャパシティ予約がない場合、インスタンスはオンデマンド型キャパシティを使用して実行されます。インスタンスはプレイスメントグループに配置されません。

> **Note**
>
> キャパシティ予約グループのみを指定してインスタンスを起動すると、インスタンスはプレイスメントグループで作成されたキャパシティ予約に起動されますが、インスタンスはプレイスメントグループには直接アタッチされません。

ローカルゾーンでのキャパシティ予約

ローカルゾーンは、ユーザーに地理的に近い AWS リージョンを拡張したものです。ローカルゾーンで作成したリソースにより、非常に低いレイテンシーの通信がローカルユーザーに提供されます。詳細については、AWS Local Zonesをご参照ください。

VPC をその親 AWS リージョンからローカルゾーンに拡張するには、そのローカルゾーンに新しいサブネットを作成します。ローカルゾーンにサブネットを作成すると、VPC はそのローカルゾーンに拡張されます。ローカルゾーンのサブネットは、VPC 内の他のサブネットと同じように動作します。

Local Zones を使用すると、ユーザーに近い複数の場所にキャパシティ予約を配置できます。通常のアベイラビリティーゾーンでキャパシティ予約 を作成して使用するのと同じ方法で、Local Zones でキャパシティ予約を作成して使用します。同じ機能とインスタンスマッチング動作が適用されま

す。Local Zones でサポートされている料金モデルの詳細については、「AWS Local Zones に関するよくある質問」を参照してください。

考慮事項

ローカルゾーンでキャパシティー予約グループを使用することはできません。

ローカルゾーンでキャパシティー予約を使用するには

1. AWS アカウントでローカルゾーンの使用を有効にします。詳細については、「AWS Local Zones ユーザーガイド」の「Getting started with AWS Local Zones」を参照してください。
2. ローカルゾーンにキャパシティー予約を作成します。アベイラビリティーゾーンには、 ローカルゾーンを選択してください。ローカルゾーンは、AWS リージョンコードの末尾に場所を示す識別子を付加して表します (us-west-2-lax-1a など)。詳細については「キャパシティ予約の作成」を参照してください。
3. ローカルゾーン内にサブネットを作成します。アベイラビリティーゾーンには、 ローカルゾーンを選択してください。詳細については、「Amazon VPC ユーザーガイド」の「Create a subnet in your VPC」を参照してください。
4. インスタンスを起動します。サブネットには、ローカルゾーンのサブネット (subnet-123abc | us-west-2-lax-1a など) を選択し、キャパシティー予約 には、ローカルゾーンで作成したキャパシティー予約に必要な仕様 (open または ID で指定したターゲット) を選択してください。詳細については「既存のキャパシティ予約へのインスタンスの起動」を参照してください。

Wavelength Zone 内のキャパシティ予約

AWS Wavelength を使用することで、デベロッパーはモバイルデバイスおよびエンドユーザー向けに、非常にレイテンシーが低いアプリケーションを構築できます。Wavelength は標準の AWS コンピューティングおよびストレージサービスを通信事業者の 5G ネットワークのエッジにデプロイします。Amazon Virtual Private Cloud (VPC) は、1 つまたは複数の Wavelength Zone に拡張できます。その後、Amazon EC2 インスタンスなどの AWS リソースを使用して、極めて低いレイテンシーやリージョンの AWS サービスへの接続を必要とするアプリケーションを実行できます。詳細については、「AWS Wavelength ゾーン」を参照してください。

オンデマンドのキャパシティ予約を作成する場合は、Wavelength Zone を選択し、Wavelength Zone に関連付けられたサブネットを指定することで、Wavelength Zone 内のキャパシティ予約にインスタンスを起動できます。Wavelength Zone は、AWS リージョンコードの末尾に場所を示す識別子を付加して表します (us-east-1-wl1-bos-wlz-1 など)。

Wavelength Zoneは、すべてのリージョンで利用できるわけではありません。Wavelength Zone をサポートするリージョンについてはAWS Wavelength デベロッパーガイドの利用可能な Wavelength Zone をご参照ください。

考慮事項

Wavelength Zone でキャパシティ予約グループを使用することはできません。

Wavelength Zone キャパシティ予約を使用するには

1. AWS アカウントで Wavelength Zone の使用を有効にします。詳細については、AWS Wavelengthデベロッパーガイドの「の使用開始AWS Wavelength」を参照してください。
2. Wavelength Zone にキャパシティ予約を作成します。[アベイラビリティーゾーン]で、Wavelength を選択してください。Wavelength は、AWS リージョンコードの末尾に場所を示す識別子を付加して表します (`us-east-1-wl1-bos-wlz-1` など)。詳細については「キャパシティ予約の作成」を参照してください。
3. Wavelength Zone にサブネットを作成します。[アベイラビリティーゾーン]で、Wavelength Zone を選択してください。詳細については、「Amazon VPC ユーザーガイド」の「Create a subnet in your VPC」を参照してください。
4. インスタンスを起動します。サブネットには、Wavelength Zone のサブネット (`subnet-123abc | us-east-1-wl1-bos-wlz-1` など) を選択し、キャパシティ予約には、Wavelength で作成したキャパシティ予約に必要な仕様 (open または ID で指定したターゲット) を選択してください。詳細については「既存のキャパシティ予約へのインスタンスの起動」を参照してください。

AWS Outposts でのキャパシティ予約

AWS Outposts は、AWS のインフラストラクチャ、サービス、API、ツールをお客様のオンプレミスまで拡張するフル管理されたサービスです。AWS は、AWS Outposts 管理されたインフラストラクチャへのローカルアクセスを提供することで、AWS リージョンと同じプログラミングインターフェイスを使用してオンプレミスでアプリケーションを構築して実行できるようにします。同時に、コンピューティングとストレージのローカルリソースを使用して、レイテンシーを短縮し、ローカルのデータ処理ニーズに対応します。

Outpost とはお客様のサイトにデプロイされる AWS のコンピューティングおよびストレージキャパシティーのプールです。AWS はAWS リージョンの一部としてこのキャパシティーを運営、監視、管理します。

ユーザーは、自分のアカウントで作成した Outposts にキャパシティ予約を作成できます。これにより、自分のサイトにある Outpost で、コンピューティング性能を予約できるようになります。通常のアベイラビリティーゾーンでキャパシティー予約を作成して使用するのと同じ方法で、Outposts でもキャパシティー予約 を作成して使用できます。同じ機能とインスタンスマッチング動作が適用されます。

また、AWS を使用することで、Outposts にあるキャパシティ予約を自分の組織内の他の AWS Resource Access Manager アカウントと共有できます。キャパシティ予約の共有については、「共有キャパシティ予約」を参照してください。

前提条件

Outpost は、自分のサイトにインストールする必要があります。詳細については、AWS Outposts ユーザーガイドのOutpost を作成し、Outpost 容量を注文するを参照してください。

考慮事項

- Outpost では、キャパシティ予約グループを使用することはできません。

Outpost でキャパシティ予約を使用するには

1. Outpost にサブネットを作成します。詳細については、AWS Outposts ユーザーガイドのサブネットの作成を参照してください。
2. Outpost にキャパシティ予約を作成します。

    a. AWS Outposts コンソールの https://console.aws.amazon.com/outposts/ を開いてください。

    b. ナビゲーションペインで [Outposts]、[アクション]、[キャパシティー予約の作成] の順に選択してください。

    c. 必要に応じてキャパシティ予約を設定し、[作成] を選択してください。詳細については「キャパシティ予約の作成」を参照してください。

    > **Note**
    >
    > [インスタンスタイプ] ドロップダウンには、選択した Outpost でサポートされているインスタンスタイプのみが表示されます。また、[アベイラビリティーゾーン] ドロップダウンには、選択したアウトポストが関連付けられているアベイラビリティーゾーンのみが一覧表示されます。

3. キャパシティー予約 でのインスタンスの起動 [サブネット] では、ステップ 1 で作成したサブネットを選択してください。また、[キャパシティー予約] では、ステップ 2 で作成したキャパシティ予約を選択してください。詳細については、AWS Outposts ユーザーガイドの、Outposts でインスタンスを起動するを参照してください。

共有キャパシティ予約

キャパシティ予約の共有を使用すると、キャパシティ予約の所有者は、リザーブドキャパシティを他の AWS アカウントと共有することや、AWS 組織内で共有することが可能になります。これにより、キャパシティ予約の作成と管理を一元的に行い、リザーブドキャパシティを複数の AWS アカウント間や AWS 組織内で共有できるようになります。

このモデルでは、キャパシティ予約を所有する AWS アカウント (所有者) が、キャパシティ予約を他の AWS アカウント (コンシューマー) と共有します。コンシューマーは、自身のアカウントで所有しているキャパシティ予約にインスタンスを起動する場合と同じように、共有を受けているキャパシティ予約にインスタンスを起動できます。キャパシティ予約の所有者は、共有した キャパシティ予約 と、そこで起動したインスタンスを管理します。所有者は、共有したキャパシティ予約でコンシューマーが起動したインスタンスを変更することはできません。コンシューマーは、共有を受けているキャパシティ予約で起動したインスタンスを管理します。コンシューマーが、キャパシティ予約の所有者や他のコンシューマーが所有するインスタンスを表示したり変更したりすることはできません。

キャパシティ予約の所有者がキャパシティ予約を共有できる相手は次のとおりです。

- AWS の組織内または組織外の特定の AWS アカウント
- AWS 組織内の組織単位
- AWS 組織全体

キャパシティ予約を共有するための前提条件

- キャパシティ予約を共有するには、その予約を自分の AWS アカウント内で所有している必要があります。自身が共有を受けているキャパシティ予約を他者に共有することはできません。
- 共有テナンシーインスタンスのキャパシティ予約のみ共有できます。専用テナンシーインスタンスのキャパシティ予約は共有できません。
- 新規の AWS アカウントや、請求制限履歴のある AWS アカウントでは、キャパシティ予約の共有を使用できません。

- AWS 組織や AWS 組織内の組織単位とキャパシティ予約を共有するには、AWS Organizations で共有を有効にする必要があります。詳細については、AWS RAM ユーザーガイドのAWS Organizations で共有を有効化するを参照してください。
- 共有できるのは `active` または `scheduled` の状態のキャパシティ予約です。その他の状態 (`assessing` や `unsupported` など) のキャパシティ予約は共有できません。

関連サービス

キャパシティ予約の共有は AWS Resource Access Manager AWS RAMと統合されます。AWS RAM は、AWS リソースを任意の AWS アカウントと共有したり、AWS Organizations 経由で共有したりするためのサービスです。AWS RAM を使用すると、リソース共有を作成することで、自身が所有するリソースを共有できます。リソース共有では、共有対象のリソースと、共有先となるコンシューマーを指定します。コンシューマーには、個人の AWS アカウントや、AWS Organizations 内の組織単位または組織全体を指定できます。

AWS RAM の詳細については、AWS RAM ユーザーガイドを参照してください。

アベイラビリティーゾーン間での共有

リソースがリージョンの複数のアベイラビリティーゾーンに分散されるようにするために、アベイラビリティーゾーンは各 アカウントの名前に個別にマッピングされます。このため、アカウントが異なると、アベイラビリティーゾーンの命名方法が異なる場合があります。例えば、`us-east-1a` アカウントのアベイラビリティーゾーン AWS の場所は、別の `us-east-1a` アカウントのアベイラビリティーゾーン AWS の場所と異なる可能性があります。

自身のアカウントを基準にしてキャパシティ予約の場所を特定するには、アベイラビリティーゾーン ID (AZ ID) を使用する必要があります。AZ ID は、すべての AWS アカウントで同じアベイラビリティーゾーンを一貫して示すための一意の識別子です。例えば、`use1-az1` は `us-east-1` リージョンの AZ ID であり、すべての AWS アカウントで同じ場所を示します。

キャパシティ予約を別のアカウントと共有する場合、キャパシティ予約は AZ ID で識別される特定の場所に関連付けられます。コンシューマーアカウントは、そのアカウント内の同じ AZ ID にマッピングされるアベイラビリティーゾーンでのみ、共有キャパシティ予約を使用できます。例えば、`us-east-1a` (AZ ID `use1-az1`) でキャパシティ予約を作成する場合、コンシューマーはそのアカウントで `use1-az1` にマッピングされるアベイラビリティーゾーンでインスタンスを起動する必要があります。そのアベイラビリティーゾーンは、`us-east-1b` などの別の名前が付いている場合があります。

アカウントのアベイラビリティーゾーンの AZ ID を表示するには

1. https://console.aws.amazon.com/ram/home で AWS RAM コンソールを開きます。
2. 現在のリージョンの AZ ID は、画面の右側にある [お客様の AZ ID] パネルに表示されます。

キャパシティ予約の共有

所有するキャパシティ予約を他の AWS アカウントと共有すると、それらのアカウントは所有者のリザーブドキャパシティにインスタンスを起動できます。オープンなキャパシティ予約を共有する場合は、意図しない形でキャパシティ予約が使用されないよう、次の点に注意してください。

- キャパシティ予約の属性に一致するインスタンスをコンシューマーが実行している場合に、`CapacityReservationPreference` パラメータが `open` に設定され、リザーブドキャパシティ内での実行がまだであれば、共有キャパシティ予約が自動的に使用されます。
- 一致する属性 (インスタンスタイプ、プラットフォーム、アベイラビリティーゾーン、テナンシー) を持ち、`CapacityReservationPreference` パラメータが `open` に設定されているインスタンスをコンシューマーが起動する場合、自動的に共有キャパシティ予約に起動されます。

キャパシティ予約を共有するには、リソース共有に追加する必要があります。リソース共有とは、AWS RAM アカウント間で自身のリソースを共有するための AWS リソースです。リソース共有では、共有対象のリソースと、共有先のコンシューマーを指定します。Amazon EC2 コンソールを使用してキャパシティ予約を共有すると、既存のリソース共有に追加されます。キャパシティ予約を新しいリソース共有に追加するには、AWS RAM コンソールを使用してリソース共有を作成する必要があります。

自分が AWS Organizations 内の組織のメンバーであり、所属する組織内での共有が有効化されている場合、共有の前提条件が満たされていれば、共有されたキャパシティ予約へのアクセス許可が組織内のコンシューマーに付与されます。キャパシティ予約が外部のアカウントと共有されている場合、コンシューマーは、リソースへの参加の招待を受け取り、その招待を受け入れた後で、共有されているキャパシティ予約に対するアクセス許可が付与されます。

> **Important**
>
> 共有されているキャパシティ予約でインスタンスを起動する前に、コンソールで共有キャパシティ予約を表示するか、describe-capacity-reservations AWS CLI コマンドを使用してこれを記述することで、共有キャパシティ予約へのアクセス権があることを確認します。コンソールで共有キャパシティ予約を表示できるか、AWS CLI を使ってそれを記述できる場合、

ユーザーはこれを使って、そこでインスタンスを起動することができます。インスタンスをキャパシティ予約で起動しようとして、共有の障害によりアクセスできない場合、そのインスタンスは、オンデマンドキャパシティで起動できます。

Console

Amazon EC2 コンソールを使用して、自身が所有するキャパシティ予約を共有するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[キャパシティーの予約] を選択してください。
3. 共有する キャパシティ予約 を選択し、[アクション]、[Share reservation] の順に選択してください。
4. キャパシティーの予約 の追加先となるリソース共有を選択し、[キャパシティ予約を共有] を選択してください。

   コンシューマーから共有キャパシティ予約にアクセスできるようになるまでに、数分かかることがあります。

AWS RAM コンソールを使用して自分が所有するキャパシティ予約を共有するには

「AWS RAM ユーザーガイド」の「リソース共有の作成」を参照してください。

AWS CLI

自分が所有するキャパシティ予約を共有するには

create-resource-share コマンドを使用します。

```
aws ram create-resource-share \
    --name my-resource-share \
    --resource-arns arn:aws:ec2:us-east-2:123456789012:capacity-
reservation/cr-1234abcd56EXAMPLE
```

PowerShell

自分が所有するキャパシティ予約を共有するには

New-RAMResourceShare コマンドレットを使用します。

```
New-RAMResourceShare `
    -Name my-resource-share `
    -ResourceArn "arn:aws:ec2:us-east-2:123456789012:capacity-
reservation/cr-1234abcd56EXAMPLE"
```

キャパシティ予約の共有を停止する

キャパシティ予約の所有者はキャパシティ予約の共有をいつでも停止できます。以下のルールが適用されます。

- コンシューマーが所有するインスタンスのうち、共有解除の時点で共有キャパシティー内で実行していたインスタンスは、リザーブドキャパシティ内で実行し続けます。コンシューマーがインスタンスを終了すると、キャパシティはキャパシティ予約に復元されます。
- キャパシティ予約の共有先コンシューマーが、このリザーブドキャパシティで新たにインスタンスを起動することはできません。

所有しているキャパシティ予約の共有を停止するには、リソース共有から削除する必要があります。

Console

Amazon EC2 コンソールを使用して、所有しているキャパシティ予約の共有を停止するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[キャパシティーの予約] を選択してください。
3. キャパシティ予約を選択して、[共有] タブを選択してください。
4. [共有] タブに、キャパシティ予約の追加先のリソース共有が一覧表示されます。キャパシティ予約を削除する対象のリソース共有を選択し、[リソース共有から削除] を選択してください。

AWS RAM コンソールを使用して自分が所有しているキャパシティ予約の共有を停止するには

AWS RAM ユーザーガイドのリソース共有の更新を参照してください。

AWS CLI

自分が所有するキャパシティ予約の共有を停止するには

disassociate-resource-share コマンドを使用します。

```
aws ram disassociate-resource-share \
    --resource-share-arn arn:aws:ram:us-east-2:123456789012:resource-share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE \
 --resource-arns arn:aws:ec2:us-east-2:123456789012:capacity-reservation/cr-1234abcd56EXAMPLE
```

PowerShell

自分が所有するキャパシティ予約の共有を停止するには

Disconnect-RAMResourceShare コマンドレットを使用します。

```
Disconnect-RAMResourceShare `
    -ResourceShareArn "arn:aws:ram:us-east-2:123456789012:resource-share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE" `
    -ResourceArn "arn:aws:ec2:us-east-2:123456789012:capacity-reservation/cr-1234abcd56EXAMPLE"
```

共有 Amazon EC2 キャパシティ予約の請求割り当て

デフォルトでは、キャパシティ予約が共有されると、所有者はキャパシティ予約で実行されるインスタンスと、キャパシティ予約で使用可能なキャパシティ (未使用のキャパシティとも呼ばれます) に対して課金されます。一方、コンシューマーは、共有キャパシティ予約で実行されるインスタンスに対してのみ課金されます。

必要に応じて、キャパシティ予約の所有者は、キャパシティ予約で使用可能なキャパシティの請求を、キャパシティ予約が共有されているアカウントのいずれかに割り当てることができます。請求が別のアカウントに割り当てられると、そのアカウントはキャパシティ予約で使用可能なキャパシティの請求の所有者になります。キャパシティ予約で使用可能なキャパシティの料金は、その時点以降、所有者のアカウントではなく、割り当てられたアカウントに請求されます。キャパシティ予約の所有者とキャパシティ予約が共有されているアカウントは、キャパシティ予約で実行されているインスタンスに対して引き続き請求されます。

> **Important**
>
> キャパシティ予約の所有者はリソース所有者であり、キャパシティ予約の管理に引き続き責任を負います。請求が割り当てられているアカウントには追加の権限はありません。キャパシティ予約をキャンセル、変更、共有することはできません。

トピック

- 仕組み
- 考慮事項
- 共有 EC2 キャパシティ予約の請求を別のアカウントに割り当てる
- 共有 EC2 キャパシティ予約の請求割り当てリクエストを表示する
- 共有 EC2 キャパシティ予約の請求を承諾または拒否する
- 共有 EC2 キャパシティ予約の請求割り当てリクエストをキャンセルまたは取り消す
- 共有キャパシティ予約の請求割り当てリクエストをモニタリングする

仕組み

キャパシティ予約の所有者のみが、共有キャパシティ予約の請求を別のアカウントに割り当てることができます。請求は、キャパシティ予約が共有され、キャパシティ予約の所有者と同じ AWS Organizations 支払者アカウントで統合されているアカウントにのみ割り当てることができます。

キャパシティ予約で使用可能なキャパシティの請求を別のアカウントに割り当てるには、キャパシティ予約の所有者が必要なアカウントへのリクエストを開始する必要があります。指定されたアカウントはリクエストを受信し、12 時間以内に承認または拒否する必要があります。

- 承認すると、キャパシティ予約で使用可能なキャパシティ (未使用のキャパシティとも呼ばれます) の請求の所有者になります。それ以降、キャパシティ予約で使用可能なキャパシティの料金は、所有者のアカウントではなく、そのアカウントに請求されます。承認された後は、キャパシティ予約の所有者のみが割り当てられたアカウントからの請求を取り消すことができます。
- 拒否すると、キャパシティ予約の所有者は、キャパシティ予約で使用可能なキャパシティの請求所有者のままになります。キャパシティ予約で使用可能なキャパシティの料金は、引き続き所有者のアカウントに請求されます。
- 12 時間以内にリクエストを承諾または拒否しない場合、リクエストは期限切れとなり、キャパシティ予約で使用可能なキャパシティの料金は引き続き所有者のアカウントに請求されます。

請求が別のアカウントに割り当てられる期間、`Reservation` および `UnusedBox` 明細項目は、所有者のコストと使用状況レポート (CUR) ではなく、割り当てられたアカウントの CUR に表示されます。

次の表は、請求が別のアカウントに割り当てられる前に、キャパシティ予約の所有者とコンシューマーアカウントの CUR に表示される明細項目を示しています。

アカウント	請求が割り当てられる前の CUR 明細項目
キャパシティ予約の所有者	• Reservation • BoxUsage * • UnusedBox
キャパシティ予約が共有されているコンシューマーアカウント	• BoxUsage *

次の表は、請求が別のアカウントに割り当てられた後に、キャパシティ予約の所有者とコンシューマーアカウントの CUR に表示される明細項目を示しています。

アカウント	請求が割り当てられた後の CUR 明細項目
キャパシティ予約の所有者	• BoxUsage *
請求が割り当てられているコンシューマーアカウント	• Reservation • BoxUsage * • UnusedBox
キャパシティ予約が共有されているその他のコンシューマーアカウント	• BoxUsage *

アカウント	請求が割り当てられた後の CUR 明細項目

Note

- * BoxUsage 明細項目は、キャパシティ予約で実行中のインスタンスがある場合にのみ、アカウントの CUR に表示されます。CUR 明細項目の詳細については、「Monitoring Capacity Reservations」を参照してください。
- CUR のキャパシティ予約 ARN を使用して、キャパシティ予約の所有者を決定します。ARN に AWS アカウント ID が含まれている場合は、キャパシティ予約の所有者です。それ以外の場合、キャパシティ予約は別のアカウントによって所有されますが、請求はユーザーに割り当てられます。
- 所有者によってキャパシティ予約に割り当てられたコスト配分タグは、コンシューマーアカウントの CUR に表示されません。コスト配分タグは、キャパシティ予約の所有者の CUR にのみ表示されます。

考慮事項

共有キャパシティ予約の請求を割り当てるときは、次の点に注意してください。

- 部分請求や分割請求の割り当てを行うことはできません。キャパシティ予約で使用可能なすべてのキャパシティの請求は、一度に 1 つのアカウントに割り当てることができます。
- キャパシティ予約の使用可能なキャパシティは、時間の経過とともに変化する可能性があります。これは、割り当てられたアカウントの請求に影響します。例えば、キャパシティ予約の所有者がキャパシティ予約のサイズを大きくした場合、または他のコンシューマーアカウントがキャパシティ予約で実行されているインスタンスを停止または終了した場合、使用可能なキャパシティが増加する可能性があります。
- 請求は、同じ AWS Organizations 支払者アカウントで統合されたコンシューマーアカウントにのみ割り当てることができます。コンシューマーが組織を離れた場合、またはキャパシティ予約が共有されなくなった場合、コンシューマーアカウントからの請求は自動的に取り消されます。
- キャパシティ予約の所有者のみが、保留中の請求割り当てリクエストをキャンセルすることができます。また、キャパシティ予約の所有者のみが、リクエストが承諾された後に、割り当てられたアカウントからの請求を取り消すことができます。

- 所有者が別のアカウントに請求が割り当てられた将来の日付のキャパシティ予約をキャンセルした場合、割り当てられている請求所有者がキャンセル料金を支払います。

共有 EC2 キャパシティ予約の請求を別のアカウントに割り当てる

共有キャパシティ予約の使用可能なキャパシティの請求を別のアカウントに割り当てるには、キャパシティ予約の所有者が必要なアカウントへのリクエストを開始する必要があります。Amazon EC2 コンソールでは、このリクエストは転送リクエストと呼ばれます。

キャパシティ予約の所有者が、キャパシティ予約で使用可能なキャパシティの請求をアカウントに割り当てることができるのは、次の場合のみです。

- キャパシティ予約が既にそのアカウントと共有されている。
- アカウントが、キャパシティ予約の所有者と同じ AWS Organizations 支払者アカウントに統合されている。

請求は、リクエストを承諾した後にのみ、指定されたアカウントに割り当てられます。

キャパシティ予約の所有者がリクエストを開始すると、Amazon EventBridge イベントがリクエストされたアカウントに送信されます。詳細については「共有キャパシティ予約の請求割り当てリクエストをモニタリングする」を参照してください。

Console

共有キャパシティ予約の請求を割り当てるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションパネルで、[キャパシティ予約] を選択し、共有キャパシティ予約を選択してください。
3. [使用可能なキャパシティの請求] セクションで、[請求の割り当て] を選択してください。
4. [請求の割り当て] 画面で、請求を割り当てるコンシューマーアカウントを選択し、[リクエスト] を選択してください。

AWS CLI

共有キャパシティ予約の請求を割り当てるには

associate-capacity-reservation-billing-owner コマンドを使用します。--capacity-reservation-id で、共有キャパシティ予約の ID を指定します。--unused-reservation-billing-owner-id で、請求を割り当てる AWS アカウントの ID を指定します。

```
aws ec2 associate-capacity-reservation-billing-owner \
    --capacity-reservation-id cr-01234567890abcdef \
    --unused-reservation-billing-owner-id 123456789012
```

PowerShell

共有キャパシティ予約の請求を割り当てるには

Register-EC2CapacityReservationBillingOwner コマンドレットを使用します。-CapacityReservationId で、共有キャパシティ予約の ID を指定します。-UnusedReservationBillingOwnerId で、請求を割り当てる AWS アカウントの ID を指定します。

```
Register-EC2CapacityReservationBillingOwner `
    -CapacityReservationId cr-01234567890abcdef `
    -UnusedReservationBillingOwnerId 123456789012
```

共有 EC2 キャパシティ予約の請求割り当てリクエストを表示する

キャパシティ予約の所有者は、開始した最新の請求割り当てリクエストのみを表示できます。また、コンシューマーアカウントは、送信された最新の請求割り当てリクエストのみを表示できます。

リクエストは、cancelled、expired または revoked 状態になってから 24 時間表示できます。24 時間後に、表示できなくなります。

Console

(キャパシティ予約の所有者) 開始したリクエストを表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションパネルで、[キャパシティ予約] を選択し、リクエストを表示する共有キャパシティ予約を選択してください。
3. [使用可能なキャパシティの請求] セクションには、最新のリクエストとその現在の状態が表示されます。

(コンシューマーアカウント) 送信されたリクエストを表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションパネルで、[キャパシティ予約] を選択してください。
3. 保留中のリクエストがある場合、[保留中の請求割り当てリクエスト] バナーが画面の上部に表示されます。バナーが表示されない場合、保留中のリクエストはありません。

   リクエストを表示するには、バナーで [リクエストの確認] を選択してください。

AWS CLI

(キャパシティ予約の所有者) 開始したリクエストを表示するには

describe-capacity-reservation-billing-requests コマンドを使用します。

```
aws ec2 describe-capacity-reservation-billing-requests \
    --role odcr-owner
```

(コンシューマーアカウント) 送信されたリクエストを表示するには

describe-capacity-reservation-billing-requests コマンドを使用します。

```
aws ec2 describe-capacity-reservation-billing-requests \
    --role unused-reservation-billing-owner
```

PowerShell

(キャパシティ予約の所有者) 開始したリクエストを表示するには

Get-EC2CapacityReservationBillingRequest コマンドレットを使用します。

```
Get-EC2CapacityReservationBillingRequest `
    -Role odcr-owner
```

(コンシューマーアカウント) 送信されたリクエストを表示するには

Get-EC2CapacityReservationBillingRequest コマンドレットを使用します。

```
Get-EC2CapacityReservationBillingRequest `
```

```
    -Role unused-reservation-billing-owner
```

リクエストは、次に示す状態のいずれかになります。

状態	説明			
pending	リクエストは承諾または拒否されていませんが、有効期限が切れていません。			
accepted	リクエストは指定されたアカウントによって承諾されました。キャパシティ予約の使用可能なキャパシティの請求は、コンシューマーアカウントに割り当てられています。			
rejected	リクエストはコンシューマーアカウントによって拒否されました。			
cancelled	リクエストは、pending 状態のときにキャパシティ予約の所有者によってキャンセルされました。			
revoked	次のいずれかの理由により、コンシューマーアカウントから請求が取り消されました。 • キャパシティ予約の所有者によって明示的に取り消された。 • キャパシティ予約がコンシューマーアカウントと共有されなくなった。			

状態	説明			
	• コンシューマーアカウントが AWS 組織の一部ではなくなった。			
`expired`	コンシューマーアカウントが 12 時間以内にリクエストを承諾または拒否しなかったため、リクエストの有効期限が切れました。			

共有 EC2 キャパシティ予約の請求を承諾または拒否する

共有されているキャパシティ予約の請求割り当てリクエストを受け取った場合は、それを承諾または拒否できます。リクエストは、承諾または拒否されるまで pending 状態のままになります。

リクエストを承諾すると、accepted 状態になり、その時点以降、そのキャパシティ予約の使用可能な (または未使用の) キャパシティの請求がアカウントに割り当てられます。リクエストを承諾すると、キャパシティ予約の所有者のみがアカウントからの請求を取り消すことができます。

リクエストを拒否すると、rejected 状態になり、キャパシティ予約の使用可能なキャパシティの請求はキャパシティ予約の所有者に割り当てられたままになります。

リクエストが 12 時間以内に承諾または拒否されない場合、リクエストは期限切れになります。リクエストの有効期限が切れても、キャパシティ予約の未使用のキャパシティの請求はキャパシティ予約の所有者に割り当てられます。

リクエストが承諾または拒否されると、Amazon EventBridge イベントがキャパシティ予約の所有者のアカウントに送信されます。リクエストの有効期限が切れると、Amazon EventBridge イベントがキャパシティ予約の所有者とコンシューマーアカウントに送信されます。詳細については「共有キャパシティ予約の請求割り当てリクエストをモニタリングする」を参照してください。

Console

リクエストを承諾または拒否するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

2. ナビゲーションパネルで、[キャパシティ予約] を選択してください。
3. 保留中のリクエストがある場合、[保留中の請求割り当てリクエスト] バナーが画面の上部に表示されます。バナーが表示されない場合、保留中のリクエストはありません。

   リクエストを表示するには、バナーで [リクエストの確認] を選択してください。
4. 承認または拒否するリクエストを選択し、[承認] または [拒否] を選択してください。

AWS CLI

リクエストを承諾するには

accept-capacity-reservation-billing-ownership コマンドを使用します。--capacity-reservation-id で、リクエストを受け入れるキャパシティ予約の ID を指定します。

```
aws ec2 accept-capacity-reservation-billing-ownership \
    --capacity-reservation-id cr-01234567890abcdef
```

リクエストを拒否するには

reject-capacity-reservation-billing-ownership コマンドを使用します。--capacity-reservation-id で、リクエストを拒否するキャパシティ予約の ID を指定します。

```
aws ec2 reject-capacity-reservation-billing-ownership \
    --capacity-reservation-id cr-01234567890abcdef
```

PowerShell

リクエストを承諾するには

Approve-EC2CapacityReservationBillingOwnership コマンドレットを使用します。

```
Approve-EC2CapacityReservationBillingOwnership `
    -CapacityReservationId cr-01234567890abcdef
```

リクエストを拒否するには

Deny-EC2CapacityReservationBillingOwnership コマンドレットを使用します。

```
Deny-EC2CapacityReservationBillingOwnership `
```

```
    -CapacityReservationId cr-01234567890abcdef
```

共有 EC2 キャパシティ予約の請求割り当てリクエストをキャンセルまたは取り消す

pending 請求割り当てリクエストをキャンセルできるのは、キャパシティ予約の所有者のみです。保留中のリクエストがキャンセルされた場合、cancelled 状態になり、キャパシティ予約の使用可能な (または未使用の) キャパシティの請求はキャパシティ予約の所有者に割り当てられたままになります。

リクエストが accepted になった後は、キャパシティ予約の所有者のみが割り当てられたアカウントからの請求を取り消すことができます。請求が取り消された場合、リクエストは revoked 状態になり、キャパシティ予約の使用可能なキャパシティの請求はキャパシティ予約の所有者に再割り当てされます。

リクエストがキャンセルまたは取り消されると、Amazon EventBridge イベントはキャパシティ予約の所有者と指定されたコンシューマーアカウントに送信されます。詳細については「共有キャパシティ予約の請求割り当てリクエストをモニタリングする」を参照してください。

Console

リクエストをキャンセルまたは取り消すには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションパネルで [キャパシティ予約] を選択し、リクエストをキャンセルまたは取り消すキャパシティ予約を選択してください。
3. [使用可能なキャパシティの請求] セクションで、リクエストの現在の状態に応じて、[移転をキャンセルする] または [移転を取り消す] を選択してください。

AWS CLI

リクエストをキャンセルまたは取り消すには

disassociate-capacity-reservation-billing-owner コマンドを使用します。--unused-reservation-billing-owner-id で、リクエストが送信された AWS アカウントの ID を指定します。

```
aws ec2 disassociate-capacity-reservation-billing-owner \
    --capacity-reservation-id cr-01234567890abcdef \
    --unused-reservation-billing-owner-id 123456789012
```

PowerShell

リクエストをキャンセルまたは取り消すには

Unregister-EC2CapacityReservationBillingOwner コマンドレットを使用します。-UnusedReservationBillingOwnerId で、リクエストが送信された AWS アカウントの ID を指定します。

```
Unregister-EC2CapacityReservationBillingOwner `
    -CapacityReservationId cr-01234567890abcdef `
    -UnusedReservationBillingOwnerId 123456789012
```

共有キャパシティ予約の請求割り当てリクエストをモニタリングする

Amazon EC2 は、請求割り当てリクエストの状態が変更されたときに Amazon EventBridge イベントを送信します。

- リクエストが accepted、rejected、expired、revoked 状態になると、イベントはキャパシティ予約の所有者に送信されます。
- リクエストが pending、expired、cancelled、revoked 状態になると、イベントはリクエストされたコンシューマーアカウントに送信されます。

Amazon EventBridge の詳細については、「Amazon EventBridge ユーザーガイド」を参照してください。

Amazon EventBridge イベントパターンを以下に示します。

```
{
   "version":"0",
   "id":"12345678-1234-1234-1234-123456789012",
   "detail-type":"On-Demand Capacity Reservation Billing Ownership Request pending|
accepted|rejected|cancelled|revoked|expired",
   "source":"aws.ec2",
   "account":"account_id",
   "time":"state_change_timestamp",
   "region":"region",
   "resources":[
      "arn:aws:ec2:region:cr_owner_account_id:capacity-reservation/cr_id"
   ],
```

```
   "detail":{
      "capacity-reservation-id":"cr_id",
      "updateTime":timestamp,
      "ownerAccountId":"cr_owner_account_id",
      "unusedReservationChargesOwnerID":"consumer_account_id",
      "status":"pending|accepted|rejected|cancelled|revoked|expired",
      "statusMessage":"message
   }
}
```

以下は、コンシューマーアカウント (111111111111) が共有キャパシティ予約 (cr-01234567890abcdef) の請求割り当てリクエストを受け入れると、キャパシティ予約の所有者 (222222222222) に送信されるイベントの例です。

```
{
   "version":"0",
   "id":"12345678-1234-1234-1234-123456789012",
   "detail-type":"On-Demand Capacity Reservation Billing Ownership Request accepted",
   "source":"aws.ec2",
   "account":"222222222222",
   "time":"2024-09-01Thh:59:59Z",
   "region":"us-east-1",
   "resources":[
      "arn:aws:ec2:us-east-1:222222222222:capacity-reservation/cr-01234567890abcdef"
   ],
   "detail":{
      "capacity-reservation-id":"cr-01234567890abcdef",
      "updateTime":"2024-08-01Thh:59:59Z",
      "ownerAccountId":"222222222222",
      "unusedReservationChargesOwnerID":"111111111111",
      "status":"accepted",
      "statusMessage":"billing transfer status message"
   }
}
```

共有キャパシティ予約 のアクセス許可

所有者のアクセス許可

共有キャパシティ予約の管理とキャンセルは、所有者が行います。所有者は、共有キャパシティ予約内で実行されており、他のアカウントが所有するインスタンスを変更することはできません。共有キャパシティ予約 で起動されされたインスタンスは、所有者が管理します。

コンシューマーのアクセス許可

コンシューマーは、共有 キャパシティ予約で実行している自身のインスタンスを管理します。コンシューマーは、共有キャパシティ予約 をどのような方法で変更することもできません。また、他のコンシューマーまたはキャパシティ予約の所有者が所有するインスタンスを表示または変更することもできません。消費者は、共有予約の合計キャパシティと使用可能なキャパシティのみを表示できます。

請求と使用量測定

キャパシティ予約の共有に追加料金はかかりません。

デフォルトでは、キャパシティ予約の所有者はキャパシティ予約内で実行されるインスタンスと未使用のリザーブドキャパシティに対して課金され、コンシューマーは共有キャパシティ予約内で実行されるインスタンスに対して課金されます。ただし、共有キャパシティ予約の使用可能なキャパシティの請求を特定のコンシューマーアカウントに割り当てることができます。詳細については「共有 Amazon EC2 キャパシティ予約の請求割り当て」を参照してください。

キャパシティ予約の所有者が別の支払いアカウントに属していて、キャパシティ予約がリージョンのリザーブドインスタンスまたは Savings Plans でカバーされている場合、キャパシティ予約の所有者には引き続きリージョンのリザーブドインスタンスまたは Savings Plans の料金が請求されます。この場合、キャパシティ予約の所有者はリージョンのリザーブドインスタンスまたは Savings Plans の料金を支払い、コンシューマーには共有キャパシティ予約で実行されるインスタンスに対して請求が行われます。

インスタンス制限

キャパシティーの予約 の使用量はすべて、キャパシティ予約の所有者のオンデマンドインスタンス制限の対象としてカウントされます。ここでは次の点について説明します。

- 使用されていないリザーブドキャパシティ
- キャパシティ予約の所有者が所有するインスタンスによる使用量
- コンシューマーが所有するインスタンスによる使用量

共有キャパシティー内でコンシューマーによって起動されたインスタンスは、キャパシティ予約の所有者のオンデマンドインスタンス制限の対象としてカウントされます。コンシューマーのインスタンス制限は、コンシューマー自身が所有するオンデマンドインスタンスの制限と、コンシューマーがアクセスできる共有キャパシティ予約内で使用可能なキャパシティーの合計です。

キャパシティー予約フリート

オンデマンドキャパシティー予約フリートとは、キャパシティー予約のグループです。

キャパシティー予約フリートリクエストには、キャパシティー予約フリートの起動に必要なすべての設定情報が含まれます。1 つのリクエストを使用して、指定したターゲット容量まで、複数のインスタンスタイプにわたって、ワークロードに大量の Amazon EC2 キャパシティーを予約できます。

キャパシティー予約フリートを作成した後は、キャパシティー予約フリートの変更やキャンセルなど、フリートのキャパシティー予約の管理を一括で行うことができます。

トピック

- キャパシティー予約フリートの仕組み
- 考慮事項
- 料金
- キャパシティ予約フリートの概念と計画
- キャパシティー予約フリートを作成する
- キャパシティー予約フリートを変更する
- キャパシティー予約フリートをキャンセルする
- キャパシティー予約フリートでの設定例
- キャパシティ予約フリートでのサービスにリンクされたロールの使用

キャパシティー予約フリートの仕組み

キャパシテ予約フリートを作成すると、フリートリクエストで指定した合計のターゲット容量を満たすため、フリートはキャパシティ予約を個別に作成しようとします。

フリートが容量を予約するインスタンスの数は、指定した合計ターゲット容量とインスタンスタイプの重みに依存します。容量を予約するインスタンスタイプは、使用する配分戦略およびインスタンスタイプ優先順位に依存します。

フリートの作成時に十分なキャパシティーがなく、その総ターゲット容量をすぐに満たすことができない場合、フリートは、要求されたキャパシティーを予約が完了するまで、キャパシティー予約を非同期的に作成しようとします。

フリートが総ターゲット容量に達すると、そのキャパシティーを維持しようとします。フリートのキャパシティー予約がキャンセルされた場合、フリートの設定に応じて、フリートは自動的に 1 つ

以上のキャパシティー予約を作成して、失われたキャパシティーを置き換え、その総ターゲット容量を維持します。

フリート内のキャパシティー予約を、個別に管理することはできません。フリートを変更することによって、まとめて管理する必要があります。フリートを変更すると、フリートのキャパシティー予約が自動的に更新され、変更が反映されます。

現在、キャパシティー予約フリートは open インスタンスの一致条件をサポートしています。フリートによって起動されるすべてのキャパシティー予約は、このインスタンス一致基準を自動的に使用します。この基準では、一致する属性 (インスタンスタイプ、プラットフォーム、アベイラビリティーゾーン、テナンシー) を持つ新しいインスタンスと既存のインスタンスは、フリートにより作成されたキャパシティ予約内で自動的に実行されます。キャパシティー予約フリートでは、 対象インスタンス一致基準はサポートされていません。

考慮事項

キャパシティー予約フリートを使用する際には、次の点に注意してください。

- キャパシティー予約フリートは、AWS CLI および AWS API を使用して作成、変更、表示、キャンセルができます。
- フリート内のキャパシティー予約を、個別に管理することはできません。これらは、フリートを変更またはキャンセルすることで、まとめて管理する必要があります。
- キャパシティー予約フリートは、複数のリージョンにまたがることはできません。
- キャパシティー予約フリートは複数のアベイラビリティーゾーンにまたがることはできません。
- キャパシティー予約フリートによって作成されたキャパシティー予約には、AWS で作成された次のタグが自動的に付けられます。
  - キー — `aws:ec2-capacity-reservation-fleet`
  - 値 – *`fleet_id`*

  このタグを使用して、キャパシティー予約フリートによって作成されたキャパシティー予約を識別できます。

料金

キャパシティー予約フリートの使用に追加料金はかかりません。キャパシティー予約フリートによって作成された、個々のキャパシティー予約に対して料金が発生します。キャパシティー予約に対する課金については、キャパシティ予約の料金と請求を参照してください 。

キャパシティ予約フリートの概念と計画

以下に、キャパシティ予約フリートの計画方法と、キャパシティ予約フリートの各概念 (合計ターゲットキャパシティ、配分戦略、インスタンスタイプの重み、インスタンスタイプの優先度など) について説明します。

トピック

- キャパシティ予約フリートを計画する
- 総ターゲット容量
- 配分戦略
- インスタンスタイプの重み
- インスタンスタイプ優先順位

キャパシティ予約フリートを計画する

キャパシティ予約フリートを計画するときは、次のことを行うことをお勧めします。

1. ワークロードに必要なコンピューティング性能の量を決定します。
2. 使用するインスタンスタイプとアベイラビリティーゾーンを決定します。
3. 要件と設定内容に基づいて、各インスタンスタイプに優先度を割り当てます。詳細については「インスタンスタイプ優先順位」を参照してください。
4. ワークロードに適したキャパシティー重み付けシステムを作成します。各インスタンスタイプに重みを割り当て、総ターゲット容量を決定します。詳細については、インスタンスタイプの重みおよび総ターゲット容量を参照してください。
5. キャパシティー予約を無期限に必要とするか、特定の期間だけ必要かを決定します。

総ターゲット容量

総ターゲット容量の定義は、キャパシティー予約フリートで予約されるコンピューティング性能の総量です。キャパシティー予約フリートを作成するときは、総ターゲット容量を指定します。フリートが作成されると、Amazon EC2 は自動的にキャパシティー予約を作成し、総ターゲット容量までキャパシティーを予約します。

キャパシティー予約フリートでキャパシティーを予約する対象のインスタンスの数は、総ターゲット容量と、キャパシティー予約フリート内のインスタンスタイプごとに指定したインスタンスタイプの重みで決まります (`total target capacity/instance type weight=number of instances`)。

ワークロードにとって意味のあるユニット数に基づいて、総ターゲット容量を割り当てることができます。例えば、ワークロードで特定の数の vCPU が必要な場合、必要な vCPU の数に基づいて総ターゲット容量を割り当てることができます。ワークロードに 2048 個の vCPU が必要な場合、合計ターゲット容量として 2048 を指定します。次に、フリートのインスタンスタイプによって提供される vCPU の数に基づいて、インスタンスタイプの重みを割り当てます。例については、インスタンスタイプの重みを参照してください。

配分戦略

キャパシティー予約フリートの割り当て戦略により、キャパシティー予約フリート設定のインスタンスタイプの仕様を基に、リザーブドキャパシティーのリクエストを満たすための方法が決定されます。

現在は、prioritized の割り当て戦略のみがサポートされています。この戦略を使用するキャパシティー予約フリートは、キャパシティ–予約フリート設定で各インスタンスタイプ仕様に割り当てた優先順位に従い、キャパシティー予約を作成します。優先度の値が低いと、使用する優先順位が高くなります。例えば、次のインスタンスタイプと優先度を使用するキャパシティー予約フリートを作成するとします。

- m4.16xlarge – 優先度 = 1
- m5.16xlarge – 優先度 = 3
- m5.24xlarge – 優先度 = 2

フリートは、まず m4.16xlarge のキャパシティー予約の作成を試みます。Amazon EC2 に十分な m4.16xlarge キャパシティーがない場合、フリートは m5.24xlarge のキャパシティー予約の作成を試みます。Amazon EC2 の m5.24xlarge キャパシティーが不十分な場合には、フリートは、m5.16xlarge でキャパシティー予約を作成します。

インスタンスタイプの重み

インスタンスタイプの重みとは、キャパシティー予約フリート内の各インスタンスタイプに割り当てる分量のことです。重みによって、その特定のインスタンスタイプの各インスタンスがフリートの総ターゲット容量にカウントされるキャパシティーのユニット数が決まります。

ワークロードにとって意味のあるユニット数に基づいて重みを割り当てることができます。例えば、ワークロードに特定の数の vCPU が必要な場合、キャパシティー予約フリートの各インスタンスタイプごとに指定した vCPU の数に基づいて重みを割り当てることができます。この場合、m4.16xlarge および m5.24xlarge インスタンスを使用してキャパシティー予約フリートを作成したとすると、次のように各インスタンスの vCPU 数に対応する重みを割り当てます。

- m4.16xlarge – vCPU 数 64、重み = 64 ユニット
- m5.24xlarge – vCPU 数 96、重み = 96 ユニット

インスタンスタイプの重みによって、キャパシティ予約フリートでキャパシティを予約する対象となる、インスタンスの数が決定されます。例えば、総ターゲット容量が 384 ユニットのキャパシティー予約フリートが、前述の例のインスタンスタイプと重みを使用する場合、フリートのキャパシティー予約は、m4.16xlarge を 6 インスタンス (総ターゲット容量 384/インスタンスタイプの重み 64 = 6 インスタンス) になることも、m5.24xlarge を 4 インスタンス (384/96 = 4) になることもあります。

インスタンスタイプのウェイトを割り当てない場合、またはインスタンスタイプの重みに 1 を割り当てた場合には、合計ターゲット容量は純粋にインスタンス数に基づきます。例えば、総ターゲット容量が 384 ユニットのキャパシティー予約フリートが前の例のインスタンスタイプを使用する場合であっても、重みを省略するか、両方のインスタンスタイプに重み 1 を指定すると、フリートのキャパシティー予約は、m4.16xlarge を 384 インスタンスか、m5.24xlarge を 384 インスタンスかのいずれかになります。

インスタンスタイプ優先順位

インスタンスタイプの優先順位は、フリートのインスタンスタイプに割り当てる値です。優先順位は、フリートに指定されているインスタンスタイプのどれに対し、使用上の優先順位を付ける必要があるかを決定します。

優先度の値は、使用する優先順位が高いことを示します。

キャパシティー予約フリートを作成する

キャパシティー予約フリートを作成すると、フリートへのリクエスト内で指定されたインスタンスタイプのキャパシティー予約が、指定された合計ターゲット容量までフリートにより自動的に作成されます。キャパシティー予約フリートがキャパシティーを予約するインスタンスの数は、リクエストで指定する合計ターゲット容量とインスタンスタイプの重みによって異なります。詳細については、インスタンスタイプの重みおよび総ターゲット容量を参照してください。

フリートを作成する際には、使用するインスタンスタイプと、それらのインスタンスタイプごとに優先順位を指定する必要があります。詳細については、配分戦略およびインスタンスタイプ優先順位を参照してください。

> **Note**
>
> サービスにリンクされた AWSServiceRoleForEC2CapacityReservationFleet ロールは、キャパシティー予約フリートを初めて作成するときに、アカウントに自動的に作成されます。詳細については「キャパシティ予約フリートでのサービスにリンクされたロールの使用」を参照してください。

現在、キャパシティー予約フリートは open のインスタンス一致条件のみをサポートしています。

AWS CLI

キャパシティー予約フリートを作成するには

create-capacity-reservation-fleet コマンドを使用します。

```
aws ec2 create-capacity-reservation-fleet \
    --total-target-capacity 24 \
    --allocation-strategy prioritized \
    --instance-match-criteria open \
    --tenancy default \
    --end-date 2021-12-31T23:59:59.000Z \
    --instance-type-specifications file://instanceTypeSpecification.json
```

以下は、instanceTypeSpecification.json の内容です。

```
[
  {
    "InstanceType": "m5.xlarge",
    "InstancePlatform": "Linux/UNIX",
    "Weight": 3.0,
    "AvailabilityZone":"us-east-1a",
    "EbsOptimized": true,
    "Priority" : 1
  }
]
```

以下は出力の例です。

```
{
    "Status": "submitted",
```

```
    "TotalFulfilledCapacity": 0.0,
    "CapacityReservationFleetId": "crf-abcdef01234567890",
    "TotalTargetCapacity": 24
}
```

PowerShell

キャパシティ予約フリートを作成するには

New-EC2CapacityReservationFleet コマンドレットを使用します。

```
New-EC2CapacityReservationFleet `
    -TotalTargetCapacity 24 `
    -AllocationStrategy "prioritized" `
    -InstanceMatchCriterion "open" `
    -Tenancy "default" `
    -EndDate 2021-12-31T23:59:59.000Z `
    -InstanceTypeSpecification $specification
```

仕様は次のように定義されます。

```
$specification = New-Object Amazon.EC2.Model.ReservationFleetInstanceSpecification
$specification.InstanceType = "m5.xlarge"
$specification.InstancePlatform = "Linux/UNIX"
$specification.Weight = 3.0
$specification.AvailabilityZone = "us-east-1a"
$specification.EbsOptimized = $true
$specification.Priority = 1
```

キャパシティー予約フリートを変更する

キャパシティー予約フリートの合計ターゲット容量と日付は、任意のタイミングで変更できます。キャパシティー予約フリートの総ターゲット容量を変更すると、フリートは、新しい総ターゲット容量を満たすように、自動的に新しいキャパシティー予約を作成したり、フリート内の既存のキャパシティー予約を変更またはキャンセルしたりします。フリートの終了日を変更すると、個々のキャパシティー予約の終了日もそれに応じて更新されます。

考慮事項

- フリートを変更すると、そのステータスは modifying に遷移します。フリートのステータスが modifying の間は、他の変更を試みることはできません。

- キャパシティー予約フリートで使用されるテナンシー、アベイラビリティーゾーン、インスタンスタイプ、インスタンスプラットフォーム、優先順位、または重みを変更することはできません。これらのパラメータのいずれかを変更する必要がある場合は、既存のフリートをキャンセルし、必要なパラメータを持つ新しいフリートを作成する必要がある場合があります。
- 同じコマンド内で、`--end-date` と `--remove-end-date` を指定することはできません。

AWS CLI

キャパシティー予約フリートを変更するには

modify-capacity-reservation-fleet コマンドを使用します。

例 1: 総ターゲットキャパシティの変更

```
aws ec2 modify-capacity-reservation-fleet \
    --capacity-reservation-fleet-id crf-01234567890abcedf \
    --total-target-capacity 160
```

例 2: 終了日の変更

```
aws ec2 modify-capacity-reservation-fleet \
    --capacity-reservation-fleet-id crf-01234567890abcedf \
    --end-date 2021-07-04T23:59:59.000Z
```

例 3: 終了日の削除

```
aws ec2 modify-capacity-reservation-fleet \
    --capacity-reservation-fleet-id crf-01234567890abcedf \
    --remove-end-date
```

PowerShell

キャパシティー予約フリートを変更するには

Edit-EC2CapacityReservationFleet コマンドレットを使用します。

例 1: 総ターゲットキャパシティの変更

```
Edit-EC2CapacityReservationFleet `
```

```
    -CapacityReservationFleetId crf-01234567890abcedf `
    -TotalTargetCapacity 160
```

例 2: 終了日の変更

```
Edit-EC2CapacityReservationFleet `
    -CapacityReservationFleetId crf-01234567890abcedf `
    -EndDate 2021-07-04T23:59:59.000Z
```

例 3: 終了日の削除

```
Edit-EC2CapacityReservationFleet `
    -CapacityReservationFleetId crf-01234567890abcedf `
    -RemoveEndDate
```

キャパシティー予約フリートをキャンセルする

キャパシティー予約フリートと予約しているキャパシティーが不要になった場合は、キャンセルできます。フリートをキャンセルすると、そのステータスが cancelled に変わり、キャパシティー予約を新たに作成することはできなくなります。また、フリート内の個々のキャパシティ予約はすべてキャンセルされます。既にリザーブドキャパシティで実行されていたインスタンスは、共有キャパシティで引き続き正常に実行されます。

AWS CLI

キャパシティ予約フリートをキャンセルするには

cancel-capacity-reservation-fleets コマンドを使用します。

```
aws ec2 cancel-capacity-reservation-fleets \
    --capacity-reservation-fleet-ids crf-abcdef01234567890
```

以下は出力の例です。

```
{
    "SuccessfulFleetCancellations": [
        {
            "CurrentFleetState": "cancelling",
            "PreviousFleetState": "active",
            "CapacityReservationFleetId": "crf-abcdef01234567890"
```

```
        }
    ],
    "FailedFleetCancellations": []
}
```

PowerShell

キャパシティ予約フリートをキャンセルするには

Stop-EC2CapacityReservationFleet コマンドレットを使用します。

```
Stop-EC2CapacityReservationFleet `
    -CapacityReservationFleetId crf-abcdef01234567890
```

キャパシティー予約フリートでの設定例

次の例では、m5.4xlarge および m5.12xlarge という 2 つのインスタンスタイプを使用するキャパシティー予約フリートを作成します。

ここでは、指定されたインスタンスタイプによって提供される vCPU の数に基づく、重み付けシステムを使用しています。総ターゲット容量の vCPU 数は 480 です。m5.4xlarge により 16 個の vCPU が提供され、重みとして 16　が得られます。一方、m5.12xlarge では 48 個の vCPU が提供され 48 の重みが得られます。この重み付けシステムは、30 個の m5.4xlarge インスタンス (480/16=30)、または 10 個の m5.12xlarge インスタンス (480/48=10) でキャパシティーを予約するように、キャパシティー予約フリートを設定します。

フリートは、m5.12xlarge の容量を優先するように設定されおり優先順位として 1 を指定します。一方、m5.4xlarge には低い優先順位 2 を指定します。これは、フリートが最初にm5.12xlarge のキャパシテ予約を試みることを意味します。Amazon EC2 の m5.12xlarge キャパシティが不十分な場合にのみ m5.4xlarge キャパシティ予約を試みます

フリートは、Windows インスタンスでキャパシティーを予約します。この予約は、October 31, 2021 の 23:59:59 (UTC) に自動的に期限切れになります。

```
aws ec2 create-capacity-reservation-fleet \
    --total-target-capacity 480 \
    --allocation-strategy prioritized \
    --instance-match-criteria open \
    --tenancy default \
    --end-date 2021-10-31T23:59:59.000Z \
```

```
    --instance-type-specifications file://instanceTypeSpecification.json
```

instanceTypeSpecification.json の内容は次のとおりです。

```
[
    {
        "InstanceType": "m5.4xlarge",
        "InstancePlatform":"Windows",
        "Weight": 16,
        "AvailabilityZone":"us-east-1a",
        "EbsOptimized": true,
        "Priority" : 2
    },
    {
        "InstanceType": "m5.12xlarge",
        "InstancePlatform":"Windows",
        "Weight": 48,
        "AvailabilityZone":"us-east-1a",
        "EbsOptimized": true,
        "Priority" : 1
    }
]
```

キャパシティ予約フリートでのサービスにリンクされたロールの使用

オンデマンドキャパシティー予約フリートは AWS Identity and Access Management (IAM) サービスにリンクされたロール にリンクされたロールを使用します。サービスにリンクされたロールは、キャパシティー予約フリートに直接リンクされた一意のタイプの IAM ロールです。サービスにリンクされたロールは、キャパシティー予約フリートによって事前に定義され、ユーザーに代わってサービスが他の AWS のサービスを呼び出すために必要なアクセス許可がすべて含まれています。

サービスにリンクされたロールを使用することで、必要なアクセス許可を手動で追加する必要がないため、キャパシティ予約フリートの設定が簡単になります。キャパシティー予約フリートは、サービスにリンクされたロールのアクセス許可を定義します。特に定義されている場合を除き、キャパシティー予約フリートのみが、そのロールを引き受けることができます。定義される許可は信頼ポリシーと許可ポリシーに含まれており、その許可ポリシーを他の IAM エンティティにアタッチすることはできません。

サービスリンクロールを削除するには、最初に関連リソースを削除する必要があります。これにより、リソースにアクセスするアクセス許可を不注意で削除する可能性がなくなるので、キャパシティー予約フリートリソースが保護されます。

キャパシティ予約フリートに対するサービスにリンクされたロールの許可

キャパシティー予約フリートでは、AWSServiceRoleForEC2CapacityReservationFleetというサービスにリンクされたロールを使用して、ユーザーに代わってキャパシティー予約フリート内のキャパシティー予約を作成、記述、変更、キャンセルします。

AWSServiceRoleForEC2CapacityReservationFleetというサービスにリンクされたロールは、以下のエンティティを信頼して次のロールを引き受けます。

- `capacity-reservation-fleet.amazonaws.com`

このロールでは `AWSEC2CapacityReservationFleetRolePolicy` AWS マネージドポリシーを使用します。詳細については「AWS マネージドポリシー: AWSEC2CapacityReservationFleetRolePolicy」を参照してください。

サービスリンクロールの作成、編集、削除を IAM エンティティ (ユーザー、グループ、ロールなど) に許可するにはアクセス許可を設定する必要があります。詳細についてはIAM ユーザーガイド の「サービスにリンクされた役割のアクセス許可」を参照してください。

キャパシティ予約フリートでのサービスにリンクされたロールの作成

サービスリンクロールを手動で作成する必要はありません。AWS CLI コマンドの `create-capacity-reservation-fleet`、あるいは `CreateCapacityReservationFleet` API を使用してキャパシティー予約フリートを作成する場合、サービスにリンクされたロールが自動的に作成されます。

このサービスリンクロールを削除した後で再度作成する必要が生じた場合は同じ方法でアカウントにロールを再作成できます。キャパシティ予約フリートを作成するたびに、キャパシティ予約フリートによってサービスにリンクされたロールが自動的に作成されます。

キャパシティ予約フリートでのサービスにリンクされたロールの編集

キャパシティー予約フリートでは、サービスにリンクされたロール AWSServiceRoleForEC2CapacityReservationFleet を編集することはできません。サービスリンクロールの作成後は、さまざまなエンティティがロールを参照する可能性があるため、ロール名を変更することはできません。ただし、IAM を使用してロールの説明を編集することはできます。詳細については、「IAM ユーザーガイド」の「サービスにリンクされたロールの編集」を参照してください。

キャパシティ予約フリートでのサービスにリンクされたロールの削除

サービスリンクロールを必要とする機能やサービスが不要になった場合は、ロールを削除することをお勧めします。そうすることで、使用していないエンティティがアクティブにモニタリングされたり、メンテナンスされたりすることがなくなります。ただし、手動で削除する前に、サービスにリンクされたロールのリソースを削除する必要があります。

> **Note**
>
> このリソースを削除する際に、キャパシティー予約フリートのサービスでロールが使用されていると、削除処理が失敗する場合があります。失敗した場合は数分待ってから操作を再試行してください。

サービスにリンクされたロールである AWSServiceRoleForEC2CapacityReservationFleet を削除するには

1. アカウント内のキャパシティー予約フリートを削除するには、AWS CLI コマンドの `delete-capacity-reservation-fleet` または `DeleteCapacityReservationFleet` API を使用します。
2. IAM コンソール、AWS CLI、または AWS API を使用して、サービスにリンクされたロールである AWSServiceRoleForEC2CapacityReservationFleet を削除します。詳細については、「IAM ユーザーガイド」の「サービスにリンクされたロールの削除」を参照してください

キャパシティ予約フリートでのサービスにリンクされたロールでサポートされるリージョン

キャパシティー予約フリートでは、このサービスが利用可能なすべてのリージョンにおいて、サービスにリンクされたロールの使用をサポートしています。詳細については、「AWS リージョンとエンドポイント」を参照してください。

CloudWatch メトリクスを使用してキャパシティ予約の使用状況をモニタリングする

CloudWatch メトリクスでは、使用状況のしきい値に達したときに通知するように CloudWatch アラームを設定することにより、キャパシティ予約を効率的にモニタリングし、未使用の容量を特定できます。これは、一定量のキャパシティ予約ボリュームを維持し、ボリュームの使用効率を高めるのに役立ちます。

キャパシティ予約 は 5 分ごとにメトリクスデータを CloudWatch に送信します。キャパシティ予約は、アクティブな期間が 5 分未満のメトリクスをサポートしていません。

CloudWatch コンソールでのメトリクスの表示の詳細については、Amazon CloudWatch メトリクスの使用を参照してください。アラームの作成の詳細については、Amazon CloudWatch アラームの作成を参照してください。

内容

- キャパシティ予約の使用状況メトリクス
- キャパシティ予約メトリクスディメンション
- キャパシティ予約の CloudWatch メトリクスの表示

キャパシティ予約の使用状況メトリクス

AWS/EC2CapacityReservations 名前空間には、以下の使用状況メトリクスが含まれています。それらのメトリクスを使用して、オンデマンド容量をモニタリングし、予約に指定したしきい値内に維持できます。

メトリクス	説明
UsedInstanceCount	現在使用中のインスタンスの数。 単位: 個
AvailableInstanceCount	使用可能なインスタンスの数。 単位: 個
TotalInstanceCount	予約済みのインスタンスの合計数。 単位: 個
InstanceUtilization	現在使用中のリザーブドキャパシティーインスタンスの割合。 単位: パーセント

キャパシティ予約メトリクスディメンション

選択したリージョンおよびアカウント内で以下のディメンションを使用して、前の表に示したメトリクスを絞り込むことができます。

ディメンション	説明
(ディメンションなし)	このディメンションは、すべてのキャパシティ予約の指定されたメトリクスをフィルタリングします。
CapacityReservationId	このディメンションは、識別されたキャパシティ予約の指定されたメトリクスをフィルタリングします。
InstanceType	このディメンションは、識別されたインスタンスタイプの指定されたメトリクスをフィルタリングします。
AvailabilityZone	このディメンションは、識別されたアベイラビリティーゾーンの指定されたメトリクスをフィルタリングします。
InstanceMatchCriteria	このディメンションは、識別されたインスタンス一致基準 (open または targeted) の指定されたメトリクスをフィルタリングします。
InstancePlatform	このディメンションは、識別されたプラットフォームの指定されたメトリクスデータをフィルタリングします。
Tenancy	このディメンションは、識別されたテナンシーの指定されたメトリクスをフィルタリングします。

キャパシティ予約の CloudWatch メトリクスの表示

メトリクスはまずサービス名前空間ごとにグループ化され、次にサポートされているディメンションごとにグループ化されます。以下の手順を使用してキャパシティ予約メトリクスを表示できます。

CloudWatch コンソールを使用してキャパシティ予約メトリクスを表示するには

1. CloudWatch コンソールの https://console.aws.amazon.com/cloudwatch/ を開いてください。
2. 必要に応じてリージョンを変更します。ナビゲーションバーから、キャパシティ予約があるリージョンを選択してください。詳細については、リージョンとエンドポイントを参照してください。
3. ナビゲーションペインで [Metrics (メトリクス)] を選択してください。
4. [All metrics (すべてのメトリクス)] で、[EC2 Capacity Reservations (EC2 容量予約)] を選択してください。
5. [すべてのキャパシティ予約]、[キャパシティ予約別]、[インスタンスタイプ別]、[アベイラビリティーゾーン別]、[プラットフォーム別]、[インスタンス一致基準別]、または [テナンシー別] で、前述のメトリクスディメンションから選択してください。メトリクスはそれぞれ、ディメンションなし、`CapacityReservationId`、`InstanceType`、`AvailabilityZone`、`Platform`、`InstanceMa` および `Tenancy` でグループ化されます。
6. メトリクスを並べ替えるには、列見出しを使用します。メトリクスをグラフ表示するには、メトリクスの横にあるチェックボックスをオンにします。

AWS CLI を使用してキャパシティ予約に関するメトリクスを表示するには

次の list-metrics コマンドを使用します。

```
aws cloudwatch list-metrics --namespace "AWS/EC2CapacityReservations"
```

キャパシティ予約の低使用率をモニタリングする

以下を使用して、キャパシティ予約の低使用率をモニタリングできます。

トピック

- Amazon EventBridge イベント
- E メールと AWS Health ダッシュボードの通知

Amazon EventBridge イベント

アカウントのキャパシティ予約の使用率が特定の期間に 20% を下回ると、AWS Health は Amazon EventBridge にイベントを送信します。EventBridge を使用することで、このようなイベントに対応

するプログラム的なアクションをトリガーするルールを設定できます。例えば、7 日間の利用率が 20% を下回った場合に、キャパシティ予約を自動的にキャンセルするルールを作成できます。

EventBridge でのイベントは、JSON オブジェクトとして表されます。イベント固有のフィールドは、JSON オブジェクトの「detail (詳細)」セクションに表示されます。「event」 フィールドにはイベント名が入ります。「result」 フィールドには、イベントをトリガーしたアクションの完了したステータスが入ります。詳細については、「Amazon EventBridge ユーザーガイド」の「Amazon EventBridge のイベントパターン」を参照してください。

詳細については、「Amazon EventBridge ユーザーガイド」を参照してください。

この機能は AWS GovCloud (US) ではサポートされていません。

イベント

キャパシティ予約のキャパシティ使用率が 20% を下回ると、AWS Health は次のイベントを送信します。

- AWS_EC2_ODCR_UNDERUTILIZATION_NOTIFICATION

  次の例は、新しく作成されたキャパシティ予約の 24 時間のキャパシティ使用率が 20% を下回ったときに生成されるイベントです。

```
{
    "version": "0",
    "id": "b3e00086-f271-12a1-a36c-55e8ddaa130a",
    "detail-type": "AWS Health Event",
    "source": "aws.health",
    "account": "123456789012",
    "time": "2023-03-10T12:03:38Z",
    "region": "ap-south-1",
    "resources": [
        "cr-01234567890abcdef"
    ],
    "detail": {
        "eventArn": "arn:aws:health:ap-south-1::event/EC2/
AWS_EC2_ODCR_UNDERUTILIZATION_NOTIFICATION/
AWS_EC2_ODCR_UNDERUTILIZATION_NOTIFICATION_cr-01234567890abcdef-6211-4d50-9286-0c9fbc243f04",
        "service": "EC2",
        "eventTypeCode": "AWS_EC2_ODCR_UNDERUTILIZATION_NOTIFICATION",
        "eventTypeCategory": "accountNotification",
        "startTime": "Fri, 10 Mar 2023 12:03:38 GMT",
        "endTime": "Fri, 10 Mar 2023 12:03:38 GMT",
```

```
        "eventDescription": [
            {
                "language": "en_US",
                "latestDescription": "A description of the event will be provided
 here"
            }
        ],
        "affectedEntities": [
            {
                "entityValue": "cr-01234567890abcdef"
            }
        ]
    }
    }
```

- AWS_EC2_ODCR_UNDERUTILIZATION_NOTIFICATION_SUMMARY

  次の例は、1 つまたは複数のキャパシティ予約の 7 日間のキャパシティ使用率が 20% を下回ったときに生成されるイベントの例です。

```
{
    "version": "0", "id":"7439d42b-3c7f-ad50-6a88-25e2a70977e2",
    "detail-type": "AWS Health Event",
    "source": "aws.health",
    "account": "123456789012",
    "time": "2023-03-07T06:06:01Z",
    "region": "us-east-1",
    "resources": [
        "cr-01234567890abcdef | us-east-1b | t3.medium | Linux/UNIX | 0.0%",
        "cr-09876543210fedcba | us-east-1a | t3.medium | Linux/UNIX | 0.0%"
    ],
    "detail": {
        "eventArn": "arn:aws:health:us-east-1::event/
EC2/AWS_EC2_ODCR_UNDERUTILIZATION_NOTIFICATION_SUMMARY/
AWS_EC2_ODCR_UNDERUTILIZATION_NOTIFICATION_SUMMARY_726c1732-d6f6-4037-b9b8-
bec3c2d3ba65",
        "service": "EC2",
        "eventTypeCode": "AWS_EC2_ODCR_UNDERUTILIZATION_NOTIFICATION_SUMMARY",
        "eventTypeCategory": "accountNotification",
        "startTime": "Tue, 7 Mar 2023 06:06:01 GMT",
        "endTime": "Tue, 7 Mar 2023 06:06:01 GMT",
        "eventDescription": [
            {
```

```
                "language": "en_US",
                "latestDescription": "A description of the event will be provided
 here"
            }
        ],
        "affectedEntities": [
            {
                "entityValue": "cr-01234567890abcdef | us-east-1b | t3.medium |
 Linux/UNIX | 0.0%"
            },
            {
                "entityValue": "cr-09876543210fedcba | us-east-1a | t3.medium |
 Linux/UNIX | 0.0%"
            }
        ]
    }
}
```

EventBridge ルールを作成します

キャパシティ予約使用率が 20% を下回ったときに E メール通知を受け取るには、Amazon SNS トピックを作成してから、AWS_EC2_ODCR_UNDERUTILIZATION_NOTIFICATION イベントの EventBridge ルールを作成します。

Amazon SNS トピックを作成するには

1. Amazon SNS コンソールの https://console.aws.amazon.com/sns/v3/home を開いてください。
2. ナビゲーションペインで、[Topics (トピック)]、[Create topic (トピックの作成)] の順に選択してください。
3. [Type (タイプ)] で、[Standard (標準)] を選択してください。
4. [名前] に新しいトピックの名前を入力してください。
5. [トピックの作成] を選択してください。
6. [Create subscription] を選択してください。
7. [プロトコル] で [E メール] を選択し、次に [エンドポイント] に通知を受信する E メールアドレスを入力してください。
8. [Create subscription] を選択してください。

9. 上記で入力した E メールアドレスには、「AWS Notification - Subscription Confirmation」という件名の E メールメッセージが届きます。指示に沿って操作し、登録を確認します。

EventBridge ルールを作成するには

1. Amazon EventBridge コンソールの https://console.aws.amazon.com/events/ を開いてください。
2. ナビゲーションペインで、[Rules (ルール)] を選択し、[Create rule (ルールの作成)] を選択してください。
3. [名前] に新しいルールの名前を入力してください。
4. [ルールタイプ] で、[イベントパターンを持つルール] を選択してください。
5. [Next] を選択してください。
6. [イベントパターン] では、次のいずれかを実行します。
   a. [イベントソース] で、[AWS のサービス] を選択してください。
   b. [AWS のサービス] で、[AWS Health] を選択してください。
   c. [イベントタイプ] で、[EC2 ODCR 低使用率通知] を選択してください。
7. [Next] を選択してください。
8. [ターゲット 1] で、以下を実行します。
   a. [ターゲットタイプ] で、[AWS のサービス] を選択してください。
   b. [Select a target] (ターゲットの選択) には、[SNS topic] (SNS トピック) を選択してください。
   c. [トピック] で、以前に作成したトピックを選択してください。
9. [次へ] を選択し、もう一度 [次へ] を選択してください。
10. [Create rule] を選択してください。

E メールと AWS Health ダッシュボードの通知

アカウントのキャパシティ予約のキャパシティ使用率が 20% を下回ると、AWS Health は次の E メールと Health Dashboard 通知を送信します。

- 新しく作成されたキャパシティ予約のうち、直近の 24 時間の使用率が 20% を下回ったものについての個別通知です。

- 直近の 7 日間の使用率が 20% を下回ったすべてのキャパシティ予約の概要通知です。

E メール通知および Health Dashboard 通知は、キャパシティ予約を所有する AWS アカウントに関連付けられている E メールアドレスに送信されます。通知には、次の情報が含まれます。

- キャパシティ予約の ID。
- キャパシティ予約のアベイラビリティーゾーン。
- キャパシティ予約の平均使用率。
- キャパシティ予約のインスタンスタイプとプラットフォーム (オペレーティングシステム)。

さらに、アカウントのキャパシティ予約の 24 時間のキャパシティ使用率と 7 日間のキャパシティ使用率が 20% を下回ると、AWS Health はイベントを EventBridge に送信します。EventBridge では、このようなイベントに応じて、E メール通知の送信や AWS Lambda 関数のトリガーなどの自動アクションをアクティブにするルールを作成できます。詳細については「キャパシティ予約の低使用率をモニタリングする」を参照してください。

将来の日付のキャパシティ予約に関する状態の変化をモニタリングする

Amazon EC2 は、将来の日付のキャパシティ予約に関する状態が変化したときに Amazon EventBridge にイベントを送信します。

このイベントの例を以下に示します。この例では、将来の日付のキャパシティ予約が `scheduled` 状態になりました。`detail-type` フィールドで強調表示されている状態を書き留めます。

```
{
   "version":"0",
   "id":"12345678-1234-1234-1234-123456789012",
   "detail-type":"EC2 Capacity Reservation Scheduled",
   "source":"aws.ec2",
   "account":"123456789012",
   "time":"yyyy-mm-ddThh:mm:ssZ",
   "region":"us-east-1",
   "resources":[
      "arn:aws:ec2:us-east-1:123456789012:capacity-reservation/cr-1234567890abcdefg"
   ],
   "detail":{
      "capacity-reservation-id":"cr-1234567890abcdefg",
      "state":"scheduled"
   }
```

```
}
```

`detail-type` フィールドに指定できる値は次のとおりです。

- `Scheduled`
- `Active`
- `Delayed`
- `Unsupported`
- `Failed`
- `Expired`

上記の状態についての詳細は、「キャパシティ予約の状態を表示する」を参照してください。

これらのイベントをモニタリングし、イベントが発生したときに特定のアクションをトリガーする Amazon EventBridge イベントを作成できます。詳細については、「Amazon EventBridge でイベントに応答するルールの作成」を参照してください。

すべてのタスク状態変更イベントをモニタリングするルールを作成するには、次のイベント パターンを使用できます。

```
{
  "source": ["aws.ec2"],
  "detail-type": [{
    "prefix": "EC2 Capacity Reservation"
  }]
}
```

特定の状態の変化のみをモニタリングするルールを作成するには、次のイベントパターンを使用できます。

```
{
  "source": ["aws.ec2"],
  "detail-type": [{
    "prefix": "EC2 Capacity Reservation state"
  }]
}
```

例えば、次のイベントパターンは、将来の日付のキャパシティ予約が `active` 状態になったときに送信されるイベントをモニタリングします。

```
{
  "source": ["aws.ec2"],
  "detail-type": [{
    "prefix": "EC2 Capacity Reservation Active"
  }]
}
```

中断可能なキャパシティ予約

中断可能なキャパシティ予約は、アカウント内の他のワークロードで未使用のキャパシティを一時的に利用できるようにするのに役立ちます。これにより、必要に応じてキャパシティを再利用できます。キャパシティを再利用すると、中断可能な予約内で実行されているインスタンスはすべて終了します。中断可能な予約を作成したら、AWS Resource Access Manager (RAM) を使用して、追加の AWS アカウントまたは AWS 組織とその予約を共有できます。

オフピーク期間中、デプロイ間、またはワークロードのスケールダウン時に未使用のリザーブドキャパシティがある場合は、中断可能なキャパシティ予約を使用します。このキャパシティを使用できる別のチームがある場合は、中断可能なキャパシティ予約を作成してキャパシティを利用できるようにします。重要なワークロードで改めてキャパシティを使う必要が生じた場合は、キャパシティを再利用できます。

中断可能なキャパシティ予約は、次のいずれかとして使用できます。

- キャパシティ所有者 – ソースのキャパシティ予約を所有し、中断可能なキャパシティ予約を作成して未使用のキャパシティを他のチームと共有しつつ、必要に応じて再利用するための制御を維持します。
- キャパシティ消費者 – 共有の中断可能な予約でインスタンスを起動します。所有者がキャパシティを再利用する場合は、インスタンスが終了する可能性があることを理解します。

トピック

- 仕組み
- 料金
- 考慮事項
- キャパシティ所有者の中断可能なキャパシティ予約
- キャパシティ消費者の中断可能なキャパシティ予約
- EventBridge と CloudTrail を使用して中断可能なキャパシティ予約をモニタリングする

仕組み

未使用のキャパシティを他のチームが利用できるようにするには、ソース予約から共有する未使用のインスタンスの数を指定して、中断可能な予約を作成します。中断可能な予約を作成すると、ソース予約からアカウント内の新しい中断可能な予約にこれらのインスタンスが転送されます。

ソース予約と中断可能なキャパシティ予約の関連付けは保持されます。その結果、お客様がキャパシティを再利用すると、実行中の消費者インスタンスはすべて終了し、キャパシティは元のソース予約に復元されます。

主な特徴:

- 未使用のキャパシティを一時的に利用できるようにすると同時に、再利用の制御を維持する
- キャパシティはいつでも再利用できます。詳細については、再利用プロセスと追跡を参照してください。
- AWS Resource Access Manager (RAM) を使用して、追加のアカウントまたは AWS 組織と共有する

料金

中断可能な予約を作成すると、独立した新しい予約として課金されます。これにより、請求が分割されます。

- ソース予約: 合計キャパシティから割り当てられたキャパシティを引いた料金が請求されます
- 中断可能な予約: 割り当てられたキャパシティに対して課金されます

オンデマンドキャパシティ予約の請求の詳細については、「キャパシティ予約の料金と請求」を参照してください。

考慮事項

中断可能なキャパシティ予約を使用する前に、キャパシティ所有者とキャパシティ消費者に適用される次の制限と要件を確認してください。

キャパシティ所有者

- 中断可能なキャパシティ予約を直接変更またはキャンセルすることはできません。編集するには、ソースのキャパシティ予約から割り当てられたキャパシティを更新します。

- 中断可能な予約では、表示、起動、タグ付け、共有、請求の割り当てのみを行うことができます。
- ソースキャパシティ予約ごとに作成できる中断可能な割り当ては 1 つだけです。

キャパシティ消費者

- 中断可能なキャパシティ予約はデフォルトでターゲットに設定されるキャパシティ予約であるため、インスタンスの起動時にターゲットに設定する必要があります。
- キャパシティ予約グループに中断可能なキャパシティ予約を追加することはできません。
- 中断可能なキャパシティ予約は、中断可能なアプリケーションにのみ使用することをお勧めします。
- インスタンスは、所有者がキャパシティを再利用するときに終了します。オンデマンドまたはスポットへのフォールバックはありません。詳細については、「中断エクスペリエンス」を参照してください。

キャパシティ所有者の中断可能なキャパシティ予約

キャパシティ所有者は、ソースキャパシティ予約を所有するアカウントです。中断可能なキャパシティ予約を作成して未使用のキャパシティを他のチームと共有しつつ、必要に応じて再利用するための制御を維持します。

このセクションでは、ユーザー (キャパシティ所有者) が中断可能なキャパシティ予約を作成、変更、再利用、追跡する方法について説明します。

トピック

- 中断可能なキャパシティ予約の作成
- 中断可能なキャパシティ予約を表示する
- 中断可能なキャパシティ予約の変更
- 再利用プロセスと追跡
- 中断可能な予約の共有

中断可能なキャパシティ予約の作成

中断可能なキャパシティ予約を作成して、ソース予約から未使用のキャパシティを他のワークロードで使用できるようにしながら、必要に応じて再利用するための制御を維持します。

前提条件

中断可能な割り当てを作成する前に、ソースのオンデマンドキャパシティ予約が次の要件を満たしていることを確認してください。

- キャパシティ予約は、終了日が設定されていないアクティブな状態である必要があります。保留中、期限切れ、キャンセル済み、または終了日がスケジュールされた予約から割り当てを作成することはできません。
- キャパシティ予約には、割り当て可能なキャパシティが必要です。使用可能なインスタンス (未使用のキャパシティとも呼ばれます) のみを割り当てることができます。
- ソースキャパシティ予約ごとに作成できる中断可能な割り当ては 1 つだけです。割り当てが既に存在する場合は、新しい割り当てを作成する前に、割り当てを変更またはキャンセルする必要があります。
- 中断可能なキャパシティ予約には、一度に最大 1,000 個のインスタンスを割り当てることができます。

コンソールまたは AWS CLI を使用して、中断可能なキャパシティ予約を作成できます。

Console

中断可能なキャパシティ予約を作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. [キャパシティ予約] を選択します。
3. キャパシティ予約を選択します。
4. [アクション]、[中断可能な割り当てを作成] を選択します。
5. [割り当てるインスタンス] で、割り当てるインスタンスの数を入力します。
6. (オプション) タグを追加します。
7. [中断可能なキャパシティ割り当てを作成] を選択します。

AWS CLI

中断可能なキャパシティ予約を作成するには

create-interruptible-capacity-reservation-allocation コマンドを使用します。

```
aws ec2 create-interruptible-capacity-reservation-allocation \
```

```
    --capacity-reservation-id cr-1234567890abcdef0 \
    --instance-count 10
```

中断可能なキャパシティ予約を表示する

中断可能なキャパシティ予約を作成したら、アカウントまたは特定のリソースから中断可能な予約を表示できます。

お使いのアカウントの中断可能なキャパシティ予約をすべて表示する

お使いのアカウントの中断可能なキャパシティ予約を表示するには、次の手順に沿って操作します。

Console

お使いのアカウントの中断可能なキャパシティ予約を表示するには

1. コンソールのキャパシティ予約ページに移動します。
2. タイプ列で [中断可能] な予約を探します。
3. 中断可能な予約を選択して詳細を表示します。

AWS CLI

お使いのアカウントの中断可能なキャパシティ予約を表示するには

```
aws ec2 describe-capacity-reservations \
    --capacity-reservation-id cr-interruptible-id \
    --filters Name=interruptible,Values=true
```

特定のソースから中断可能なキャパシティ予約を表示する

次の手順を使用して、特定のソースのキャパシティ予約から作成された中断可能なキャパシティ予約を表示します。

```
aws ec2 describe-capacity-reservations \
    --capacity-reservation-id cr-source-id
```

レスポンスには中断可能なキャパシティ予約 ID と割り当ての詳細を含む `interruptibleCapacityAllocations` オブジェクトが表示されます。レスポンス構造の詳細に

ついては、「Amazon EC2 API リファレンス」の「InterruptibleCapacityAllocation」を参照してください。

中断可能なキャパシティ予約の変更

中断可能なキャパシティ予約を編集またはキャンセルするには、次の手順に沿って操作します。

> **Note**
>
> - 割り当てを減らすと、まず利用可能なインスタンスを再利用してから、リクエストされた数に達するまでインスタンスを実行します。使用可能なインスタンスで数を完全に満たすことができる場合、終了は発生しません。割り当てられたインスタンス数に対するすべての変更は、中断可能なキャパシティ予約ではなく、ソースキャパシティ予約を通じて行われます。
> - 中断可能なキャパシティ予約は、一度に最大 1,000 インスタンスしか変更できません (増減)。

中断可能なキャパシティ予約を編集する

中断可能なキャパシティ予約を編集するには、次の手順に沿って操作します。

Console

1. ソースキャパシティ予約の詳細ページで、[アクション] を選択します。次に、[中断可能なキャパシティ予約を編集] を選択します。
2. [割り当てるインスタンス] で、新しい数を入力します。
   - 共有するキャパシティをさらに追加する
   - ソースキャパシティ予約にキャパシティを呼び戻す
3. [更新] を選択します。

AWS CLI

```
aws ec2 update-interruptible-capacity-reservation-allocation \
    --capacity-reservation-id cr-1234567890abcdef0 \
    --target-instance-count 80
```

中断可能なキャパシティ予約をキャンセルする

次の手順を使用して、割り当てを完全に削除し、すべてのキャパシティを戻します。

Console

1. ソースキャパシティ予約の詳細ページから、中断可能なキャパシティ割り当ての詳細に移動します。
2. [中断可能な割り当てを編集] を選択します。
3. インスタンス数には、0 と入力します。
4. [更新] を選択します。

AWS CLI

```
aws ec2 update-interruptible-capacity-reservation-allocation \
--capacity-reservation-id cr-1234567890abcdef0 \
--target-instance-count 0
```

再利用プロセスと追跡

キャパシティを再利用する場合:

- 実行中のインスタンスは、EventBridge イベントを通じて 2 分間の中断警告を受け取ります。
- 通知期間後、再利用されたキャパシティで実行中のインスタンスはシャットダウン状態になり、終了します。
- 終了すると、ソースキャパシティ予約で、再利用されたインスタンスをすぐに使用できるようになります。
- 完了すると、割り当てステータスは [更新中] から [アクティブ] に変わります。

インスタンスタイプとシャットダウン時間によっては、完全な再利用までに数分かかる場合があります。プロセスの完了時に受け取る EventBridge 通知の詳細については、「再利用の完了」を参照してください。

再利用ステータスを追跡する

ソース予約を記述して、再利用の進行状況をモニタリングします。

```
aws ec2 describe-capacity-reservations \
--capacity-reservation-id cr-1234567890abcdef0
```

レスポンスには、 interruptibleCapacityAllocation オブジェクト内の以下のフィールドが表示されます。

- instance-count: 現在割り当てられているインスタンス
- target-instance-count: 再利用後にリクエストされた数量
- status: 再利用時には [更新中]、完了時には [アクティブ]

中断可能な予約の共有

中断可能な予約は、AWS Resource Access Manager (RAM) を使用して AWS 組織内でのみ共有できます。

考慮事項:

- 消費者アカウントが組織を離れると、中断可能な予約はそのアカウントから自動的に共有解除されます。
- 共有されていない予約で実行されているすべてのインスタンスは、最終的に終了します。
- 他のすべての共有機能は、標準のキャパシティ予約と同じように機能します。

完全な共有手順については、「キャパシティ予約の共有」を参照してください。

キャパシティ消費者の中断可能なキャパシティ予約

キャパシティ消費者は、共有の中断可能なキャパシティ予約でインスタンスを起動するアカウントです。所有者がキャパシティを再利用するときにはインスタンスが終了する可能性があることを理解します。

このセクションでは、ユーザー (キャパシティ消費者) が中断可能なキャパシティ予約でインスタンスを起動する方法と、所有者がキャパシティを再利用したときに起きることについて説明します。

トピック

- 中断可能なキャパシティ予約を表示する
- 中断可能な予約でインスタンスを起動する

- 中断エクスペリエンス

中断可能なキャパシティ予約を表示する

中断可能なキャパシティ予約を表示するには、次の手順に沿って操作します。

Console

アカウントで中断可能なキャパシティ予約を表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. [キャパシティ予約] を選択します。
3. [タイプ] 列で、[中断可能] とマークされた予約を探します。
4. インスタンスの起動で使用する予約 ID を確認します。

AWS CLI

アカウント内のすべての中断可能なキャパシティ予約を検索するには

describe-capacity-reservations コマンドを使用します。

```
aws ec2 describe-capacity-reservations \
--filters Name=state,Values=active
```

レスポンスで `Interruptible` が `true` に設定されている予約を探します。

中断可能な予約をフィルタリングするには

以下のコマンドを使用します。

```
aws ec2 describe-capacity-reservations \
--capacity-reservation-ids cr-example123 \
--query 'CapacityReservations[?Interruptible==`true`]'
```

> **Note**
>
> 中断可能なキャパシティ予約はデフォルトでターゲットとなるキャパシティ予約であるため、インスタンスの起動時にターゲットに設定する必要があります。オープン予約とは異な

り、中断可能な予約は一致するインスタンスを自動的にカバーしません。起動時に予約 ID を明示的に指定する必要があります。

中断可能な予約でインスタンスを起動する

アカウント内の中断可能なキャパシティ予約で Amazon EC2 インスタンスを起動するには、次の手順に沿って操作します。

> **Note**
>
> 中断可能なキャパシティ予約は、中断可能なアプリケーションにのみ使用することをお勧めします。

Console

中断可能なキャパシティ予約でインスタンスを起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開きます。
2. Amazon EC2 ダッシュボードから、[インスタンスの作成] を選択します。
3. インスタンス設定を構成します。
4. キャパシティ予約の [高度な詳細] で、[アクティブな予約で中断可能なインスタンスを起動] を選択します。
5. 中断可能な予約 ID と新しいインスタンス購入オプションを選択します。
6. [インスタンスを起動] を選択してください。

AWS CLI

```
aws ec2 run-instances \
--instance-type m5.large \
--count 2 \
--image-id ami-12345678 \
--instance-market-options '{
    "MarketType": "interruptible-capacity-reservation"
}' \
--capacity-reservation-specification '{
    "CapacityReservationTarget": {
```

```
        "CapacityReservationId": "cr-abcdef1234567890"
    }
}'
```

Auto Scaling グループを使用してインスタンスを起動する

起動テンプレートで Auto Scaling グループを使用して、中断可能な予約でインスタンスを起動することもできます。中断可能な市場タイプと予約 ID を使用して起動テンプレートを設定し、そのテンプレートを使用して Auto Scaling グループを作成します。詳細については、「EC2 Auto Scaling を使用した中断可能なキャパシティ予約」を参照してください。

中断エクスペリエンス

所有者がキャパシティを再利用すると、インスタンスの終了の 2 分前に中断通知が送信されます。この警告は EventBridge イベントを通じて発生し、インスタンス消費者が次を行う時間を確保します。

- 作業を保存するか、アプリケーションのチェックポイントを行う
- プロセスをシャットダウンする
- インスタンスの終了に備える

EventBridge イベントには、どのインスタンスが終了するかについての詳細と正確な終了時間が含まれます。詳細については、「インスタンスの中断の警告」を参照してください。

EventBridge と CloudTrail を使用して中断可能なキャパシティ予約をモニタリングする

中断可能なキャパシティ予約は EventBridge 通知と CloudTrail イベントを送信し、キャパシティの変更のモニタリングと対応をサポートします。

トピック

- EventBridge 通知
- CloudTrail のイベント

EventBridge 通知

インスタンス消費者は、2 種類の EventBridge 通知を受け取ります。EventBridge 通知を設定する方法については、「Amazon EventBridge ルールの作成」を参照してください。

インスタンスの中断の警告

中断可能な予約でインスタンスを実行している場合、インスタンスが終了する 2 分前にこの通知を受信します。

```
{
    "version": "0",
    "id": "12345678-1234-1234-1234-123456789012",
    "detail-type": "EC2 Capacity Reservation Instance Interruption Warning",
    "source": "aws.ec2",
    "account": "[instance owner Account ID]",
    "time": "[Current time in yyyy-mm-ddThh:mm:ssZ]",
    "resources": "[instance arn]",
    "region": "[region]",
    "detail": {
        "instance-id": "[instance-id]",
        "instance-action": "terminate",
        "instance-termination-time": "yyyy-mm-ddThh:mm:ssZ",
        "azId": "[availability-zone-id]"
    }
}
```

再利用の完了

ソース予約を所有している場合、キャパシティの再利用が完了すると、この通知を受信します。

```
{
    "version": "0",
    "id": "12345678-1234-1234-1234-123456789012",
    "detail-type": "EC2 Interruptible Capacity Reservation Allocation Reclamation
 Completed",
    "source": "aws.ec2",
    "account": "[source Capacity Reservation Owner Account ID]",
    "time": "[Current time in yyyy-mm-ddThh:mm:ssZ]",
    "region": "us-east-1",
    "resources": ["source_cr_arn"],
    "detail": {
        "sourceCapacityReservationId": "string",
        "instanceType": "string",
        "availabilityZoneId": "string",
        "TotalInstanceCount": "current total count in the source",
        "ReclaimedInstanceCount": "count of instances added to the source",
        "targetInstanceCount": "number"
```

```
    }
}
```

CloudTrail のイベント

CloudTrail は、中断可能なキャパシティ予約についてこれらのイベントを記録します。

- `InterruptibleCapacityReservationCreated` — お客様が中断可能な割り当てを作成する場合
- `InterruptibleCapacityReservationAllocationUpdated` — お客様が割り当てを変更する場合
- `InterruptibleCapacityReservationCancelled` — お客様が割り当てをキャンセルする場合
- `CapacityReservationModified` — 当社が割り当てのソース予約を変更する場合
- `InterruptibleCapacityReservationInstancesTerminated` — 当社が再利用中にインスタンスを終了する場合

機械学習用のキャパシティブロック

ML 用のキャパシティブロックを使用すると、短期間の機械学習ワークロードをサポートするため、GPU ベースの高速コンピューティングインスタンスを将来の日付で予約できます。キャパシティブロック内で実行されるインスタンスは、Amazon EC2 UltraClusters 内に自動的に互いに近く配置され、低レイテンシーでペタビットスケールのノンブロッキングネットワーキングを実現します。

キャパシティブロックを使用して、Amazon EC2 UltraServers のキャパシティを予約することもできます。UltraServers は、低レイテンシー、高帯域幅のアクセラレーター相互接続内で複数の Amazon EC2 インスタンスを接続します。UltraServers を使用することにより、トレーニング、ファインチューニング、推論において、コンピューティングとメモリを大量に消費する AI/ML ワークロードを処理できます。詳細については、「Amazon EC2 UltraServers」を参照してください。

キャパシティブロックを使用すると、GPU インスタンスのキャパシティを今後いつ使用できるかを確認でき、都合のよい時間にキャパシティブロックを開始するようにスケジュールできます。キャパシティブロックを予約すると、GPU インスタンスのキャパシティを予測して確保することができます。料金は必要な時間分しか発生しません。ML ワークロードを一度に数日間または数週間サポートするために GPU が必要であり、GPU インスタンスを使用していない間は予約の料金を支払いたくないという場合は、キャパシティブロックをお勧めします。

キャパシティブロックの一般的なユースケースは以下のとおりです。

- 機械学習モデルトレーニングとファインチューニング — 機械学習モデルトレーニングとファインチューニングを完了するために予約した GPU インスタンスに、中断なしにアクセスできます。
- ML 実験とプロトタイプ — GPU インスタンスを必要とする実験の実行およびプロトタイプの構築を短期間で行えます。

キャパシティブロックは、一部の AWS リージョンで特定のインスタンスタイプで使用できます。詳細については「サポートされているインスタンスタイプとリージョン」を参照してください。

キャパシティブロックは、最大 8 週間先を開始時刻に設定して予約することができます。各キャパシティブロックには最大 64 件、キャパシティブロック全体では最大 256 件のインスタンスを含めることが可能です。

トピック

- サポートされているインスタンスタイプとリージョン
- サポートされているプラットフォーム
- 考慮事項
- 関連リソース
- Amazon EC2 キャパシティブロックの仕組み
- キャパシティブロックの料金と請求
- キャパシティブロックを見つけて購入する
- キャパシティブロックを使用してインスタンスを起動する
- キャパシティブロックを表示する
- キャパシティブロックを拡張
- キャパシティブロックを共有する
- UltraServer キャパシティブロックのリソースグループを作成する
- EventBridge を使用してキャパシティブロックをモニタリングする
- AWS CloudTrail を使用したキャパシティブロック API コールのロギング

サポートされているインスタンスタイプとリージョン

インスタンスおよび UltraServer キャパシティブロックは、次のインスタンスタイプと AWS リージョンで使用できます。

> **Note**
>
> 64 インスタンスのキャパシティブロックサイズは、すべての AWS リージョン のすべてのインスタンスタイプでサポートされているわけではありません。

インスタンスキャパシティブロック

インスタンスタイプ	米国東部 (バージニア北部) us-east	米国東部 (オハイオ) us-east	米国西部 (北カリフォルニア) us-west	米国西部 (オレゴン) us-west	欧州 (ストックホルム) (eu-north-1)	欧州 (ロンドン) eu-west	欧州 (スペイン) eu-south-2	アジアパシフィック (東京) ap-northeast-	アジアパシフィック (ソウル) ap-northeast-	アジアパシフィック (ムンバイ) ap-south-1	アジアパシフィック (シドニー ap-southeast-	アジアパシフィック (ジャカルタ) ap-southeast-	アジアパシフィック (メルボルン) ap-southeast-	南米 (サンパウロ) sa-east	AWS GovCloud (米国東部): us-gov-east-1	AWS GovCloud (米国西部): us-gov-west-1	米国東部 (アトランタ) us-east-atl-2	米国西部 (フェニックス) us-west-2-phx-2a
p6-b300.8xlarge	✓			✓											✓		✓	
p6-b200.8xlarge	✓	✓		✓						✓					✓	✓		
p5.4xlarge	✓	✓		✓		✓		✓		✓	✓			✓				
p5.48xlarge	✓	✓	✓	✓	✓	✓		✓		✓	✓	✓		✓			✓	

インスタンスタイプ	米国東部(バージニア北部) us-east	米国東部(オハイオ) us-east	米国西部(北カリフォルニア) us-west	米国西部(オレゴン) us-west	欧州(ストックホルム) (eu-north-1)	欧州(ロンドン) eu-west	欧州(スペイン) eu-south-2	アジアパシフィック(東京) ap-northeast-	アジアパシフィック(ソウル) ap-northeast-	アジアパシフィック(ムンバイ) ap-south-1	アジアパシフィック(シドニー ap-southeast-	アジアパシフィック(ジャカルタ) ap-southeast-	アジアパシフィック(メルボルン) ap-southeast-	南米(サンパウロ) sa-east	AWS GovCloud(米国東部): us-gov-east-1	AWS GovCloud(米国西部): us-gov-west-1	米国東部(アトランタ) us-east-atl-2	米国西部(フェニックス) us-west-2-phx-2a
p5e.4rge	✓	✓	✓	✓	✓	✓		✓		✓	✓	✓		✓				✓
p5en.arge	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓						
p4d.2rge	✓	✓		✓														
p4de.arge	✓			✓														
trn1.arge	✓	✓		✓	✓					✓	✓		✓					
trn2.rge													✓	✓				
trn2.arge		✓																

UltraServer キャパシティブロック

インスタンスタイプ	米国東部 (オハイオ) us-east-2	米国東部 (ダラス) us-east-1-dfw-2a
Trn2	✓	
P6e-GB200		✓

サポートされているプラットフォーム

ML 用のキャパシティブロックは、現在、デフォルトテナンシーのインスタンスおよび UltraServer のみをサポートしています。AWS マネジメントコンソール を使用してキャパシティブロックを購入する場合、デフォルトのプラットフォームは Linux/UNIX です。AWS Command Line Interface AWS CLIまたは AWS SDK を使用してキャパシティブロックを購入する場合、以下のプラットフォームオプションを使用できます。

- Linux/UNIX
- Red Hat Enterprise Linux
- RHEL with HA
- SUSE Linux
- Ubuntu Pro

考慮事項

キャパシティブロックを使用するときは、事前に以下の詳細と制限を念頭におきます。

- UltraServer キャパシティブロックに影響を与える障害が検出された場合、ユーザーに通知されますが、通常はキャパシティブロックのインスタンスを終了するアクションが実行されません。ワークロードへの意図しない中断を最小限に抑えるためです。この通知を受け取った後に UltraServer キャパシティブロックを現状のままで引き続き使用するか、UltraServer キャパシティブロックのインスタンスをすべて終了し、AWS サポートケースを送信することで修復をリクエストできます。サポートケースを受け取ったら、修復が完了した際にユーザーが通知されます。インスタンスを UltraServer キャパシティブロックに再起動できます。
- P6e-GB200 UltraServer キャパシティブロックの場合、キャパシティブロックの終了時刻の 60 分以上前にインスタンスを終了する必要があります。

- ローカルゾーン内のキャパシティブロックを購入して使用するには、ローカルゾーンにオプトインされている必要があります。
- 各キャパシティブロックには最大 64 件、キャパシティブロック全体では最大 256 件のインスタンスを含めることが可能です。
- 早くて 30 分で開始できるキャパシティブロックサービスを記述できます。
- キャパシティブロックは、協定世界時 (UTC) の午前 11 時 30 分に終了します。
- キャパシティブロック内で実行しているインスタンスの終了プロセスは、予約の最終日の協定世界時 (UTC) 午前 11 時に始まります。
- キャパシティブロックの開始時刻は最大 8 週間先を予約できます。
- キャパシティブロックはキャンセルはできません。
- UltraServer キャパシティブロックは AWS アカウント間や AWS 組織内で共有することはできません。
- キャパシティブロックを移動または分割することはできません。
- リソースグループで使用できるのは UltraServer キャパシティブロックのみです。インスタンスキャパシティブロックはリソースグループでは使用できません。詳細については「UltraServer キャパシティブロックのリソースグループを作成する」を参照してください。
- AWS 組織内の全アカウントでキャパシティブロックに予約できるインスタンスの合計数は、特定の日に 256 インスタンスを超えることはできません。
- キャパシティブロックを使用するには、インスタンスが予約 ID を明確にターゲットにしている必要があります。
- キャパシティブロック内のインスタンスは、オンデマンドインスタンスの制限にはカウントされません。
- カスタム AMI を使用する P5 インスタンスの場合は、EFA に必要なソフトウェアと設定があることを確認してください。
- Amazon EKS 管理された型ノードグループについては、「Create a managed node group with Amazon EC2 Capacity Blocks for ML」を参照してください。Amazon EKS セルフ管理された型ノードグループについては、「セルフ管理された型ノードで機械学習用のキャパシティブロックを使用する」を参照してください。

関連リソース

キャパシティブロックを作成したら、キャパシティブロックを使用して次の操作を実行できます。

- インスタンスをキャパシティブロックで起動します。詳細については「キャパシティブロックを使用してインスタンスを起動する」を参照してください。
- Amazon EC2 Auto Scaling グループを作成します。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Use Capacity Blocks for machine learning workloads」を参照してください。

> **Note**
>
> Amazon EC2 Auto Scaling または Amazon EKS を使用する場合は、キャパシティブロック予約の開始時にスケーリングを実行するようにスケジュールできます。スケジュールされたスケーリングでは、AWS が再試行を自動的に処理するため、一時的な障害を処理するための再試行ロジックの実装について心配する必要はありません。

- AWS Parallel Computing Service を使用して機械学習ワークフローを強化します。詳細については、「Capacity Blocks support for AWS Parallel Computing Service」を参照してください。
- AWS ParallelCluster で ML ワークフローを強化します。詳細については、「Enhancing ML workflows with AWS ParallelCluster and Amazon EC2 Capacity Blocks for ML」を参照してください。

AWS Parallel Computing Service の詳細については、「AWS Parallel Computing Service とは」を参照してください。

AWS ParallelClusterの詳細については、とはAWS ParallelClusterを参照してください。

Amazon EC2 キャパシティブロックの仕組み

キャパシティブロックは次の仕様で予約できます。

- 最大 8 週間先の開始時刻を予約する
- 予約期間を 1 ~ 14 日または最大 182 日の 7 の倍数の日数 (例: 21 日、28 日) に設定する
- キャパシティブロックごとに最大 64 個のインスタンスを設定する
- 複数のキャパシティブロック全体で最大 256 個のインスタンスを設定する

Amazon EC2 UltraServers では、各 UltraServer は 1 つのキャパシティブロックに対応します。1 つのリクエストで複数の UltraServer をリクエストできます。

キャパシティブロックを予約するには、インスタンスタイプまたは UltraServer タイプ、必要なインスタンスまたは UltraServer 数、日数、最も早い開始日、最も遅い終了日など、必要なキャパシティを最初に指定します。そうすると、その要件を満たす、利用可能なキャパシティブロックのサービスを確認できます。キャパシティブロックのサービスには、開始時刻、アベイラビリティーゾーン、予約料金などの詳細が記されています。キャパシティブロックサービスの料金は、サービスが提供される時点の需要と供給の状況によって異なります。キャパシティブロックの予約後に料金が変わることはありません。詳細については「キャパシティブロックの料金と請求」を参照してください。

キャパシティブロックのサービスを購入すると、選択した日付とインスタンス数で予約が作成されます。キャパシティブロックの予約が開始されたら、起動リクエストで予約 ID を指定すると、インスタンスの起動をターゲットに設定できます。

予約したすべてのインスタンスを使用できるのは、キャパシティブロックの終了時刻の 30 分前 (インスタンスタイプの場合) または 60 分前 (UltraServer タイプの場合) までです。キャパシティブロックの予約が残り 30 分 (インスタンスタイプの場合) または 60 分 (UltraServer タイプの場合) になると、キャパシティブロックで実行中のすべてのインスタンスの終了プロセスが開始されます。この時間を使ってインスタンスをクリーンアップしてから、キャパシティブロックを次の利用者に渡します。当社は、終了プロセスが始まる 10 分前に EventBridge を通じてイベントを送信します。詳細については「EventBridge を使用してキャパシティブロックをモニタリングする」を参照してください。

キャパシティブロックの料金と請求

Amazon EC2 Capacity Blocks for ML では、料金は予約した分のみ発生します。キャパシティブロックの料金は、購入時のキャパシティブロックの需要と供給の状況に応じて異なります。キャパシティブロックサービスの料金は、予約前に確認できます。キャパシティブロックの料金は、予約時に前払いで請求されます。特定の日付範囲でキャパシティブロックを検索すると、利用可能なキャパシティブロックの中で最も安価なものが表示されます。キャパシティブロックの予約後に料金が変わることはありません。

キャパシティブロックを使用する場合、インスタンスの実行時に使用した、オペレーティングシステムの料金が請求されます。オペレーティングシステムの料金の詳細については、「Amazon EC2 Capacity Blocks for ML Pricing」を参照してください。

請求

キャパシティブロックサービスの料金は前払い制です。キャパシティブロックを購入した後、5 分から 12 時間以内にご利用の AWS アカウントに支払いが処理されます。支払いの処理中は、キャパシティブロックの予約リソースは `payment-pending` の状態となります。支払いがブロック開始時刻

の少なくとも 5 分前、あるいは 12 時間以内 (いずれか早い方) に処理できない場合、キャパシティブロックは解放されて予約状態が payment-failed に変更されます。

料金の処理に成功すると、キャパシティブロックのリソース状態は payment-pending から scheduled に変わります。1 回限りの前払い料金が反映された請求書がお手元に届きます。請求書では、支払い金額をキャパシティブロックの予約 ID に関連付けることが可能です。

キャパシティブロックの予約が開始されると、その予約でインスタンスが実行されている間に使用した、オペレーティングシステムのみに基づいて料金が請求されます。ご自身の使用量および関連の料金は、AWS Cost and Usage Report で、使用した月の請求書で確認できます。

> **Note**
>
> Savings Plans とリザーブドインスタンス割引はキャパシティブロックには適用されません。

請求を表示する

請求書は AWS Billing and Cost Management コンソールで確認できます。キャパシティブロックの前払い料金は、予約の購入月に表示されます。

予約の開始後は、請求書にはブロック予約の使用時間と未使用の時間とが別々の行に表示されます。これらの行項目を使って、予約にどの程度の時間を使用したかを確認できます。プレミアムオペレーティングシステムを使用している場合は、使用時間の行には使用料金のみが表示されます。詳細については「キャパシティブロックの料金と請求」を参照してください。未使用の時間には、追加料金は発生しません。

詳細についてはAWS Billing and Cost Management ユーザーガイドの請求書の表示を参照してください。

キャパシティブロックの開始日が予約の購入月とは異なる場合、前払い料金と予約の使用量は、異なる請求月で表示されます。ではAWS Cost and Usage Report、キャパシティブロックの予約 ID は前払い料金の Reservation/ReservationARN の行項目と、毎月の請求書の LineItem/ResourceID に記載されます。したがって、使用量を対応する前払い価格に関連付けることができます。

キャパシティブロックを見つけて購入する

キャパシティブロックを予約するには、まず、自分のニーズを満たすキャパシティを、利用できる時間帯を見つける必要があります。予約できるキャパシティブロックを見つけるには、以下を指定します。

- 必要なインスタンス数
- インスタンスを必要とする期間
- 予約が必要な日数の範囲

利用可能なキャパシティブロックサービスを見つけるには、予約期間とインスタンス数を指定します。予約期間は、1 日単位で最大 14 日間まで、7 日単位で最大 182 日間まで指定する必要があります。各キャパシティブロックには最大 64 件、キャパシティブロック全体では最大 256 件のインスタンスを含めることが可能です。

仕様に一致するキャパシティブロックをリクエストすると、最大 6 つの利用可能なブロックの詳細が提供されます。すべてのキャパシティブロックは UTC の午前 11 時 30 分に終了するため、同じ日に開始されるブロックには目的の期間に最も近い期間が含まれます。あるブロックには目的の期間より多少短い期間を含む一方、もう 1 つのブロックは目的の期間より多少長い期間を含みます。

サービスの詳細には、予約の開始時刻、予約のアベイラビリティーゾーン、予約の料金が記されています。詳細については「キャパシティブロックの料金と請求」を参照してください。

表示されているキャパシティブロックを購入することもできますし、検索条件を変えて利用可能な他のサービスを探すこともできます。サービスの有効期限は事前に設定されていませんが、サービスの利用は申し込み順となります。

キャパシティブロックのサービスを購入すると、キャパシティブロックが予約されたことを確認する返信がすぐに届きます。その後、アカウントに新しいキャパシティ予約が、予約タイプ `capacity-block` と `start-date` が、購入したサービスの開始時刻に設定されて、表示されます。キャパシティブロックに予約は、`payment-pending` の状態で作成されます。前払い料金の処理が完了すると、予約状態は `scheduled` に変更されます。詳細については「請求」を参照してください。

> **Note**
>
> ローカルゾーン内のキャパシティブロックを購入して使用するには、ローカルゾーンにオプトインされている必要があります。

Console

キャパシティブロックを見つけ、購入するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

2. 画面上部のナビゲーションバーで、AWS リージョンを選択してください。64 インスタンスのキャパシティブロックサイズは、すべてのリージョンのすべてのインスタンスタイプでサポートされているわけではないため、この選択は重要です。
3. ナビゲーションペインで [キャパシティ予約]、[キャパシティブロックの作成] を選択します。
4. [キャパシティブロックタイプ] で、[インスタンス] または [UltraServers] を選択します。
5. [キャパシティの属性] ではキャパシティブロックの検索パラメータを定義できます。デフォルトでは、プラットフォームは Linux です。別のオペレーティングシステムを選択する場合は、AWS CLI を使用します。詳細については「サポートされているプラットフォーム」を参照してください。
6. [合計キャパシティ] (インスタンスの場合) または [UltraServer 数] (UltraServers の場合) で、予約するインスタンスまたは UltraServer の数を指定します。
7. [期間] に予約が必要な日数を入力してください。
8. [キャパシティブロックを検索する日付範囲] で、予約の開始を希望する最も早い日付を入力してください。
9. [キャパシティブロックを検索] を選択してください。
10. 要件を満たすキャパシティブロックがある場合、[おすすめのキャパシティブロック] にそのサービスが表示されます。要件を満たすキャパシティブロックが複数ある場合、最も早く利用できるサービスが表示されます。他のキャパシティブロックサービスを表示するときは、検索条件を変更し、再度 [キャパシティブロックを検索] を選択してください。
11. 購入したいキャパシティブロックサービスが見つかったら、[次へ] を選択してください。
12. (オプション) [タグを追加] ページで、[新しいタグを追加] を選択してください。
13. [確認と購入] ページに、開始日と終了日、期間、インスタンスの合計数、料金が表示されます。

> **Note**
>
> 予約後は、キャパシティブロックをキャンセルできません。

14. ポップアップウィンドウの [キャパシティブロックを購入] で [確認] を選択し、[購入] を選択してください。

AWS CLI

インスタンスキャパシティブロックを検索する方法

「describe-capacity-block-extension-offerings」コマンドを使用します。

次の例では、インスタンスキャパシティブロックを見つけます。

```
aws ec2 describe-capacity-block-offerings \
--instance-type p5.48xlarge \
--instance-count 16 \
--start-date-range 2023-08-14T00:00:00Z \
--end-date-range 2023-10-22-T00:00:00Z \
--capacity-duration-hours 48 \
--all-availability-zones
```

次の例では、UltraServer キャパシティブロックを見つけます。

```
aws ec2 describe-capacity-block-offerings \
--ultraserver-type u-p6e-gb200x72 \
--ultraserver-count 1 \
--start-date-range 2023-08-14T00:00:00Z \
--end-date-range 2023-10-22-T00:00:00Z \
--capacity-duration-hours 48
```

キャパシティブロックを購入するには

「purchase-capacity-block」コマンドを前の例の出力によるキャパシティブロックで提供される ID と一緒に使用します。

```
aws ec2 purchase-capacity-block \
--capacity-block-offering-id cb-0123456789abcdefg \
--instance-platform Linux/UNIX
```

PowerShell

キャパシティブロックを見つける方法

Get-EC2CapacityBlockOffering コマンドレットを使用します。

次の例では、インスタンスキャパシティブロックを見つけます。

```
Get-EC2CapacityBlockOffering `
-InstanceType p5.48xlarge `
-InstanceCount 16 `
-CapacityDurationHour 48 `
-StartDateRange 2023-08-14T00:00:00Z `
```

```
-EndDateRange 2023-10-22-T00:00:00Z `
-AllAvailabilityZones $true
```

次の例では、UltraServer キャパシティブロックを見つけます。

```
Get-EC2CapacityBlockOffering `
-UltraserverType u-p6e-gb200x72 `
-UltraserverCount  1 `
-CapacityDurationHour 48 `
-StartDateRange 2023-08-14T00:00:00Z `
-EndDateRange 2023-10-22-T00:00:00Z
```

キャパシティブロックを購入するには

New-EC2EC2CapacityBlock コマンドレットを前の例の出力からのキャパシティブロックの提供 ID と共に使用します。

```
New-EC2EC2CapacityBlock `
-CapacityBlockOfferingId cb-0123456789abcdefg `
-InstancePlatform Linux/UNIX
```

キャパシティブロックを使用してインスタンスを起動する

キャパシティブロックを使用するには、インスタンスの起動時にキャパシティブロックの予約 ID を指定する必要があります。キャパシティブロックでインスタンスを起動すると、起動したインスタンスの数だけ、使用できるキャパシティの数が減ります。例えば、購入したインスタンスのキャパシティが 8 インスタンスで、4 つのインスタンスを起動した場合、使用できるキャパシティは 4 つ減ります。

予約が終了する前にキャパシティブロックで実行中のインスタンスを終了すると、新しいインスタンスを代わりに起動することができます。キャパシティブロック内のインスタンスを停止または終了すると、インスタンスのクリーンアップに数分かかります。置き換える別のインスタンスを起動できるのは、その後です。この間、インスタンスは停止または shutting-down 状態になります。このプロセスが完了すると、インスタンスの状態が stopped か terminated に変わります。その後、キャパシティブロックの利用可能な容量が更新され、使用できる別のインスタンスが表示されます。

要件

- キャパシティブロックがあるアベイラビリティーゾーンとは異なるアベイラビリティーゾーンのサブネットでインスタンスを起動することはできません。

- インスタンスは、キャパシティブロックのプラットフォームとは異なるプラットフォームの AMI を使用して起動することはできません。
- ローカルゾーン内のキャパシティブロックを使用するには、ローカルゾーンにオプトインされている必要があります。

Console

キャパシティブロックでインスタンスを起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 画面上部のナビゲーションバーで、キャパシティブロックの予約の [リージョン] を選択してください。
3. Amazon EC2 コンソールダッシュボードで、[インスタンスを起動] を選択してください。
4. インスタンスを起動するための手順に従います。
5. [高度な詳細] を展開し、[購入オプション] で [キャパシティブロック] を選択します。次に、以下のいずれかを行ってください。
   - 特定のキャパシティブロックでインスタンスを起動するには、[キャパシティ予約] で [キャパシティ予約を指定] を選択し、キャパシティブロックを選択します。
   - (UltraServers のみ) UltraServer キャパシティブロックリソースグループでインスタンスを起動するには、[キャパシティ予約] で [キャパシティ予約リソースグループを指定] を選択し、次にリソースグループを選択します。
6. [インスタンスを起動] を選択してください。

AWS CLI

キャパシティブロックを使用してインスタンスを起動するには

`instance-market-options MarketType` オプションで run-instances コマンドを使用します。

次の例では、インスタンスを特定のキャパシティブロックで起動します。

```
aws ec2 run-instances \
--image-id ami-0abcdef1234567890 \
--count 1 \
--instance-type p5.48xlarge \
--key-name my-key-pair \
```

```
--subnet-id subnet-0abcdef1234567890 \
--instance-market-options MarketType='capacity-block' \
--capacity-reservation-specification
 CapacityReservationTarget={CapacityReservationId=capacity_block_id}
```

次の例では、インスタンスを UltraServer のキャパシティブロックリソースグループに起動します。

```
aws ec2 run-instances \
--image-id ami-0abcdef1234567890 \
--count 1 \
--instance-type p6e-gb200.36xlarge \
--key-name my-key-pair \
--subnet-id subnet-0abcdef1234567890 \
--instance-market-options MarketType='capacity-block' \
--capacity-reservation-specification
 CapacityReservationTarget={CapacityReservationResourceGroupArn=resource_group_arn}
```

PowerShell

キャパシティブロックでインスタンスを起動するには

次に定義されているように、-InstanceMarketOption オプションで New-EC2Instance コマンドレットを使用します。

```
$marketoption = New-Object Amazon.EC2.Model.InstanceMarketOptionsRequest
$marketoption.MarketType = "capacity-block"
```

次の例では、インスタンスを特定のキャパシティ予約のみで起動します。

```
New-EC2Instance `
-ImageId ami-0abcdef1234567890 `
-InstanceType p5.48xlarge `
-KeyName "my-key-pair" `
-SubnetId subnet-0abcdef1234567890 `
-InstanceMarketOptions $marketoption `
-CapacityReservationTarget_CapacityReservationId capacity_block_id
```

次の例では、インスタンスを UltraServer のキャパシティブロックリソースグループで起動します。

```
New-EC2Instance `
```

```
-ImageId ami-0abcdef1234567890 `
-InstanceType p6e-gb200.36xlarge `
-KeyName "my-key-pair" `
-SubnetId subnet-0abcdef1234567890 `
-InstanceMarketOptions $marketoption `
-CapacityReservationTarget_CapacityReservationResourceGroupArn "resource_group_arn"
```

関連リソース

- キャパシティブロックをターゲットとする起動テンプレートを作成するには、「the section called "起動テンプレート"」を参照してください。
- EC2 フリートを使用してキャパシティブロックでインスタンスを起動するには、「チュートリアル: キャパシティブロックにインスタンスを起動するように EC2 フリートを設定する」を参照してください。
- キャパシティブロックを使用して EKS マネージド型ノードグループを設定するには、「Amazon EKS ユーザーガイド」の「Capacity Blocks for ML を使用してマネージドノードグループを作成する」を参照してください。
- キャパシティブロックを使用して AWS ParallelCluster を設定するには、「ML on AWS ParallelCluster」を参照してください。

キャパシティブロックを表示する

キャパシティブロックを予約すると、AWS アカウントにキャパシティブロックの予約が表示されます。`start-date` と `end-date` を見ると、予約の開始日と終了日を確認できます。キャパシティブロック予約が始まるまで、利用可能なキャパシティは 0 と表示されます。キャパシティブロックで利用できるインスタンスの数は、タグキー `aws:ec2capacityreservation:incrementalRequestedQuantity` のタグ値を見ると確認できます。

キャパシティブロックの予約が始まると、予約状態は `scheduled` から `active` に変わります。Amazon EventBridge を通じて、キャパシティブロックが使用可能になったことを知らせるイベントが発行されます。詳細については「EventBridge を使用してキャパシティブロックをモニタリングする」を参照してください。

キャパシティブロックには、以下のような状態があります。

- `payment-pending` — 前払い料金の処理が完了していない。

- payment-failed — 前払い料金の処理が 12 時間以内に完了しなかった。キャパシティブロックが解除された。
- scheduled — 料金の処理は完了したが、キャパシティブロックの予約はまだ始まっていない。
- active - リザーブドキャパシティを使用できる。
- expired — キャパシティブロックの予約の有効期限が予約リクエストで指定された日時に自動的に切れた。リザーブドキャパシティーも使用できなくなります。

Console

キャパシティブロックを表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[キャパシティーの予約] を選択してください。
3. [キャパシティ予約の概要] ページに、すべてのキャパシティ予約リソースの詳細が記載されたリソーステーブルが表示されます。キャパシティブロック予約を検索するには、[キャパシティ予約 ID] の上にあるドロップダウンリストから[キャパシティブロック] を選択してください。表には、開始日、終了日、期間、状態など、キャパシティブロックに関する情報が表示されています。
4. キャパシティブロックの詳細は、表示するキャパシティブロックの予約 ID を選択すると、表示されます。[キャパシティ予約の詳細] ページには、予約のすべてのプロパティと、キャパシティブロックで使用中かつ使用可能なインスタンスの数が表示されています。

> **Note**
>
> キャパシティブロック予約が始まるまで、利用可能なキャパシティは 0 と表示されます。キャパシティブロックが開始されたときに利用できるインスタンスの数は、タグキー aws:ec2capacityreservation:incrementalRequestedQuantity の以下のタグ値を使用して確認できます。

AWS CLI

キャパシティブロックを表示するには

デフォルトでは、describe-capacity-reservations コマンドを使用すると、オンデマンドキャパシティ予約とキャパシティブロック予約の両方が一覧表示されます。キャパシティブロック予約のみを表示するには、タイプ capacity-block の予約をフィルタリングします。

```
aws ec2 describe-capacity-reservations \
    --filters Name=reservation-type,Values=capacity-block
```

PowerShell

キャパシティブロックを表示するには

Get-EC2CapacityReservation コマンドレットを使用します。デフォルトでは、オンデマンドキャパシティ予約とキャパシティブロック予約の両方が一覧表示されます。キャパシティブロック予約のみを表示するには、タイプ capacity-block の予約をフィルタリングします。

```
Get-EC2CapacityReservation `
    -Filter @{Name="reservation-type"; Values="capacity-block"}
```

キャパシティブロックを拡張

キャパシティブロックを使用するとワークロードのコンピューティングキャパシティを予約し、可用性と一貫性を確保できます。変化するニーズに対応するには、必要に応じて既存のキャパシティブロックの期間を延長できます。

キャパシティブロックを拡張するには、ステータスが active または scheduled であり、拡張機能が payment-pending でない必要があります。キャパシティブロックの期間は、期限が切れるまでの、1 時間以上 57 日以内で、延長するようにリクエストできます。キャパシティブロックは 1 日単位で合計最大 14 日間、7 日単位で合計最大 182 日間 (26 週間) まで延長できます。キャパシティブロックを拡張すると終了日が更新され、インスタンスが中断することなく継続的に実行できるようになります。

- キャパシティブロックに適用できる拡張機能の数に制限はありません
- キャパシティ予約 ID は、ブロックを拡張した後も変わりません。
- キャパシティブロックは、サポートするために十分なキャパシティがある場合に限って拡張できます。ただし、キャパシティは保証されません。

請求

キャパシティブロックサービスの料金は前払い制です。請求書が支払われるまで、延長機能は payment-pending に残ります。支払いが 12 時間以内、あるいはキャパシティブロックの終了がスケジュールされる最大 35 分前 (いずれか早い方) に処理できない場合、拡張機能は正常に処理され

ず、ステータスは payment-failed に変更されます。キャパシティブロックの予約は active のままになり、元の終了日に終了します。

支払いが正常に処理されると、キャパシティブロックの拡張機能ステータスが payment-succeeded に変更され、キャパシティブロック予約の終了日が新しい終了日に更新されます。拡張機能の詳細はコンソールの キャパシティブロック拡張機能の詳細 セクションで表示するか、DescribeCapacityBlockExtensionHistory コマンドを使用して表示することもできます。

キャパシティブロックを拡張

Console

キャパシティブロックを延長するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[キャパシティ予約] を選択してください。
3. [キャパシティ予約の概要] ページには、すべてのキャパシティ予約リソースの詳細が記載されたリソーステーブルが表示されます。拡張するキャパシティブロックの予約 ID を選択してください。
4. [アクション] ドロップダウンメニューで [キャパシティブロックを拡張] を選択してください。
5. [期間] で予約を延長する必要な日数を入力してください。
6. [キャパシティブロックを検索] を選択してください。
7. 要件を満たすキャパシティブロックがある場合、[おすすめのキャパシティブロック] にサービスが表示されます。他のキャパシティブロックサービスを表示するときは、検索条件を変更し、再度 [キャパシティブロックを検索] を選択してください。
8. 購入するキャパシティブロックのサービスが見つかったとき、[延長] を選択してください。
9. ポップアップウィンドウの [キャパシティブロックの拡張] に確認と入力したら、[拡張] を選択してください。

AWS CLI

キャパシティブロック延長を見つけるには

describe-capacity-block-extension-offerings コマンドを使用します。次の例では、指定した予約の 48 時間のキャパシティブロック延長を検索します。

```
aws ec2 describe-capacity-block-extension-offerings \
```

```
    --capacity-reservation-id cr-1234567890abcdefg \
    --capacity-block-extension-duration-hours 48
```

キャパシティブロックを延長するには

purchase-capacity-block-extension コマンドを使用します。前の例の出力からの延長提供 ID を指定します。

```
aws ec2 purchase-capacity-block-extension \
    --capacity-block-extension-offering-id cbe-0123456789abcdefg \
    --capacity-reservation-id cr-1234567890abcdefg
```

PowerShell

キャパシティブロック延長を見つけるには

Get-EC2CapacityBlockExtensionOffering コマンドレットを使用します。次の例では、指定した予約の 48 時間のキャパシティブロック延長を検索します。

```
Get-EC2CapacityBlockExtensionOffering `
    -CapacityReservationId cr-1234567890abcdefg `
    -CapacityBlockExtensionDurationHour 48
```

キャパシティブロックを延長するには

Invoke-EC2CapacityBlockExtension コマンドレットを使用します。前の例の出力からの延長提供 ID を指定します。

```
Invoke-EC2CapacityBlockExtension `
    -CapacityBlockExtensionOfferingId cbe-0123456789abcdefg `
    -CapacityReservationId cr-1234567890abcdefg
```

キャパシティブロックを共有する

キャパシティブロックを共有すると、キャパシティブロック所有者は Amazon EC2 キャパシティブロックを AWS Organization 内の他の AWS アカウントと共有できます。これにより、予約された GPU 容量をさまざまなチームやプロジェクトで最大限に活用して、キャパシティブロックを効率的に使用できます。

キャパシティブロックを所有する AWS アカウント (所有者) はキャパシティブロックを他の AWS アカウント (コンシューマー) と共有できます。所有者は、キャパシティブロックを AWS Organization

内の複数の AWS アカウント、AWS Organization 内の 1 つの組織単位、または AWS Organization 全体と共有できます。コンシューマーは、自分が所有するキャパシティブロックにインスタンスを起動する場合と同じように、共有されているキャパシティブロックにインスタンスを起動できます。

キャパシティブロック共有の前提条件

キャパシティブロックを共有するには、次の条件を満たす必要があります。

- キャパシティブロックを所有している必要があります – 共有されているキャパシティブロックを共有することはできません。
- キャパシティブロックの状態はアクティブまたはスケジュール済みである必要があります – 他の状態 (`expired` や `payment-pending` など) のキャパシティブロックは共有できません。
- AWS Organization 内での共有のみ – 所有者はキャパシティブロックを AWS Organization 内の複数の AWS アカウント、AWS Organization 内の 1 つの組織単位、または AWS Organization 全体と共有できます。
- UltraServer キャパシティブロックはサポートされていません – Amazon EC2 UltraServers のキャパシティブロックは共有できません。
- アカウントの適格性 - キャパシティブロック共有は、新規の AWS アカウント、および制限付き請求履歴のある AWS アカウントでは利用できません。

関連サービス

キャパシティブロック共有は AWS Resource Access Manager (AWS RAM) と統合します。AWS RAM は、AWS リソースを任意の AWS アカウントと共有したり、AWS Organizations 経由で共有したりするためのサービスです。AWS RAM を使用した リソース共有。これにより、自身が所有するリソースを共有できます。リソース共有では、共有対象のリソースと、共有先となるコンシューマーを指定します。コンシューマーには、個人の AWS アカウントや、AWS Organizations 内の組織単位または組織全体を指定できます。

AWS RAM の詳細については、AWS RAM ユーザーガイドを参照してください。

共有キャパシティブロックのアクセス許可

所有者のアクセス許可

キャパシティブロックの所有者は、キャパシティブロックの管理 (拡張、共有など) とそのキャパシティブロックに起動するインスタンスに引き続き責任を負います。所有者は、共有されているキャパシティブロックにコンシューマーが起動したインスタンスを変更することはできません。

コンシューマーのアクセス許可

コンシューマーは共有キャパシティにインスタンスを起動できます。コンシューマーは、それらのインスタンスを管理する責任があります。コンシューマーは、他のコンシューマーやキャパシティブロック所有者が所有するインスタンスを表示することや変更することはできません。コンシューマーが表示できるのは、共有キャパシティブロックの合計キャパシティと使用可能なキャパシティのみです。

キャパシティブロックを共有する

キャパシティブロックを共有するには、キャパシティブロックをリソース共有に追加する必要があります。リソース共有とは、AWS アカウント間で自身のリソースを共有するための AWS RAM リソースです。

AWS Organization 全体と共有されているリソース共有にキャパシティブロックを追加した場合、組織内のコンシューマーには共有キャパシティブロックへのアクセス許可が付与されます。

Console

所有しているキャパシティブロックを Amazon EC2 コンソール経由で共有する

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[キャパシティ予約] を選択してください。
3. 共有するキャパシティブロックを選択し、[アクション]、[予約を共有] の順に選択します。
4. キャパシティブロックの追加先となるリソース共有を選択し、[キャパシティ予約を共有] を選択します。

   共有されたキャパシティブロックにコンシューマーがアクセスできるようになるまで数分かかる場合があります。

新しいリソース共有にキャパシティブロックを追加する

まず、AWS RAM コンソールを使用してリソース共有を作成する必要があります。詳細については、「AWS RAM ユーザーガイド」の「Creating a resource share」を参照してください。

AWS CLI

自分が所有するキャパシティブロックを共有するには

create-resource-share および associate-resource-share コマンドを使用します。

```
aws ram create-resource-share \
    --name my-resource-share \
    --resource-arns arn:aws:ec2:us-east-2:123456789012:capacity-
reservation/cr-1234abcd56EXAMPLE
```

```
aws ram associate-resource-share \
    --resource-share-arn arn:aws:ram:us-east-2:123456789012:resource-share/7ab63972-
b505-7e2a-420d-6f5d3EXAMPLE \
    --resource-arns arn:aws:ec2:us-east-2:123456789012:capacity-
reservation/cr-1234abcd56EXAMPLE
```

PowerShell

自分が所有するキャパシティブロックを共有するには

New-RAMResourceShare コマンドレットと Connect-RAMResourceShare コマンドレットを使用します。

```
New-RAMResourceShare `
    -Name my-resource-share `
    -ResourceArn "arn:aws:ec2:us-east-2:123456789012:capacity-
reservation/cr-1234abcd56EXAMPLE"
```

```
Connect-RAMResourceShare `
    -ResourceShareArn "arn:aws:ram:us-east-2:123456789012:resource-share/7ab63972-
b505-7e2a-420d-6f5d3EXAMPLE" `
    -ResourceArn "arn:aws:ec2:us-east-2:123456789012:capacity-
reservation/cr-1234abcd56EXAMPLE"
```

キャパシティブロックは、所有権のステータスに関係なく、すべてのアカウントで先着順で動作します。キャパシティブロックを共有する場合、所有者の前にコンシューマーがインスタンスを起動すると、それらのインスタンスは、コンシューマーがインスタンスを終了するかキャパシティブロックの有効期限が切れる 30 分前までキャパシティを占有します。

キャパシティブロックの共有を停止する

キャパシティブロックの共有は、ブロックの有効期限の 30 分前までいつでも停止できます。

共有を停止した場合の結果:

- コンシューマーは、共有が解除されたキャパシティブロックで新しいインスタンスを起動できなくなります。
- 実行中のインスタンスは、コンシューマーによって終了されない限り、キャパシティブロックの有効期限の 30 分前まで実行され続けます。

Console

所有しているキャパシティ予約の共有を Amazon EC2 コンソール経由で停止する

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[キャパシティ予約] を選択してください。
3. キャパシティブロックを選択して、[共有] タブを選択します。
4. [共有] タブに、キャパシティブロックが追加されたリソース共有のリストが表示されます。キャパシティブロックを削除するリソース共有を選択します。
5. [リソース共有から削除] を選択します。

AWS CLI

自分が所有するキャパシティブロックの共有を停止するには

disassociate-resource-share コマンドを使用します。

```
aws ram disassociate-resource-share \
    --resource-share-arn arn:aws:ram:us-east-2:123456789012:resource-share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE \
    --resource-arns arn:aws:ec2:us-east-2:123456789012:capacity-reservation/cr-1234abcd56EXAMPLE
```

PowerShell

自分が所有するキャパシティブロックの共有を停止するには

Disconnect-RAMResourceShare コマンドレットを使用します。

```
Disconnect-RAMResourceShare `
    -ResourceShareArn "arn:aws:ram:us-east-2:123456789012:resource-share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE" `
```

```
    -ResourceArn "arn:aws:ec2:us-east-2:123456789012:capacity-
reservation/cr-1234abcd56EXAMPLE"
```

共有キャパシティブロックの使用状況をモニタリングする

キャパシティブロックの所有者は、共有キャパシティブロックを使用しているアカウントをモニタリングし、アカウントごとにインスタンスの使用状況を追跡できます。

AWS CLI

キャパシティブロックの使用状況をモニタリングするには

get-capacity-reservation-usage コマンドを使用します。

```
aws ec2 get-capacity-reservation-usage \
    --capacity-reservation-id cr-1234abcd56EXAMPLE
```

この API を使用すると、所有者は次のことを行うことができます。

- キャパシティブロックを現在使用しているアカウントを表示する。
- 各アカウントが実行しているインスタンスの数を確認する。

インスタンスの終了通知

キャパシティブロックの予約が終了する 40 分前になると、インスタンスが実行しているキャパシティブロックの所有者アカウントとコンシューマーアカウントに EventBridge イベントが送信され、予約で実行しているインスタンスが 10 分後に終了するが示されます。詳細については「EventBridge を使用してキャパシティブロックをモニタリングする」を参照してください。

キャパシティブロックの拡張

共有中、キャパシティブロックを拡張することができます。共有キャパシティブロックを拡張できるのは、所有者アカウントのみです。

キャパシティブロックが拡張されると、所有者またはコンシューマーによって起動された実行中のインスタンスは自動的に新しい有効期限を継承し、コンシューマーは新しい有効期限までインスタンスを中断することなく共有キャパシティを引き続き使用できます。

料金と請求

共有キャパシティブロックの請求は所有者に対して行われ、所有者はキャパシティブロックの購入時に前払いします。所有者は、キャパシティブロックで実行するインスタンスのオペレーティングシステムの料金も支払います。

コンシューマーに対しては、共有キャパシティブロックで実行するインスタンスのオペレーティングシステムの料金のみが課金されます。コンシューマーには、キャパシティブロックの予約そのものに対する課金は行われません。

UltraServer キャパシティブロックのリソースグループを作成する

AWS Resource Groups を使用して、UltraServer キャパシティブロックの論理コレクションを作成できます。リソースグループを作成したら、アカウントで所有している UltraServer キャパシティブロックを追加できます。UltraServer キャパシティブロックを追加すると、個々のキャパシティブロックではなくリソースグループに対してインスタンスの起動をターゲットにできます。リソースグループをターゲットとするインスタンスは、一致する属性と使用可能な容量を持つグループ内の任意の UltraServer キャパシティブロックと一致します。一致する属性と使用可能な容量を持つ UltraServer キャパシティーブロックがリソースグループにない場合、インスタンスの起動は失敗します。

UltraServer キャパシティブロックに実行中のインスタンスがあるときにリソースグループから削除された場合、それらのインスタンスはキャパシティブロックで引き続き実行されます。グループ内の UltraServer キャパシティブロックがインスタンスの実行中に終了すると、それらのインスタンスは終了します。

インスタンスキャパシティブロックをリソースグループに追加することはできません。

UltraServer キャパシティブロックのリソースグループを作成するには、次のいずれかの方法を使用します。

AWS CLI

UltraServer キャパシティブロックのリソースグループを作成するには

create-groupAWS CLI コマンドを使用し、`--configuration` に以下を指定します。

```
{
  "Configuration": [
    {
      "Type": "AWS::EC2::CapacityReservationPool",
```

```
      "Parameters": [
        {
          "Name": "instance-type",
          "Values": [
            "instance_type"
          ]
        },
        {
          "Name": "reservation-type",
          "Values": [
            "capacity-block"
          ]
        }
      ]
    },
    {
      "Type": "AWS::ResourceGroups::Generic",
      "Parameters": [
        {
          "Name": "allowed-resource-types",
          "Values": [
            "AWS::EC2::CapacityReservation"
          ]
        }
      ]
    }
  ]
}
```

PowerShell

UltraServer キャパシティブロックのリソースグループを作成するには

New-RGGroup コマンドレットを使用します。[-Configuration] で、以下を指定します。

```
{
  "Configuration": [
    {
      "Type": "AWS::EC2::CapacityReservationPool",
      "Parameters": [
        {
          "Name": "instance-type",
          "Values": [
            "instance_type"
```

```
          ]
        },
        {
          "Name": "reservation-type",
          "Values": [
            "capacity-block"
          ]
        }
      ]
    },
    {
      "Type": "AWS::ResourceGroups::Generic",
      "Parameters": [
        {
          "Name": "allowed-resource-types",
          "Values": [
            "AWS::EC2::CapacityReservation"
          ]
        }
      ]
    }
  ]
}
```

UltraServer キャパシティブロックのリソースグループを作成したら、次のいずれかの方法を使用して既存の UltraServer キャパシティブロックを追加します。

AWS CLI

UltraServer キャパシティブロックをリソースグループに追加するには

group-resources コマンドを使用します。--group には、作成したリソースグループの名前を指定します。--resource-arns には、追加する UltraServer キャパシティブロックの ARN を指定します。

```
aws resource-groups group-resources \
--group MyCRGroup \
--resource-arns CapacityReservationArn
```

PowerShell

UltraServer キャパシティブロックをリソースグループに追加するには

Add-RGResource コマンドレットを使用します。-Group には、作成したリソースグループの名前を指定します。-ResourceArn には、追加する UltraServer キャパシティブロックの ARN を指定します。

次の例では、指定したグループに 2 つのキャパシティ予約を追加します。

```
Add-RGResource `
-Group MyCRGroup `
-ResourceArn CapacityReservationArn
```

EventBridge を使用してキャパシティブロックをモニタリングする

キャパシティブロックの予約が始まると、Amazon EC2 は EventBridge を通じて、キャパシティが使用可能になったことを知らせるイベントを送信します。キャパシティブロック予約の終了 40 分前になると、予約で実行中のインスタンスが 10 分後に終了プロセスを開始することを知らせる、別のEventBridge イベントが手元に届きます。EventBridge イベントの詳細については、「Amazon EventBridge イベント」を参照してください。

キャパシティブロックに関して発生するイベントのイベント構造を以下に示します。

キャパシティブロックの配信

以下に示す例は、キャパシティブロックの配信のイベントです。

```
{
  "customer_event_id": "[Capacity Reservation Id]-delivered",
  "detail_type": "Capacity Block Reservation Delivered",
  "source": "aws.ec2",
  "account": "[Customer Account ID]",
  "time": "[Current time]",
  "resources": [
    "[ODCR ARN]"
  ],
  "detail": {
    "capacity-reservation-id": "[ODCR ID]",
    "end-date": "[ODCR End Date]"
  }
}
```

キャパシティブロックの有効期限切れの警告

以下に示す例は、キャパシティブロックの有効期限切れの警告のイベントです。

```
{
  "customer_event_id": "[Capacity Reservation Id]-approaching-expiry",
  "detail_type": "Capacity Block Reservation Expiration Warning",
  "source": "aws.ec2",
  "account": "[Customer Account ID]",
  "time": "[Current time]",
  "resources": [
    "[ODCR ARN]"
  ],
  "detail": {
    "capacity-reservation-id": "[ODCR ID]",
    "end-date": "[ODCR End Date]"
  }
}
```

キャパシティ予約インスタンスの中断に関する警告

次の例は、EC2 キャパシティ予約インスタンスの中断警告のイベントを示しています。

```
{
    "version": "0",
    "id": "12345678-1234-1234-1234-123456789012",
    "detail_type": "EC2 Capacity Reservation Instance Interruption Warning",
    "source": "aws.ec2",
    "account": "[Customer Account ID]",
    "time": "[Current time]",
    "region": "[Region]",
    "resources": [
        "[Instance ARN]"
    ],
    "detail": {
        "instance-id": "[Instance ID]",
        "instance-action": "terminate",
        "instance-termination-time": "[Current time]",
        "availability-zone-id": "[Availability Zone ID]",
        "instance-lifecycle": "capacity-block"
    }
}
```

AWS CloudTrail を使用したキャパシティブロック API コールのロギング

キャパシティブロックは、キャパシティブロックのユーザー、ロール、AWS サービスが実行したアクションを記録する AWS CloudTrail と連携しています。CloudTrail は、キャパシティブロックの API コールをイベントとしてキャプチャします。キャプチャされたコールには、キャパシティブロックコンソールからのコールと、キャパシティブロック API オペレーションへのコードコールが含まれています。証跡を作成する場合は、キャパシティブロックのイベントなど、Amazon S3 バケットへの CloudTrail イベントの継続的配信を有効にすることができます。証跡を設定しない場合でも、CloudTrail コンソールの [イベント履歴] で最新のイベントを表示できます。CloudTrail で収集された情報を使用して、キャパシティブロックに対するリクエスト、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を確認できます。

CloudTrail の詳細については、「AWS CloudTrail ユーザーガイド」を参照してください。

CloudTrail のキャパシティブロック情報

CloudTrail は、AWS アカウント を作成すると、その中で有効になります。キャパシティブロックでアクティビティが発生すると、そのアクティビティは [イベント履歴] で AWS のその他のサービスのイベントとともに CloudTrail イベントにレコードされます。最近のイベントは、AWS アカウント で表示、検索、ダウンロードできます。詳細については、 CloudTrail イベント履歴でのイベントの表示を参照してください。

イベント履歴のイベントなど、AWS アカウント のイベントの継続的な記録については、証跡を作成します。証跡により、CloudTrail はログファイルを Amazon S3 バケットに配信できます。デフォルトでは、コンソールで証跡を作成するときに、証跡がすべての AWS リージョン に適用されます。証跡は、AWS パーティションのすべてのリージョンからのイベントをログに記録し、指定した Amazon S3 バケットにログファイルを配信します。さらに、CloudTrail ログで収集したイベントデータをより詳細に分析し、それに基づいて対応するため、他の AWS サービスを構成できます。詳細については、次を参照してください:

- 追跡を作成するための概要
- CloudTrail がサポートされているサービスと統合
- 「CloudTrail の Amazon SNS 通知の設定」
- 複数のリージョンから CloudTrail ログファイルを受け取るおよび複数のアカウントから CloudTrail ログファイルを受け取る

すべてのキャパシティブロックアクションは CloudTrail によって ログ記録され、Amazon EC2 API リファレンスに記録されます。例えば、`CapacityBlockScheduled` と `CapacityBlockActive` の各アクションを呼び出すと、CloudTrail ログファイルにエントリが生成されます。

各イベントまたはログエントリには、誰がリクエストを生成したかという情報が含まれます。アイデンティティ情報は、以下を判別するのに役立ちます:

- リクエストが、ルート認証情報と AWS Identity and Access Management (IAM) ユーザー認証情報のどちらを使用して送信されたか。
- リクエストがロールまたはフェデレーションユーザーのテンポラリなセキュリティ認証情報を使用して行われたかどうか。
- リクエストが別の AWS サービスによって行われたかどうか。

詳細については、「CloudTrail userIdentity エレメント」を参照してください。

キャパシティブロックのログファイルエントリについて

「トレイル」は、指定した Amazon S3 バケットにイベントをログファイルとして配信するように設定できます。CloudTrail のログファイルは、単一か複数のログエントリを含みます。イベントは、任意の出典からの単一のリクエストを表し、リクエストされたアクション、アクションの日時、リクエストパラメータなどに関する情報が含まれます。CloudTrail ログファイルは、パブリック API 呼び出しの順序付けられたスタックトレースではないため、特定の順序では表示されません。

以下は、次の CloudTrail ログエントリの例です。

- TerminateCapacityBlocksInstances
- CapacityBlockPaymentFailed
- CapacityBlockScheduled
- CapacityBlockActive
- CapacityBlockFailed
- CapacityBlockExpired

> **Note**
>
> こちらの例では、データのプライバシーを保護するため一部のフィールドが削除されています。

TerminateCapacityBlocksInstances


```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "accountId": "123456789012",
    "invokedBy": "AWS Internal;"
  },
  "eventTime": "2023-10-02T00:06:08Z",
  "eventSource": "ec2.amazonaws.com",
  "eventName": "TerminateCapacityBlockInstances",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "203.0.113.25",
  "userAgent": "aws-cli/1.15.61 Python/2.7.10 Darwin/16.7.0 botocore/1.10.60",
  "requestParameters": null,
  "responseElements": null,
  "eventID": "a1b2c3d4-EXAMPLE",
  "readOnly": false,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::EC2::Instance",
      "ARN": "arn:aws:ec2:us-east-1:123456789012:instance/i-1234567890abcdef0"
    }
    {
      "accountId": "123456789012",
      "type": "AWS::EC2::Instance",
      "ARN": "arn:aws:ec2:us-east-1:123456789012:instance/i-0598c7d356eba48d7"
    }
  ],
  "eventType": "AwsServiceEvent",
  "recipientAccountId": "123456789012",
  "serviceEventDetails": {
      "capacityReservationId": "cr-12345678",
      }
}
```


CapacityBlockPaymentFailed


```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "accountId": "123456789012",
```

```
    "invokedBy": "AWS Internal;"
  },
  "eventTime": "2023-10-02T00:06:08Z",
  "eventSource": "ec2.amazonaws.com",
  "eventName": "CapacityBlockPaymentFailed",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "203.0.113.25",
  "userAgent": "aws-cli/1.15.61 Python/2.7.10 Darwin/16.7.0 botocore/1.10.60",
  "requestParameters": null,
  "responseElements": null,
  "eventID": "a1b2c3d4-EXAMPLE",
  "readOnly": false,
  "resources": [
    {
      "ARN": "arn:aws:ec2:us-east-1:123456789012:capacity-reservation/cr-12345678",
      "accountId": "123456789012",
      "type": "AWS::EC2::CapacityReservation"
    }
  ],
  "eventType": "AwsServiceEvent",
  "recipientAccountId": "123456789012",
  "serviceEventDetails": {
      "capacityReservationId": "cr-12345678",
      "capacityReservationState": "payment-failed"
      }
}
```

CapacityBlockScheduled

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "accountId": "123456789012",
    "invokedBy": "AWS Internal;"
  },
  "eventTime": "2023-10-02T00:06:08Z",
  "eventSource": "ec2.amazonaws.com",
  "eventName": "CapacityBlockScheduled",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "203.0.113.25",
  "userAgent": "aws-cli/1.15.61 Python/2.7.10 Darwin/16.7.0 botocore/1.10.60",
  "requestParameters": null,
  "responseElements": null,
```

```
  "eventID": "a1b2c3d4-EXAMPLE",
  "readOnly": false,
  "resources": [
    {
      "ARN": "arn:aws:ec2:us-east-1:123456789012:capacity-reservation/cr-12345678",
      "accountId": "123456789012",
      "type": "AWS::EC2::CapacityReservation"
    }
  ],
  "eventType": "AwsServiceEvent",
  "recipientAccountId": "123456789012",
  "serviceEventDetails": {
      "capacityReservationId": "cr-12345678",
      "capacityReservationState": "scheduled"
      }
}
```

CapacityBlockActive

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "accountId": "123456789012",
    "invokedBy": "AWS Internal;"
  },
  "eventTime": "2023-10-02T00:06:08Z",
  "eventSource": "ec2.amazonaws.com",
  "eventName": "CapacityBlockActive",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "203.0.113.25",
  "userAgent": "aws-cli/1.15.61 Python/2.7.10 Darwin/16.7.0 botocore/1.10.60",
  "requestParameters": null,
  "responseElements": null,
  "eventID": "a1b2c3d4-EXAMPLE",
  "readOnly": false,
  "resources": [
    {
      "ARN": "arn:aws:ec2:us-east-1:123456789012:capacity-reservation/cr-12345678",
      "accountId": "123456789012",
      "type": "AWS::EC2::CapacityReservation"
    }
  ],
  "eventType": "AwsServiceEvent",
```

```
  "recipientAccountId": "123456789012",
  "serviceEventDetails": {
      "capacityReservationId": "cr-12345678",
      "capacityReservationState": "active"
      }
 }
```

CapacityBlockFailed

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "accountId": "123456789012",
    "invokedBy": "AWS Internal;"
  },
  "eventTime": "2023-10-02T00:06:08Z",
  "eventSource": "ec2.amazonaws.com",
  "eventName": "CapacityBlockFailed",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "203.0.113.25",
  "userAgent": "aws-cli/1.15.61 Python/2.7.10 Darwin/16.7.0 botocore/1.10.60",
  "requestParameters": null,
  "responseElements": null,
  "eventID": "a1b2c3d4-EXAMPLE",
  "readOnly": false,
  "resources": [
    {
      "ARN": "arn:aws:ec2:us-east-1:123456789012:capacity-reservation/cr-12345678",
      "accountId": "123456789012",
      "type": "AWS::EC2::CapacityReservation"
    }
  ],
  "eventType": "AwsServiceEvent",
  "recipientAccountId": "123456789012",
  "serviceEventDetails": {
      "capacityReservationId": "cr-12345678",
      "capacityReservationState": "failed"
      }
 }
```

CapacityBlockExpired

```
{
```

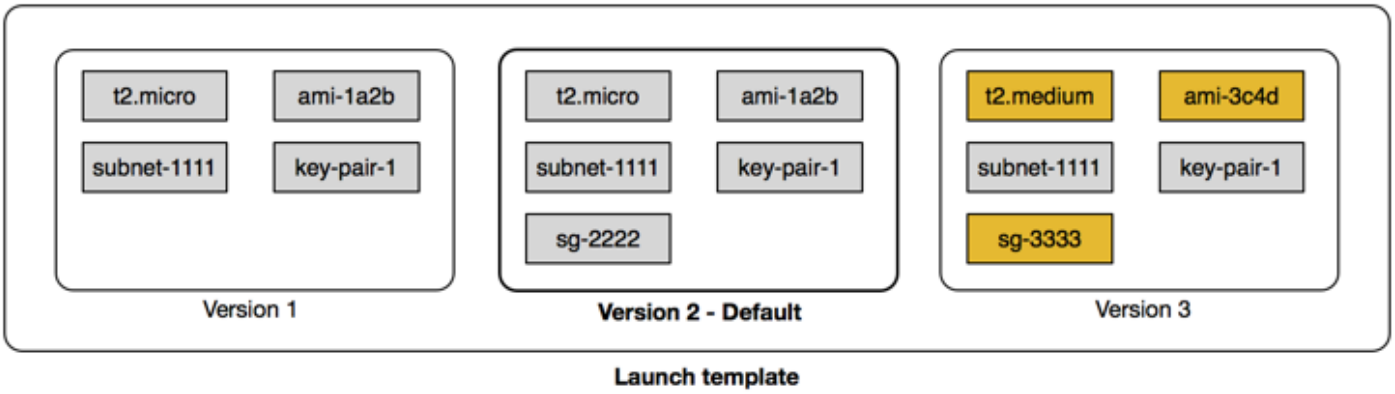
```
  "eventVersion": "1.05",
  "userIdentity": {
    "accountId": "123456789012",
    "invokedBy": "AWS Internal;"
  },
  "eventTime": "2023-10-02T00:06:08Z",
  "eventSource": "ec2.amazonaws.com",
  "eventName": "CapacityBlockExpired",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "203.0.113.25",
  "userAgent": "aws-cli/1.15.61 Python/2.7.10 Darwin/16.7.0 botocore/1.10.60",
  "requestParameters": null,
  "responseElements": null,
  "eventID": "a1b2c3d4-EXAMPLE",
  "readOnly": false,
  "resources": [
    {
      "ARN": "arn:aws:ec2:us-east-1:123456789012:capacity-reservation/cr-12345678",
      "accountId": "123456789012",
      "type": "AWS::EC2::CapacityReservation"
    }
  ],
  "eventType": "AwsServiceEvent",
  "recipientAccountId": "123456789012",
  "serviceEventDetails": {
      "capacityReservationId": "cr-12345678",
      "capacityReservationState": "expired"
      }
}
```


Amazon EC2 起動テンプレートにインスタンス起動パラメータを保存する

Amazon EC2 起動テンプレートを使用してインスタンス起動パラメータを保存すると、Amazon EC2 インスタンスを起動するたびにパラメータを指定する必要がなくなります。例えば、AMI ID やインスタンスタイプ、通常インスタンスの起動に使用しているネットワーク設定を保存する起動テンプレートを作成することができます。Amazon EC2 コンソール、AWS SDK、コマンドラインツールのいずれかを使用してインスタンスを起動するときは、パラメータを再度入力する代わりに、起動テンプレートを指定できます。

各起動テンプレートについて、1 つ以上の番号付きの起動テンプレートのバージョンを作成できます。各バージョンに異なる起動パラメータを指定できます。起動テンプレートからインスタンスを起動する際、起動テンプレートのいずれかのバージョンを使用できます。バージョンを指定しない場合は、デフォルトバージョンが使用されます。いずれかの起動テンプレートをデフォルトバージョンとして設定できます — デフォルトでは、起動テンプレートの最初のバージョンです。

以下の図は、3 つのバージョンの起動テンプレートを示しています。最初のバージョンでは、インスタンスの起動に使用するインスタンスタイプ、AMI ID、サブネット、およびキーペアが指定されています。2 番目のバージョンは最初のバージョンに基づいており、インスタンスのセキュリティグループも指定しています。3 番目のバージョンは、パラメータの一部に異なる値を使用しています。バージョン 2 がデフォルトバージョンとして設定されています。この起動テンプレートからインスタンスを起動すると、他のバージョンを指定しない限りバージョン 2 の起動パラメータが使用されます。



内容

- Amazon EC2 起動テンプレートの制限
- Amazon EC2 起動テンプレートに必要な IAM アクセス許可
- Amazon EC2 起動テンプレートを使用して Amazon EC2 インスタンスの起動を制御する
- Amazon EC2 起動テンプレートを作成する
- 起動テンプレートの変更 (起動テンプレートのバージョンの管理)
- 起動テンプレートまたは起動テンプレートのバージョンを削除する

Amazon EC2 起動テンプレートの制限

起動テンプレートおよび起動テンプレートのバージョンには次の制限が適用されます。

- クォータ – 起動テンプレートのクォータと起動テンプレートバージョンのクォータを確認するには、[Service Quotas] コンソールを開くか、AWS CLI コマンドの list-service-quotas を使用します。各 AWS アカウントでは、1 つのリージョンあたり最大で 5,000 の起動テンプレート、1 つの起動テンプレートあたり最大で 10,000 のバージョンを起動します。アカウントには、作成してからの期間や使用履歴に基づいて異なるクォータが設定されている場合があります。
- パラメータはオプション - 起動テンプレートのパラメータはオプションです。ただし、インスタンスの起動リクエストに必要なパラメータがすべて含まれている必要があります。例えば、起動テンプレートに AMI ID が含まれていない場合、この起動テンプレートでインスタンスを起動するときに AMI ID を指定する必要があります。
- パラメータは未検証 – 起動テンプレートパラメータは、起動テンプレート作成の際には完全には検証されていません。間違った値を指定したり、サポートされていないパラメータの組み合わせを使用したりすると、この起動テンプレートを使用してインスタンスを起動できなくなります。問題を回避するには、正しい値を指定し、サポートされているパラメータの組み合わせを使用してください。例えば、プレイスメントグループ内でインスタンスを起動するには、サポートされているインスタンスタイプを指定する必要があります。
- タグ – 起動テンプレートにはタグ付けできますが、起動テンプレートのバージョンにはタグ付けできません。
- 変更不可能 – 起動テンプレートは変更不可能です。起動テンプレートを変更するには、起動テンプレートの新しいバージョンを作成する必要があります。
- バージョン番号 – 起動テンプレートのバージョンには、作成された順序で番号が付けられます。起動テンプレートのバージョンを作成する場合、自分でバージョン番号を指定することはできません。

Amazon EC2 起動テンプレートに必要な IAM アクセス許可

IAM アクセス許可を使用して、ユーザーが起動テンプレートまたは起動テンプレートのバージョンをリスト、表示、作成、または削除できるかどうかを制御できます。

> **Important**
>
> リソースレベルのアクセス許可を使用して、ユーザーが起動テンプレートまたは起動テンプレートのバージョンを作成するときに起動テンプレートで指定できるリソースを制限することはできません。したがって、信頼できる管理者のみに起動テンプレートと起動テンプレートのバージョンを作成するアクセス許可が与えられるようにしてください。

起動テンプレートを使用するユーザーには、起動テンプレートで指定されたリソースの作成とアクセスに必要なアクセス許可を付与しなければなりません。例えば、次のようになります。

- 共有のプライベート Amazon マシンイメージ (AMI) からインスタンスを起動するには、ユーザーに AMI の起動許可が必要です。
- 既存のスナップショットからタグ付きの EBS ボリュームを作成するには、ユーザーはスナップショットへの読み取りのアクセス許可と、ボリュームを作成してタグ付けを行うためのアクセス許可が必要です。

内容

- ec2:CreateLaunchTemplate
- ec2:DescribeLaunchTemplates
- ec2:DescribeLaunchTemplateVersions
- ec2:DeleteLaunchTemplate
- バージョニングアクセス許可の制御
- 起動テンプレートのタグへのアクセスを制御する

ec2:CreateLaunchTemplate

コンソールで、または API を使用して起動テンプレートを作成するには、プリンシパルが IAM ポリシーで `ec2:CreateLaunchTemplate` アクセス許可を持っている必要があります。可能な限り、タグを使用してアカウントで起動テンプレートへのアクセスを制御できるようにします。

例えば、次の IAM ポリシーステートメントは、テンプレートが指定されたタグ (*purpose*=*testing*) を使用している場合にのみ、プリンシパルに起動テンプレートを作成する許可を付与します。

```
{
    "Sid": "IAMPolicyForCreatingTaggedLaunchTemplates",
    "Action": "ec2:CreateLaunchTemplate",
    "Effect": "Allow",
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceTag/purpose": "testing"
        }
    }
```

```
}
```

起動テンプレートを作成するプリンシパルには、次のような関連するアクセス許可が必要な場合があります。

- ec2:CreateTags — CreateLaunchTemplate 操作時に起動テンプレートにタグを追加するには、CreateLaunchTemplate の呼び出し元が IAM ポリシーで ec2:CreateTags アクセス許可を持っている必要があります。
- ec2:RunInstances – 作成した起動テンプレートから EC2 インスタンスを起動するには、プリンシパルは IAM ポリシーで ec2:RunInstances アクセス許可も持っている必要があります。

タグを適用するリソース作成アクションでは、ユーザーが ec2:CreateTags アクセス許可を持っている必要があります。次の IAM ポリシーステートメントは、ec2:CreateAction 条件キーを使用して、ユーザーが CreateLaunchTemplate のコンテキストでのみタグを使用できるようにしています。ユーザーは、既存の起動テンプレートにも他のリソースにもタグ付けできません。詳細については、「Amazon EC2 リソース作成時にタグ付けするアクセス許可の付与」を参照してください。

```
{
    "Sid": "IAMPolicyForTaggingLaunchTemplatesOnCreation",
    "Action": "ec2:CreateTags",
    "Effect": "Allow",
    "Resource": "arn:aws:ec2:us-east-1:111122223333:launch-template/*",
    "Condition": {
        "StringEquals": {
            "ec2:CreateAction": "CreateLaunchTemplate"
        }
    }
}
```

起動テンプレートを作成した IAM ユーザーに、作成した起動テンプレートを使用するアクセス許可が自動で付与されることはありません。他のプリンシパルと同様に、起動テンプレートの作成者は、IAM ポリシーを使用してアクセス許可を取得する必要があります。IAM ユーザーが起動テンプレートから EC2 インスタンスを起動する場合は、ec2:RunInstances アクセス許可が必要です。このアクセス許可を付与するときに、ユーザーが特定のタグまたは特定の ID を含む起動テンプレートのみを使用できるように指定できます。また、RunInstances 呼び出しに対するリソースレベルのアクセス許可を指定することで、起動テンプレートを使用するすべてのユーザーがインスタンスの起動時に参照および使用できる AMI やその他のリソースを制御できます。エンドポイントポリシーの例については、「起動テンプレート」を参照してください。

ec2:DescribeLaunchTemplates

アカウントの起動テンプレートをリストおよび表示するには、プリンシパルが IAM ポリシーで ec2:DescribeLaunchTemplates アクセス許可を持っている必要があります。Describe アクションはリソースレベルのアクセス許可をサポートしていないため、条件なしで指定する必要があります。また、ポリシーのリソース要素の値は "*" である必要があります。

例えば、次の IAM ポリシーステートメントでは、アカウントのすべての起動テンプレートをリストおよび表示する許可をプリンシパルに付与します。

```
{
    "Sid": "IAMPolicyForDescribingLaunchTemplates",
    "Action": "ec2:DescribeLaunchTemplates",
    "Effect": "Allow",
    "Resource": "*"
}
```

ec2:DescribeLaunchTemplateVersions

起動テンプレートをリストおよび表示するプリンシパルは、起動テンプレートを構成する属性セット全体を取得するための ec2:DescribeLaunchTemplateVersions アクセス許可も持つようにしてください。

アカウントの起動テンプレートのバージョンをリストおよび表示するには、プリンシパルが IAM ポリシーで ec2:DescribeLaunchTemplateVersions アクセス許可を持っている必要があります。Describe アクションはリソースレベルのアクセス許可をサポートしていないため、条件なしで指定する必要があります。また、ポリシーのリソース要素の値は "*" である必要があります。

例えば、次の IAM ポリシーステートメントでは、アカウントにおけるすべての起動テンプレートのバージョンをリストおよび表示する許可をプリンシパルに付与します。

```
{
    "Sid": "IAMPolicyForDescribingLaunchTemplateVersions",
    "Effect": "Allow",
    "Action": "ec2:DescribeLaunchTemplateVersions",
    "Resource": "*"
}
```

ec2:DeleteLaunchTemplate

> **Important**
>
> プリンシパルにリソースを削除する許可を与えるときは、注意する必要があります。起動テンプレートを削除すると、起動テンプレートに依存する AWS リソースに障害が発生する可能性があります。

起動テンプレートを削除するには、プリンシパルが IAM ポリシーで `ec2:DeleteLaunchTemplate` アクセス許可を持っている必要があります。可能な限り、タグベースの条件キーを使用してアクセス許可を制限します。

例えば、次の IAM ポリシーステートメントは、テンプレートが指定されたタグ (*purpose*=*testing*) がある場合にのみ、プリンシパルに起動テンプレートを削除する許可を付与します。

```
{
    "Sid": "IAMPolicyForDeletingLaunchTemplates",
    "Action": "ec2:DeleteLaunchTemplate",
    "Effect": "Allow",
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceTag/purpose": "testing"
        }
    }
}
```

または、ARN を使用して IAM ポリシーが適用される起動テンプレートを指定することもできます。

起動テンプレートには次の ARN が含まれます。

```
"Resource": "arn:aws:ec2:us-east-1:111122223333:launch-template/lt-09477bcd97b0d310e"
```

複数の ARN を一覧で囲んで指定することも、Condition 要素を使用せず Resource 値に "*" を指定して、プリンシパルがアカウントの任意の起動テンプレートを削除できるようにすることもできます。

バージョニングアクセス許可の制御

信頼できる管理者には、以下の例のように IAM ポリシーを使用して、起動テンプレートのバージョンを作成および削除したり、起動テンプレートのデフォルトバージョンを変更したりするためのアクセス許可を付与できます。

Important

起動テンプレートのバージョンを作成したり、起動テンプレートを変更したりするアクセス許可をプリンシパルに付与する場合は、注意が必要です。

- 起動テンプレートのバージョンを作成すると、Amazon EC2 がユーザーに代わって `Latest` バージョンでインスタンスを起動できるようにするすべての AWS リソースに影響します。
- 起動テンプレートを変更すると、どのバージョンが `Default` になるかを変更できます。したがって、Amazon EC2 がこの変更済みバージョンを使用してユーザーに代わってインスタンスを起動できるようにするすべての AWS リソースに影響します。

また、EC2 フリートやスポットフリートなど、`Latest` または `Default` の起動テンプレートバージョンとやり取りする AWS リソースの取り扱い方にも注意が必要です。「`Latest`」または「`Default`」に別の起動テンプレートバージョンが使用されている場合、Amazon EC2 は、フリートの目標容量を満たすために新しいインスタンスを起動する際に、完了すべきアクションのユーザーアクセス許可を再確認することはありません。これは、AWS リソースとユーザーのやり取りがないためです。`CreateLaunchTemplateVersion` API と `ModifyLaunchTemplate` API を呼び出すアクセス許可をユーザーに付与すると、インスタンスプロファイル (IAM ロールのコンテナ) を含む別の起動テンプレートバージョンをフリートに指定する場合、ユーザーに `iam:PassRole` アクセス許可も効果的に付与できます。つまり、場合によっては `iam:PassRole` アクセス許可がなくても、起動テンプレートを更新して IAM ロールをインスタンスに渡すことができます。このリスクは、起動テンプレートバージョンを作成および管理できるユーザーにアクセス許可を付与する際に注意することで管理できます。

ec2:CreateLaunchTemplateVersion

起動テンプレートの新しいバージョンを作成するには、プリンシパルが IAM ポリシーで起動テンプレートに対する `ec2:CreateLaunchTemplateVersion` アクセス許可を持っている必要があります。

例えば、次の IAM ポリシーステートメントは、バージョンが指定されたタグ (*environment*=*production*) を使用している場合にのみ、プリンシパルに起動テンプレートのバージョンを作成する許可を付与します。あるいは、1 つまたは複数の起動テンプレートの ARN を指定することも、Condition 要素を使用せず Resource 値に "*" を指定して、プリンシパルがアカウントにおける任意の起動テンプレートのバージョンを作成できるようにすることもできます。

```
{
    "Sid": "IAMPolicyForCreatingLaunchTemplateVersions",
    "Action": "ec2:CreateLaunchTemplateVersion",
    "Effect": "Allow",
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceTag/environment": "production"
        }
    }
}
```

ec2:DeleteLaunchTemplateVersion

> **Important**
>
> プリンシパルにリソースを削除する権限を与えるときは、いつものように注意する必要があります。起動テンプレートのバージョンを削除すると、起動テンプレートのバージョンに依存する AWS リソースに障害が発生する可能性があります。

起動テンプレートのバージョンを削除するには、プリンシパルが IAM ポリシーで起動テンプレートに対する ec2:DeleteLaunchTemplateVersion アクセス許可を持っている必要があります。

例えば、次の IAM ポリシーステートメントは、バージョンが指定されたタグ (*environment*=*production*) を使用している場合にのみ、プリンシパルに起動テンプレートのバージョンを削除する許可を付与します。あるいは、1 つまたは複数の起動テンプレートの ARN を指定することも、Condition 要素を使用せず Resource 値に "*" を指定して、プリンシパルがアカウントにおける任意の起動テンプレートのバージョンを削除できるようにすることもできます。

```
{
    "Sid": "IAMPolicyForDeletingLaunchTemplateVersions",
    "Action": "ec2:DeleteLaunchTemplateVersion",
    "Effect": "Allow",
```

```
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceTag/environment": "production"
        }
    }
}
```

ec2:ModifyLaunchTemplate

起動テンプレートに関連付けられている Default バージョンを変更するには、プリンシパルが IAM ポリシーで起動テンプレートに対する ec2:ModifyLaunchTemplate アクセス許可を持っている必要があります。

例えば、次の IAM ポリシーステートメントは、起動テンプレートが指定されたタグ (*environment*=*production*) を使用している場合にのみ、プリンシパルに起動テンプレートを変更する許可を付与します。あるいは、1 つまたは複数の起動テンプレートの ARN を指定することも、Condition 要素を使用せず Resource 値に "*" を指定して、プリンシパルがアカウントにおける任意の起動テンプレートのバージョンを変更できるようにすることもできます。

```
{
    "Sid": "IAMPolicyForModifyingLaunchTemplates",
    "Action": "ec2:ModifyLaunchTemplate",
    "Effect": "Allow",
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceTag/environment": "production"
        }
    }
}
```

起動テンプレートのタグへのアクセスを制御する

リソースが起動テンプレートの場合、条件キーを使用してタグ付け許可を制限できます。例えば、次の IAM ポリシーでは、指定されたアカウントとリージョンの起動テンプレートから *temporary* キーを持つタグのみを削除できます。

```
{
    "Sid": "IAMPolicyForDeletingTagsOnLaunchTemplates",
    "Action": "ec2:DeleteTags",
    "Effect": "Allow",
```

```
    "Resource": "arn:aws:ec2:us-east-1:111122223333:launch-template/*",
    "Condition": {
        "ForAllValues:StringEquals": {
            "aws:TagKeys": ["temporary"]
        }
    }
}
```

Amazon EC2 リソースに適用できるタグキーとタグ値を制御するのに使用できる条件キーの詳細については、「特定のタグへのアクセスの制御」を参照してください。

Amazon EC2 起動テンプレートを使用して Amazon EC2 インスタンスの起動を制御する

ユーザーが起動テンプレートを使用する場合、インスタンスのみを起動できるようにして、さらに特定の起動テンプレートのみを使用できるように指定することで、Amazon EC2 インスタンスの設定を制御できます。また、起動テンプレートや起動テンプレートのバージョンを作成、変更、記述、削除できる人を制御することもできます。

起動テンプレートを使用した起動パラメータの制御

起動テンプレートには、起動時にインスタンスを設定するためのパラメータのすべてまたは一部を含めることができます。ただし、起動テンプレートを使用してインスタンスを起動するときは、起動テンプレートで指定されたパラメータを上書きできます。または、起動テンプレートにない追加のパラメータを指定できます。

> **Note**
>
> 起動時に起動テンプレートパラメータを削除することはできません (例えば、パラメータに null 値を指定することはできません)。パラメータを削除するには、起動テンプレートの新しいバージョンをパラメータなしで作成し、そのバージョンを使用してインスタンスを起動します。

インスタンスを起動するには、ユーザーは `ec2:RunInstances` アクションを使用するためのアクセス許可が必要です。また、ユーザーは、インスタンスに作成または関連付けられたリソースを作成または使用する許可が必要です。`ec2:RunInstances` アクションのリソースレベルのアクセス権限を使用して、ユーザーが指定できる起動パラメータを管理できます。または、起動テンプレートを使用してインスタンスを起動するアクセス権限をユーザーに付与することもできます。これによ

り、IAM ポリシーではなくむしろ起動テンプレートで起動パラメータを管理できるようになり、インスタンス起動の権限を付与するための手段として起動テンプレートを使用できます。例えば、ユーザーが起動テンプレートを使用してのみインスタンスを起動できるようにして、さらに特定の起動テンプレートのみを使用できるように指定することができます。また、ユーザーが起動テンプレートで上書きする起動パラメータを制御することもできます。エンドポイントポリシーの例については、起動テンプレートを参照してください。

起動テンプレートの使用の管理

デフォルトでは、 ユーザーには起動テンプレートを使用するためのアクセス許可がありません。起動テンプレートと起動テンプレートバージョンの作成、変更、記述、および削除のアクセス許可をユーザーに付与するポリシーを作成できます。一部の起動テンプレートアクションにリソースレベルのアクセス許可を適用して、ユーザーがこれらのアクションに対する特定のリソースを使用する機能を制御することもできます。詳細については、例: 起動テンプレートの使用のポリシー例を参照してください。

ユーザーに `ec2:CreateLaunchTemplate` および `ec2:CreateLaunchTemplateVersion` のアクションを使用するアクセス許可を付与するには注意が必要です。リソースレベルのアクセス許可を使用して、ユーザーが起動テンプレートで指定できるリソースを制御することはできません。インスタンス起動のために使用されるリソースを制限するには、起動テンプレートと起動テンプレートのバージョンを作成するアクセス許可を、適切な管理者のみに付与していることを確認してください。

EC2 フリートまたはスポットフリートで起動テンプレートを使用する際の重要なセキュリティ上の懸念

> **Note**
>
> スポットフリートは、計画された投資なしでレガシー API を使用します。代わりに EC2 フリートまたは Auto Scaling グループを使用することをお勧めします。詳細については、「フリートにどの方法を使用するのが最適か」を参照してください。

起動テンプレートを使用するには、起動テンプレートおよび起動テンプレートのバージョンを作成、変更、記述、および削除するアクセス許可をユーザーに付与する必要があります。`ec2:CreateLaunchTemplate` および `ec2:CreateLaunchTemplateVersion` アクションへのアクセスを制御することで、起動テンプレートと起動テンプレートのバージョンをどのユーザーが作成できるかを制御できます。`ec2:ModifyLaunchTemplate` アクションへのアクセスを制御することで、起動テンプレートを変更できるユーザーを管理することもできます。

> **Important**
>
> EC2 フリートまたはスポットフリートが「最新」または「デフォルト」の起動テンプレートバージョンを使用するように設定されている場合、後でそれらが変更されて起動テンプレートの別のバージョンを指すようになっても、そのフリートでは認識されません。「最新」または「デフォルト」に別の起動テンプレートバージョンが使用されている場合、フリートの目標容量を満たすために新しいインスタンスを起動する際に、Amazon EC2 により完了すべきアクションのアクセス許可が再確認されることはありません。これは、起動テンプレートバージョン (特にユーザーがデフォルトの起動テンプレートバージョンを変更できる `ec2:ModifyLaunchTemplate` アクション) を作成および管理できるユーザーにアクセス許可を付与する際の重要な考慮事項です。

起動テンプレート API の EC2 アクションを使用するアクセス許可をユーザーに付与すると、インスタンスプロファイル (IAM ロールのコンテナ) を含む別の起動テンプレートバージョンを指すように EC2 フリートまたはスポットフリートを作成および更新する場合でも、ユーザーに `iam:PassRole` アクセス許可が効果的に付与されます。つまり、場合によっては `iam:PassRole` アクセス許可がなくても、起動テンプレートを更新して IAM ロールをインスタンスに渡すことができます。詳細および IAM ポリシーの例については、「IAM ユーザーガイド」の「Amazon EC2 インスタンスで実行されるアプリケーションに IAM ロールを使用してアクセス許可を付与する」を参照してください。

詳細については、「起動テンプレートの使用の管理」および「例: 起動テンプレートの使用」を参照してください。

Amazon EC2 起動テンプレートを作成する

Amazon EC2 起動テンプレートを作成するには、インスタンス設定パラメータに独自の値を指定するか、既存の起動テンプレートまたは Amazon EC2 インスタンスから値を取得します。

起動テンプレートのすべてのパラメータに値を指定する必要はありません。起動テンプレートを作成するには、インスタンス設定パラメータを 1 つだけ指定する必要があります。指定しないパラメータを指定するには、コンソールの使用時に [起動テンプレートに含めない] を選択してください。コマンドラインツールを使用する場合は、起動テンプレートで指定しないことを示すパラメータを含めないでください。

起動テンプレートで AMI を指定する場合は、AMI を選択するか、インスタンスの起動時に AMI を指す Systems Manager パラメータを指定できます。

起動テンプレートを使用してインスタンスを起動すると、起動テンプレートで指定された値を使用して、対応するインスタンスパラメータを設定します。起動テンプレートで値が指定されていない場合は、対応するインスタンスパラメータのデフォルト値が使用されます。

タスク

- パラメータを指定して起動テンプレートを作成する
- 既存の起動テンプレートからの起動テンプレートの作成
- インスタンスからの起動テンプレートの作成
- AMI ID のかわりに Systems Manager パラメータを使用する

パラメータを指定して起動テンプレートを作成する

起動テンプレートを作成する際には、そのテンプレートの名と、少なくとも 1 つのインスタンス設定パラメーターを指定する必要があります。

各パラメータの説明については、「Amazon EC2 インスタンス設定パラメータのリファレンス」を参照してください。

Console

起動テンプレートを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[起動テンプレート]、[起動テンプレートの作成] の順に選択してください。
3. [起動テンプレートの名前と説明] で、次の操作を行います。
   a. [起動テンプレート名] に、起動テンプレートのわかりやすい名前を入力してください。
   b. [Template version description] (テンプレートバージョンの説明) に、起動テンプレートバージョンの短い説明を入力してください。
   c. 作成時に起動テンプレートにタグを付けるには、[テンプレートタグ] を展開し、[新しいタグの追加] を選択して、タグのキーと値のペアを入力してください。追加するタグごとに [新しいタグを追加] を選択してください。

> **Note**
>
> インスタンスの起動時に作成されるリソースにタグを付けるには、[Resource tags] (リソースタグ) のタグを指定する必要があります。詳細については、この手順のステップ 9 を参照してください。

4. [アプリケーションイメージと OS イメージ (Amazon マシンイメージ)] では、[起動テンプレートに含めない] を選択したままにするか、インスタンスのオペレーティングシステム (OS) を選択してから AMI を選択できます。または、AMI を指定する代わりに Systems Manager パラメータを指定することもできます。詳細については、「AMI ID のかわりに Systems Manager パラメータを使用する」を参照してください。

   AMI は、インスタンスの起動に必要なオペレーティングシステムとソフトウェアを含むテンプレートです。

5. [インスタンスタイプ] で、[起動テンプレートに含めない] を選択したままにするか、インスタンスタイプを選択するか、インスタンス属性を指定して Amazon EC2 がそれらの属性でインスタンスタイプを識別できるようにすることができます。

   > **Note**
   >
   > インスタンス属性の指定がサポートされるのは、Auto Scaling グループ、EC2 フリート、およびスポットフリートによってインスタンスを起動するために起動テンプレートが使用される場合のみです。詳細については「Create mixed instances group using attribute-based instance type selection」と「EC2 フリートまたはスポットフリートのインスタンスタイプを選択するための属性を指定する」を参照してください。
   > インスタンス起動ウィザードまたは RunInstances API で起動テンプレートを使用する予定がある場合は、インスタンスタイプ属性を指定できません。

   インスタンスタイプによって、インスタンスに使用されるホストコンピュータのハードウェア構成 (CPU、メモリ、ストレージ、ネットワークキャパシティ) とサイズが決まります。

   どのインスタンスタイプを選択するかわからない場合は、次の操作を実行できます。

- [インスタンスタイプの比較] を選択して、vCPU の数、アーキテクチャ、メモリ量 (GiB)、ストレージ量 (GB)、ストレージタイプ、ネットワークパフォーマンスの属性によってさまざまなインスタンスタイプを比較します。
- [アドバイスの取得] を選択して、EC2 インスタンスタイプファインダーからインスタンスタイプに関するガイダンスやアドバイスを取得します。詳細については、「EC2 インスタンスタイプファインダーからレコメンデーションを取得する」を参照してください。

> **Note**
>
> アカウントの作成日に応じて、Amazon EC2 を無料利用枠で使用できる場合があります。
> 2025 年 7 月 15 日より前に AWS アカウントを作成していて、12 か月が経過していない場合は、[t2.micro] インスタンスタイプ ([t2.micro] が利用できないリージョンでは [t3.micro] インスタンスタイプ) を選択することで、Amazon EC2 の無料利用枠を使用できます。[t3.micro] インスタンスを起動すると、デフォルトで [Unlimited] モードになり、CPU 使用率に基づいて追加料金が発生する可能性があることに注意してください。インスタンスタイプを無料利用枠で使用できる場合、[無料利用枠の対象] とラベル付けされます。
> 2025 年 7 月 15 日以降に AWS アカウント を作成した場合、t3.micro、t3.small、t4g.micro、t4g.small、c7i-flex.large、m7i-flex.large インスタンスタイプを 6 か月間、またはクレジットを使い切るまで使用できます。
> 詳細については、「2025 年 7 月 15 日より前とそれ以降の無料利用枠の特典」を参照してください。

6. [キーペア (ログイン)] の [キーペア名] で、[起動テンプレートに含めない] を選択したままにするか、既存のキーペアを選択するか、新しいキーペアを作成します。
7. [ネットワーク設定] では、[起動テンプレートに含めない] を選択したままにするか、さまざまなネットワーク設定の値を指定できます。
8. [ストレージの設定] で、起動テンプレートで AMI を指定した場合、AMI にはルートボリューム ([ボリューム 1 (AMI ルート)]) を含む、1 つまたは複数のストレージボリュームが含まれます。オプションで、インスタンスにアタッチする追加のボリュームを指定できます。新しいボリュームを追加するには、[Add new volume] を選択してください。
9. [リソースタグ] で、インスタンスの起動時に作成されるリソースにタグを付けるには、[タグの追加] を選択し、タグのキーと値のペアを入力してください。[Resource types] (リソースタイプ) で、作成時にタグを付けるリソースを指定します。すべてのリソースに同じタグを

指定することも、リソースごとに異なるタグを指定することもできます。追加するタグごとに [Add tag] (タグの追加) を選択してください。

起動テンプレートの使用時に作成される次のリソースにタグを指定できます。

- インスタンス
- ボリューム
- Elastic graphics
- スポットインスタンスリクエスト
- ネットワークインターフェイス

> **Note**
>
> 起動テンプレート自体にタグを付けるには、[Template tags] (テンプレートタグ) でタグを指定する必要があります。詳細については、この手順のステップ 3 を参照してください。

10. [高度な詳細] では、セクションを展開してフィールドを表示し、必要に応じてインスタンスの追加パラメータを指定します。
11. [概要] パネルから起動テンプレートの設定を確認します。リンクを選択して任意のセクションに移動し、必要な変更を加えることができます。
12. 起動テンプレートを作成する準備ができたら、[Create launch template] (起動テンプレートの作成) をクリックしてください。

AWS CLI

起動テンプレートを作成するには

create-launch-template コマンドを使用します。

```
aws ec2 create-launch-template \
    --launch-template-name TemplateForWebServer \
    --version-description WebVersion1 \
    --tag-specifications 'ResourceType=launch-
template,Tags=[{Key=purpose,Value=production}]' \
    --launch-template-data file://template-data.json
```

インスタンス設定の起動テンプレートデータを指定する JSON の例を以下に示します。コマンド例に示すように、JSON をファイルに保存して --launch-template-data パラメータに含めます。

```
{
    "NetworkInterfaces": [{
        "AssociatePublicIpAddress": true,
        "DeviceIndex": 0,
        "Ipv6AddressCount": 1,
        "SubnetId": "subnet-0abcdef1234567890"
    }],
    "ImageId": "ami-0abcdef1234567890",
    "InstanceType": "r5.4xlarge",
    "TagSpecifications": [{
        "ResourceType": "instance",
        "Tags": [{
            "Key":"Name",
            "Value":"webserver"
        }]
    }],
    "CpuOptions": {
        "CoreCount":4,
        "ThreadsPerCore":2
    }
}
```

以下は出力の例です。

```
{
    "LaunchTemplate": {
        "LatestVersionNumber": 1,
        "LaunchTemplateId": "lt-01238c059e3466abc",
        "LaunchTemplateName": "TemplateForWebServer",
        "DefaultVersionNumber": 1,
        "CreatedBy": "arn:aws:iam::123456789012:root",
        "CreateTime": "2017-11-27T09:13:24.000Z"
    }
}
```

PowerShell

起動テンプレートを作成するには

New-EC2LaunchTemplate コマンドレットを使用します。

```
$launchTemplateData = [Amazon.EC2.Model.RequestLaunchTemplateData]@{
    ImageId = 'ami-0abcdef1234567890'
    InstanceType = 'r5.4xlarge'
    NetworkInterfaces = @(

 [Amazon.EC2.Model.LaunchTemplateInstanceNetworkInterfaceSpecificationRequest]@{
            AssociatePublicIpAddress = $true
            DeviceIndex = 0
            Ipv6AddressCount = 1
            SubnetId = 'subnet-0abcdef1234567890'
        }
    )
    TagSpecifications = @(
        [Amazon.EC2.Model.LaunchTemplateTagSpecificationRequest]@{
            ResourceType = 'instance'
            Tags = [Amazon.EC2.Model.Tag]@{
                Key = 'Name'
                Value = 'webserver'
            }
        }
    )
    CpuOptions = [Amazon.EC2.Model.LaunchTemplateCpuOptionsRequest]@{
        CoreCount = 4
        ThreadsPerCore = 2
    }
}
$tagSpecificationData = [Amazon.EC2.Model.TagSpecification]@{
    ResourceType = 'launch-template'
    Tags = [Amazon.EC2.Model.Tag]@{
        Key = 'purpose'
        Value = 'production'
    }
}
New-EC2LaunchTemplate -LaunchTemplateName 'TemplateForWebServer' `
    -VersionDescription 'WebVersion1' `
    -LaunchTemplateData $launchTemplateData `
    -TagSpecification $tagSpecificationData
```

以下は出力の例です。

```
CreatedBy            : arn:aws:iam::123456789012:root
```

```
CreateTime           : 9/19/2023 16:57:55
DefaultVersionNumber : 1
LatestVersionNumber  : 1
LaunchTemplateId     : lt-01238c059eEXAMPLE
LaunchTemplateName   : TemplateForWebServer
Tags                 : {purpose}
```

既存の起動テンプレートからの起動テンプレートの作成

既存の起動テンプレートをクローンして、新しい起動テンプレートを作成するようにパラメータを調整できます。ただし、これを実行できるのは、Amazon EC2 コンソールを使用している場合のみです。AWS CLI はテンプレートのクローン作成をサポートしていません。各パラメータの説明については、「Amazon EC2 インスタンス設定パラメータのリファレンス」を参照してください。

Console

既存の起動テンプレートから起動テンプレートを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[起動テンプレート]、[起動テンプレートの作成] の順に選択してください。
3. [起動テンプレート名] に、起動テンプレートのわかりやすい名前を入力してください。
4. [Template version description] (テンプレートバージョンの説明) に、起動テンプレートバージョンの短い説明を入力してください。
5. 作成時に起動テンプレートにタグを付けるには、[テンプレートタグ] を展開し、[新しいタグの追加] を選択して、タグのキーと値のペアを入力してください。
6. [Source template] (ソーステンプレート) を展開し、[Launch template name] (起動テンプレート名) で、新しい起動テンプレートのベースとなる起動テンプレートを選択してください。
7. [Source template version] (ソーステンプレートのバージョン) で、新しい起動テンプレートのベースとなる起動テンプレートのバージョンを選択してください。
8. 必要に応じて起動パラメータを調整し、[起動テンプレートの作成] を選択してください。

インスタンスからの起動テンプレートの作成

既存の Amazon EC2 インスタンスのパラメータのクローンを作成し、パラメータを調整して起動テンプレートを作成できます。各パラメータの説明については、「Amazon EC2 インスタンス設定パラメータのリファレンス」を参照してください。

Console

インスタンスから起動テンプレートを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選び、[アクション]、[イメージとテンプレート]、[インスタンスからテンプレートを作成] の順に選択してください。
4. 名前、説明、およびタグを入力し、必要に応じて起動パラメータを調整します。

> **Note**
>
> インスタンスから起動テンプレートを作成するとき、そのインスタンスのネットワークインターフェイス ID と IP アドレスはテンプレートに含まれません。

5. [起動テンプレートの作成] を選択してください。

AWS CLI

AWS CLI を使用して既存のインスタンスから起動テンプレートを作成するには、まずインスタンスから起動テンプレートデータを取得し、次に起動テンプレートデータを使用して起動テンプレートを作成します。

インスタンスから起動テンプレートデータを取得するには

- get-launch-template-data コマンドを使用して、インスタンス ID を指定します。出力をベースとして使用して、新しい起動テンプレートや起動テンプレートのバージョンを作成できます。デフォルトでは、起動テンプレートデータで指定できない最上位レベルの `LaunchTemplateData` オブジェクトが出力に含まれています。このオブジェクトを除外するには、`--query` オプションを使用します。

```
aws ec2 get-launch-template-data \
    --instance-id i-0123d646e8048babc \
    --query "LaunchTemplateData"
```

出力例を次に示します。

```
    {
```

```
        "Monitoring": {},
        "ImageId": "ami-8c1be5f6",
        "BlockDeviceMappings": [
            {
                "DeviceName": "/dev/xvda",
                "Ebs": {
                    "DeleteOnTermination": true
                }
            }
        ],
        "EbsOptimized": false,
        "Placement": {
            "Tenancy": "default",
            "GroupName": "",
            "AvailabilityZone": "us-east-1a"
        },
        "InstanceType": "t2.micro",
        "NetworkInterfaces": [
            {
                "Description": "",
                "NetworkInterfaceId": "eni-35306abc",
                "PrivateIpAddresses": [
                    {
                        "Primary": true,
                        "PrivateIpAddress": "10.0.0.72"
                    }
                ],
                "SubnetId": "subnet-7b16de0c",
                "Groups": [
                    "sg-7c227019"
                ],
                "Ipv6Addresses": [
                    {
                        "Ipv6Address": "2001:db8:1234:1a00::123"
                    }
                ],
                "PrivateIpAddress": "10.0.0.72"
            }
        ]
    }
```

出力を次のようにファイルに直接書き込むことができます。

```
aws ec2 get-launch-template-data \
    --instance-id i-0123d646e8048babc \
    --query "LaunchTemplateData" >> instance-data.json
```

起動テンプレートデータを使用して起動テンプレートを作成するには

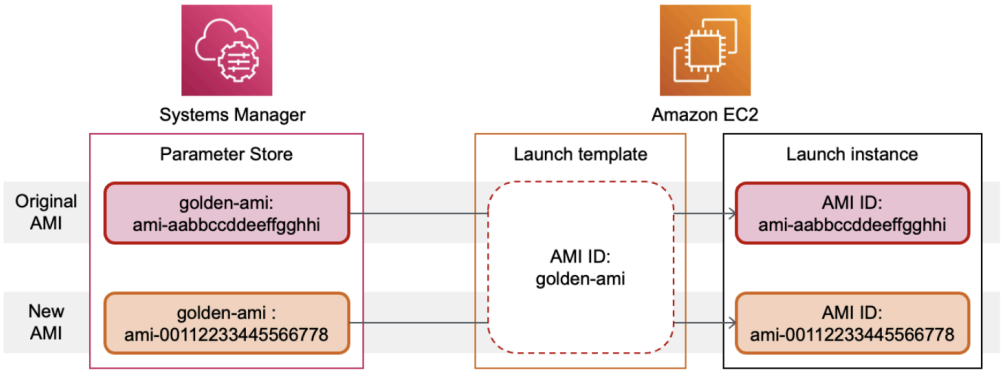
- create-launch-template コマンドを使用して、前の手順の出力を使用して起動テンプレートを作成します。AWS CLI を使用した起動テンプレートの作成の詳細については、パラメータを指定して起動テンプレートを作成するを参照してください。

AMI ID のかわりに Systems Manager パラメータを使用する

起動テンプレートで AMI ID を指定する代わりに、AWS Systems Manager パラメータを指定できます。AMI ID が変更された場合は、Systems Manager パラメータストアの Systems Manager パラメータを更新することで、AMI ID を一箇所で更新できます。パラメータは、他の AWS アカウントと共有することもできます。AMI パラメータは、1 つのアカウントで一元的に保存および管理し、これを参照する必要がある他のすべてのアカウントに共有することができます。Systems Manager パラメータを使用すると、すべての起動テンプレートを 1 回のアクションで更新できます。

Systems Manager パラメータは、AWS Systems Manager Parameter Store で作成できるユーザー定義のキーと値のペアです。パラメータストアは、アプリケーションの設定値を保存するための一元的な場所を提供します。

次の図では、golden-ami パラメータは最初にパラメータストア内の元の AMI ami-aabbccddeeffgghhi にマッピングされます。起動テンプレートの AMI ID の値は golden-ami です。この起動テンプレートを使用してインスタンスを起動するとき、AMI ID は ami-aabbccddeeffgghhi に変換されます。その後、AMI が更新され、新しい AMI ID が取得されます。パラメータストアでは、golden-ami パラメータが新しい ami-00112233445566778 パラメータにマッピングされます。起動テンプレートは変更されずに引き継がれます。この起動テンプレートを使用してインスタンスを起動するとき、AMI ID は新しい ami-00112233445566778 に変換されます。



AMI ID の Systems Manager パラメータ形式

起動テンプレートを AMI ID の代わりに使用する場合、ユーザー定義の Systems Manager パラメータを次の形式に従う必要があります。

- パラメータ型: `String`
- パラメータデータ型: `aws:ec2:image` - これにより、パラメータストアは入力した値が AMI ID の適切な形式であることを検証します。

AMI ID での有効なパラメータの作成については、「AWS Systems Manager ユーザーガイド」の「Systems Manager パラメータの作成」を参照してください。

起動テンプレートの Systems Manager パラメータ形式

起動テンプレートの AMI ID の代わりに Systems Manager パラメータを使用するには、起動テンプレートでパラメータを指定するときに次のいずれかの形式を使用する必要があります。

パブリックパラメータを参照するには:

- `resolve:ssm:public-parameter`

同じアカウントに保存されているパラメータを参照するには:

- `resolve:ssm:parameter-name`
- `resolve:ssm:parameter-name:version-number` - バージョン番号自体がデフォルトのラベルです
- `resolve:ssm:parameter-name:label`

別の AWS アカウント から共有されたパラメータを参照するには:

- `resolve:ssm:parameter-ARN`
- `resolve:ssm:parameter-ARN:version-number`
- `resolve:ssm:parameter-ARN:label`

パラメータバージョン

Systems Manager パラメータはバージョニングされたリソースです。パラメータを更新すると、そのパラメータの新しいバージョンが連続して作成されます。Systems Manager は、特定のバージョンのパラメータにマッピングできるパラメータラベルをサポートしています。

例えば、golden-ami パラメータには、1、2 および 3 の 3 つのバージョンがあります。バージョン 2 にマッピングするパラメータラベル beta と、バージョン 3 にマップするパラメータラベル prod を作成できます。

起動テンプレートでは、次のいずれかの形式を使用して golden-ami パラメータのバージョン 3 を指定できます。

- `resolve:ssm:golden-ami:3`
- `resolve:ssm:golden-ami:prod`

バージョンまたはラベルの指定は任意です。バージョンやパラメータを指定しない場合は最新バージョンのパラメータが使用されます。

起動テンプレートの Systems Manager パラメータを指定する

起動テンプレートまたは起動テンプレートの新しいバージョンを作成するときに、起動テンプレートで AMI ID の代わりに Systems Manager パラメータを指定できます。

Console

起動テンプレートの Systems Manager パラメータを指定するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[起動テンプレート]、[起動テンプレートの作成] の順に選択してください。
3. [起動テンプレート名] に、起動テンプレートのわかりやすい名前を入力してください。

4. [Application and OS Images (Amazon Machine Image)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で、[Browse more AMI] (その他の AMI を閲覧する) を選択してください。
5. 検索バーの右側にある矢印ボタンを選択したら、[カスタム値を指定 / Systems Manager パラメータ] を選択してください。
6. [カスタム値または Systems Manager のパラメータを指定] ダイアログボックスで、次の手順を実行します。

   a. [AMI ID または Systems Manager パラメータ文字列] の場合、次の形式のいずれかを使用して Systems Manager パラメータ名を入力してください。

      パブリックパラメータを参照するには:

      - **resolve:ssm:*public-parameter***

      同じアカウントに保存されているパラメータを参照するには:

      - **resolve:ssm:*parameter-name***
      - **resolve:ssm:*parameter-name*:*version-number***
      - **resolve:ssm:*parameter-name*:*label***

      別の AWS アカウント から共有されたパラメータを参照するには:

      - **resolve:ssm:*parameter-ARN***
      - **resolve:ssm:*parameter-ARN*:*version-number***
      - **resolve:ssm:*parameter-ARN*:*label***

   b. [Save] を選択してください。
7. 必要に応じて他の起動テンプレートパラメータを指定し、その後に [起動テンプレートの作成] を選択してください。

詳細については、「パラメータを指定して起動テンプレートを作成する」を参照してください。

AWS CLI

起動テンプレートの Systems Manager パラメータを指定するには

- create-launch-template コマンドを使用して、起動テンプレートを作成します。使用する AMI を指定するには、次のいずれかの形式を使用して Systems Manager パラメータ名を入力してください。

  パブリックパラメータを参照するには:

  - **resolve:ssm:*public-parameter***

  同じアカウントに保存されているパラメータを参照するには:

  - **resolve:ssm:*parameter-name***
  - **resolve:ssm:*parameter-name*:*version-number***
  - **resolve:ssm:*parameter-name*:*label***

  別の AWS アカウント から共有されたパラメータを参照するには:

  - **resolve:ssm:*parameter-ARN***
  - **resolve:ssm:*parameter-ARN*:*version-number***
  - **resolve:ssm:*parameter-ARN*:*label***

  次の例では、以下を指定する起動テンプレートを作成します。

  - 起動テンプレートの名前 (*TemplateForWebServer*)
  - 起動テンプレートのタグ (*purpose*=*production*)
  - JSON ファイルで指定された、インスタンス設定のデータ。
    - 使用する AMI (resolve:ssm:*golden-ami*)
    - 起動するインスタンスタイプ (*m5.4xlarge*)
    - インスタンスに付けるタグ (*Name*=*webserver*)

```
aws ec2 create-launch-template \
    --launch-template-name TemplateForWebServer \
```

```
    --tag-specifications 'ResourceType=launch-
template,Tags=[{Key=purpose,Value=production}]' \
    --launch-template-data file://template-data.json
```

インスタンスを設定するための起動テンプレートデータを含む、JSON ファイルの例を以下に示します。ImageId の値は Systems Manager パラメータ名で、必要な形式 `resolve:ssm:golden-ami` で入力されます。

```
{"LaunchTemplateData": {
    "ImageId": "resolve:ssm:golden-ami",
    "InstanceType": "m5.4xlarge",
    "TagSpecifications": [{
        "ResourceType": "instance",
        "Tags": [{
            "Key":"Name",
            "Value":"webserver"
        }]
    }]
  }
}
```

起動テンプレートが正しい AMI ID を取得していることを確認する

Systems Manager のパラメータを実際の AMI ID に変換するには

describe-launch-template-versions コマンドを使用して、--resolve-alias パラメータを含めます。

```
aws ec2 describe-launch-template-versions \
    --launch-template-name my-launch-template \
    --versions $Default \
    --resolve-alias
```

レスポンスには、ImageId の AMI ID が含まれます。この例では、この起動テンプレートを使用してインスタンスを起動すると、AMI ID は ami-0ac394d6a3example に変換されます。

```
{
    "LaunchTemplateVersions": [
        {
            "LaunchTemplateId": "lt-089c023a30example",
            "LaunchTemplateName": "my-launch-template",
```

```
            "VersionNumber": 1,
            "CreateTime": "2022-12-28T19:52:27.000Z",
            "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
            "DefaultVersion": true,
            "LaunchTemplateData": {
                "ImageId": "ami-0ac394d6a3example",
                "InstanceType": "t3.micro",
            }
        }
    ]
}
```

関連リソース

Systems Manager パラメータの使用に関する詳細は、Systems Manager のドキュメントの次のリファレンス資料を参照してください。

- Amazon EC2 でサポートされている AMI パブリックパラメータを検索する方法については、「Calling AMI public parameters」を参照してください。
- パラメータを他の AWS アカウントと共有する方法、または AWS Organizations を介して共有する方法については、「Working with shared parameters」を参照してください。
- パラメータが正常に作成されたかどうかをモニタリングする方法については、「Native parameter support for Amazon Machine Image IDs」を参照してください。

制限事項

- タイプが `instant` の EC2 フリートのみが、AMI ID の代わりに Systems Manager パラメータが指定されている起動テンプレートを使用できます。
- タイプが `maintain` と `request` の EC2 フリートと、さらにスポットフリートでは、AMI ID の代わりに Systems Manager パラメータが指定されている起動テンプレートを使用できません。タイプが `maintain` と `request` の EC2 フリートと、さらにスポットフリートでは、起動テンプレートに AMI を指定する場合、AMI ID を指定する必要があります。
- EC2 フリートで属性ベースのインスタンスの選択を使用した場合は、AMI ID の代わりに Systems Manager パラメータを指定できません。属性ベースのインスタンスの選択を使用する場合は、AMI ID を指定する必要があります。
- Amazon EC2 Auto Scaling には、その他の制限があります。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Use AWS Systems Manager parameters instead of AMI IDs in launch templates」を参照してください。

起動テンプレートの変更 (起動テンプレートのバージョンの管理)

起動テンプレートは変更不可能です。起動テンプレートを作成したら、それを変更することはできません。代わりに、必要な変更を含む新しいバージョンの起動テンプレートを作成できます。

起動テンプレートの別のバージョンの作成、デフォルトバージョンの設定、起動テンプレートバージョンの説明、不要になったバージョンの削除を行うことができます。

タスク

- 起動テンプレートのバージョンの作成
- デフォルトの起動テンプレートのバージョンの設定
- 起動テンプレートのバージョンの説明

起動テンプレートのバージョンの作成

起動テンプレートのバージョンを作成する際、新しいバージョンに新しい起動パラメータを指定するか、または既存のバージョンをベースとして使用できます。各パラメータの説明については、「Amazon EC2 インスタンス設定パラメータのリファレンス」を参照してください。

Console

起動テンプレートのバージョンを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Launch Templates] を選択してください。
3. 起動テンプレートを選択し、[アクション]、[Modify template (Create new version)] の順に選択してください。
4. [Template version description] (テンプレートバージョンの説明) に、起動テンプレートバージョンについての説明を入力してください。
5. (オプション) [Source template] (ソーステンプレート) を展開し、新しい起動テンプレートバージョンのベースとして使用する起動テンプレートのバージョンを選択してください。新しい起動テンプレートバージョンは、この起動テンプレートバージョンから起動パラメータを継承します。
6. 必要に応じて起動パラメータを変更します。
7. [起動テンプレートの作成] を選択してください。

AWS CLI

起動テンプレートのバージョンを作成するには

create-launch-template-version コマンドを使用します。新しいバージョンのベースとなるソースバージョンを指定できます。新しいバージョンはこのバージョンの起動パラメータを継承し、--launch-template-data を使用してパラメータを上書きできます。次の例では、起動テンプレートのバージョン 1 に基づいて新しいバージョンを作成し、異なる AMI ID を指定します。

```
aws ec2 create-launch-template-version \
    --launch-template-id lt-0abcd290751193123 \
    --version-description WebVersion2 \
    --source-version 1 \
    --launch-template-data "ImageId=ami-0abcdef1234567890"
```

PowerShell

起動テンプレートのバージョンを作成するには

New-EC2LaunchTemplateVersion コマンドレットを使用します。新しいバージョンのベースとなるソースバージョンを指定できます。新しいバージョンはこのバージョンの起動パラメータを継承し、LaunchTemplateData を使用してパラメータを上書きできます。次の例では、起動テンプレートのバージョン 1 に基づいて新しいバージョンを作成し、異なる AMI ID を指定します。

```
New-EC2LaunchTemplateVersion `
    -LaunchTemplateId lt-0abcd290751193123 `
    -VersionDescription WebVersion2 `
    -SourceVersion 1 `
    -LaunchTemplateData (
        New-Object `
            -TypeName Amazon.EC2.Model.RequestLaunchTemplateData `
            -Property @{ImageId = 'ami-0abcdef1234567890'}
    )
```

デフォルトの起動テンプレートのバージョンの設定

起動テンプレートにデフォルトバージョンを設定できます。起動テンプレートからインスタンスを起動し、バージョンを指定しない場合、インスタンスはデフォルトバージョンのパラメータを使用して起動されます。

Console

デフォルトの起動テンプレートのバージョンを設定するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Launch Templates] を選択してください。
3. 起動テンプレートを選択し、[アクション]、[デフォルトバージョンの設定] を選択してください。
4. [テンプレートバージョン] で、デフォルトバージョンとして設定するバージョン番号を選択し、[デフォルトバージョンとして設定] を選択してください。

AWS CLI

デフォルトの起動テンプレートのバージョンを設定するには

modify-launch-template コマンドを使用します。

```
aws ec2 modify-launch-template \
    --launch-template-id lt-0abcd290751193123 \
    --default-version 2
```

PowerShell

デフォルトの起動テンプレートのバージョンを設定するには

Edit-EC2LaunchTemplate コマンドレットを使用します。

```
Edit-EC2LaunchTemplate `
    -LaunchTemplateId lt-0abcd290751193123 `
    -DefaultVersion 2
```

起動テンプレートのバージョンの説明

コンソールを使用して、選択した起動テンプレートのすべてのバージョンを表示したり、特定のバージョン番号と一致する最新バージョンやデフォルトバージョンの起動テンプレートを一覧表示したりできます。AWS CLI を使用すると、指定した起動テンプレートのすべてのバージョン、各バージョン、特定範囲のバージョンを表示できます。また、アカウント内にあるすべての起動テンプレートに

ついて、すべての最新バージョンを表示したり、すべてのデフォルトバージョンを表示したりすることもできます。

Console

起動テンプレートのバージョンを説明するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Launch Templates] を選択してください。
3. 特定の起動テンプレートのバージョンを表示したり、特定のバージョン番号と一致する最新バージョンやデフォルトバージョンの起動テンプレートを一覧表示したりできます。
   - 起動テンプレートのバージョンを表示するには、起動テンプレートを選択してください。[バージョン] タブの [バージョン] から、詳細を表示するバージョンを選択してください。
   - 特定のバージョン番号と一致する最新バージョンの起動テンプレートを一覧表示するには、検索バーから [最新バージョン] を選択し、バージョン番号を選択してください。
   - 特定のバージョン番号と一致するデフォルトバージョンの起動テンプレートを一覧表示するには、検索バーから [デフォルトバージョン] を選択し、バージョン番号を選択してください。

AWS CLI

起動テンプレートのバージョンを説明するには

describe-launch-template-versions コマンドを使用して、バージョン番号を指定します。次の例では、バージョン *1* と *3* を指定しています。

```
aws ec2 describe-launch-template-versions \
    --launch-template-id lt-0abcd290751193123 \
    --versions 1 3
```

アカウント内にある起動テンプレートの最新バージョンやデフォルトバージョンを記述するには

delete-launch-template-versions コマンドを使用し、$Latest または $Default を指定するか、両方を指定します。呼び出しでは、起動テンプレートの ID と名前を省略する必要があります。バージョン番号を指定することはできません。

```
aws ec2 describe-launch-template-versions \
```

```
    --versions "$Latest,$Default"
```

PowerShell

起動テンプレートのバージョンを説明するには

Get-EC2TemplateVersion コマンドレットを使用して、バージョン番号を指定します。次の例では、バージョン *1* と *3* を指定しています。

```
Get-EC2TemplateVersion `
    -LaunchTemplateId lt-0abcd290751193123 `
    -Version 1,3
```

アカウント内にある起動テンプレートの最新バージョンやデフォルトバージョンを記述するには

Get-EC2TemplateVersion コマンドレットを使用して $Latest、$Default、またはその両方を指定します。呼び出しでは、起動テンプレートの ID と名前を省略する必要があります。バージョン番号を指定することはできません。

```
Get-EC2TemplateVersion `
    -Version '$Latest','$Default'
```

起動テンプレートまたは起動テンプレートのバージョンを削除する

起動テンプレートが不要になった場合には、それを削除することができます。起動テンプレートを削除すると、すべてのバージョンが削除されます。起動テンプレートの特定のバージョンのみを削除する場合は、起動テンプレートの他のバージョンを保持したまま削除できます。

起動テンプレートまたは起動テンプレートのバージョンを削除しても、起動テンプレートから起動したインスタンスには影響しません。

起動テンプレートとそのすべてのバージョンを削除する

起動テンプレート (すべてのバージョンを含む) が不要になった場合は、起動テンプレートを削除できます。起動テンプレートを削除すると、すべてのバージョンが削除されます。

Console

起動テンプレートとそのすべてのバージョンを削除するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Launch Templates] を選択してください。
3. 起動テンプレートを選択し、[アクション]、[テンプレートの削除] を選択してください。
4. **`Delete`** と入力して削除を確認し、[Delete (削除)] を選択してください。

AWS CLI

起動テンプレートとそのすべてのバージョンを削除するには

delete-launch-template コマンドを使用して、起動テンプレートを指定します。

```
aws ec2 delete-launch-template --launch-template-id lt-01238c059e3466abc
```

PowerShell

起動テンプレートとそのすべてのバージョンを削除するには

Remove-EC2LaunchTemplate AWS Tools for PowerShellコマンドを使用して、起動テンプレートを指定します。`-Force` を省略すると、PowerShell で確認を求められます。

```
Remove-EC2LaunchTemplate -LaunchTemplateId lt-0123456789example -Force
```

起動テンプレートのバージョンの削除

起動テンプレートのバージョンが不要になった場合には、それを削除することができます。

考慮事項

- 削除後にバージョン番号を置き換えることはできません。
- 起動テンプレートのデフォルトバージョンは削除できません。まず、デフォルトとして別のバージョンを割り当てる必要があります。デフォルトバージョンが起動テンプレートの唯一のバージョンである場合は、 起動テンプレート全体を削除する必要があります。
- コンソールを使用する場合、一度に削除できる起動テンプレートバージョンは 1 つです。AWS CLI を使用する場合は、一度のリクエストで最大 200個 の起動テンプレートバージョンを削除で

きます。1 回のリクエストで 200 を超えるバージョンを削除するには、起動テンプレートを削除します。その場合、バージョンもすべて削除されます。

Console

起動テンプレートのバージョンを削除するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Launch Templates] を選択してください。
3. 起動テンプレートを選択し、[アクション]、[テンプレートのバージョンの削除] を選択してください。
4. 削除するバージョンを選択し、[削除] を選択してください。

AWS CLI

起動テンプレートのバージョンを削除するには

delete-launch-template-versions コマンドを使用して、削除するバージョン番号を指定します。1 回のリクエストで削除する起動テンプレートバージョンを最大 200 個指定できます。

```
aws ec2 delete-launch-template-versions \
    --launch-template-id lt-0abcd290751193123 \
    --versions 1
```

PowerShell

起動テンプレートのバージョンを削除するには

Remove-EC2TemplateVersion コマンドレットを使用して、削除するバージョン番号を指定します。1 回のリクエストで削除する起動テンプレートバージョンを最大 200 個指定できます。

```
Remove-EC2TemplateVersion `
    -LaunchTemplateId lt-0abcd290751193123 `
    -Version 1
```

Amazon EC2 インスタンスの起動

インスタンスは、AWS クラウド内の仮想サーバーです。Amazon Machine Image (AMI) からインスタンスを起動します。AMI はインスタンスに対して、オペレーティングシステム、アプリケーションサーバー、およびアプリケーションを提供します。

AWS アカウントを作成すると、AWS 無料利用枠を利用して、Amazon EC2 を無料で使い始めることができます。無料利用枠の特典は、AWS アカウントを作成した時期によって異なります。2025 年 7 月 15 日より前に AWS アカウントを作成していて、12 か月が経過していない場合は、無料利用枠を使用して `t2.micro` インスタンスを起動し、無料で使用できます (`t2.micro` が利用できないリージョンでは、無料利用枠で `t3.micro` インスタンスを使用できます)。インスタンスが実行されている間は、アイドル状態であっても、そのインスタンスまたは使用量についての料金が発生し、無料利用枠の制限にカウントされます。詳細については、Amazon EC2 料金表 を参照してください。2025 年 7 月 15 日以降に AWS アカウントを作成した場合は、`t3.micro`、`t3.small`、`t4g.micro`、`t4g.small`、`c7i-flex.large`、および `m7i-flex.large` インスタンスタイプを 6 か月間、またはクレジットを使い切るまで使用できます。詳細については、2025 年 7 月 15 日より前とそれ以降の無料利用枠の特典を参照してください。

インスタンスを起動する場合、次のいずれかのリソースに関連付けられているサブネットでインスタンスを起動できます。

- アベイラビリティーゾーン - このオプションはデフォルトです。
- ローカルゾーン - ローカルゾーンでインスタンスを起動するには、ローカルゾーンにオプトインし、このゾーンにサブネットを作成する必要があります。詳細については、「Get started with Local Zones」を参照してください。
- Wavelength Zone - Wavelength Zone でインスタンスを起動するには、Wavelength Zone にオプトインし、このゾーンにサブネットを作成する必要があります。Wavelength Zone でインスタンスを起動する方法については、「AWS Wavelength の開始方法」を参照してください。
- Outpost - Outpost でインスタンスを起動するには、Outpost を作成する必要があります。Outpost の作成方法については、「AWS Outposts の開始方法」を参照してください。

インスタンスを起動した後、インスタンスに接続して使用できます。最初、インスタンスの状態は `pending` です。インスタンスの状態が `running` の場合、インスタンスは起動を開始します。インスタンスに接続するまで、少し時間がかかることがあります。ベアメタルインスタンスタイプの起動には時間がかかることがあります。

インスタンスへの接続方法によっては、インスタンスの起動中に特定の設定を行う場合があります。これらの設定には、特定のトラフィックのインバウンドセキュリティグループルールの指定や、インスタンスプロファイルロールの関連付けが含まれます。接続に使用できる接続方法とその要件の詳細については、「EC2 インスタンスに接続する」を参照してください。

インスタンスは、パブリック DNS 名を受信します。この DNS 名はインターネットからインスタンスに接続する場合に使用できます。また、インスタンスはプライベート DNS 名も受け取ります。これは、同じ VPC 内の他のインスタンスがインスタンスに接続するために使用できます。

インスタンスの使用が終了したら、不要なコストが発生しないように、必ずインスタンスを終了してください。詳細については、Amazon EC2 インスタンスを終了するを参照してください。

多数のインスタンスを起動したり、複数のインスタンスタイプを使用したり、オンデマンドインスタンス、リザーブドインスタンス、スポットインスタンスなどの複数の購入オプションを使用する必要がある場合は、EC2 フリートの使用を検討してください。詳細については、EC2 フリートとスポットフリートを参照してください。

インスタンスのライフサイクルを自動化する場合 (自動スケーリング、ヘルスチェック、異常なインスタンスの交換など) は、Amazon EC2 Auto Scaling の使用を検討してください。

インスタンスを起動する方法の一部を次に示します。

方法	ツール	ドキュメント
インスタンス起動ウィザードを使用して、起動パラメータを指定します。	Amazon EC2 コンソール	コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する
起動テンプレートを作成して、起動テンプレートからインスタンスを起動します。	Amazon EC2 コンソール	起動テンプレートを使用した EC2 インスタンスの起動
既存のインスタンスを基本として使用します。	Amazon EC2 コンソール	既存のインスタンスの詳細を使用した EC2 インスタンスの起動
AWS Marketplace から購入した AMI を使用します。	Amazon EC2 コンソール	AWS Marketplace AMI から Amazon EC2 インスタンスを起動する

方法	ツール	ドキュメント
指定した AMI を使用します。	AWS CLI	AWS CLI にある Amazon EC2 インスタンスの起動、一覧表示、削除
指定した AMI を使用します。	AWS Tools for Windows PowerShell	Windows PowerShell を使用した Amazon EC2 インスタンスの起動
EC2 フリートを使用すると、キャパシティのプロビジョニングを異なる EC2 インスタンスタイプとアベイラビリティーゾーン間で行うことも、オンデマンドインスタンス、リザーブドインスタンス、スポットインスタンスの各購入オプション全体で行うこともできます。	AWS CLI	EC2 フリートとスポットフリート
CloudFormation テンプレートを使用してインスタンスを指定します。	AWS CloudFormation	AWS::EC2::Instance AWS CloudFormationユーザーガイド
言語固有の AWS SDK を使用してインスタンスを起動します。	AWS SDK	AWS SDK for .NET AWS SDK for C++ AWS SDK for Go AWS SDK for Java AWS SDK for JavaScript AWS SDK for PHP V3 AWS SDK for Python AWS SDK for Ruby V3

EC2 インスタンスを起動するためのチュートリアル

Amazon EC2 インスタンスを起動して設定するには、さまざまな方法があります。使用する方法と設定は、特定のユースケースによって異なります。

以下のチュートリアルは、EC2 インスタンスを起動する方法を学ぶのに役立ちます。Amazon EC2 を初めて使用する場合は、最初のチュートリアルから始めることをお勧めします。チュートリアルは、基本を紹介することから始まります。また、さらに多くの設定オプションを紹介することで、基本を基にステップアップするのに役立ちます。

目的	チュートリアルへのリンク
最初の EC2 インスタンスを起動する Amazon EC2 インスタンス起動ウィザードのデフォルト設定を使用して、Amazon EC2 インスタンスをすばやく起動する方法について説明します。また、インスタンス設定フィールドを確認し、インスタンスを終了する方法についても説明します。 所要時間: 10 分	チュートリアル 1: 最初の Amazon EC2 インスタンスを起動する
テスト EC2 インスタンスを起動し接続する テスト目的で使用できる Amazon EC2 インスタンスを起動する方法について説明します。このインスタンスには高度な設定はなく、機密情報は保存されません。また、基本的なインスタンス設定、インスタンスへの接続方法、およびインスタンスを停止する方法についても説明します。 所要時間: 30 分	チュートリアル 2: テスト EC2 インスタンスを起動して接続する

他のチュートリアルをお探しですか。

- チュートリアル: AL2023 に LAMP サーバーをインストールする

- チュートリアル: AL2023 に SSL/TLS を設定する
- チュートリアル: AL2023 で WordPress ブログをホストする
- チュートリアル: EC2 Instance Connect を使用してインスタンスに接続するために必要な設定を完了する
- チュートリアル: Amazon RDS データベースに Amazon EC2 インスタンスを接続する

チュートリアル 1: 最初の Amazon EC2 インスタンスを起動する

チュートリアルの目的	Amazon EC2 インスタンス起動ウィザードのデフォルト設定を使用して、Amazon EC2 インスタンスをすばやく起動する方法について説明します。また、インスタンス設定フィールドを確認し、インスタンスを終了する方法についても説明します。
EC2 エクスペリエンス	初心者
期間	10 分
コスト	無料利用枠の対象 AWS アカウントを作成すると、AWS 無料利用枠を利用して、Amazon EC2 を無料で使い始めることができます。 AWS アカウントを作成したのが 2025 年 7 月 15 日より前で、12 か月が経過しておらず、Amazon EC2 の無料利用枠の特典をまだ使い切っていない場合、無料利用枠の特典に収まるオプションを選択できるように支援いたしますので、このチュートリアルを完了するのに一切料金はかかりません。それ以外の場合、インスタンスを起動したとき (アイドル状態のままであっても) から終了するまで、標準の Amazon EC2 使用料金が発生します。

	2025 年 7 月 15 日以降に AWS アカウントを作成し、6 か月が経過しておらず、すべてのクレジットを使い切っていない場合、無料利用枠の特典に収まるオプションを選択できるように支援いたしますので、このチュートリアルを完了するのに一切料金はかかりません。 無料利用枠の対象となるかどうかを判断する手順については、「the section called “無料利用枠の使用状況の追跡”」を参照してください。
前提条件	• AWS アカウントを所有し、管理者アクセス権を持つユーザーを設定し、管理者ユーザーを使用して AWS マネジメントコンソールにサインインする必要があります。これを行う方法がわかりませんか。次のチュートリアルを試してください: AWS 環境のセットアップ • AWS コンソール全般に精通している必要があります。どこから始めればよいかわかりませんか。この入門ガイドを試してください: AWS マネジメントコンソールの使用開始

チュートリアルの概要

このチュートリアルは、Amazon EC2 を使用した経験のない初心者向けに設計されています。EC2 コンソールを使用して、最初の EC2 インスタンスを作成 (これを起動と呼びます) するためのステップについて説明します。インスタンスは基本的に AWS クラウド内のウェブサーバーです。インスタンスを起動した後、コンソールでインスタンスを検索する方法を説明します。最後に、コストの管理に役立つように、インスタンスを削除 (これを終了と呼びます) する方法を説明します。

このチュートリアルは、次の短いタスクで構成されます。次のタスクに進む前に、各タスクを完了する必要があります。

- タスク 1: インスタンスを起動する
- タスク 2: インスタンスを検索する
- タスク 3: インスタンス設定を表示する
- タスク 4: インスタンスを終了する

タスク 1: インスタンスを起動する

このタスクでは、必須項目のみを実行して、インスタンスを起動する最も迅速な方法を取ります。EC2 インスタンス起動ウィザードを使用します。これは、インスタンスを設定および起動するためのすべてのフィールドを指定するウェブベースのフォームです。インスタンス設定フィールドにデフォルト値を指定することで、プロセスを簡素化します。

開始する前に

管理者ユーザーで AWS マネジメントコンソールにサインインするなど、前の表に記載されている前提条件を満たしていることを確認してください。

インスタンスをすばやく起動するには、次のステップに従います

1. Amazon EC2 コンソールを開く:

   https://console.aws.amazon.com/ec2/ に移動します。
2. EC2 インスタンス起動ウィザードを開く:

   EC2 ダッシュボードから、[インスタンスの起動] を選択します。

   [インスタンスの起動] ウェブベースのフォームが開きます。これが EC2 インスタンス起動ウィザードです。
3. インスタンスに名前を付ける:

   [名前とタグ] の [名前] に、**My first EC2 instance** のようなわかりやすい名前を入力します。

   インスタンスに名前を付ける必要はありませんが、後でインスタンスを識別するのに役立ちます。
4. キーペアなしで続行する:

   [キーペア (ログイン)] の [キーペア名] で、[キーペアなしで続行 (非推奨)] を選択してください。

   キーペアは、安全なログインに使用できます。ただし、このチュートリアルではインスタンスにログインしないため、現時点ではキーペアは必要ありません。
5. インスタンスを起動する:

   右側の [概要] パネルで、[インスタンスの起動] を選択してください。

Amazon EC2 は、デフォルト設定を使用してインスタンスをすばやく起動します。[成功] バナーは、インスタンスが起動されたことを示します。

おめでとうございます。最初の EC2 インスタンスが正常に起動されました。

タスク 2: インスタンスを検索する

このタスクでは、EC2 コンソールで起動したインスタンスを見つけます。

EC2 コンソールでインスタンスを検索するには、次のステップに従います

1. [インスタンス] ページを開く:

   成功ページがまだ表示されている場合は、画面上部のパンくずリストで [インスタンス] を選択します。アクセスするには、まず 3 点アイコンを選択する必要がある場合があります。

   移動した場合は、ナビゲーションペインから [インスタンス] を選択します。
2. インスタンスを見つける:

   [名前] 列で、指定した名前でインスタンスを検索します。

タスク 3: インスタンス設定を表示する

このタスクでは、インスタンスの設定の詳細表示について理解します。

インスタンスの設定を表示するには、次のステップに従います

1. インスタンス ID を見つける:

   [インスタンス ID] 列で、インスタンスの一意の ID を書き留めます。[i-01aeed690c9fb5322] など、[i] で始まり、17 文字の英数字が続きます。

   インスタンス ID は、起動時にインスタンスに自動的に割り当てられます。
2. インスタンスの詳細ページを開く:

   [インスタンス ID] 列で、ID リンクを選択してインスタンスの詳細ページを開き、設定を確認できます。
3. インスタンス設定の詳細を調べる:

インスタンスの設定の詳細について、少し時間をかけて確認してください。次のチュートリアルでは、設定について詳しく説明します。ここでは、この時間を利用してインスタンスの詳細ページについて理解してください。

ヒント: フィールドをすばやく見つけるには、キーボードで Ctrl+F キーまたは Command+F キーを押します。

a. インスタンスタイプ: インスタンスタイプを見つけられますか。例えば、[t3.micro] などです。

b. パブリック IPv4 アドレス: インスタンスに割り当てられたパブリック IPv4 アドレスを見つけられますか。これは、[34.242.148.128] のような形式です。

c. インスタンス所有者: このインスタンスの所有者を識別できますか。所有者はあなたです。AWS アカウント番号は、[所有者] フィールドの下に表示されます。

d. インスタンスタグ: インスタンスに付けた名前は、実際にはタグになります。インスタンスタグを見つけられますか。[タグ] タブを選択します。キーは [名前] で、値は指定した名前です。

e. 起動時間: インスタンスをいつ起動したかわかりますか。[詳細] タブを選択し、[起動時間] フィールドを見つけます。

f. インスタンスの状態: インスタンスの状態を確認できますか。[実行] になっているはずです。

他のインスタンス設定フィールドについても、少し時間をかけて確認してください。準備ができたら、次のタスクに進みます。

タスク 4: インスタンスを終了する

> **Warning**
>
> インスタンスの終了は恒久的で、元に戻すことができません。
> インスタンスを終了すると、そのインスタンスには接続できなくなり、復元することもできません。アタッチされている Amazon EBS ボリュームで、終了時に削除されるように設定されているものも、すべて恒久的に削除され、復元できません。インスタンスストアボリュームに保存されているデータは、すべて恒久的に失われます。詳細については「インスタンスの終了方法」を参照してください。

インスタンスを終了する前に、終了後も保持する必要があるすべてのデータを永続的ストレージにバックアップしたことを確認してください。

このタスクでは、インスタンスを削除して、無料利用枠の特典を維持します。EC2 では、終了はインスタンスの削除に使用される用語です。

インスタンスを終了するには、次のステップに従います

1. 終了を開始する:

   インスタンスの詳細ページがまだ表示されている場合は、[インスタンス状態] メニュー (右上) を選択し、[インスタンスの終了 (削除)] を選択します。

   移動した場合は、ナビゲーションペインから [インスタンス] を選択します。次に、[インスタンス] ページでインスタンスの名前の横にあるチェックボックスをオンにしたら、[インスタンス状態] メニュー (右上) を選択し、[インスタンスの終了 (削除)] を選択します。
2. 終了を確認する:

   開いた [インスタンスの終了 (削除)] ウィンドウで、[終了 (削除)] ボタンを選択して、インスタンスを終了することを確認します。
3. インスタンスの状態をモニタリングする:

   [インスタンス] ページで、[インスタンスの状態] 列を確認します。インスタンスの状態が [シャットダウン] に変更されます。全文が表示されない場合は、列を拡大してみてください。

   インスタンスがシャットダウンすると、Amazon EC2 はインスタンスを削除し、[インスタンス] ページから消えます。

重要なポイント

このチュートリアルでは、次の主要な概念について説明しました。

- インスタンスは、AWS クラウド内の Amazon EC2 ウェブサーバーを指します。
- 起動は、EC2 インスタンスの作成を指します。
- 終了は、EC2 インスタンスの削除を指します。
- EC2 インスタンス起動ウィザードには、インスタンス設定のデフォルト値が含まれているため、インスタンスをすばやく簡単に起動できます。

- インスタンス ID はインスタンスに自動的に割り当てられた一意の識別子であり、インスタンス名は識別を容易にするために割り当てることができるオプションのタグです。

次のステップ

インスタンスの起動と終了について自信を持てるように、このチュートリアルのステップを繰り返すことを検討してください。無料利用枠の特典を維持するために、起動したインスタンスは必ずすべて終了してください。

これらの基本に慣れたら、次のチュートリアルに進みます。次のチュートリアルでは、キーインスタンス設定フィールドについて詳しく説明します。

チュートリアル 2: テスト EC2 インスタンスを起動して接続する

チュートリアルの目的	テスト目的で使用できる Amazon EC2 インスタンスを起動する方法について説明します。このインスタンスには高度な設定はなく、機密情報は保存されません。また、基本的なインスタンス設定、インスタンスへの接続方法、およびインスタンスを停止する方法についても説明します。
EC2 エクスペリエンス	初心者
期間	30 分
コスト	無料利用枠の対象 AWS アカウントを作成すると、AWS 無料利用枠を利用して、Amazon EC2 を無料で使い始めることができます。 AWS アカウントを作成したのが 2025 年 7 月 15 日より前で、12 か月が経過しておらず、Amazon EC2 の無料利用枠の特典をまだ使い切っていない場合、無料利用枠の特典に収まるオプションを選択できるように支援いたしますので、このチュートリアルを完了するのに一切料金はかかりません。それ以外の場

	合、インスタンスを起動したとき (アイドル状態のままであっても) から終了するまで、標準の Amazon EC2 使用料金が発生します。 2025 年 7 月 15 日以降に AWS アカウントを作成し、6 か月が経過しておらず、すべてのクレジットを使い切っていない場合、無料利用枠の特典に収まるオプションを選択できるように支援いたしますので、このチュートリアルを完了するのに一切料金はかかりません。 無料利用枠の対象となるかどうかを判断する手順については、「the section called “無料利用枠の使用状況の追跡”」を参照してください。
前提条件	チュートリアル 1: 最初の Amazon EC2 インスタンスを起動する を完了します。

チュートリアルの概要

このチュートリアルは、テスト目的で使用できる EC2 インスタンスを起動したい初心者向けに設計されています。

キーインスタンス設定フィールドについて説明し、EC2 コンソールのデフォルト値を使用してテストインスタンスを起動するステップについて説明します。インスタンスを起動したら、インスタンスにログイン (接続と呼びます) する方法を説明します。このチュートリアルでは、インスタンスへの接続に必要なキーペアを作成する方法も説明します。最後に、コストの管理に役立つよう、使用料の発生を避けるためにインスタンスを停止する方法を説明します。

このチュートリアルでは、Linux インスタンスを起動します。このチュートリアルのステップは、他のオペレーティングシステムでインスタンスを起動するために使用できますが、インスタンスに接続する手順は Linux インスタンスに固有です。

このチュートリアルは、次の短いタスクで構成されます。次のタスクに進む前に、各タスクを完了する必要があります。

- タスク 1: インスタンスを起動するための主要なコンポーネントを理解する
- タスク 2: 技術図を確認する
- タスク 3: キーペアを作成する

- タスク 4: テストインスタンスを起動する
- タスク 5: インスタンスを検索する
- タスク 6: インスタンス設定を表示する
- タスク 7: インスタンスに接続するためのキーコンポーネントを理解する
- タスク 8: インスタンスに接続する
- タスク 9: インスタンスを停止する

タスク 1: インスタンスを起動するための主要なコンポーネントを理解する

このタスクでは、EC2 インスタンスの起動に必要な主要なコンポーネントについて説明します。これらは、AMI、インスタンスタイプ、キーペア、セキュリティグループ、ネットワーク (VPC とサブネット)、Amazon EBS ボリュームです。また、オプションのコンポーネントである [名前] タグについても説明します。

これらのコンポーネントをイメージしやすいように、インスタンスを賃貸住宅に例えてみましょう。住宅を借りることで、不動産を所有したり維持したりしなくても生活できる場所が得られるのと同様に、EC2 インスタンスを使用することで、基盤となるインフラストラクチャを所有したり維持したりすることなくコンピューティング能力を得ることができます。

起動するインスタンスの種類を決定するときは、住宅に求める条件について検討するのと同様に、インスタンスの設定条件について検討します。この例えは物事を単純化していますが、コンポーネントに慣れるまでは、コンポーネントをイメージするのに役立つ方法となります。

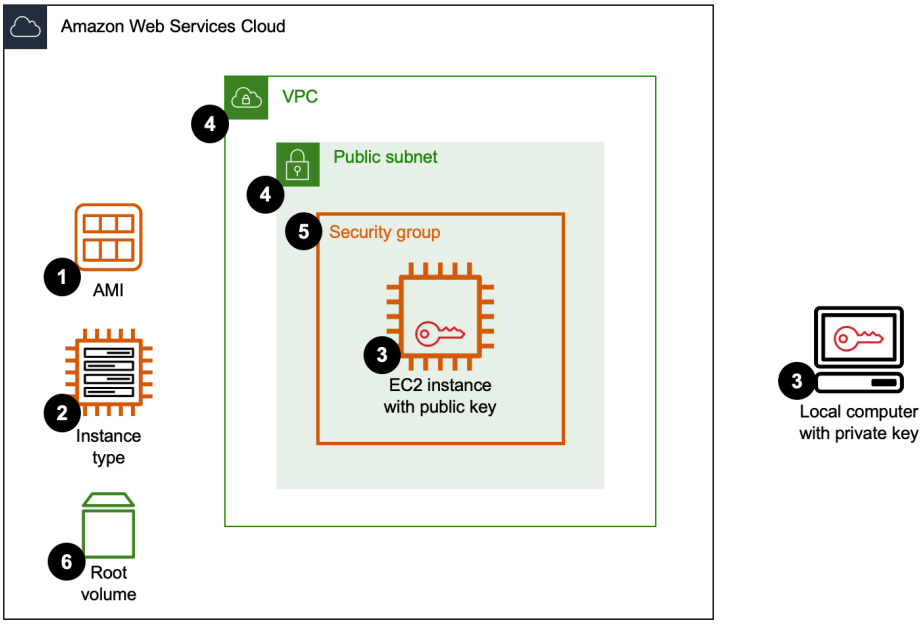
- AMI – 住宅の建築資材とアメニティ: Amazon マシンイメージ (AMI) は、インスタンスが開始するオペレーティングシステムとアプリケーションを決定します。これは、住宅の建築資材 (レンガ、スチール、木など) とアメニティ (電化製品や家具など) を選択するようなものです。ベース AMI は基本的な電化製品を備えた家具のない住宅のようなもので、ソフトウェアがプリインストールされたカスタム AMI は家具付きの住宅のようなものです。
- インスタンスタイプ – 住宅のサイズと電力: インスタンスタイプは、住宅のサイズ、部屋数、エネルギーキャパシティを選択するのと同様に、EC2 インスタンスのサイズと機能を定義します。各インスタンスタイプは、インスタンスの CPU、メモリ、ストレージ、ネットワークキャパシティを決定します。選択した AMI は、選択できるインスタンスタイプを制限する場合があります。
- キーペア – フロントドアキー: キーペアは、住宅のフロントドアのロックとキーのようなものです。パブリックキーはインスタンスのロックとして機能し、プライベートキーはローカルコンピュータで安全に保持する必要があるキーです。他のユーザーがプライベートキーを持っている場

合、そのユーザーはあなたのインスタンスにアクセスすることができます。これは、フロントドアキーを持っている人があなたの家に入ることができるのと同様です。

- ネットワーク (VPC とサブネット) – 敷地の境界、区画された工リア、および住宅番号: 仮想プライベートクラウド (VPC) は、住宅がある敷地全体のようなもので、サブネットは住宅周辺の区画された工リアです。敷地に複数の住宅 (インスタンス) がある場合は、目的に応じて異なる工リア (異なるサブネット) に区画することができます。訪問者が庭を自由に歩き回ることができる住宅 (インターネットアクセスのあるパブリックサブネット) もあれば、立ち入りを制限するために柵で囲まれた庭がある住宅 (インターネットアクセスのないプライベートサブネット) もあります。各サブネットには、住宅番号と同様に、サブネット内のインスタンスに割り当てることができるさまざまな IP アドレスが含まれています。
- セキュリティグループ - 門番: セキュリティグループは門番のような役目を果たし、誰が住宅を訪問できるかを管理します。これにより、インスタンスに到達できるトラフィックを制御する一連のルールが適用されます。例えば、特定の IP アドレスからの SSH トラフィックを許可するルールは、食料品の配達に特定の人のみを許可する門番のようなものです。同様に、任意の場所からの HTTPS トラフィックを許可するのは、一般の人々が住宅の外観を見に来るのを許可するようなものです。
- Amazon EBS ボリューム – ストレージユニット: EBS ボリュームは、持ち物を保存できるストレージユニットのようなものです。各インスタンスにはルートボリューム (AMI が保存される場所) があり、必要に応じていつでもボリューム (ストレージ) を追加できます。
- 名前タグ – 家の名前: [名前] タグは、住んでいる人を簡単に特定できるようにする住宅の標識のような役目を果たします。[名前] タグを使用すると、インスタンスの区別が容易になりますが、インスタンスを起動するときには必要ありません。

タスク 2: 技術図を確認する

このタスクでは、AWS ドキュメントで使用する一般的な技術図について学びます。次の図は、このチュートリアルで起動するテストインスタンスの設定を示しています。前のタスクでは、賃貸住宅の例えを使用してこれらのコンポーネントを紹介しました。ここでは、実際の EC2 コンポーネント自体に焦点を当てます。番号付きのラベルは、次の説明に対応しています。



1. AMI – AMI は、インスタンスの起動時に選択するイメージです。これは、インスタンス上で実行するオペレーティングシステムとソフトウェアを含むテンプレートです。例えば、Linux インスタンスを起動する場合は、Amazon Linux 2023 AMI を選択できます。または、Windows インスタンスを起動する場合は、Microsoft Windows Server 2022 Base AMI を選択できます。Amazon EC2 コンソールの AMI カタログには、選択できる数千のイメージが含まれています。
2. インスタンスタイプ – インスタンスタイプは、インスタンスに使用されるホストコンピュータの CPU、メモリ、ストレージ、およびネットワークキャパシティを決定するハードウェアです。Amazon EC2 には、ハードウェアの設定とサイズがそれぞれ異なる 600 を超えるインスタンスタイプから選択できるため、アプリケーションのニーズに最適なインスタンスを選択できます。
3. キーペア – キーペアは、インスタンスへの接続時にユーザーの ID を証明するのに使用する一連のセキュリティ認証情報です。パブリックキーはインスタンス上、プライベートキーはローカルコンピュータ上にあります。

   EC2 では、インスタンスへの接続は、ローカルコンピュータからインスタンスにログインすることを指します。インスタンスに安全に接続する方法は他にもありますが、このチュートリアルではキーペアを使用します。
4. ネットワーク – ネットワークは VPC と 1 つ以上のサブネットで構成されます。VPC は、AWS クラウド 内の仮想ネットワークです。すべての AWS のお客様は、AWS アカウント専用の独自の VPC を持っています。インスタンスを VPC のサブネットに起動します。サブネットは、VPC 内の IP アドレスの範囲です。デフォルトのサブネットはパブリックサブネットです。つまり、パブ

リック IP アドレスを割り当て、Amazon ネットワーク外からインスタンスへのインターネットアクセスを提供します。

5. セキュリティグループ – セキュリティグループは、インスタンスへのトラフィックを制御するファイアウォールとして機能します。セキュリティグループには、特定のタイプのトラフィックがインスタンスに入ることを許可するルールが含まれています。ローカルコンピュータからインスタンスに SSH 経由で (キーペアを使用して) 接続するには、ローカルコンピュータからの SSH トラフィックを許可するルールが必要です。
6. EBS ボリューム – Amazon EBS ボリュームは、物理ハードドライブのように機能するストレージデバイスです。インスタンスにはルートボリュームが付属しています。これは、インスタンスの起動に必要なオペレーティングシステムとソフトウェアに AMI を保存する特殊な EBS ボリュームです。任意でデータボリュームを追加できます。ただし、テストインスタンスは機密データを保存しないため、暗号化されたデータボリュームを追加する必要はありません。

おめでとうございます。このチュートリアルの概念的なタスクが完了しました。以下のタスクでは、Amazon EC2 コンソールを使用して、学習したコンポーネントを作成します。

タスク 3: キーペアを作成する

このタスクでは、キーペアを作成します。キーペアは、インスタンスに追加するパブリックキーと、インスタンスに安全に接続するために使用する対応するプライベートキーの 2 つの部分で構成されます。次のタスクでは、インスタンスの起動時にこのキーペアを選択します。これにより、パブリックキーが自動的にインスタンスに追加されます。プライベートキーにアクセスできるユーザーはインスタンスに接続できるため、ローカルコンピュータにプライベートキーを安全に保存することが重要です。

テストインスタンスの起動時に既存のキーペアを使用する場合は、このタスクをスキップしてください。それ以外の場合は、新しいキーペアの作成に進みます。

開始する前に

管理者ユーザーで AWS マネジメントコンソールにサインインするなど、前の表に記載されている前提条件を満たしていることを確認してください。

キーペアを作成するには、次のステップに従います

1. Amazon EC2 コンソールを開く:

   https://console.aws.amazon.com/ec2/ に移動します。

2. キーペアコンソールページに移動する:

   ナビゲーションペインの [Network & Security] で、[Key Pairs] を選択してください。

   - 以前にキーペアを作成した場合は、テーブルに表示されます。
   - キーペアが存在しない場合、テーブルは空になります。

3. 新しいキーペアを作成する:

   [キーペアの作成] ボタン (右上) を選択して [キーペアの作成] ウェブベースフォームを開き、次のようにキーペアの詳細を入力します。

   a. キーペアに名前を付ける: [名前] には、**`test-instance-key-pair`** などのキーペアを認識するのに役立つ名前を入力します。

      名前には、最大 255 文字の ASCII 文字を含めることができます。先頭または末尾にスペースを含めることはできません。

   b. キーペアタイプを選択する: [キーペアタイプ] で、[ED25519] を選択します。

      Linux インスタンスは RSA と ED25519 の両方のキータイプをサポートしますが、Windows インスタンスは RSA のみをサポートします。このチュートリアルでは Linux インスタンスを起動するため、ED25519 キーを使用できます。

   c. プライベートキーファイル形式を選択する: [プライベートキーファイル形式] で、[.pem] を選択します。

      これは、プライベートキーファイルを保存する形式です。

4. パブリックキーを Amazon EC2 に保存し、プライベートキーをダウンロードする:

   [キーペアの作成] ボタン (右下) を選択します。

   Amazon EC2 はパブリックキーを保存し、ブラウザはプライベートキーファイルを自動的にローカルコンピュータにダウンロードします。ファイルはキーペアに指定した名前に従って名前が付けられ、拡張子は選択したファイル形式です。プライベートキーファイルをコンピュータ上の安全な場所に移動します。

   > **Important**
   > プライベートキーファイルを保存できるのは、このタイミングだけです。

5. キーに対するアクセス許可を設定する (macOS および Linux ユーザーの場合):

macOS または Linux コンピュータで SSH を使用してインスタンスに接続する場合は、プライベートキーファイルに正しいアクセス許可を設定する必要があります。ターミナルウィンドウを開き、次のコマンドを実行し、*test-instance-key-pair* をキーペアの名前に置き換えます。

```
chmod 400 test-instance-key-pair.pem
```

このコマンドにより、インスタンスへの安全な接続を確立するために必要なプライベートキーファイルを読み取ることができるのは自分だけになります。これらのアクセス許可がないと、このキーペアを使用して接続することはできません。

おめでとうございます。キーペアが正常に作成されました。

タスク 4: テストインスタンスを起動する

このタスクでは、EC2 インスタンス起動ウィザードを使用してテストインスタンスをすばやく起動します。Linux インスタンスのメインインスタンス設定を設定し、他のフィールドにはデフォルト値を使用します。

コストを管理するために、[無料利用枠対象] コンポーネントを選択することをお勧めします。

テストインスタンスを起動するには、次のステップに従います

1. Amazon EC2 コンソールを開く:

   https://console.aws.amazon.com/ec2/ に移動します。
2. EC2 インスタンス起動ウィザードを開く:

   EC2 ダッシュボードから、[インスタンスの起動] を選択します。

   [インスタンスの起動] ウェブベースのフォームが開きます。これが EC2 インスタンス起動ウィザードです。
3. インスタンスに名前を付ける:

   [名前とタグ] の [名前] に、**Test instance** のようなわかりやすい名前を入力します。

   インスタンス名はタグで、キーは [Name] (名前)、値は指定した名前です。

ヒント: テストインスタンスの場合は、名前タグで十分です。ただし、本番稼働インスタンスでは、タグ付けポリシーを確立して、すべてのリソースでタグ付けを標準化することをお勧めします。

4. オペレーティングシステムとソフトウェア - Amazon マシンイメージ (AMI) を選択する:

   [アプリケーションイメージと OS イメージ (Amazon マシンイメージ)] の [Amazon マシンイメージ (AMI)] で、デフォルトの選択は [Amazon Linux 2023 AMI] です。この AMI は [無料利用枠対象] としてマークされています。このチュートリアルでは Linux インスタンスを起動するため、無料利用枠の制限内に留まるようにデフォルト設定を保持します。

5. ハードウェア - インスタンスタイプを選択する:

   このチュートリアルでは、[インスタンスタイプ] の [インスタンスタイプ] で、デフォルト選択をそのまま使用します。デフォルトのインスタンスタイプは無料利用枠で使用でき、そのハードウェアはテストインスタンスに適しています。

6. キーペアを使用した安全なログインの準備をする:

   [キーペア (ログイン)] の [キーペア名] で、前のタスクで作成したキーペアを選択します。リストにキーペアが表示されない場合は、更新アイコン (リストの右側) を選択します。

   インスタンスが起動すると、パブリックキーがインスタンスに配置されます。起動後にインスタンスに接続するには、前のタスクでダウンロードした対応するプライベートキーを使用します。

7. インターネットアクセスを有効にするようにネットワーク設定を設定する:

   [ネットワーク設定] で、[ネットワーク] (VPC) フィールドと [サブネット] フィールドがデフォルトで設定されます。このチュートリアルでは、すぐに開始できるようにデフォルト設定をそのまま使用します。デフォルトのサブネットを変更していない場合、インスタンスはインターネットにアクセスできます。

   ヒント: デフォルトのサブネットはパブリックサブネットです。つまり、パブリック IP アドレスを割り当て、Amazon ネットワーク外からインスタンスへのインターネットアクセスを提供します。テストインスタンスの場合、インターネットアクセスを提供するデフォルトのサブネット設定を使用しても問題ありません。ただし、本番稼働インスタンスでは、絶対に必要な場合にのみパブリック IP アドレスを割り当て、インターネットアクセスを備えたサブネットを使用することをお勧めします。

8. インスタンスファイアウォール (セキュリティグループ) を設定する:

[ネットワーク設定] の [ファイアウォール (セキュリティグループ)] で、[任意の場所 (0.0.0.0) から SSH トラフィックを許可] チェックボックスをオンのままにします。これにより、任意の IP アドレスからの SSH トラフィックを許可するテストインスタンスの新しいセキュリティグループが作成されます。

セキュリティグループは、インスタンスへのトラフィックを制御するファイアウォールとして機能します。ローカルコンピュータからインスタンスに SSH 経由で接続するには、ローカルコンピュータからの SSH トラフィックを許可するルールが必要です。

ヒント: インターネットサービスプロバイダーが動的 IP 割り当てを使用している場合、ローカルコンピュータの IP アドレスは時間の経過とともに変更される可能性があります。テスト目的でインスタンスを使用する場合、インスタンスを使用して機密情報を保存しないことを前提としているため、セキュリティ対策の制限が緩和されている可能性があります。テストインスタンスでは、IP アドレスが変更されても常に接続できるように、任意の IP アドレス (`0.0.0.0/0`) からのトラフィックを許可することは一般的に許容されます。ただし、本番稼働インスタンス、特に機密データを持つインスタンスでは、既知の IP アドレスからのトラフィックのみを許可することがベストプラクティスです。

9. インスタンスストレージを設定する:

   [ストレージの設定] で、[ルートボリューム (暗号化)] フィールドがデフォルトで設定されます。無料利用枠の制限内に収まるように、設定はそのままにしておきます。

   テストインスタンスは機密データを保存しないため、暗号化されたデータボリュームを追加する必要はありません。

10. インスタンス設定を確認する:

    右側の [概要] パネルでは、インスタンスを起動する前に、概要レベルの設定を確認できます。

11. インスタンスを起動する:

    インスタンスを起動する準備ができたら、[概要] パネルで [インスタンスの起動] を選択します。

    Amazon EC2 は、指定した設定を使用してインスタンスをすばやく起動します。設定を指定しなかった場合は、デフォルトが使用されます。[成功] バナーは、インスタンスが起動されたことを示します。

おめでとうございます。テストインスタンスが正常に起動されました。

タスク 5: インスタンスを検索する

このタスクでは、EC2 コンソールで起動したインスタンスを見つけます。

EC2 コンソールでインスタンスを検索するには、次のステップに従います

1. [インスタンス] ページを開く:

   成功ページがまだ表示されている場合は、[成功] バナーでインスタンス ID を選択します。

   移動した場合は、ナビゲーションペインから [インスタンス] を選択します。
2. インスタンスを見つける:

   [名前] 列で、指定した名前でインスタンスを検索します。

タスク 6: インスタンス設定を表示する

このタスクでは、インスタンスの設定の詳細表示について理解します。

インスタンスの設定を表示するには、次のステップに従います

1. インスタンスを見つける:

   [名前] 列で、指定した名前でインスタンスを検索します。
2. インスタンスの詳細ページを開く:

   インスタンス名の横にあるチェックボックスをオンにしたら、[アクション] メニュー (右上) を選択して [詳細を表示] を選択してインスタンスの詳細ページを開き、設定を確認できます。

   前のチュートリアルでは、インスタンスの ID リンクを選択して、インスタンスの詳細ページを開きました。EC2 コンソールでタスクを実行する方法が複数あることがわかります。
3. インスタンス設定の詳細を調べる:

   インスタンスの設定の詳細について、少し時間をかけて確認してください。

   ヒント: フィールドをすばやく見つけるには、キーボードで Ctrl+F キーまたは Command+F キーを押します。

   a. AMI: インスタンスの起動に使用した AMI を見つけられますか。この情報は、[詳細] タブの [AMI ID] と [AMI 名] で確認できます。

b. インスタンスタイプ: インスタンスタイプを見つけられますか。例えば、[t3.micro] などです。

c. キーペア: インスタンスの起動時に選択したキーペアを見つけられますか。これは、[起動時に割り当てられたキーペア] で指定されます。今後キーペアを変更しても、ここでの値は変更されないことに注意してください。

d. VPC: VPC の ID を見つけられますか。[ネットワーク] タブには、ネットワーク関連の設定がすべて表示されます。VPC ID は、次の例のような形式です: vpc-1a2b3c4d

e. サブネット: インスタンスを起動したサブネットの ID を見つけられますか。次の例のような形式です: subnet-1a2b3c4d

f. パブリック IPv4 アドレス: インスタンスに割り当てられたパブリック IPv4 アドレスを見つけられますか。これは、[34.242.148.128] のような形式です。

g. セキュリティグループ: 任意の場所 (0.0.0.0./0) からの SSH トラフィックを許可するように作成されたインバウンドルールを見つけられますか。[セキュリティ] タブには、セキュリティ関連の設定がすべて表示されます。

h. ストレージ: このインスタンス用に作成されたボリュームを見つけられますか。[ストレージ] タブには、ストレージ関連の設定がすべて表示されます。

i. インスタンスタグ: インスタンスに付けた名前は、実際にはタグになります。インスタンスタグを見つけられますか。[タグ] タブを選択します。キーは [名前] で、値は指定した名前です。

j. インスタンスの状態: インスタンスの状態を確認できますか。[実行] になっているはずです。

他のインスタンス設定フィールドについても、少し時間をかけて確認してください。準備ができたら、次のタスクに進みます。

タスク 7: インスタンスに接続するためのキーコンポーネントを理解する

このタスクでは、EC2 インスタンスへの接続に必要な主要なコンポーネントについて説明します。これらは、接続プロトコル、パブリック DNS、セキュリティグループ、キーペア、インスタンスユーザー名です。

これらのコンポーネントをイメージしやすいように、インスタンスに接続することを自宅に行くことに例えてみましょう。

- 接続プロトコル – 移動手段: 帰宅方法を選択するのと同様に、インスタンスに移動するための接続プロトコルを選択します。このチュートリアルでは、SSH (Secure Shell) を使用します。これは、

インターネット経由でコンピュータをインスタンスに接続するための安全なトンネルを作成します。

- パブリック DNS – 自宅の住所: 自宅に一意の住所があるのと同様に、EC2 インスタンスには独自のパブリック DNS 名 (ec2-18-201-118-201.eu-west-1.compute.amazonaws.com など) があります。このパブリック DNS 名により、SSH はインスタンスに直接接続できます。
- セキュリティグループ – 門番: 自宅に、誰が出入りできるかを管理する門番がいると想像してください。同様に、EC2 インスタンスには門番のような役目を果たすセキュリティグループがあり、インスタンスに出入りできるネットワークトラフィックのタイプを制御します。明示的に許可するトラフィック (コンピュータの IP アドレスからの SSH トラフィックなど) のみが許可されます。
- プライベートキー – フロントドアキー: インスタンスを起動したとき、キーペアを指定しました。パブリックキーはインスタンスに配置され、プライベートキーはコンピュータ上に保管しました。プライベートキーはフロントドアキーとして機能します。プライベートキーがなければ、インスタンスに入ることはできません。
- インスタンスユーザー名 – 居住者: 自宅に着いたら、自分が居住者であることを証明するために身分証明をする必要があります。同様に、インスタンスに接続するときは、ユーザー名を指定します。オペレーティングシステムに応じて、インスタンスごとに異なるデフォルトのユーザー名が付けられます。例えば、Amazon Linux インスタンスはデフォルトのユーザー名として ec2-user を使用します。

接続コマンド

EC2 インスタンスに接続するには、ターミナルウィンドウで次のコマンドを使用します。

```
ssh -i "test-instance-key-pair.pem" ec2-user@ec2-18-201-118-201.eu-
west-1.compute.amazonaws.com
```

コマンドの動作の内訳は次のとおりです。

- ssh – このコマンドは接続プロトコルを指定し、インスタンスへの SSH (Secure Shell) 接続を開始します。
- -i "test-instance-key-pair.pem" – -i フラグは、接続の認証に必要なプライベートキーファイルを示します。このプライベートキーファイルは、インスタンスの起動時に指定したキーペアと一致する必要があります。プライベートキーファイルが特定のフォルダに保存されている場合は、ファイルへのフルパスを指定します。

- `ec2-user` – これはインスタンスにログインするためのユーザー名です。Amazon Linux インスタンスの場合、デフォルトのユーザー名は `ec2-user` です。他の AMI では、異なるデフォルトユーザー名が使用される場合があります (例: Ubuntu インスタンスの場合は ubuntu)。
- `@` – この記号は、ユーザー名をインスタンスのアドレスから区切ります。
- `ec2-18-201-118-201.eu-west-1.compute.amazonaws.com` – これはインスタンスのパブリックアドレス (パブリック DNS) で、パブリック IPv4 アドレスと AWS リージョンが含まれます。インスタンスを一意に識別します。

コマンドを実行すると起きること

コマンドを実行すると、SSH は安全なトンネルを確立し、プライベートキーで認証します。インスタンスのセキュリティグループがトラフィックを許可すると、EC2 インスタンスにアクセスできます。インスタンスの目の前に座っているかのように、コンピュータからインスタンスを制御できるようになりました。ローカルマシン上で行うのと同様に、コマンドの実行、ソフトウェアのインストール、ファイルの管理を行うことができます。

タスク 8: インスタンスに接続する

このタスクでは、コンピュータの SSH クライアントを使用してインスタンスに接続します。前のタスクでは、自宅に行く例を使って、インスタンスに接続するためのコンポーネントを紹介しました。ここでは、実際の EC2 インスタンスへの接続に焦点を当てます。

インスタンスに接続するには、さまざまな方法があります。接続に使用する方法は、インスタンスのオペレーティングシステムによって異なります。Linux インスタンスを起動してから、ローカルコンピュータで SSH クライアントを使用します。

まず、コンピュータに SSH クライアントがインストールされているかどうかを確認します

ほとんどのコンピュータには、SSH クライアントがプリインストールされています。確認するには、コンピュータでターミナルウィンドウを開き、次のコマンドを実行します。

```
ssh
```

コマンドが認識されれば、接続する準備は完了です。

コマンドが認識されない場合は、SSH クライアントをインストールする必要があります。SSH クライアントをインストールする手順は、このチュートリアルの範囲外です。サポートが必要な場合は、このユーザーガイドの「SSH 接続の前提条件」を参照するか、オペレーティングシステムに SSH クライアントをインストールする手順についてオンラインで検索してください。

インスタンスに接続するには、次のステップに従います

1. 接続を開始する:

   Amazon EC2 コンソールのインスタンスの詳細ページが表示されている場合は、[接続] ボタン (右上) を選択します。

   移動した場合は、ナビゲーションペインから [インスタンス] を選択します。次に、[インスタンス] ページで、インスタンス名の横にあるチェックボックスをオンにし、[接続] ボタン (右上) を選択します。

   これにより、[インスタンスへの接続] ページが開きます。
2. 接続方法を選択する:

   [インスタンスに接続] ページで、[SSH クライアント] タブを選択してください。

   このページのテキストを確認してください。これは次に実行するステップです。
3. SSH コマンドを確認する:

   [例] には、インスタンスの詳細を使用して自動的に生成およびカスタマイズされるコマンドが表示されます。プライベートキー名は、起動時に指定されたパブリックキーの名前から取得されます。

   コマンドは次のようになります。

   ```
   ssh -i "test-instance-key-pair.pem" ec2-user@ec2-18-201-118-201.eu-
   west-1.compute.amazonaws.com
   ```
4. SSH コマンドをコピーする:

   サンプル SSH コマンドの横にあるコピーアイコンを選択します。
5. ターミナルウィンドウを開く:

   ローカルコンピュータでターミナルウィンドウを開きます。
6. SSH コマンドを貼り付けて実行する:

   SSH コマンドをターミナルウィンドウに貼り付けます。プライベートキーファイルを特定のフォルダに保存した場合は、コマンドを編集してファイルパス全体を含めます。

   キーボードで Enter キーを押します。

次のようなレスポンスが表示されます。

```
The authenticity of host 'ec2-18-201-118-201.eu-west-1.compute.amazonaws.com
 (18-201-118-201)' can't be established.
ED25519 key fingerprint is SHA256:examplehxj9aOr1MogvKOoMNskVVIRBQBoq0example.This
 key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])?
```

7. 接続を完了する:

   yes と入力し、キーボードの Return キーを押します。

   フィンガープリントの検証は、このチュートリアルの範囲外です。詳細については「(オプション) インスタンスのフィンガープリントを取得する」を参照してください。

   接続が成功すると、ターミナルプロンプトが変わり、インスタンスのパブリック DNS が表示されます。

おめでとうございます。インスタンスに正常に接続されました。

タスク 9: インスタンスを停止する

このタスクでは、インスタンスを停止して、無料利用枠の特典を維持します。インスタンスが停止すると、そのインスタンスに対するコストが発生しなくなります。2025 年 7 月 15 日より前に AWS アカウント を作成し、無料利用枠の対象である場合は、EBS ストレージのコストが引き続き発生します。

インスタンスを停止するには、次のステップに従います

1. 停止を開始する:

   [インスタンスに接続] ページがまだ表示されている場合は、パンくずリストから [インスタンス] を選択します。移動した場合は、ナビゲーションペインから [インスタンス] を選択します。

   次に、[インスタンス] ページで、インスタンスの名前の横にあるチェックボックスを選択し、[インスタンス状態] メニュー (右上) を選択し、[インスタンスを停止] を選択します。プロンプトが表示されたら、[停止] を選択します。

2. インスタンスの状態をモニタリングする:

[インスタンス] ページで、[インスタンスの状態] 列を確認します。インスタンスの状態が [停止中] に変わり、次に [停止済み] に変わります。全文が表示されない場合は、列を拡大してみてください。

インスタンスの状態が [停止中] から [停止済み] に変更されたと思われるが、まだ表示されない場合は、更新アイコン (テーブルの上) を選択して [インスタンス] テーブルを更新します。

重要なポイント

このチュートリアルでは、次の主要な概念について説明しました。

- AMI は、Amazon マシンイメージを指します。これは、インスタンスの起動に必要なオペレーティングシステムとソフトウェアを含むテンプレートです。
- インスタンスタイプは、インスタンスに使用されるホストコンピュータのハードウェアを指します。インスタンスの CPU、メモリ、ストレージ、ネットワークキャパシティを決定します。
- キーペアは、インスタンスに安全に接続するために使用できるパブリックキーとプライベートキーのセットを指します。
- ネットワークは、VPC (AWS クラウド内のアカウント専用の仮想プライベートクラウド) とサブネット (VPC 内の IP アドレスの範囲) を指します。
- セキュリティグループは、インスタンスに到達できるトラフィックを制御する一連のルールを指します。
- EBS ボリュームは、インスタンスのデータストレージを指します。すべてのインスタンスには、AMI を保存するためのルートボリュームと 1 つ以上のオプションのデータボリュームがあります。
- タグは、オプションでインスタンスに割り当てることができるメタデータです。インスタンス名はタグで、その [キー] は [名前] であり、[値] はユーザーが選択できます。
- 接続は、インターネット経由でインスタンスにアクセスすることを指します。
- SSH は、インスタンスへの接続に使用できる Secure Shell 接続プロトコルを指します。
- パブリック DNS は、インスタンスの一意のパブリックアドレスです。
- インスタンスユーザー名は、インスタンスのオペレーティングシステムによって決定され、接続に必要です。
- インスタンスを停止すると、インスタンスに対する料金は停止しますが、EBS ストレージ料金は継続されます。

次のステップ

インスタンスの起動、接続、および停止について自信を持てるように、このチュートリアルのステップを繰り返すことを検討してください。無料利用枠の特典を維持するために、起動したインスタンスは必ずすべて終了してください。

これらの基本に慣れたら、より高度なチュートリアルを試すことができます。その他のチュートリアルについては、「他のチュートリアルをお探しですか。」を参照してください

2025 年 7 月 15 日より前に AWS アカウントを作成した場合、次の 6 分間の動画を見ることを検討してください: How can I avoid charges on my account when using AWS 無料利用枠 services。

2025 年 7 月 15 日以降に AWS アカウントを作成した場合、「AWS Billing ユーザーガイド」の「Explore AWS services with AWS Free Tier」を参照してください。

Amazon EC2 インスタンス設定パラメータのリファレンス

Amazon EC2 コンソールのインスタンス起動ウィザードと起動テンプレートには、Amazon EC2 インスタンスを設定するためのすべてのパラメータが用意されています。

キーペアを除き、インスタンス起動ウィザードは各パラメータのデフォルト値を提供します。デフォルトの一部またはすべてを受け入れるか、独自の値を使用してインスタンスを設定することができます。起動テンプレートを作成する場合、パラメータはオプションです。起動テンプレートを使用してインスタンスを起動する場合、起動テンプレートで指定されたパラメータは、インスタンス起動ウィザードのデフォルト値を上書きします。起動テンプレートで指定されていないパラメータは、デフォルトでインスタンス起動ウィザードで指定された値になります。

パラメータは、インスタンス起動ウィザードと起動テンプレートでグループ化されます。以下の説明は、コンソールのパラメータグループに従って表示されます。

インスタンス設定のパラメータ

- 名前とタグ
- アプリケーションと OS イメージ (Amazon マシンイメージ)
- インスタンスタイプ
- キーペア (ログイン)
- ネットワーク設定
- ストレージの設定
- 高度な詳細
- 概要

名前とタグ

インスタンス名はタグで、キーは [Name] (名前)、値は指定した名前です。インスタンス、ボリューム、ネットワークインターフェイスにタグ付けできます。スポットインスタンスの場合、スポットインスタンスリクエストにのみタグを付けることができます。タグの詳細については、Amazon EC2 リソースのタグ付けを参照してください。

インスタンス名と追加のタグを指定することはオプションです。

- [Name] (名前) に、インスタンスのわかりやすい名前を入力します。名前を指定しない場合は、インスタンスをその ID で識別できます。ID は、インスタンスの起動時に自動的に生成されます。
- タグを追加するには、[Add additional tag] (追加のタグを追加) を選択します。[Add tag] (タグを追加) を選択し、キーと値を入力し、タグ付けするリソースタイプを選択します。追加するタグごとに [Add tag] (タグの追加) を選択してください。

インスタンス名はインスタンスの起動時にのみ指定できます。起動テンプレートの作成時にインスタンスに名前を付けることはできませんが、インスタンスの起動時に作成されるリソースにタグを追加できます。

アプリケーションと OS イメージ (Amazon マシンイメージ)

Amazon マシンイメージ (AMI) には、インスタンスの作成に必要な情報が含まれています。例えば、ある AMI には、ウェブサーバーとして動作するのに必要なソフトウェア (Linux、Apache、ウェブサイトなど) が含まれていたりします。

適切な AMI は、次の手順で確認できます。AMI を検索する各オプションで、[Cancel] (キャンセル) (右上) を選択すれば、AMI を選択せずにインスタンス起動ウィザードに戻ることができます。

検索バー

利用可能なすべての AMI を検索するには、AMI 検索バーにキーワードを入力し、[Enter] キーを押します。AMI を選択するには、[Select] (選択) を選択します。

Recents (最新情報)

最近使用した AMI が表示されます。

[Recently launched] (最近の起動) または [Currently in use] (現在使用中) を選択し、[Amazon Machine Image (AMI)] (Amazon マシンイメージ (AMI)) から AMI を選択します。

マイ AMI

お客様が所有しているプライベート AMI、またはお客様が共有しているプライベート AMI。

[Owned by me] (ユーザーによる所有) または [Shared with me] (共有されている) を選択し、[Amazon Machine Image (AMI)] (Amazon マシンイメージ (AMI)) から AMI を選択します。

クイックスタート

AMI はオペレーティングシステム (OS) ごとにグループ化されているため、すぐに作業を開始できます。

まず、必要な OS を選択し、次に [Amazon Machine Image (AMI)] (Amazon マシンイメージ (AMI)) で、AMI を選択します。AWS 無料利用枠で使用できる AMI を選択するには、AMI が [無料利用枠の対象] とマークされていることを確認してください。

[Browse more AMI] (AMI をさらに表示する)

AMI カタログ全体を表示するには、[Browse more AMI] (AMI をさらに表示する) を選択します。

- 利用可能な AMI すべてを検索するには、検索バーにキーワードを入力し、[Enter] キーを押します。
- Systems Manager パラメータを使用して AMI を検索するには、検索バーの右側にある矢印ボタンを選択し、[Search by Systems Manager parameter] (Systems Manager パラメータで検索) を選択します。詳細については、「Systems Manager パラメータを使用して AMI を参照する」を参照してください。
- カテゴリで検索するには、[Quickstart AMI] (AMI のクイックスタート)、[My AMI] (私の AMI)、[AWS Marketplace AMI]、または[Community AMI] (コミュニティ AMI) を選択します。

  AWS Marketplace は、AMIを含む AWS 上で動作するソフトウェアを購入することができるオンラインストアです。AWS Marketplace からのインスタンスの起動の詳細については、AWS Marketplace AMI から Amazon EC2 インスタンスを起動する を参照してください。[Community AMI] (コミュニティ AMI)では、AWS のコミュニティのメンバーが他の人が利用可能とした AMI を見つけることができます。Amazon または検証済みパートナーからの AMI は、[Verified provider] (検証済みプロバイダー) のマークが付されます。
- AMIのリストをフィルターするには、画面左の [Refine results] (結果を絞り込む) で1つまたは複数のチェックボックスをオンにします。フィルターオプションは、選択した検索カテゴリに応じて異なります。
- 各 AMI の [Root device type] を確認します。必要なタイプはどの AMI かに注意してください。タイプは [ebs] (Amazon EBS でバックアップ) または [instance-store] (インスタンスストアでバックアップ) です。詳細については、ルートボリュームタイプを参照してください。

- 各 AMI の [Virtualization type] を確認します。必要なタイプはどの AMI かに注意してください。タイプは [hvm] または [paravirtual] です。例えば、一部のインスタンスタイプには HVM が必要です。Linux 仮想化タイプの詳細については、「仮想化タイプ」を参照してください。
- 各 AMI に記載された [Boot Mode] (起動モード) を確認します。必要なブートモードがどの AMI を使用するのか注意してください。必要なブートモードは [legacy-bios]、[uefi]、または [uefi-preferred] です。詳細については、「Amazon EC2 ブートモードでのインスタンスの起動動作」を参照してください。
- ニーズを満たす AMI を選択し、[Select] を選択します。

AMI を変更するときに警告します

インスタンスを起動するときに、選択した AMI に関連付けられているボリュームまたはセキュリティグループの設定を変更し、別の AMI を選択すると、現在の設定の一部が変更または削除されることを警告するウィンドウが開きます。セキュリティグループおよびボリュームに対する変更を確認できます。さらに、追加および削除されるボリュームを表示することも、追加されるボリュームのみを表示することもできます。この警告は、起動テンプレートの作成時には表示されません。

インスタンスタイプ

インスタンスタイプは、インスタンスのハードウェア設定とサイズを定義します。インスタンスタイプが大きくなると、CPU およびメモリも増えます。詳細については、「Amazon EC2 インスタンスタイプ」を参照してください。

- [インスタンスタイプ]: インスタンスタイプが、指定した AMI と互換性があることを確認します。詳細については、「Amazon EC2 インスタンスタイプ」を参照してください。

  無料利用枠 – [無料利用枠の対象] のラベルが付いたインスタンスタイプは、無料利用枠として無料で使用できます。具体的なインスタンスタイプは、AWS アカウントを作成した時期によって異なります。

  2025 年 7 月 15 日より前に AWS アカウントを作成していて、12 か月が経過していない場合は、[t2.micro] インスタンスタイプ、または [t2.micro] が利用できないリージョンで [t3.micro] インスタンスタイプを選択すると、無料利用枠で Amazon EC2 を使用できます。[t3.micro] インスタンスを起動すると、デフォルトで [Unlimited] モードになり、CPU 使用率に基づいて追加料金が発生する可能性があることに注意してください。

2025 年 7 月 15 日以降に AWS アカウント を作成している場合は、t3.micro、t3.small、t4g.micro、t4g.small、c7i-flex.large、および m7i-flex.large インスタンスタイプを 6 か月間、またはクレジットを使い切るまで使用できます。

詳細については、「2025 年 7 月 15 日より前とそれ以降の無料利用枠の特典」を参照してください。

- [Compare instance types] (インスタンスタイプの比較): vCPU の数、アーキテクチャ、メモリ量 (GiB)、ストレージ量 (GB)、ストレージタイプ、ネットワークパフォーマンスなどの属性ごとにさまざまなインスタンスタイプを比較できます。
- [アドバイスの取得]: インスタンスタイプに関するガイダンスやアドバイスは、EC2 インスタンスタイプファインダーから入手できます。詳細については、「EC2 インスタンスタイプファインダーからレコメンデーションを取得する」を参照してください。
- (起動テンプレートのみ) [アドバンスト]: インスタンス属性を指定し、Amazon EC2 がそれらの属性を持つインスタンスタイプを識別できるようにするには、[アドバンスト] を選択してから、[インスタンスタイプの属性を指定する] を選択します。
  - [Number of vCPUs] (vCPU 数): コンピューティング要件に対応する vCPU の最小数と最大数を入力します。制限がないことを示すには、最小値に **0** を入力し、最大値を空白のままにします。
  - Amount of memory (MiB) (メモリの量 (MiB)): コンピューティング要件に対応する最小メモリ量と最大メモリ量を MiB 単位で入力します。制限がないことを示すには、最小値に **0** を入力し、最大値を空白のままにします。
  - コンピューティング要件をより詳細に表現するには、[Optional instance type attributes] (オプションのインスタンスタイプ属性) を展開し、[Add attribute] (属性の追加) を選択します。各属性の詳細については、Amazon EC2 API リファレンスのInstanceRequirementsRequestを参照してください。
  - [Resulting instance types] (インスタンスタイプの結果): 指定した属性に一致するインスタンスタイプをプレビューできます。インスタンスタイプを除外するには、[Add attribute] (属性の追加) を選択し、[Attribute] (属性) リストから [Excluded instance types] (インスタンスタイプの除外) を選択します。[Attribute value] (属性値) リストから、除外したいインスタンスタイプを選択します。

キーペア (ログイン)

[Key pair name] (キーペア名) には、既存のキーペアを選択するか、[Create new key pair] (新しいキーペアを作成) を選択して新しいキーペアを作成します。詳細については、Amazon EC2 のキーペアと Amazon EC2 インスタンスを参照してください。

> **Important**
>
> [Proceed without key pair] (キーペアなしで進む) オプションを選択した場合 (非推奨)、ユーザーが別の方法でログインすることを許可するように設定された AMI を選択した場合でなければ、インスタンスに接続できなくなります。

ネットワーク設定

ネットワーク設定では、インスタンスの IP アドレス、セキュリティグループ、ネットワークインターフェイスが定義されます。デフォルトのネットワーク設定を使用することも、必要に応じて設定することもできます。

- (インスタンス起動ウィザードのみ) [VPC]: インスタンスの既存の VPC を選択します。リージョンのデフォルト VPC はデフォルトで選択されます。作成した VPC または自分と共有された VPC を選択することもできます。詳細については、「EC2 インスタンスの仮想プライベートクラウド」を参照してください。
- [サブネット]: インスタンスのサブネットを選択するか、[新しいサブネットの作成] を選択して、Amazon VPC コンソールを使用して新しいサブネットを作成します。
  - 選択した VPC の任意のアベイラビリティーゾーン、Local Zones、Wavelength Zone、または Outpost にサブネットを作成できます。
  - IPv6 のみのサブネットでインスタンスを起動するには、インスタンスが Nitro ベースのインスタンスである必要があります。
- (インスタンス起動ウィザードのみ) パブリック IP の自動割り当て: パブリック IPv4 アドレスの自動割り当てを有効または無効にします。デフォルトのサブネットにインスタンスを起動する場合、デフォルト値は 有効 です。デフォルト以外のサブネットでインスタンスを起動する場合、デフォルト値は 無効 です。詳細については、「パブリック IPv4 アドレス」を参照してください。

  セカンダリネットワークインターフェイスを追加する場合、デフォルト以外のサブネットに対してこのオプションを有効にすることはできません。詳細については、「the section called “起動時にパブリック IPv4 アドレスを割り当てる”」を参照してください。

- （インスタンス起動ウィザードのみ）IPv6 IP の自動割り当て: IPv6 アドレスの自動割り当てを有効または無効にします。詳細については、「IPv6 アドレス」を参照してください。
- [ファイアウォール (セキュリティ グループ)]: 既存のセキュリティグループを選択するか、新しいセキュリティグループを作成します。セキュリティグループに、インスタンスとの間のトラフィックを許可するルールがあることを確認してください。他のトラフィックはすべて無視されます。

  新しいセキュリティグループを作成すると、SSH (Linux インスタンス) または RDP (Windows インスタンス) 経由ですべての IP アドレスからインスタンスに接続できるインバウンドルールが自動的に作成されます。必要に応じて、このルールを削除または変更できます。必要に応じてルールを追加できます。詳細については、「セキュリティグループルールの設定」を参照してください。

  > **Warning**
  >
  > すべての IP アドレスが SSH または RDP 経由でインスタンスにアクセスできるようにするルールは、テスト用のインスタンスを短時間で立ち上げ、すぐに停止または終了させる場合には許容されます。本番環境では安全ではありません。インスタンスへのアクセスを特定の IP アドレス範囲のみに許可する必要があります。

  このセキュリティグループは、プライマリネットワークインターフェイスとセカンダリネットワークインターフェイスに追加されます。ネットワークインターフェイスに追加のセキュリティグループを選択できます。ただし、ここで選択したセキュリティグループは削除できません。
- [高度なネットワーク設定] — 必要に応じてプライマリネットワークインターフェイスを設定できます。セカンダリネットワークインターフェイスを追加するには、ネットワークインターフェイスを追加するを選択します。追加できるネットワークインターフェイスの数は、選択したインスタンス タイプによって異なります。このセクションは、サブネットを選択した場合にのみ表示されます。
  - [デバイスインデックス]: デバイスインデックス。プライマリ ネットワーク インターフェイスはインデックス 0 に割り当てる必要があります
  - [ネットワークインターフェイス]: ネットワークインターフェイス。[新しいインターフェイス]を選択して Amazon EC2 に新しいインターフェイスを作成させるか、既存の利用可能なネットワークインターフェイスを選択します。プライマリネットワークインターフェイスとして既存のネットワークインターフェイスを選択した場合、デフォルト以外のサブネットに対して[パブリック IP の自動割り当て] を有効にすることはできません。
  - 説明: 新しいネットワークインターフェイスの説明。

- [Subnet] (サブネット): 新しいネットワークインターフェイスを作成するサブネット。インスタンスは、プライマリネットワークインターフェイスと同じサブネットで起動されます。

  セカンダリネットワークインターフェイスのサブネットは、プライマリネットワークインターフェイスのサブネットと同じアベイラビリティーゾーンから選択する必要があります。別の VPC からサブネットを選択すると、ネットワークインターフェイスの横に [マルチ VPC] というラベルが表示されます。これにより、ネットワークとセキュリティの設定が異なる VPC にまたがるマルチホームインスタンスを作成できます。

  EC2 インスタンスを IPv6 専用サブネットで起動するには、Nitro ベースのインスタンスを使用する必要があります。IPv6 専用インスタンスを起動すると、DHCPv6 がインスタンスに IPv6 DNS ネームサーバーをすぐに提供しない可能性があります。この初期遅延の間、インスタンスはパブリックドメインを解決できない可能性があります。設定ファイルを変更して AMI のイメージを再作成すると、ブート時にすぐにファイルに IPv6 DNS ネームサーバーアドレスが含まれるようになります。
- [セキュリティグループ]: ネットワークインターフェイスに関連付けるセキュリティグループ。ネットワークインターフェイスのサブネットと同じ VPC からセキュリティグループを選択する必要があります。
- (起動テンプレートのみ) [パブリック IP の自動割り当て]: インスタンスがパブリック IPv4 アドレスを受け取るかどうかを指定します。デフォルトで、デフォルトのサブネットにあるインスタンスはパブリック IPv4 アドレスを受け取り、デフォルト以外のサブネットにあるインスタンスは受け取りません。[Enable] または [Disable] を選択すると、これがサブネットのデフォルト設定より優先されます。詳細については、パブリック IPv4 アドレス を参照してください。
- [プライマリ IP]: サブネットの範囲からの プライマリプライベート IPv4 アドレス。Amazon EC2 で自分用のプライベート IPv4 を選択するには、空白のままにします。
- [セカンダリ IP]: サブネットの範囲内にある追加のプライベート IPv4 アドレス。[手動で割り当てる] を選択し、IPv4 アドレスを入力します。別の IPv4 アドレスを追加するには、[IP を追加] を選択します。あるいは、[自動的に割り当てる] を選択し、Amazon EC2 によって選択される IPv4 アドレスの数を示す値を入力します。
- (IPv6 のみ) [IPv6 IP]: サブネットの範囲の IPv6 アドレス。[手動で割り当てる] を選択し、IPv6 アドレスを入力します。別の IPv6 アドレスを追加するには、[IP を追加] を選択します。あるいは、[自動的に割り当てる] を選択し、Amazon EC2 によって選択される IPv6 アドレスの数を示す値を入力します。
- [IPv4 Prefixes] (IPv4 プレフィクス): ネットワークインターフェイスの IPv4 プレフィクス。[手動で割り当てる] を選択し、IPv4 プレフィックスを入力します。あるいは、[自動的に割り当て

る] を選択し、Amazon EC2 によって選択される IPv4 プレフィックスの数を示す値を入力します。

- [IPv6 Prefixes] (IPv6 プレフィクス): ネットワークインターフェイスの IPv6 プレフィクス。[手動で割り当てる] を選択し、IPv6 プレフィックスを入力します。あるいは、[自動的に割り当てる] を選択し、Amazon EC2 によって選択される IPv6 プレフィックスの数を示す値を入力します。
- (デュアルスタックおよび IPv6-only) [プライマリ IPv6 IP を割り当てる]: デュアルスタックまたは IPv6-onlyサブネットを選択する場合は、プライマリ IPv6 アドレスを割り当てます。これは、インスタンスまたはネットワークインターフェイスへのトラフィックの中断を防ぐのに役立ちます。IPv6アドレスが変わらないことを前提とする場合は、このオプションを有効にします。後でプライマリ IPv6 アドレスを削除することはできません。IPv6 GUA アドレスをプライマリ IPv6 にすることを有効にすると、インスタンスが終了するか、ネットワークインターフェースがデタッチされるまで、最初の IPv6 GUA がプライマリ IPv6 アドレスになります。ネットワークインターフェイスに複数の IPv6 アドレスが関連付けられており、Amazon EC2 にプライマリ IPv6 アドレスを割り当てさせる場合、ネットワークインターフェイスに関連付けられた最初の IPv6 GUA アドレスがプライマリ IPv6 アドレスになります。
- [終了時に削除]: インスタンス終了時にネットワークインターフェイスを削除するかどうかを示します。
- インターフェイスタイプ: ネットワークインターフェイスタイプ:
  - ENA: CPU 使用率を最小限に抑えながら、TCP/IP プロトコルの高スループットとパケット毎秒のレートを処理するように設計された高性能ネットワークインターフェイス。これは、デフォルト値です。ENA の詳細については、「Elastic Network Adapter」を参照してください。
  - ENA を使用した EFA: ENA デバイスと EFA デバイス両方のサポートにより、従来の TCP/IP ベースのトランスポートと SRD ベースのトランスポートを提供するネットワークインターフェイス。EFA の詳細については、「Elastic Fabric Adapter」を参照してください。
  - EFA のみ: オペレーティングシステムスタックをバイパスしながら、SRD ベーストランスポートでの高スループットで低レイテンシーのノード間通信を処理するように設計された高性能ネットワークインターフェイス。EFA 専用ネットワークインターフェイスは IP アドレスをサポートしていません。EFA の詳細については、「Elastic Fabric Adapter」を参照してください。
- Elastic Fabric Adapter: ネットワークインターフェイスが Elastic Fabric Adapter かどうかを示します。詳細については、「Amazon EC2 の AI/ML および HPC ワークロード用の Elastic Fabric Adapter」を参照してください。
- ネットワークカードインデックス: ネットワークカードのインデックス。プライマリネットワークインターフェイスは、ネットワークカードインデックス 0 に割り当てる必要があります。イ

ンスタンスタイプによっては、複数のネットワークカードがサポートされているものもあります。

- ENA Express: ENA Express は、AWS Scalable Reliable Datagram (SRD) テクノロジーを搭載しています。SRD テクノロジーは、パケットスプレーメカニズムを使用して負荷を分散し、ネットワークの混雑を回避します。ENA Express を有効にすると、サポートされているインスタンスは、可能な場合は通常の TCP トラフィックに加えて SRD を使用して通信できるようになります。リストから [有効化] または [無効化] を選択しない限り、インスタンス起動ウィザードまたは起動テンプレートにはインスタンスの ENA Express 設定は含まれません。
- [ENA Express UDP]: ENA Express を有効にしている場合は、オプションで UDP トラフィックに使用できます。[有効化] または [無効化] を選択しない限り、インスタンス起動ウィザードまたは起動テンプレートにはインスタンスの ENA Express 設定は含まれません。

ストレージの設定

選択した AMI には、ルートボリュームを含む、1 つまたは複数のストレージボリュームが含まれます。インスタンスにアタッチする追加のボリュームを指定できます。

(インスタンス起動ウィザードのみ) [シンプル] ビューまたは [詳細] ビューを使用できます。[Simple] (シンプル) ビューでは、ボリュームのサイズとタイプを指定します。すべてのボリュームパラメータを指定するには、[Advanced] (アドバンスト) ビュー (カードの右上) を選択します。

[Advanced] (アドバンスト) ビューでは、各ボリュームを以下のように設定できます。

- [Storage type] (ストレージタイプ): インスタンスと関連付ける Amazon EBS またはインスタンスストアボリュームを選択します。一覧で利用できるボリュームタイプは、選択したインスタンスタイプに応じて異なります。詳細については、「EC2 インスタンス用のインスタンスストアの一時ブロックストレージ」および「Amazon EBS ボリューム」を参照してください。
- [Device name] (デバイス名): ボリュームで利用できるデバイス名の一覧から選択します。
- [Snapshot] (スナップショット): ボリュームを復元するスナップショットを選択します。[Snapshot] (スナップショット) フィールドにテキストを入力して、利用できる共有スナップショットとパブリックスナップショットを検索することもできます。
- [Size (GiB)] (サイズ (GiB)): EBS ボリュームの場合、ストレージサイズを指定できます。
- [Volume Type] (ボリュームタイプ): EBS ボリュームの場合、ボリュームタイプを選択します。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームの種類」を参照してください。

- [IOPS]: io1、io2、または gp3 ボリュームタイプを選択した場合は、そのボリュームでサポートされている 1 秒あたりの I/O 操作回数 (IOPS) を入力できます。io1、io2、および gp3 ボリュームに必要です。gp2、st1、sc1、またはスタンダードボリュームではサポートされていません。起動テンプレートでこのパラメータを省略した場合は、起動テンプレートからインスタンスを起動するときに、パラメータの値を指定する必要があります。
- [Delete on termination] (終了時に削除): Amazon EBS ボリュームで、関連付けられたインスタンスの終了時にボリュームを削除する場合、[Yes] (はい) を選択し、ボリュームを保持する場合は [No] (いいえ) を選択します。詳細については、インスタンスの終了時にデータを保持するを参照してください。
- [Encrypted] (暗号化): インスタンスタイプが EBS 暗号化をサポートしている場合、[Yes] (はい) を選択し、ボリュームの暗号化を有効にできます。このリージョンでデフォルトで暗号化を有効にした場合、暗号化は有効になります。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS 暗号化」を参照してください。
- [KMS Key] (KMS キー): [Encrypted] (暗号化) で [Yes] (はい) を選択し、ボリュームで暗号化を使用する場合には、カスタマーマネージド型キーを選択する必要があります。このリージョンでデフォルトの暗号化を有効にした場合は、自動的にデフォルトのカスタマーマネージド型キーが選択されます。別のキーを選択するか、作成した顧客管理キーの ARN を指定できます。
- スループット: gp3 ボリュームタイプを選択した場合は、そのボリュームがサポートできるスループットを MiB/秒で入力できます。
- ボリューム初期化レート: スナップショットを選択した場合は、オプションでボリューム初期化レートを MiB/秒で指定できます。スナップショットブロックはこのレートで Amazon S3 からボリュームにダウンロードされます。詳細については、「ボリュームの初期化に Amazon EBS プロビジョンドレートを使用する」を参照してください。デフォルトの初期化レート、または高速スナップショット復元 (選択したスナップショットで有効化されている場合) を使用する場合は、レートを指定しないでください。
- [File systems] (ファイルシステム): Amazon EFS または Amazon FSx ファイルシステムをインスタンスにマウントします。Amazon EFS ファイルシステムのマウントの詳細については、「Amazon EC2 EC2 Linux インスタンスでの Amazon EC2 EFS の使用」を参照してください。Amazon FSx ファイルシステムのマウントの詳細については、「Amazon EC2 インスタンスでの Amazon FSx の使用」を参照してください。

高度な詳細

[Advanced details] で、セクションを開いてフィールドを表示し、インスタンスの追加パラメータを指定します。

- (インスタンス起動ウィザードのみ) [ドメイン結合ディレクトリ]: 起動後にインスタンスを結合する Directory Service ディレクトリ (ドメイン) を選択します。ドメインを選択する場合は、必要なアクセス許可を持つ IAM ロールを選択する必要があります。ドメイン参加の詳細については、「Seamlessly join an Amazon EC2 Linux instance to your AWS Managed Microsoft AD directory」(Linux インスタンス) および「Seamlessly join an Amazon EC2 Windows instance to your AWS Managed Microsoft AD directory」(Windows インスタンス) を参照してください。
- [IAM インスタンスプロファイル]: インスタンスに関連付ける IAM インスタンスプロファイルを選択します。これは IAM ロールのコンテナです。詳細については、「Amazon EC2 の IAM ロール」を参照してください。
- [Hostname type] (ホスト名タイプ): インスタンスのゲスト OS ホスト名をリソース名または IP 名に含めるかどうかを選択します。詳細については、EC2 インスタンスのホスト名とドメイン を参照してください。
- [DNS Hostname] (DNS ホスト名): リソース名または IP 名への DNS クエリが、([Hostname type] (ホスト名タイプ) に何を選択したのかによって) IPv4 アドレス (A レコード)、IPv6 アドレス (AAAA レコード)、またはその両方で応答するかどうかを決定します。詳細については、「EC2 インスタンスのホスト名とドメイン」を参照してください。
- [インスタンスの自動リカバリ]: 有効にすると、システムステータスチェックが失敗した場合にインスタンスが復旧します。この設定は、サポートされているインスタンスタイプでは起動時にデフォルトで有効になります。詳細については、「Amazon EC2 インスタンスで簡易自動復旧を設定する」を参照してください。
- [Shutdown behavior]: シャットダウン時にインスタンスを停止するか終了するかを選択します。詳細については、インスタンスによるシャットダウン動作を変更するを参照してください。
- [Stop - Hibernate behavior] (停止 – 休止動作): 休止を有効にするには、[Enable] (有効) を選択します。このフィールドは、インスタンスが休止の前提条件を満たしている場合にのみ使用できます。詳細については、Amazon EC2 インスタンスの休止 を参照してください。
- [Termination protection] (終了の保護): 偶発的な終了を防ぐには、[Enable] (有効) を選択します。詳細については、「インスタンスの終了保護を変更する」を参照してください。
- 停止保護: 偶発的な停止を防ぐには、[Enable] (有効化) を選択します。詳細については、「停止保護を有効にします」を参照してください。
- [Detailed CloudWatch monitoring] (詳細な CloudWatch モニタリング): Amazon CloudWatch を使用したインスタンスの詳細なモニタリングをオンにする場合、[Enable] (有効) を選択します。別途料金がかかります。詳細については、「CloudWatch を使用したインスタンスのモニタリング」を参照してください。

- [Credit specification] (クレジット指定): アプリケーションがベースラインを越えて必要なだけバーストできることを有効にするには、[Unlimited] (無制限) を選択します。このフィールドは、T インスタンスでのみ有効です。追加料金が適用される場合があります。詳細については、「バーストパフォーマンスインスタンス」を参照してください。
- [プレイスメントグループ]: インスタンスを起動する先のプレイスメントグループを指定します。既存のプレイスメントグループを選択するか、新しいグループを作成します。すべてのインスタンスタイプが、プレイスメントグループでのインスタンスの起動をサポートしているわけではありません。詳細については、Amazon EC2 インスタンスのプレイスメントグループ を参照してください。
- [EBS-optimized instance] (EBS 最適化インスタンス): Amazon EBS に最適化されたインスタンスは、最適化された設定スタックを使用し、Amazon EBS I/O に対して追加の専用容量を提供します。ご使用のインスタンスタイプでこの機能がサポートされている場合は、[Enable] (有効) を選択して有効にします。別途 料金がかかります。詳細については、「the section called “EBS 最適化”」を参照してください。
- [インスタンス帯域幅設定]: ネットワーク帯域幅または EBS 帯域幅のいずれかをブーストできます。サポートされているインスタンスタイプのみ。詳細については、「EC2 インスタンスの帯域幅の重み設定」を参照してください。
- [購入オプション]: [スポットインスタンス] を選択して、オンデマンド料金を上限とするスポット料金でスポットインスタンスをリクエストし、[スポットインスタンスのオプションをカスタマイズ] を選択して、スポットインスタンスのデフォルト設定を変更します。上限料金を設定し (非推奨)、リクエストタイプ、リクエスト期間、中断動作を変更できます。スポットインスタンスをリクエストしない場合、Amazon EC2 はデフォルトでオンデマンドインスタンスを起動します。詳細については、「スポットインスタンスを管理する」を参照してください。
- [Capacity Reservation] (容量の予約): インスタンスを起動する容量予約を指定します。任意の 容量の予約 ([Open] (オープン))、特定の容量の予約 ([Target by ID] (ID を対象とする))、または 容量の予約グループ ([Target by group] (グループを対象とする)) のいずれかから選択します。容量の予約を使用しないように指定するには、[None] (なし) を選択します。詳細については、既存のキャパシティ予約へのインスタンスの起動を参照してください。
- [テナンシー]: インスタンスを共有ハードウェア ([共有])、独立した専有ハードウェア ([専有])、あるいは Dedicated Host ([Dedicated host (専有ホスト)])で実行するかを選択します。Dedicated Host でインスタンスを起動する場合は、インスタンスをホストリソースグループ内で起動するかどうかを指定できます。または、特定の Dedicated Host をターゲットとして設定できます。追加料金が適用される場合があります。詳細については、Amazon EC2 ハードウェア専有インスタンスおよびAmazon EC2 専有ホストを参照してください。

- [RAM disk ID] (RAM ディスク ID): (準仮想化 (PV) AMI に対してのみ有効) インスタンスの RAM ディスクを選択します。カーネルを選択した場合は、サポートするドライバーとともに特定の RAM ディスクを選択しなければならない可能性があります。
- [Kernel ID] (カーネル ID) (準仮想化 (PV) AMI に対してのみ有効) インスタンスのカーネルを選択します。
- [Nitro Enclaves]: Amazon EC2 インスタンスから、エンクレーブと呼ばれる分離された実行環境を作成することを許可します。AWS Nitro Enclaves のインスタンスを有効にするには、[Enable] (有効) を選択します。詳細については、AWS Nitro Enclaves ユーザーガイドのAWS Nitro Enclaves とはを参照してください。
- [ライセンス設定]: 指定したライセンス設定に対してインスタンスを起動して、ライセンスの使用状況を追跡できます。詳細については、AWS License Manager ユーザーガイドの「Create a license configuration」(ライセンス設定の作成) を参照してください。
- [CPU オプションを指定する]: インスタンス起動ウィザードでは、このフィールドは、選択したインスタンスタイプが CPU オプションの指定をサポートしている場合にのみ表示されます。起動中に [CPU オプションを指定] を選択して、カスタム数の vCPU を指定します。CPU コアの数とコアごとのスレッド数を設定します。詳細については、「Amazon EC2 インスタンスに関する CPU オプション」を参照してください。
- [アクセス可能なメタデータ]: インスタンスメタデータサービス (IMDS) へのアクセスを有効または無効にできます。詳細については、「新規インスタンスのインスタンスメタデータオプションの設定」を参照してください。
- [メタデータの転送]: IMDS IPv6 アドレス [`fd00:ec2::254`] を使用してインスタンスのメタデータを取得できるようにインスタンスを設定できます。このオプションは、Nitro ベースのインスタンスを IPv6 対応サブネット (デュアルスタックまたは IPv6 専用) で起動している場合にのみ使用できます。インスタンスメタデータの取得の詳細については、「EC2 インスタンスのインスタンスメタデータにアクセスする」を参照してください。
- [メタデータのバージョン]: インスタンスメタデータへのアクセスを有効にする場合、IMDS をリクエストするときにインスタンスメタデータサービスバージョン 2 の使用を必須にすることができます。詳細については、「新規インスタンスのインスタンスメタデータオプションの設定」を参照してください。
- [メタデータレスポンスのホップ制限]: IMDS を有効にする場合、メタデータトークンに許容されるネットワークホップ数を設定できます。詳細については、「新規インスタンスのインスタンスメタデータオプションの設定」を参照してください。
- [Allow tags in metadata] (メタデータ内のタグを許可する): [Enable] (有効) を選択した場合、インスタンスはメタデータ内のすべてのタグへのアクセスを許可します。値を指定しない場合、インスタ

ンスメタデータ内のタグへのアクセスはデフォルトで無効になります。詳細については、インスタンスメタデータ内のタグへのアクセスを有効にする を参照してください。

- [ユーザーデータ]: 起動時にインスタンスを設定するユーザーデータ、または設定スクリプトを実行するユーザーデータを指定できます。Linux インスタンスのユーザーデータの詳細については、「ユーザーデータ入力を使用して EC2 インスタンスを起動するときにコマンドを実行する」を参照してください。Windows インスタンスのユーザーデータの詳細については、「Amazon EC2 が Windows インスタンスのユーザーデータを処理する方法」を参照してください。

概要

[Summary] (サマリー) パネルを使用して、起動するインスタンスの数を指定し、インスタンス構成を確認し、インスタンスを起動します。

- [Number of instances]: 起動するインスタンスの数を入力します。すべてのインスタンスは、同じ設定で起動します。

> **Tip**
>
> インスタンスの起動を高速化するには、大きなリクエストをより小さなバッチに分割します。例えば、1 つの起動リクエストに 500 インスタンスが含まれている場合は、それを 5 つの起動リクエスト (各 100 インスタンス) に分割します。

- (オプション) 複数のインスタンスを指定した場合は、アプリケーションの要求に対処できるだけのインスタンス数が確保されるように、[consider EC2 Auto Scaling] (EC2 Auto Scaling を考慮) を選択して起動テンプレートと Auto Scaling グループを作成することができます。Auto Scaling によって、指定どおりにグループのインスタンス数がスケーリングされます。詳細については、Amazon EC2 Auto Scaling ユーザーガイドを参照してください。

> **Note**
>
> Amazon EC2 Auto Scaling が Auto Scaling グループ内のインスタンスを異常とマークすると、そのインスタンスの置き換えが自動的にスケジュールされます。この場合、インスタンスは終了されて別のインスタンスが起動され、元のインスタンスのデータは失われます。インスタンスを停止または再起動するか、別のイベントがインスタンスを異常としてマークすると、インスタンスは異常としてマークされます。詳細については、Amazon EC2 Auto Scaling ユーザーガイドの「Auto Scaling グループ内のインスタンスのヘルスチェック」を参照してください。

- インスタンスの詳細を確認し、必要な変更を加えます。[Summary] (サマリー) パネルのリンクを選択すると、セクションに直接移動することができます。
- インスタンスを起動する準備ができたら、[Launch instance] (インスタンスの起動) を選択します。

コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する

Amazon EC2 コンソールのインスタンス起動ウィザードを使用して、Amazon EC2 インスタンスを起動できます。ウィザードは起動パラメータのデフォルト値を提供します。この値は、要件に合わせてそのまま使用するか変更することができます。指定されていない唯一のパラメータはキーペアです。デフォルト値をそのまま使用する場合は、キーペアのみを選択するだけでインスタンスをすばやく起動できます。

> **Important**
>
> インスタンスが `running` 状態にある間は、アイドル状態のままであっても、インスタンスに対して料金が発生します。ただし、無料利用枠の対象となると、料金が発生しない場合があります。詳細については、「Amazon EC2 の無料利用枠の使用状況を追跡する」を参照してください。

インスタンス起動ウィザードの各パラメータの説明については、「Amazon EC2 インスタンス設定パラメータのリファレンス」を参照してください。

トピック

- インスタンスをすばやく起動する
- 定義済みのパラメータを使用したインスタンスの起動

インスタンスをすばやく起動する

テスト目的でインスタンスをすばやくセットアップするには、次のステップに従います。オペレーティングシステムとキーペアを選択し、デフォルト値を受け入れます。キーペアを除き、インスタンス起動ウィザードはすべてのパラメータのデフォルト値を提供します。デフォルトの一部またはすべてを受け入れるか、各パラメータに独自の値を指定してインスタンスを設定することができます。

インスタンス起動ウィザードの各パラメータの説明については、「Amazon EC2 インスタンス設定パラメータのリファレンス」を参照してください。

インスタンス起動ウィザードを使用してインスタンスをすばやく起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 画面の上のナビゲーションバーで、現在の AWS リージョンが表示されます (例: 米国東部 (オハイオ))。必要に応じて、インスタンスを起動する別のリージョンを選択します。
3. Amazon EC2 コンソールダッシュボードで、[インスタンスを起動] を選択してください。
4. (オプション) [Names and tags](名前とタグ) における [Name] (名前) では、インスタンス用にわかりやすい名前を入力してください。
5. [Application and OS Images (Amazon Machine Image)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で [Quick Start] (クイックスタート) を選択し、インスタンスのオペレーティングシステム (OS) を選択します。
6. [Key pair (login)] (キーペア (ログイン)) の [Key pair name] (キーペア名) で、既存のキーペアを選択するか、新しいキーペアを作成します。
7. [Summary] (サマリー) パネルで、[Launch instance] (インスタンスの起動) を選択します。

定義済みのパラメータを使用したインスタンスの起動

本番環境で使用するインスタンスを起動する場合は、要件に合わせてインスタンスを設定する必要があります。インスタンス起動ウィザードの各パラメータの説明については、「Amazon EC2 インスタンス設定パラメータのリファレンス」を参照してください。

インスタンス起動ウィザードを使用してすべての起動パラメータを定義してインスタンスを起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 画面の上のナビゲーションバーで、現在の AWS リージョンが表示されます (例: 米国東部 (オハイオ))。必要に応じて、インスタンスを起動する別のリージョンを選択します。
3. Amazon EC2 コンソールダッシュボードで、[インスタンスを起動] を選択してください。
4. (オプション) [名前とタグ] で、[名前] にインスタンスのわかりやすい名前を入力して、インスタンスを簡単に追跡できるようにします。

   インスタンス名はタグで、キーは [Name] (名前)、値は指定した名前です。

5. [アプリケーションおよび OS イメージ (Amazon マシンイメージ)] で、インスタンスのオペレーティングシステム (OS) を選択してから、AMI を選択します。

   AMI は、インスタンスの起動に必要なオペレーティングシステムとソフトウェアを含むテンプレートです。

6. [Instance type] (インスタンスタイプ) でインスタンスタイプを選択します。

   インスタンスタイプによって、インスタンスに使用されるホストコンピュータのハードウェア構成 (CPU、メモリ、ストレージ、ネットワークキャパシティ) とサイズが決まります。

   どのインスタンスタイプを選択するかわからない場合は、次の操作を実行できます。

   - [インスタンスタイプの比較] を選択して、vCPU の数、アーキテクチャ、メモリ量 (GiB)、ストレージ量 (GB)、ストレージタイプ、ネットワークパフォーマンスの属性によってさまざまなインスタンスタイプを比較します。
   - [アドバイスの取得] を選択して、EC2 インスタンスタイプファインダーからインスタンスタイプに関するガイダンスやアドバイスを取得します。詳細については、「EC2 インスタンスタイプファインダーからレコメンデーションを取得する」を参照してください。

> **Note**
>
> アカウントの作成時期によっては、インスタンスタイプを無料利用枠の範囲内で無料で使用できる場合があります。これらのインスタンスタイプには、[無料利用枠の対象] というラベルが表示されます。
> 2025 年 7 月 15 日より前に AWS アカウントを作成していて、12 か月が経過していない場合は、[t2.micro] インスタンスタイプ、または [t2.micro] が利用できないリージョンで [t3.micro] インスタンスタイプを選択すると、無料利用枠の Amazon EC2 を使用できます。[t3.micro] インスタンスを起動すると、デフォルトで [Unlimited] モードになり、CPU 使用率に基づいて追加料金が発生する可能性があることに注意してください。
> 2025 年 7 月 15 日以降に AWS アカウント を作成している場合は、t3.micro、t3.small、t4g.micro、t4g.small、c7i-flex.large、および m7i-flex.large インスタンスタイプを 6 か月間、またはクレジットを使い切るまで使用できます。
> 詳細については、「2025 年 7 月 15 日より前とそれ以降の無料利用枠の特典」を参照してください。

7. [Key pair (login)] (キーペア (ログイン)) の [Key pair name] (キーペア名) で、既存のキーペアを選択するか、新しいキーペアを作成します。インスタンスへの接続にキーペアが必要ない場合は、[キーペアなしで続行 (非推奨)] を選択できます。
8. [ネットワーク設定] では、テストインスタンスを起動する場合、デフォルトのままにします。本番稼働用のインスタンスを起動する場合は、定義したネットワーク設定とセキュリティグループを使用して、インスタンスに出入りするトラフィックを制御するのがベストプラクティスです。
9. [ストレージの設定] で、デフォルトのままにするか、追加のストレージを指定できます。選択した AMI には、ルートボリュームを含む、1 つまたは複数のストレージボリュームが含まれます。インスタンスにアタッチする追加のボリュームを指定できます。

   シンプルまたはアドバンストビューを使用できます。[Simple] (シンプル) ビューでは、ボリュームのサイズとタイプを指定します。すべてのボリュームパラメータを指定するには、[Advanced] (アドバンスト) ビュー (カードの右上) を選択します。
10. [高度な詳細] で、セクションを開いてフィールドを表示し、インスタンスの追加パラメータを指定します。
11. [概要] パネルで、以下を実行できます。

    a. 起動するインスタンスの数を指定します。

    b. インスタンス設定を確認し、リンクを選択してセクションに直接移動します。

    c. インスタンスを起動する準備ができたら、[Launch instance] (インスタンスの起動) を選択してください。

    インスタンスが起動しないか、状態が `terminated` ではなくすぐに `running` になる場合は、Amazon EC2 インスタンスの起動に関する問題のトラブルシューティングを参照してください。
12. (オプション) インスタンスの請求アラートを作成できます。確認画面で、[Next Steps] (次のステップ) で [Create billing alerts] (請求アラートの作成) を選択し、指示に従います。請求アラートは、インスタンスの起動後に作成することもできます。詳細については、「Amazon CloudWatch ユーザーガイド」の「推定の AWS 料金をモニタリングする請求アラームを作成」を参照してください。

起動テンプレートを使用した EC2 インスタンスの起動

Amazon EC2 起動テンプレートには、インスタンス起動パラメータが保存されるため、インスタンスを起動するたびにパラメータを指定する必要はありません。

一部のインスタンス起動サービスは、インスタンスの起動時にオプションで起動テンプレートを使用できますが、EC2 フリートなどの他のサービスでは、起動テンプレートを使用しないとインスタンスを起動できません。このトピックでは、EC2 インスタンス起動ウィザード、Amazon EC2 Auto Scaling、EC2 フリート、スポットフリートを使用してインスタンスを起動するときに起動テンプレートを使用する方法について説明します。

起動テンプレートの作成方法など、起動テンプレートの詳細については、Amazon EC2 起動テンプレートにインスタンス起動パラメータを保存するを参照してください。

トピック

- 起動テンプレートを使用して Amazon EC2 インスタンスを起動する
- 起動テンプレートを使用して Amazon EC2 Auto Scaling グループでインスタンスを起動する
- 起動テンプレートを使用して EC2 フリートを起動する
- 起動テンプレートを使用してスポットフリートを起動する

起動テンプレートを使用して Amazon EC2 インスタンスを起動する

起動テンプレートに含まれているパラメータを使用して Amazon EC2 インスタンスを起動できます。起動テンプレートを選択した後、インスタンスを起動する前に、起動パラメータを変更できます。

起動テンプレートを使用して起動されたインスタンスには、`aws:ec2launchtemplate:id` と `aws:ec2launchtemplate:version` のキーを使用して自動的に 2 つのタグが割り当てられます。これらのタグを削除したり、編集することはできません。

Console

起動テンプレートを使用してインスタンスを起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 次のオプションのいずれかを使用して、起動テンプレートを選択します。
   - Amazon EC2 コンソールダッシュボードで、[インスタンスの起動] の横にある下矢印を選択し、[テンプレートからインスタンスを起動] を選択し、[ソーステンプレート] で起動テンプレートを選択します。
   - ナビゲーションペインで、[テンプレートの起動] を選択して、起動テンプレートを選択し、[アクション]、[テンプレートからインスタンスを起動] の順に選択します。

3. [Source template version] (ソーステンプレートのバージョン) で、使用する起動テンプレートのバージョンを選択します。
4. (オプション) 任意の起動パラメータの値を変更することができます。値を変更しない場合、起動テンプレートで定義された値が使用されます。起動テンプレートに値が指定されていない場合は、パラメータのデフォルト値が使用されます。
5. [概要] パネルの [インスタンス数] に、起動するインスタンス数を指定します。
6. [インスタンスを起動] を選択してください。

   インスタンスが起動しないか、状態が `terminated` ではなくすぐに `running` になる場合はAmazon EC2 インスタンスの起動に関する問題のトラブルシューティングを参照してください。

AWS CLI

起動テンプレートからインスタンスを起動するには

- run-instances コマンドを使用して `--launch-template` パラメータを指定します。オプションで、使用する起動テンプレートのバージョンを指定します。バージョンを指定しない場合は、デフォルトバージョンが使用されます。

```
aws ec2 run-instances \
    --launch-template LaunchTemplateId=lt-0abcd290751193123,Version=1
```

- 起動テンプレートパラメータを上書きするには、run-instances コマンドでパラメータを指定します。次の例では、起動テンプレートで指定されたインスタンスタイプを上書きします (ある場合)。

```
aws ec2 run-instances \
    --launch-template LaunchTemplateId=lt-0abcd290751193123 \
    --instance-type t2.small
```

- 複雑な構造の一部である入れ子状のパラメータを指定した場合、インスタンスは、起動テンプレートで指定された複雑な構造および、指定した入れ子状の追加パラメータを使用して起動されます。

  次の例で、インスタンスは、タグ *Owner*=*TeamA* および起動テンプレートで指定された他のタグを使用して起動されます。起動テンプレートに既存の *Owner* のキーのタグがある場合、値は *TeamA* に置き換えられます。

```
aws ec2 run-instances \
    --launch-template LaunchTemplateId=lt-0abcd290751193123 \
    --tag-specifications "ResourceType=instance,Tags=[{Key=Owner,Value=TeamA}]"
```

次の例で、インスタンスは、*/dev/xvdb* というデバイス名のボリューム、および起動テンプレートで指定されたその他のブロックデバイスマッピングを使用して起動されます。起動テンプレートに */dev/xvdb* 用に定義された既存のボリュームがある場合、値は指定された値に置き換えられます。

```
aws ec2 run-instances \
    --launch-template LaunchTemplateId=lt-0abcd290751193123 \
    --block-device-mappings "DeviceName=/dev/
xvdb,Ebs={VolumeSize=20,VolumeType=gp2}"
```

インスタンスが起動しないか、状態が `terminated` ではなくすぐに `running` になる場合は、Amazon EC2 インスタンスの起動に関する問題のトラブルシューティングを参照してください。

PowerShell

AWS Tools for PowerShellを使用して起動テンプレートからインスタンスを起動するには

- [New-EC2Instance] コマンドを使用して、`-LaunchTemplate` パラメータを指定します。オプションで、使用する起動テンプレートのバージョンを指定します。バージョンを指定しない場合は、デフォルトバージョンが使用されます。

```
Import-Module AWS.Tools.EC2
New-EC2Instance `
    -LaunchTemplate (
        New-Object -TypeName Amazon.EC2.Model.LaunchTemplateSpecification -
Property @{
            LaunchTemplateId = 'lt-0abcd290751193123';
            Version          = '4'
    }
)
```

- 起動テンプレートパラメータを上書きするには、[New-EC2Instance] コマンドでパラメータを指定します。次の例では、起動テンプレートで指定されたインスタンスタイプを上書きします (ある場合)。

```
Import-Module AWS.Tools.EC2
New-EC2Instance `
    -InstanceType t4g.small `
    -LaunchTemplate (
        New-Object -TypeName Amazon.EC2.Model.LaunchTemplateSpecification -
Property @{
            LaunchTemplateId = 'lt-0abcd290751193123';
            Version          = '4'
    }
)
```

- 複雑な構造の一部である入れ子状のパラメータを指定した場合、インスタンスは、起動テンプレートで指定された複雑な構造および、指定した入れ子状の追加パラメータを使用して起動されます。

  次の例で、インスタンスは、タグ *Owner=TeamA* および起動テンプレートで指定された他のタグを使用して起動されます。起動テンプレートに既存の *Owner* のキーのタグがある場合、値は *TeamA* に置き換えられます。

```
Import-Module AWS.Tools.EC2
New-EC2Instance `
    -InstanceType t4g.small  `
    -LaunchTemplate (
        New-Object -TypeName Amazon.EC2.Model.LaunchTemplateSpecification -
Property @{
            LaunchTemplateId = 'lt-0abcd290751193123';
            Version          = '4'
        }
) `
    -TagSpecification (
        New-Object -TypeName Amazon.EC2.Model.TagSpecification -Property @{
            ResourceType = 'instance';
            Tags         = @(
                @{key = "Owner"; value = "TeamA" },
                @{key = "Department"; value = "Operations" }
            )
        }
)
```

  次の例で、インスタンスは、*/dev/xvdb* というデバイス名のボリューム、および起動テンプレートで指定されたその他のブロックデバイスマッピングを使用して起動されます。起動テン

プレートに */dev/xvdb* 用に定義された既存のボリュームがある場合、値は指定された値に置き換えられます。

```
Import-Module AWS.Tools.EC2
New-EC2Instance `
    -InstanceType t4g.small  `
    -LaunchTemplate (
        New-Object -TypeName Amazon.EC2.Model.LaunchTemplateSpecification -
Property @{
            LaunchTemplateId = 'lt-0abcd290751193123';
            Version          = '4'
    }
) `
    -BlockDeviceMapping  (
        New-Object -TypeName Amazon.EC2.Model.BlockDeviceMapping -Property @{
            DeviceName = '/dev/xvdb';
            EBS        = (
                New-Object -TypeName Amazon.EC2.Model.EbsBlockDevice -Property @{
                    VolumeSize = 25;
                    VolumeType = 'gp3'
                }
            )
        }
)
```

インスタンスが起動しないか、状態が `terminated` ではなくすぐに `running` になる場合は、Amazon EC2 インスタンスの起動に関する問題のトラブルシューティングを参照してください。

起動テンプレートを使用して Amazon EC2 Auto Scaling グループでインスタンスを起動する

Auto Scaling グループを作成して、グループに使用する起動テンプレートを指定できます。Auto Scalingグループ内で Amazon EC2 Auto Scaling がインスタンスを起動する際、関連する起動テンプレートで定義された起動パラメータが使用されます。

起動テンプレートを使用して Auto Scaling グループを作成するには、Auto Scaling グループのインスタンスの起動に必要なパラメータを含む起動テンプレートを作成する必要があります。AMI の ID など、一部のパラメータは必須であり、一部のパラメータは Auto Scaling グループでは使用できま

せん。コンソールには、Amazon EC2 Auto Scaling で使用できるテンプレートの、作成に役立つガイダンスが用意されています。

コンソールを使用して起動テンプレートで Auto Scaling グループを作成するには

- 手順については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Create an Auto Scaling group using a launch template」を参照してください。

AWS CLI を使用して、起動テンプレートを使って Auto Scaling グループを作成または更新するには

- create-auto-scaling-group または update-auto-scaling-group コマンドを使用して `--launch-template` パラメータを指定します。

詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の次のトピックを参照してください。

- Auto Scaling グループの起動テンプレートを作成する
- 詳細設定を使用して起動テンプレートを作成する
- AWS Command Line Interface (AWS CLI) を使用した起動テンプレートの作成および管理の例 – さまざまなパラメータの組み合わせで起動テンプレートを作成する方法を示す例を提供します。
- 起動テンプレートを使用して Auto Scaling グループを作成する
- Auto Scaling グループを更新する

起動テンプレートを使用して EC2 フリートを起動する

起動テンプレートは、EC2 フリートリクエストを作成する際に必要です。Amazon EC2 は、EC2 フリート リクエストを満たす際、関連する起動テンプレートで定義された起動パラメータを使用します。起動テンプレートで指定されたパラメータの一部を上書きすることができます。詳細については、EC2 フリートの作成を参照してください。

AWS CLI により起動テンプレートを使用して、EC2 フリートを作成するには

- create-fleet コマンドを使用します。`--launch-template-configs` パラメータを使用して、起動テンプレートと起動テンプレートの上書きを指定します。

起動テンプレートを使用してスポットフリートを起動する

> **Important**
>
> スポットフリートは、計画された投資なしでレガシー API を使用します。代わりに EC2 フリートまたは Auto Scaling グループを使用することをお勧めします。詳細については、「フリートにどの方法を使用するのが最適か」を参照してください。

スポットフリートリクエストを作成する場合、起動テンプレートはオプションです。起動テンプレートを使用しない場合は、起動パラメータを手動で指定できます。起動テンプレートを使用すると、Amazon EC2 は、スポットフリートリクエストを処理する際、関連する起動テンプレートで定義された起動パラメータを使用します。起動テンプレートで指定されたパラメータの一部を上書きすることができます。詳細については、スポットフリートを作成するを参照してください。

起動テンプレートを使用してスポットフリートリクエストを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. [Request Spot Instances (スポットインスタンスのリクエスト)] を選択します。
4. [Launch parameters] (起動パラメータ) で、[Use a launch template] (起動テンプレートを使用する) を選択します。
5. [Launch tempate] (起動テンプレート) で、起動テンプレートを選択し、右側のフィールドから起動テンプレートのバージョンを選択します。
6. この画面で別のオプションを選択して、スポットフリートを設定します。オプションの詳細については、定義済みパラメータを使用してスポットフリートリクエストを作成するを参照してください。
7. スポットフリートを作成する準備が整ったら、[Launch] (起動) を選択します。

起動テンプレートを使用してスポットフリートリクエストを作成するには

- request-spot-fleet コマンドを使用します。`LaunchTemplateConfigs` パラメータを使用して、起動テンプレートと起動テンプレートの上書きを指定します。

既存のインスタンスの詳細を使用した EC2 インスタンスの起動

Amazon EC2 コンソールには、現在のインスタンスを他のインスタンスを起動するためのベースとして使用可能にする、[Launch More Like This] (同様のインスタンスをさらに起動) オプションが用意されています。このオプションでは、Amazon EC2 インスタンス起動ウィザードで、選択されたインスタンスから自動的に特定の設定が入力されます。

考慮事項

- インスタンスのクローニングは行わず、設定の詳細の一部だけを複製します。インスタンスのコピーを作成するには、最初にインスタンスから AMI を作成して、AMI からさらに多くのインスタンスを起動します。起動テンプレートを作成して、同じ起動詳細を使用してインスタンスを起動するようにしてください。
- 現在のインスタンスは `running` の状態である必要があります。

コピーされる詳細

次の設定詳細は、選択されたインスタンスからインスタンス起動ウィザードにコピーされます。

- AMI ID
- インスタンスタイプ
- アベイラビリティーゾーン、または選択されたインスタンスがある VPC とサブネット
- パブリック IPv4 アドレス。選択されたインスタンスに現在パブリック IPv4 アドレスがある場合、選択されたインスタンスのデフォルトのパブリック IPv4 アドレス設定に関係なく、新しいインスタンスはパブリック IPv4 アドレスを受け取ります。パブリック IPv4 アドレスの詳細については、パブリック IPv4 アドレスを参照してください。
- プレイスメントグループ (該当する場合)
- 該当する場合は、インスタンスに関連付けられた IAM ロール
- シャットダウン動作の設定 (停止または終了)
- 終了保護設定 (true または false)
- CloudWatch モニタリング (有効または無効)
- Amazon EBS 最適化設定 (true または false)
- VPC (共有または専用) に起動する場合は、テナンシー設定
- 該当する場合は、カーネル ID および RAM ディスク ID
- ユーザーデータ (指定された場合)

- 該当する場合は、インスタンスに関連付けられたタグ
- インスタンスに関連付けられたセキュリティグループ
- [Windows インスタンス] 関連付け情報。選択したインスタンスが設定ファイルに関連付けられている場合、同じファイルが自動的に新しいインスタンスに関連付けられます。設定ファイルに結合ドメインの設定が含まれる場合は、新しいインスタンスが同じドメインに結合されます。ドメインの結合の詳細については、「AWS Directory Service 管理ガイド」の「Windows EC2 インスタンスと AWS マネージド Microsoft AD Active Directory のシームレスな結合」を参照してください。

コピーされない詳細

次の設定の詳細は、選択したインスタンスからはコピーされません。代わりに、ウィザードはデフォルトの設定または動作を適用します。

- ネットワークインターフェイスの数 - デフォルトでは、1 つのネットワークインターフェイス、つまりプライマリネットワークインターフェイス (eth0) です。
- ストレージ - デフォルトのストレージ設定は AMI およびインスタンスタイプによって決まります。

既存のインスタンスと同様により多くのインスタンスを起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択し、[アクション]、[イメージとテンプレート]、[同様のものを起動] を選択します。
4. インスタンス起動ウィザードが開きます。この画面で別のオプションを選択して、インスタンス設定に必要な変更を加えることができます。

   インスタンスを起動する準備ができたら、[Launch instance] (インスタンスの起動) を選択します。
5. インスタンスが起動しないか、状態が `terminated` ではなくすぐに `running` になる場合はAmazon EC2 インスタンスの起動に関する問題のトラブルシューティングを参照してください。

AWS Marketplace AMI から Amazon EC2 インスタンスを起動する

Amazon EC2 コンソールまたはコマンドラインツールを使用して、AWS Marketplace AMI をサブスクライブし、AMI からインスタンスを起動できます。AWS Marketplace AMI の詳細については「Amazon EC2 インスタンスの AWS Marketplace の有料 AMI」を参照してください。

起動後に AMI へのサブスクリプションをキャンセルするにはまず AMI から起動されたすべてのインスタンスを終了する必要があります。詳細については「AWS Marketplace サブスクリプションを管理する」を参照してください。

Amazon EC2 コンソールを使用して AWS Marketplace AMI からインスタンスを起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. Amazon EC2 コンソールダッシュボードで、[インスタンスを起動] を選択してください。
3. (オプション) [Names and tags](名前とタグ) における [Name] (名前) ではインスタンス用にわかりやすい名前を入力してください。
4. [Application and OS Images (Amazon マシンイメージ)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で、[Browse more AMI] (さらに AMI を参照) を選択し、[AWS Marketplace AMI] タブを選択してください。カテゴリを参照するか、検索機能を使用して適切な AMI を見つけます。製品を選択するには[Select] (選択) を選択してください。
5. 選択した製品の概要と共に、ウィンドウが開きます。価格情報と、ベンダーが提供したその他の情報を表示できます。準備ができたら、[サブスクライブして起動] を選択します。選択すると、サブスクリプションが直ちに開始されます。サブスクリプションの進行中はこの手順のステップを続行してインスタンスを設定できます。クレジットカードの詳細に問題がある場合はアカウントの詳細を更新するように求められます。

   > **Note**
   >
   > AMI でインスタンスを起動するまで、製品の使用料は発生しません。インスタンスタイプを選択する際に、サポートされている各インスタンスタイプの料金を書き留めておいてください。追加の税金が製品に適用される場合があります。

6. [Instance type] (インスタンスタイプ) で、インスタンスのインスタンスタイプを選択してください。インスタンスタイプは起動するインスタンスのハードウェア設定とサイズを定義します。
7. [Key pair (login)] (キーペア (ログイン)) の [Key pair name] (キーペア名) で、既存のキーペアを選択するか、新しいキーペアを作成します。

8. [ネットワーク設定] の [ファイアウォール (セキュリティグループ)] で、製品のベンダーの仕様に従って作成された新しいセキュリティグループを書き留めます。セキュリティグループにはLinux の SSH (ポート 22) または Windows の RDP (ポート 3389) ですべての IPv4 アドレス (`0.0.0.0/0`) を許可するルールが含まれる場合があります。これらのルールを調整して、特定のアドレスまたはアドレスの範囲のみが、これらのポート経由でインスタンスにアクセスできるようにすることをお勧めします。
9. 画面上の他のフィールドを使用して、インスタンスの設定、ストレージの追加、およびタグの追加を行うことができます。設定できるさまざまなオプションの詳細については「Amazon EC2 インスタンス設定パラメータのリファレンス」を参照してください。
10. [合計mary] (概要) パネルの [Software Image (AMI)] (ソフトウェアイメージ (AMI)) で、インスタンスを起動しようとしている AMI の詳細を確認します。また、指定した他の設定の詳細も確認します。 インスタンスを起動する準備ができたら、[Launch instance] (インスタンスの起動) を選択してください。
11. 受信登録した製品によってはインスタンスの起動には数分以上かかります。クレジットカードの詳細に問題がある場合はアカウントの詳細を更新するように求められます。起動確認ページが表示されたら、[View all instances] (すべてのインスタンスを表示) を選択して [Instances] (インスタンス) ページに移動します。

> **Note**
>
> インスタンスが `running` 状態である限り、アイドル状態であっても、受信登録費用が発生します。インスタンスが停止している場合でも、ストレージに対して課金されることがあります。

12. インスタンスの状態が [`running`] の場合、そのインスタンスに接続することができます。これを実行するにはリストでインスタンスを選択し、[Connect] (接続) を選択して、接続オプションを選択してください。インスタンスへの接続の詳細についてはEC2 インスタンスに接続するを参照してください。

> **Important**
>
> インスタンスに接続するには特定のユーザー名を使用しなければならない場合があるため、ベンダーの使用手順を慎重に確認してください。受信登録の詳細へのアクセスについてはAWS Marketplace サブスクリプションを管理する を参照してください。

13. インスタンスが起動しないか、状態が `terminated` ではなくすぐに `running` になる場合はAmazon EC2 インスタンスの起動に関する問題のトラブルシューティングを参照してください。

コマンドラインツールを使用して AWS Marketplace AMI からインスタンスを起動するには

コマンドラインツールを使用して、AWS Marketplace 製品からインスタンスを起動するにはまず製品に登録していることを確認します。次の方法を使用して、製品の AMI ID でインスタンスを起動できます。

方法	ドキュメント
AWS CLI	run-instances コマンドを使用するか、詳細について「インスタンス の起動」を参照しAWS Command Line Interface てください。
AWS Tools for Windows PowerShell	New-EC2Instance コマンドを使用するか、詳細についてWindows PowerShell を使用した Amazon EC2 インスタンスの起動を参照してください。
クエリ API	RunInstancesリクエストを使用します。

EC2 インスタンスに接続する

Amazon EC2 インスタンスは AWS クラウド 上の仮想サーバーです。インスタンスにログオンするには、インスタンスへの接続を確立する必要があります。インスタンスへの接続方法は、インスタンスのオペレーティングシステムと、インスタンスへの接続に使用するコンピュータのオペレーティングシステムによって異なります。次の表に、各接続方法の要件を示します。

接続オプション	インスタンスのオペレーティングシステム	インバウンドトラフィックルール	IAM アクセス許可	インスタンスプロファイルロール	インスタンス上のソフトウェア	接続システムのソフトウェア	キーペア
SSH クライアント	Linux	はい	いいえ	いいえ	なし	はい	はい
EC2 Instance Connect	Linux	はい	はい	いいえ	はい [1]	いいえ	いいえ
PuTTY	Linux	はい	いいえ	いいえ	なし	はい	はい
RDP クライアント	Windows	はい	いいえ	いいえ	なし	はい	はい [2]
Fleet Manager	Windows	いいえ	はい	はい	はい [1]	いいえ	はい
セッションマネージャー	Linux、Windows	いいえ	はい	はい	はい [1]	いいえ	いいえ
EC2 Instance Connect エンドポイント	Linux、Windows	はい	はい	いいえ	いいえ	いいえ	なし [3]

[1] 必要なソフトウェアは、特定の AMI にのみプリインストールされています。必要に応じて、サポートされているオペレーティングシステムに必要なソフトウェアを手動でインストールできます。

[2] キーペアは、ローカル管理者ユーザーアカウントにランダムに生成されたパスワードを使用している場合にのみ必要です。

[3] SSH 接続方法を使用する場合は、キーペアが必要です。

詳細については、使用する接続オプションのドキュメントを参照してください。

接続オプション

- SSH クライアントを使用して Linux インスタンスに接続する
- PuTTY を使用して Linux インスタンスに接続する
- RDP クライアントを使用して Windows インスタンスに接続する
- Fleet Manager を使用して Windows インスタンスに接続する
- Session Manager による接続
- パブリック IP と EC2 Instance Connect を使用して接続する
- プライベート IP と EC2 Instance Connect エンドポイントを使用して接続する

一般的な接続の前提条件

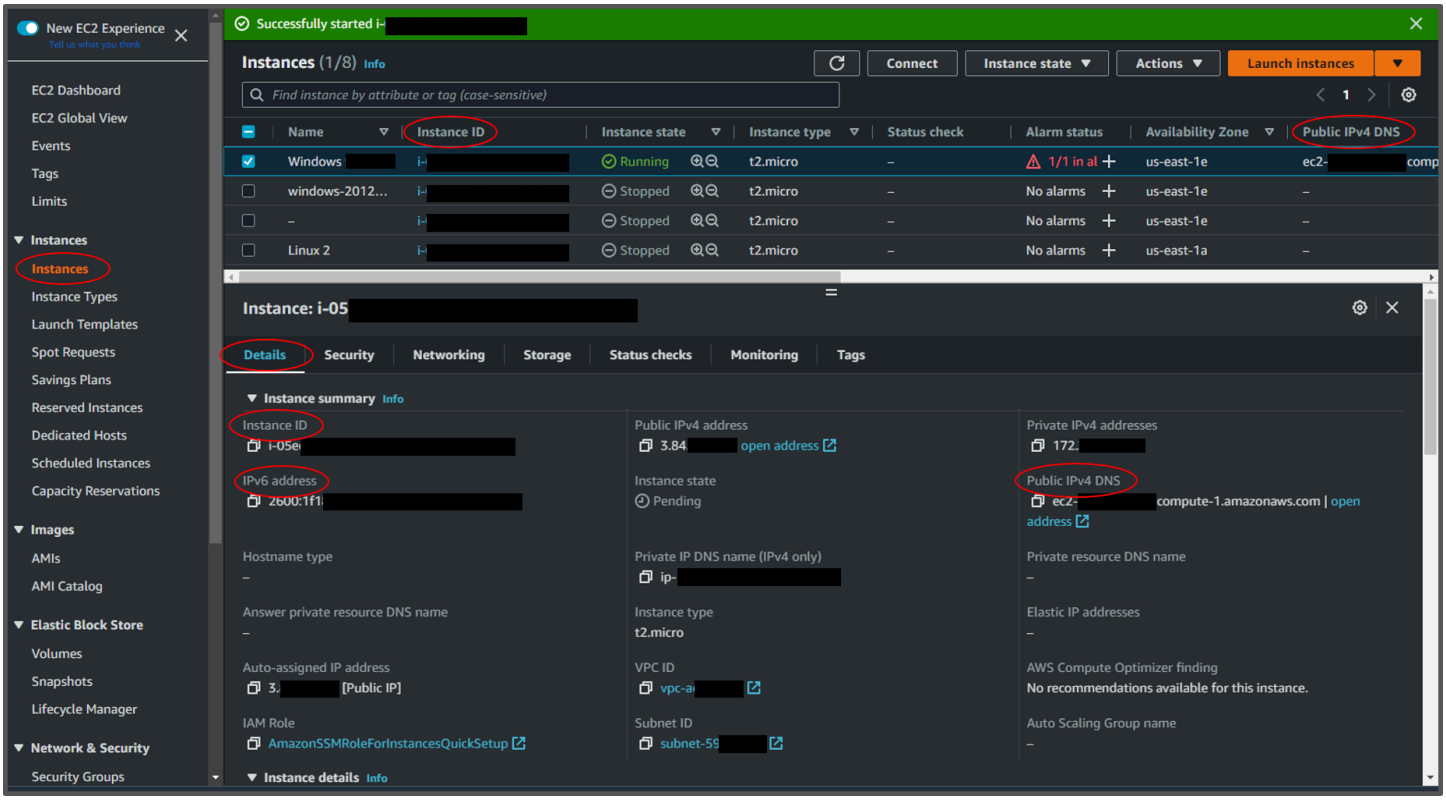
インスタンスに接続するための一般的な前提条件を次に示します。選択した接続オプションに固有の前提条件が別に存在する場合もあります。

一般的な前提条件

- インスタンスのステータスチェックが成功していることを確認します。インスタンスが接続リクエストを受け入れる準備が整うまでに、数分かかることがあります。詳細については、「ステータスチェックの表示」を参照してください。
- 必要なインスタンスの詳細を取得する.
- プライベートキーを見つけ、許可を設定する.
- (オプション) インスタンスのフィンガープリントを取得する.

必要なインスタンスの詳細を取得する

インスタンスに接続する準備をするには、Amazon EC2 コンソールまたはコマンドラインを使用して次の情報を取得します。



- インスタンスのパブリック DNS 名を取得します。

  Amazon EC2 コンソールから、インスタンスのパブリック DNS を取得できます。[インスタンス] ペインの [パブリック IPv4 DNS] 列を確認します。この列が非表示になっている場合は、画面右上部にある設定アイコン を選択し、[パブリック IPv4 DNS] を選択してください。パブリック DNS は、[インスタンス] ペインのインスタンス情報セクションにもあります。Amazon EC2 コンソールの [インスタンス] ペインでインスタンスを選択すると、そのインスタンスに関する情報がページの下半分に表示されます。[詳細] タブで、[パブリック IPv4 DNS] を探します。

  その代わりに、describe-instances AWS CLIまたは Get-EC2Instance AWS Tools for Windows PowerShellコマンドを使用することもできます。

  [パブリック IPv4 DNS] が表示されない場合は、[インスタンスの状態] が [実行中] であり、プライベートサブネットでインスタンスを起動していないことを確認します。インスタンス起動ウィザードを使用してインスタンスを起動した場合、[ネットワーク設定] の [パブリック IP の自動割り当て] フィールドを編集して、値を [無効] に変更した可能性があります。[パブリック IP の自動割り当て] オプションを無効にすると、インスタンスは起動時にパブリック IP アドレスが割り当てられません。

- (IPv6 のみのインスタンス) インスタンスの IPv6 アドレスを取得します。

  自分のインスタンスに IPv6 アドレスを割り当てている場合は、オプションで、パブリック IPv4 アドレスまたはパブリック IPv4 DNS のホスト名の代わりに、IPv6 アドレスを使用してインスタンスに接続することも可能です。ローカルコンピュータに IPv6 アドレスがあり、IPv6 を使用するように設定されている必要があります。Amazon EC2 コンソールから、インスタンスの IPv6 アドレスを取得できます。[インスタンス] ペインの [IPv6 IPs] 列を確認します。または、インスタンス情報セクションで IPv6 アドレスを確認できます。Amazon EC2 コンソールの [インスタンス] ペインでインスタンスを選択すると、そのインスタンスに関する情報がページの下半分に表示されます。[詳細] タブで、[IPv6 アドレス] を探します。

  その代わりに、describe-instances AWS CLIまたは Get-EC2Instance AWS Tools for Windows PowerShellコマンドを使用することもできます。IPv6 の詳細については、IPv6 アドレスを参照してください。

- (Linux インスタンス) インスタンスのユーザー名を取得します。

  インスタンスに接続するにはユーザーアカウントのユーザー名、またはインスタンスの起動に使用した AMI のデフォルトのユーザー名を使用します。

  - ユーザーアカウントのユーザー名を取得します。

    ユーザーアカウントの作成方法についてはAmazon EC2 Linux インスタンスのシステムユーザーを管理するを参照してください。

  - インスタンスの起動に使用した AMI のデフォルトのユーザー名を取得します。
    - Amazon Linux - `ec2-user`
    - CentOS – `centos` または `ec2-user`
    - Debian - `admin`
    - Fedora – `fedora` または `ec2-user`
    - FreeBSD – `ec2-user`
    - RHEL – `ec2-user` または `root`
    - SUSE – `ec2-user` または `root`
    - Ubuntu - `ubuntu`
    - Oracle – `ec2-user`
    - Bitnami – `bitnami`
    - Rocky Linux – `rocky`
    - その他 – AMI プロバイダーに確認してください。

プライベートキーを見つけ、許可を設定する

SSH を使用した Linux インスタンスへ初期接続、または RDP を使用した Windows インスタンスへの初期接続を行うには、プライベートキーファイルの場所を知る必要があります。SSH 接続の場合、お客様のみがプライベートキーを読み込みできるようにファイルのアクセス許可を設定する必要があります。

Amazon EC2 を使用する際のキーペアの仕組みについては、「Amazon EC2 のキーペアと Amazon EC2 インスタンス」を参照してください。

- プライベートキーを見つけます。

  インスタンスの起動時に指定したキーペアの .pem ファイルの、コンピュータ上の場所への完全修飾パスを取得します。詳細については、「the section called “起動時に指定されたパブリックキーを特定する”」を参照してください。

  プライベートキーファイルが見つからない場合は、「プライベートキーを紛失しました。自分のインスタンスに接続するにはどうすればよいですか?」を参照してください。

  (Linux インスタンス) PuTTY を使用してインスタンスに接続していて、.pem ファイルを .ppk に変換する必要がある場合は、「PuTTYgen を使用してプライベートキーを変換する」を参照してください。

- (Linux インスタンス) プライベートキーへのアクセス許可を設定し、お客様のみが読み込みできるようにします。

  - macOS または Linux から接続する

    macOS または Linux コンピュータの SSH クライアントを使用して Linux インスタンスに接続する予定がある場合は、自分以外のユーザーが読み込むことができないように、次のコマンドを使用してプライベートキーファイルの許可を設定します。

    ```
    chmod 400 key-pair-name.pem
    ```

    これらのアクセス権限を設定しないと、このキーペアを使用してインスタンスに接続できません。詳細については、「エラー: Unprotected Private キー ファイル (保護されていないプライベートキーファイル)」を参照してください。

  - Windows から接続する

ファイルエクスプローラーを開き、.pem ファイルを右クリックしてください。[プロパティ] を選択し、[セキュリティ] タブ、[詳細設定] の順に選択してください。それから [継承の無効化] を選択してください。現在のユーザーを除くすべてのユーザーのアクセスを削除します。

(オプション) インスタンスのフィンガープリントを取得する

中間者攻撃から自身を保護する場合、表示されるフィンガープリントを検証することで、接続しようとしているインスタンスの信頼性を検証できます。フィンガープリントの検証は、サードパーティが提供するパブリック AMI からインスタンスを起動した場合に役立ちます。

タスクの概要

まず、インスタンスのインスタンスフィンガープリントを取得します。次に、インスタンスに接続してフィンガープリントを確認するように求められたら、この手順で取得したフィンガープリントと表示されるフィンガープリントを比較します。これらのフィンガープリントが一致しない場合、何者かが中間者 (MITM) 攻撃を試みている可能性があります。一致する場合には、安心してインスタンスに接続できます。

インスタンスのフィンガープリントを取得するための前提条件

- インスタンスが pending の状態であってはいけません。フィンガープリントは、インスタンスの最初の起動が完了した後にのみ使用できます。
- コンソール出力を取得するには、インスタンス所有者である必要があります。
- インスタンスフィンガープリントを取得するには、さまざまな方法があります。AWS CLI を使用する場合は、ローカルコンピュータにインストールする必要があります。AWS CLI のインストールについては、AWS Command Line Interface デベロッパーガイドの Getting started with the AWS CLIを参照し.てください。

インスタンスのフィンガープリントを取得するには

ステップ 1 では、インスタンスフィンガープリントを含むコンソール出力を取得します。ステップ 2 では、コンソール出力でインスタンスフィンガープリントを見つけます。

1. コンソール出力を取得するには、以下のいずれかの方法を使用します。

   Console

   1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

2. 左のナビゲーターの [インスタンス] を選択してください。
3. インスタンスを選択してから、[アクション]、[モニタリングとトラブルシューティング]、[システムログを取得] の順に選択してください。

AWS CLI

使用するローカルコンピュータ (お客様が接続しているインスタンス上ではなく) で get-console-output コマンドを使用します。出力が大きい場合は、出力をテキストファイルにパイプして読み込みやすいようにすることができます。

```
aws ec2 get-console-output \
    --instance-id i-1234567890abcdef0 \
    --query Output \
    --output text > temp.txt
```

PowerShell

ローカルコンピュータで、次の Get-EC2ConsoleOutput コマンドレットを使用します。

```
$encodedOutput = (Get-EC2ConsoleOutput -InstanceId i-1234567890abcdef0).Output
[System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String($encodedOutput
```

2. コンソール出力で、BEGIN SSH HOST KEY FINGERPRINTS にあるインスタンス (ホスト) フィンガープリントを見つけます。インスタンスフィンガープリントが複数ある場合があります。インスタンスに接続すると、フィンガープリントの 1 つだけが表示されます。

正確な出力は、オペレーティングシステム、AMI バージョン、AWS でキーペアを作成したかどうかによって異なります。以下は出力の例です。

```
ec2:#############################################################
ec2: -----BEGIN SSH HOST KEY FINGERPRINTS-----
ec2: 256 SHA256:l4UB/neBad9tvkgJf1QZWxheQmR59WgrgzEimCG6kZY no comment (ECDSA)
ec2: 256 SHA256:kpEa+rw/Uq3zxaYZN8KT501iBtJOIdHG52dFi66EEfQ no comment (ED25519)
ec2: 2048 SHA256:L8l6pepcA7iqW/jBecQjVZClUrKY+o2cHLI0iHerbVc no comment (RSA)
ec2: -----END SSH HOST KEY FINGERPRINTS-----
ec2: #############################################################
```

> **Note**
>
> このフィンガープリントは、インスタンスに接続するときに参照してください。

SSH を使用した Linux インスタンスへの接続

SSH を使用して Linux インスタンスに接続するには、いくつかの方法があります。接続元のローカルコンピュータのオペレーティングシステムによっては、接続方法が異なることもあります。また、EC2 Instance Connect や AWS Systems Manager Session Manager などブラウザベースの方法もあり、こうした方法はどのコンピュータからでも使用できます。SSH を使用して Linux インスタンスに接続してコマンドを実行することも、SSH を使用してローカルコンピュータとインスタンス間でファイルを転送することもできます。

SSH を使用して Linux インスタンスに接続する前に、以下の前提条件を満たしていることを確認してください。

- インスタンスのステータスチェックが成功していることを確認します。インスタンスが接続リクエストを受け入れる準備が整うまでに、数分かかることがあります。詳細については、「ステータスチェックの表示」を参照してください。
- インスタンスに関連付けられているセキュリティグループで、IP アドレスからの受信 SSH トラフィックが許可されていることを確認します。詳細については、「コンピュータからのインスタンスへの接続ルール」を参照してください。
- 必要なインスタンスの詳細を取得する.
- プライベートキーを見つけ、許可を設定する.
- (オプション) インスタンスのフィンガープリントを取得する.

次に、以下のオプションのいずれかを選択して SSH で Linux インスタンスに接続します。

- SSH クライアントを使用して接続する
- PuTTY を使用した接続
- SCP を使用してファイルを転送する

インスタンスに接続できず、トラブルシューティングの手助けが必要な場合は、「Amazon EC2 Linux インスタンスへの接続に関する問題のトラブルシューティング」を参照してください。

SSH クライアントを使用して Linux インスタンスに接続する

Secure Shell (SSH) を使用して、ローカルコンピュータから Linux インスタンスに接続できます。その他のオプションの詳細については、「EC2 インスタンスに接続する」を参照してください。

> **Note**
>
> インスタンスに接続しようとしているときにエラーが発生した場合は、インスタンスが SSH 接続の前提条件 のすべてを満たしていることを確認してください。前提条件をすべて満たしているにもかかわらず Linux インスタンスに接続できない場合は、「Amazon EC2 Linux インスタンスへの接続に関する問題のトラブルシューティング」を参照してください。

内容

- SSH 接続の前提条件
- SSH クライアントを使用して Linux インスタンスに接続する

SSH 接続の前提条件

SSH を使用して Linux インスタンスに接続する前に、以下のタスクを実行してください。

一般的な前提条件を満たします。

- インスタンスのステータスチェックが成功していることを確認します。インスタンスが接続リクエストを受け入れる準備が整うまでに、数分かかることがあります。詳細については、「ステータスチェックの表示」を参照してください。
- 必要なインスタンスの詳細を取得する.
- プライベートキーを見つけ、許可を設定する.
- (オプション) インスタンスのフィンガープリントを取得する.

IP アドレスからのインバウンド SSH トラフィックを許可します。

インスタンスに関連付けられているセキュリティグループで、IP アドレスからの受信 SSH トラフィックが許可されていることを確認します。詳細については、「コンピュータからのインスタンスへの接続ルール」を参照してください。

ローカルコンピュータに SSH クライアントをインストールします (必要な場合)。

ローカルコンピュータには、デフォルトで SSH クライアントがインストールされている場合があります。これを確認するには、ターミナルウィンドウで次のコマンドを入力します。ご使用の

コンピュータでこのコマンドが認識されない場合、SSH クライアントをインストールする必要があります。

```
ssh
```

Windows で使用できるオプションの一部を次に示します。コンピュータで別のオペレーティングシステムを実行している場合は、そのオペレーティングシステムのドキュメントで SSH クライアントのオプションを確認してください。

Windows に OpenSSH をインストールする

Windows に OpenSSH をインストールしたら、SSH を使用して Windows コンピュータから Linux インスタンスに接続できます。開始する前に、以下の前提条件を満たしていることを確認してください。

Windows のバージョン

コンピュータの Windows のバージョンは、Windows Server 2019 以降である必要があります。

それより前のバージョンの Windows の場合は、代わりに Win32-OpenSSH をダウンロードしてインストールします。

PowerShell の要件

PowerShell を使用して Windows OS に OpenSSH をインストールするには、PowerShell バージョン 5.1 以降を実行していて、アカウントがビルトインの管理者グループのメンバーである必要があります。PowerShell から `$PSVersionTable.PSVersion` を実行して、PowerShell のバージョンを確認します。

自分がビルトインの管理者グループのメンバーかどうかを確認するには、次の PowerShell コマンドを実行します。

```
(New-Object
 Security.Principal.WindowsPrincipal([Security.Principal.WindowsIdentity]::GetCurrent())).Is
```

ビルトインの管理者グループのメンバーである場合、出力は `True` です。

PowerShell を使用して OpenSSH for Windows をインストールするには、次の PowerShell コマンドを実行します。

```
Add-WindowsCapability -Online -Name OpenSSH.Client~~~~0.0.1.0
```

以下は出力例です。

```
Path          :
Online        : True
RestartNeeded : False
```

PowerShell を使用して Windows から OpenSSH をアンインストールするには、次の PowerShell コマンドを実行します。

```
Remove-WindowsCapability -Online -Name OpenSSH.Client~~~~0.0.1.0
```

以下は出力例です。

```
Path          :
Online        : True
RestartNeeded : True
```

Windows Subsystem for Linux (WSL) をインストールする

Windows に WSL をインストールしたら、SSH クライアントなどの Linux コマンドラインツールを使用して、Windows コンピュータから Linux インスタンスに接続できます。

「EC2 Windows インスタンスに Windows Subsystem for Linux をインストールする」の手順に従います。Microsoft のインストールガイドの手順に従うと、Linux の Ubuntu ディストリビューションがインストールされます。必要に応じて、別の Linux ディストリビューションをインストールできます。

WSL ターミナルウィンドウで、Windows から WSL に .pem ファイル (インスタンスの起動時に指定したキーペアの場合) をコピーします。インスタンスに接続する際に使用する、WSL の .pem ファイルへの完全修飾パスをメモします。Windows ハードドライブへのパスを指定する方法の詳細については、C ドライブにアクセスする方法を参照してください。

```
cp /mnt/<Windows drive letter>/path/my-key-pair.pem ~/WSL-path/my-key-pair.pem
```

Windows Subsystem for Linux のアンインストールの詳細については、WSL ディストリビューションをアンインストールする方法について参照してください。

SSH クライアントを使用して Linux インスタンスに接続する

SSH クライアントを使用して Linux インスタンスに接続するには、次の手順に従います。

SSH クライアントを使用してインスタンスに接続するには

1. コンピュータでターミナルウィンドウを開きます。
2. ssh コマンドを使用してインスタンスに接続します。ここで、前提条件の一部として収集したインスタンスの詳細が必要になります。例えば、プライベートキー (.pem ファイル) の場所、ユーザー名、およびパブリック DNS 名または IPv6 アドレスが必要です。コマンドの例を次に示します。
   - (パブリック DNS) パブリック DNS 名を使用するには、次のコマンドを入力します。

     ```
     ssh -i /path/key-pair-name.pem instance-user-name@instance-public-dns-name
     ```
   - (IPv6) インスタンスに IPv6 アドレスがある場合、IPv6 アドレスを使用するには、次のコマンドを入力します。

     ```
     ssh -i /path/key-pair-name.pem instance-user-name@2001:db8::1234:5678:1.2.3.4
     ```

   以下に、応答の例を示します。

   ```
   The authenticity of host 'ec2-198-51-100-1.compute-1.amazonaws.com (198-51-100-1)'
    can't be established.
   ECDSA key fingerprint is l4UB/neBad9tvkgJf1QZWxheQmR59WgrgzEimCG6kZY.
   Are you sure you want to continue connecting (yes/no)?
   ```
3. (オプション) セキュリティアラートのフィンガープリントが、インスタンスのフィンガープリントと一致することを確認します。これらのフィンガープリントが一致しない場合、何者かが中間者 (MITM) 攻撃を試みている可能性があります。一致した場合は、次のステップに進んでください。詳細については、「インスタンスのフィンガープリントを取得する」を参照してください。
4. **yes** と入力します。

   以下のようなレスポンスが表示されます。

   ```
   Warning: Permanently added 'ec2-198-51-100-1.compute-1.amazonaws.com' (ECDSA) to
    the list of known hosts.
   ```

PuTTY を使用して Linux インスタンスに接続する

Windows 向けの無料の SSH クライアントである PuTTY を使用して、Linux インスタンスに接続できます。

Windows Server 2019 以降を実行している場合は、SSH プロトコルを使用したリモートログイン用のオープンソース接続ツールである OpenSSH の使用をお勧めします。

> **Note**
>
> インスタンスに接続しようとしているときにエラーが発生した場合は、インスタンスが SSH 接続の前提条件 のすべてを満たしていることを確認してください。前提条件をすべて満たしているにもかかわらず Linux インスタンスに接続できない場合は、「Amazon EC2 Linux インスタンスへの接続に関する問題のトラブルシューティング」を参照してください。

内容

- 前提条件
- (オプション) PuTTYgen を使用してプライベートキーを変換する
- Linux インスタンスへの接続

前提条件

PuTTY を使用して Linux インスタンスに接続する前に、以下のタスクを完了しておきます。

一般的な前提条件を満たします。

- インスタンスのステータスチェックが成功していることを確認します。インスタンスが接続リクエストを受け入れる準備が整うまでに、数分かかることがあります。詳細については「ステータスチェックの表示」を参照してください。
- 必要なインスタンスの詳細を取得する.
- プライベートキーを見つけ、許可を設定する.
- (オプション) インスタンスのフィンガープリントを取得する.

IP アドレスからのインバウンド SSH トラフィックを許可します。

インスタンスに関連付けられているセキュリティグループで、IP アドレスからの受信 SSH トラフィックが許可されていることを確認します。詳細については「コンピュータからのインスタンスへの接続ルール」を参照してください。

ローカルコンピュータに PuTTY をインストールする (必要な場合)。

PuTTY のダウンロードページから、PuTTY をダウンロードしてインストールします。既にインストールされている旧バージョンの PuTTY がある場合は、最新バージョンをダウンロードすることをお勧めします。必ずスイート全体をインストールします。

PuTTYgen を使用してプライベートキーを PPK 形式に変換する。

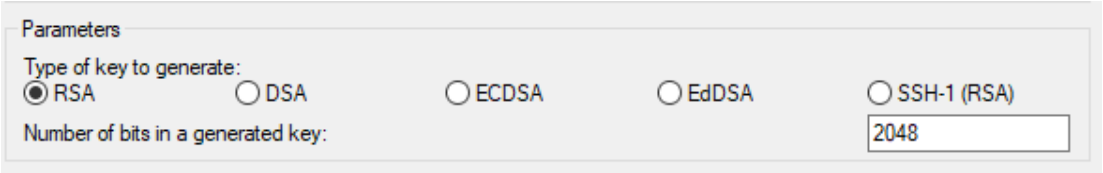
インスタンスの起動時に指定したキーペアのプライベートキーを指定する必要があります。プライベートキーを .pem 形式で作成した場合は、PPK ファイルに変換して PuTTY で使用できるようにする必要があります。プライベート (.pem) ファイルを検索し、PuTTYgen を使用してプライベートキーを変換するのステップに従ってください。

(オプション) PuTTYgen を使用してプライベートキーを変換する

PuTTY は、SSH キーの PEM 形式をネイティブにサポートしていません。PuTTY には、PEM キーを PuTTY が必要とする PPK 形式に変換する PuTTYgen というツールが用意されています。PPK 形式ではなく PEM 形式を使用してキーを作成した場合は、プライベートキー (.pem ファイル) をこの形式 (.ppk ファイル) に変換して、PuTTY で使用できるようにする必要があります。

プライベートキーを PEM 形式から PPK 形式に変換するには

1. [スタート] メニューで、[すべてのプログラム]、[PuTTY]、[PuTTYgen] の順に選択します。
2. [Type of key to generate (生成するキーのタイプ)] で、[RSA] を選択します。お使いの PuTTYgen のバージョンにこのオプションが含まれていない場合は、[SSH-2 RSA] を選択します。



3. ロード] を選択します。PuTTYgen では、デフォルトでは .ppk 拡張子を持つファイルだけが表示されます。.pem ファイルの場所を特定するには、すべてのタイプのファイルを表示するオプションを選択します。

4. インスタンスの起動時に指定したキーペアの .pem ファイルを選択し、[開く] を選択します。PuTTYgen により、.pem ファイルが正常にインポートされたことが表示されます。[OK] を選択します。
5. プライベートキーを PuTTY で使用できる形式で保存するには、[プライベートキーを保存] を選択します。PuTTYgen に、パスフレーズなしでキーを保存することに関する警告が表示されます。[Yes] を選択します。

> **Note**
>
> プライベートキーのパスフレーズは追加の保護レイヤーです。プライベートキーが検出されても、パスフレーズがなければ使用できません。パスフレーズを使用することの欠点は、インスタンスにログオンしたり、ファイルをインスタンスにコピーしたりするのに人間の介入が必要となるため、オートメーションが難しくなることです。

6. キーペアに使用した名前と同じ名前 (key-pair-name など) をキーに指定し、[保存] を選択します。PuTTY により、.ppk ファイルに拡張子が自動的に追加されます。

プライベートキーが PuTTY で使用するための正しい形式となりました。これで、PuTTY の SSH クライアントを使用してインスタンスに接続することができます。

Linux インスタンスへの接続

PuTTY を使用して Linux インスタンスに接続するには、次の手順に従います。秘密キーに作成した .ppk ファイルが必要になります。詳細については、前のセクションの(オプション) PuTTYgen を使用してプライベートキーを変換するを参照してください。インスタンスの接続で エラーが発生した場合は、Amazon EC2 Linux インスタンスへの接続に関する問題のトラブルシューティングを参照してください。

最後にテストされたバージョン: PuTTY .78

PuTTY を使用してインスタンスに接続するには

1. PuTTY を起動します ([スタート] メニューから [PuTTY] を検索し、[開く] を選択します)。
2. [Category (カテゴリ)] ペインで [Session (セッション)] を選択し、次のフィールドに入力します。

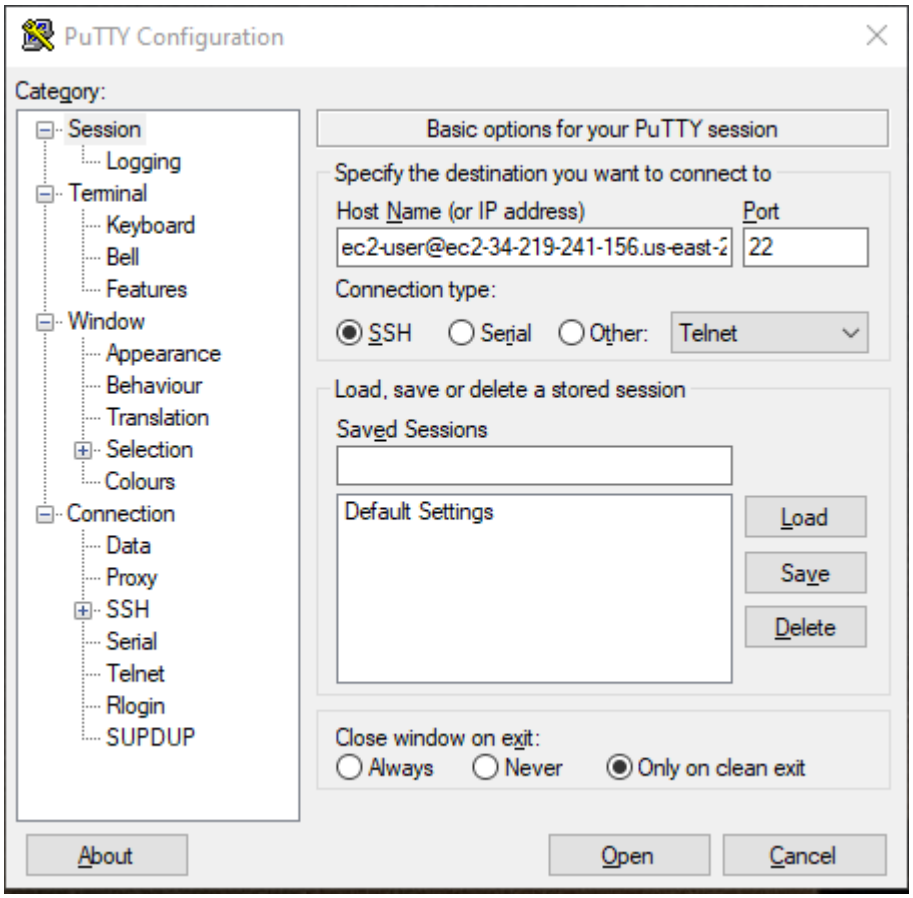
   a. [Host Name (ホスト名)] ボックスで、次のいずれかの操作を行います。

- (パブリック DNS) インスタンスのパブリック DNS 名を使用して接続するには、*instance-user-name*@*instance-public-dns-name* と入力します。
- (IPv6) インスタンスに IPv6 アドレスがある場合、インスタンスの IPv6 アドレスを使用して接続するには、*instance-user-name*@*2001:db8::1234:5678:1.2.3.4* と入力します。

インスタンスのユーザー名、およびインスタンスのパブリック DNS 名または IPv6 アドレスを取得する方法については、「必要なインスタンスの詳細を取得する」を参照してください。

b. [Port (ポート)] の値が 22 であることを確認します。

c. [Connection type (接続タイプ)] で [SSH] を選択します。



3. (オプション) セッションをアクティブに保つため、定期的にキープアライブデータを自動的に送信するように PuTTY を設定できます。これは、セッションがアイドル状態になった際にインスタンスから切断されないようにするのに便利です。[カテゴリー] ペインで [接続] を選択し、[キープアライブ間の秒数] フィールドで必要な間隔を入力します。例えば、10 分間アイド

ル状態が続いた後にセッションが切断される場合、180 と入力して PuTTY を設定し、キープアライブデータを 3 分ごとに送信するようにします。

4. [カテゴリー] ペインで、[接続]、[SSH] の順に展開し、[Auth] を選択します。[認証情報] を選択します。
5. [認証用プライベートキーファイル] の横にある [参照] を選択します。[プライベートキーファイルの選択] ダイアログで、.ppkキーペア用に生成したファイルを選択します。ファイルをダブルクリックするか、[プライベートキーファイルの選択] ダイアログで [開く] を選択します。
6. (オプション) このセッションの後にインスタンスに再度接続する場合は、今後使用するためにセッション情報を保存できます。[カテゴリー] ペインで、[セッション] を選択します。[保存されたセッション] にセッションの名前を入力し、[保存] を選択します。
7. インスタンスに接続するには、[開く] を選択します。
8. このインスタンスに接続するのが初めての場合、PuTTY は接続先のホストを信頼するかどうかを確認するセキュリティアラートダイアログボックスを表示します。

   a. (オプション) セキュリティアラートダイアログボックスのフィンガープリントが、(オプション) インスタンスのフィンガープリントを取得する で前に取得したフィンガープリントと一致することを確認します。これらのフィンガープリントが一致しない場合、「中間者 (MITM)」攻撃を受けている可能性があります。一致した場合は、次のステップに進んでください。

   b. [Accept (承諾)] を選択します。ウィンドウが開き、インスタンスに接続した状態になります。

> **Note**
>
> プライベートキーを PuTTY フォーマットに変換するときにパスフレーズを指定した場合は、インスタンスにログインする際にそのパスフレーズを指定する必要があります。

インスタンスの接続でエラーが発生した場合は、Amazon EC2 Linux インスタンスへの接続に関する問題のトラブルシューティングを参照してください。

SCP を使用して Linux インスタンスにファイルを転送する

ローカルコンピュータと Linux インスタンスの間でファイルを転送する方法の 1 つとして、セキュアコピープロトコル (SCP) を使用します。SCP は、1 回限りのファイルコピーなどの簡単な操作に

適しています。SCP は、SSH を使用してインスタンスに接続する時に使用するのと同じ .pem ファイルを使用して、ファイルの転送を保護します。ファイルの同期を維持する必要がある場合、またはファイルが大きい場合、rsync は SCP よりも高速で効率的です。rsync はデフォルトではプレーンテキストを使用してデータを転送するため、セキュリティ確保のため rsync は SSH 経由で使用すること。

SCP を使用して Linux インスタンスに接続する前に、以下のタスクを完了してください。

- 一般的な前提条件を満たします。
  - インスタンスのステータスチェックが成功していることを確認します。インスタンスが接続リクエストを受け入れる準備が整うまでに、数分かかることがあります。詳細については、「ステータスチェックの表示」を参照してください。
  - 必要なインスタンスの詳細を取得する.
  - プライベートキーを見つけ、許可を設定する.
  - (オプション) インスタンスのフィンガープリントを取得する.
- IP アドレスからのインバウンド SSH トラフィックを許可します。

  インスタンスに関連付けられているセキュリティグループで、IP アドレスからの受信 SSH トラフィックが許可されていることを確認します。詳細については、「コンピュータからのインスタンスへの接続ルール」を参照してください。
- SCP クライアントをインストールします。

  ほとんどの Linux、Unix、および Apple コンピュータには、デフォルトで SCP クライアントが含まれています。含まれていない場合は、OpenSSH プロジェクトから、SSH ツールの完全なスイートの無料実装が提供されており、これに SCP クライアントが含まれます。詳細については、https://www.openssh.com を参照してください。

以下では、インスタンスのパブリック DNS 名、またはインスタンスに IPv6 アドレスがある場合は IPv6 アドレスを使用し、SCP でファイルを転送する手順を示します。

SCP を使用してコンピュータとインスタンス間でファイルを転送するには

1. コンピュータ上のソースファイルの場所と、インスタンス上の送信先パスを決定します。以下の例では、プライベートキーファイルの名前が `key-pair-name.pem`、転送するファイルが `my-file.txt`、インスタンスのユーザー名が ec2-user、インスタンスのパブリック DNS の名前が `instance-public-dns-name` で、インスタンスの IPv6 アドレスが `2001:db8::1234:5678:1.2.3.4` です。

- (パブリック DNS) インスタンスの送信先にファイルを転送するには、コンピュータから次のコマンドを入力します。

```
scp -i /path/key-pair-name.pem /path/my-file.txt ec2-user@instance-public-dns-name:path/
```

- (IPv6) インスタンスに IPv6 アドレスがある場合、インスタンスの送信先にファイルを転送するには、コンピュータから次のコマンドを入力します。IPv6 アドレスは、(\) でエスケープした角かっこ ([ ]) で囲む必要があります。

```
scp -i /path/key-pair-name.pem /path/my-file.txt ec2-user@
\[2001:db8::1234:5678:1.2.3.4\]:path/
```

2. SSH を使用してインスタンスに接続していない場合は、次のようなレスポンスが表示されます。

```
The authenticity of host 'ec2-198-51-100-1.compute-1.amazonaws.com (10.254.142.33)'
can't be established.
RSA key fingerprint is 1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f.
Are you sure you want to continue connecting (yes/no)?
```

(オプション) オプションで、セキュリティアラートのフィンガープリントがインスタンスのフィンガープリントと一致することを確認できます。詳細については、「(オプション) インスタンスのフィンガープリントを取得する」を参照してください。

yes と入力してください。

3. 転送が成功した場合、レスポンスは以下のようになります。

```
Warning: Permanently added 'ec2-198-51-100-1.compute-1.amazonaws.com' (RSA)
to the list of known hosts.
my-file.txt                                   100%   480     24.4KB/s   00:00
```

4. 逆の方向 (Amazon EC2 インスタンスからコンピュータ) にファイルを転送するには、ホストパラメータの順番を逆にします。例えば、次の例に示すように、EC2 インスタンスからローカルコンピュータの送信先に `my-file.txt` として `my-file2.txt` を転送できます。

- (パブリック DNS) コンピュータの送信先にファイルを転送するには、コンピュータから次のコマンドを入力します。

```
scp -i /path/key-pair-name.pem ec2-user@instance-public-dns-name:path/my-file.txt path/my-file2.txt
```

- (IPv6) インスタンスに IPv6 アドレスがある場合、コンピュータの送信先にファイルを転送するには、コンピュータから次のコマンドを入力します。IPv6 アドレスは、(\) でエスケープした角かっこ ([ ]) で囲む必要があります。

```
scp -i /path/key-pair-name.pem ec2-user@\[2001:db8::1234:5678:1.2.3.4\]:path/my-file.txt path/my-file2.txt
```

Amazon EC2 Linux インスタンスのシステムユーザーを管理する

各 Linux インスタンスは、デフォルトの Linux システムユーザーで起動されます。インスタンスには、ユーザーを追加することも、削除することもできます。

デフォルトのユーザーの場合、デフォルトのユーザー名は、インスタンスの起動時に指定した AMI によって決まります。

> **Note**
>
> デフォルトでは、パスワード認証とルートログインは無効になっており、また、sudo は有効化されています。インスタンスにログインするには、キーペアを使用する必要があります。ログインの詳細については、「SSH を使用した Linux インスタンスへの接続」を参照してください。
> ユーザーは、インスタンスのパスワード認証とルートログインを許可できます。詳細については、「インスタンスのオペレーティングシステムに関するドキュメント」を参照してください。

> **Note**
>
> Linux システムユーザーと IAM ユーザーを混同しないようにしてください。詳細については、『IAM ユーザーガイド』のIAM ユーザーを参照してください。

内容

- デフォルトのユーザー名

- 考慮事項
- ユーザーの作成
- ユーザーの削除

デフォルトのユーザー名

EC2 インスタンスのデフォルトのユーザー名は、インスタンスの起動時に指定した AMI によって決まります。

デフォルトのユーザー名は以下のとおりです。

- Amazon Linux AMI の場合、ユーザー名は `ec2-user` です。
- Centos AMI の場合、ユーザー名は `centos` または `ec2-user` です。
- Debian AMI の場合は、ユーザー名は `admin` です。
- Fedora AMI の場合、ユーザー名は `fedora` または `ec2-user` です。
- FreeBSD AMI の場合、ユーザー名は `ec2-user` です。
- RHEL AMI の場合、ユーザー名は `ec2-user` または `root` です。
- SUSE AMI の場合、ユーザー名は `ec2-user` または `root` です。
- Ubuntu AMI の場合、ユーザー名は `ubuntu` です。
- Oracle AMI の場合、ユーザー名は `ec2-user` です。
- Bitnami AMI の場合、ユーザー名は `bitnami` です。

> **Note**
>
> 他の Linux ディストリビューションのデフォルトのユーザー名を確認するには、AMI プロバイダーに確認してください。

考慮事項

デフォルトのユーザーを使用するのが多くのアプリケーションに適しています。ただし、個人が自分のファイルとワークスペースを持つことができるように、ユーザーを追加することを選択できます。さらに、新しいユーザー用にユーザーを作成することは、デフォルトユーザーへのアクセス権を複数のユーザーに (経験のないユーザーも含めて) 与えるよりも、はるかに安全です。これはデフォルトのユーザーが不適切に使用された場合、システムにさまざまな損害を与える可能性があるためです。

Linux システムのユーザーを使用してユーザーが EC2 インスタンスに SSH アクセスできるようにするには、SSH キーをユーザーと共有する必要があります。または、EC2 Instance Connect を使用して、SSH キーを共有および管理せずにユーザーにアクセスを提供できます。詳細については、「パブリック IP アドレスと EC2 Instance Connect を使用して Linux インスタンスに接続する」を参照してください。

ユーザーの作成

最初にユーザーを作成してから、ユーザーがインスタンスに接続してログインできるようにする SSH パブリックキーを追加します。

> **Important**
>
> この手順のステップ 1 では、新しいキーペアを作成します。キーペアはパスワードのように機能するため、安全に取り扱うことが非常に重要です。ユーザー用のキーペアを作成する場合は、プライベートキーを安全に送信する必要があります。または、ユーザーが自分のマシン上で独自のキーペアを作成し、プライベートキーを安全に保持することでステップ 1 と 2 を完了してから、あなたにパブリックキーを送信してステップ 3 以降の手順を完了してもらうこともできます。

ユーザーを作成するには

1. 新しいキーペアを作成します。この .pem ファイルは、ユーザーを作成するユーザーに提供する必要があります。ユーザーがインスタンスに接続するには、このファイルを使用する必要があります。
2. 前のステップで作成したキーペアからパブリックキーを取得します。

   ```
   $ ssh-keygen -y -f /path_to_key_pair/key-pair-name.pem
   ```

   コマンドは、次の例に示すように、パブリックキーを返します。

   ```
   ssh-rsa
    AAAAB3NzaC1yc2EAAAADAQABAAABAQClKsfkNkuSevGj3eYhCe53pcjqP3maAhDFcvBS7O6Vhz2ItxCih
   +PnDSUaw+WNQn/mZphTk/a/gU8jEzoOWbkM4yxyb/wB96xbiFveSFJuOp/
   d6RJhJOI0iBXrlsLnBItntckiJ7FbtxJMXLvvwJryDUilBMTjYtwB+QhYXUMOzce5Pjz5/
   i8SeJtjnV3iAoG/cQk+0FzZqaeJAAHco
   +CY/5WrUBkrHmFJr6HcXkvJdWPkYQS3xqC0+FmUZofz221CBt5IMucxXPkX4rWi
   +z7wB3RbBQoQzd8v7yeb7OzlPnWOyN0qFU0XA246RA8QFYiCNYwI3f05p6KLxEXAMPLE
   ```

3. インスタンスに接続します。
4. adduser コマンドを使用して、ユーザーを作成し、システムに追加します (/etc/passwd ファイルにエントリが追加されます)。このコマンドでも、グループが作成され、ユーザーのホームディレクトリが作成されます。この例では、ユーザーは *newuser* という名前になります。

   - AL2023 と Amazon Linux 2

     AL2023 および Amazon Linux 2 では、デフォルトでパスワード認証が無効化された状態で、ユーザーが作成されます。

     ```
     [ec2-user ~]$ sudo adduser newuser
     ```

   - Ubuntu

     パスワード認証が無効化されたユーザーを作成するには、--disabled-password パラメータを含めます。

     ```
     [ubuntu ~]$ sudo adduser newuser --disabled-password
     ```

5. 新しいユーザーに切り替えて、作成するディレクトリとファイルが適切な所有権を持つようにします。

   ```
   [ec2-user ~]$ sudo su - newuser
   ```

   シェルセッションが新しいユーザーに切り替わったことを示すために ec2-user から *newuser* に変更するように求められます。
6. ユーザーに SSH パブリックキーを追加します。以下のサブステップで説明しているように、最初に SSH キーファイル用のディレクトリをユーザーのホームディレクトリに作成し、次にキーファイルを作成して、最後に公開キーをキーファイルに貼り付けます。

   a. .ssh ホームディレクトリに *newuser* ディレクトリを作成し、そのファイルのアクセス許可を 700 (所有者のみ、読み取り、書き込み、削除が可能) に変更します。

      ```
      [newuser ~]$ mkdir .ssh
      ```

      ```
      [newuser ~]$ chmod 700 .ssh
      ```

> **Important**
> 厳密なファイル権限がなければ、ユーザーはログインできません。

b. authorized_keys という名前のファイルを .ssh ディレクトリに作成し、そのファイルのアクセス許可を 600 (所有者のみ、読み取りおよび書き込みが可能) に変更します。

```
[newuser ~]$ touch .ssh/authorized_keys
```

```
[newuser ~]$ chmod 600 .ssh/authorized_keys
```

> **Important**
> 厳密なファイル権限がなければ、ユーザーはログインできません。

c. お好みのテキストエディタ (例: vim や nano) で、authorized_keys ファイルを開きます。

```
[newuser ~]$ nano .ssh/authorized_keys
```

ステップ 2 で取得したパブリックキーをファイルに貼り付け、変更を保存します。

> **Important**
> パブリックキーは、必ず 1 つの連続した行に貼り付けてください。パブリックキーを複数行に分割することはできません。

これで、authorized_keys ファイルに追加したパブリックキーの対であるプライベートキーを使用して、インスタンスの *newuser* ユーザーにログインできるようになりました。Linux インスタンスに接続するさまざまな方法の詳細については、SSH を使用した Linux インスタンスへの接続を参照してください。

ユーザーの削除

ユーザーが不要になった場合、今後使用されないようにそのユーザーを削除できます。

システムからユーザーアカウントを削除するには、userdel コマンドを使用します。-r パラメータを指定すると、ユーザーのホームディレクトリとメールスプールが削除されます。ユーザーのホームディレクトリとメールスプールを維持するには、-r パラメータを省略します。

```
[ec2-user ~]$ sudo userdel -r olduser
```

RDP を使用した Windows インスタンスへの接続

ほとんどの Windows Amazon マシンイメージ (AMI) から作成された Amazon EC2 インスタンスは、リモートデスクトップを使用して接続することができます。リモートデスクトップは、Remote Desktop Protocol (RDP) を使用して接続でき、目の前のコンピュータ (ローカルコンピュータ) を使用するのと同じ方法でインスタンスを使用します。これは、Windows のほとんどのエディションで使用でき、Mac OS でも利用できます。

Windows Server オペレーティング システムのライセンスでは、2 つの同時リモート接続を管理目的で使用できます。Windows Server のライセンスは、Windows インスタンスの価格に含まれています。同時リモート接続が 3 つ以上必要な場合は、リモートデスクトップサービス (RDS) ライセンスを購入する必要があります。3 番目の接続を試みると、エラーが発生します。

> **Tip**
>
> AWS Nitro System で構築されたインスタンスの起動、ネットワーク構成、および他の問題をトラブルシューティングするためにインスタンスに接続する必要がある場合は、インスタンス用 EC2 シリアルコンソール を使用できます。

内容

- RDP クライアントを使用して Windows インスタンスに接続する
- Fleet Manager を使用して Windows インスタンスに接続する
- RDP を使用して Windows インスタンスにファイルを転送する

RDP クライアントを使用して Windows インスタンスに接続する

以下のように、RDP クライアントを使用して Windows インスタンスに接続できます。

> **Tip**
>
> または、Systems Manager Fleet Manager または EC2 Instance Connect エンドポイントを使用して Windows インスタンスに接続することもできます。

前提条件

RDP クライアントを使用して Windows インスタンスに接続するには、次の前提条件を満たす必要があります。

- 一般的な前提条件を満たします。
  - インスタンスのステータスチェックが成功していることを確認します。インスタンスが接続リクエストを受け入れる準備が整うまでに、数分かかることがあります。詳細については、「ステータスチェックの表示」を参照してください。
  - 必要なインスタンスの詳細を取得する.
  - プライベートキーを見つけ、許可を設定する.
  - (オプション) インスタンスのフィンガープリントを取得する.
- RDP クライアントをインストールします。
  - (Windows) Windows にはデフォルトで RDP クライアントが備わっています。確認するには、コマンドプロンプトウィンドウで mstsc と入力してください。コンピュータがこのコマンドを認識しない場合は、Microsoft ストアから Microsoft リモートデスクトップアプリをダウンロードします。
  - (macOS X) Mac App Store から Microsoft リモートデスクトップアプリをダウンロードします。
  - (Linux) Remmina を使用します。
- IP アドレスからのインバウンド RDP トラフィックを許可します。

  インスタンスに関連付けられているセキュリティグループで、IP アドレスからの受信 RDP トラフィックが許可されていることを確認します。詳細については、「コンピュータからのインスタンスへの接続ルール」を参照してください。

管理者パスワードを取得する

インスタンスをドメインに参加させた場合は、Directory Service のドメイン認証情報を使用してインスタンスに接続できます。リモートデスクトップのログイン画面で、ローカルコンピュータ名と

生成されたパスワードを使用する代わりに、管理者の完全修飾ユーザー名 (例:**corp.example.com\Admin**) と、そのアカウントのパスワードを入力してください。

RDP を使用して Windows インスタンスに接続するには、初期管理者パスワードを取得し、インスタンスに接続するときにこのパスワードを入力する必要があります。インスタンスの起動後、パスワードが利用可能になるまでに数分かかります。アカウントには、GetPasswordData アクションを呼び出すアクセス許可が必要です。詳細については、「Amazon EC2 API へのアクセスを制御するポリシーの例」を参照してください。

管理者アカウントのデフォルトのユーザー名は、AMI に含まれるオペレーティングシステム (OS) の言語によって異なります。正しいユーザー名を確認するには、AMI の OS の言語を特定し、対応するユーザー名を選択してください。例えば、英語 OS の場合、ユーザー名は `Administrator` で、フランス語 OS の場合は `Administrateur`、ポルトガル語 OS の場合は `Administrador` です。OS の言語バージョンに同じ言語のユーザー名がない場合は、ユーザー名 `Administrator (Other)` を選択してください。詳細については、Microsoft TechNet Wiki のLocalized Names for Administrator Account in Windowsを参照してください。

初期管理者パスワードを取得するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. 対象のインスタンスを選択し、[Connect] (接続) をクリックしてください。
4. [インスタンスに接続] ページで、[RDP クライアント] タブを選択してください。
5. [ユーザー名] で、管理者アカウントのデフォルトのユーザー名を選択してください。選択するユーザー名は、インスタンスの起動に使用した AMI に含まれるオペレーティングシステム (OS) の言語と一致する必要があります。使用する OS と同じ言語のユーザー名がない場合は、[Administrator (Other)] を選択してください。
6. [パスワードを取得] を選択してください。
7. [Windows パスワードを取得] ページで、次の操作を行います。

    a. [プライベートキーファイルのアップロード] を選択し、インスタンスの起動時に指定したプライベートキー `.pem`ファイルに移動します。ファイルを選択した上で、[Open] (開く) を選択して、ファイルの内容をすべてウィンドウにコピーします。

    b. [パスワードを復号化] を選択してください。[Windows パスワードを取得] ページが閉じて、インスタンスのデフォルトの管理者パスワードが、[パスワード] の下に表示されます。前に表示されていた [パスワードを取得] のリンクは削除されます。

c. パスワードをコピーして、安全な場所に保存します。このパスワードはインスタンスに接続するのに必要です。

Windows インスタンスに接続する

次の手順では、Windows のリモートデスクトップ接続クライアント (MSTSC) を使用します。別の RDP クライアントを使用する場合は RDP ファイルをダウンロードし、RDP 接続を確立する手順については RDP クライアントのドキュメントを参照してください。

RDP クライアントを使用して Windows インスタンスに接続するには

1. [インスタンスに接続] ページで、[リモートデスクトップファイルのダウンロード] を選択してください。ファイルのダウンロードが完了したら、[キャンセル] を選択し、[インスタンス] ページに戻ります。RDP ファイルが `Downloads` フォルダにダウンロードされます。
2. `mstsc.exe` を実行して RDP クライアントを開きます。
3. [オプションを表示] を展開して [開く] を選択し、`Downloads` フォルダから .rdp ファイルを選択してください。
4. デフォルトでは、[コンピュータ] はインスタンスのパブリック IPv4 DNS 名で、[ユーザー名] は管理者アカウントです。代わりに IPv6 を使用してインスタンスに接続するには、インスタンスのパブリック IPv4 DNS 名を IPv6 アドレスに置き換えます。必要に応じて、デフォルト設定を確認して変更します。
5. [接続]を選択してください。リモート接続のパブリッシャーが不明であるという警告が表示された場合は、[接続] を選択して続行します。
6. 以前に保存したパスワードを入力し、[OK] を選択してください。
7. 自己署名証明書の性質上、セキュリティ証明書を認証できないという警告が表示されることがあります。次のいずれかを行います。
   - 証明書を信頼する場合は、[はい] を選択してインスタンスに接続します。
   - [Windows] 続行する前に、証明書のサムプリントとシステムログの値を比較して、リモートコンピュータの ID を確認します。[証明書を表示] を選択し、[詳細] タブから [サムプリント] を選択してください。この値を [アクション]、[モニタリングとトラブルシューティング]、[システムログの取得] の `RDPCERTIFICATE-THUMBPRINT` の値と比較します。
   - [Mac OS X] 続行する前に、証明書のフィンガープリントとシステムログの値を比較して、リモートコンピュータの ID を確認します。[証明書を表示] を選択し、[詳細] を展開し、[SHA1 フィンガープリント] を選択してください。この値を [アクション]、[モニタリン

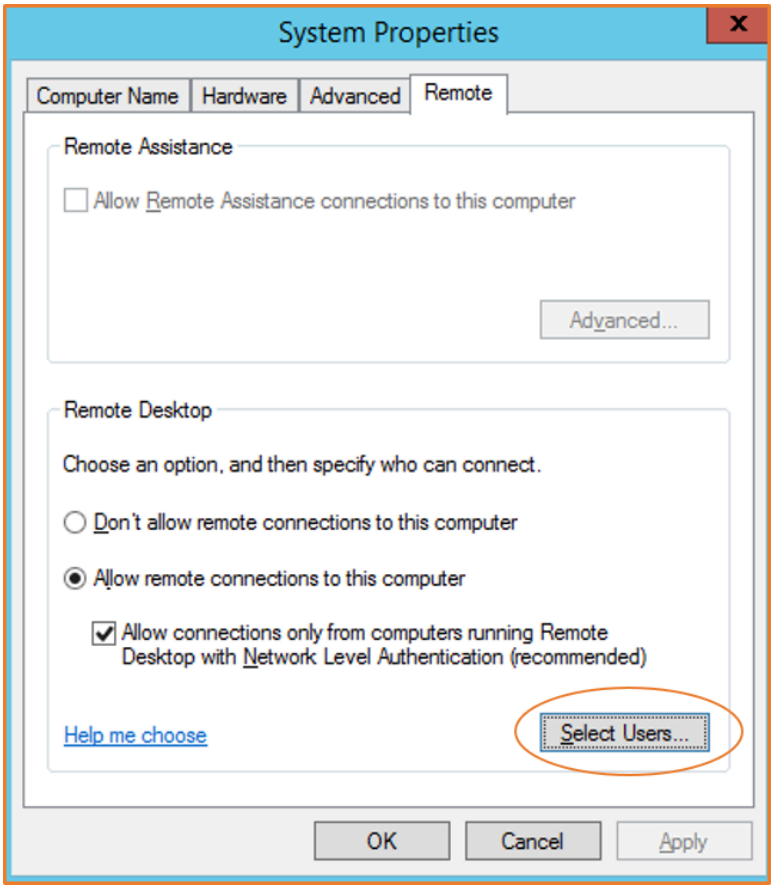
グとトラブルシューティング]、[システムログの取得] の RDPCERTIFICATE-THUMBPRINT の値と比較します。

8. RDP 接続が成功すると、RDP クライアントに Windows ログイン画面が表示され、次に Windows デスクトップが表示されます。代わりにエラーメッセージが表示された場合は、「the section called “リモートデスクトップからリモートコンピュータに接続できません”」を参照してください。RDP 接続が完了したら、RDP クライアントを閉じることができます。

ユーザーアカウントを設定する

RDP でインスタンスに接続したら、次のタスクを実行することをお勧めします。

- 管理者パスワードをデフォルト値から変更します。Windows Server を実行しているコンピュータと同様、パスワードの変更は、インスタンス自体にログインした状態で行うことができます。
- インスタンスに管理者権限を持つユーザーをもう 1 つ作成します。これは、管理者パスワードを忘れた場合や、管理者アカウントで問題が発生した場合の安全策です。新しいユーザーには、インスタンスにリモートからアクセスするための許可が必要です。[システムのプロパティ] を開くには、Windows デスクトップまたはエクスプローラーで [This PC] アイコンを右クリックして [プロパティ] を選択してください。[リモートの設定]、ユーザーの選択] の順に選択して、[リモートデスクトップユーザー] グループにユーザーを追加します。



Fleet Manager を使用して Windows インスタンスに接続する

AWS Systems Manager の機能である Fleet Manager を使用すると、Remote Desktop Protocol (RDP) を使用して Windows インスタンスに接続し、AWS マネジメントコンソール の同じページに最大 4 つの Windows インスタンスを表示できます。Amazon EC2 コンソールの [インスタンス] ページから Fleet Manager Remote Desktop ディレクトリの最初のインスタンスに直接接続できます。Fleet Manager の詳細については、「AWS Systems Manager ユーザーガイド」の「リモートデスクトップを使用して管理対象インスタンスに接続する」を参照してください。

Fleet Manager を使用して接続する場合は、IP アドレスからの着信 RDP トラフィックを特に許可する必要はありません。Fleet Manager が処理します。

前提条件

Fleet Manager を使用してインスタンスに接続する前に、環境をセットアップする必要があります。詳細については、「AWS Systems Manager ユーザーガイド」の「環境のセットアップ」を参照してください。

Fleet Manager を使用して Windows インスタンスに接続するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインから [Instances] を選択してください。
3. 対象のインスタンスを選択し、[Connect] (接続) をクリックしてください。
4. [RDP クライアント] タブの [接続タイプ] で、[Fleet Manager を使用して接続する] を選択してください。
5. [Fleet Manager リモートデスクトップ] を選択してください。これにより、AWS Systems Manager コンソールで [Fleet Manager Remote Desktop] (Fleet Manager リモートデスクトップ) ページが開きます。
6. 認証情報を入力し、[接続] を選択してください。
7. RDP 接続が成功すると、Fleet Manager に Windows デスクトップが表示されます。セッションが終了したら、[アクション]、[セッションの終了] を選択してください。

詳細については、「AWS Systems Manager ユーザーガイド」の「Connecting to a Windows Server managed instance using Remote Desktop」を参照してください。

RDP を使用して Windows インスタンスにファイルを転送する

Windows インスタンスの使い方は、通常の Windows Server と同じです。例えば、Microsoft リモートデスクトップ接続 (RDP) ソフトウェアのローカルファイル共有機能を使用して、Windows インスタンスとローカルコンピュータの間でファイルを転送できます。例えば、ハードディスクドライブ、DVD ドライブ、ポータブルメディアドライブ、およびマップされたネットワークドライブ上のローカルファイルにアクセスすることができます。

Windows インスタンスからローカルファイルにアクセスするには、リモートセッションドライブをローカルドライブにマッピングして、ローカルファイル共有機能を有効にする必要があります。この手順は、ローカルコンピュータのオペレーティングシステムが Windows または macOS X のどちらであるかによって若干異なります。

RDP を使用して接続するための前提条件の詳細については、「前提条件」を参照してください。

Windows

リモートセッションドライブをローカル Windows コンピュータのローカルドライブにマップするには

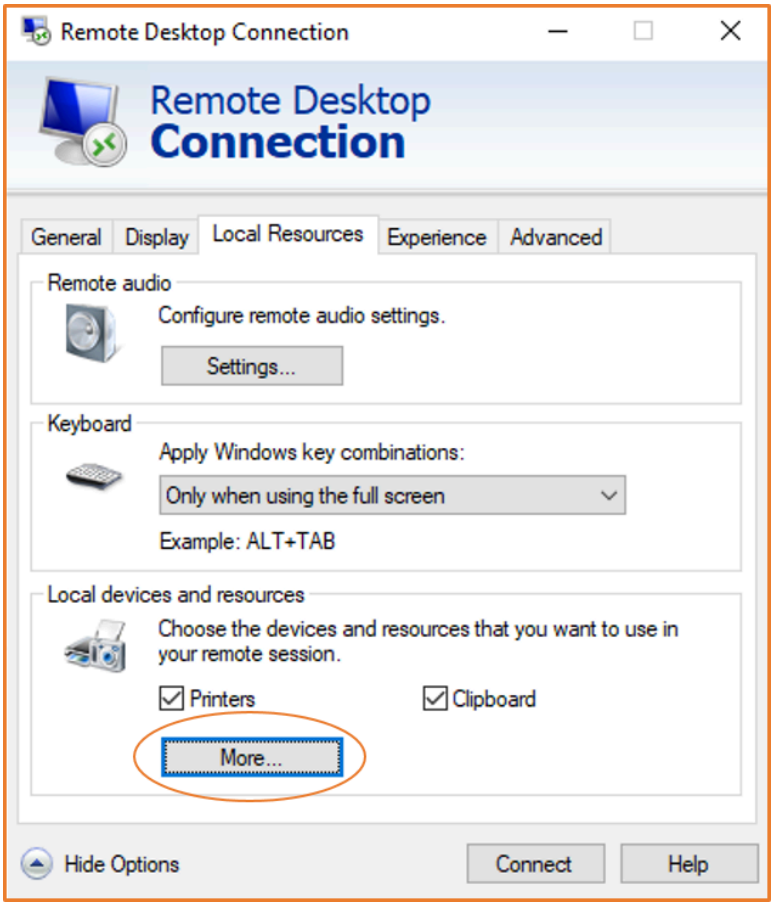
1. リモートデスクトップ接続クライアントを開きます。

2. [Show Options] を選択してください。
3. 次のように、インスタンスのホスト名を [コンピュータ] フィールドに追加し、ユーザー名を [ユーザー名] フィールドに追加します。

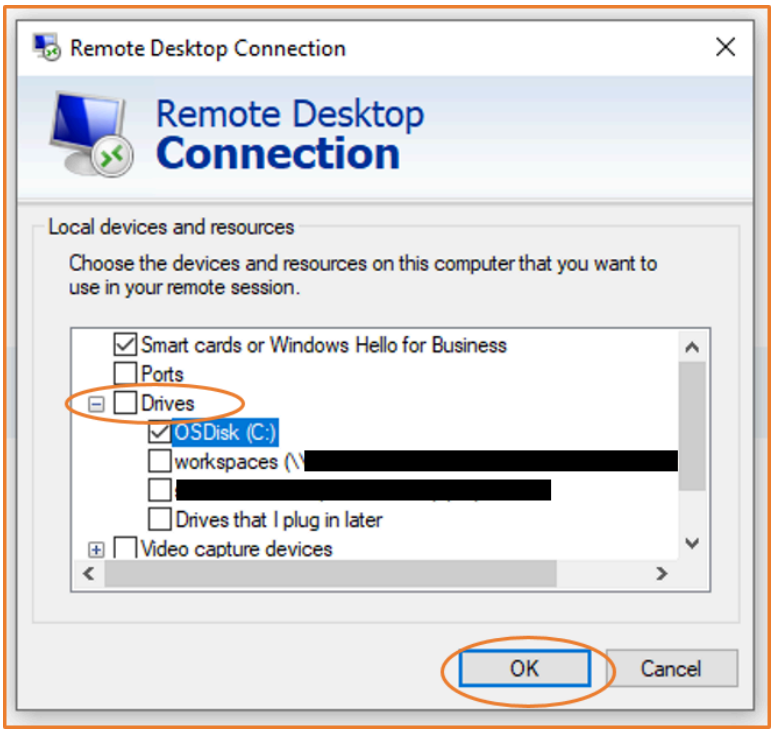
   a. [Connection settings] (接続設定) で、[Open...] (...を開く) を選択し、Amazon EC2 コンソールからダウンロードした RDP ショートカットファイルを参照してください。このファイルには、インスタンスを識別するパブリック IPv4 DNS ホスト名と、管理者ユーザー名が含まれます。

   b. ファイルを選択し、[Open] (開く) を選択してください。[Computer] (コンピュータ) フィールドと [User name] (ユーザー名) フィールドには、RDP ショートカットファイルの値が入力されます。

   c. [Save] を選択してください。
4. [ローカルリソース] タブを選択してください。
5. [Local devices and resources] (ローカルデバイスとリソース) で、[More...] (詳細...) を選択してください。



6. [ドライブ] を開き、Windows インスタンスにマッピングするローカルドライブを選択してください。
7. [OK] を選択してください。

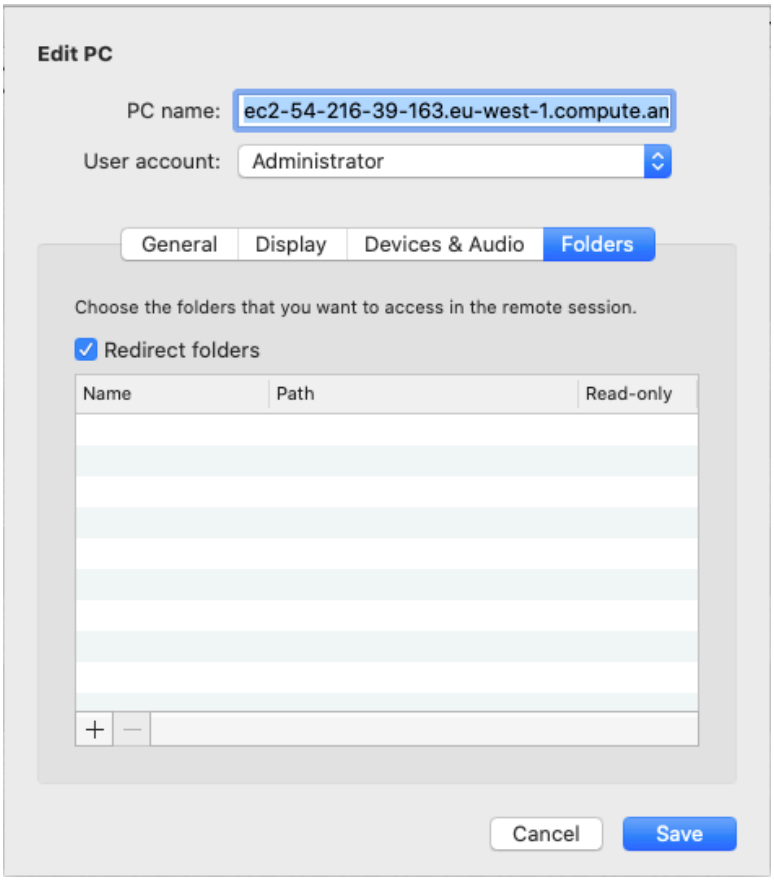


8. [接続] を選択して Windows インスタンスに接続します。

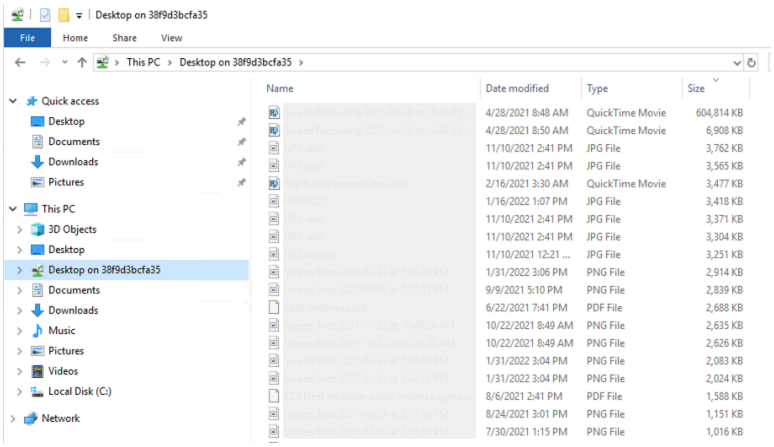
macOS X

ローカルの macOS X コンピュータのローカルフォルダにリモートセッションドライブをマッピングするには

1. リモートデスクトップ接続クライアントを開きます。
2. Amazon EC2 コンソールの インスタンスへの初回接続時) からダウンロードした RDP ファイルを参照し、Remote Desktop Connection クライアントにドラッグします。
3. RDP ファイルを右クリックし、[Edit] (編集) を選択してください。
4. [フォルダ] タブを選択し、[フォルダをリダイレクト] のチェックボックスをオンにします。



5. 左下の + アイコンを選択し、マッピングするフォルダを参照して、[Open] (開く) を選択してください。マッピングする各フォルダについて、この手順を繰り返します。
6. [Save] を選択してください。
7. [接続] を選択して Windows インスタンスに接続します。パスワードを入力するよう求められます。
8. インスタンスのファイルエクスプローラーで [This PC] (この PC) を展開し、ローカルファイルにアクセスできる共有フォルダを見つけます。次のスクリーンショットでは、ローカルコンピュータのデスクトップフォルダがインスタンスのリモートセッションドライブにマッピングされています。



Mac コンピュータのリモートセッションでローカルデバイスを使用できるようにする方法の詳細については、「Get started with the macOS client」(macOS の開始方法) を参照してください。

Session Manager を使用して Amazon EC2 インスタンスに接続する

Session Manager はフルマネージド型の AWS Systems Manager 機能です。ブラウザベースのインタラクティブなワンクリックシェルまたは AWS CLI を介して Amazon EC2 インスタンスを管理します。Session Manager を使用して、アカウント内のインスタンスとのセッションを開始できます。セッションの開始後は、他の接続タイプと同様に、インスタンスでインタラクティブコマンドを実行できます。Session Manager の詳細については、AWS Systems Manager ユーザーガイドのAWS Systems Manager Session Managerを参照してください。

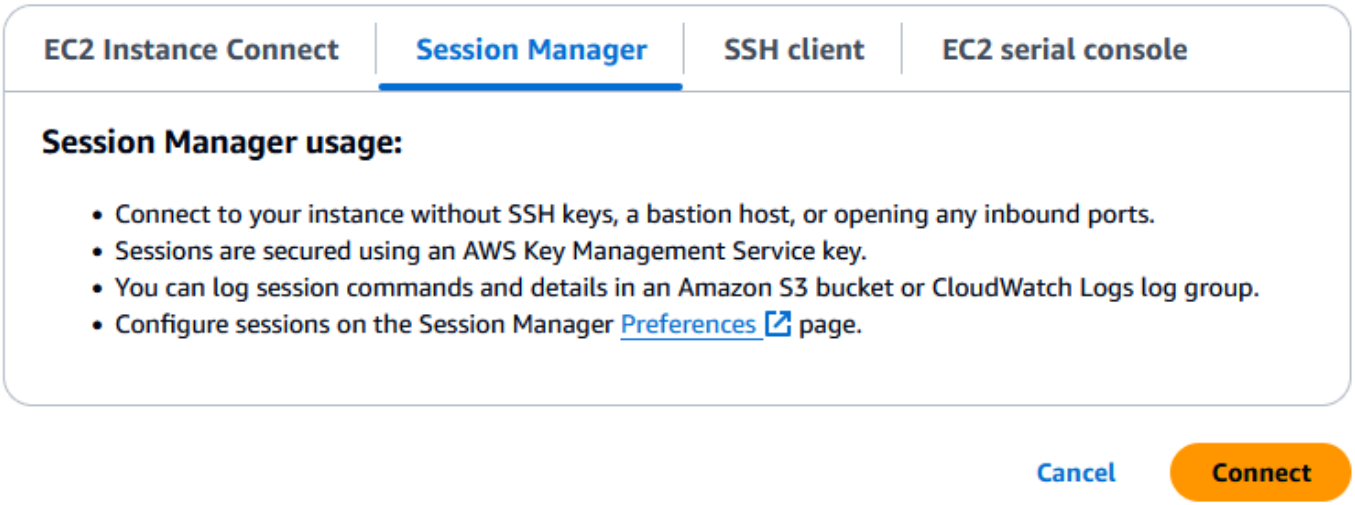
前提条件

Session Manager を使用してインスタンスへの接続を試みる前に、必要なセットアップ手順を完了しておく必要があります。例えば、SSM でインスタンスを管理し、[AmazonSSMManagedInstanceCore] ポリシーを使用してインスタンスを IAM ロールにアタッチしておく必要があります。詳細については、「Session Manager のセットアップ」を参照してください。

Amazon EC2 コンソールで Session Manager を使用して Amazon EC2 インスタンスに接続するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択し、[接続] を選択してください。
4. 接続方法では、[Session Manager] を選択します。

5. [接続] を選択してセッションを開始します。



トラブルシューティング

1 つ以上の Systems Manager アクション (`ssm:command-name`) の実行は承認されていないというエラーが表示された場合は、Amazon EC2 コンソールからセッションを開始できるようにポリシーを更新する必要があります。詳細と手順については、「AWS Systems Manager ユーザーガイドユーザーガイド」の「Session Manager のデフォルトの IAM ポリシーのクイックスタート」を参照してください。

パブリック IP アドレスと EC2 Instance Connect を使用して Linux インスタンスに接続する

Amazon EC2 Instance Connect を使用すると、Secure Shell (SSH) 経由で Linux インスタンスに安全に接続できます。EC2 Instance Connect では、AWS Identity and Access Management (IAM) ポリシーおよびプリンシパルを使用して、インスタンスへの SSH によるアクセスをコントロールします。SSH キーを共有および管理する必要はありません。EC2 Instance Connect を使用したすべての接続リクエストは、AWS CloudTrail に口グとして記録されるため、接続リクエストを監査できます。

EC2 Instance Connect では、Amazon EC2 コンソールまたは任意の SSH クライアントを使用してインスタンスに接続できます。

EC2 Instance Connect を使用してインスタンスに接続すると、EC2 Instance Connect API から SSH パブリックキーがインスタンスメタデータにプッシュされ、60 秒間保持されます。ユーザーにアタッチされた IAM ポリシーにより、ユーザーはパブリックキーをインスタンスメタデータにプッ

シュすることを許可されます。SSH デーモンは、インスタンスメタデータから承認用のパブリックキーを検索してユーザーをインスタンスに接続するために、EC2 Instance Connect のインストール時に設定される `AuthorizedKeysCommand` と `AuthorizedKeysCommandUser` を使用します。

> **Tip**
>
> EC2 Instance Connect は Linux インスタンスに接続するためのオプションの 1 つです。他のオプションについては、「SSH を使用した Linux インスタンスへの接続」を参照してください。Windows インスタンスに接続するには、「RDP を使用した Windows インスタンスへの接続」を参照してください。

料金

EC2 Instance Connect は、追加のコストなしで利用できます。

リージョンの可用性

EC2 Instance Connect は、すべての AWS リージョンで利用できます。Local Zones ではサポートされていません。

内容

- チュートリアル: EC2 Instance Connect を使用してインスタンスに接続するために必要な設定を完了する
- EC2 Instance Connect の前提条件
- IAM への EC2 Instance Connect のアクセス許可の付与
- EC2 インスタンスでの EC2 Instance Connect のインストール
- EC2 Instance Connect を使用して Linux インスタンスに接続する
- EC2 Instance Connect のアンインストール

EC2 Instance Connect を使用して拠点ホストのセキュリティを強化する方法に関して説明しているブログ投稿については、「Amazon EC2 Instance Connect を使用した拠点ホストの保護」を参照してください。

チュートリアル: EC2 Instance Connect を使用してインスタンスに接続するために必要な設定を完了する

Amazon EC2 コンソールで EC2 Instance Connect を使用してインスタンスに接続するには、まず、インスタンスに正常に接続するための前提条件の設定を完了する必要があります。このチュートリアルの目的は、前提条件となる設定を完了するためのタスクを順を追って説明することです。

チュートリアルの概要

このチュートリアルでは、以下の 4 タスクを完了します。

- タスク 1: EC2 Instance Connect を使用するために必要なアクセス許可を付与する

  まず、パブリックキーをインスタンスメタデータにプッシュできる IAM アクセス許可を含む IAM ポリシーを作成します。IAM ID (ユーザー、ユーザーグループ、またはロール) にこのポリシーをアタッチして、IAM ID がこれらのアクセス許可を取得できるようにします。
- タスク 2: EC2 Instance Connect サービスからインスタンスへのインバウンドトラフィックを許可する

  次に、EC2 Instance Connect サービスからインスタンスへのトラフィックを許可するセキュリティグループを作成します。これは、Amazon EC2 コンソールで EC2 Instance Connect を使用してインスタンスに接続するときに必要です。
- タスク 3: インスタンスを起動する

  次に、EC2 Instance Connect にあらかじめインストールされている AMI を使用して EC2 インスタンスを起動し、前のステップで作成したセキュリティグループを追加します。
- タスク 4: インスタンスに接続する

  最後に、Amazon EC2 コンソールで EC2 Instance Connect を使用してインスタンスに接続します。接続できたら、タスク 1、2、3 で完了した前提条件の設定が成功したことを確認できます。

タスク 1: EC2 Instance Connect を使用するために必要なアクセス許可を付与する

EC2 Instance Connect を使用してインスタンスに接続すると、EC2 Instance Connect API から SSH パブリックキーがインスタンスメタデータにプッシュされ、60 秒間保持されます。パブリックキーをインスタンスメタデータにプッシュするために必要なアクセス許可を付与するには、IAM ID (ユーザー、ユーザーグループ、またはロール) にアタッチされた IAM ポリシーが必要です。

タスクの目標

パブリックキーをインスタンスにプッシュするアクセス許可を付与する IAM ポリシーを作成します。許可する具体的なアクションは `ec2-instance-connect:SendSSHPublicKey` です。Amazon EC2 コンソールでインスタンスを表示して選択できるように、`ec2:DescribeInstances` アクションを許可する必要もあります。

ポリシーを作成したら、そのポリシーを IAM ID (ユーザー、ユーザーグループ、またはロール) にアタッチして、IAM ID にアクセス許可が付与されるようにします。

以下のように構成されたポリシーを作成します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "ec2-instance-connect:SendSSHPublicKey",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeInstances",
            "Resource": "*"
        }
    ]
}
```

Important

このチュートリアルで作成した IAM ポリシーは非常に強力なアクセス許可を持つポリシーで、任意の AMI ユーザー名を使用して任意のインスタンスに接続できます。チュートリアルをシンプルにし、このチュートリアルで説明する特定の設定に焦点を当てるために、この非常に強力なアクセス許可を持つポリシーを使用しています。ただし、本番環境では、最小権限のアクセス許可を付与するように IAM ポリシーを設定することをお勧めします。IAM ポリシーの例は IAM への EC2 Instance Connect のアクセス許可の付与 を参照してください。

EC2 Instance Connect を使用してインスタンスに接続することを許可する IAM ポリシーを作成してアタッチするには

1. まず、IAM ポリシーを作成します

   a. IAM コンソールの https://console.aws.amazon.com/iam/ を開いてください。

   b. ナビゲーションペインで、ポリシー を選択してください。

   c. [ポリシーの作成] を選択してください。

   d. [アクセス許可の指定] ページで、以下の操作を実行します。

      i. [サービス] で、[EC2 Instance Connect] を選択してください。

      ii. [許可されたアクション] の検索フィールドに **send** を入力すると関連するアクションが表示されるので、[SendSSHPublicKey] を選択してください。

      iii. [リソース] で、[すべて] を選択してください。本番環境では、ARN でインスタンスを指定することをお勧めしますが、このチュートリアルではすべてのインスタンスを許可します。

      iv. [アクセス許可の追加] を選択してください。

      v. サービスで EC2 を選択してください。

      vi. [許可されたアクション] の検索フィールドに **describein** を入力すると関連するアクションが表示されるので、[DescribeInstances] を選択してください。

      vii. [Next] を選択してください。

   e. [レビューと作成] ページで、以下の操作を実行します。

      i. [Policy Name] にこのポリシーの名前を入力してください。

      ii. [Create policy] を選択してください。

2. 次に、ポリシーを自分の ID にアタッチします。

   a. IAM コンソールのナビゲーションペインから、[ポリシー] を選択してください。

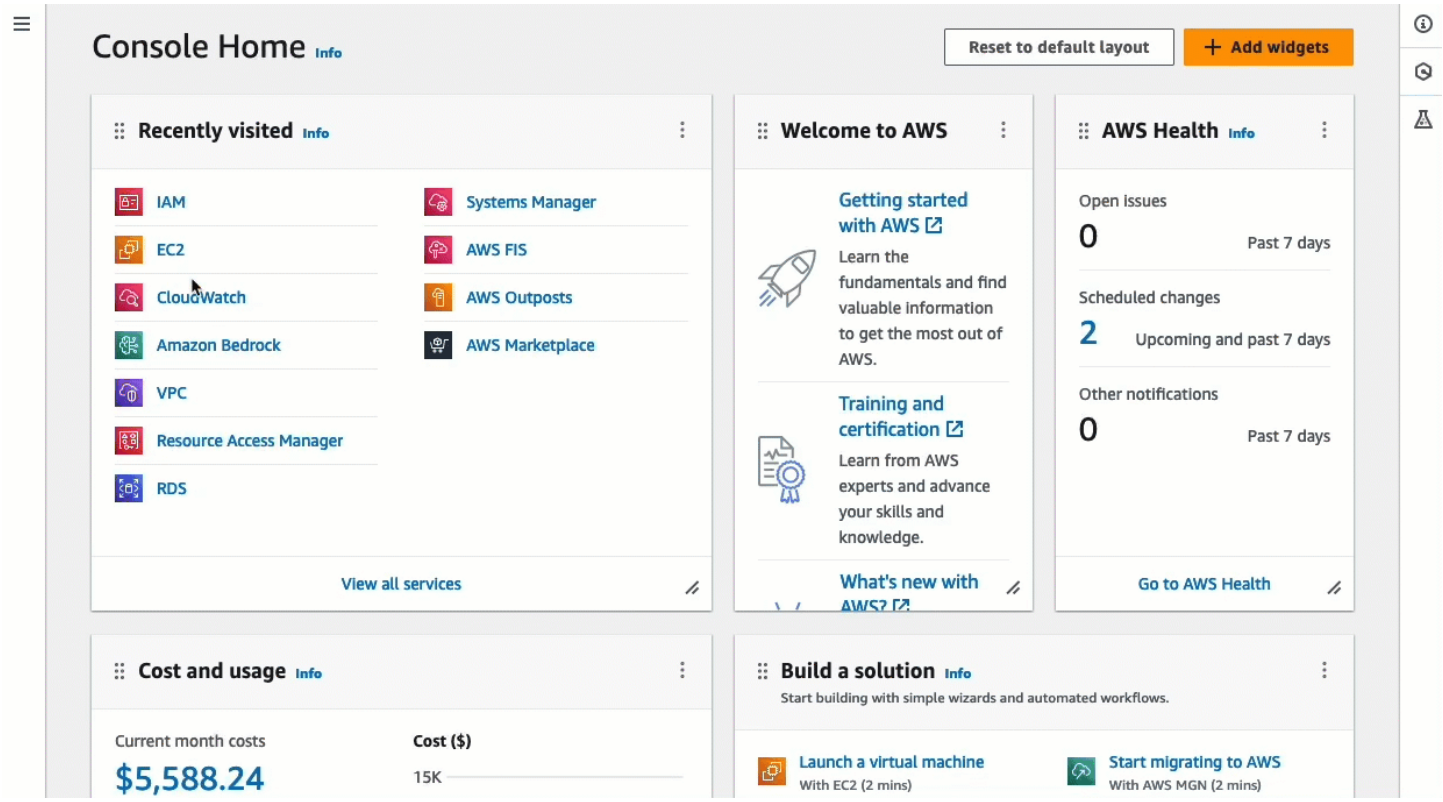
   b. ポリシーのリストで、作成したポリシーの名前の横にあるラジオボックスをオンにします。検索ボックスを使用して、ポリシーのリストをフィルタリングできます。

   c. [アクション]、[アタッチ] の順にクリックしてください。

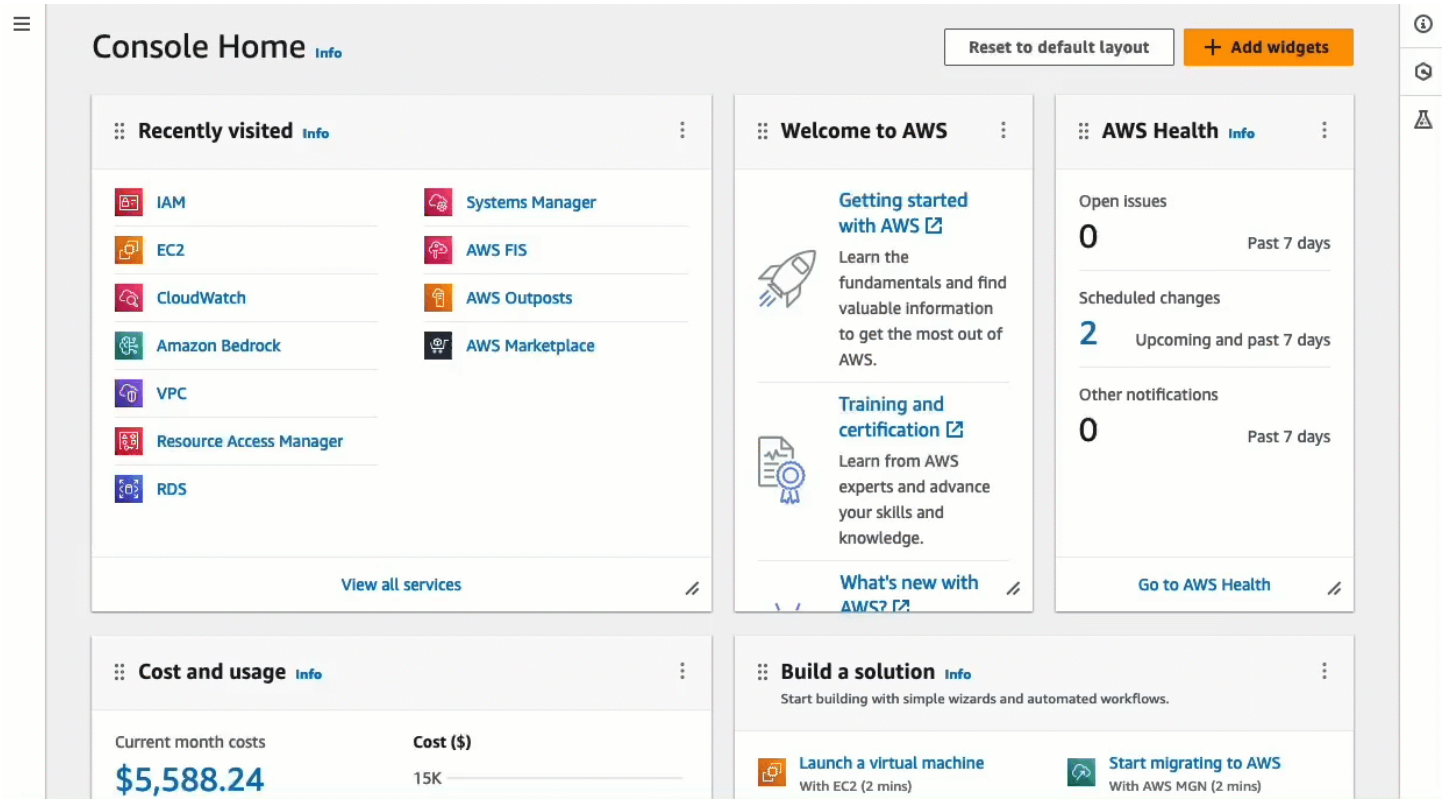
   d. [IAM エンティティ] で、ID (ユーザー、ユーザーグループ、ロール) の横にあるチェックボックスをオンにします。検索ボックスを使用して、エンティティのリストをフィルタリングできます。

   e. [Attach policy] (ポリシーのアタッチ) を選択してください。

アニメーションを表示: IAM ポリシーの作成



アニメーションを表示: IAM ポリシーのアタッチ



タスク 2: EC2 Instance Connect サービスからインスタンスへのインバウンドトラフィックを許可する

Amazon EC2 コンソールで EC2 Instance Connect を使用してインスタンスに接続する場合、インスタンスに到達できるようにする必要があるトラフィックは、EC2 Instance Connect サービスからのトラフィックです。これはローカルコンピュータからインスタンスへの接続とは異なります。その場合は、ローカルコンピュータからインスタンスへのトラフィックを許可する必要があります。EC2 Instance Connect サービスからのトラフィックを許可するには、EC2 Instance Connect サービスの IP アドレス範囲からのインバウンド SSH トラフィックを許可するセキュリティグループを作成する必要があります。

AWS はプレフィックスリストを使用して IP アドレス範囲を管理します。EC2 Instance Connect プレフィックスリストの名前は次のとおりです。*#####* はリージョンコードに置き換えられます。

- IPv4 プレフィックスリスト名: `com.amazonaws.region.ec2-instance-connect`
- IPv6 プレフィックスリスト名: `com.amazonaws.region.ipv6.ec2-instance-connect`

タスクの目標

インスタンスが配置されているリージョンの IPv4 プレフィックスリストからポート 22 のインバウンド SSH トラフィックを許可するセキュリティグループを作成します。

EC2 Instance Connect サービスからインスタンスへのインバウンドトラフィックを許可するセキュリティグループを作成する方法

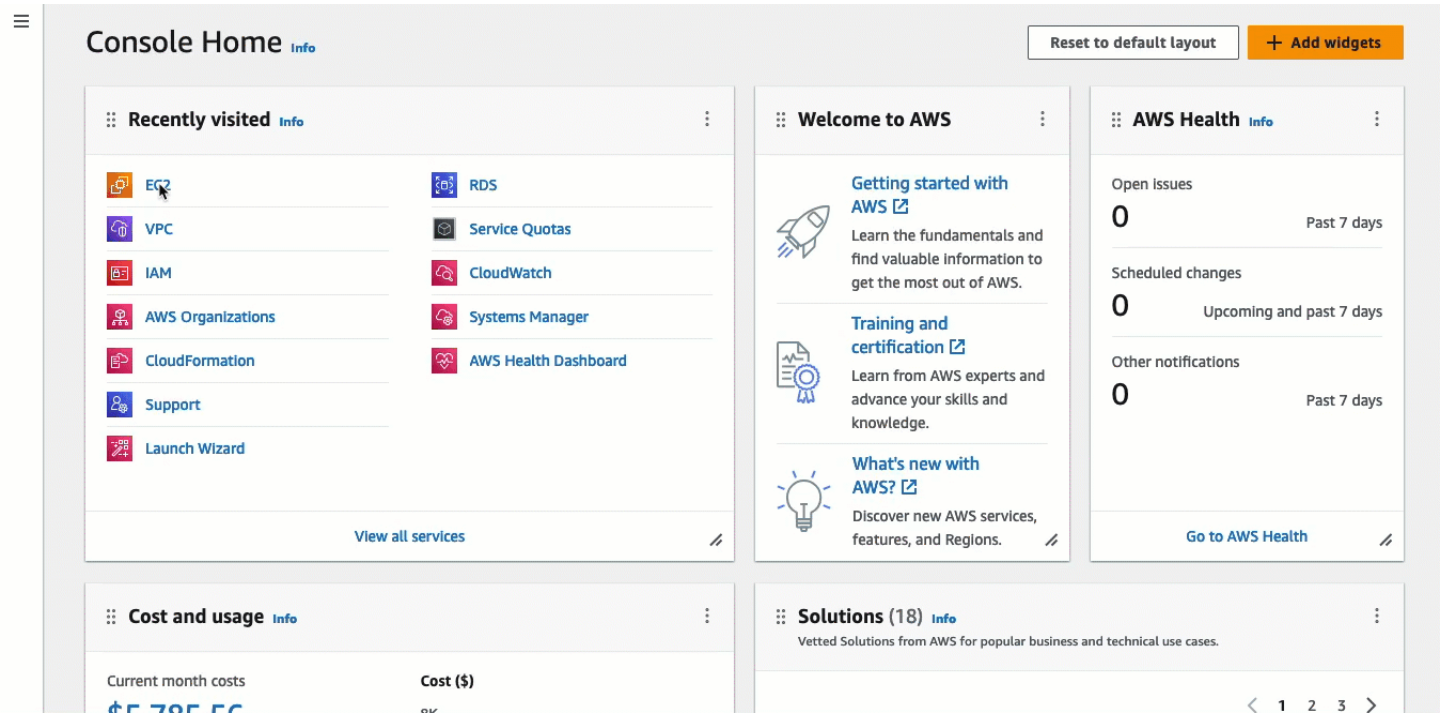
1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Security Groups] を選択してください。
3. [セキュリティグループの作成] を選択してください。
4. [基本的な詳細] で、次の操作を行います。
   a. [セキュリティグループ名] に、セキュリティグループのわかりやすい名前を入力してください。
   b. [説明] に、セキュリティグループのわかりやすい説明を入力してください。
5. [インバウンドルール] で、次の操作を行います。
   a. [ルールを追加] を選択してください。
   b. [タイプ] で SSH] を選択してください。
   c. [ソース] は [カスタム] のままにします。

d. [ソース] の横にあるフィールドで、EC2 Instance Connect のプレフィックスリストを選択してください。

   例えば、インスタンスが米国東部 (バージニア北部) us-east-1リージョンにあり、ユーザーがパブリック IPv4 アドレスに接続する場合は、[com.amazonaws.us-east-1.ec2-instance-connect] というプレフィックスリストを選択してください。

6. [セキュリティグループの作成] を選択してください。

アニメーションを表示: セキュリティグループを作成する



タスク 3: インスタンスを起動する

インスタンスを起動するときは、インスタンスの起動に必要な情報を含む AMI を指定する必要があります。EC2 Instance Connect があらかじめインストールされているかどうかにかかわらず、インスタンスの起動を選択できます。このタスクでは、EC2 Instance Connect にあらかじめインストールされている AMI を指定します。

EC2 Instance Connect のプリインストール無しでインスタンスを起動し、EC2 Instance Connect を使用してインスタンスに接続する場合は、追加の設定手順を実行する必要があります。これらの手順は、このチュートリアルの範囲外です。

タスクの目標

EC2 Instance Connect がプリインストールされている Amazon Linux 2023 AMI を使用してインスタンスを起動します。Amazon EC2 コンソールで EC2 Instance Connect を使用してインスタンスに接続できるように、前に作成したセキュリティグループも指定します。EC2 Instance Connect を使用してインスタンスに接続することで、パブリックキーをインスタンスのメタデータにプッシュするため、インスタンスを起動するときに SSH キーを指定する必要はありません。

Amazon EC2 コンソールで EC2 Instance Connect を使用できるインスタンスを起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 画面の上のナビゲーションバーに、現在の AWS リージョンが表示されます (アイルランドなど)。インスタンスを起動するリージョンを選択してください。特定のリージョンのトラフィックを許可するセキュリティグループを作成したため、インスタンスを起動するのと同じリージョンを選択する必要があり、この選択が重要となります。
3. Amazon EC2 コンソールダッシュボードで、[インスタンスを起動] を選択してください。
4. (オプション) [Names and tags](名前とタグ) における [Name] (名前) では、インスタンス用にわかりやすい名前を入力してください。
5. [アプリケーションと OS イメージ (Amazon マシンイメージ)] で、[クイックスタート] を選択してください。[Amazon Linux] がデフォルトで選択されています。[Amazon マシンイメージ (AMI)] ではデフォルトで、[Amazon Linux 2023 AMI] が選択されています。このタスクは既定の選択のままにします。
6. [インスタンスタイプ]で、[インスタンスタイプ] のデフォルトの選択状態のままにするか、別のインスタンスタイプを選択してください。
7. [キーペア (ログイン)] の [キーペア名] で、[キーペアなしで続行 (非推奨)] を選択してください。EC2 Instance Connect を使用してインスタンスに接続すると、EC2 Instance Connect はキーペアをインスタンスのメタデータにプッシュします。接続に使用されるのはこのキーペアです。
8. [Network settings] (ネットワーク設定) で、次の操作を行います：

   a. [自動割り当てパブリック IP] は、[有効] のままにします。

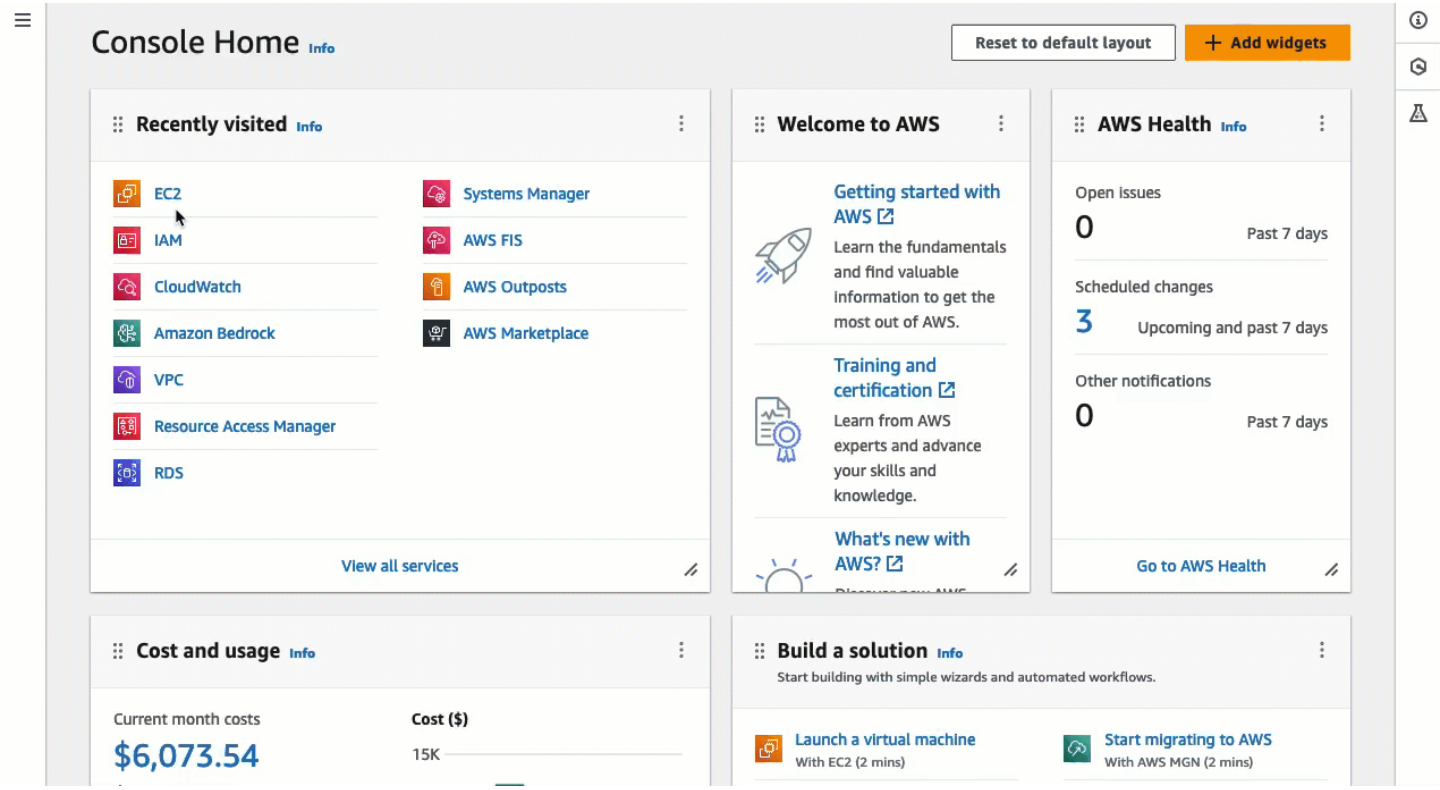
   > **Note**
   >
   > Amazon EC2 コンソールで EC2 Instance Connect を使用してインスタンスに接続するには、そのインスタンスにパブリック IPv4 または IPv6 アドレスが必要です。

   b. [ファイアウォール (セキュリティグループ)] で、[既存のセキュリティグループを選択する] を選択してください。

c. [共通セキュリティグループ] で、先ほど作成したセキュリティグループを選択してください。

9. [Summary] (サマリー) パネルで、[Launch instance] (インスタンスの起動) を選択してください。

アニメーションを表示: インスタンスを起動する



タスク 4: インスタンスに接続する

EC2 Instance Connect を使用してインスタンスに接続すると、EC2 Instance Connect API から SSH パブリックキーがインスタンスメタデータにプッシュされ、60 秒間保持されます。SSH デーモンは、`AuthorizedKeysCommand` および `AuthorizedKeysCommandUser` を使用して、インスタンスメタデータからパブリックキーを見つけて認証を行い、ユーザーをインスタンスに接続します。

タスクの目標

このタスクでは、Amazon EC2 コンソールで EC2 Instance Connect を使用してインスタンスに接続します。前提条件となるタスク 1、2、3 を完了していれば、接続は成功するはずです。

インスタンスに接続する手順

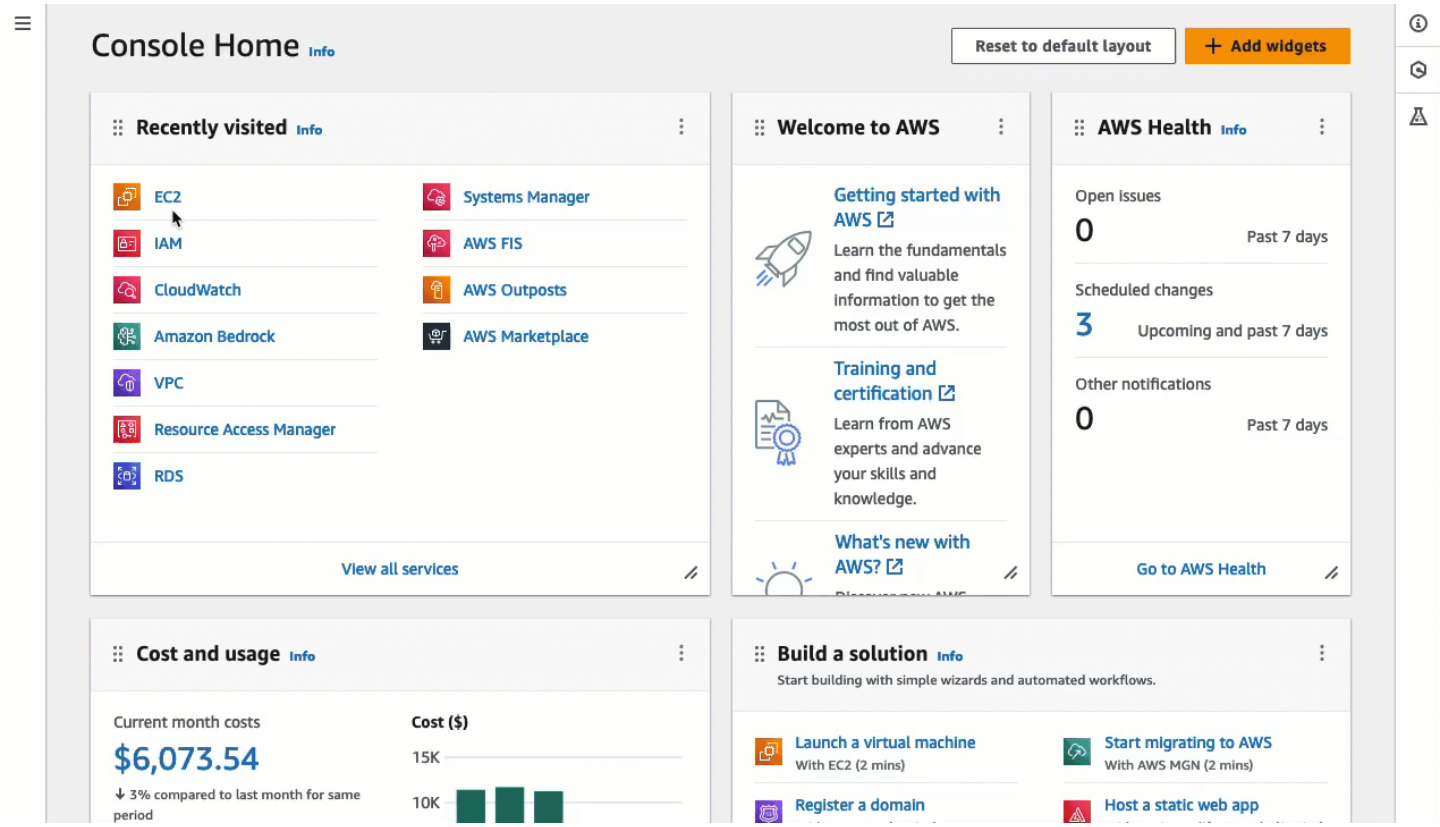
インスタンスに接続するには、次の手順に従ってください。手順のアニメーションを見る場合は、「アニメーションを表示: インスタンスに接続する」を参照してください。

Amazon EC2 コンソールで EC2 Instance Connect を使用してインスタンスを起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 画面の上のナビゲーションバーに、現在の AWS リージョンが表示されます (アイルランドなど)。インスタンスが存在するリージョンを選択してください。
3. ナビゲーションペインで、[インスタンス] を選択してください。
4. インスタンスを選択し、[接続] を選択してください。
5. [EC2 Instance Connect] タブを選択してください。
6. [パブリック IP を使用して接続] を選択します。
7. [接続]を選択してください。

   ブラウザでターミナルウィンドウが開き、インスタンスに接続されます。

アニメーションを表示: インスタンスに接続する



EC2 Instance Connect の前提条件

EC2 Instance Connect をインストールして使用するための前提条件は次のとおりです。

- EC2 Instance Connect のインストール
- ネットワーク接続を確立する
- EC2 Instance Connect プロキシへのアウトバウンドトラフィックを許可する
- インバウンド SSH トラフィックを許可する
- 許可を付与する
- 使用するローカルコンピュータに SSH クライアントをインストールする
- ユーザー名の要件を満たす

EC2 Instance Connect のインストール

EC2 Instance Connect を使用してインスタンスに接続するには、インスタンスに EC2 Instance Connect がインストールされている必要があります。EC2 Instance Connect にプリインストールされている AMI を使用してインスタンスを起動することも、サポートされている AMI で起動されたインスタンスに EC2 Instance Connect をインストールすることもできます。詳細については「EC2 インスタンスでの EC2 Instance Connect のインストール」を参照してください。

ネットワーク接続を確立する

インスタンスは、ユーザーがインターネットまたはインスタンスのプライベート IP アドレスを介してインスタンスに接続できるように設定できます。ユーザーが EC2 Instance Connect を使用してインスタンスに接続する方法によって、次のネットワークアクセスを設定する必要があります。

- ユーザーがインターネット経由でインスタンスに接続する場合、インスタンスはパブリック IPv4 または IPv6 アドレスを持ち、インターネットへのルートが設定されたパブリックサブネット内に配置されている必要があります。デフォルトのパブリックサブネットを変更していない場合、IPv4 専用のインターネットへのルートが含まれており、IPv6用のルートは含まれていません。詳細については、「Amazon VPC ユーザーガイド」の「インターネットゲートウェイを使用して VPC インターネットアクセスを有効にする」を参照してください。
- ユーザーがインスタンスのプライベート IP アドレスを介してインスタンスに接続する場合は、AWS Direct Connect、AWS Site-to-Site VPN、または VPC ピアリングを使用して VPC へのプライベートネットワーク接続を確立し、ユーザーがインスタンスのプライベート IPv4 アドレスに到達できるようにする必要があります。

インスタンスにパブリック IPv4 または IPv6 アドレスがなく、上記のようにネットワークアクセスを設定しない場合は、EC2 Instance Connect エンドポイントを EC2 Instance Connect の代わりとして検討できます。EC2 Instance Connect エンドポイントを使用すると、インスタンスにパブリック IPv4 または IPv6 アドレスがない場合でも、SSH または RDP を使用してインスタンスに接続できます。詳細については、「Amazon EC2 コンソールを使用した Linux インスタンスへの接続」を参照してください。

EC2 Instance Connect プロキシへのアウトバウンドトラフィックを許可する

Amazon EC2 コンソールを使用してインスタンスに接続すると、ブラウザはリージョン固有のエンドポイントで EC2 Instance Connect プロキシサービスへの WebSocket 接続をポート 443 で確立します。

エンドポイントの形式はパーティションによって異なります。

- 商用 AWS リージョンと AWS GovCloud (米国) リージョンは、エンドポイント形式を使用します (`prod.region.oneclickv2-proxy.ec2.aws.dev`)。
- 中国リージョンはエンドポイント形式 を使用します (`prod.region.oneclickv2-proxy.ec2.a2z.org.cn`)。

次の例では、各パーティションのエンドポイントが示されています。

- 米国東部 (バージニア北部) リージョンのエンドポイントは `prod.us-east-1.oneclickv2-proxy.ec2.aws.dev` です。
- 中国 (北京) リージョンのエンドポイントは `prod.cn-north-1.oneclickv2-proxy.ec2.a2z.org.cn` です。

> **Note**
>
> ブラウザはこのエンドポイントに到達する必要があります。ファイアウォールまたは VPN が接続をブロックした場合、接続は失敗します。

インバウンド SSH トラフィックを許可する

Amazon EC2 コンソールを使用してインスタンスに接続するには

ユーザーが Amazon EC2 コンソールでインスタンスに接続する場合、インスタンスに到達できるようにする必要があるトラフィックは、EC2 Instance Connect サービスからのトラフィックです。サービスは、プレフィックスリストを通じて AWS を管理する特定の IP アドレス範囲によって識別されます。EC2 Instance Connect サービスからインバウンド SSH トラフィックを許可するセキュリティグループを作成する必要があります。これを設定するには、インバウンドルールの [ソース] の横にあるフィールドで、EC2 Instance Connect プレフィックスリストを選択してください。

AWS は、各リージョンの IPv4 アドレスと IPv6 アドレスの異なる管理されたプレフィックスリストを提供します。EC2 Instance Connect プレフィックスリストの名前は次のとおりです。*#####* はリージョンコードに置き換えられます。

- IPv4 プレフィックスリスト名: `com.amazonaws.region.ec2-instance-connect`
- IPv6 プレフィックスリスト名: `com.amazonaws.region.ipv6.ec2-instance-connect`

セキュリティグループの作成方法については、「タスク 2: EC2 Instance Connect サービスからインスタンスへのインバウンドトラフィックを許可する」を参照してください。詳細については、「Amazon VPC ユーザーガイド」の「AWS 管理されたプレフィックスリストの提供」を参照してください。

CLI または SSH を使用してインスタンスに接続するには

インスタンスに関連付けられているセキュリティグループで、IP アドレスまたはネットワークからのインバウンド SSH トラフィックがポート 22 で許可されることを確認します。VPC のデフォルトのセキュリティグループでは、着信 SSH トラフィックはデフォルトでは許可されません。インスタンス起動ウィザードで作成されたセキュリティグループは、デフォルトで受信 SSH トラフィックを許可します。詳細については「コンピュータからのインスタンスへの接続ルール」を参照してください。

許可を付与する

EC2 Instance Connect を使用してインスタンスに接続するすべての IAM ユーザーに、必要なアクセス許可を付与する必要があります。詳細については「IAM への EC2 Instance Connect のアクセス許可の付与」を参照してください。

使用するローカルコンピュータに SSH クライアントをインストールする

ユーザーが SSH を使用して接続する場合、ローカルコンピュータに SSH クライアントがあることを確認する必要があります。

ほとんどの場合、ユーザーのローカルコンピュータにはデフォルトで SSH クライアントがインストールされています。SSH クライアントがあるかどうかを確認するには、コマンドラインで ssh と入力してください。使用するローカルコンピュータでこのコマンドが認識されない場合、SSH クライアントをインストールできます。Linux または macOS X に SSH クライアントをインストールする詳細については、http://www.openssh.comを参照してください。Windows 10 に SSH クライアントをインストールする詳細については、Windows の OpenSSHを参照してください。

ユーザーが Amazon EC2 コンソールのみを使用してインスタンスに接続する場合、ローカルコンピュータに SSH クライアントをインストールする必要はありません。

ユーザー名の要件を満たす

EC2 Instance Connect を使用してインスタンスに接続する場合、ユーザー名は次の要件を満たす必要があります。

- 1 字目: アルファベット (A-Z、a-z)、数字 (0-9)、下線 _のいずれか。
- 後続には、アルファベット (A-Z、a-z)、数字 (0-9)、または以下の文字を使用できます: @ . _ -
- 最小長: 1 文字
- 最大長: 31 文字

IAM への EC2 Instance Connect のアクセス許可の付与

EC2 Instance Connect を使用してインスタンスに接続するには、以下のアクションと条件に対するアクセス許可をユーザーに付与する IAM ポリシーを作成する必要があります。

- ec2-instance-connect:SendSSHPublicKey アクション – パブリックキーをインスタンスにプッシュするためのアクセス許可を付与します。
- ec2:osuser 条件 – パブリックキーをインスタンスにプッシュできる OS ユーザーの名前を指定します。インスタンスの起動に使用した AMI のデフォルトのユーザー名を使用します。デフォルトのユーザー名は AL2023 と Amazon Linux 2 では ec2-user、Ubuntu では ubuntu です。
- ec2:DescribeInstances アクション — ラッパーがこのアクションを呼び出すため、EC2 コンソールを使用するときに必要です。ユーザーには、別のポリシーからこのアクションを呼び出すためのアクセス許可が既に付与されている場合があります。
- ec2:DescribeVpcs アクション – IPv6 アドレスに接続するときに必要です。

特定の EC2 インスタンスへのアクセスを制限することを検討してください。それ以外の場合、ec2-instance-connect:SendSSHPublicKey アクションのアクセス許可を持つすべての IAM プリンシパルは、すべての EC2 インスタンスに接続できます。リソース ARN を指定するか、リソースタグを条件キーとして使用して、アクセスを制限することができます。

詳細については、Amazon EC2 Instance Connect のアクション、リソース、および条件キー を参照してください。

IAM ポリシーの作成について詳しくは、「IAM ユーザーガイド」の「IAM ポリシーの作成」を参照してください。

ユーザーに特定のインスタンスへの接続を許可

次の IAM ポリシーは、リソース ARN で識別される特定のインスタンスに接続するアクセス許可を付与します。

次の IAM ポリシーの例では、以下のアクションと条件が指定されています。

- ec2-instance-connect:SendSSHPublicKey アクションは、リソース ARN で指定された 2 つのインスタンスに接続するためのアクセス許可をユーザーに付与します。すべての EC2 インスタンスに接続するためのアクセス許可をユーザーに付与するには、リソース ARN を * (ワイルドカード) に置き換えます。
- ec2:osuser 条件により、接続時に *ami-username* が指定されている場合にのみ、インスタンスに接続するためのアクセス許可が付与されます。
- ec2:DescribeInstances アクションは、コンソールを使用してインスタンスに接続するユーザーにアクセス許可を付与するように指定されます。ユーザーが SSH クライアントのみを使用してインスタンスに接続する場合は、ec2:DescribeInstances を省略できます。ec2:Describe* API アクションはリソースレベルのアクセス許可をサポートしないことに注意してください。したがって、Resource の要素には、* (ワイルドカード) が必要です。
- ec2:DescribeVpcs アクションは、IPv6 アドレスでインスタンスに接続するために、コンソールを使用してユーザーにアクセス許可を付与するように指定されます。ユーザーがパブリック IPv4 アドレスのみを使用する場合は、ec2:DescribeVpcs を省略できます。ec2:Describe* API アクションはリソースレベルのアクセス許可をサポートしないことに注意してください。したがって、Resource の要素には、* (ワイルドカード) が必要です。

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "ec2-instance-connect:SendSSHPublicKey",
            "Resource": [
                "arn:aws:ec2:us-
east-1:111122223333:instance/i-1234567890abcdef0",
                "arn:aws:ec2:us-east-1:111122223333:instance/i-0598c7d356eba48d7"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:osuser": "ami-username"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeVpcs"
            ],
            "Resource": "*"
        }
    ]
}
```

ユーザーに特定のタグを持つインスタンスへの接続を許可

属性ベースのアクセス制御 (ABAC) は、ユーザーおよび AWS リソースにアタッチできるタグに基づいてアクセス許可を定義する認可戦略です。リソースタグを使用してインスタンスへのアクセスを制御することもできます。AWS リソースへのアクセスを制御するタグの使用の詳細については、「IAM ユーザーガイド」の「AWS リソースへのアクセス制御」を参照してください。

次の IAM ポリシーの例では、`ec2-instance-connect:SendSSHPublicKey` アクションは、インスタンスに key=tag-key と value=tag-value が付いたリソースタグが割り当てられていることを条件に、(リソース ARN の * (ワイルドカード) で示される) 任意のインスタンスに接続するためのアクセス許可をユーザーに付与します。

`ec2:DescribeInstances` アクションは、コンソールを使用してインスタンスに接続するユーザーにアクセス許可を付与するように指定されます。ユーザーが SSH クライアントのみを使用してインスタンスに接続する場合は、`ec2:DescribeInstances` を省略できます。`ec2:Describe*` API

アクションはリソースレベルのアクセス許可をサポートしないことに注意してください。したがって、Resource の要素には、* (ワイルドカード) が必要です。

ec2:DescribeVpcs アクションは、IPv6 アドレスでインスタンスに接続するために、コンソールを使用してユーザーにアクセス許可を付与するように指定されます。ユーザーがパブリック IPv4 アドレスのみを使用する場合は、ec2:DescribeVpcs を省略できます。ec2:Describe* API アクションはリソースレベルのアクセス許可をサポートしないことに注意してください。したがって、Resource の要素には、* (ワイルドカード) が必要です。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "ec2-instance-connect:SendSSHPublicKey",
            "Resource": "arn:aws:ec2:us-east-1:111122223333:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/tag-key": "tag-value"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeVpcs"
            ],
            "Resource": "*"
        }
    ]
}
```

EC2 インスタンスでの EC2 Instance Connect のインストール

EC2 Instance Connect を使用して Linux インスタンスに接続するには、EC2 Instance Connect がインストールされている必要があります。EC2 Instance Connect をインストールすると、インスタンスに SSH デーモンが設定されます。

EC2 Instance Connect パッケージの詳細については、GitHub ウェブサイトの「aws/aws-ec2-instance-connect-config」を参照してください。

> **Note**
>
> SSH 認証の `AuthorizedKeysCommand` および `AuthorizedKeysCommandUser` 設定を構成した場合、EC2 Instance Connect をインストールしても更新されません。そのため、EC2 Instance Connect は使用できません。

インストール条件

EC2 Instance Connect をインストールする前に、以下の前提条件を満たしていることを確認してください。

- インスタンスが以下のいずれかを使用していることを確認します。
  - バージョン 2.0.20190618 より前の Amazon Linux 2 *
  - AL2023 最小 AMI または Amazon ECS 最適化 AMI
  - CentOS Stream 8 および 9
  - 14.2.1 より前の MacOS Sonoma、13.6.3 より前の Ventura、12.7.2 より前の Monterey *
  - Red Hat Enterprise Linux (RHEL) 8 および 9
  - Ubuntu 16.04 および 18.04

  > **Tip**
  >
  > * Amazon Linux 2、MacOS、Ubuntu の場合: 上記のバージョンよりも新しいバージョンを使用してインスタンスを起動した場合、EC2 Instance Connect はプリインストールされており、手動インストールは不要です。

- EC2 Instance Connect の一般的な前提条件を確認します。

  詳細については「EC2 Instance Connect の前提条件」を参照してください。
- ローカルマシン上の SSH クライアントを使用してインスタンスに接続するための前提条件を確認します。

  詳細については「SSH を使用した Linux インスタンスへの接続」を参照してください。
- インスタンスの ID を取得します。

自分のインスタンスの ID は、Amazon EC2 コンソールを使用して ([インスタンス ID] 列から) 取得できます。その代わりに、describe-instances AWS CLIまたは Get-EC2Instance AWS Tools for Windows PowerShellコマンドを使用することもできます。

EC2 Instance Connect を手動でインストールする

> **Note**
>
> 以下のいずれかの AMI を使用してインスタンスを起動した場合は、EC2 Instance Connect がプリインストールされているため、このステップをスキップできます。
>
> - AL2023 標準 AMI
> - Amazon Linux 2 2.0.20190618 以降
> - macOS Sonoma 14.2.1 以降
> - macOS Ventura 13.6.3 以降
> - macOS Monterey 12.7.2 以降
> - Ubuntu 20.04 以降

インスタンスのオペレーティングシステムに応じて、以下のいずれかの手順を使用して EC2 Instance Connect をインストールします。

Amazon Linux 2

EC2 Instance Connect を Amazon Linux 2 で起動したインスタンスにインストールするには

1. SSH を使用してインスタンスに接続します。

   次のコマンド内のサンプル値を独自の値に置き換えます。インスタンスの起動時にインスタンスに割り当てた SSH キーペアと、インスタンスを起動するために使用した AMI のデフォルトのユーザー名を使用します。Amazon Linux 2 の場合、デフォルトのユーザー名は `ec2-user` です。

   ```
   $ ssh -i my_ec2_private_key.pem ec2-user@ec2-a-b-c-d.us-
   west-2.compute.amazonaws.com
   ```

インスタンスへの接続の詳細については[SSH クライアントを使用して Linux インスタンスに接続する]を参照してください。

2. インスタンスに EC2 Instance Connect パッケージをインストールします。

```
[ec2-user ~]$ sudo yum install ec2-instance-connect
```

3 つの新しいスクリプトが /opt/aws/bin/ フォルダに表示されます。

```
eic_curl_authorized_keys
eic_parse_authorized_keys
eic_run_authorized_keys
```

3. (オプション) EC2 Instance Connect がインスタンスに正常にインストールされたことを確認します。

```
[ec2-user ~]$ sudo less /etc/ssh/sshd_config
```

AuthorizedKeysCommand 行と AuthorizedKeysCommandUser 行に以下の値が含まれていれば、EC2 Instance Connect は正常にインストールされています。

```
AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_keys %u %f
AuthorizedKeysCommandUser ec2-instance-connect
```

- AuthorizedKeysCommand は、インスタンスメタデータからキーを探すように eic_run_authorized_keys スクリプトを設定します。
- AuthorizedKeysCommandUser は、システムユーザーを ec2-instance-connect として設定します。

> **Note**
>
> AuthorizedKeysCommand や AuthorizedKeysCommandUser を設定済みである場合は、EC2 Instance Connect をインストールしても値は変更されないため、EC2 Instance Connect は使用できません。

CentOS

EC2 Instance Connect を CentOS で起動したインスタンスにインストールするには

1. SSH を使用してインスタンスに接続します。

   次のコマンド内のサンプル値を独自の値に置き換えます。インスタンスの起動時にインスタンスに割り当てた SSH キーペアと、インスタンスを起動するために使用した AMI のデフォルトのユーザー名を使用します。CentOS の場合、デフォルトのユーザー名は `centos` または `ec2-user` です。

   ```
   $ ssh -i my_ec2_private_key.pem centos@ec2-a-b-c-d.us-
   west-2.compute.amazonaws.com
   ```

   インスタンスへの接続の詳細については[SSH クライアントを使用して Linux インスタンスに接続する]を参照してください。

2. HTTP または HTTPS プロキシを使用する場合は、現在のシェルセッションで `http_proxy` または `https_proxy` の環境変数を設定する必要があります。

   プロキシを使用していない場合は、この手順を省略できます。

   - HTTP プロキシサーバーの場合は、次のコマンドを実行します。

     ```
     $ export http_proxy=http://hostname:port
     $ export https_proxy=http://hostname:port
     ```

   - HTTPS プロキシサーバーの場合は、次のコマンドを実行します。

     ```
     $ export http_proxy=https://hostname:port
     $ export https_proxy=https://hostname:port
     ```

3. 次のコマンドを実行して、インスタンスに EC2 Instance Connect パッケージをインストールします。

   CentOS 用の EC2 Instance Connect 設定ファイルは Red Hat Package Manager (RPM) パッケージで提供され、CentOS 8 と CentOS 9 用の異なる RPM パッケージ、および Intel/AMD (x86_64) または ARM (AArch64) で実行されるインスタンスタイプ用の異なる RPM パッケージが含まれています。

オペレーティングシステムと CPU アーキテクチャに合ったコマンドブロックを使用してください。

- CentOS 8

  Intel/AMD (x86_64)

```
[ec2-user ~]$ mkdir /tmp/ec2-instance-connect
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-
west-2.s3.us-west-2.amazonaws.com/latest/linux_amd64/ec2-instance-
connect-2.0.0-5.rhel8.x86_64.rpm -o /tmp/ec2-instance-connect/ec2-instance-
connect.rpm
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_amd64/ec2-instance-connect-
selinux-2.0.0-5.noarch.rpm -o /tmp/ec2-instance-connect/ec2-instance-connect-
selinux.rpm
[ec2-user ~]$ sudo yum install -y /tmp/ec2-instance-connect/ec2-instance-
connect.rpm /tmp/ec2-instance-connect/ec2-instance-connect-selinux.rpm
```

  ARM (AArch64)

```
[ec2-user ~]$ mkdir /tmp/ec2-instance-connect
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-
west-2.s3.us-west-2.amazonaws.com/latest/linux_arm64/ec2-instance-
connect-2.0.0-5.rhel8.aarch64.rpm -o /tmp/ec2-instance-connect/ec2-instance-
connect.rpm
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_arm64/ec2-instance-connect-
selinux-2.0.0-5.noarch.rpm -o /tmp/ec2-instance-connect/ec2-instance-connect-
selinux.rpm
[ec2-user ~]$ sudo yum install -y /tmp/ec2-instance-connect/ec2-instance-
connect.rpm /tmp/ec2-instance-connect/ec2-instance-connect-selinux.rpm
```

- CentOS 9

  Intel/AMD (x86_64)

```
[ec2-user ~]$ mkdir /tmp/ec2-instance-connect
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-
west-2.s3.us-west-2.amazonaws.com/latest/linux_amd64/ec2-instance-
```

```
connect-2.0.0-5.rhel9.x86_64.rpm -o /tmp/ec2-instance-connect/ec2-instance-
connect.rpm
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_amd64/ec2-instance-connect-
selinux-2.0.0-5.noarch.rpm -o /tmp/ec2-instance-connect/ec2-instance-connect-
selinux.rpm
[ec2-user ~]$ sudo yum install -y /tmp/ec2-instance-connect/ec2-instance-
connect.rpm /tmp/ec2-instance-connect/ec2-instance-connect-selinux.rpm
```

ARM (AArch64)

```
[ec2-user ~]$ mkdir /tmp/ec2-instance-connect
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-
west-2.s3.us-west-2.amazonaws.com/latest/linux_arm64/ec2-instance-
connect-2.0.0-5.rhel9.aarch64.rpm -o /tmp/ec2-instance-connect/ec2-instance-
connect.rpm
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_arm64/ec2-instance-connect-
selinux-2.0.0-5.noarch.rpm -o /tmp/ec2-instance-connect/ec2-instance-connect-
selinux.rpm
[ec2-user ~]$ sudo yum install -y /tmp/ec2-instance-connect/ec2-instance-
connect.rpm /tmp/ec2-instance-connect/ec2-instance-connect-selinux.rpm
```

次の新しいスクリプトが /opt/aws/bin/ フォルダに表示されます。

```
eic_run_authorized_keys
```

4. (オプション) EC2 Instance Connect がインスタンスに正常にインストールされたことを確認します。

   - CentOS 8 の場合:

     ```
     [ec2-user ~]$ sudo less /lib/systemd/system/sshd.service.d/ec2-instance-
     connect.conf
     ```

   - CentOS 9 の場合:

     ```
     [ec2-user ~]$ sudo less /etc/ssh/sshd_config.d/60-ec2-instance-connect.conf
     ```

AuthorizedKeysCommand 行と AuthorizedKeysCommandUser 行に以下の値が含まれていれば、EC2 Instance Connect は正常にインストールされています。

```
AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_keys %u %f
AuthorizedKeysCommandUser ec2-instance-connect
```

- AuthorizedKeysCommand は、インスタンスメタデータからキーを探すように eic_run_authorized_keys スクリプトを設定します。
- AuthorizedKeysCommandUser は、システムユーザーを ec2-instance-connect として設定します。

> **Note**
>
> AuthorizedKeysCommand や AuthorizedKeysCommandUser を設定済みである場合は、EC2 Instance Connect をインストールしても値は変更されないため、EC2 Instance Connect は使用できません。

macOS

EC2 Instance Connect を macOS で起動したインスタンスにインストールするには

1. SSH を使用してインスタンスに接続します。

   次のコマンド内のサンプル値を独自の値に置き換えます。インスタンスの起動時にインスタンスに割り当てた SSH キーペアと、インスタンスを起動するために使用した AMI のデフォルトのユーザー名を使用します。macOS インスタンスの場合、デフォルトのユーザー名は ec2-user です。

   ```
   $ ssh -i my_ec2_private_key.pem ec2-user@ec2-a-b-c-d.us-
   west-2.compute.amazonaws.com
   ```

   インスタンスへの接続の詳細については[SSH クライアントを使用して Linux インスタンスに接続する]を参照してください。

2. 次のコマンドを使用して Homwbrew を更新します。このアップデートでは、Homebrew が認識しているソフトウェアが一覧表示されます。EC2 Instance Connect パッケージ

は、macOS インスタンスでは Homebrew 経由で提供されます。詳細については「Amazon EC2 Mac インスタンス上のオペレーティングシステムとソフトウェアの更新」を参照してください。

```
[ec2-user ~]$ brew update
```

3. インスタンスに EC2 Instance Connect パッケージをインストールします。これによりソフトウェアがインストールされ、sshd がそれを使用するように設定されます。

```
[ec2-user ~]$ brew install ec2-instance-connect
```

次の新しいスクリプトが /opt/aws/bin/ フォルダに表示されます。

```
eic_run_authorized_keys
```

4. (オプション) EC2 Instance Connect がインスタンスに正常にインストールされたことを確認します。

```
[ec2-user ~]$ sudo less /etc/ssh/sshd_config.d/60-ec2-instance-connect.conf
```

AuthorizedKeysCommand 行と AuthorizedKeysCommandUser 行に以下の値が含まれていれば、EC2 Instance Connect は正常にインストールされています。

```
AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_keys %u %f
AuthorizedKeysCommandUser ec2-instance-connect
```

- AuthorizedKeysCommand は、インスタンスメタデータからキーを探すように eic_run_authorized_keys スクリプトを設定します。
- AuthorizedKeysCommandUser は、システムユーザーを ec2-instance-connect として設定します。

> **Note**
>
> AuthorizedKeysCommand や AuthorizedKeysCommandUser を設定済みである場合は、EC2 Instance Connect をインストールしても値は変更されないため、EC2 Instance Connect は使用できません。

RHEL

EC2 Instance Connect を Red Hat Enterprise Linux (RHEL) で起動したインスタンスにインストールするには

1. SSH を使用してインスタンスに接続します。

   次のコマンド内のサンプル値を独自の値に置き換えます。インスタンスの起動時にインスタンスに割り当てた SSH キーペアと、インスタンスを起動するために使用した AMI のデフォルトのユーザー名を使用します。RHEL の場合、デフォルトのユーザー名は `ec2-user` または `root` です。

   ```
   $ ssh -i my_ec2_private_key.pem ec2-user@ec2-a-b-c-d.us-
   west-2.compute.amazonaws.com
   ```

   インスタンスへの接続の詳細についてはSSH クライアントを使用して Linux インスタンスに接続するを参照してください。

2. HTTP または HTTPS プロキシを使用する場合は、現在のシェルセッションで `http_proxy` または `https_proxy` の環境変数を設定する必要があります。

   プロキシを使用していない場合は、この手順を省略できます。

   - HTTP プロキシサーバーの場合は、次のコマンドを実行します。

     ```
     $ export http_proxy=http://hostname:port
     $ export https_proxy=http://hostname:port
     ```

   - HTTPS プロキシサーバーの場合は、次のコマンドを実行します。

     ```
     $ export http_proxy=https://hostname:port
     $ export https_proxy=https://hostname:port
     ```

3. 次のコマンドを実行して、インスタンスに EC2 Instance Connect パッケージをインストールします。

   RHEL 用の EC2 Instance Connect 設定ファイルは Red Hat Package Manager (RPM) パッケージで提供され、RHEL 8 と RHEL 9 用の異なる RPM パッケージ、および Intel/AMD (x86_64) または ARM (AArch64) で実行されるインスタンスタイプ用の異なる RPM パッケージが含まれています。

オペレーティングシステムと CPU アーキテクチャに合ったコマンドブロックを使用してください。

- RHEL 8

  Intel/AMD (x86_64)

```
[ec2-user ~]$ mkdir /tmp/ec2-instance-connect
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-
west-2.s3.us-west-2.amazonaws.com/latest/linux_amd64/ec2-instance-
connect-2.0.0-5.rhel8.x86_64.rpm -o /tmp/ec2-instance-connect/ec2-instance-
connect.rpm
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_amd64/ec2-instance-connect-
selinux-2.0.0-5.noarch.rpm -o /tmp/ec2-instance-connect/ec2-instance-connect-
selinux.rpm
[ec2-user ~]$ sudo yum install -y /tmp/ec2-instance-connect/ec2-instance-
connect.rpm /tmp/ec2-instance-connect/ec2-instance-connect-selinux.rpm
```

  ARM (AArch64)

```
[ec2-user ~]$ mkdir /tmp/ec2-instance-connect
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-
west-2.s3.us-west-2.amazonaws.com/latest/linux_arm64/ec2-instance-
connect-2.0.0-5.rhel8.aarch64.rpm -o /tmp/ec2-instance-connect/ec2-instance-
connect.rpm
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_arm64/ec2-instance-connect-
selinux-2.0.0-5.noarch.rpm -o /tmp/ec2-instance-connect/ec2-instance-connect-
selinux.rpm
[ec2-user ~]$ sudo yum install -y /tmp/ec2-instance-connect/ec2-instance-
connect.rpm /tmp/ec2-instance-connect/ec2-instance-connect-selinux.rpm
```

- RHEL 9

  Intel/AMD (x86_64)

```
[ec2-user ~]$ mkdir /tmp/ec2-instance-connect
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-
west-2.s3.us-west-2.amazonaws.com/latest/linux_amd64/ec2-instance-
```

```
connect-2.0.0-5.rhel9.x86_64.rpm -o /tmp/ec2-instance-connect/ec2-instance-
connect.rpm
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_amd64/ec2-instance-connect-
selinux-2.0.0-5.noarch.rpm -o /tmp/ec2-instance-connect/ec2-instance-connect-
selinux.rpm
[ec2-user ~]$ sudo yum install -y /tmp/ec2-instance-connect/ec2-instance-
connect.rpm /tmp/ec2-instance-connect/ec2-instance-connect-selinux.rpm
```

ARM (AArch64)

```
[ec2-user ~]$ mkdir /tmp/ec2-instance-connect
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-
west-2.s3.us-west-2.amazonaws.com/latest/linux_arm64/ec2-instance-
connect-2.0.0-5.rhel9.aarch64.rpm -o /tmp/ec2-instance-connect/ec2-instance-
connect.rpm
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_arm64/ec2-instance-connect-
selinux-2.0.0-5.noarch.rpm -o /tmp/ec2-instance-connect/ec2-instance-connect-
selinux.rpm
[ec2-user ~]$ sudo yum install -y /tmp/ec2-instance-connect/ec2-instance-
connect.rpm /tmp/ec2-instance-connect/ec2-instance-connect-selinux.rpm
```

次の新しいスクリプトが /opt/aws/bin/ フォルダに表示されます。

```
eic_run_authorized_keys
```

4. (オプション) EC2 Instance Connect がインスタンスに正常にインストールされたことを確認します。

   - RHEL 8 の場合:

     ```
     [ec2-user ~]$ sudo less /lib/systemd/system/sshd.service.d/ec2-instance-
     connect.conf
     ```

   - RHEL 9 の場合:

     ```
     [ec2-user ~]$ sudo less /etc/ssh/sshd_config.d/60-ec2-instance-connect.conf
     ```

AuthorizedKeysCommand 行と AuthorizedKeysCommandUser 行に以下の値が含まれていれば、EC2 Instance Connect は正常にインストールされています。

```
AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_keys %u %f
AuthorizedKeysCommandUser ec2-instance-connect
```

- AuthorizedKeysCommand は、インスタンスメタデータからキーを探すように eic_run_authorized_keys スクリプトを設定します。
- AuthorizedKeysCommandUser は、システムユーザーを ec2-instance-connect として設定します。

> **Note**
>
> AuthorizedKeysCommand や AuthorizedKeysCommandUser を設定済みである場合は、EC2 Instance Connect をインストールしても値は変更されないため、EC2 Instance Connect は使用できません。

Ubuntu

EC2 Instance Connect を Ubuntu 16.04 で起動したインスタンスにインストールするには

1. SSH を使用してインスタンスに接続します。

   次のコマンド内のサンプル値を独自の値に置き換えます。インスタンスの起動時にインスタンスに割り当てた SSH キーペアを使用し、インスタンスを起動するために使用した AMI のデフォルトのユーザー名を使用します。Ubuntu AMI の場合、ユーザー名は ubuntu です。

   ```
   $ ssh -i my_ec2_private_key.pem ubuntu@ec2-a-b-c-d.us-
   west-2.compute.amazonaws.com
   ```

   インスタンスへの接続の詳細については[SSH クライアントを使用して Linux インスタンスに接続する]を参照してください。
2. (オプション) インスタンスに最新の Ubuntu AMI があることを確認します。

   次のコマンドを実行して、インスタンスのすべてのパッケージを更新します。

```
ubuntu:~$ sudo apt-get update
```

```
ubuntu:~$ sudo apt-get upgrade
```

3. インスタンスに EC2 Instance Connect パッケージをインストールします。

```
ubuntu:~$ sudo apt-get install ec2-instance-connect
```

3 つの新しいスクリプトが /usr/share/ec2-instance-connect/ フォルダに表示されます。

```
eic_curl_authorized_keys
eic_parse_authorized_keys
eic_run_authorized_keys
```

4. (オプション) EC2 Instance Connect がインスタンスに正常にインストールされたことを確認します。

```
ubuntu:~$ sudo less /lib/systemd/system/ssh.service.d/ec2-instance-connect.conf
```

AuthorizedKeysCommand 行と AuthorizedKeysCommandUser 行に以下の値が含まれていれば、EC2 Instance Connect は正常にインストールされています。

```
AuthorizedKeysCommand /usr/share/ec2-instance-connect/eic_run_authorized_keys %
%u %%f
AuthorizedKeysCommandUser ec2-instance-connect
```

- AuthorizedKeysCommand は、インスタンスメタデータからキーを探すように eic_run_authorized_keys スクリプトを設定します。
- AuthorizedKeysCommandUser は、システムユーザーを ec2-instance-connect として設定します。

> **Note**
>
> `AuthorizedKeysCommand` や `AuthorizedKeysCommandUser` を設定済みである場合は、EC2 Instance Connect をインストールしても値は変更されないため、EC2 Instance Connect は使用できません。

EC2 Instance Connect を使用して Linux インスタンスに接続する

以下の手順では、Amazon EC2 コンソール、AWS CLI、または SSH クライアントから EC2 Instance Connect を使用して Linux インスタンスに接続する方法について説明します。

EC2 Instance Connect を使用してインスタンスに接続するとAWS CLI、EC2 Instance Connect API から SSH パブリックキーがインスタンスメタデータにプッシュされ、60 秒間保持されます。ユーザーにアタッチされた IAM ポリシーは、このアクションを承認します。独自の SSH キーを使用する場合は、SSH クライアントを使用し、EC2 Instance Connect を使用して SSH キーをインスタンスに明示的にプッシュできます。

考慮事項

EC2 Instance Connect を使用してインスタンスに接続した後、接続は SSH セッションが終了するまで維持されます。接続の期間は、IAM 認証情報の期間によって決定されません。IAM 認証情報の有効期限が切れても、接続は維持されます。EC2 Instance Connect コンソールエクスペリエンスを使用しているときに IAM 認証情報の有効期限が切れた場合は、ブラウザページを閉じて接続を終了します。独自の SSH クライアントと EC2 Instance Connect を使用してキーをプッシュする場合は、SSH タイムアウト値を設定して SSH セッションを自動的に終了できます。

要件

始める前に、前提条件を確認してください。

接続オプション

- Amazon EC2 コンソールを使用した接続
- AWS CLI を使用して接続する
- 独自のキーと SSH クライアントを使用して接続する
- トラブルシューティング

Amazon EC2 コンソールを使用した接続

Amazon EC2 コンソールから EC2 Instance Connect を使用してインスタンスに接続できます。

要件

Amazon EC2 コンソールを使用して接続するには、インスタンスにパブリック IPv4 または IPv6 アドレスが必要です。インスタンスにプライベート IPv4 アドレスのみが存在する場合は、ec2-instance-connectAWS CLIコマンド を使用して接続できます。

Amazon EC2 コンソールを使用して Linux インスタンスに接続するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択し、[接続] を選択してください。
4. [EC2 Instance Connect] タブを選択してください。
5. [パブリック IP を使用して接続] を選択します。
6. 選択する場合は、接続する IP アドレスを選択してください。それ以外の場合は、IP アドレスが自動的に選択されます。
7. [ユーザー名] でユーザー名を確認します。
8. コネクト を選択して接続を確立します。ブラウザ内ターミナルウィンドウが開きます。

AWS CLI を使用して接続する

ec2-instance-connect AWS CLI コマンドを使用すると、SSH クライアントを使用してインスタンスに接続できます。EC2 Instance Connect は、指定された接続タイプに基づいて、事前定義された順序で使用可能な IP アドレスを使用して接続を確立しようとします。IP アドレスが使用できない場合、次の IP アドレスを順番に自動的に試行します。

接続タイプ

`auto` (デフォルト)

EC2 は、次の順序でインスタンスの IP アドレスを使用し、対応する接続タイプを使用して接続を試みます。

1. パブリック IPv4: `direct`
2. プライベート IPv4: `eice`

3. IPv6: `direct`

`direct`

EC2 は、次の順序でインスタンスの IP アドレスを使用して接続を試みます。

1. パブリック IPv4
2. IPv6
3. プライベート IPv4 (EC2 Instance Connect エンドポイント経由で接続しない)

`eice`

EC2 Instance Connect は、インスタンスのプライベート IPv4 アドレスと EC2 Instance Connect エンドポイント を使用して接続を試みます。

> **Note**
>
> 将来的には、`auto` 接続タイプの動作を変更する可能性があります。希望する接続タイプが確実に使用されるようにするには、`--connection-type` を `direct` または `eice` のいずれかに明示的に設定することをお勧めします。

要件

AWS CLI バージョン 2 を使用する必要があります。詳細については、「AWS CLI の最新バージョンを使用してインストールまたは更新を行う」を参照してください。

インスタンス ID を使用してインスタンスに接続するには

インスタンス ID のみがわかっていて、インスタンスへの接続時に使用する接続タイプを EC2 Instance Connect に決定させる場合、インスタンス ID に ec2-instance-connect ssh の CLI コマンドを使用します。

```
aws ec2-instance-connect ssh --instance-id i-1234567890example
```

インスタンス ID と EC2 Instance Connect エンドポイントを使用してインスタンスに接続するには

EC2 Instance Connect エンドポイント を介してインスタンスに接続する場合は、前述のコマンドを使用し、`--connection-type` パラメータと `eice` 値も指定します。

```
aws ec2-instance-connect ssh --instance-id i-1234567890example --connection-type eice
```

インスタンス ID と独自のプライベートキーファイルを使用してインスタンスに接続するには

独自のプライベートキーを使用して EC2 Instance Connect エンドポイント経由でインスタンスに接続する場合は、インスタンス ID とプライベートキーファイルへのパスを指定します。パスに *file://*を含めないでください。次のようなパスは失敗します: *file:///path/to/key*。

```
aws ec2-instance-connect ssh --instance-id i-1234567890example --private-key-file /
path/to/key.pem
```

> **Tip**
>
> このコマンドの使用時にエラーが発生した場合は、ssh パラメータがこのメジャーバージョンでのみ使用できるため、AWS CLI バージョン 2 を使用していることを確認してください。また、AWS CLI バージョン 2 の最新マイナーバージョンに定期的に更新して、最新の機能にアクセスすることをお勧めします。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS CLI バージョン 2 について」を参照してください。

独自のキーと SSH クライアントを使用して接続する

EC2 Instance Connect API の使用中に、独自の SSH キーを使用して、選択した SSH クライアントからインスタンスに接続できます。これは、インスタンスにパブリックキーをプッシュする EC2 Instance Connect 機能からメリットを得ることを可能にします。この接続方法は、パブリック IP アドレスとプライベート IP アドレスを持つインスタンスに対して機能します。

要件

- キーペアの要件
  - サポートされているタイプ: RSA (OpenSSH および SSH2) および ED25519
  - サポートされている長さ: 2048 および 4096
  - 詳細については「サードパーティー製のツールを使用してキーペアを作成し、Amazon EC2 にパブリックキーをインポートする」を参照してください。
- プライベート IP アドレスのみがあるインスタンスに接続する場合、SSH セッションを開始するローカルコンピュータには、EC2 Instance Connect サービスエンドポイントへの接続 (SSH パブリックキーをインスタンスにプッシュするため) と、SSH セッションを確立するためのインスタ

ンスのプライベート IP アドレスへのネットワーク接続が必要です。EC2 Instance Connect のサービスエンドポイントには、インターネットまたは Direct Connect パブリック仮想インターフェイス経由で到達が可能です。インスタンスのプライベート IP アドレスに接続するには、Direct Connect、AWS Site-to-Site VPN や VPC ピアリングなどのサービスを利用できます。

独自のキーと任意の SSH クライアントを使用してインスタンスに接続するには

1. (オプション) 新しい SSH プライベートキーとパブリックキーを生成する

   新しい SSH プライベートキーとパブリックキー (my_key および my_key.pub) は、次のコマンドを使用して生成できます。

   ```
   ssh-keygen -t rsa -f my_key
   ```

2. SSH パブリックキーをインスタンスにプッシュする

   send-ssh-public-key コマンドを使用して、SSH パブリックキーをインスタンスにプッシュします。AL2023 または Amazon Linux 2 を使用してインスタンスを起動した場合、AMI のデフォルトのユーザー名は ec2-user です。Ubuntu を使用してインスタンスを起動した場合、AMI のデフォルトのユーザー名は ubuntu です。

   以下に、ec2-user を認証するために、指定されたアベイラビリティーゾーンで指定されたインスタンスにパブリックキーをプッシュする例を示しています。

   ```
   aws ec2-instance-connect send-ssh-public-key \
       --region us-west-2 \
       --availability-zone us-west-2b \
       --instance-id i-001234a4bf70dec41EXAMPLE \
       --instance-os-user ec2-user \
       --ssh-public-key file://my_key.pub
   ```

3. プライベートキーを使用してインスタンスに接続する

   パブリックキーがインスタンスメタデータから削除される前に (削除されるまでの時間は 60 秒です)、プライベートキーを使用してインスタンスに接続するには、ssh コマンドを使用します。パブリックキーに対応するプライベートキー、インスタンスを起動するために使用した AMI のデフォルトのユーザー名、およびインスタンスのパブリック DNS 名を指定します (プライベートネットワーク経由で接続する場合は、プライベート DNS 名または IP アドレスを指定します)。IdentitiesOnly=yes オプションを追加し、ssh config 内のファイルと指定したキーのみが接続に使用されるようにします。

```
ssh -o "IdentitiesOnly=yes" -i my_key ec2-
user@ec2-198-51-100-1.compute-1.amazonaws.com
```

次の例では、timeout 3600 を使用して SSH セッションが 1 時間後に終了するように設定します。セッション中に開始されたプロセスは、セッション終了後もインスタンスで実行し続ける場合があります。

```
timeout 3600 ssh -o “IdentitiesOnly=yes” -i my_key ec2-
user@ec2-198-51-100-1.compute-1.amazonaws.com
```

トラブルシューティング

インスタンスへの接続を試みた際にエラーが発生した場合は、以下を参照してください。

- Amazon EC2 Linux インスタンスへの接続に関する問題のトラブルシューティング
- EC2 Instance Connect を使用して EC2 インスタンスへ接続しようとしたときの問題をトラブルシューティングするにはどうすればよいですか?

EC2 Instance Connect のアンインストール

EC2 Instance Connect を無効にするには、Linux インスタンスに接続し、Linux にインストールされている ec2-instance-connect パッケージをアンインストールします。sshd 設定が EC2 Instance Connect をインストールしたときのまま変更されていない場合、ec2-instance-connect をアンインストールすると、sshd 設定も削除されます。sshd 設定が EC2 Instance Connect のインストール後に変更されている場合は、それを手動で更新する必要があります。

Amazon Linux

EC2 Instance Connect が事前設定されている AL2023 および Amazon Linux 2 2.0.20190618 以降では EC2 Instance Connect をアンインストールできます。

Amazon Linux を使用して起動したインスタンスから EC2 Instance Connect をアンインストールするには

1. SSH を使用してインスタンスに接続します。インスタンスの起動時に使用した SSH キーペアと AL2023 または Amazon Linux 2 AMI のデフォルトのユーザー名 ec2-userを指定します。

例えば、次の ssh コマンドは、キーペア ec2-a-b-c-d.us-west-2.compute.amazonaws.com を使用して、パブリック DNS 名 my_ec2_private_key.pem でインスタンスに接続します。

```
$ ssh -i my_ec2_private_key.pem ec2-user@ec2-a-b-c-d.us-west-2.compute.amazonaws.com
```

2. yum コマンドを使用して ec2-instance-connect パッケージをアンインストールします。

```
[ec2-user ~]$ sudo yum remove ec2-instance-connect
```

Ubuntu

Ubuntu AMI を使用して起動したインスタンスから EC2 Instance Connect をアンインストールするには

1. SSH を使用してインスタンスに接続します。インスタンスの起動時に使用した SSH キーペアと、Ubuntu AMI のデフォルトのユーザー名 ubuntuを指定します。

   例えば、次の ssh コマンドは、キーペア ec2-a-b-c-d.us-west-2.compute.amazonaws.com を使用して、パブリック DNS 名 my_ec2_private_key.pem でインスタンスに接続します。

```
$ ssh -i my_ec2_private_key.pem ubuntu@ec2-a-b-c-d.us-west-2.compute.amazonaws.com
```

2. apt-get コマンドを使用して ec2-instance-connect パッケージをアンインストールします。

```
ubuntu:~$ sudo apt-get remove ec2-instance-connect
```

プライベート IP アドレスと EC2 Instance Connect Endpoint を使用した、インスタンスへの接続

EC2 Instance Connect エンドポイントを使用すると、踏み台ホストを使用したり、仮想プライベートクラウド (VPC) がインターネットに直接接続したりする必要がなく、インターネットからインスタンスに安全に接続できます。

利点

- インスタンスにパブリック IPv4 または IPv6 アドレスがなくても、インスタンスに接続できます。AWS は、実行中のインスタンスに関連付けられたパブリック IPv4 アドレスや Elastic IP アドレスを含む、すべてのパブリック IPv4 アドレスに対して料金を請求します。詳細については、「Amazon VPC の料金」ページの「パブリック IPv4 アドレス」タブを参照してください。
- VPC にインターネットゲートウェイ経由の直接インターネット接続を必須としなくても、インターネットからインスタンスに接続できます。
- IAM ポリシーとアクセス許可を使用して、インスタンスに接続する EC2 Instance Connect エンドポイント の作成と使用へのアクセスを制御できます。
- インスタンスへの接続を試みると、成功または失敗を問わず、すべて CloudTrail にログが作成されます。

料金

EC2 Instance Connect エンドポイントの使用に追加コストはかかりません。EC2 Instance Connect エンドポイント を使用して、別のアベイラビリティーゾーンにあるインスタンスに接続する場合、アベイラビリティーゾーン間のデータ転送に追加料金がかかります。

内容

- 仕組み
- 考慮事項
- EC2 Instance Connect エンドポイント を使用するためのアクセス許可の付与
- EC2 Instance Connect エンドポイントのセキュリティグループ
- EC2 Instance Connect エンドポイントの作成
- EC2 Instance Connect エンドポイントを変更する
- EC2 Instance Connect エンドポイントを削除する
- EC2 Instance Connect エンドポイントを使用して Amazon EC2 インスタンスに接続する

- EC2 Instance Connect エンドポイントを介して確立された接続のログの作成
- EC2 Instance Connect エンドポイントのサービスにリンクされたロール
- EC2 Instance Connect エンドポイントのクォータ

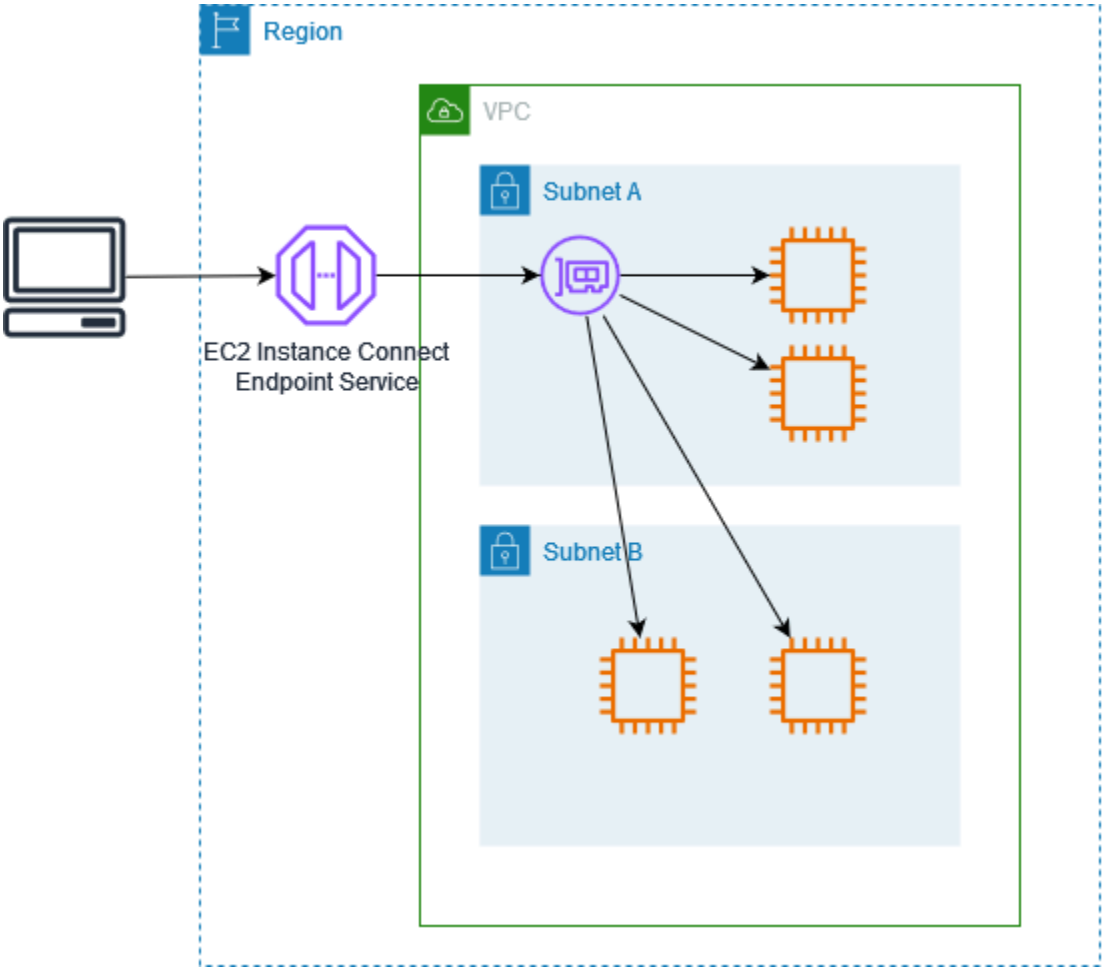
仕組み

EC2 Instance Connect エンドポイントは ID を認識する TCP プロキシです。EC2 Instance Connect エンドポイントサービスは、IAM エンティティの認証情報を使用して、コンピュータからエンドポイントへのプライベートトンネルを確立します。トラフィックは VPC に到達する前に認証され、承認されます。

追加のセキュリティグループルールを設定すると、インスタンスへのインバウンドトラフィックを制限できます。例えば、インスタンスのインバウンドルールを使用して、EC2 Instance Connect エンドポイントのみから管理ポートへのトラフィックを許可できます。

ルートテーブルルールを設定すると、エンドポイントが VPC の任意のサブネットの任意のインスタンスに接続できるようにすることができます。

次の図は、ユーザーが EC2 Instance Connect エンドポイントを使用してインターネットからインスタンスに接続する方法を示しています。まず、サブネット A に [EC2 Instance Connect エンドポイント]を作成します。サブネット内のエンドポイント用のネットワークインターフェイスを作成します。これは VPC 内のインスタンス宛のトラフィックのエントリポイントとして機能します。サブネット B のルートテーブルがサブネット A からのトラフィックを許可している場合、エンドポイントを使用してサブネット B のインスタンスに到達できます。



考慮事項

開始する前に、以下を考慮してください。

- EC2 Instance Connect エンドポイントは、特に管理トラフィックのユースケースを対象としており、大量のデータ転送を想定していません。大量のデータ転送はスロットリングされます。
- EC2 Instance Connect エンドポイントを作成すると、プライベート IPv4 アドレスまたは IPv6 アドレスを持つインスタンスへのトラフィックをサポートできます。エンドポイントの IP アドレスタイプは、インスタンスの IP アドレスと一致する必要があります。すべての IP アドレスタイプをサポートするエンドポイントを作成できます。
- (Linux インスタンス) 独自のキーペアを使用すると、任意の Linux AMI を使用できます。その他の場合、インスタンスには EC2 Instance Connect がインストールされている必要があります。どの AMI に EC2 Instance Connect が含まれているか、およびサポートされている他の AMI にインストールする方法については、EC2 Instance Connect のインストール を参照してください。

- セキュリティグループを EC2 Instance Connect エンドポイントに割り当てることができます。それ以外の場合は、VPC のデフォルトのセキュリティグループを使用します。EC2 Instance Connect エンドポイントのセキュリティグループは、宛先インスタンスへのアウトバウンドトラフィックを許可する必要があります。詳細については「EC2 Instance Connect エンドポイントのセキュリティグループ」を参照してください。
- EC2 Instance Connect エンドポイント を設定し、リクエストをインスタンスにルーティングするときに、クライアントのソース IP アドレスを保持することができます。それ以外の場合、ネットワークインターフェイスの IP アドレスは、すべての受信トラフィックのクライアント IP になります。
  - クライアント IP 保存を有効にする場合、インスタンスのセキュリティグループはクライアントからのトラフィックを許可する必要があります。また、インスタンスは EC2 Instance Connect エンドポイント と同じ VPC にある必要があります。
  - クライアント IP 保存を無効にする場合、インスタンスのセキュリティグループは VPC からのトラフィックを許可する必要があります。これがデフォルトです。
  - クライアント IP 保存は、IPv4 EC2 Instance Connect エンドポイントでのみサポートされています。クライアント IP 保存を使用するには、EC2 Instance Connect エンドポイントの IP アドレスタイプが IPv4 である必要があります。IP アドレスタイプがデュアルスタックまたは IPv6 の場合、クライアント IP 保存はサポートされません。
  - インスタンスタイプが C1、CG1、CG2、G1、HI1、M1、M2、M3、T1 である場合、クライアント IP 保存をサポートしません。クライアント IP の保存を有効にし、EC2 Instance Connect エンドポイントを使用してこれらのインスタンスタイプのいずれかのインスタンスに接続しようとすると、接続は失敗します。
  - トラフィックがトランジットゲートウェイを経由してルーティングされる場合、クライアント IP の保存はサポートされません。
- EC2 Instance Connectエンドポイント を作成した場合、AWS Identity and Access Management (IAM) の Amazon EC2 サービスに対して、サービスにリンクされたロールが自動的に作成されます。Amazon EC2 は、サービスにリンクされたロールを使用してアカウントインターフェイスをプロビジョニングします。これは、EC2 Instance Connect エンドポイントを作成するときに必要です。詳細については「EC2 Instance Connect エンドポイントのサービスにリンクされたロール」を参照してください。
- EC2 Instance Connect エンドポイントを VPC およびサブネットごとに 1 つだけ作成できます。詳細については「EC2 Instance Connect エンドポイントのクォータ」を参照してください。別の EC2 Instance Connect エンドポイントを同じ VPC 内の別のアベイラビリティーゾーンに作成する必要がある場合は、まず既存の EC2 Instance Connect エンドポイントを削除する必要があります。そうしないと、クォータエラーが発生します。

- 各 EC2 インスタンス接続エンドポイントは、最大 20 の同時接続をサポートできます。
- 確立された TCP 接続の最大持続時間: 1 時間 (3,600 秒)。IAM ポリシーでは最大許容時間を最大 3,600 秒で指定できます。詳細については「ユーザーに EC2 Instance Connect エンドポイントを使用してインスタンスへ接続するためのアクセス許可」を参照してください。

  接続の期間は、IAM 認証情報の期間によって決定されません。IAM 認証情報の有効期限が切れても、指定した最大期間に達するまで接続は維持されます。EC2 Instance Connect エンドポイントのコンソール機能を使用してインスタンスに接続する場合は、[最大トンネル期間 (秒)] を IAM 認証情報の期間より短い値に設定します。IAM 認証情報が早期に期限切れになった場合は、ブラウザページを閉じてインスタンスへの接続を終了します。

EC2 Instance Connect エンドポイント を使用するためのアクセス許可の付与

デフォルトでは、IAM エンティティには EC2 Instance Connect エンドポイントを作成、記述、変更するためのアクセス許可はありません。IAM 管理者は、必要なリソースに対して特定のアクションを実行するために必要な IAM ポリシーを作成できます。

IAM ポリシーの作成について詳しくは、「IAM ユーザーガイド」の「IAM ポリシーの作成」を参照してください。

以下のポリシー例では、EC2 Instance Connect エンドポイントへのアクセス許可を制御する方法を紹介します。

例

- EC2 Instance Connect エンドポイントを作成、記述、削除するアクセス許可
- ユーザーに EC2 Instance Connect エンドポイントを使用してインスタンスへ接続するためのアクセス許可
- 特定の IP アドレス範囲からのみ接続できるアクセス許可

EC2 Instance Connect エンドポイントを作成、記述、削除するアクセス許可

EC2 Instance Connect エンドポイントを作成して変更するには、ユーザーには次のアクションのアクセス許可が必要です。

- `ec2:CreateInstanceConnectEndpoint`
- `ec2:CreateNetworkInterface`
- `ec2:CreateTags`

- ec2:ModifyInstanceConnectEndpoint
- iam:CreateServiceLinkedRole

EC2 Instance Connect エンドポイントを記述して削除するには、ユーザーに次のアクションに対するアクセス許可が必要です。

- ec2:DescribeInstanceConnectEndpoints
- ec2:DeleteInstanceConnectEndpoint

すべてのサブネットで EC2 Instance Connect エンドポイントを作成、記述、削除するアクセス許可を付与するポリシーを作成できます。または、サブネット ARN を許可済みの Resource として指定するか、ec2:SubnetID 条件キーを使用して、指定したサブネットのアクションのみを制限することもできます。aws:ResourceTag 条件キーを使用して、特定のタグによるエンドポイントの作成を明示的に許可または拒否することもできます。詳細については、「 IAM ユーザーガイド」の「IAM のポリシーとアクセス許可」を参照してください。

IAM ポリシーの例

次の IAM ポリシーの例では、Resource セクションはアスタリスク (*) で指定されたすべてのサブネットでエンドポイントを作成、変更、削除するアクセス許可が付与されます。ec2:Describe* API アクションは、リソースレベルのアクセス許可をサポートしていません。したがって、Resource の要素には、* (ワイルドカード) が必要です。

ユーザーに EC2 Instance Connect エンドポイントを使用してインスタンスへ接続するためのアクセス許可

ec2-instance-connect:OpenTunnel アクションは、EC2 Instance Connect エンドポイントを介して接続するインスタンスへの TCP 接続を確立するためのアクセス許可を付与します。使用する EC2 Instance Connect エンドポイントを指定できます。または、アスタリスク (*) が付いている Resource を使用すると、ユーザーは利用可能な任意の EC2 Instance Connect エンドポイントを使用できます。条件キーとしてのリソースタグの有無に基づいて、インスタンスへのアクセスを制限することもできます。

条件

- ec2-instance-connect:remotePort — TCP 接続の確立に使用できるインスタンスのポート。この条件キーを使用した場合、ポリシーで指定されているポート以外のポート上のインスタンスに接続しようとすると失敗します。

- `ec2-instance-connect:privateIpAddress` — TCP 接続を確立するインスタンスに関連する宛先プライベート IP アドレス。例えば、1 つの IP アドレス (`10.0.0.1/32` など) を指定することも、CIDR を介して IP 範囲 (`10.0.1.0/28` など) を指定することもできます。この条件キーを使用した場合、別のプライベート IP アドレスまたは CIDR 範囲外のインスタンスに接続しようとすると失敗します。
- `ec2-instance-connect:maxTunnelDuration` — 確立された TCP 接続の最大期間。単位は秒で、期間の範囲は最小 1 秒から最大 3,600 秒 (1 時間) です。条件が指定されていない場合、デフォルトの時間は 3,600 秒 (1 時間) に設定されます。IAM ポリシーで指定された期間よりも長く、またはデフォルトの最大期間よりも長くインスタンスに接続しようとすると、失敗します。指定した期間が経過すると、接続は切断されます。

  `maxTunnelDuration` は IAM ポリシーで指定されていて、指定した値が 3,600 秒 (デフォルト) 未満の場合は、インスタンスに接続するときにコマンドで `--max-tunnel-duration` を指定する必要があります。インスタンスへの接続方法については、「EC2 Instance Connect エンドポイントを使用して Amazon EC2 インスタンスに接続する」を参照ください。

ユーザーには、EC2 Instance Connect エンドポイントに対するリソースタグの有無に基づいて、インスタンスへの接続を確立するためのアクセス許可を付与することもできます。詳細については、「IAM ユーザーガイド」の「IAM のポリシーとアクセス許可」を参照してください。

Linux インスタンスでは、`ec2-instance-connect:SendSSHPublicKey` アクションでパブリックキーをインスタンスにプッシュするアクセス許可が付与されます。`ec2:osuser` 条件は、パブリックキーをインスタンスにプッシュできる OS (オペレーティングシステム) ユーザーの名前を指定します。インスタンスの起動に使用した AMI のデフォルトのユーザー名を使用します。詳細については「IAM への EC2 Instance Connect のアクセス許可の付与」を参照してください。

IAM ポリシーの例

次の IAM ポリシーの例では、IAM プリンシパルは、指定されたエンドポイント ID `eice-123456789abcdef` で識別される指定された EC2 Instance Connect エンドポイントのみを使用してインスタンスに接続できます。接続は、すべての条件が満たされた場合にのみ正常に確立されます。

> **Note**
>
> `ec2:Describe*` API アクションは、リソースレベルのアクセス許可をサポートしていません。したがって、`Resource` の要素には、`*` (ワイルドカード) が必要です。

Linux

この例では、インスタンスへの接続がポート 22 (SSH) で確立されているかどうか、インスタンスのプライベート IP アドレスが 10.0.1.0/31 (10.0.1.0 ~ 10.0.1.1) の範囲内であるかどうか、および maxTunnelDuration が 3600 秒以下であるかどうかが評価されます。3600 秒(1 時間) 後 に接続が切断されます。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Sid": "EC2InstanceConnect",
            "Action": "ec2-instance-connect:OpenTunnel",
            "Effect": "Allow",
            "Resource": "arn:aws:ec2:us-east-1:111122223333:instance-connect-
endpoint/eice-123456789abcdef",
            "Condition": {
                "NumericEquals": {
                    "ec2-instance-connect:remotePort": "22"
                },
                "IpAddress": {
                    "ec2-instance-connect:privateIpAddress": "10.0.1.0/31"
                },
                "NumericLessThanEquals": {
                    "ec2-instance-connect:maxTunnelDuration": "3600"
                }
            }
        },
        {
            "Sid": "SSHPublicKey",
            "Effect": "Allow",
            "Action": "ec2-instance-connect:SendSSHPublicKey",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ec2:osuser": "ami-username"
                }
            }
        },
        {
            "Sid": "Describe",
```

```
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceConnectEndpoints"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

Windows

この例では、インスタンスへの接続がポート 3389 (RDP) で確立されているかどうか、インスタンスのプライベート IP アドレスが 10.0.1.0/31 (10.0.1.0 ~ 10.0.1.1) の範囲内であるかどうか、および maxTunnelDuration が 3600 秒以下であるかどうかが評価されます。3600 秒(1 時間) 後 に接続が切断されます。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Sid": "EC2InstanceConnect",
            "Action": "ec2-instance-connect:OpenTunnel",
            "Effect": "Allow",
            "Resource": "arn:aws:ec2:us-east-1:111122223333:instance-connect-
endpoint/eice-123456789abcdef",
            "Condition": {
                "NumericEquals": {
                    "ec2-instance-connect:remotePort": "3389"
                },
                "IpAddress": {
                    "ec2-instance-connect:privateIpAddress": "10.0.1.0/31"
                },
                "NumericLessThanEquals": {
                    "ec2-instance-connect:maxTunnelDuration": "3600"
                }
            }
        },
        {
            "Sid": "Describe",
```

```
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceConnectEndpoints"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

特定の IP アドレス範囲からのみ接続できるアクセス許可

次の IAM ポリシーの例では、ポリシーで指定された IP アドレス範囲内の IP アドレスから接続することを条件に、IAM プリンシパルがインスタンスに接続することを許可しています。IAM プリンシパルが 192.0.2.0/24 (このポリシーの IP アドレス範囲の例) 内ではない IP アドレスから OpenTunnel の呼び出しを行った場合、応答は Access Denied になります。詳細については、「IAM ユーザーガイド」の「*aws:SourceIp*」を参照してください。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "ec2-instance-connect:OpenTunnel",
            "Resource": "arn:aws:ec2:us-east-1:111122223333:instance-connect-
endpoint/eice-123456789abcdef",
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": "192.0.2.0/24"
                },
                "NumericEquals": {
                    "ec2-instance-connect:remotePort": "22"
                }
            }
        },
        {
            "Sid": "SSHPublicKey",
            "Effect": "Allow",
            "Action": "ec2-instance-connect:SendSSHPublicKey",
            "Resource": "*",
```

```
            "Condition": {
                "StringEquals": {
                    "ec2:osuser": "ami-username"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceConnectEndpoints"
            ],
            "Resource": "*"
        }
    ]
}
```

EC2 Instance Connect エンドポイントのセキュリティグループ

セキュリティグループは、関連付けられたリソースに到達できるトラフィックおよびリソースから発信できるトラフィックを制御します。例えば、Amazon EC2 インスタンスに出入りするトラフィックは、そのインスタンスに関連付けられているセキュリティグループによって特に許可されていない限り、拒否する場合を考えます。

次の例は、EC2 Instance Connect エンドポイントとターゲットインスタンスのセキュリティグループルールを設定する方法を示しています。

例

- EC2 Instance Connect エンドポイント セキュリティグループルール
- ターゲットインスタンスセキュリティグループのルール

EC2 Instance Connect エンドポイント セキュリティグループルール

EC2 Instance Connect エンドポイントのセキュリティグループは、エンドポイントから出る対象のインスタンスに対する、アウトバウンドトラフィックを許可する必要があります。送信先として、インスタンスセキュリティグループ、または VPC の IPv4 または IPv6 アドレス範囲のいずれかを指定できます。

エンドポイントへのトラフィックは EC2 Instance Connect エンドポイントサービスから送信され、エンドポイントセキュリティグループのインバウンドルールに関係なく許可されます。EC2 Instance

Connect エンドポイント を使用してインスタンスに接続できるユーザーを制御するには、IAM ポリシーを使用します。詳細については「ユーザーに EC2 Instance Connect エンドポイントを使用してインスタンスへ接続するためのアクセス許可」を参照してください。

アウトバウンドルールの例: セキュリティグループ参照

次の例では、セキュリティグループ参照を使用しています。つまり、宛先はターゲットインスタンスに関連付けられたセキュリティグループです。このルールにより、エンドポイントからこのセキュリティグループを使用するすべてのインスタンスへのアウトバウンドトラフィックが許可されます。

プロトコル	送信先	ポート範囲	コメント
TCP	*########## ####### ID*	22	インスタンスセキュリティグループに関連付けられているすべてのインスタンスへのアウトバウンド SSH トラフィックを許可します。

アウトバウンドルールの例: IPv4 アドレス範囲

次の例では、指定した IPv4 アドレス範囲へのアウトバウンドトラフィックを許可します。インスタンスの IPv4 アドレスはサブネットから割り当てられるため、VPC の IPv4 アドレス範囲を使用できます。

プロトコル	送信先	ポート範囲	コメント
TCP	*VPC IPv4 CIDR*	22	VPC へのアウトバウンド SSH トラフィックを許可します

アウトバウンドルールの例: IPv6 アドレス範囲

次の例では、指定した IPv6 アドレス範囲へのアウトバウンドトラフィックを許可します。インスタンスの IPv6 アドレスはサブネットから割り当てられるため、VPC の IPv6 アドレス範囲を使用できます。

プロトコル	送信先	ポート範囲	コメント
TCP	*VPC IPv6 CIDR*	22	VPC へのアウトバウンド SSH トラフィックを許可します

ターゲットインスタンスセキュリティグループのルール

ターゲットインスタンスのセキュリティグループルールで、EC2 Instance Connect エンドポイントからのインバウンドトラフィックを許可する必要があります。ソースとして、エンドポイントセキュリティグループ、または IPv4 または IPv6 アドレス範囲のいずれかを指定できます。IPv4 アドレス範囲を指定する場合、送信元はクライアント IP の保存がオフかオンかによって異なります。詳細については「考慮事項」を参照してください。

セキュリティグループはステートフルであるため、インスタンスのセキュリティグループのアウトバウンドルールに関係なく、レスポンストラフィックは VPC から出ることができます。

インバウンドルールの例: セキュリティグループ参照

次の例では、セキュリティグループ参照を使用しています。つまり、ソースはエンドポイントに関連付けられたセキュリティグループです。このルールは、クライアント IP 保護がオンかオフかに関係なく、エンドポイントからこのセキュリティグループを使用するすべてのインスタンスへのインバウンド SSH トラフィックを許可します。SSH 用のインバウンドセキュリティグループルールが他にない場合、インスタンスはエンドポイントからの SSH トラフィックのみを受け入れます。

プロトコル	ソース	ポート範囲	コメント
TCP	*########## ####### ID*	22	エンドポイントセキュリティグループに関連付けられているリソースからのインバウンド SSH トラフィックを許可します

インバウンドルールの例: クライアント IP 保護オフ

次の例では、指定した IPv4 アドレス範囲からのインバウンド SSH トラフィックを許可します。クライアント IP 保存が無効であるため、ソース IPv4 アドレスはエンドポイントのネットワークインターフェイスのアドレスになります。エンドポイントのネットワークインターフェイスのアドレスは

サブネットから割り当てられるため、VPC の IPv4 アドレス範囲を使用して VPC 内のすべてのインスタンスへの接続を許可できます。

プロトコル	ソース	ポート範囲	コメント
TCP	*VPC IPv4 CIDR*	22	VPC からのインバウンド SSH トラフィックを許可する

インバウンドルールの例: クライアント IP 保護オン

次の例では、指定した IPv4 アドレス範囲からのインバウンド SSH トラフィックを許可します。クライアント IP 保存が有効であるため、ソース IPv4 アドレスはクライアントのアドレスになります。

プロトコル	ソース	ポート範囲	コメント
TCP	*##### IPv4 # #####*	22	指定したクライアント IPv4 アドレス範囲からのインバウンドトラフィックを許可します

EC2 Instance Connect エンドポイントの作成

EC2 Instance Connect エンドポイントを作成して、インスタンスに安全に接続できるようにします。

考慮事項

- 共有サブネット – EC2 Instance Connect エンドポイントを、共有しているサブネットに作成できます。ただし、VPC 所有者が、共有しているサブネットに作成した EC2 Instance Connect エンドポイントを使用することはできません。
- IP アドレスタイプ – EC2 Instance Connect エンドポイントは、次のアドレスタイプをサポートしますが、これらはサブネットと互換性がある必要があります。
  - `ipv4` – プライベート IPv4 アドレスを持つ EC2 インスタンスにのみ接続します。
  - `dualstack` – プライベート IPv4 アドレスまたは IPv6 アドレスを使用して EC2 インスタンスに接続します。
  - `ipv6` – IPv6 アドレスを持つ EC2 インスタンスにのみ接続します。

前提条件

EC2 Instance Connect エンドポイントを作成するには、必要な IAM アクセス許可が付与されている必要があります。詳細については「EC2 Instance Connect エンドポイントを作成、記述、削除するアクセス許可」を参照してください。

Console

EC2 Instance Connect エンドポイントを作成するには

1. Amazon VPC コンソールの https://console.aws.amazon.com/vpc/ を開いてください。
2. 左のナビゲーションペインで [エンドポイント] を選択してください。
3. [エンドポイントの作成] を選択し、次のようにエンドポイント設定を指定します。
   a. (オプション) [名前タグ] にエンドポイントの名前を入力してください。
   b. [タイプ] に [EC2 Instance Connect エンドポイント] を選択してください。
   c. [VPC] の [ネットワーク設定] で、ターゲットインスタンスが含まれている VPC を選択してください。
   d. (オプション) クライアント IP アドレスを保持するには、[その他の設定] を展開して [クライアント IP の保持] チェックボックスを選択してください。それ以外の場合、デフォルトではエンドポイントのネットワークインターフェースをクライアント IP アドレスとして使用します。

      > **Note**
      >
      > このオプションは、エンドポイントの IP アドレスタイプが IPv4 として設定されている場合にのみ使用できます。

   e. (オプション) [セキュリティグループ] で、エンドポイントに関連付けるセキュリティグループを選択してください。それ以外の場合は、デフォルトでは VPC のデフォルトセキュリティグループを使用します。詳細については「EC2 Instance Connect エンドポイントのセキュリティグループ」を参照してください。
   f. [サブネット] で、エンドポイントを作成するサブネットを選択してください。
   g. [IP アドレスタイプ] で、エンドポイントの IP アドレスタイプを選択します。インスタンスへの IPv4 と IPv6 の両方の接続をサポートする必要がある場合は、[デュアルスタック] を選択します。クライアント IP 保存をサポートする必要がある場合は、[IPv4] を選択します。

h. (オプション) タグを追加するには、[新しいタグを追加] を選択し、そのタグのキーと値を入力してください。

4. 設定を確認したら、[プロジェクトの作成] を選択してください。

   エンドポイントの初期状態は [保留中] です。このエンドポイントを使用してインスタンスに接続する前に、ステータスが [使用可能] になるまで待つ必要があります。これは数分かかることがあります。

5. エンドポイントを使用してインスタンスに接続するには、インスタンスへの接続 を参照してください。

AWS CLI

EC2 Instance Connect エンドポイントを作成するには

create-instance-connect-endpoint コマンドを使用します。

```
aws ec2 create-instance-connect-endpoint \
    --subnet-id subnet-0123456789example
```

エンドポイントがサポートするトラフィックのタイプを指定するには、--ip-address-type パラメータを含めます。有効な値は ipv4、dualstack、または ipv6 です。サブネットは、指定した IP アドレスタイプをサポートしている必要があります。--ip-address-type パラメータを省略した場合、デフォルト値はサブネットでサポートされている IP アドレスタイプによって決まります。

```
aws ec2 create-instance-connect-endpoint \
    --subnet-id subnet-0123456789example \
    --ip-address-type ipv4
```

以下は出力の例です。

```
{
        "OwnerId": "111111111111",
        "InstanceConnectEndpointId": "eice-0123456789example",
        "InstanceConnectEndpointArn": "arn:aws:ec2:us-east-1:111111111111:instance-
connect-endpoint/eice-0123456789example",
        "State": "create-complete",
        "StateMessage": "",
```

```
        "DnsName": "eice-0123456789example.0123abcd.ec2-instance-connect-
endpoint.us-east-1.amazonaws.com",
        "FipsDnsName": "eice-0123456789example.0123abcd.fips.ec2-instance-connect-
endpoint.us-east-1.amazonaws.com",
        "NetworkInterfaceIds": [
            "eni-0123abcd"
        ],
        "VpcId": "vpc-0123abcd",
        "AvailabilityZone": "us-east-1a",
        "AvailabilityZoneId": "use1-az4",
        "CreatedAt": "2023-04-07T15:43:53.000Z",
        "SubnetId": "subnet-0123abcd",
        "PreserveClientIp": false,
        "SecurityGroupIds": [
            "sg-0123abcd"
        ],
        "Tags": [],
        "IpAddressType": "ipv4"
}
```

作成ステータスをモニタリングするには

State フィールドの初期値は create-in-progress です。このエンドポイントを使用してインスタンスに接続するには、状態が create-complete になるまで待ってください。describe-instance-connect-endpoints コマンドを使用して EC2 Instance Connect エンドポイントのステータスを監視します。--query パラメータは、State フィールドに結果をフィルタリングします。

```
aws ec2 describe-instance-connect-endpoints --instance-connect-endpoint-
ids eice-0123456789example --query InstanceConnectEndpoints[*].State --output text
```

以下は出力の例です。

```
create-complete
```

PowerShell

EC2 Instance Connect エンドポイントを作成する方法

New-EC2InstanceConnectEndpoint コマンドレットを使用します。

```
New-EC2InstanceConnectEndpoint -SubnetId subnet-0123456789example
```

エンドポイントがサポートするトラフィックのタイプを指定するには、-IpAddressType パラメータを含めます。有効な値は ipv4、dualstack、または ipv6 です。サブネットは、指定した IP アドレスタイプをサポートしている必要があります。-IpAddressType パラメータを省略した場合、デフォルト値はサブネットでサポートされている IP アドレスタイプによって決まります。

```
New-EC2InstanceConnectEndpoint -SubnetId subnet-0123456789example -
IpAddressType ipv4
```

以下は出力の例です。

```
OwnerId                     : 111111111111
InstanceConnectEndpointId   : eice-0123456789example
InstanceConnectEndpointArn  : arn:aws:ec2:us-east-1:111111111111:instance-connect-
endpoint/eice-0123456789example
State                       : create-complete
StateMessage                :
DnsName                     : eice-0123456789example.0123abcd.ec2-instance-connect-
endpoint.us-east-1.amazonaws.com
FipsDnsName                 : eice-0123456789example.0123abcd.fips.ec2-instance-
connect-endpoint.us-east-1.amazonaws.com
NetworkInterfaceIds         : {eni-0123abcd}
VpcId                       : vpc-0123abcd
AvailabilityZone            : us-east-1a
AvailabilityZoneId          : use1-az4
CreatedAt                   : 4/7/2023 3:43:53 PM
SubnetId                    : subnet-0123abcd
PreserveClientIp            : False
SecurityGroupIds            : {sg-0123abcd}
Tags                        : {}
IpAddressType               : ipv4
```

作成ステータスをモニタリングするには

State フィールドの初期値は create-in-progress です。このエンドポイントを使用してインスタンスに接続するには、状態が create-complete になるまで待ってください。Get-EC2InstanceConnectEndpoint コマンドレットを使用して EC2 Instance Connect エンドポイントのステータスをモニタリングします。.State.Value は結果を State フィールドにフィルタリングします。

```
(Get-EC2InstanceConnectEndpoint -InstanceConnectEndpointId
 "eice-0123456789example").State.Value
```

以下は出力の例です。

```
create-complete
```

EC2 Instance Connect エンドポイントを変更する

AWS CLI または SDK を使用して既存の EC2 Instance Connect エンドポイントを変更できます。Amazon EC2 コンソールでは、エンドポイントの変更がサポートされていません。

開始する前に、必須の IAM のアクセス許可が必要です。詳細については「EC2 Instance Connect エンドポイントを作成、記述、削除するアクセス許可」を参照してください。

変更できるパラメータ

次の EC2 Instance Connect エンドポイントパラメータを変更できます。

セキュリティグループ

EC2 Instance Connect エンドポイントに新しいセキュリティグループを指定できます。新しいセキュリティグループは現在のセキュリティグループを置き換えます。

セキュリティグループを変更する際、以下の事項を指定する必要があります。

- VPC のデフォルトのセキュリティグループであっても、少なくとも 1 つのセキュリティグループ。
- セキュリティグループの名前ではなく、ID。

IP アドレスタイプ

EC2 Instance Connect エンドポイントに新しい IP アドレスタイプを指定できます。

有効な値: `ipv4` | `dualstack` | `ipv6`

クライアント IP 設定の保持

クライアント IP アドレスをソースとして保持するかどうかを指定できます。

> **Note**
>
> クライアント IP の保存は、IPv4 EC2 Instance Connect エンドポイントでのみサポートされています。PreserveClientIp を有効にする場合、エンドポイントの既存の IP アドレスタイプは ipv4 である必要があります。または同じリクエストで IP アドレスタイプを変更する場合、新しい値は ipv4 である必要があります。

AWS CLI

EC2 Instance Connect エンドポイントを変更する方法

modify-instance-connect-endpoint コマンドを使用し、変更する EC2 Instance Connect エンドポイントおよびパラメータを指定します。次の の例では、1 つのリクエストですべてのパラメータを変更します。

```
aws ec2 modify-instance-connect-endpoint \
    --instance-connect-endpoint-id eice-0123456789example \
    --security-group-ids sg-0123456789example \
    --ip-address-type dualstack \
    --no-preserve-client-ip
```

以下は出力の例です。

```
{
    "Return": true
}
```

更新ステータスをモニタリングする方法

変更中、EC2 Instance Connect エンドポイントのステータスは update-in-progress に変わります。更新プロセスは非同期的に実行され、update-complete または update-failed のステータスで完了します。エンドポイントは、ステータスが update-complete に変わるまで古い設定を使用します。

describe-instance-connect-endpoints コマンドを使用して、更新ステータスをモニタリングします。--query パラメータは、State フィールドに結果をフィルタリングします。

```
aws ec2 describe-instance-connect-endpoints \
    --instance-connect-endpoint-ids eice-0123456789example \
```

```
    --query InstanceConnectEndpoints[*].State --output text
```

以下は出力の例です。

```
update-complete
```

PowerShell

EC2 Instance Connect エンドポイントを変更する方法

Edit-EC2InstanceConnectEndpoint コマンドレットを使用して、変更する EC2 Instance Connect エンドポイントおよびパラメータを指定します。次の の例では、1 つのリクエストですべてのパラメータを変更します。

```
Edit-EC2InstanceConnectEndpoint `
    -InstanceConnectEndpointId eice-0123456789example `
    -SecurityGroupIds sg-0123456789example `
    -IpAddressType dualstack `
    -PreserveClientIp $false
```

以下は出力の例です。

```
True
```

更新ステータスをモニタリングする方法

変更中、EC2 Instance Connect エンドポイントのステータスは update-in-progress に変わります。更新プロセスは非同期的に実行され、update-complete または update-failed のステータスで完了します。エンドポイントは、ステータスが update-complete に変わるまで古い設定を使用します。

Get-EC2InstanceConnectEndpoint コマンドを使用して更新ステータスをモニタリングします。.State.Value は結果を State フィールドにフィルタリングします。

```
(Get-EC2InstanceConnectEndpoint -InstanceConnectEndpointId
 "eice-0123456789example").State.Value
```

以下は出力の例です。

```
update-complete
```

EC2 Instance Connect エンドポイントを削除する

EC2 Instance Connect エンドポイントの使用が終わったら、削除することができます。

EC2 Instance Connect エンドポイントを作成するには、必要な IAM アクセス許可が付与されている必要があります。詳細については「EC2 Instance Connect エンドポイントを作成、記述、削除するアクセス許可」を参照してください。

コンソールを使用して EC2 Instance Connect エンドポイントを削除すると、[削除中] 状態になります。削除が成功すると、削除されたエンドポイントは表示されなくなります。削除に失敗すると、delete-failed 状態になり、[ステータスメッセージ] が失敗の理由を表示します。

AWS CLI を使用して EC2 Instance Connect エンドポイントを削除すると、`delete-in-progress` 状態になります。削除が成功した場合、`delete-complete` 状態となります。削除に失敗すると、`delete-failed` 状態になり、`StateMessage` が失敗の理由を表示します。

Console

EC2 Instance Connect エンドポイントを削除するには

1. Amazon VPC コンソールの https://console.aws.amazon.com/vpc/ を開いてください。
2. 左のナビゲーションペインで [エンドポイント] を選択してください。
3. エンドポイントを選択してください。
4. [アクション 、[VPC エンドポイントを削除] の順に選択してください。
5. 確認を求められたら、**delete** をクリックしてください。
6. [削除] を選択してください。

AWS CLI

EC2 Instance Connect エンドポイントを削除するには

delete-instance-connect-endpoint コマンドを使用し、削除する EC2 Instance Connect エンドポイントの ID を指定します。

```
aws ec2 delete-instance-connect-endpoint --instance-connect-endpoint-
id eice-03f5e49b83924bbc7
```

以下は出力の例です。

```
{
    "InstanceConnectEndpoint": {
        "OwnerId": "111111111111",
        "InstanceConnectEndpointId": "eice-0123456789example",
        "InstanceConnectEndpointArn": "arn:aws:ec2:us-east-1:111111111111:instance-
connect-endpoint/eice-0123456789example",
        "State": "delete-in-progress",
        "StateMessage": "",
        "NetworkInterfaceIds": [],
        "VpcId": "vpc-0123abcd",
        "AvailabilityZone": "us-east-1d",
        "AvailabilityZoneId": "use1-az2",
        "CreatedAt": "2023-02-07T12:05:37+00:00",
        "SubnetId": "subnet-0123abcd"
    }
}
```

PowerShell

EC2 Instance Connect エンドポイントを削除するには

Remove-EC2InstanceConnectEndpoint コマンドレットを使用し、削除する EC2 Instance Connect エンドポイントの ID を指定します。

```
Remove-EC2InstanceConnectEndpoint -InstanceConnectEndpointId eice-03f5e49b83924bbc7
```

以下は出力の例です。

```
@{
    InstanceConnectEndpoint = @{
        OwnerId = "111111111111"
        InstanceConnectEndpointId = "eice-0123456789example"
        InstanceConnectEndpointArn = "arn:aws:ec2:us-east-1:111111111111:instance-
connect-endpoint/eice-0123456789example"
        State = "delete-in-progress"
        StateMessage = ""
        NetworkInterfaceIds = @()
        VpcId = "vpc-0123abcd"
        AvailabilityZone = "us-east-1d"
        AvailabilityZoneId = "use1-az2"
        CreatedAt = "2023-02-07T12:05:37+00:00"
        SubnetId = "subnet-0123abcd"
    }
```

```
}
```

EC2 Instance Connect エンドポイントを使用して Amazon EC2 インスタンスに接続する

EC2 Instance Connect エンドポイントを使用して、SSH または RDP をサポートする Amazon EC2 インスタンスに接続できます。

前提条件

- EC2 Instance Connect エンドポイントに接続するには、必要な IAM アクセス許可が必要です。詳細については「ユーザーに EC2 Instance Connect エンドポイントを使用してインスタンスへ接続するためのアクセス許可」を参照してください。
- EC2 Instance Connect エンドポイントは、次のいずれかの状態である必要があります。
  - 新しいエンドポイントの場合、create-complete
  - 変更中の既存のエンドポイントの場合、update-in-progress、update-complete、または update-failed エンドポイントを変更すると、ステータスが update-complete に変わるまで元の設定が引き続き使用されます。

    VPC に EC2 Instance Connect エンドポイントがない場合は作成できます。詳細については「EC2 Instance Connect エンドポイントの作成」を参照してください。
- EC2 Instance Connect エンドポイントの IP アドレスタイプは、インスタンスの IP アドレスタイプと互換性がある必要があります。エンドポイントの IP アドレスタイプがデュアルスタックの場合、IPv4 アドレスと IPv6 アドレスの両方で機能します。
- (Linux インスタンス) Amazon EC2 コンソールを使用してインスタンスに接続する、または CLI を使用して接続し、EC2 Instance Connect にエフェメラルキーを処理させるには、インスタンスに EC2 Instance Connect がインストールされている必要があります。詳細については「EC2 Instance Connect のインストール」を参照してください。
- インスタンスのセキュリティグループで EC2 Instance Connect エンドポイント からのインバウンド SSH トラフィックが許可されていることを確認します。詳細については「ターゲットインスタンスセキュリティグループのルール」を参照してください。

接続オプション

- Amazon EC2 コンソールを使用した Linux インスタンスへの接続
- SSH を使用した Linux インスタンスへの接続

- AWS CLI を使用してインスタンス ID で Linux インスタンスに接続する
- RDP を使用した Windows インスタンスへの接続
- トラブルシューティング

Amazon EC2 コンソールを使用した Linux インスタンスへの接続

インスタンスへの接続は、次のように Amazon EC2 コンソールのブラウザベースのクライアント) を使用して実行することができます。

Amazon EC2 コンソールを使用して Linux インスタンスに接続するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択してから、[接続] 選択してください。
4. [EC2 Instance Connect] タブを選択してください。
5. [接続タイプ] で、[プライベート IP を使用して接続] を選択します。
6. [プライベート IPv4 アドレス] または [IPv6 アドレス] を選択します。これらのオプションは、インスタンスに割り当てられた IP アドレスに基づいて使用できます。オプションがグレー表示されている場合、インスタンスにはそのタイプの IP アドレスが割り当てられていません。
7. [EC2 Instance Connect エンドポイント] については、EC2 Instance Connect エンドポイント の ID を選択してください。

   > **Note**
   >
   > EC2 Instance Connect エンドポイントは、前のステップで選択した IP アドレスと互換性がある必要があります。エンドポイントの IP アドレスタイプがデュアルスタックの場合、IPv4 アドレスと IPv6 アドレスの両方で機能します。詳細については「EC2 Instance Connect エンドポイントの作成」を参照してください。

8. [ユーザー名] には、インスタンスの起動に使用した AMI が `ec2-user` 以外のユーザー名を使用している場合は、正しいユーザー名を入力してください。
9. [最大トンネル期間 (秒)] に、SSH 接続の最大許容期間を入力してください。

   期間は IAM ポリシーで指定されている `maxTunnelDuration` 条件を満たしている必要があります。IAM ポリシーを更新するためのアクセス許可がない場合は、管理者に連絡してください。

10. [接続]を選択してください。これにより、インスタンスのターミナルウィンドウが開きます。

SSH を使用した Linux インスタンスへの接続

SSH を使用して Linux インスタンスに接続し、open-tunnel コマンドを使用してプライベートトンネルを確立できます。open-tunnel はシングル接続またはマルチ接続モードで使用できます。インスタンス ID、プライベート IPv4 アドレス、または IPv6 アドレスを指定できます。

SSH を使用したインスタンスに接続するための AWS CLI の使用の詳細については、「AWS CLI を使用して接続する」を参照してください。

以下の例では OpenSSH を使用しています。プロキシモードをサポートする他の SSH クライアントを使用できます。

シングル接続

SSH と **open-tunnel** コマンドを使用してインスタンスに 1 つの接続のみを許可するには

ssh と open-tunnel AWS CLI コマンドを次のように使用します。-o プロキシコマンドには、インスタンスへのプライベートトンネルを作成する open-tunnel コマンドが含まれています。

```
ssh -i my-key-pair.pem ec2-user@i-1234567890abcdef0 \
    -o ProxyCommand='aws ec2-instance-connect open-tunnel --instance-
id i-1234567890abcdef0'
```

内容:

- -i — インスタンスの起動に使用されたキーペアを指定します。
- *ec2-user*@*i-1234567890abcdef0* — インスタンスの起動に使用された AMI のユーザー名とインスタンス ID を指定します。IPv6 アドレスを持つインスタンスの場合、インスタンス ID ではなく IPv6 アドレスを指定する必要があります。
- --instance-id — 接続するインスタンスの ID を指定します。または、ユーザーからインスタンス ID を抽出するように %h を指定します。IPv6 アドレスを持つインスタンスの場合、--instance-id *i-1234567890abcdef0* を --private-ip-address *2001:db8::1234:5678:1.2.3.4* に置き換えます。

マルチ接続

1 つのインスタンスに複数の接続を許可するには、最初に open-tunnel AWS CLI コマンドを実行して新しい TCP 接続のリスニングを開始し、次に ssh を使用して新しい TCP 接続とインスタンスへのプライベートトンネルを作成します。

SSH と **open-tunnel** コマンドを使用してインスタンスへの複数の接続を許可するには

1. ローカルマシンの特定のポートで新しい TCP 接続のリスニングを開始するには、次のコマンドを実行します。

```
aws ec2-instance-connect open-tunnel \
    --instance-id i-1234567890abcdef0 \
    --local-port 8888
```

正常な出力:

```
Listening for connections on port 8888.
```

2. 新しいターミナルウィンドウで、次の ssh コマンドを実行して、新しい TCP 接続とインスタンスへのプライベートトンネルを作成します。

```
ssh -i my-key-pair.pem ec2-user@localhost -p 8888
```

期待される出力 – 最初のターミナルウィンドウには、以下が表示されます。

```
[1] Accepted new tcp connection, opening websocket tunnel.
```

以下が表示される可能性があります。

```
[1] Closing tcp connection.
```

AWS CLI を使用してインスタンス ID で Linux インスタンスに接続する

インスタンス ID のみを知っている場合、ec2-instance-connect ssh の AWS CLI コマンドを使用して SSH クライアントでインスタンスに接続できます。詳細については「AWS CLI を使用して接続する」を参照してください。

前提条件

- AWS CLI バージョン 2 をインストールし、認証情報を使用して構成します。詳細については、AWS Command Line Interfaceユーザーガイドの「AWS CLI の最新バージョンのインストールまたは更新」および「AWS CLI の設定」を参照してください。
- あるいは、AWS CloudShellを開き、事前認証されたシェルで AWS CLI コマンドを実行することもできます。

インスタンス ID と EC2 Instance Connect エンドポイントを使用してインスタンスに接続するには

インスタンス ID のみを知っている場合、ec2-instance-connect ssh の CLI コマンドを使用し、ssh コマンド、インスタンス ID、および eice 値を指定した --connection-type パラメータを指定して、EC2 Instance Connect エンドポイントを使用します。インスタンスに IPv6 アドレスのみがある場合は、IPv6 アドレスとともに --instance-ip パラメータも含める必要があります。

- インスタンスにプライベート IPv4 アドレスがある場合 (IPv6 アドレスも可能)、次のコマンドとパラメータを使用します。

```
aws ec2-instance-connect ssh \
    --instance-id i-1234567890example \
    --os-user ec2-user \
    --connection-type eice
```

- インスタンスに IPv6 アドレスのみがある場合は、IPv6 アドレスとともに --instance-ip パラメータを含めます。

```
aws ec2-instance-connect ssh \
    --instance-id i-1234567890example \
    --instance-ip 2001:db8::1234:5678:1.2.3.4 \
    --os-user ec2-user \
    --connection-type eice
```

> **Tip**
>
> エラーが発生した場合は、AWS CLI バージョン 2 を使用していることを確認してください。ssh パラメータは、AWS CLI バージョン 2 でのみ使用できます。詳細については、

「AWS Command Line Interface ユーザーガイド」の「AWS CLI バージョン 2 について」を参照してください。

RDP を使用した Windows インスタンスへの接続

EC2 Instance Connectエンドポイント 経由でリモートデスクトッププロトコル (RDP) を使用して、パブリック IPv4 アドレスまたはパブリック DNS 名を使用しなくても Windows インスタンスに接続できます。

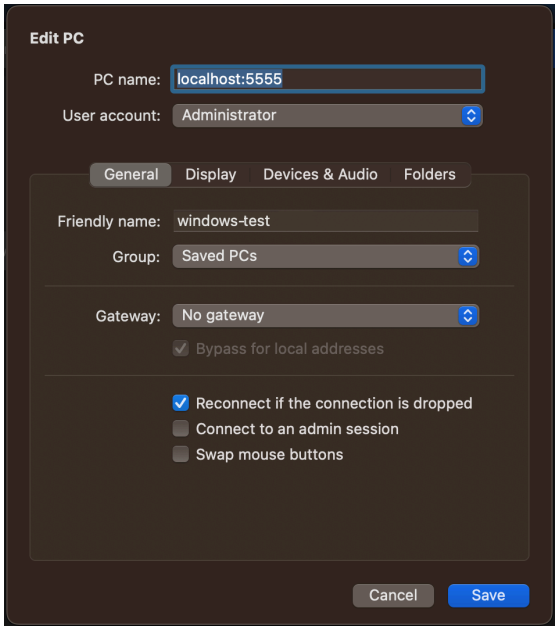
RDP クライアントを使用して Windows インスタンスに接続するには

1. 「RDP を使用した Windows インスタンスへの接続」のステップ 1～8 を実行します。ステップ 8 で RDP デスクトップファイルをダウンロードすると、[接続できません] というメッセージが表示されます。これは、インスタンスにパブリック IP アドレスがないためです。
2. 次のコマンドを実行して、インスタンスがある VPC へのプライベートトンネルを確立します。RDP がデフォルトでポート 3389 を使用しているため、`--remote-port` は 3389 になります。

```
aws ec2-instance-connect open-tunnel \
    --instance-id i-1234567890abcdef0 \
    --remote-port 3389 \
    --local-port any-port
```

3. [ダウンロード] フォルダで、ダウンロードした RDP デスクトップファイルを検索して、RDP クライアントウィンドウにドラッグします。
4. RDP デスクトップファイルを右クリックし、[編集] を選択してください。
5. [PC編集] ウィンドウの [PC名] (接続するインスタンス) に `localhost:`*`local-port`* と入力してください。ここで、*`local-port`* はステップ 2 で指定したものと同じ値を使用します。その後、[保存] をクリックしてください。

   次の [PC の編集] ウィンドウのスクリーンショットは、Mac 上の Microsoft リモートデスクトップのものであることに注意してください。Windows クライアントを使用している場合は、ウィンドウが異なる場合があります。



6. RDP クライアントで、(先ほど設定した) PC を右クリックし、[接続] を選択してインスタンスに接続します。
7. プロンプトに従って、管理者アカウントの復号化されたパスワードを入力してください。

トラブルシューティング

以下の情報は、EC2 Instance Connect エンドポイントを使用してインスタンスを接続するときに発生する可能性のある問題の診断と修復に役立ちます。

インスタンスに接続できない

インスタンスに接続できない一般的な理由は次のとおりです。

- セキュリティ グループ – EC2 Instance Connect エンドポイントとインスタンスに割り当てられたセキュリティグループを確認してください。必要なセキュリティグループルールの詳細については、「EC2 Instance Connect エンドポイントのセキュリティグループ」を参照してください。
- インスタンスの状態 – インスタンスの状態が [running] であることを確認します。
- キーペア — 接続に使用しているコマンドにプライベートキーが必要な場合は、インスタンスにパブリックキーと、対応するプライベートキーがあることを確認します。
- IAM アクセス許可 — 必要な IAM アクセス許可があることを確認します。詳細については「EC2 Instance Connect エンドポイント を使用するためのアクセス許可の付与」を参照してください。

Linux インスタンスのトラブルシューティングのさらなるヒントについては、「Amazon EC2 Linux インスタンスへの接続に関する問題のトラブルシューティング」を参照してください。Windows インスタンスのトラブルシューティングのヒントについては、「the section called “Windows インスタンスの RDP 問題”」を参照してください。

ErrorCode: AccessDeniedException

`AccessDeniedException` エラーが発生し、`maxTunnelDuration` 条件が IAM ポリシーで指定されている場合は、インスタンスに接続するときに必ず `--max-tunnel-duration` パラメータを指定してください。このパラメータの詳細については、「AWS CLI コマンドリファレンス」の「open-tunnel」を参照してください。

EC2 Instance Connect エンドポイントを介して確立された接続のログの作成

リソース操作のログを作成し、AWS CloudTrail ログを使用して EC2 Instance Connect エンドポイントを介して確立された接続を監査できます。

Amazon EC2 での AWS CloudTrail の使用に関する詳細は、「AWS CloudTrail を使用して Amazon EC2 API コールをログに記録する」を参照してください。

AWS CloudTrail を使用した EC2 Instance Connect Endpoint API コールのログの作成

EC2 Instance Connect エンドポイントのリソース操作は、管理イベントとして CloudTrail にログが作成されます。次の API コールが行われると、アクティビティは CloudTrail イベントとして [イベント履歴] にログが作成されます。

- `CreateInstanceConnectEndpoint`
- `DescribeInstanceConnectEndpoints`
- `DeleteInstanceConnectEndpoint`

最近のイベントは、AWS アカウント で表示、検索、ダウンロードできます。詳細については、「AWS CloudTrail ユーザーガイド」の「CloudTrail イベント履歴でのイベントの表示」を参照してください。

EC2 Instance Connect エンドポイントを使用してインスタンスに接続するユーザーを監査するための AWS CloudTrail の使用

EC2 Instance Connect エンドポイントを経由してインスタンスに接続を試みると、CloudTrail の [イベント履歴] にログが作成されます。インスタンスへの接続が EC2 Instance Connect エンドポイン

トを経由して開始されると、その接続は OpenTunnel の eventName という CloudTrail 管理イベントとしてログが作成されます。

CloudTrail イベントをターゲットにルーティングする Amazon EventBridge ルールを作成できます。詳細については、「Amazon EventBridge ユーザーガイド」を参照してください。

CloudTrail にログが作成された OpenTunnel 管理イベントの例を以下に示します。

```
{
     "eventVersion": "1.08",
     "userIdentity": {
         "type": "IAMUser",
         "principalId": "ABCDEFGONGNOMOOCB6XYTQEXAMPLE",
         "arn": "arn:aws:iam::1234567890120:user/IAM-friendly-name",
         "accountId": "123456789012",
         "accessKeyId": "ABCDEFGUKZHNAW4OSN2AEXAMPLE",
         "userName": "IAM-friendly-name"
     },
     "eventTime": "2023-04-11T23:50:40Z",
     "eventSource": "ec2-instance-connect.amazonaws.com",
     "eventName": "OpenTunnel",
     "awsRegion": "us-east-1",
     "sourceIPAddress": "1.2.3.4",
     "userAgent": "aws-cli/1.15.61 Python/2.7.10 Darwin/16.7.0 botocore/1.10.60",
     "requestParameters": {
         "instanceConnectEndpointId": "eici-0123456789EXAMPLE",
         "maxTunnelDuration": "3600",
         "remotePort": "22",
         "privateIpAddress": "10.0.1.1"
     },
     "responseElements": null,
     "requestID": "98deb2c6-3b3a-437c-a680-03c4207b6650",
     "eventID": "bbba272c-8777-43ad-91f6-c4ab1c7f96fd",
     "readOnly": false,
     "resources": [{
         "accountId": "123456789012",
         "type": "AWS::EC2::InstanceConnectEndpoint",
         "ARN": "arn:aws:ec2:us-east-1:123456789012:instance-connect-endpoint/
eici-0123456789EXAMPLE"
     }],
     "eventType": "AwsApiCall",
     "managementEvent": true,
     "recipientAccountId": "123456789012",
```

```
    "eventCategory": "Management"
}
```

EC2 Instance Connect エンドポイントのサービスにリンクされたロール

Amazon EC2 は、AWS Identity and Access Management (IAM) サービスにリンクされたロールを使用します。サービスにリンクされたロールは、Amazon EC2 に直接リンクされた一意のタイプの IAM ロールです。サービスにリンクされたロールは Amazon EC2 によって事前に定義されており、Amazon EC2 がユーザーに代わって他の AWS のサービスを呼び出すために必要なすべてのアクセス許可がロールには含まれています。詳細については「IAM ユーザーガイド」の「Service-linked roles」を参照してください。

EC2 Instance Connect エンドポイントのサービスリンクロールアクセス許可

Amazon EC2 は、AWSServiceRoleForEC2InstanceConnect を使用して、EC2 Instance Connect エンドポイントが必要とする、アカウント内のネットワークインターフェイスを作成し、管理します。

AWSServiceRoleForEC2InstanceConnect サービスにリンクされたロールは、以下のサービスを信頼してロールを引き受けます。

- `ec2-instance-connect.amazonaws.com`

AWSServiceRoleForEC2InstanceConnect サービスリンクロールは、次の管理ポリシーを使用します。

- Ec2InstanceConnectEndpoint

管理ポリシーのアクセス許可を確認するには、AWS 管理されたポリシーリファレンスの「Ec2InstanceConnectEndpoint」を参照してください。

EC2 Instance Connect エンドポイントのサービスリンクロールを作成する

サービスにリンクされたこのロールを手動で作成する必要はありません。EC2 Instance Connect エンドポイントを作成する際、Amazon EC2 によってサービスリンクロールが作成されます。

EC2 Instance Connect エンドポイントのサービスリンクロールの編集

EC2 Instance Connect エンドポイントでは、AWSServiceRoleForEC2InstanceConnect サービスにリンクされたロールを編集することはできません。

EC2 Instance Connect エンドポイントのサービスリンクロールを削除する

EC2 Instance Connect エンドポイントを使用する必要がなくなった場合は、AWSServiceRoleForEC2InstanceConnect サービスにリンクされたロールを削除することをお勧めします。

サービスリンクロールを削除する前に、すべての EC2 Instance Connect エンドポイントのリソースを削除する必要があります。

サービスリンクロールの削除については、IAM ユーザーガイドの.サービスリンクロールの削除 を参照してください。

サービスリンク役割の作成、編集、削除を IAM エンティティ (ユーザー、グループ、役割) に許可するにはアクセス許可を設定する必要があります。詳細についてはIAM ユーザーガイド の「サービスにリンクされた役割のアクセス許可」を参照してください。

EC2 Instance Connect エンドポイントのクォータ

AWS アカウント には、AWS のサービスごとにデフォルトのクォータ (以前は制限と呼ばれていました) があります。特に明記していない限り、クォータはリージョン固有です。

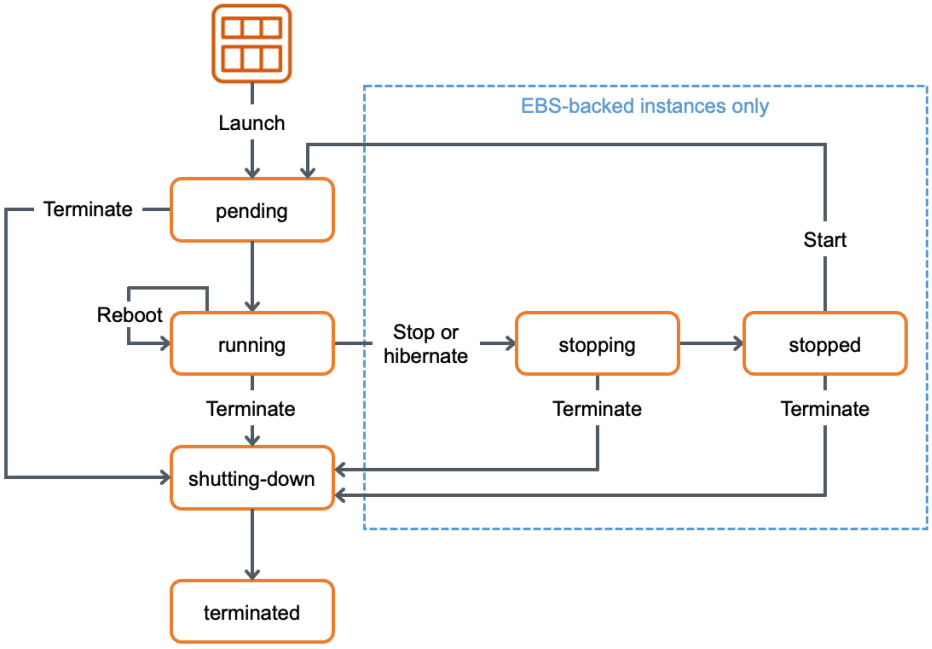
AWS アカウントには、EC2 Instance Connect エンドポイントに関連する、次のクォータがあります。

名前	デフォルト	引き上げ可能
AWS アカウント ごとの AWS リージョン あたりの EC2 Instance Connect エンドポイントの最大数	5	いいえ
VPC あたりの EC2 Instance Connect エンドポイントの最大数	1	いいえ
サブネットあたりの EC2 Instance Connect エンドポイントの最大数	1	いいえ
EC2 Instance Connect エンドポイントあたりの同時接続の最大数	20	いいえ

Amazon EC2 インスタンスの状態変更

Amazon EC2 インスタンスは、起動時から終了まで、さまざまな状態に移行します。

次の図は、インスタンス状態の遷移を示しています。



インスタンスの状態が変更されたときに通知を受け取ることができます。詳細については、「the section called “状態変更イベント”」を参照してください。

インスタンスの状態別の請求

次の表は、各インスタンスの状態の概要と、インスタンスの使用量の請求有無を示しています。Amazon EBS ボリュームや Elastic IP アドレスなど一部の AWS リソースでは、インスタンスの状態に関係なく利用料金が発生します。詳細については、AWS Billing ユーザーガイドの予想外の料金の回避を参照してください。

インスタンスの状態	説明	インスタンス使用の請求
pending	インスタンスは running 状態への移行準備中です。インスタンスは、起動時または stopped 状	非請求対象

インスタンスの状態	説明	インスタンス使用の請求
	態になってから起動すると、pending 状態になります。	
running	インスタンスは実行中で、使用できる状態です。	請求済み付
stoppir	インスタンスは停止の準備中です。	非請求対象 **Note** インスタンスを休止すると、インスタンスが stopping 状態にある間は課金されます。
stopped	インスタンスはシャットダウンされているため、使用できません。インスタンスはいつでも起動できます。	課金されない
shuttir down	インスタンスは削除準備中です。	課金されない
termina d	インスタンスは完全に削除されているため、起動することはできません。	課金されない **Note** 終了したインスタンスに適用されるリザーブドインスタンスは、支払いオプションに従って、契約期間末まで請求が発生します。詳細については、Amazon EC2 のリザーブドインスタンスの概要を参照してください。

保留中のインスタンス

インスタンスを起動すると、インスタンスはpending状態に移行します。起動時に指定したインスタンスタイプによって、インスタンスのホストコンピュータのハードウェアが決定します。起動時に指定されたAmazon Machine Image (AMI) を使って、インスタンスを再作成します。インスタンスの準備ができると、running 状態へ移行します。実行中のインスタンスに接続して、自分の前にあるコンピュータと同じように使用することができます。

インスタンスが running 状態に遷移するとすぐに、インスタンスの実行時間に応じて (インスタンスがアイドル状態のままで、接続されていなくても) 最低 1 分以上の秒単位で使用料金が発生します。

停止したインスタンス

インスタンスのステータスチェックに失敗するか、インスタンスでアプリケーションが想定通りに動作しておらず、インスタンスのルートボリュームが Amazon EBS である場合、インスタンスの停止と起動を行い、問題が解決するか試してみることができます。

インスタンスを停止した場合、インスタンスはstopping状態に移行してから、stopped状態になります。インスタンスの使用料やデータ転送料金は、stopped 時点では請求されません。どの Amazon EBS ボリュームのストレージにも料金が発生します。インスタンスがstopped状態の間、インスタンスタイプなど、インスタンスの特定の属性を変更できます。

インスタンスを起動して pending 状態になると、インスタンスは新しいホストコンピュータに移動します (ただし、場合によっては、インスタンスは現在のホストに残ることもあります)。インスタンスの停止と起動を行うと、以前のホストコンピュータに接続されていたインスタンスストアボリュームのすべてのデータが失われます。

インスタンスはプライベートIPv 4アドレスを保持します。つまり、プライベートIPv 4アドレスまたはネットワークインターフェイスに関連付けられた Elastic IPアドレスは、インスタンスに関連付けられたままになります。インスタンスに IPv6 アドレスがある場合、IPv6 アドレスは保持されます。

stopped から running に移行したインスタンスについては、その実行中に秒単位の料金が発生します。また、インスタンスの起動時には、1 分間分の最低料金が課金されます。

インスタンスの停止と開始の詳細については、「Amazon EC2 インスタンスの停止と開始」を参照してください。

休止したインスタンス

インスタンスを休止すると、オペレーティングシステムに休止を実行するように合図します (ディスクの停止)。これにより、内容がインスタンスのメモリ (RAM) から Amazon EBS ルートボリュームに保存されます。インスタンスの Amazon EBS ルートボリュームとアタッチされた Amazon EBS データボリュームは保持されます。インスタンスを起動すると、Amazon EBS ルートボリュームは以前の状態に復元され、RAM の内容が再ロードされます。以前にアタッチされたデータボリュームは再アタッチされ、インスタンスはそのインスタンス ID を保持します。

インスタンスを休止した場合、インスタンスは `stopping` 状態に移行してから、`stopped` 状態になります。休止状態にあるインスタンスが `stopped` 状態にある間は料金は課金されませんが、(休止せずにインスタンスを停止した場合とは異なり) `stopping` 状態にある間は料金が発生します。データ転送料金に対して使用料を課金しませんが、RAM データのストレージを含め、Amazon EBS ボリュームのストレージに対しては課金します。

休止状態のインスタンスを起動して `pending` 状態になると、インスタンスは新しいホストコンピュータに移動されます (ただし、場合によっては、インスタンスが現在のホストに残ることもあります)。

プライベート IPv4 アドレスは保持されます。つまり、プライベート IPv4 アドレスまたはネットワークインターフェイスに関連付けられていた Elastic IP アドレスは、インスタンスとの関連付けが継続されるということです。インスタンスに IPv6 アドレスがある場合、IPv6 アドレスは保持されます。

詳細については、「Amazon EC2 インスタンスの休止」を参照してください。

インスタンスの再起動

Amazon EC2 コンソール、コマンドラインツール、Amazon EC2 API を使って、インスタンスを再起動できます。インスタンスからオペレーティングシステムの再起動コマンドを実行する代わりに、Amazon EC2 を使ってインスタンスを再起動することをお勧めします。

インスタンスの再起動は、オペレーティングシステムの再起動と同等です。インスタンスは同じホストコンピュータに残り、そのパブリック DNS 名、プライベート IP アドレス、およびその他のデータをインスタンスストアボリュームに維持します。通常、再起動が完了するまでに数分かかりますが、再起動に必要な時間は、インスタンスの設定によって異なります。

インスタンスを再起動しても、新しいインスタンスの課金時間は開始されず、最低 1 分間分の料金はなしで秒単位の課金が継続します。

詳細については、「Amazon EC2 インスタンスを再起動する」を参照してください。

終了したインスタンス

インスタンスが必要なくなったら、削除することができます。インスタンスのステータスが shutting-down または terminated に変わったら、そのインスタンスへの課金は停止します。

停止保護が有効な場合、コンソール、CLI、または API を使用してインスタンスを削除することはできません。

インスタンスの削除後、インスタンスはしばらくの間コンソールに表示されたままですが、エントリは自動的に削除されます。CLI および API を使って、削除したインスタンスを記述することもできます。(タグなどの) リソースは削除されたインスタンスから徐々に関連付けが解除されるため、しばらくすると、削除されたインスタンスで表示されなくなる可能性があります。削除したインスタンスへの接続や復旧はできません。

Amazon EBS-Backed インスタンスはそれぞれ、インスタンス自体からシャットダウンを開始したとき (Linux で shutdown コマンドを使用してなど)、インスタンスを停止するか終了するかを制御する InstanceInitiatedShutdownBehavior 属性をサポートしています。デフォルトの動作は、インスタンスの停止です。インスタンスの実行中または停止中に、この属性の設定を変更できます。

各 Amazon EBS ボリュームは DeleteOnTermination 属性をサポートします。この属性は、アタッチされたインスタンスを終了するときに、ボリュームの削除や保持を制御します。デフォルトでは、ルートボリュームを削除し、それ以外に EBS ボリュームがあれば保持します。

詳細については、「Amazon EC2 インスタンスを終了する」を参照してください。

インスタンスの状態の違い

次の表に、インスタンスの再起動、停止、休止、終了の主な違いをまとめました。

特徴	再起動	停止/開始 (Amazon EBS-Backed インスタンスのみ)	休止 (Amazon EBS Backed インスタンスのみ)	終了
ホストコンピュータ	インスタンスは、同じホストコンピュータで保持される	インスタンスは新しいホストコンピュータに移動されます (ただし、場合によっては、インスタン	インスタンスは新しいホストコンピュータに移動されます (ただし、場合によっては、インスタン	なし

特徴	再起動	停止/開始 (Amazon EBS-Backed インスタンスのみ)	休止 (Amazon EBS Backed インスタンスのみ)	終了
		スが現在のホストに残ることもあります)。	スが現在のホストに残ることもあります)。	
プライベート IPv4 アドレス	インスタンスはプライベート IPv4 アドレスを保持します。	インスタンスはプライベート IPv4 アドレスを保持します。	インスタンスはプライベート IPv4 アドレスを保持します。	なし
パブリック IPv4 アドレス	インスタンスは、IPv4 アドレスを保持する	インスタンスは新しいパブリック IPv4 アドレスを取得します。ただし、セカンダリネットワークインターフェースがある場合、または Elastic IP アドレスに関連付けられたセカンダリプライベート IPv4 アドレスがある場合は除きます。	インスタンスは新しいパブリック IPv4 アドレスを取得します。ただし、セカンダリネットワークインターフェースがある場合、または Elastic IP アドレスに関連付けられたセカンダリプライベート IPv4 アドレスがある場合は除きます。	なし
Elastic IP アドレス (IPv4)	Elastic IP アドレスは、インスタンスに関連付けられたまま維持される	Elastic IP アドレスは、インスタンスに関連付けられたまま維持される	Elastic IP アドレスは、インスタンスに関連付けられたまま維持される	Elastic IP アドレスはインスタンスの関連付けが解除される
IPv6 アドレス	インスタンスは、IPv6 アドレスを保持する	インスタンスは、IPv6 アドレスを保持する	インスタンスは、IPv6 アドレスを保持する	なし

特徴	再起動	停止/開始 (Amazon EBS-Backed インスタンスのみ)	休止 (Amazon EBS Backed インスタンスのみ)	終了
インスタンスストアボリューム	データは保持される	データは消去される	データは消去される	データは消去される
ルートボリューム	ボリュームは保持される	ボリュームは保持される	ボリュームは保持される	ボリュームはデフォルトで削除される
RAM (メモリの内容)	RAM は消去される	RAM は消去される	RAM はルートボリュームにあるファイルに保存される	RAM は消去される
請求	インスタンスの課金時間は変更されません。	インスタンスの状態が stopping に変わるとすぐに、そのインスタンスへの課金が停止されます。インスタンスが stopped 状態から running 状態に移行するたびに新しいインスタンスの課金時間が開始され、インスタンスの開始時には 1 分間の最低料金が発生します。	インスタンスが stopping 状態にある間は課金されますが、そのインスタンスが stopped 状態にある場合、課金は停止します。インスタンスが stopped 状態から running 状態に移行するたびに新しいインスタンスの課金時間が開始され、インスタンスの開始時には 1 分間の最低料金が発生します。	インスタンスの状態が shutting-down に変わるとすぐに、そのインスタンスに対して課金されなくなります。

オペレーティングシステムのシャットダウンコマンドを実行すると、インスタンスストアのルートボリュームを持つインスタンスは必ず停止されます。オペレーティングシステムのシャットダウンコマンドによって EBS ルートボリュームを持つインスタンスを停止または終了するかどうかを制御でき

ます。詳細については、「インスタンスによるシャットダウン動作を変更する」を参照してください。

Amazon EC2 インスタンスの停止と開始

インスタンスにルートボリュームとして Amazon EBS ボリュームがある場合、そのインスタンスを停止して開始できます。ユーザーがインスタンスを停止すると、インスタンスはシャットダウンされます。インスタンスを開始すると、通常、インスタンスは基盤となる新しいホストコンピュータに移行され、新しいパブリック IPv4 アドレスが割り当てられます。

インスタンスの停止は、ユーザーが開始する (インスタンスを手動で停止する場合) ことも、AWS が開始する (インスタンスの基盤となるホストの回復不可能な障害が AWS によって検出されたときに発生するスケジュールされた停止イベントに応じて) こともできます。

ユーザーが開始する停止の場合は、インスタンスからオペレーティングシステムの停止コマンドを実行するのではなく、Amazon EC2 コンソール、CLI、または API を使用することをお勧めします。Amazon EC2 を使用する場合、インスタンスが数分以内に正常にシャットダウンされない場合、Amazon EC2 はハードシャットダウンを実行します。さらに、AWS CloudTrail はインスタンスが停止したたときの API レコードを作成します。

このトピックでは、ユーザーが開始する停止を実行する方法について説明します。AWS によって実行される停止の詳細については、「停止またはリタイアがスケジュールされている Amazon EC2 インスタンスを管理する」を参照してください。

インスタンスを停止しても、そのインスタンスは削除されません。インスタンスが必要なくなったら、終了することができます。詳細については「Amazon EC2 インスタンスを終了する」を参照してください。インスタンスを休止状態にしてインスタンスメモリ (RAM) の内容を保存する場合はを参照してください。Amazon EC2 インスタンスの休止インスタンスライフサイクルアクションの違いについてはを参照してください。インスタンスの状態の違い

内容

- EC2 インスタンスの停止および開始の動作
- インスタンスを停止するメソッド
- インスタンスを手動で停止して開始する
- インスタンスを自動的に停止して開始する
- 実行中および停止中のインスタンスをすべて検索
- 初期と最新の起動時間を確認

- EC2 インスタンスに対する停止保護を有効にする

EC2 インスタンスの停止および開始の動作

Amazon EC2 インスタンスを停止すると、変更はインスタンスのオペレーティングシステム (OS) レベルで登録され、一部のリソースは失われますが、一部のリソースは存続します。インスタンスを起動すると、変更はインスタンスレベルで登録されます。

トピック

- インスタンスを停止するとどうなるか
- インスタンスを起動するとどうなるか
- アプリケーションの応答をテストして停止して起動する
- インスタンスの起動と停止に関連するコスト

インスタンスを停止するとどうなるか

デフォルトの停止方法を使用してインスタンスを停止したときの一般的な動作については、次のとおりに示されます。使用する停止メソッドによっては、一部動作が異なる場合がありますのでご注意ください。

OS レベルでの変更の登録

- API リクエストはボタンのクリックイベントをゲストに送信します。
- ボタンのクリックイベントの結果として、さまざまなシステムサービスが停止します。正常な OS シャットダウンは、ハイパーバイザーによる ACPI シャットダウンボタンのクリックイベントによってトリガーされます。
- ACPI シャットダウンが開始されます。
- このインスタンスは、正常な OS シャットダウンプロセスが終了したときにシャットダウンされます。設定可能な OS シャットダウン時間はありません。
- インスタンス OS が数分以内に正常にシャットダウンされない場合はハードシャットダウンが実行されます。
- インスタンスが実行を停止します。
- インスタンスのステータスが `stopping` に変更されたら、`stopped` になります。
- [自動スケーリング] インスタンスが `running` 以外の Amazon EC2 状態のとき、インスタンスが Auto Scaling グループにあるか、ステータスチェックのステータスが `impaired` になった場

合、Amazon EC2 Auto Scaling はインスタンスを異常と見なして置き換えます。詳細についてはAmazon EC2 Auto Scaling ユーザーガイドの「Auto Scaling グループ内のインスタンスのヘルスチェック」を参照してください。

- [Windows インスタンス] Windows インスタンスを停止および起動した場合、起動エージェントはアタッチされている Amazon EBS ボリュームのドライブ文字を変更するなど、インスタンスでタスクを実行します。これらのデフォルトおよび変更方法については「the section called "EC2Launch v2"」を参照してください。

失われるリソース

- RAM に保存されているデータ。
- インスタンスストアボリュームに保存されているデータは失われます。
- 起動時または開始時に Amazon EC2 がインスタンスに自動的に割り当てられたパブリック IPv4 アドレス。パブリック IPv4 アドレスに変更を加えないようにするにはインスタンスに Elastic IP アドレスを関連付けます。

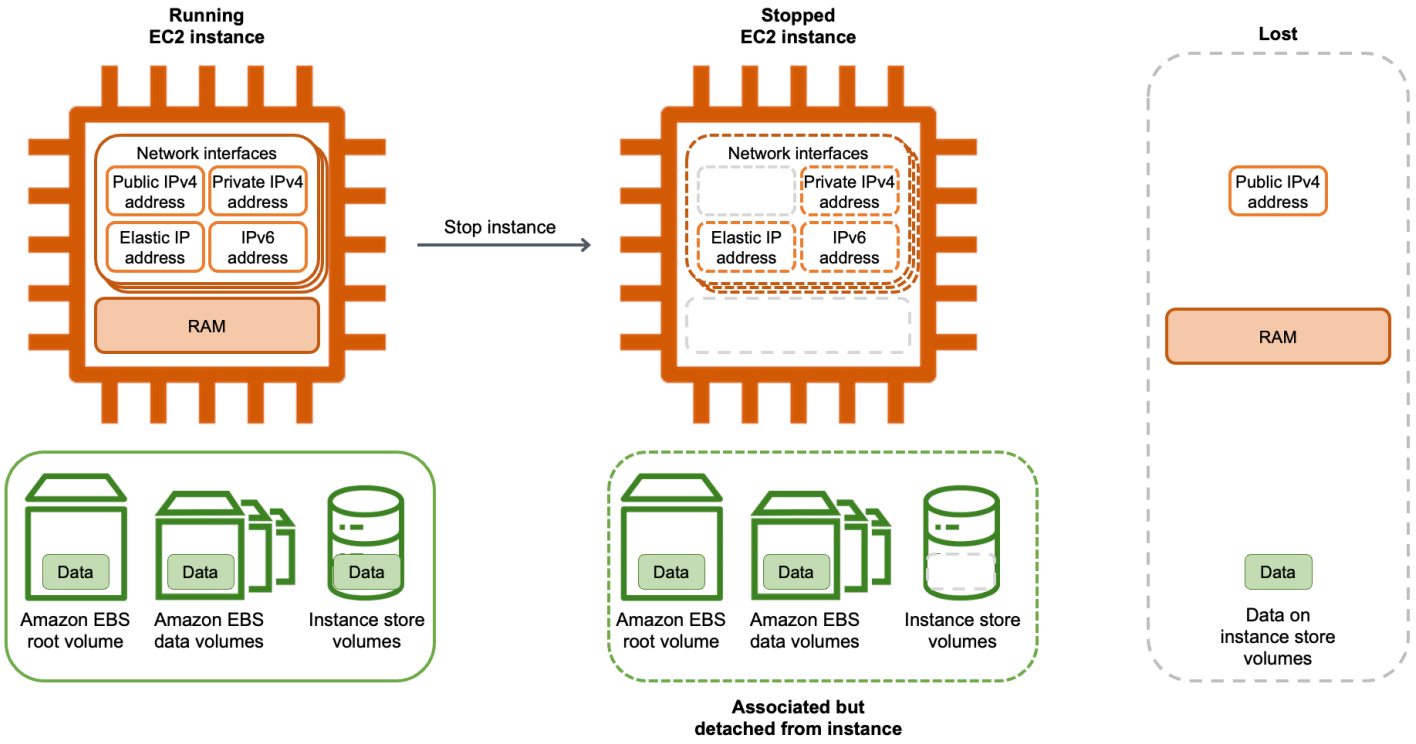
存続するリソース

- アタッチされた Amazon EBS ボリュームとデータボリューム。
- Amazon EBS ボリュームに保存されているデータ。
- アタッチされたネットワークインターフェイス。

  ネットワークインターフェイスには次のリソースが含まれており、これらも保持されます。

  - プライベート IPv4 アドレス。
  - IPv6 アドレス。
  - インスタンスに関連付けられた Elastic IP アドレス。インスタンスが停止すると、関連する Elastic IP アドレスに対する課金が始まります。

次の図は、EC2 インスタンスを停止したときに保持されるものと失われるものとを示しています。図は 3 つの要素からなります。1 つ目の実行中の EC2 インスタンスでは、`running` 状態のインスタンスとそのリソースが描かれています。2 つ目の停止した EC2 インスタンスでは、`stopped` 状態のインスタンスと、そこで保持されているリソースが描かれています。3 つ目の紛失では、インスタンスが停止したときに失われたリソースが描かれています。



Mac インスタンスを停止したときに起きることについては「Amazon EC2 Mac インスタンスを停止または終了する」を参照してください。

インスタンスを起動するとどうなるか

- ほとんどの場合、基盤となる新しいホストコンピュータにインスタンスが移行します (ただし、専有ホスト設定でインスタンスがホストに割り当てられた場合などは現在のホストにインスタンスが残ります)。
- 関連付けられた EBS ボリュームおよびネットワークインターフェイスがインスタンスに再アタッチされます。
- インスタンスがパブリック IPv4 アドレスを受信するように設定されている場合、Amazon EC2 は新しいパブリック IPv4 アドレスをインスタンスに割り当てます。ただし、Elastic IP アドレスに関連付けられたセカンダリネットワークインターフェイスまたはセカンダリプライベート IPv4 アドレスがある場合は割り当てしません。
- プレイスメントグループ内のインスタンスを停止して再起動しても、そのインスタンスは同じプレイスメントグループ内で実行されます。ただし、インスタンスに対して十分な容量がない場合、起動は失敗します。既にインスタンスを実行中のプレイスメントグループ内のインスタンスを起動するときにキャパシティエラーを受信した場合は、プレイスメントグループ内のすべてのインスタンスを停止して、もう一度起動してください。インスタンスを起動すると、すべてのリクエストしたインスタンスに応じた容量があるハードウェアにインスタンスが移行される場合があります。

アプリケーションの応答をテストして停止して起動する

インスタンスが停止後に起動された場合のアプリケーションの応答をテストするにはAWS Fault Injection Service を使用します。詳細についてはAWS Fault Injection Service ユーザーガイドを参照してください。

インスタンスの起動と停止に関連するコスト

インスタンスの停止と起動には以下のコストがかかります。

停止 – インスタンスの状態が `shutting-down` または `terminated` に変わると、そのインスタンスの料金は発生しなくなります。停止したインスタンスの使用料やデータ転送料は請求されません。Amazon EBS ストレージボリュームの保存には料金がかかります。

起動 – 停止したインスタンスを再起動するたびに、1 分間分の最低料金が課金されます。1 分経過した後は使用した秒数のみ課金されます。例えば、インスタンスを 20 秒間実行して停止した場合は1分間分課金されます。インスタンスを 3 分 40 秒実行した場合はちょうど 3 分 40 秒間分課金されます。

インスタンスを停止するメソッド

ユーザー主導の停止を実行するには、デフォルトの停止、OS シャットダウンをスキップした停止、強制停止、OS シャットダウンをスキップした強制停止の 4 つの方法があります。次のテーブルでは、停止メソッドの主な違いが比較されます。

停止メソッド	主な目的	ユースケース	CLI コマンド
デフォルトの停止	正常な OS シャットダウンの試行による通常のインスタンスシャットダウン。	一般的なインスタンスの停止。	`aws ec2 stop-instances \ --instance-id i-1234567890abcdef0`
OS シャットダウンをスキップ	インスタンスを停止するときに正常な OS シャットダウンをバイパスします。	正常な OS シャットダウンをバイパスする必要がある場合。	`aws ec2 stop-instances \ --instance-id i-1234567`

停止メソッド	主な目的	ユースケース	CLI コマンド
して停止する			890abcdef0 \ --skip-os-shutdown
強制停止する	スタックしたインスタンスを処理します。最初にデフォルトの停止を試みます。インスタンスが停止に失敗した場合、インスタンスを強制的に停止します。	インスタンスが stopping 状態でスタックしている場合。	aws ec2 stop-instances \ --instance-id i-1234567890abcdef0 \ --force
OS シャットダウンをスキップして強制停止する	インスタンスを停止する際、正常な OS シャットダウンが強制停止されてバイパスされます。	正常な OS シャットダウンの強制停止およびバイパスする必要がある場合。	aws ec2 stop-instances \ --instance-id i-1234567890abcdef0 \ --force \ --skip-os-shutdown

各メソッドの使用方法の手順については、次の内容を参照してください。

- 正常な OS シャットダウンでインスタンスを停止する
- インスタンスを停止して正常な OS シャットダウンをバイパスする
- インスタンスを強制停止する

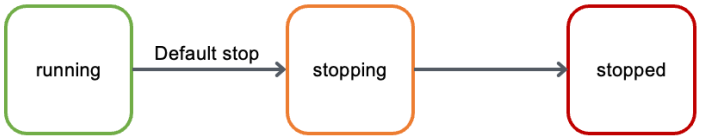
停止メソッド:

- デフォルトの停止
- OS シャットダウンをスキップして停止する
- 強制停止する
- OS シャットダウンをスキップして強制停止する

次のセクションでは、ユーザー主導の 4 つの異なる停止メソッドに関する詳細情報が記載されています。

デフォルトの停止

デフォルトの停止メソッドは、インスタンスを停止する標準的な方法です。StopInstances コマンドを発行すると、次の図で示されるように、インスタンスは `running` 状態から `stopping` 状態に移行したら、最後に `stopped` 状態に移行します。



目的: 正常な OS シャットダウンの試行による通常のインスタンスシャットダウン。

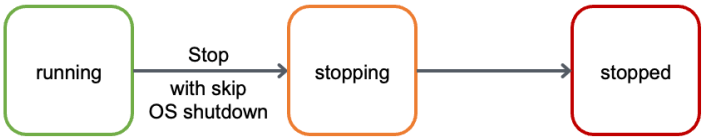
データへの影響: EBS ルートボリュームおよびデータボリューム上にデータが保持されます。インスタンスストアボリューム上のデータが失われます。

使用するタイミング: 一般的な停止の最初の停止試行。

> **Note**
>
> OS シャットダウンをスキップした停止をすでに試行している場合、同じ状態移行セッション中の後続のデフォルトの停止試行では、正常な OS シャットダウンが実行されません。正常な OS シャットダウンをバイパスすると、インスタンスの現在のセッションでは元に戻せません。

OS シャットダウンをスキップして停止する

正常な OS シャットダウンをバイパスする必要がある場合、次の図で示されるように、OS シャットダウンをスキップして停止するメソッドを使用してインスタンスを停止し、正常な OS シャットダウンをバイパスできます。



> **Warning**
>
> 正常な OS シャットダウンをバイパスすると、データの損失または破損 (例えば、メモリ内容がディスクにフラッシュされなかったり、処理中の IO が失われたりする) や、シャットダウンスクリプトがスキップされる可能性があります。

目的: インスタンスを停止するときに正常な OS シャットダウンをバイパスします。

データへの影響: データの損失や破損につながる可能性があります。メモリ内容がディスクにフラッシュされず、処理中の IO が失われる可能性があります。シャットダウンスクリプトがスキップされる可能性があります。

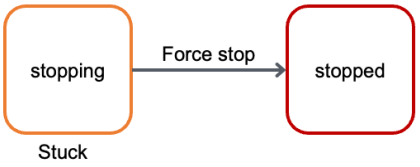
使用するタイミング: 正常な OS シャットダウンをバイパスする必要がある場合。正常な OS シャットダウンによるデフォルトの停止の進行中に使用した場合、正常な OS シャットダウンがバイパスされます。

> **Note**
>
> 正常な OS シャットダウンをバイパスすると、インスタンスの現在の状態移行セッションは元に戻せません。このセッション中に行われる次のデフォルトの停止試行では、正常な OS シャットダウンは試行されません。

強制停止する

強制停止メソッドは、`stopping` 状態でスタックしているインスタンスを処理するために使用されます。通常、基盤となるハードウェアの問題 (システムステータスチェックの失敗によって示される) によってインスタンスがスタックします。

強制停止メソッドでは、最初にデフォルトの停止が試行されます。インスタンスが `stopping` 状態でスタックした場合、`force` パラメータによってインスタンスが強制的にシャットダウンされ、次の図で示されるようにインスタンスを `stopped` 状態に移行されます。



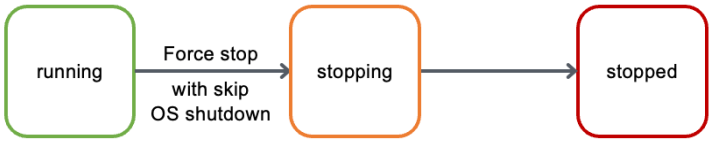
目的: stopping 状態にスタックしたインスタンスを処理します。最初にデフォルトの停止が試行されます。インスタンスの停止に失敗した場合、インスタンスが強制的にシャットダウンされます。

データへの影響: 最初にデフォルトの停止が試行されますが、強制停止が進行された場合、データ損失や破損が発生する可能性があります。まれに EBS ボリュームやその他の共有リソースに停止後の書き込みが発生する場合があります。

使用するタイミング: デフォルトの停止後にインスタンスがスタック状態になったときに 2 回目の停止を試みます。詳細については「Amazon EC2 インスタンスの停止に関する問題のトラブルシューティング」を参照してください。

OS シャットダウンをスキップして強制停止する

正常な OS シャットダウンを強制停止およびバイパスする必要がある場合、次の図で示されるように、OS シャットダウンをスキップして強制停止するメソッドを使用して、インスタンスを stopped 状態にすることができます。



目的: インスタンスを停止するとき、強制停止と正常な OS シャットダウンのバイパスを組み合わせます。

データへの影響: OS のシャットダウンをスキップすると、データの損失や破損が発生する可能性があります。メモリ内容がディスクにフラッシュされず、処理中の IO が失われる可能性があります。シャットダウンスクリプトがスキップされる可能性があります。強制停止が進行すると、追加のデータ損失や破損を引き起こす可能性があります。まれに、EBS ボリュームやその他の共有リソースに停止後の書き込みが発生する場合があります。

使用するタイミング: インスタンスが停止することを確認し、正常な OS シャットダウンをバイパスする場合。正常な OS シャットダウンによるデフォルトの停止の進行中に使用した場合、正常な OS シャットダウンがバイパスされます。

インスタンスを手動で停止して開始する

ユーザーは Amazon EBS-backed インスタンス (EBS ルートボリュームがあるインスタンス) を停止および開始できます。インスタンスストアルートボリュームがあるインスタンスを停止および開始することはできません。

デフォルトのメソッドを使用してインスタンスを停止すると、正常なオペレーティングシステム (OS) のシャットダウンが試行されます。正常な OS シャットダウンをバイパスすることはできますが、データの整合性がリスクにさらされる可能性があります。

> **Warning**
>
> インスタンスを停止すると、インスタンスストアボリューム上のデータは消去されます。インスタンスを停止する前に、必要なデータをインスタンスストアボリュームから永続的ストレージ (Amazon EBS や Amazon S3 など) にコピーしていることを確認します。

[Linux インスタンス] インスタンスから OS halt コマンドを使用しても、シャットダウンは開始されません。haltコマンドを使用すると、インスタンスは終了せず、代わりに CPU をに配置して HLT CPU 操作を一時停止します。インスタンスは実行状態のままです。

OSの shutdown コマンドまたは poweroff コマンドを使用してシャットダウンを開始できます。OS コマンドを使用すると、インスタンスはデフォルトで停止します。この動作はいつでも変更できます。詳細については「インスタンスによるシャットダウン動作を変更する」を参照してください。

> **Note**
>
> Amazon EBS-Backed インスタンスを停止し、`stopping` 状態に 「stuck」 が表示されている場合、インスタンスを強制終了できます。詳細については「Amazon EC2 インスタンスの停止に関する問題のトラブルシューティング」を参照してください。

メソッドの停止と開始

- 正常な OS シャットダウンでインスタンスを停止する
- インスタンスを停止して正常な OS シャットダウンをバイパスする
- インスタンスの開始

正常な OS シャットダウンでインスタンスを停止する

デフォルトの停止メソッドを使用してインスタンスを停止できます。これには正常な OS シャットダウンの試行が含まれます。詳細については「デフォルトの停止」を参照してください。

Console

デフォルトの停止メソッドを使用してインスタンスを停止する方法

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [インスタンス] を選択し、該当するインスタンスを選択してください。
3. [インスタンスの状態]、[インスタンスの停止] の順に選択してください。このオプションが無効になっている場合はインスタンスが既に停止しているか、またはルートボリュームがインスタンスストアボリュームです。
4. 確認を求められたら、[Stop] を選択してください。インスタンスが停止するまで、数分かかる場合があります。

AWS CLI

デフォルトの停止メソッドを使用してインスタンスを停止する方法

stop-instances コマンドを使用します。

```
aws ec2 stop-instances --instance-ids i-1234567890abcdef0
```

PowerShell

デフォルトの停止メソッドを使用してインスタンスを停止する方法

「Stop-EC2Instance」コマンドレットを使用します。

```
Stop-EC2Instance -InstanceId i-1234567890abcdef0
```

インスタンスを停止して正常な OS シャットダウンをバイパスする

インスタンスを停止するとき、正常な OS シャットダウンをバイパスできます。詳細については「OS シャットダウンをスキップして停止する」を参照してください。

> **Warning**
>
> 正常な OS シャットダウンをバイパスすると、データの損失または破損 (例えば、メモリ内容がディスクにフラッシュされなかったり、処理中の IO が失われたりする) や、シャットダウンスクリプトがスキップされる可能性があります。

Console

インスタンスを停止して正常な OS シャットダウンをバイパスする方法

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Instances] を選択し、インスタンスを選択してください。
3. [Instance state (インスタンスの状態)]、[Stop instance (インスタンスの停止)] の順に選択します。
4. [OS シャットダウンをスキップ] で、[OS シャットダウンをスキップ] チェックボックスをオンにします。コンソールにこのオプションが表示されない場合、現在のリージョンのコンソールではまだ利用できません。ただし、AWS CLI または SDK を使用してこの機能にアクセスするか、コンソールで別のリージョンを試すことができます。
5. [停止] を選択します。

AWS CLI

インスタンスを停止して正常な OS シャットダウンをバイパスする方法

`--skip-os-shutdown` で「stop-instances」コマンドを使用します。

```
aws ec2 stop-instances \
    --instance-ids i-1234567890abcdef0 \
    --skip-os-shutdown
```

PowerShell

インスタンスを停止して正常な OS シャットダウンをバイパスする方法

`-SkipOsShutdown $true` で「Stop-EC2Instance」コマンドレットを使用します。

```
Stop-EC2Instance `
```

```
    -InstanceId i-1234567890abcdef0 `
    -SkipOsShutdown $true
```

インスタンスの開始

停止したインスタンスを開始できます。

Console

インスタンスを開始するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左のナビゲーションペインの [インスタンス] を選択してください。
3. インスタンスを選択したら、[インスタンスの状態] および [インスタンスを開始] を選択してください。

   インスタンスが running 状態になるまで、数分かかる場合があります。

AWS CLI

インスタンスを開始するには

start-instances コマンドを使用します。

```
aws ec2 start-instances --instance-ids i-1234567890abcdef0
```

PowerShell

インスタンスを開始するには

Start-EC2Instance コマンドレットを使用します。

```
Start-EC2Instance -InstanceId i-1234567890abcdef0
```

インスタンスを自動的に停止して開始する

次のサービスを使用して、インスタンスの停止と開始を自動化できます。

AWS でインスタンススケジューラを使用する

インスタンススケジューラを AWS で使用して、EC2 インスタンスの開始と停止を自動化することができます。詳細については「CloudFormation で Instance Scheduler を使用して EC2 インスタンスをスケジュールするにはどうすればよいですか?」を参照してください。追加料金が適用されることに注意してください。

AWS Lambda および Amazon EventBridge ルールを使用する

Lambda と EventBridge ルールを使用して、スケジュール上のインスタンスを停止および開始することができます。詳細については「Lambda を使用して、Amazon EC2 インスタンスを一定の間隔で停止および開始するにはどうすればよいですか?」を参照してください。

Amazon EC2 Auto Scaling

アプリケーションの負荷を処理できる Amazon EC2 インスタンスの数が適切であることを確認するにはAuto Scaling グループを作成します。 Amazon EC2 Auto Scalingは、トラフィック需要を処理する適切な容量をアプリケーションで常に確保できるようにし、必要な場合にのみインスタンスを起動することでコストを節約できます。不要なインスタンスを停止するのではなく、Amazon EC2 Auto Scaling終了させることに注意してください。自動スケーリンググループを設定するには「Amazon EC2 Auto Scalingをはじめる」を参照してください。

実行中および停止中のインスタンスをすべて検索

Amazon EC2 グローバルビューではすべてにわたって、実行中と停止中のすべての AWS リージョン のインスタンスを 1 つのページで確認できます。これはインベントリを取得し、忘れられたインスタンスを見つけるのに特に有用です。グローバルビューを使用する方法については「AWS Global View を使用して複数のリージョンにわたるリソースを表示する」を参照してください。

あるいは、インスタンスがある各リージョンで、コマンドまたはコマンドレットを実行できます。

AWS CLI

リージョンにある EC2 インスタンスの数を入手するには

次の describe-instances コマンドを使用して、現在のリージョンにあるインスタンスの数を数えます。このコマンドは、インスタンスがあるリージョンごとに実行する必要があります。

```
aws ec2 describe-instances \
    --region us-east-2 \
    --query "length(Reservations[].Instances[])"
```

以下は出力の例です。

```
27
```

リージョン内の EC2 インスタンスに関する概要情報を取得するには

次の describe-instances コマンドを使用します。このコマンドは、インスタンスがあるリージョンごとに実行する必要があります。

```
aws ec2 describe-instances \
    --region us-east-2 \
    --query "Reservations[].Instances[].[InstanceId,InstanceType,PrivateIpAddress]"
 \
    --output table
```

以下は出力の例です。

```
-------------------------------------------------------------
|                   DescribeInstances                       |
+---------------------+---------------+-----------------+
|  i-0e3e777f4362f1bf7|  t2.micro     |  10.0.12.9      |
|  i-09453945dcf1529e9|  t2.micro     |  10.0.143.213   |
|  i-08fd74f3f1595fdbd|  m7i.4xlarge  |  10.0.1.103     |
+---------------------+---------------+-----------------+
```

PowerShell

リージョンにある EC2 インスタンスの数を入手するには

Get-EC2Instance コマンドレットを使用します。

```
(Get-EC2Instance -Region us-east-2).Instances.Length
```

以下は出力の例です。

```
27
```

リージョン内の EC2 インスタンスに関する概要情報を取得するには

Get-EC2Instance コマンドレットを使用します。このコマンドは、インスタンスがあるリージョンごとに実行する必要があります。

```
(Get-EC2Instance).Instances | Select InstanceId, InstanceType, PrivateIpAddress
```

以下は出力の例です。

```
InstanceId          InstanceType PrivateIpAddress
----------          ------------ ---------------
i-0e3e777f4362f1bf7 t2.micro     10.0.12.9
i-09453945dcf1529e9 t2.micro     10.0.143.213
i-08fd74f3f1595fdbd m7i.4xlarge  10.0.1.103
```

初期と最新の起動時間を確認

インスタンスを記述するとき、インスタンスの起動時間は最新の起動時間です。インスタンスを停止して開始すると、起動時間に新しいインスタンスの開始時刻が反映されます。インスタンスの初回起動時間を確認するにはインスタンスを停止して開始した後でも、プライマリネットワークインターフェイスがインスタンスにアタッチされた時刻を確認してください。

Console

直近の起動時間を見つけるには

インスタンスを選択し、[詳細] タブの [インスタンスの詳細] から、[起動時間] を見つけます。

最初の起動時刻を見つけるには

インスタンスを選択し、[ネットワーク] タブの [ネットワークインターフェイス] から [プライマリネットワークインターフェイス] (デバイスインデックスは 0) を見つけます。

AWS CLI

最初の起動時間と直近の起動時間を見つけるには

次の describe-instances コマンドを使用して、指定したインスタンスの最初の起動時間と直近の起動時間の両方を表示します。

```
aws ec2 describe-instances \
    --instance-id i-1234567890abcdef0 \
    --query 'Reservations[].Instances[].
{InstanceID:InstanceId,InitialLaunch:NetworkInterfaces[0].Attachment.AttachTime,LastLaunch:L
```

以下は出力の例です。

```
[
    {
        "InstanceID": "i-1234567890abcdef0",
        "InitialLaunch": "2024-04-19T00:47:08+00:00",
        "LastLaunch": "2024-05-27T06:24:06+00:00"
    }
]
```

PowerShell

直近の起動時間を見つけるには

Get-EC2Instance コマンドレットを使用します。

```
(Get-EC2Instance -InstanceId i-1234567890abcdef0).Instances.LaunchTime
```

以下は出力の例です。

```
Monday, May 27, 2024 6:24:06 AM
```

最初の起動時刻を見つけるには

Get-EC2Instance コマンドレットを使用します。

```
(Get-EC2Instance -
InstanceId i-1234567890abcdef0).Instances.NetworkInterfaces.Attachment.AttachTime
```

以下は出力の例です。

```
Friday, April 19, 2024 12:47:08 AM
```

EC2 インスタンスに対する停止保護を有効にする

インスタンスが誤って停止するのを防ぐために、インスタンスに対する停止保護を有効にすることができます。停止保護はインスタンスを偶発的な終了からも保護します。

Amazon EC2 ModifyInstanceAttribute API の DisableApiStop 属性はAmazon EC2 コンソール、AWS CLI、Amazon EC2 API を使用してインスタンスを停止できるかどうかを制御しま

す。この属性の値はインスタンスの起動時、インスタンスの実行中、またはインスタンスの停止時に設定できます。

考慮事項

- 停止保護を有効にしても、shutdown や poweroff などのオペレーティングシステムコマンドによりインスタンスからシャットダウンを開始してインスタンスを誤って停止することは防げません。
- 停止保護を有効にしても、インスタンスにインスタンスを停止する予定されたイベントがある場合、AWS がインスタンスを停止するのを防ぐことはできません。
- 停止保護を有効にしても、インスタンスが異常な場合やスケールインイベント中にAmazon EC2 Auto Scaling がインスタンスを終了するのを防ぐことはできません。スケールイン時に Auto Scaling グループが特定のインスタンスを終了できるかどうかを制御するにはインスタンスのスケールイン保護を使用します。
- 停止保護はインスタンスが誤って停止するのを防ぐだけでなく、コンソール、AWS CLI、または API を使用して誤って終了するのを防ぎます。ただし、`DisableApiTermination` 属性は自動的に変更されません。`DisableApiStop` 属性が `false` に設定されている場合、`DisableApiTermination` 属性の設定によって、コンソール、AWS CLI 、またはAPIを使用してインスタンスを終了できるかどうかが決まります。詳細については、「Amazon EC2 インスタンスを終了する」を参照してください。
- インスタンスストアのルートボリュームを持つインスタンスの停止保護は有効にできません。
- スポットインスタンスの停止保護は有効にできません。
- 停止保護を有効または無効にすると、Amazon EC2 API は最終的な整合性モデルに従います。つまり、停止保護属性を設定するコマンドを実行した結果が、それ以降に実行するすべてのコマンドにすぐには表示されない場合があります。詳細については「Amazon EC2 デベロッパーガイド」の「Amazon EC2 API の結果整合性」を参照してください。

停止保護タスク

- 起動時にインスタンスに対する停止保護を有効にします
- 実行中または停止したインスタンスに対する停止保護を有効にします
- 実行中または停止したインスタンスに対する停止保護を無効にします

起動時にインスタンスに対する停止保護を有効にします

インスタンスを起動するときに、インスタンスに対する停止保護を有効にできます。

Console

起動時にインスタンスに対する停止保護を有効にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ダッシュボードで、[Launch Instance (インスタンスの起動)] を選択してください。
3. [new launch instance wizard] (新しいインスタンス起動ウィザード) でインスタンスを設定します。
4. ウィザードで、[高度な詳細] の [保護停止] で [有効にする] を選択して、保護を停止します。

AWS CLI

起動時にインスタンスに対する停止保護を有効にするには

run-instances コマンドを使用して、インスタンスを起動します。次のパラメータを追加します。

```
--disable-api-stop
```

PowerShell

起動時にインスタンスに対する停止保護を有効にするには

New-EC2Instance コマンドレットを使用してインスタンスを起動します。次のパラメータを追加します。

```
-DisableApiStop $true
```

実行中または停止したインスタンスに対する停止保護を有効にします

インスタンスの実行中に、またはインスタンスを停止したときに、インスタンスに対する停止保護を有効にできます。

Console

インスタンスに対する停止保護を有効にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左のナビゲーションペインで、[Instances] (インスタンス) をクリックしてください。

3. インスタンスを選択してから、[アクション] > [インスタンスの設定] > [保護停止を変更する] を選択してください。
4. [有効化] チェックボックスをオンにしたら、[保存] を選択してください。

AWS CLI

インスタンスに対する停止保護を有効にするには

modify-instance-attribute コマンドを使用します。

```
aws ec2 modify-instance-attribute \
    --instance-id i-1234567890abcdef0 \
    --disable-api-stop
```

PowerShell

インスタンスに対する停止保護を有効にするには

Edit-EC2InstanceAttribute コマンドレットを使用します。

```
Edit-EC2InstanceAttribute `
    -InstanceId i-1234567890abcdef0 `
    -DisableApiStop $true
```

実行中または停止したインスタンスに対する停止保護を無効にします

次のいずれかの方法を使用して、実行中または停止したインスタンスに対する停止保護を無効にすることができます。

Console

実行中または停止中のインスタンスの停止保護を無効にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左のナビゲーションペインで、[Instances] (インスタンス) をクリックしてください。
3. インスタンスを選択してから、[Actions] (アクション)、[Instance Settings] (インスタンスの設定)、[Change Stop Protection] (停止保護の変更) を選択してください。
4. [有効化] チェックボックスをオフにしたら、[保存] を選択してください。

AWS CLI

実行中または停止中のインスタンスの停止保護を無効にするには

modify-instance-attribute コマンドを使用します。

```
aws ec2 modify-instance-attribute \
    --instance-id i-1234567890abcdef0 \
    --no-disable-api-stop
```

PowerShell

インスタンスに対する停止保護を無効にするには

Edit-EC2InstanceAttribute コマンドレットを使用します。

```
Edit-EC2InstanceAttribute `
    -InstanceId i-1234567890abcdef0 `
    -DisableApiStop $false
```

Amazon EC2 インスタンスの休止

インスタンスを休止すると、Amazon EC2 によってオペレーティングシステムに休止の実行 (suspend-to-disk) が指示されます。休止状態に入ると、インスタンスメモリ (RAM) に置かれていた内容が、Amazon Elastic Block Store (Amazon EBS) のルートボリュームに保存されます。インスタンスの EBS ルートボリュームとアタッチされた EBS データボリュームは、Amazon EC2 により保持されます。インスタンスが起動したとき、

- EBS ルートボリュームは前の状態に復元されます。
- RAM の内容が再ロードされます。
- インスタンスで以前に実行されていたプロセスが再開されます。
- 以前にアタッチされていたデータボリュームが再アタッチされ、インスタンスがそのインスタンス ID を保持します。

インスタンスは、休止が有効になっており、休止の前提条件を満たしている場合のみ、休止状態にすることができます。

インスタンスまたはアプリケーションが、ブートストラップし、メモリフットプリントを構築して完全に生産性を発揮するのに時間がかかる場合は、休止を使用してインスタンスを事前ウォーミングできます。インスタンスを事前ウォーミングするには、次の操作を行います。

1. 休止を有効にしてインスタンスを起動します。
2. インスタンスを必要な状態に移行させます。
3. 休止状態にして、必要なときにいつでも望ましい状態に回復されるようにします。

インスタンスが stopped 状態にある場合の休止状態のインスタンスにも、RAM の内容が EBS ルートボリュームに転送される場合のデータ転送にも、課金はされません。EBS ボリュームのストレージに対しては、RAM の内容のストレージも含めて、料金が発生します。

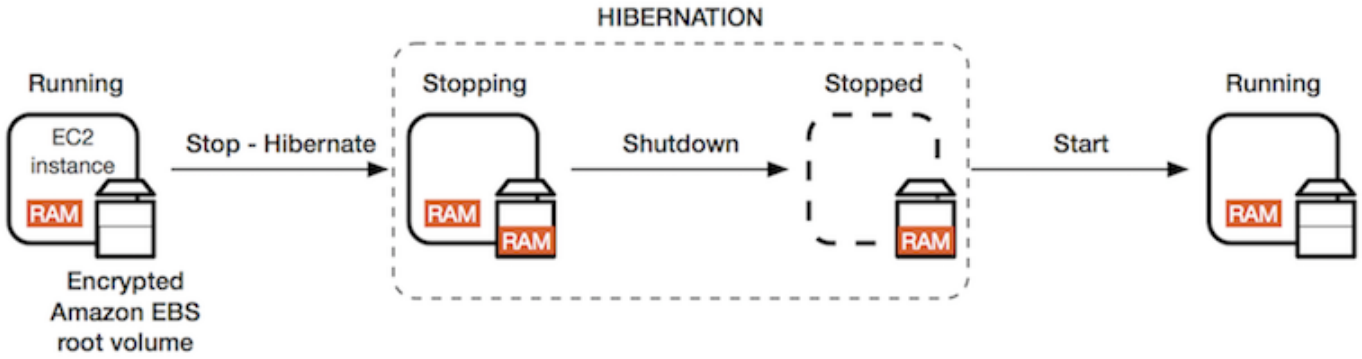
インスタンスが必要なくなった場合、stopped (休止) 状態にある場合を含め、いつでも終了することができます。詳細については「Amazon EC2 インスタンスを終了する」を参照してください。

内容

- Amazon EC2 インスタンスの休止の仕組み
- EC2 インスタンスの休止の前提条件
- Linux AMI で休止がサポートされるように設定する
- Amazon EC2 インスタンスの休止の有効化
- インスタンスでの KASLR の無効化 (Ubuntu のみ)
- Amazon EC2 インスタンスの休止
- 休止した Amazon EC2 インスタンスの起動
- Amazon EC2 インスタンスの休止のトラブルシューティング

Amazon EC2 インスタンスの休止の仕組み

次の図は、EC2 インスタンスの休止処理の基本的な概要を示しています。



インスタンスを休止するとどうなるか

インスタンスを休止すると、次の処理が実行されます。

- インスタンスはstopping状態に移行します。Amazon EC2 が、オペレーティングシステムに対して休止処理 (suspend-to-disk) を指示します。休止に伴ってすべてのプロセスがフリーズされ、RAM の内容が EBS ルートボリュームに保存されます。その後に、通常のシャットダウンが実行されます。
- シャットダウンプロセスが完了した後、インスタンスは stopped 状態に移行します。
- EBS ボリュームはインスタンスにアタッチされたままとなり、保存された RAM の内容も含めて、データは保持されます。
- Amazon EC2 インスタンスストアボリュームはインスタンスにアタッチされたままになりますが、インスタンスストアボリューム上のデータは失われます。
- 殆どの場合、インスタンスは基盤となる新しいホストコンピュータが起動したときに移行されます。これは、インスタンスを停止して起動した場合と同じです。
- インスタンスを起動すると、インスタンスのブートアッププロセスが実行され、オペレーティングシステムが EBS ルートボリュームから RAM の内容を読み取ります。次に、プロセスのフリーズが解除されて以前の状態が回復されます。
- インスタンスのプライベート IPv4 アドレスとすべての IPv6 アドレスは保持されます。インスタンスを起動すると、インスタンスは引き続きプライベート IPv4 アドレスとすべての IPv6 アドレスを保持します。
- Amazon EC2 はパブリック IPv4 アドレスをリリースします。インスタンスを起動すると、Amazon EC2 は新しいパブリック IPv4 アドレスをインスタンスに割り当てます。
- インスタンスには関連付けられた Elastic IP アドレスが保持されます。休止状態のインスタンスに関連付けられた Elastic IP アドレスに対して課金されます。

休止と再起動、停止、および終了の違いについては、インスタンスの状態の違いを参照してください。

制限事項

- インスタンスを休止すると、インスタンスストアボリューム上のデータは失われます。
- (Linux インスタンス) RAM が 150 GiB を超える Linux インスタンスを休止することはできません。
- (Windows インスタンス) RAM が 16 GiB を超える Windows インスタンスを休止することはできません。
- インスタンスが休止されている間にインスタンスを変更することはできません。これは、インスタンスのタイプやサイズなどの特定の属性を変更できる、停止された (休止ではない) インスタンスとは異なります。
- 休止状態になっている、または休止機能が有効になっているインスタンスからスナップショットまたは AMI を作成した場合、その AMI (あるいは、そのスナップショットから作成した AMI) から起動した新しいインスタンスに接続できないことがあります。
- (スポットインスタンスのみ) Amazon EC2 がスポットインスタンスを休止した場合、インスタンスを再開できるのは Amazon EC2 のみです。スポットインスタンスを休止状態 (ユーザー起動の休止) にする場合、ユーザーはインスタンスを再開できます。休止したスポットインスタンスは、容量が空いていて、スポット料金が指定した上限料金以下である場合、再開できます。
- Auto Scaling グループ内のインスタンス、または Amazon ECS が使用しているインスタンスを休止することはできません。インスタンスが Auto Scaling グループにあり、そのインスタンスを休止しようとしている場合、Amazon EC2 Auto Scaling サービスは停止したインスタンスを異常と判断し、そのインスタンスを終了して代わりのインスタンスを起動する場合があります。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling グループ内のインスタンスのヘルスチェック」を参照してください。
- UEFI Secure Boot を有効にした状態で、UEFI モードで起動するように設定されたインスタンスを休止することはできません。
- キャパシティーの予約では、キャパシティ予約で起動されたインスタンスを休止状態にする場合、そのインスタンスを再開しても、休止した時点の状態が維持されることを保証していません。
- 連邦情報処理標準 (FIPS) モードが有効になっている場合、5.10 未満のカーネルを使用するインスタンスを休止状態にすることはできません。
- 60 日間以上に及ぶインスタンス休止はサポートしていません。60 日より長くインスタンスを保持するには、休止したインスタンスを起動し、停止して、また起動する必要があります。

- 当社では、継続的にプラットフォームをアップグレードやセキュリティパッチで更新しており、休止されている既存のインスタンスと競合する可能性があります。シャットダウンまたは再起動を実行して必要なアップグレードとセキュリティパッチを適用できるように、休止されているインスタンスの起動が必要になる重要な更新については、通知を受け取ります。

スポットインスタンスを休止する場合の注意点

- ユーザーがスポットインスタンスを休止した場合、容量が空いていて、スポット料金が、指定した上限料金以下である場合、ユーザーがこれを再開できます。
- Amazon EC2 がスポットインスタンスを休止した場合は、
  - インスタンスを再開できるのは Amazon EC2 だけです。
  - Amazon EC2 は、容量が利用可能になり、スポット料金が、指定した上限料金以下である場合、休止したスポットインスタンスを再開します。
  - Amazon EC2 がスポットインスタンスを休止するときは、休止が始まる 2 分前にユーザーに中断通知が届きます。

  詳細については「スポットインスタンスの中断。」を参照してください。

EC2 インスタンスの休止の前提条件

オンデマンドインスタンスまたはスポットインスタンスの休止のサポートは、起動時に有効にすることができます。実行中または停止状態の既存のインスタンスで休止を有効にすることはできません。詳細については「インスタンスの休止の有効化」を参照してください。

インスタンスを休止するための要件

- AWS リージョン
- AMI
- インスタンスファミリー
- インスタンスの RAM サイズ
- ルートボリュームタイプ
- ルートボリュームサイズ
- ルートボリュームの暗号化
- EBS ボリュームタイプ
- スポットインスタンスリクエスト

AWS リージョン

すべての AWS リージョンのインスタンスで休止を使用できます。

AMI

休止をサポートする HVM AMI を使用する必要があります。次の AMI はハイバネーションをサポートします。

Linux AMI

Intel および AMD インスタンスタイプの AMI

- AL2023 AMI は 2023 年 9 月 20 日以降にリリース[1]
- Amazon Linux 2 AMI (2019 年 8 月 29 日以降にリリース)
- Amazon Linux AMI 2018.03 (2018 年 11 月 16 日以降にリリース)
- CentOS バージョン 8 AMI [2] (追加設定 が必要です)
- Fedora バージョン 34 以降の AMI [2] (追加設定 が必要です)
- Red Hat Enterprise Linux (RHEL) 9 AMI [2] (追加設定 が必要です)
- Red Hat Enterprise Linux (RHEL) 8 AMI [2] (追加設定 が必要です)
- 20230303 以降のシリアル番号でリリースされた Ubuntu 22.04.2 LTS (Jammy Jellyfish) AMI [3]
- 20210820 以降のシリアル番号でリリースされた Ubuntu 20.04 LTS (Focal Fossa) AMI [3]
- 20190722.1 以降のシリアル番号でリリースされた Ubuntu 18.04 LTS (Bionic Beaver) AMI [3] [5]
- Ubuntu 16.04 LTS (Xenial Xerus) AMI [3] [4] [5] (追加設定 が必要です)

Graviton インスタンスタイプの AMI

- 2024 年 7 月 1 日以降にリリースされた AL2023 AMI (64 ビット Arm) [1]
- 2024 年 6 月 20 日以降にリリースされた Amazon Linux 2 AMI (64 ビット Arm)
- 20240701 以降のシリアル番号でリリースされた Ubuntu 22.04.2 LTS (64 ビット Arm) (Jammy Jellyfish) AMI [3]
- 20240701 以降のシリアル番号でリリースされた Ubuntu 20.04 LTS (64 ビット Arm) (Focal Fossa) AMI [3]

[1] AL2023 最小 AMI の場合、追加の設定が必要です。

[2] CentOS、Fedora、Red Hat Enterprise Linux の場合、ハイバネーションは Nitro ベースのインスタンスでのみサポートされます。

[3] Ubuntu 22.04.2 LTS (Jammy Jellyfish)、Ubuntu 20.04 LTS (Focal Fossa)、Ubuntu 18.04 LTS (Bionic Beaver)、Ubuntu 16.04 LTS (Xenial Xerus) を使用するインスタンスで KASLR を無効にすることをお勧めします。詳細については「インスタンスでの KASLR の無効化 (Ubuntu のみ)」を参照してください。

[4] Ubuntu 16.04 LTS (Xenial Xerus) AMI の場合、ハイバネーションは `t3.nano` インスタンスタイプでサポートされません。Ubuntu (Xenial Xerus) が 2021 年 4 月にサポートを終了したため、パッチは利用できません。`t3.nano` インスタンスタイプを使用したい場合は、Ubuntu 22.04.2 LTS (Jammy Jellyfish)、Ubuntu 20.04 LTS (Focal Fossa) AMI または Ubuntu 18.04 LTS (Bionic Beaver) AMI にアップグレードすることをお勧めします。

[5] Ubuntu 18.04 LTS (Bionic Beaver) および Ubuntu 16.04 LTS (Xenial Xerus) のサポートは終了しました。

独自の AMI が休止をサポートするように設定するには、Linux AMI で休止がサポートされるように設定するを参照してください。

他のバージョンの Ubuntu および他のオペレーティングシステムはまもなくサポートされる予定です。

Windows AMI

- Windows Server 2022 AMI (2023 年 9 月 13 日以降にリリース)
- Windows Server 2019 AMI (2019 年 9 月 11 日以降にリリース)
- Windows Server 2016 AMI (2019 年 9 月 11 日以降にリリース)
- Windows Server 2012 R2 AMI (2019 年 9 月 11 日以降にリリース)
- Windows Server 2012 AMI (2019 年 9 月 11 日以降にリリース)

インスタンスファミリー

休止をサポートするインスタンスファミリーを使用する必要があります。ただし、ベアメタルインスタンスはサポートされていません。

- 汎用:
  M3、M4、M5、M5a、M5ad、M5d、M6a、M6g、M6gd、M6i、M6id、M6idn、M6in、M7a、M7g、M7gd

flex、M8a、M8azn、M8g、M8gb、M8gd、M8gn、M8i、M8i-
flex、M8in、M8idn、M8ib、M8idb、M9g、M9gd、T2、T3、T3a、T4g

- コンピューティング最適化:
C3、C4、C5、C5d、C6a、C6g、C6gd、C6gn、C6i、C6id、C6in、C7a、C7g、C7gd、C7gn、C7i、C7
flex、C8a、C8g、C8gb、C8gd、C8gn、C8i、C8i-flex、C8in、C8ib、C9g、C9gd
- メモリ最適化:
R3、R4、R5、R5a、R5ad、R5d、R6a、R6g、R6gd、R6idn、R6in、R7a、R7g、R7gd、R7i、R7iz、R8
flex、R8in、R8idn、R8ib、R8idb、X2gd、X8aedz、X8i
- ストレージ最適化: I3、I3en、I4g、I7i、I7ie、I8g、I8ge、Im4gn、Is4gen

Console

休止をサポートしているインスタンスタイプを取得するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンスタイプ]を選択します。
3. [オンデマンド休止のサポート] = [true] でフィルタリングします。
4. (オプション) フィルターを追加して、対象の、特定のインスタンスタイプをさらに絞り込みます。

AWS CLI

休止をサポートしているインスタンスタイプを取得するには

describe-instance-types コマンドを使用します。使用可能なインスタンスタイプはリージョンごとに異なることに注意してください。

```
aws ec2 describe-instance-types \
    --filters Name=hibernation-supported,Values=true \
    --query "InstanceTypes[*].[InstanceType]" \
    --output text | sort
```

PowerShell

休止をサポートしているインスタンスタイプを取得するには

Get-EC2InstanceType コマンドレットを使用します。使用可能なインスタンスタイプはリージョンごとに異なることに注意してください。

```
(Get-EC2InstanceType `
    -Filter @{Name="hibernation-supported"; Values="true"}).InstanceType | Sort-
Object
```

インスタンスの RAM サイズ

Linux インスタンス – 150 GiB 未満である必要があります。

Windows インスタンス – 16 GiB 以下である必要があります。T3 または T3a インスタンスの休止には、最低 1 GiB の RAM をお勧めします。

ルートボリュームタイプ

ルートボリュームは、インスタンスストアボリュームではなく EBS ボリュームにする必要があります。

ルートボリュームサイズ

ルートボリュームは、RAM の内容を保存し、OS やアプリケーションなどの予想される使用量に対応できる大きさにする必要があります。休止を有効にすると、RAM を保存するために起動時にルートボリュームでスペースが割り当てられます。

ルートボリュームの暗号化

休止時にメモリ内にある機密性の高いコンテンツを保護するためにルートボリュームを暗号化する必要があります。RAM データを EBS ルートボリュームに移動する場合は、常に暗号化します。ルートボリュームの暗号化は、インスタンスの起動時に適用されます。

ルートボリュームが暗号化された EBS ボリュームであることを確認するには、次の 3 つのオプションのいずれかを使用します。

- デフォルトでの EBS 暗号化: EBS 暗号化をデフォルトで有効にして、AWS アカウントで作成されたすべての新しい EBS ボリュームを暗号化できます。この方法では、インスタンスの起動時に暗号化のインテントを指定することなく、インスタンスの休止を有効にすることができます。詳細については、「デフォルトで暗号化を有効にする」を参照してください。
- EBS の「シングルステップ」暗号化: 暗号化されていない AMI から暗号化された EBS-Backed EC2 インスタンスを起動し、同時に休止状態を有効にすることができます。詳細については「EBS-backed AMI での暗号化の利用」を参照してください。
- 暗号化された AMI: 暗号化された AMI を使用してインスタンスを起動することで、EBS 暗号化を有効にすることができます。暗号化されたルートスナップショットが AMI にない場合は、それを

新しい AMI にコピーして暗号化をリクエストできます。詳細については、「コピー時に暗号化されていないイメージを暗号化する」および「AMI のコピー」を参照してください。

EBS ボリュームタイプ

EBS ボリュームは、次のいずれかの EBS ボリュームタイプを使用する必要があります。

- 汎用 SSD (gp2 および gp3)
- プロビジョンド IOPS SSD (io1 および io2)

プロビジョンド IOPS SSD ボリュームタイプを選択した場合、休止状態の最適なパフォーマンスを実現するには、適切な IOPS で EBS ボリュームをプロビジョニングする必要があります。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームの種類」を参照してください。

スポットインスタンスリクエスト

スポットインスタンスには、次の要件が適用されます。

- スポットインスタンスのリクエストのタイプは `persistent` である必要があります。
- スポットインスタンスリクエストで起動グループを指定することはできません。

Linux AMI で休止がサポートされるように設定する

以下の Linux AMI は、Amazon EC2 インスタンスの休止に対応できます。ただし、ここで説明する追加の設定手順を完了した場合に限ります。

その他の設定も必要です。

- AL2023 最小 AMI は 2023 年 9 月 20 日以降にリリース
- Amazon Linux 2 minimal AMI (2019 年 8 月 29 日以降にリリース)
- 2019 年 8 月 29 日以前にリリースされた Amazon Linux 2
- 2018 年 11 月 16 日以前にリリースされた Amazon Linux
- CentOS バージョン 8 以降
- Fedora バージョン 34 以降
- Red Hat Enterprise Linux バージョン 8 または 9 以降
- シリアル番号 20210820 よりも前にリリースされた Ubuntu 20.04 LTS (Focal Fossa)

- シリアル番号 20190722.1 よりも前にリリースされた Ubuntu 18.04 (Bionic Beaver)
- Ubuntu 16.04 (Xenial Xerus)

休止に対応し、追加で設定を行う必要がない Linux と Windows AMI については、「AMI」を参照してください。

詳細については、「Amazon Linux 2 インスタンスでのインスタンスソフトウェアの更新」を参照してください。

AL2023 最小 AMI は 2023 年 9 月 20 日以降にリリース

ハイバネーションをサポートするために 2023 年 9 月 20 日以降にリリースされた AL2023 最小 AMI の設定方法

1. ec2-hibinit-agentパッケージをリポジトリからインストールします。

```
[ec2-user ~]$ sudo dnf install ec2-hibinit-agent
```

2. サービスを再起動します。

```
[ec2-user ~]$ sudo systemctl start hibinit-agent
```

Amazon Linux 2 minimal AMI (2019 年 8 月 29 日以降にリリース)

2019 年 8 月 29 日以降にリリースされた Amazon Linux 2 minimal AMI で休止がサポートされるように設定するには

1. ec2-hibinit-agentパッケージをリポジトリからインストールします。

```
[ec2-user ~]$ sudo yum install ec2-hibinit-agent
```

2. サービスを再起動します。

```
[ec2-user ~]$ sudo systemctl start hibinit-agent
```

2019 年 8 月 29 日以前にリリースされた Amazon Linux 2

2019 年 8 月 29 日以前にリリースされた Amazon Linux 2 AMI で休止がサポートされるように設定するには

1. `4.14.138-114.102`以降にカーネルを更新します。

```
[ec2-user ~]$ sudo yum update kernel
```

2. `ec2-hibinit-agent`パッケージをリポジトリからインストールします。

```
[ec2-user ~]$ sudo yum install ec2-hibinit-agent
```

3. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

4. 次のコマンドを実行して、カーネルバージョンが `4.14.138-114.102` 以降に更新されていることを確認します。

```
[ec2-user ~]$ uname -a
```

5. インスタンスを停止し、AMI を作成します。詳細については「Amazon EBS-backed AMI を作成する」を参照してください。

2018 年 11 月 16 日以前にリリースされた Amazon Linux

2018 年 11 月 16 日以前にリリースされた Amazon Linux AMI で休止がサポートされるように設定するには

1. `4.14.77-70.59`以降にカーネルを更新します。

```
[ec2-user ~]$ sudo yum update kernel
```

2. `ec2-hibinit-agent`パッケージをリポジトリからインストールします。

```
[ec2-user ~]$ sudo yum install ec2-hibinit-agent
```

3. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

4. 次のコマンドを実行して、カーネルバージョンが 4.14.77-70.59 以降に更新されていることを確認します。

```
[ec2-user ~]$ uname -a
```

5. インスタンスを停止し、AMI を作成します。詳細については「Amazon EBS-backed AMI を作成する」を参照してください。

CentOS バージョン 8 以降

休止状態をサポートするように CentOS バージョン 8 以降の AMI を設定するには

1. 4.18.0-305.7.1.el8_4.x86_64以降にカーネルを更新します。

```
[ec2-user ~]$ sudo yum update kernel
```

2. このステップでは、Fedora Extra Packages for Enterprise Linux (EPEL) リポジトリをインストールします。

```
[ec2-user ~]$ sudo yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm
```

3. ec2-hibinit-agentパッケージをリポジトリからインストールします。

```
[ec2-user ~]$ sudo yum install ec2-hibinit-agent
```

4. 起動時に休止状態エージェントを起動できるようにします。

```
[ec2-user ~]$ sudo systemctl enable hibinit-agent.service
```

5. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

6. 次のコマンドを実行して、カーネルバージョンが 4.18.0-305.7.1.el8_4.x86_64 以降に更新されていることを確認します。

```
[ec2-user ~]$ uname -a
```

Fedora バージョン 34 以降

休止状態をサポートするために Fedora バージョン 34 以降の AMI を設定するには

1. 5.12.10-300.fc34.x86_64以降にカーネルを更新します。

```
[ec2-user ~]$ sudo yum update kernel
```

2. ec2-hibinit-agentパッケージをリポジトリからインストールします。

```
[ec2-user ~]$ sudo dnf install ec2-hibinit-agent
```

3. 起動時に休止状態エージェントを起動できるようにします。

```
[ec2-user ~]$ sudo systemctl enable hibinit-agent.service
```

4. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

5. 次のコマンドを実行して、カーネルバージョンが 5.12.10-300.fc34.x86_64 以降に更新されていることを確認します。

```
[ec2-user ~]$ uname -a
```

Red Hat Enterprise Linux バージョン 8 または 9 以降

休止状態をサポートするように Red Hat Enterprise Linux 8 または 9 AMI を設定するには

1. 4.18.0-305.7.1.el8_4.x86_64以降にカーネルを更新します。

```
[ec2-user ~]$ sudo yum update kernel
```

2. このステップでは、Fedora Extra Packages for Enterprise Linux (EPEL) リポジトリをインストールします。

RHEL バージョン 8:

```
[ec2-user ~]$ sudo yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm
```

RHEL バージョン 9:

```
[ec2-user ~]$ sudo yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-9.noarch.rpm
```

3. ec2-hibinit-agentパッケージをリポジトリからインストールします。

```
[ec2-user ~]$ sudo yum install ec2-hibinit-agent
```

4. 起動時に休止状態エージェントを起動できるようにします。

```
[ec2-user ~]$ sudo systemctl enable hibinit-agent.service
```

5. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

6. 次のコマンドを実行して、カーネルバージョンが 4.18.0-305.7.1.el8_4.x86_64 以降に更新されていることを確認します。

```
[ec2-user ~]$ uname -a
```

シリアル番号 20210820 よりも前にリリースされた Ubuntu 20.04 LTS (Focal Fossa)

シリアル番号 20210820 よりも前にリリースされた Ubuntu 20.04 LTS (Focal Fossa) AMI で休止がサポートされるように設定するには

1. linux-aws-kernel を 5.8.0-1038.40 以降に、grub2 を 2.04-1ubuntu26.13 以降に更新します。

```
[ec2-user ~]$ sudo apt update
[ec2-user ~]$ sudo apt dist-upgrade
```

2. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

3. 次のコマンドを実行して、カーネルバージョンが 5.8.0-1038.40 以降に更新されていることを確認します。

```
[ec2-user ~]$ uname -a
```

4. 次のコマンドを実行して、grub2 バージョンが 2.04-1ubuntu26.13 以降に更新されていることを確認します。

```
[ec2-user ~]$ dpkg --list | grep grub2-common
```

シリアル番号 20190722.1 よりも前にリリースされた Ubuntu 18.04 (Bionic Beaver)

シリアル番号 20190722.1 以前にリリースされた Ubuntu 18.04 LTS AMI で休止がサポートされるように設定するには

1. 4.15.0-1044以降にカーネルを更新します。

```
[ec2-user ~]$ sudo apt update
[ec2-user ~]$ sudo apt dist-upgrade
```

2. ec2-hibinit-agentパッケージをリポジトリからインストールします。

```
[ec2-user ~]$ sudo apt install ec2-hibinit-agent
```

3. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

4. 次のコマンドを実行して、カーネルバージョンが 4.15.0-1044 以降に更新されていることを確認します。

```
[ec2-user ~]$ uname -a
```

Ubuntu 16.04 (Xenial Xerus)

Ubuntu 16.04 LTS で休止がサポートされるように設定するには、linux-aws-hwe カーネルパッケージバージョン 4.15.0-1058-aws 以降および ec2-hibinit-agent をインストールする必要があります。

> **Important**
>
> `linux-aws-hwe` カーネルパッケージは、Canonical でサポートされています。Ubuntu 16.04 LTS の標準サポートは 2021 年 4 月に終了し、パッケージは定期的な更新を受信しなくなりました。ただし、拡張セキュリティメンテナンスのサポートが2024年に終了するまで、追加のセキュリティアップデートを受け取ります。詳細については、Ubuntu 16.04 LTS 用 Amazon EC2 の休止機能が利用可能にを参照してください。
> Ubuntu 20.04 LTS (Focal Fossa) AMI または Ubuntu 18.04 LTS (Bionic Beaver) AMI にアップグレードすることをお勧めします。

Ubuntu 16.04 LTS AMI で休止がサポートされるように設定するには

1. `4.15.0-1058-aws`以降にカーネルを更新します。

```
[ec2-user ~]$ sudo apt update
[ec2-user ~]$ sudo apt install linux-aws-hwe
```

2. `ec2-hibinit-agent`パッケージをリポジトリからインストールします。

```
[ec2-user ~]$ sudo apt install ec2-hibinit-agent
```

3. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

4. 次のコマンドを実行して、カーネルバージョンが `4.15.0-1058-aws` 以降に更新されていることを確認します。

```
[ec2-user ~]$ uname -a
```

Amazon EC2 インスタンスの休止の有効化

インスタンスを休止するには、まずインスタンスを起動するときに休止を有効にする必要があります。

> **Important**
>
> インスタンスの起動後に、そのインスタンスの休止を有効または無効にすることはできません。

タスク

- オンデマンドインスタンスの休止を有効にする
- スポットインスタンスの休止を有効にする
- インスタンスで休止が有効かどうかを表示する

オンデマンドインスタンスの休止を有効にする

オンデマンドインスタンスの休止を有効にできます。

Console

オンデマンドインスタンスの休止を有効にするには

1. 手順に従ってインスタンスを起動しますが、次のステップを完了して休止状態を有効にするまでインスタンスを起動しないでください。
2. 休止状態を有効にするには、インスタンス起動ウィザードで次のフィールドを設定します。
   a. [Application and OS Images (Amazon Machine Image)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で、休止状態をサポートする AMI を選択してください。詳細については「AMI」を参照してください。
   b. [Instance type] (インスタンスタイプ) で、サポートされているインスタンスタイプを選択してください。詳細については「インスタンスファミリー」を参照してください。
   c. [Configure storage] (ストレージを設定) で、[Advanced] (高度) (右側) を選択し、ルートボリュームに関する次の情報を指定します。

- [サイズ (GiB)] に、EBS ルートボリュームのサイズを入力してください。ボリュームは、RAM の内容を格納して予想使用量に対応できるだけのサイズにする必要があります。
- [Volume type] (ボリュームタイプ) で、サポートされている EBS ボリュームタイプである汎用 SSD (gp2 および gp3) またはプロビジョンド IOPS SSD (io1 および io2) を選択してください。
- [Encrypted] (暗号化) で、[Yes] (はい) を選択してください。この AWS リージョンでデフォルトで暗号化を有効にした場合、[Yes] (はい) が選択されます。
- [KMS key] (KMS キー) で、ボリュームの暗号化キーを選択してください。この AWS リージョンでデフォルトで暗号化を有効にした場合、デフォルトの暗号化キーが選択されます。

ルートボリュームの前提条件の詳細については、EC2 インスタンスの休止の前提条件を参照してください。

d. [Advanced details] (高度な詳細) を展開し、[Stop - Hibernate behavior] (停止 - 休止状態の動作) で [Enable] (有効にする) を選択してください。

3. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択してください。詳細については「コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する」を参照してください。

AWS CLI

オンデマンドインスタンスの休止を有効にするには

run-instances コマンドを使用して、インスタンスを起動します。`--block-device-mappings file://mapping.json` パラメータを使用して EBS ルートボリュームのパラメータを指定し、`--hibernation-options Configured=true` パラメータを使用して休止状態を有効にします。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type m5.large \
    --block-device-mappings file://mapping.json \
    --hibernation-options Configured=true \
    --count 1 \
    --key-name MyKeyPair
```

mapping.json で、以下を指定します。

```
[
    {
        "DeviceName": "/dev/xvda",
        "Ebs": {
            "VolumeSize": 30,
            "VolumeType": "gp2",
            "Encrypted": true
        }
    }
]
```

DeviceName の値は、AMI に関連付けられているルートデバイス名と一致する必要があります。ルートデバイス名を確認するには、次のように describe-images コマンドを使用します。

```
aws ec2 describe-images --image-id ami-0abcdef1234567890
```

この AWS リージョンで暗号化をデフォルトで有効にした場合は、"Encrypted": true を省略できます。

PowerShell

オンデマンドインスタンスの休止を有効にするには

New-EC2Instance コマンドを使用してインスタンスを起動します。EBS ルートボリュームを指定します。最初にブロックデバイスマッピングを定義し、次に -BlockDeviceMappings パラメータを使用してそれをコマンドに追加します。-HibernationOptions_Configured $true パラメータを使用して休止状態を有効にします。

```
$ebs_encrypt = New-Object Amazon.EC2.Model.BlockDeviceMapping
$ebs_encrypt.DeviceName = "/dev/xvda"
$ebs_encrypt.Ebs = New-Object Amazon.EC2.Model.EbsBlockDevice
$ebs_encrypt.Ebs.VolumeSize = 30
$ebs_encrypt.Ebs.VolumeType = "gp2"
$ebs_encrypt.Ebs.Encrypted = $true

New-EC2Instance `
    -ImageId ami-0abcdef1234567890 `
    -InstanceType m5.large `
    -BlockDeviceMappings $ebs_encrypt `
```

```
    -HibernationOptions_Configured $true `
    -MinCount 1 `
    -MaxCount 1 `
    -KeyName MyKeyPair
```

DeviceName の値は、AMI に関連付けられているルートデバイス名と一致する必要があります。ルートデバイス名を確認するには、次のように Get-EC2Image コマンドを使用します。

```
Get-EC2Image -ImageId ami-0abcdef1234567890
```

この AWS リージョンで暗号化をデフォルトで有効にした場合は、ブロックデバイスマッピングから Encrypted = $true を省略できます。

スポットインスタンスの休止を有効にする

スポットインスタンスの休止を有効にできます。中断時のスポットインスタンスの休止に関する詳細は、「スポットインスタンスの中断。」を参照してください。

Console

スポットインスタンスの休止を有効にするには

1. 次の手順に従って、インスタンス起動ウィザードを使ってスポットインスタンスをリクエストしますが、次のステップを完了して休止を有効にするまで、インスタンスを起動しないでください。
2. 休止状態を有効にするには、インスタンス起動ウィザードで次のフィールドを設定します。
   a. [Application and OS Images (Amazon Machine Image)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で、休止状態をサポートする AMI を選択してください。詳細については「AMI」を参照してください。
   b. [Instance type] (インスタンスタイプ) で、サポートされているインスタンスタイプを選択してください。詳細については「インスタンスファミリー」を参照してください。
   c. [Configure storage] (ストレージを設定) で、[Advanced] (高度) (右側) を選択し、ルートボリュームに関する次の情報を指定します。
      - [サイズ (GiB)] に、EBS ルートボリュームのサイズを入力してください。ボリュームは、RAM の内容を格納して予想使用量に対応できるだけのサイズにする必要があります。

- [Volume type] (ボリュームタイプ) で、サポートされている EBS ボリュームタイプである汎用 SSD (gp2 および gp3) またはプロビジョンド IOPS SSD (io1 および io2) を選択してください。
- [Encrypted] (暗号化) で、[Yes] (はい) を選択してください。この AWS リージョンでデフォルトで暗号化を有効にした場合、[Yes] (はい) が選択されます。
- [KMS key] (KMS キー) で、ボリュームの暗号化キーを選択してください。この AWS リージョンでデフォルトで暗号化を有効にした場合、デフォルトの暗号化キーが選択されます。

ルートボリュームの前提条件の詳細については、EC2 インスタンスの休止の前提条件を参照してください。

d. [詳細設定] を展開し、スポットインスタンスを設定するフィールドに加えて次の操作を行います。

i. [リクエストタイプ] で [永続的] を選択してください。

ii. [中断動作] で [休止] を選択してください。または、[停止 - 休止動作] で [有効] を選択してください。どちらのフィールドも、スポットインスタンスの休止を有効にします。いずれか 1 つ設定すれば済みます。

3. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択してください。詳細については「コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する」を参照してください。

AWS CLI

スポットインスタンスの休止を有効にするには

run-instances コマンドを使用してスポットインスタンスをリクエストします。--block-device-mappings file://mapping.json パラメータを使用して EBS ルートボリュームのパラメータを指定し、--hibernation-options Configured=true パラメータを使用して休止状態を有効にします。スポットのリクエストのタイプ SpotInstanceTypeは persistent である必要があります。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type c4.xlarge \
    --block-device-mappings file://mapping.json \
    --hibernation-options Configured=true \
```

```
    --count 1 \
    --key-name MyKeyPair
    --instance-market-options
        {
           "MarketType":"spot",
           "SpotOptions":{
              "MaxPrice":"1",
              "SpotInstanceType":"persistent"
            }
        }
```

mapping.json の EBS ルートボリュームパラメータを次のとおり指定します。

```
[
    {
        "DeviceName": "/dev/xvda",
        "Ebs": {
            "VolumeSize": 30,
            "VolumeType": "gp2",
            "Encrypted": true
        }
    }
]
```

DeviceName の値は、AMI に関連付けられているルートデバイス名と一致する必要があります。ルートデバイス名を確認するには、次のように describe-images コマンドを使用します。

```
aws ec2 describe-images --image-id ami-0abcdef1234567890
```

この AWS リージョンで暗号化をデフォルトで有効にした場合は、"Encrypted": true を省略できます。

PowerShell

スポットインスタンスの休止を有効にするには

New-EC2Instance コマンドを使用してスポットインスタンスをリクエストします。EBS ルートボリュームを指定します。最初にブロックデバイスマッピングを定義し、次に -BlockDeviceMappings パラメータを使用してそれをコマンドに追加します。-HibernationOptions_Configured $true パラメータを使用して休止状態を有効にします。

```
$ebs_encrypt = New-Object Amazon.EC2.Model.BlockDeviceMapping
```

```
$ebs_encrypt.DeviceName = "/dev/xvda"
$ebs_encrypt.Ebs = New-Object Amazon.EC2.Model.EbsBlockDevice
$ebs_encrypt.Ebs.VolumeSize = 30
$ebs_encrypt.Ebs.VolumeType = "gp2"
$ebs_encrypt.Ebs.Encrypted = $true

New-EC2Instance `
    -ImageId ami-0abcdef1234567890 `
    -InstanceType m5.large `
    -BlockDeviceMappings $ebs_encrypt `
    -HibernationOptions_Configured $true `
    -MinCount 1 `
    -MaxCount 1 `
    -KeyName MyKeyPair `
    -InstanceMarketOption @(
        MarketType = spot;
        SpotOptions @{
        MaxPrice = 1;
        SpotInstanceType = persistent}
    )
```

DeviceName の値は、AMI に関連付けられているルートデバイス名と一致する必要があります。ルートデバイス名を確認するには、次のように Get-EC2Image コマンドを使用します。

```
Get-EC2Image -ImageId ami-0abcdef1234567890
```

この AWS リージョンで暗号化をデフォルトで有効にした場合は、ブロックデバイスマッピングから Encrypted = $true を省略できます。

インスタンスで休止が有効かどうかを表示する

インスタンスの休止が有効であるかどうかを確認できます。

Console

インスタンスで休止が有効かどうかを表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。

3. インスタンスを選択し、[Details (詳細)] タブの [Instance details (インスタンスの詳細)] セクションで、[Stop-hibernate behavior (停止 - 休止動作)] を確認します。[有効] は、インスタンスが休止に対して有効であることを示します。

AWS CLI

インスタンスで休止が有効かどうかを表示するには

describe-instances コマンドを使用し、`--filters "Name=hibernation-options.configured,Values=true"` パラメータを指定して、休止が有効になっているインスタンスをフィルタリングします。

```
aws ec2 describe-instances \
    --filters "Name=hibernation-options.configured,Values=true"
```

次の出力フィールドは、インスタンスで休止が有効になっていることを示しています。

```
"HibernationOptions": {
    "Configured": true
}
```

PowerShell

インスタンスで休止が有効かどうかを表示するには

Get-EC2Instance コマンドレットを使用し、休止が有効になっているインスタンスをフィルタリングします。

```
(Get-EC2Instance `
    -Filter @{Name="hibernation-options.configured"; Values="true"}).Instances
```

インスタンスでの KASLR の無効化 (Ubuntu のみ)

Ubuntu 16.04 LTS (Xenial Xerus)、Ubuntu 18.04 LTS (Bionic Beaver) (シリアル番号 20190722.1 以降でリリース)、または Ubuntu 20.04 LTS (Focal Fossa) (シリアル番号 20210820 以降でリリース) で新しく起動されたインスタンスで休止を使用するには、KASLR (Kernel Address Space Layout Randomization) を無効にするようお勧めします。Ubuntu 16.04 LTS、Ubuntu 18.04 LTS、または Ubuntu 20.04 LTS では、デフォルトで KASLR が有効になっています。

KASLR は、Linux カーネルに対する標準的なセキュリティ機能であり、カーネルのベースアドレス値をランダム化することにより、未知のメモリアクセス脆弱性による露出と影響を軽減するために役立ちます。KASLR が有効になっている場合は、インスタンスを休止後に再開できないこともあります。

KASLR の詳細については、Ubuntu の機能に関する記述を参照してください。

Ubuntu で起動したインスタンスで KASLR を無効にするには

1. SSH を使用してインスタンスに接続します。詳細については「SSH を使用した Linux インスタンスへの接続」を参照してください。
2. 適切なエディタで、/etc/default/grub.d/50-cloudimg-settings.cfg ファイルを開きます。次の例のように、GRUB_CMDLINE_LINUX_DEFAULT 行を編集して、行末に nokaslr オプションを追加します。

   ```
   GRUB_CMDLINE_LINUX_DEFAULT="console=tty1 console=ttyS0
    nvme_core.io_timeout=4294967295 nokaslr"
   ```
3. ファイルを保存し、エディタを終了します。
4. grub 設定を再構築するには、次のコマンドを実行します。

   ```
   sudo update-grub
   ```
5. インスタンスを再起動します。

   ```
   sudo reboot
   ```
6. 次のコマンドを実行して、nokaslr が追加されたことを確認します。

   ```
   cat /proc/cmdline
   ```

   コマンドの出力には、nokaslr オプションが含まれている必要があります。

Amazon EC2 インスタンスの休止

インスタンスが EBS ベースのインスタンスであり、休止が有効になっており、休止の前提条件を満たしている場合、オンデマンドインスタンスまたはスポットインスタンスで休止を開始できます。インスタンスを休止できない場合、通常のシャットダウンが実行されます。

Console

インスタンスを休止するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択し、[Instance state (インスタンスの状態)]、[Hibernate instance (インスタンスの休止)] の順に選択してください。[Hibernate instance (インスタンスの休止)] が無効になっている場合は、インスタンスが既に休止または停止しているか、休止できません。詳細については「EC2 インスタンスの休止の前提条件」を参照してください。
4. 確認を求めるメッセージが表示されたら、[休止] を選択してください。インスタンスが休止するまで、数分かかる場合があります。インスタンスの状態は、最初に停止中に変化し、インスタンスが休止状態になったときに停止に変化します。

AWS CLI

インスタンスを休止するには

stop-instances コマンドを使用して `--hibernate` パラメータを指定します。

```
aws ec2 stop-instances \
    --instance-ids i-1234567890abcdef0 \
    --hibernate
```

PowerShell

インスタンスを休止するには

Stop-EC2Instance コマンドレットを使用します。

```
Stop-EC2Instance `
    -InstanceId i-1234567890abcdef0 `
    -Hibernate $true
```

インスタンスで休止が開始されたかどうかを確認できます。

Console

インスタンスで休止が開始されたかどうかを表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択し、[詳細] タブの [インスタンスの詳細] セクションで、[状態遷移メッセージ] の値を確認します。

   Client.UserInitiatedHibernate: User initiated hibernate というメッセージは、オンデマンドインスタンスまたはスポットインスタンスで休止が開始されたことを示しています。

AWS CLI

インスタンスで休止が開始されたかどうかを表示するには

describe-instances コマンドを使用して、`state-reason-code` フィルターを指定し、休止が開始されたインスタンスを確認します。

```
aws ec2 describe-instances \
    --filters "Name=state-reason-code,Values=Client.UserInitiatedHibernate"
```

以下の出力のフィールドは、そのオンデマンドインスタンスまたはスポットインスタンスで休止が開始されたことを示しています。

```
"StateReason": {
    "Code": "Client.UserInitiatedHibernate"
}
```

PowerShell

インスタンスで休止が開始されたかどうかを表示するには

Get-EC2Instance コマンドレットを使用し、`state-reason-code` フィルタを指定して休止が開始されたインスタンスを確認します。

```
Get-EC2Instance `
    -Filter @{Name="state-reason-code";Value="Client.UserInitiatedHibernate"}
```

休止した Amazon EC2 インスタンスの起動

休止したインスタンスは、停止したインスタンスを起動するのと同じ方法で起動します。

スポットインスタンスの場合、Amazon EC2 がインスタンスを休止にした場合、それを再開できるのは Amazon EC2 のみです。ユーザーは、自分で休止した場合のみ、休止したスポットインスタンスを再開できます。スポットインスタンスは、容量が空いていて、スポット料金が、指定した上限料金以下である場合、再開できます。

Console

休止したインスタンスの起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. 休止したインスタンスを選択し、[Instance state (インスタンスの状態)]、[Start instance (インスタンスの開始)] の順に選択してください。インスタンスが `running` 状態になるまで、数分かかる場合があります。この間、インスタンスの ステータスチェック では、インスタンスが起動するまで、インスタンスは失敗状態にあるように表示されます。

AWS CLI

休止したインスタンスの起動するには

start-instances コマンドを使用します。

```
aws ec2 start-instances --instance-ids i-1234567890abcdef0
```

PowerShell

休止したインスタンスの起動するには

Start-EC2Instance コマンドレットを使用します。

```
Start-EC2Instance -InstanceId i-1234567890abcdef0
```

Amazon EC2 インスタンスの休止のトラブルシューティング

次の情報を使用して、インスタンスを休止するときに発生する可能性がある問題の診断や修復を行います。

休止に関する問題

- 起動直後に休止できません
- stopping から stopped への移行に時間がかかりすぎ、起動後にメモリ状態が復元されません
- インスタンスが stopping 状態で止まりました
- 休止の直後にスポットインスタンスを起動できない
- スポットインスタンスを再開できない

起動直後に休止できません

インスタンスの起動後にすぐ休止しようとすると、エラーが発生します。

起動後、Linux インスタンスの場合は約 2 分、Windows インスタンスの場合は約 5 分待ってから休止する必要があります。

stopping から stopped への移行に時間がかかりすぎ、起動後にメモリ状態が復元されません

休止しているインスタンスが `stopping` 状態から `stopped` に移行するのに時間がかかり過ぎ、メモリの状態が起動後に復元されない場合は、休止が正しく設定されていない可能性があります。

Linux インスタンス

インスタンスのシステムログをチェックして、休止に関連するメッセージを探します。システムログにアクセスするには、インスタンスに接続するか、get-console-output コマンドを使用します。`hibinit-agent` からログ行を見つけます。ログ行が失敗を示している場合、またはログ行がない場合、起動時に休止の設定に失敗している可能性が高いと思われます。

例えば、メッセージ「`hibinit-agent: Insufficient disk space. Cannot create setup for hibernation. Please allocate a larger root device.`」は、インスタンスのルートボリュームの大きさが十分ではないことを示しています。

`hibinit-agent` からの最後のログ行が `hibinit-agent: Running: swapoff /swap` である場合、休止は正常に設定されています。

これらのプロセスで何もログが表示されない場合、AMI が休止をサポートしていない可能性があります。サポート対象の AMI の詳細については、EC2 インスタンスの休止の前提条件を参照してくだ

さい。独自の Linux AMI を使用した場合は、必ず Linux AMI で休止がサポートされるように設定するの指示に従ってください。

Windows Server 2016 以降

EC2 起動ログをチェックして、休止に関連するメッセージを探します。EC2 起動ログにアクセスするには、インスタンスに接続し、テキストエディタで C:\ProgramData\Amazon\EC2-Windows\Launch\Log\Ec2Launch.log ファイルを開きます。EC2Launch v2 を使用している場合は、C:\ProgramData\Amazon\EC2Launch\log\agent.log を開きます。

> **Note**
>
> Windows では、デフォルトで C:\ProgramData 以下のファイルとフォルダは非表示になります。EC2 起動ディレクトリおよびファイルを表示するには、Windows エクスプローラーにパスを入力するか、フォルダのプロパティを変更して非表示のファイルおよびフォルダを表示します。

休止に関するログ行を見つけます。ログ行が失敗を示している場合、またはログ行がない場合、起動時に休止の設定に失敗している可能性が高いと思われます。

例えば、「Message: Failed to enable hibernation.」というメッセージは、休止の設定に失敗したことを示しています。エラーメッセージに 10 進数の ASCII 値が含まれている場合は、ASCII 値をプレーンテキストに変換すると、エラーメッセージ全体を読み取ることができます。

ログ行に HibernationEnabled: true が含まれている場合、休止は正常に設定されています。

Windows Server 2012 R2 以前

EC2 設定ログをチェックして、休止に関連するメッセージを探します。EC2 設定ログにアクセスするには、インスタンスに接続し、テキストエディタで C:\Program Files\Amazon\Ec2ConfigService\Logs\Ec2ConfigLog.txt ファイルを開きます。SetHibernateOnSleep のログ行を見つけます。ログ行が失敗を示している場合、またはログ行がない場合、起動時に休止の設定に失敗している可能性が高いと思われます。

例えば、メッセージ「SetHibernateOnSleep: Failed to enable hibernation: Hibernation failed with the following error: There is not enough space on the disk.」は、インスタンスのルートボリュームの大きさが十分ではないことを示しています。

ログ行が SetHibernateOnSleep: HibernationEnabled: true である場合、休止は正常に設定されています。

Windows インスタンスサイズ

1 GiB 未満の RAM を持つ T3 または T3a Windows インスタンスを使用している場合は、インスタンスのサイズを少なくとも 1 GiB の RAM に増加してみてください。

インスタンスが stopping 状態で止まりました

インスタンスを休止し、stopping 状態で「stuck」まったように見える場合は、インスタンスを強制終了できます。詳細については「Amazon EC2 インスタンスの停止に関する問題のトラブルシューティング」を参照してください。

休止の直後にスポットインスタンスを起動できない

休止にしてから 2 分以内にスポットインスタンスを起動しようとすると、次のエラーが発生する場合があります。

```
You failed to start the Spot Instance because the associated Spot Instance request is not in an appropriate state to support start.
```

Linux インスタンスの場合は約 2 分、Windows インスタンスの場合は約 5 分待ってから、インスタンスの起動を再試行してください。

スポットインスタンスを再開できない

スポットインスタンスが正常に休止されたが再開に失敗し、代わりに再起動 (休止状態を維持せずに新たに再起動) した場合、ユーザーデータに次のスクリプトが含まれていたことが原因である可能性があります。

```
/usr/bin/enable-ec2-spot-hibernation
```

起動テンプレートの [ユーザーデータ] フィールドからこのスクリプトを削除し、新しいスポットインスタンスをリクエストしてください。

休止状態を維持せずにインスタンスの再開に失敗した場合でも、インスタンスは stopped 状態から開始するのと同じ方法で起動できることに注意してください。

Amazon EC2 インスタンスを再起動する

インスタンスの再起動は、オペレーティングシステムの再起動と同等です。ほとんどの場合、インスタンスの再起動には数分しかかかりません。

インスタンスを再起動しても、次の状態が維持されます。

- パブリック DNS 名 (IPv4)
- プライベート IPv4 アドレス
- パブリック IPv4 アドレス
- IPv6 アドレス (該当する場合)
- インスタンスストアボリューム上のすべてのデータ

インスタンスの停止および開始 (1分間の最低料金で新しい課金期間が開始される) の場合とは異なり、インスタンスを再起動しても、インスタンスの課金期間が新しく開始されることはありません。

インスタンスの再起動は、ユーザーが開始する (インスタンスを手動で再起動する) か、(自動インスタンス復旧の場合や、再起動を必要とする更新を適用するなど、必要なメンテナンスのためにスケジュールされた再起動イベントに応答する場合には) AWS によって開始されることがあります。

ユーザーが開始する再起動の場合は、インスタンスからオペレーティングシステムの再起動コマンドを実行するのではなく、Amazon EC2 コンソール、CLI、または API を使用することをお勧めします。Amazon EC2 を使用する場合、インスタンスが数分以内に正常にシャットダウンされない場合、Amazon EC2 はハードリブートを実行します。さらに、AWS CloudTrail はインスタンスが再起動されたときの API レコードを作成します。

このトピックでは、ユーザーが開始する再起動を実行する方法について説明します。AWS によって実行される再起動の詳細については、「インスタンスの自動復旧」および「再起動が予定されている Amazon EC2 インスタンスを管理する」を参照してください。

> **Important**
>
> インスタンスに更新プログラムがインストールされている場合、すべての更新プログラムがインストールされるまで、Amazon EC2 コンソールやコマンドラインを使用してインスタンスを再起動またはシャットダウンしないでください。Amazon EC2 コンソールやコマンドラインを使用してインスタンスを再起動またはシャットダウンすると、インスタンスがハードリブートされる恐れがあります。更新ファイルのインストール中にハードリブートされると、インスタンスが不安定な状態になることがあります。

Console

インスタンスを再起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択してから、[Instance state] (インスタンス状態)、[Reboot instance] (インスタンスの再起動) の順に選択します。
4. 確認を求めるメッセージが表示されたら、[Yes, Reboot (再起動する)] を選択します。

   インスタンスは `running` 状態を維持します。

AWS CLI

インスタンスを再起動するには

reboot-instances コマンドを使用します。

```
aws ec2 reboot-instances --instance-ids i-1234567890abcdef0
```

PowerShell

インスタンスを再起動するには

Restart-EC2Instance コマンドレットを使用します。

```
Restart-EC2Instance -InstanceId i-1234567890abcdef0
```

制御された故障注入実験を実行するには

AWS Fault Injection Service を使用して、インスタンスが再起動されたときのアプリケーションの応答をテストできます。詳細については、AWS Fault Injection Service ユーザーガイドを参照してください。

Amazon EC2 インスタンスを終了する

> **Warning**
>
> インスタンスの終了は恒久的で、元に戻すことができません。
> インスタンスを終了すると、そのインスタンスには接続できなくなり、復元することもできません。アタッチされている Amazon EBS ボリュームで、終了時に削除されるように設定されているものも、すべて恒久的に削除され、復元できません。インスタンスストアボリュー

ムに保存されているデータは、すべて恒久的に失われます。詳細については「インスタンスの終了方法」を参照してください。
インスタンスを終了する前に、終了後も保持する必要があるすべてのデータを永続的ストレージにバックアップしたことを確認してください。

不要になったインスタンスは削除できます。これは、インスタンスの終了 と呼ばれます。インスタンスの状態が shutting-down または terminated に変わったら、そのインスタンスへの課金は停止します。

インスタンスを削除した後に、接続または起動することはできません。ただし、同じ AMI を使用して新たなインスタンスを起動することができます。

インスタンスを停止または休止する場合、「Amazon EC2 インスタンスの停止と開始」または「Amazon EC2 インスタンスの休止」を参照してください。詳細については、「インスタンスの状態の違い」を参照してください。

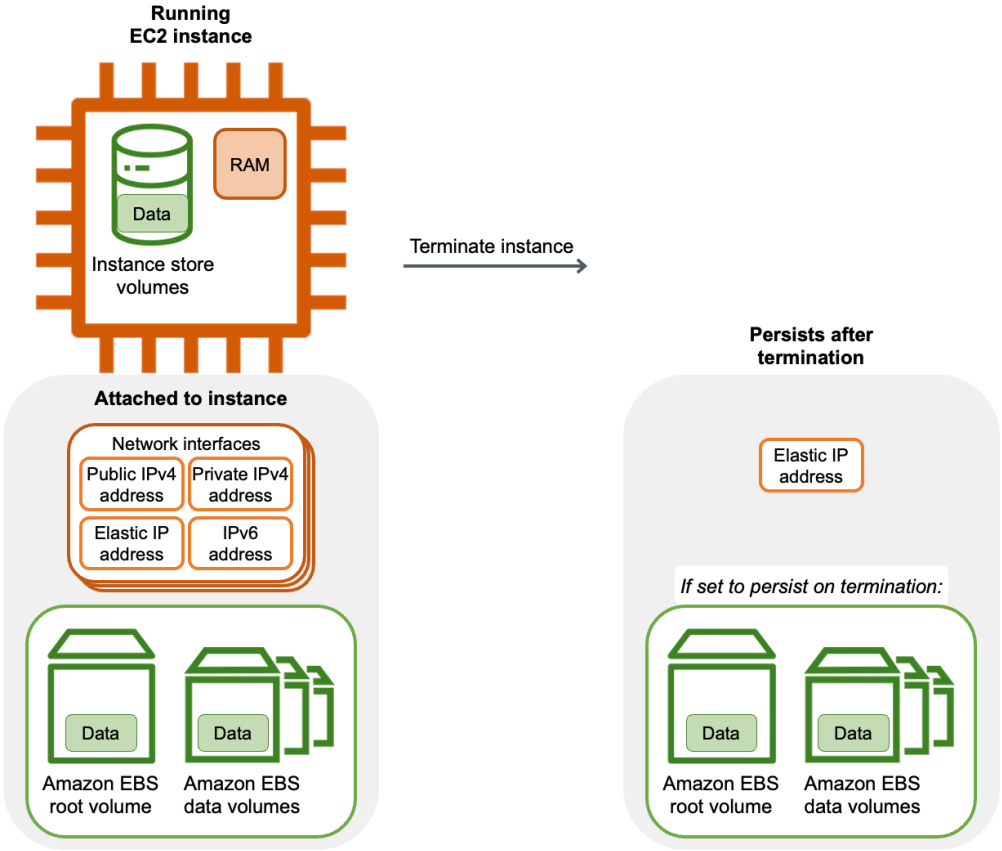
内容

- インスタンスの終了方法
- インスタンスを終了するメソッド
- 正常な OS シャットダウンでインスタンスを終了する
- インスタンスを終了して正常な OS シャットダウンをバイパスする
- インスタンスの終了のトラブルシューティング
- インスタンスの終了保護を変更する
- インスタンスによるシャットダウン動作を変更する
- インスタンスの終了時にデータを保持する

インスタンスの終了方法

インスタンスを終了すると、変更はインスタンスのオペレーティングシステム (OS) レベルで登録され、一部のリソースは失われますが、存続するリソースもあります。

次の図は、Amazon EC2 インスタンスを終了したときに失われるものと残るものを示しています。インスタンスが終了されると、インスタンスストアボリューム上のデータおよびインスタンス RAM に保存されたデータは消去されます。インスタンスに関連付けられた Elastic IP アドレスはデタッチされます。Amazon EBS のルートボリュームおよびデータボリュームの場合、結果は各ボリュームの [終了時に削除] の設定によって異なります。



考慮事項

- データの永続性
  - インスタンスストアボリューム: インスタンスが終了すると、すべてのデータが完全に削除されます。
  - EBS ルートボリューム
    - 起動時にアタッチすると、インスタンスが終了するとデフォルトで削除されます。
    - 起動後にアタッチすると、インスタンスが終了するとデフォルトで保持されます。
  - EBS データボリューム
    - コンソールを使用して起動時にアタッチした場合: インスタンスが終了するとデフォルトで保持されます。
    - CLI を使用して起動時にアタッチした場合: インスタンスが終了するとデフォルトで削除されます。
    - コンソールまたは CLI を使用して起動後にアタッチした場合: インスタンスが終了するとデフォルトで保持されます。

> **Note**
>
> インスタンスの終了時に削除されなかったボリュームは、引き続き料金が発生します。ボリュームが削除されるか、インスタンスの終了時に保持されるように、設定を変更しできます。詳細については、「インスタンスの終了時にデータを保持する」を参照してください。

- 偶発的な終了に対する保護
  - インスタンスを誤って誰かに終了されないようにするには、停止保護を有効にします。
  - インスタンスのシャットダウンの開始時に、インスタンスの停止または終了を制御するには、インスタンスが開始するシャットダウン動作を変更します。
- シャットダウンスクリプト – インスタンスの終了時にスクリプトを実行した場合、シャットダウンスクリプトの実行を保証する方法がないため、インスタンスが正常に終了されない可能性があります。Amazon EC2 はインスタンスを正常にシャットダウンし、システムシャットダウンスクリプトの実行を試みます。ただし、特定のイベント (ハードウェア障害など) がこれらのシステムシャットダウンスクリプトの実行を防止する可能性があります。
- ベアメタルインスタンス – x86 ベアメタルインスタンスは協調シャットダウンをサポートしていません。

インスタンスを削除するとどうなるか

OS レベルでの変更の登録

- API リクエストはボタンのクリックイベントをゲストに送信します。
- ボタンのクリックイベントの結果として、さまざまなシステムサービスが停止します。システムの正常なシャットダウンは、systemd (Linux) またはシステムプロセス (Windows) によって行われます。適切なシャットダウンはハイパーバイザーからの ACPI シャットダウンボタンのクリックイベントによってトリガーされます。
- ACPI シャットダウンが開始されます。
- インスタンスは、適切なシャットダウンプロセスが終了した後にシャットダウンします。設定可能な OS シャットダウン時間はありません。インスタンスはしばらくの間コンソールに表示されたままですが、エントリは自動的に削除されます。

失われるリソース

- インスタンスストアボリュームに保存されているデータは失われます。
- DeleteOnTermination 属性が true に設定されている場合の EBS ルートボリューム。
- DeleteOnTermination 属性が true に設定されている場合の EBS データボリューム(起動時または起動後にアタッチ)。

存続するリソース

- DeleteOnTermination 属性が false に設定されている場合の EBS ルートボリューム。
- DeleteOnTermination 属性が false に設定されている場合の EBS データボリューム(起動時または起動後にアタッチ)。

インスタンスの終了に対するアプリケーションの応答をテスト

AWS Fault Injection Service を使用すると、インスタンスが終了した場合のアプリケーションの応答をテストできます。詳細についてはAWS Fault Injection Service ユーザーガイドを参照してください。

インスタンスを終了するメソッド

> **Warning**
>
> インスタンスの終了は恒久的で、元に戻すことができません。
> インスタンスを終了すると、そのインスタンスには接続できなくなり、復元することもできません。アタッチされている Amazon EBS ボリュームで、終了時に削除されるように設定されているものも、すべて恒久的に削除され、復元できません。インスタンスストアボリュームに保存されているデータは、すべて恒久的に失われます。詳細については「インスタンスの終了方法」を参照してください。
> インスタンスを終了する前に、終了後も保持する必要があるすべてのデータを永続的ストレージにバックアップしたことを確認してください。

ユーザー主導のインスタンス終了を実行する方法は 4 つあります。デフォルトの終了、OS シャットダウンをスキップした終了、強制終了、OS シャットダウンをスキップした強制終了という 4 つの方法です。次のテーブルでは、終了方法の主な違いが比較されます。

> **Note**
>
> 終了保護が有効になっている場合、インスタンスを終了することはできません。詳細については、「インスタンスの終了保護を変更する」を参照してください。

終了メソッド	主な目的	ユースケース	CLI コマンド
デフォルトの終了	正常な OS シャットダウンの試行による通常のインスタンスシャットダウン。	一般的なインスタンスの終了。	`aws ec2 terminate-instances \ --instance-id i-1234567890abcdef0`
OS シャットダウンをスキップして終了する	インスタンスを終了するとき、正常な OS シャットダウンがバイパスされます。	正常な OS シャットダウンをバイパスする必要がある場合。	`aws ec2 terminate-instances \ --instance-id i-1234567890abcdef0 \ --skip-os-shutdown`
強制終了	スタックしたインスタンスを処理します。最初にデフォルトの終了を試みます。インスタンスが終了しない場合、インスタンスを強制的に終了します。	インスタンスが shutting-down 状態でスタックしている場合。	`aws ec2 terminate-instances \ --instance-id i-1234567890abcdef0 \ --force`
OS シャットダウ	インスタンスを終了するとき、強制終了し、正常な OS シャットダウンをバイパスします。	強制終了し、正常な OS シャットダウンをバイパスする必要がある場合。	`aws ec2 terminate-instances \`

終了メソッド	主な目的	ユースケース	CLI コマンド
ンをスキップして強制終了する			`--instance-id i-1234567890abcdef0 \ --force \ --skip-os-shutdown`

各メソッドの使用方法の手順については、次の内容を参照してください。

- 正常な OS シャットダウンでインスタンスを終了する
- インスタンスを終了して正常な OS シャットダウンをバイパスする
- インスタンスを強制終了する

正常な OS シャットダウンでインスタンスを終了する

デフォルトの終了メソッドを使用してインスタンスを終了できます。このメソッドには、正常な OS シャットダウンの試行が含まれます。詳細については、「インスタンスを終了するメソッド」を参照してください。

Console

デフォルトの終了メソッドを使用してインスタンスを終了する方法

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択し、[インスタンスの状態]、[インスタンスの終了 (削除)] の順に選択します。
4. 確認を求めるメッセージが表示されたら、[終了 (削除)] を選択します。
5. インスタンスの終了後、インスタンスはしばらくの間、`terminated` の状態で表示されたままになります。

   終了に失敗した場合や、終了したインスタンスが数時間以上表示されている場合は、「表示されているインスタンスを削除する」を参照してください。

AWS CLI

デフォルトの終了メソッドを使用してインスタンスを終了する方法

terminate-instances コマンドを使用します。

```
aws ec2 terminate-instances --instance-ids i-1234567890abcdef0
```

PowerShell

デフォルトの終了メソッドを使用してインスタンスを終了する方法

Remove-EC2Instance コマンドレットを使用します。

```
Remove-EC2Instance -InstanceId i-1234567890abcdef0
```

インスタンスを終了して正常な OS シャットダウンをバイパスする

インスタンスを終了するとき、正常な OS シャットダウンをバイパスできます。詳細については、「インスタンスを終了するメソッド」を参照してください。

Console

インスタンスを終了して正常な OS シャットダウンをバイパスする方法

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択し、[インスタンスの状態]、[インスタンスの終了 (削除)] の順に選択します。
4. [OS シャットダウンをスキップ] で、[OS シャットダウンをスキップ] チェックボックスをオンにします。コンソールにこのオプションが表示されない場合、現在のリージョンのコンソールではまだ利用できません。ただし、AWS CLI または SDK を使用してこの機能にアクセスするか、コンソールで別のリージョンを試すことができます。
5. [終了 (削除)] を選択します。
6. インスタンスの終了後、インスタンスはしばらくの間、terminated の状態で表示されたままになります。

   終了に失敗した場合や、終了したインスタンスが数時間以上表示されている場合は、「表示されているインスタンスを削除する」を参照してください。

AWS CLI

インスタンスを終了して正常な OS シャットダウンをバイパスする方法

`--skip-os-shutdown` で「terminate-instances」コマンドを使用します。

```
aws ec2 terminate-instances \
    --instance-ids i-1234567890abcdef0 \
    --skip-os-shutdown
```

PowerShell

インスタンスを終了して正常な OS シャットダウンをバイパスする方法

`-SkipOsShutdown $true` で「Remove-EC2Instance」コマンドレットを使用します。

```
Remove-EC2Instance `
    -InstanceId i-1234567890abcdef0 `
    -SkipOsShutdown $true
```

インスタンスの終了のトラブルシューティング

リクエスタには `ec2:TerminateInstances` を呼び出すアクセス許可が必要です。詳細については、インスタンスメタデータを使用する際のポリシーの例に関するページを参照してください。

インスタンスを終了して別のインスタンスを起動する場合、通常 EC2 フリート や Amazon EC2 Auto Scaling などの機能を通じて自動スケーリングを設定している可能性があります。詳細については、「インスタンスが自動的に起動または終了される」を参照してください。

> **Note**
>
> 終了保護が有効になっている場合、インスタンスを終了することはできません。詳細については、「インスタンスの終了保護を変更する」を参照してください。

インスタンスの `shutting-down` 状態が通常よりも長い場合は、強制終了を試みることができます。インスタンスの `shutting-down` 状態が続く場合、Amazon EC2 サービス内の自動プロセスによってクリーンアップ (終了) されるはずです。詳細については、「インスタンスの削除の遅延」を参照してください。

インスタンスの終了保護を変更する

Amazon EC2 API の使用中にインスタンスが誤って終了するのを防ぐには、`TerminateInstances` を直接呼び出すか、別のインターフェイス (Amazon EC2 コンソールなど) を使用するかにかかわらず、インスタンスの終了保護を有効にします。`DisableApiTermination` 属性は、インスタンスを終了可能にするかどうかを制御します。デフォルトでは、インスタンスの削除保護は無効になっています。この属性の値は、インスタンスの起動時、もしくはインスタンスの実行中または停止時に、設定することができます。

`InstanceInitiatedShutdownBehavior` 属性が `terminate` に設定された場合、`DisableApiTermination` 属性は (システムシャットダウン用のオペレーティングシステムコマンドを使用するなどして) インスタンスからシャットダウンを開始して、インスタンスを終了できます。詳細については、「インスタンスによるシャットダウン動作を変更する」を参照してください。

考慮事項

- 終了保護を有効にしても、インスタンスにインスタンスを終了する予定されたイベントがある場合、AWS によるインスタンスの終了は防げません。
- 終了保護を有効にしても、インスタンスが異常な場合やスケールインイベント中に Amazon EC2 Auto Scaling がインスタンスを終了することは防げません。スケールイン時に Auto Scaling グループが特定のインスタンスを終了できるかどうかを制御するには、インスタンスのスケールイン保護を使用します。Auto Scaling グループが異常なインスタンスを終了できるかどうかを制御するには、ReplaceUnhealthy スケーリングプロセスを中断します。
- スポットインスタンス の削除保護を有効にすることはできません。

Console

起動時にインスタンスに対する終了保護を有効にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ダッシュボードで、[Launch Instance (インスタンスの起動)] を選択してください。
3. [Advanced Details] (高度な詳細) を展開します。[終了保護] で、[有効化] を選択します。
4. インスタンスの詳細の指定が終了したら、[インスタンスの起動] を選択します。

インスタンスに対する終了保護を変更するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択してください。
4. インスタンスを選択してから、[アクション]、[インスタンスの設定]、[削除保護の変更] の順に選択します。
5. [終了保護] で、[有効化] を選択またはクリアします。
6. [保存] を選択します。

AWS CLI

インスタンスに対する終了保護を有効にするには

modify-instance-attribute コマンドを使用します。

```
aws ec2 modify-instance-attribute \
    --instance-id i-1234567890abcdef0 \
    --disable-api-termination
```

インスタンスに対する終了保護を無効化するには

modify-instance-attribute コマンドを使用します。

```
aws ec2 modify-instance-attribute \
    --instance-id i-1234567890abcdef0 \
    --no-disable-api-termination
```

PowerShell

インスタンスに対する終了保護を有効にするには

Edit-EC2InstanceAttribute コマンドレットを使用します。

```
Edit-EC2InstanceAttribute `
    -InstanceId i-1234567890abcdef0 `
    -DisableApiTermination $true
```

インスタンスに対する終了保護を無効化するには

Edit-EC2InstanceAttribute コマンドレットを使用します。

```
Edit-EC2InstanceAttribute `
    -InstanceId i-1234567890abcdef0 `
    -DisableApiTermination $false
```

終了保護を使用して複数のインスタンスを終了する

複数のアベイラビリティーゾーンにある複数のインスタンスを同じリクエストで同時に終了する場合、指定した中に終了保護が有効になっているインスタンスが 1 つ以上存在すると、そのリクエストは失敗し次のような結果が返されます。

- 保護されたインスタンスと同じアベイラビリティーゾーンにあるインスタンスは終了されません。
- 保護されたインスタンスが他に存在しないアベイラビリティーゾーンでは、特定のインスタンスを正常に終了することができます。

例

2 つのアベイラビリティーゾーンに次の 4 つのインスタンスがあるとします。

インスタンス	アベイラビリティーゾーン	終了保護
インスタンス 1	AZ A	Disabled
インスタンス 2		Disabled
インスタンス 3	AZ B	Enabled
インスタンス 4		Disabled

これらのインスタンスすべてを同じリクエストで終了しようとすると、リクエストは以下のような結果とともに失敗を返します。

- インスタンス 1 とインスタンス 2 は、どちらのインスタンスも終了保護が有効化されていないため、正常に終了します。
- インスタンス 3 とインスタンス 4 は、インスタンス 3 で終了保護が有効になっているため、終了に失敗します。

インスタンスによるシャットダウン動作を変更する

> **Warning**
>
> インスタンスの終了は恒久的で、元に戻すことができません。
> インスタンスを終了すると、そのインスタンスには接続できなくなり、復元することもできません。アタッチされている Amazon EBS ボリュームで、終了時に削除されるように設定されているものも、すべて恒久的に削除され、復元できません。インスタンスストアボリュームに保存されているデータは、すべて恒久的に失われます。詳細については「インスタンスの終了方法」を参照してください。
> インスタンスを終了する前に、終了後も保持する必要があるすべてのデータを永続的ストレージにバックアップしたことを確認してください。

デフォルトで、Amazon EBS backed インスタンスからシャットダウンを開始すると (shutdown や poweroff などのコマンドを使用すると)、インスタンスは停止します。インスタンスの `InstanceInitiatedShutdownBehavior` 属性を変更すると、この動作を変更して、停止ではなく終了するようにできます。インスタンスの実行中または停止中に、この属性を変更できます。

halt コマンドはシャットダウンを開始しません。使用した場合、インスタンスは終了しません。代わりに、CPU が HLT 状態になり、インスタンスは実行されたままになります。

> **Note**
>
> `InstanceInitiatedShutdownBehavior` 属性は、インスタンス自体のオペレーティングシステムからシャットダウンを実行した場合にのみ適用されます。`StopInstances` API または Amazon EC2 コンソールを使用してインスタンスを停止した場合、適用されません。

Console

インスタンスによって開始されたシャットダウン動作を変更するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択します。
4. [Actions (アクション)]、[Instance settings (インスタンスの設定)]、[Change shutdown behavior (シャットダウン動作の変更)] の順に選択します

[シャットダウン動作] に現在の動作が表示されます。

5. 動作を変更するには、[シャットダウン動作] で [停止] または [終了] を選択します。
6. [保存] を選択します。

AWS CLI

インスタンスによって開始されたシャットダウン動作を変更するには

modify-instance-attribute コマンドを使用します。

```
aws ec2 modify-instance-attribute \
    --instance-id i-1234567890abcdef0 \
    --instance-initiated-shutdown-behavior terminate
```

PowerShell

インスタンスによって開始されたシャットダウン動作を変更するには

Edit-EC2InstanceAttribute コマンドレットを使用します。

```
Edit-EC2InstanceAttribute `
    -InstanceId i-1234567890abcdef0 `
    -InstanceInitiatedShutdownBehavior terminate
```

インスタンスの終了時にデータを保持する

Amazon EC2 インスタンスが終了すると、データをインスタンスストアボリュームまたは Amazon EBS ボリュームに保存できます。このトピックでは、インスタンスの終了後もデータが保持されるようにする方法について説明します。

インスタンスの終了がルートボリュームとデータボリュームに与える影響

インスタンスストアボリューム

インスタンスが終了すると、インスタンスストアボリュームは自動的に削除され、データが失われます。インスタンスのライフタイムを超えてこのデータを保持するには、インスタンスを終了する前に、Amazon EBS ボリューム、Amazon S3 バケット、Amazon EFS ファイルシステムなどの永続的なストレージに手動でコピーしてください。詳細については、「Amazon EC2 インスタンスのストレージオプション」を参照してください。

Amazon EBS ボリューム

インスタンスが終了すると、各ボリュームの `DeleteOnTermination` 属性の値に応じて EBS ボリュームは削除または保持されます。

- [はい] (コンソール) / `true` (CLI) – インスタンスが終了すると、ボリュームは削除されます。
- [いいえ] (コンソール) / `false` (CLI) – インスタンスが終了すると、ボリュームは保持されます。保持されたボリュームは、引き続き料金が発生します。

> **Note**
>
> インスタンスが終了した後、保持されたボリュームのスナップショットを作成するか、別のインスタンスにアタッチできます。不要な料金の発生を回避するには、ボリュームを削除する必要があります。

EBS ボリュームのデフォルトの削除動作

デフォルト `DeleteOnTermination` 値はボリュームタイプ、起動時または起動後にボリュームがアタッチされたか、ボリュームのアタッチに使用された方法 (コンソールまたは CLI) によって異なります。

ボリュームタイプ	アタッチ時	アタッチ方法	インスタンス終了時のデフォルの動作
ルートボリューム	起動時	コンソールまたは CLI	削除
ルートボリューム	After launch	コンソールまたは CLI	Preserve
データボリューム	起動時	コンソール	Preserve
データボリューム	起動時	CLI	削除
データボリューム	After launch	コンソールと CLI	Preserve

ボリューム永続性設定を確認する

EBS ボリュームの起動時のデフォルト値は、AMI で設定された `DeleteOnTermination` 属性によって決まります。インスタンスの起動時に値を変更し、AMI 設定をオーバーライドできます。イ

ンスタンスを起動した後、DeleteOnTermination 属性のデフォルト設定を確認することをお勧めします。

インスタンスの終了時に Amazon EBS ボリュームが削除されるかどうか確認する方法

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択します。
4. [ストレージ] タブを選択します。
5. [ブロックデバイス] で、右にスクロールして [終了時に削除] の列を確認します。
   - [はい] の場合、ボリュームはインスタンスの終了時に削除されます。
   - [いいえ] の場合、ボリュームはインスタンスの終了時に削除されません。削除されていないボリュームは、引き続き料金が発生します。

起動時にルートボリュームが存続するように変更する

インスタンスを起動するときに、EBS ルートボリュームの DeleteOnTermination 属性を変更できます。データボリュームに次の手順を使用することもできます。

Console

起動時にインスタンスのルートボリュームが存続するように変更するには

1. 手順に従ってインスタンスを起動しますが、次のステップを完了してルートボリュームを存続するように変更するまでインスタンスを起動しないでください。
2. [ストレージを設定] ペインで [アドバンスト] を選択します。
3. [EBS ボリューム] で、ルートボリュームの情報を展開します。
4. [終了時に削除] には、[いいえ] を選択します。
5. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択してください。詳細については、「コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する」を参照してください。

AWS CLI

起動時にインスタンスのルートボリュームが存続するように変更するには

run-instances コマンドを使用して、ブロックデバイスマッピングの DeleteOnTermination の値を変更します。

--block-device-mappings オプションを追加します。

```
--block-device-mappings file://mapping.json
```

mapping.json では、デバイス名を指定し (例: /dev/sda1 または /dev/xvda)、DeleteOnTermination で false を指定します。

```
[
  {
    "DeviceName": "device_name",
    "Ebs": {
      "DeleteOnTermination": false
    }
  }
]
```

PowerShell

起動時にインスタンスのルートボリュームが存続するように変更するには

New-EC2Instance コマンドレットを使用して、ブロックデバイスマッピングの DeleteOnTermination の値を 変更します。

-BlockDeviceMapping オプションを追加します。

```
-BlockDeviceMapping $bdm
```

bdm では、デバイス名を指定し (例: /dev/sda1 または /dev/xvda)、DeleteOnTermination で false を指定します。

```
$ebd = New-Object -TypeName Amazon.EC2.Model.EbsBlockDevice
$ebd.DeleteOnTermination = false
$bdm = New-Object -TypeName Amazon.EC2.Model.BlockDeviceMapping
$bdm.DeviceName = "/dev/sda1"
$bdm.Ebs = $ebd
```

実行中のインスタンスのルートボリュームが存続するように変更する

実行中のインスタンスの EBS ルートボリュームが永続するように、変更できます。データボリュームに次の手順を使用することもできます。

AWS CLI

ルートボリュームを永続的に変更するには

modify-instance-attribute コマンドを使用します。

```
aws ec2 modify-instance-attribute \
    --instance-id i-1234567890abcdef0  \
    --block-device-mappings file://mapping.json
```

mapping.json では、デバイス名を指定し (例: /dev/sda1 または /dev/xvda)、--DeleteOnTermination で false を指定します。

```
[
  {
    "DeviceName": "device_name",
    "Ebs": {
      "DeleteOnTermination": false
    }
  }
]
```

PowerShell

ルートボリュームを永続的に変更するには

Edit-EC2InstanceAttribute コマンドレットを使用します。

-BlockDeviceMapping オプションを追加します。

```
-BlockDeviceMapping $bdm
```

bdm では、デバイス名を指定し (例: /dev/sda1 または /dev/xvda)、DeleteOnTermination で false を指定します。

```
$ebd = New-Object -TypeName Amazon.EC2.Model.EbsBlockDevice
```

```
$ebd.DeleteOnTermination = false
$bdm = New-Object -TypeName Amazon.EC2.Model.BlockDeviceMapping
$bdm.DeviceName = "/dev/sda1"
$bdm.Ebs = $ebd
```

インスタンスのリタイア

インスタンスをホストしている基盤のハードウェアで回復不可能な障害が検出されると、AWS によってインスタンスのリタイアが予定されます。インスタンスのルートボリュームタイプはインスタンスのリタイア動作を決定します。

- インスタンスのルートボリュームが Amazon EBS ボリュームである場合、インスタンスは停止されますが、その後いつでも再び起動できます。停止したインスタンスを開始すると、新しいハードウェアに移行されます。
- インスタンスのルートボリュームがインスタンスストアボリュームである場合、インスタンスは終了され、再び使用することはできません。

インスタンスイベントのタイプの詳細については Amazon EC2 インスタンスの予定されているイベントを参照してください。

内容

- リタイアが予定されているインスタンスの特定
- リタイアが予定されている EBS-バック インスタンスに対して実行するアクション
- リタイアが予定されている instance-store バック インスタンスに対して実行するアクション

リタイアが予定されているインスタンスの特定

インスタンスのリタイアが予定された場合、イベントの前に、当該のインスタンス ID とリタイア日を記載したメールが送信されます。廃止が予定されているインスタンスを確認することもできます。

> **Important**
>
> インスタンスの廃止が予定されている場合は、インスタンスが既に到達不能になっている可能性もあるため、できるだけ早急に措置を講じることが推奨されます。詳細については「Check if your instance is reachable」を参照してください。

廃止が予定されているインスタンスを特定する方法

- アカウントの連絡先になっている E メールをモニタリングする
- インスタンスを確認する

アカウントの連絡先になっている E メールをモニタリングする

インスタンスの廃止が予定されている場合、アカウントの主要連絡先とオペレーションの連絡先に、廃止前に E メールが届きます。この E メールには、インスタンス ID と廃止予定日が記載されています。詳細については、「AWS アカウント管理 リファレンスガイド」の「Update the primary contact for your AWS account」および「Update the alternate contacts for your AWS account」を参照してください。

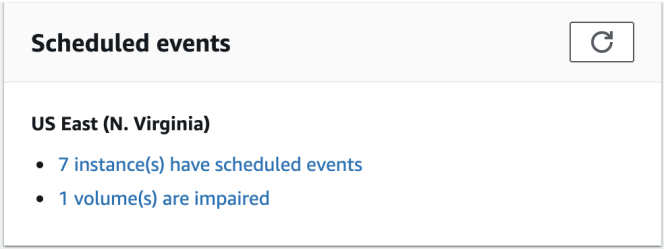
インスタンスを確認する

定期的なチェックを行っていない E メールアカウントがある場合、インスタンス廃止の通知を見逃している可能性ががあります。インスタンスの廃止が予定されているかどうかは随時確認できます。

Console

廃止が予定されているインスタンスを確認するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[EC2 ダッシュボード] を選択してください。[スケジュールされたイベント] に、Amazon EC2 インスタンスおよびボリュームに関連付けられたイベントが、リージョン別に整理されて表示されます。



3. インスタンスに予定されたイベントが表示されている場合はリージョン名の下のリンクを選択して [Events] ページにアクセスします。
4. [Events (イベント)] ページにはすべてのリソースとそれに関連付けられたイベントが一覧表示されます。リタイアが予定されているインスタンスを表示するには1 つ目のフィルタリストから [Instance resources] を選択し、2 つ目のフィルタリストから [Instance stop or retirement] を選択してください。

5. フィルタの結果にインスタンスのリタイアが予定されていることが表示されたら、当該のインスタンスを選択し、詳細ペインの [Start time] フィールドの日時を書き留めます。これがインスタンスのリタイア日です。

AWS CLI

廃止が予定されているインスタンスを特定するには

次の describe-instance-status コマンドを使用します。インスタンスを実行している各リージョンで同じ作業を繰り返します。

```
aws ec2 describe-instance-status --filters Name=event.code,Values=instance-
retirement
```

PowerShell

廃止が予定されているインスタンスを特定するには

Get-EC2InstanceStatus コマンドレットを使用します。インスタンスを実行している各リージョンで同じ作業を繰り返します。

```
Get-EC2InstanceStatus -Filter @{Name="event.code"; Values="instance-retirement"}
```

リタイアが予定されている EBS-バック インスタンスに対して実行するアクション

リタイアが予定されているインスタンスのデータを保持するには次のいずれかのアクションを実行できます。予期しないダウンタイムやデータ消失を防ぐために、インスタンスのリタイア日より前にこのアクションを実行することが重要です。

Linux インスタンスにおいて、インスタンスが EBS とインスタンスストアのどちらからバッキングされているかわからない場合は「Amazon EC2 インスタンスのルートボリューム」を参照してください。

インスタンスが到達可能かどうかを確認する

インスタンスのリタイアが予定されていることが通知された場合はできるだけ早く以下のアクションを実行することをお勧めします。

- インスタンスに接続するか、インスタンスに ping を実行して、インスタンスが到達可能かどうかを確認します。

- インスタンスに到達可能な場合はスケジュールされたリタイア日の前の適切なタイミングで、その影響が最小限であるときにインスタンスを停止/開始するよう計画する必要があります。インスタンスの停止と開始、インスタンスを停止したときに予想される影響 (インスタンスに関連付けられたパブリック IP アドレス、プライベート IP アドレス、および Elastic IP アドレスへの影響など) の詳細についてはAmazon EC2 インスタンスの停止と開始を参照してください。インスタンスを停止して起動すると、インスタンスストアボリュームのデータが失われることに注意してください。
- インスタンスに到達できない場合は直ちにアクションを実行し、停止/開始を実行してインスタンスを復元する必要があります。
- またはインスタンスを削除する場合はできるだけ早く削除するよう計画し、インスタンスの料金が発生しないようにしてください。

インスタンスのバックアップを作成する

バックアップが作成されるように、インスタンスから EBS-Backed AMI を作成します。データの整合性を確保するにはAMI を作成する前にインスタンスを停止します。予定されたリタイア日を待つことができます。その日になるとインスタンスが停止できます。またはリタイア日の前に自分でインスタンスを停止します。インスタンスはいつでも再開できます。詳細については「Amazon EBS-backed AMI を作成する」を参照してください。

代わりのインスタンスを起動する

インスタンスから AMI を作成した後、AMI を使用して代替インスタンスを起動できます。Amazon EC2 コンソールから、新しい AMI を選択して、[AMI からインスタンスを起動] を選択します。インスタンスのパラメータを設定し、[インスタンスを起動] を選択します。各フィールドの詳細については「コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する」を参照してください。

リタイアが予定されている instance-store バック インスタンスに対して実行するアクション

リタイアが予定されているインスタンスのデータを保持するには次のいずれかのアクションを実行できます。予期しないダウンタイムやデータ消失を防ぐために、インスタンスのリタイア日より前にこのアクションを実行することが重要です。

> **Warning**
>
> インスタンスがインスタンスストアのルートボリュームを持っていて、かつ廃止日を過ぎている場合、インスタンスが終了し、インスタンスやインスタンスに格納されていたデータを復元できなくなります。インスタンスストアボリュームのデータは、インスタンスのルートボリュームタイプにかかわらず、EBS ルートボリュームを持つインスタンスにボリュームがアタッチされている場合でも、インスタンスが廃止されると失われます。

インスタンスが到達可能かどうかを確認する

インスタンスのリタイアが予定されていることが通知された場合はできるだけ早く以下のアクションを実行することをお勧めします。

- インスタンスに接続するか、インスタンスに ping を実行して、インスタンスが到達可能かどうかを確認します。
- インスタンスに到達できない場合、インスタンスを復元するために実行できることはほとんどありません。詳細については「到達できない Amazon EC2 インスタンスのトラブルシューティング」を参照してください。AWS は予定されたリタイア日にインスタンスを削除するため、到達不能なインスタンスについてはお客様自身ですぐにインスタンスを削除できます。

代わりのインスタンスを起動する

AMI ツールを使用してインスタンスから Amazon S3-backed AMI を作成します (Amazon S3-backed AMI を作成するを参照)。Amazon EC2 コンソールから、新しい AMI を選択して、[AMI からインスタンスを起動] を選択します。インスタンスのパラメータを設定し、[インスタンスを起動] を選択します。各フィールドの詳細については「コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する」を参照してください。

インスタンスを EBS-backed インスタンスに変換する

データを EBS ボリュームに転送し、ボリュームのスナップショットを作成した後、スナップショットから AMI を作成します。新しい AMI から代替インスタンスを起動できます。詳細については「Amazon S3-backed AMI を EBS-backed AMI に変換する」を参照してください。

インスタンスの自動復旧

> **Important**
>
> このセクションではEC2 インスタンスで復旧メカニズムをプロアクティブに設定する方法について説明します。これらの復旧メカニズムは がシステムステータスチェックが失敗する原因となる基盤となるハードウェアまたはソフトウェアの問題AWSを検出したときに、インスタンスの可用性を復元するように設計されています。インスタンスへのアクセスで現在問題が発生している場合は「EC2 インスタンスのトラブルシューティング」を参照してください。

基盤となるハードウェアの障害によりインスタンスが使用できないと AWS により判断された場合、インスタンスの耐障害性を設定して可用性を復元できるメカニズムとして簡易自動復旧または Amazon CloudWatch アクションベスの復旧のどちらかの方法で行われます。インスタンスの可用性の復元はインスタンスの復旧とも呼ばれます。

インスタンス復旧プロセス中に、 AWSは基盤となるハードウェアまたはソフトウェアの問題があるホストから別のホストにインスタンスを移動しようとします。インスタンスの復旧が成功すると、インスタンスには予期しない再起動として表示されます。インスタンスの復旧が発生したかどうかを確認できます。

復旧プロセスが失敗した場合、基盤となるハードウェアまたはソフトウェアの問題により、インスタンスがホストで引き続き実行される可能性があります。この場合、手動による介入が必要です。インスタンスのシステムステータスチェックの失敗が続く場合はインスタンスを手動で停止および開始することをお勧めします。インスタンスを開始すると、通常、インスタンスは基盤となる新しいホストコンピュータに移行され、新しいパブリック IPv4 アドレスが割り当てられます。ただし、インスタンスがパブリック IPv4 アドレスを保持する自動インスタンスリカバリとは異なり、再起動されたインスタンスはElastic IP アドレスがない限り、新しいパブリック IPv4 アドレスを受け取ります。

自動復旧メカニズムを活用するにはシステムステータスチェックが失敗する前に、インスタンスで事前に設定する必要があります。サポート対象のインスタンスを起動すると、簡易自動復旧がデフォルトで有効になります。オプションで、起動後の Amazon CloudWatch アクションベースの復旧を設定できます。これらのメカニズムのいずれかを設定すると、インスタンスの耐障害性が向上します。

簡易自動復旧と Amazon CloudWatch アクションベースの復旧はサポートされているインスタンスでのみ使用できます。詳細については簡易自動復旧の要件と制限およびCloudWatch アクションベースの復旧の要件と制限を参照してください。

> **Warning**
>
> 基盤となるハードウェアまたはソフトウェアの問題により がインスタンスをAWS復旧する場合、次の結果に注意してください。揮発性メモリ (RAM) に保存されているデータは失われ、オペレーティングシステムの稼働時間はゼロから開始されます。さらに、CloudWatch アクションベースの復旧ではインスタンスストアボリュームのデータも失われます。データ損失を防ぐために、重要なデータのバックアップを定期的に作成することをお勧めします。Amazon EC2 インスタンスのバックアップと復旧のベストプラクティスの詳細については「Amazon EC2 のベストプラクティス」を参照してください。
> 自動インスタンス復旧メカニズムは個々のインスタンス用に設計されています。回復力のあるシステムの構築に関するガイダンスについては「」を参照してください回復力のあるシステムを構築する。

トピック

- 自動インスタンス復旧の主な概念
- 簡易自動復旧と CloudWatch アクションベースの復旧の違い
- 回復力のあるシステムを構築する
- 自動インスタンス復旧が発生したかどうかを確認する
- Amazon EC2 インスタンスで簡易自動復旧を設定する
- EC2 インスタンスでの CloudWatch アクションベースの復旧の設定

自動インスタンス復旧の主な概念

自動インスタンスリカバリは基盤となるハードウェアまたはソフトウェアの障害が発生したときにインスタンスの可用性を自動的に復元し、Amazon EC2 EC2 機能です。

自動インスタンス復旧の主な概念は次のとおりです。

設定オプション

自動インスタンス復旧をサポートするように 2 つのメカニズムを設定できます。

- 簡素化された自動リカバリ: サポートされているインスタンスではデフォルトで有効になっています。
- CloudWatch アクションベースの復旧: サポートされているインスタンスで手動設定が必要です。

システムステータスのチェック

システムステータスチェックはEC2 インスタンスが実行されるAWSインフラストラクチャを自動的にモニタリングします。

- システムステータスチェックが失敗すると、 は自動インスタンス復旧AWSを開始し、影響を受けるインスタンスを別のハードウェアに移行しようとします。
- システムステータスチェックが失敗した場合はホストのハードウェアまたはソフトウェアに問題があることを示し、インスタンス自体に問題がないことを示します。自動インスタンス復旧ではシステムステータスチェックに失敗したインスタンスを復旧できます。ただし、インスタンスステータスチェックのみが失敗した場合、自動インスタンス復旧は動作しません。
- インスタンスとシステムのステータスチェックの違いについては「ステータスチェックのタイプ」を参照してください。

基盤となるハードウェアまたはソフトウェアの問題の例

システムステータスチェックが失敗する原因となるハードウェアまたはソフトウェアの問題にはネットワーク接続の喪失、システム電源の喪失、物理ホスト上のソフトウェアの問題、およびネットワークの到達可能性に影響を与える物理ホスト上のハードウェアの問題が含まれます。

復旧されたインスタンスの特徴

復元されたインスタンスは失われた要素を除いて、元のインスタンスと同じです。

保存済み要素：

- [インスタンス ID]
- パブリック IP アドレス、プライベート IP アドレス、Elastic IP アドレス
- インスタンスメタデータ
- 配置グループ
- アタッチされた EBS ボリューム
- アベイラビリティーゾーン

失われた要素：

- 揮発性メモリ (RAM) に保存されるデータ
- インスタンスストアボリュームに保存されているデータ (CloudWatch アクションベースの復旧にのみ適用)
- オペレーティングシステムの稼働時間がゼロにリセットされる

CloudWatch によるシステムステータスチェックのモニタリング

CloudWatch の StatusCheckFailed_System メトリクスはシステムステータスチェックが成功したか失敗したかを示します。

メトリクス値

- 0 – システムステータスチェックに合格しました。
- 1 – 失敗したシステムステータスチェック

Health Dashboard でのイベント。

自動インスタンス復旧の試行中に、 は設定された復旧メカニズムとその結果Health Dashboardに基づいてイベントを AWSに送信します。

- 簡易自動復旧
  - 成功イベント: `AWS_EC2_SIMPLIFIED_AUTO_RECOVERY_SUCCESS`
  - 失敗イベント: `AWS_EC2_SIMPLIFIED_AUTO_RECOVERY_FAILURE`
- CloudWatch アクションに基づく復旧
  - 成功イベント: `AWS_EC2_INSTANCE_AUTO_RECOVERY_SUCCESS`
  - 失敗イベント: `AWS_EC2_INSTANCE_AUTO_RECOVERY_FAILURE`

簡易自動復旧と CloudWatch アクションベースの復旧の違い

次の表は簡易自動復旧と CloudWatch アクションベースの復旧の主な違いを比較したものです。

比較ポイント	簡易自動復旧	CloudWatch アクションに基づく復旧
設定	サポートされているインスタンスでデフォルトで有効	CloudWatch アラームとアクションの手動設定が必要です
柔軟性	によって管理される復旧動作を修正しました。 AWS	カスタマイズ可能なアクションと条件
Notification	Health Dashboard での通知	SNS によるカスタマイズ可能な通知
ベアメタルインスタンスサイズ	除外	含まれる

比較ポイント	簡易自動復旧	CloudWatch アクションに基づく復旧
インスタンスストアボリュームはインスタンスの起動時にのみアタッチされます。	起動時にインスタンスストアボリュームをアタッチするインスタンスではサポートされていません	選択したインスタンスタイプでサポートされています。インスタンスストアボリュームのデータはインスタンスの復旧中に失われることに注意してください。
復旧時間	標準復旧の試行	簡易自動復旧よりも高速な復旧の試行
移行中のホストの問題の解決	移行がキャンセルされ、インスタンスが元のホストに留まる可能性がある	新しいホストへの移行の継続
Cost	追加料金	CloudWatch 料金が発生する可能性があります

回復力のあるシステムを構築する

簡易自動復旧と CloudWatch アクションベースの復旧は個々のインスタンスの可用性を維持するのに効果的ですが、AWS では正常なインスタンスへのトラフィックのフェイルオーバーを可能にする高可用性アーキテクチャを実装することをお勧めします。

これを実現するにはElastic Load Balancing (受信トラフィックを複数の EC2 インスタンスに分散させる) や Amazon EC2 Auto Scaling (需要と状態に基づいてインスタンス数を自動的に調整する) などの AWS サービスの使用を検討してください。

EC2 インスタンスを使用して回復力と耐障害性を備えたシステムを構築する方法の詳細については以下のリソースを参照してください。

- AWS YouTube チャンネルの「基本に戻る: EC2 での障害に対する設計」
- のディザスタリカバリ (DR) アーキテクチャ、AWSパート I: AWS クラウドにおけるリカバリ
- Application Load Balancer のユーザーガイド
- Amazon EC2 Auto Scaling ユーザーガイド

- REL11-BP02 信頼性の柱AWS Well-Architected フレームワークの正常なリソースにフェイルオーバーする

自動インスタンス復旧が発生したかどうかを確認する

インスタンスがオフラインになっていて、予期せず再起動されたように見える場合は基盤となるハードウェアまたはソフトウェアの問題に応じて、インスタンスが自動復旧された可能性があります。これを確認するには でインスタンスの自動復旧イベントを確認しますHealth Dashboard。StatusCheckFailed_System Amazon CloudWatch メトリクスを確認することで、インスタンスの基盤となるハードウェアまたはソフトウェアの問題が検出されたかどうかを確認することもできます。

でイベントを確認する Health Dashboard

自動インスタンス復旧の試行が発生すると、 AWSはイベントを に送信しますHealth Dashboard。特定のイベントは設定された復旧メカニズムと、試行が成功したか失敗したかによって異なります。

で自動インスタンス復旧イベントを確認するには Health Dashboard

1. https://phd.aws.amazon.com/phd/home#/ で 「Health Dashboard」 を開きます。
2. 自動インスタンス復旧に関連するイベントを探します。これらのイベントが存在すると、自動インスタンス復旧の試行が発生したかどうか、およびその結果を確認できます。
   - 簡易自動復旧
     - 成功イベント: `AWS_EC2_SIMPLIFIED_AUTO_RECOVERY_SUCCESS`
     - 失敗イベント: `AWS_EC2_SIMPLIFIED_AUTO_RECOVERY_FAILURE`
   - CloudWatch アクションに基づく復旧
     - 成功イベント: `AWS_EC2_INSTANCE_AUTO_RECOVERY_SUCCESS`
     - 失敗イベント: `AWS_EC2_INSTANCE_AUTO_RECOVERY_FAILURE`

CloudWatch でシステムステータスチェックをモニタリングする

CloudWatch で StatusCheckFailed_System メトリクスを確認することで、インスタンスの基盤となるハードウェアまたはソフトウェアの問題が検出されたかどうかを確認できます。メトリクス値はシステムステータスチェックに合格した (ハードウェアまたはソフトウェアの問題がない) か、失敗した (ハードウェアまたはソフトウェアの問題) かを示します。

基盤となるハードウェアまたはソフトウェアの問題が検出されたかどうかを確認するには

1. CloudWatch コンソールの https://console.aws.amazon.com/cloudwatch/home?#metricsV2 を開いてください。
2. EC2 インスタンスと同じリージョンで操作していることを確認します。
3. メトリクス検索フィールドに次のメトリクスを貼り付け、エンター キーを押します。

```
StatusCheckFailed_System
```

4. EC2 メトリクス. 、[インスタンス別メトリクス] を選択します。
5. テーブルで、チェックするインスタンスの横にあるチェックボックスをオンにします。
6. クエリ期間を、復旧イベントが発生したと思われる時間に変更します。
7. グラフ化されたメトリクスタブを選択し、StatusCheckFailed_System で次の操作を行います。
   a. 統計 で、平均、最大、最小のいずれかを選択してください。
   b. [期間] は[1 minute] を選択してください。
8. StatusCheckFailed_System の値を確認します。
   - 値 0: システムステータスチェックに合格し、基盤となるハードウェアまたはソフトウェアの問題がないことを示します。
   - 値 1: システムステータスチェックが失敗し、基盤となるハードウェアまたはソフトウェアの問題が示されます。

詳細については「インスタンスの自動復旧」を参照してください。

Amazon EC2 インスタンスで簡易自動復旧を設定する

> **Important**
>
> このセクションではEC2 インスタンスで復旧メカニズムをプロアクティブに設定する方法について説明します。これらの復旧メカニズムは がシステムステータスチェックが失敗する原因となる基盤となるハードウェアまたはソフトウェアの問題AWSを検出したときに、インスタンスの可用性を復元するように設計されています。インスタンスへのアクセスで現在問題が発生している場合は「EC2 インスタンスのトラブルシューティング」を参照してください。

が基盤となるハードウェアまたはソフトウェアの問題によりインスタンスが使用できないことをAWS検出した場合、簡易自動復旧は基盤となる問題のあるインスタンスをホストから別のホストに移動することで、インスタンスの可用性を自動的に復元できます。

簡易自動復旧が発生した場合、 AWSは結果に応じてAWS Health Dashboard、次のいずれかのイベントを に送信します。

- 成功イベント: `AWS_EC2_SIMPLIFIED_AUTO_RECOVERY_SUCCESS`
- 失敗イベント: `AWS_EC2_SIMPLIFIED_AUTO_RECOVERY_FAILURE`

これらのイベントの通知を受け取るには通知を設定できます。詳細についてはAWS User Notifications「ユーザーガイド」の での「最初の通知設定の作成」AWS User Notificationsを参照してください。Amazon EventBridge ルールを使用して、次のイベントコードを使って簡易自動復旧のイベントをモニタリングすることもできます。

サポート対象のインスタンスを起動すると、簡易自動復旧がデフォルトで有効になります。ただし、インスタンスが `running`状態であり、 にサービスイベントがリストされておらずAWS Health Dashboard、インスタンスタイプに使用可能な容量がある場合にのみ動作できます。大幅な停止などの状況では容量の制約により復旧の試行が失敗する可能性があります。詳細については「the section called “簡易自動復旧障害のトラブルシューティング”」を参照してください。

起動中または起動後に簡易自動復旧を無効にし、必要に応じて後で再度有効にすることができます。

> **Warning**
>
> 基盤となるハードウェアまたはソフトウェアの問題により がインスタンスをAWS復旧する場合、次の結果に注意してください。揮発性メモリ (RAM) に保存されているデータは失われ、オペレーティングシステムの稼働時間はゼロから開始されます。データ損失を防ぐために、重要なデータのバックアップを定期的に作成することをお勧めします。Amazon EC2 インスタンスのバックアップと復旧のベストプラクティスの詳細については「Amazon EC2 のベストプラクティス」を参照してください。
>
> 自動インスタンス復旧メカニズムは個々のインスタンス用に設計されています。回復力のあるシステムの構築に関するガイダンスについては回復力のあるシステムを構築するを参照してください。

内容

- 簡易自動復旧の要件と制限

- 簡易自動復旧を設定する
- 簡易自動復旧障害のトラブルシューティング

簡易自動復旧の要件と制限

簡易自動復旧は次の条件を満たすインスタンスで有効にできます。

インスタンスのタイプ

- 汎用:
  A1、M3、M4、M5、M5a、M5n、M5zn、M6a、M6g、M6i、M6in、M7a、M7g、M7i、M7i-flex、M8a、M8azn、M8g、M8gb、M8gn、M8i、M8i-flex、M8in、M8ine、M8ib、M9g、T1、T2、T3、T3a、T4g
- コンピューティング最適化:
  C3、C4、C5、C5a、C5n、C6a、C6g、C6gn、C6i、C6in、C7a、C7g、C7gn、C7i、C7i-flex、C8a、C8g、C8gb、C8gn、C8i、C8i-flex、C8in、C8ine、C8ib、C9g
- メモリ最適化:
  R3、R4、R5、R5a、R5b、R5n、R6a、R6g、R6i、R6in、R7a、R7g、R7i、R7iz、R8a、R8g、R8gb flex、R8in、R8ib、U-3tb1、U-6tb1、U-9tb1、U-12tb1、U-18tb1、U-24tb1、U7i-6tb、U7i-8tb、U7i-12
- 高速コンピューティング: G3、G5g、Inf1、P3、VT1
- ハイパフォーマンスコンピューティング: Hpc6a、Hpc7a、Hpc7g、Hpc8a

テナンシー

- Shared
- ハードウェア専有インスタンス

  詳細については「Amazon EC2 ハードウェア専有インスタンス」を参照してください。

制限事項

以下の特性を持つインスタンスでは、簡易自動復旧はサポートされていません。

- インスタンスサイズ: `metal`インスタンス
- テナンシー: 専用ホスト。専用ホスト の場合は代わりに Dedicated Host Auto Recovery を使用します。
- ストレージ: インスタンスストアボリュームを持つインスタンス
- ネットワーク: エラスティックファブリックアダプタ を使用するインスタンス

- Auto Scaling: Auto Scaling グループの一部であるインスタンス
- メンテナンス: スケジュールされたメンテナンスイベントを現在実行しているインスタンス

簡易自動復旧を設定する

サポート対象のインスタンスを起動すると、簡易自動復旧がデフォルトで有効になります。インスタンスの起動中または起動後の自動復旧動作を `disabled` に設定できます。

簡易自動復旧をサポートしていないインスタンスタイプではこの機能は`default` 設定で有効化されません。

Console

起動時に簡易自動復旧を無効にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Instances] (インスタンス)、[Launch instance] (インスタンスの起動) の順に選択してください。
3. [Advanced details] (高度な詳細) セクションの [Instance auto-recovery] (インスタンスの自動復旧) で、[Disabled] (無効) を選択してください。
4. 必要に応じて残りのインスタンスの起動設定を設定し、インスタンスを起動します。

起動後に簡易自動復旧を無効にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. [Actions (アクション)]、[Instance settings (インスタンスの設定)]、[Change shutdown behavior (自動復旧動作を変更)] の順に選択します 現在の動作が選択されます。
4. [Off] (オフ) を選択した上で、[Save] (保存) をクリックしてください。

起動後に簡易自動復旧を有効にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. [Actions (アクション)]、[Instance settings (インスタンスの設定)]、[Change shutdown behavior (自動復旧動作を変更)] の順に選択します 現在の動作が選択されます。

4. [デフォルト] を選択した上で、[保存] をクリックしてください。

AWS CLI

起動時に簡易自動復旧を無効にするには

`--maintenance-options` オプションで run-instances コマンドを使用します。

```
--maintenance-options AutoRecovery=Disabled
```

起動後に簡易自動復旧を無効にするには

modify-instance-maintenance-options コマンドを使用します。

```
aws ec2 modify-instance-maintenance-options \
    --instance-id i-1234567890abcdef0 \
    --auto-recovery disabled
```

起動後に簡易自動復旧を有効にするには

modify-instance-maintenance-options コマンドを使用します。

```
aws ec2 modify-instance-maintenance-options \
    --instance-id i-1234567890abcdef0 \
    --auto-recovery default
```

PowerShell

起動時に簡易自動復旧を無効にするには

New-EC2Instance コマンドレットを使用します。

```
-MaintenanceOptions_AutoRecovery Disabled
```

起動後に簡易自動復旧を無効にするには

Edit-EC2InstanceMaintenanceOption コマンドレットを使用します。

```
Edit-EC2InstanceMaintenanceOption `
    -InstanceId i-1234567890abcdef0 `
    -AutoRecovery Disabled
```

起動後に簡易自動復旧を有効にするには

Edit-EC2InstanceMaintenanceOption コマンドレットを使用します。

```
Edit-EC2InstanceMaintenanceOption `
    -InstanceId i-1234567890abcdef0 `
    -AutoRecovery Enabled
```

簡易自動復旧障害のトラブルシューティング

次の問題により、インスタンスの自動復旧が失敗する可能性があります。

- AWS サービスイベントが実行されている

  簡易自動復旧は AWS Health Dashboard のサービスイベント中には動作しません。これらのイベントが原因で復旧が失敗しても、その通知を受信しない可能性があります。最新のサービス可用性情報については「サービスヘルス」のステータスページを参照してください。
- 容量不足

  インスタンスを移行するための代替ハードウェアが一時的に不足しています。
- 1 日あたりの最大復旧試行回数に達しました

  インスタンスが、1 日に許可されている復旧試行の最大数に達しました。自動復旧が失敗し、元のシステムステータスチェックエラーの根本原因がハードウェアの機能低下であると判断された場合、対象のインスタンスが使用停止になることがあります。

複数の復旧試行にもかかわらずインスタンスのシステムステータスチェックエラーが続く場合は「ステータスチェックに失敗したインスタンスのトラブルシューティング」を参照して、追加のガイダンスを確認してください。

EC2 インスタンスでの CloudWatch アクションベースの復旧の設定

> **Important**
>
> このセクションではEC2 インスタンスで復旧メカニズムをプロアクティブに設定する方法について説明します。これらの復旧メカニズムは がシステムステータスチェックが失敗する原因となる基盤となるハードウェアまたはソフトウェアの問題AWSを検出したときに、インスタンスの可用性を復元するように設計されています。インスタンスへのアクセスで現在問

題が発生している場合は「EC2 インスタンスのトラブルシューティング」を参照してください。

は基盤となるハードウェアまたはソフトウェアの問題によりインスタンスが使用できないことをAWS検出した場合、CloudWatch アクションベースのリカバリは基盤となる問題のあるホストから別のホストにインスタンスを移動することで、インスタンスの可用性を自動的に復元できます。

CloudWatch アクションベースの復旧が発生した場合、 AWSは結果に応じてAWS Health Dashboard、次のいずれかのイベントを に送信します。

- 成功イベント: `AWS_EC2_INSTANCE_AUTO_RECOVERY_SUCCESS`
- 失敗イベント: `AWS_EC2_INSTANCE_AUTO_RECOVERY_FAILURE`

Amazon CloudWatch アクションベースの復旧を設定して、Amazon CloudWatch アラームに復旧アクションを追加できます。CloudWatch アクションベースの復旧は `StatusCheckFailed_System` メトリクスを使用して機能します。CloudWatch アクションベースの復旧機能は、復旧アクションと結果に関する、最新の復旧応答時間の詳細度と Amazon Simple Notification Service (Amazon SNS) 通知を提供します。これらの設定オプションにより、簡易自動復旧と比較して、システムステータスチェックの障害イベントの応答をより詳細に制御して、より迅速な復旧試行が可能になります。使用可能な CloudWatch オプションの詳細については「インスタンスのステータスチェック」を参照してください。

ただし、CloudWatch アクションベースの復旧はインスタンスが `running`状態にあり、 にサービスイベントがリストされておらずAWS Health Dashboard、インスタンスタイプに使用可能な容量がある場合にのみ動作できます。大幅な停止などの状況では容量の制約により復旧の試行が失敗する可能性があります。詳細については「the section called “トラブルシューティング”」を参照してください。

> **Warning**
>
> 基盤となるハードウェアまたはソフトウェアの問題により がインスタンスをAWS復旧する場合、次の結果に注意してください。揮発性メモリ (RAM) とインスタンスストアボリュームに保存されているデータは失われ、オペレーティングシステムの稼働時間はゼロから開始されます。データ損失を防ぐために、重要なデータのバックアップを定期的に作成することをお勧めします。Amazon EC2 インスタンスのバックアップと復旧のベストプラクティスの詳細については「Amazon EC2 のベストプラクティス」を参照してください。

自動インスタンス復旧メカニズムは個々のインスタンス用に設計されています。回復力のあるシステムの構築に関するガイダンスについては回復力のあるシステムを構築するを参照してください。

内容

- CloudWatch アクションベースの復旧の要件と制限
- CloudWatch アクションベースの復旧
- CloudWatch アクションベースの復旧障害のトラブルシューティング

CloudWatch アクションベースの復旧の要件と制限

CloudWatch アクションベースの復旧は次の条件を満たすインスタンスで有効にできます。

インスタンスのタイプ

- 汎用:
  A1、M3、M4、M5、M5a、M5n、M5zn、M6a、M6g、M6i、M6in、M7a、M7g、M7i、M7i-flex、M8a、M8azn、M8g、M8gb、M8gn、M8i、M8i-flex、M8in、M8ine、M8ib、M9g、T1、T2、T3、T3a、T4g
- コンピューティング最適化:
  C3、C4、C5、C5a、C5n、C6a、C6g、C6gn、C6i、C6in、C7a、C7g、C7gn、C7i、C7i-flex、C8a、C8g、C8gb、C8gn、C8i、C8i-flex、C8in、C8ine、C8ib、C9g
- メモリ最適化:
  R3、R4、R5、R5a、R5b、R5n、R6a、R6g、R6i、R6in、R7a、R7g、R7i、R7iz、R8a、R8g、R8gb flex、R8in、R8ib、U-3tb1、U-6tb1、U-9tb1、U-12tb1、U-18tb1、U-24tb1、U7i-6tb、U7i-8tb、U7i-12
- 高速コンピューティング: G3、G5g、Inf1、P3、VT1
- ハイパフォーマンスコンピューティング: Hpc6a、Hpc7a、Hpc7g、Hpc8a
- メタルインスタンス: メタルインスタンスサイズを持つ上記のインスタンス タイプのいずれか。
- 起動時にインスタンスストアボリュームが追加された場合： 次のインスタンス タイプのみがサポートされます: M3、C3、R3、X1、X1e、X2idn、X2iedn

テナンシー

- Shared

- ハードウェア専有インスタンス

詳細については「Amazon EC2 ハードウェア専有インスタンス」を参照してください。

制限事項

CloudWatch アクションベースの復旧は次の特性を持つインスタンスではサポートされていません。

- テナンシー: 専用ホスト。専用ホスト の場合は代わりに Dedicated Host Auto Recovery を使用します。
- ネットワーク: エラスティックファブリックアダプタ を使用するインスタンス
- Auto Scaling: Auto Scaling グループの一部であるインスタンス
- メンテナンス: スケジュールされたメンテナンスイベントを現在実行しているインスタンス

サポートされているインスタンスタイプの検索

CloudWatch アクションベースの復旧をサポートする、インスタンスタイプを表示することができます。

Console

Amazon CloudWatch アクションベースの復旧をサポートするインスタンスタイプを表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左のナビゲーションペインで、[Instance Types] (インスタンスタイプ) を選択してください。
3. フィルタバーで、[自動復旧のサポート] = [true] をフィルタリングします。[インスタンスタイプ] テーブルには、Amazon CloudWatch アクションベースの復旧をサポートするすべてのインスタンスタイプが表示されます。
4. (オプション) フィルターを追加して、対象の、特定のインスタンスタイプをさらに絞り込みます。

AWS CLI

Amazon CloudWatch アクションベースの復旧をサポートするインスタンスタイプを表示するには

describe-instance-types コマンドを auto-recovery-supported フィルターと共に使用します。

```
aws ec2 describe-instance-types \
    --filters Name=auto-recovery-supported,Values=true \
    --query "InstanceTypes[*].[InstanceType]" \
    --output text | sort
```

PowerShell

Amazon CloudWatch アクションベースの復旧をサポートするインスタンスタイプを表示するには

Get-EC2InstanceType コマンドレットを auto-recovery-supported フィルタと共に使用します。

```
Get-EC2InstanceType `
    -Filter @{Name="auto-recovery-supported";Values="true"} | `
    Select InstanceType | Sort-Object InstanceType
```

CloudWatch アクションベースの復旧

EC2 インスタンスの CloudWatch アクションベースの復旧を設定するには指定されたインスタンスの StatusCheckFailed_Systemメトリクスをモニタリングする CloudWatch アラームを作成します。メトリクス値が 1 の場合にトリガーされるようにアラームを設定します。これはシステムステータスチェックが失敗したことを示します。トリガーされたときにインスタンスを自動的に復旧するようにアラームアクションを設定します。

Amazon EC2 コンソールまたは CloudWatch コンソールを使用してアラームを作成できます。手順についてはこのユーザーガイドの「Amazon CloudWatch アラームへの復旧アクションの追加」、または「Amazon CloudWatch ユーザーガイド」の「Amazon CloudWatch アラームへの復旧アクションの追加」を参照してください。

CloudWatch アクションベースの復旧障害のトラブルシューティング

次の問題により、CloudWatch アクションベースの復旧によるインスタンスの復旧が失敗する場合があります。

- AWS サービスイベントが実行されている

CloudWatch アクションベースの復旧はAWS Health Dashboard のサービスイベント中は動作しません。これらのイベントが原因で復旧が失敗しても、その通知を受信しない可能性があります。最新のサービス可用性情報については「サービスヘルス」のステータスページを参照してください。

- 容量不足

  インスタンスを移行するための代替ハードウェアが一時的に不足しています。

- 1 日あたりの最大復旧試行回数に達しました

  インスタンスが、1 日に許可されている復旧試行の最大数に達しました。自動復旧が失敗し、元のシステムステータスチェックエラーの根本原因がハードウェアの機能低下であると判断された場合、対象のインスタンスが使用停止になることがあります。

複数の復旧試行にもかかわらずインスタンスのシステムステータスチェックエラーが続く場合は「ステータスチェックに失敗したインスタンスのトラブルシューティング」を参照して、追加のガイダンスを確認してください。

インスタンスメタデータを使用して EC2 インスタンスを管理する

インスタンスメタデータはインスタンスに関するデータで、実行中のインスタンスを設定または管理するために使用します。インスタンスメタデータには以下が含まれます。

インスタンスメタデータのプロパティ

インスタンスメタデータのプロパティはホスト名、イベント、およびセキュリティグループなどでカテゴリ分けされます。

動的データ

動的データはインスタンスアイデンティティドキュメントなど、インスタンスの起動時に生成されるメタデータです。詳細については「動的データのカテゴリ」を参照してください。

ユーザーデータ

インスタンスメタデータを使用して、インスタンスの起動時に指定したユーザーデータにアクセスすることもできます。例えば、インスタンスを設定するためにパラメータを指定したり、単純なスクリプトを含めたりすることができます。汎用 AMI を構築し、ユーザーデータを使用して起動時に提供された設定ファイルを変更することができます。例えば、さまざまな小規模ビジネスを対象としたウェブサーバーを実行する場合に、すべてのサーバーで同じ汎用 AMI を使用し、起動時にユーザーデータで指定した Amazon S3 バケットからコンテンツを取得できます。

随時新規顧客を追加するには顧客のバケットを作成し、そのコンテンツを追加し、ユーザーデータのコードに提供された固有のバケット名を使って AMI を起動します。同じ RunInstances 呼び出しを使用して複数のインスタンスを起動する場合、ユーザーデータはその予約においてすべてのインスタンスで使用可能です。同じリザベーションの一部である各インスタンスには固有の ami-launch-index 番号があるため、インスタンスが実行する操作を制御するコードを書くことができます。例えば、最初のホストはクラスター内の最初のノードとしてそれ自体を選択する場合があります。詳細な AMI 起動の例については「1 つのリクエストで起動された各インスタンスを特定する」を参照してください。

Important

インスタンスメタデータおよびユーザーデータにはそのインスタンス自体内からのみアクセスできるものの、データは認証または暗号化手法によって保護されていません。インスタンス、そしてインスタンス上で実行される任意のソフトウェアに対して直接アクセス権がある可能性がある人はメタデータを表示できます。そのため、パスワードまたは存続期間の長い暗号化キーなどの機密データはユーザーデータとして保管しないようにしてください。

目次

- インスタンスメタデータのカテゴリ
- 動的データのカテゴリ
- EC2 インスタンスのインスタンスメタデータにアクセスする
- インスタンスメタデータサービスのオプションを設定する
- ユーザーデータ入力を使用して EC2 インスタンスを起動するときにコマンドを実行する
- 1 つのリクエストで起動された各インスタンスを特定する

インスタンスメタデータのカテゴリ

インスタンスメタデータプロパティはいくつかのカテゴリに分けられます。インスタンスメタデータのプロパティを取得するにはリクエストでカテゴリを指定します。すると、メタデータがレスポンスで返されます。

新しいカテゴリがリリースされると、新しいインスタンスメタデータビルドが新しいバージョン番号で作成されます。次の表では[Version when category was released] (カテゴリがリリースされたときのバージョン) 列が、インスタンスメタデータカテゴリがリリースされたときのビルドバージョンを

指定しています。Amazon EC2 が新しいインスタンスメタデータビルドをリリースするたびにコードを更新する必要をなくすために、メタデータリクエストのバージョン番号の代わりに、`latest` を使用することが推奨されます。詳細については使用できるインスタンスメタデータのバージョンを取得する を参照してください。

Amazon EC2 が新しいインスタンスメタデータカテゴリをリリースすると、新しいカテゴリのインスタンスメタデータが既存のインスタンスで使用できなくなる場合があります。Nitro ベースのインスタンスでは起動時に用意されたカテゴリのインスタンスメタデータのみが取得可能です。Xen ハイパーバイザーを使用するインスタンスの場合は一度停止した後に開始することで、そのインスタンスで使用可能なカテゴリを更新できます。

次の表はインスタンスメタデータのカテゴリをまとめたものです。カテゴリ名のいくつかにはインスタンスに固有のデータのためのプレースホルダーが含まれています。例えば、*mac* はネットワークインターフェイスの MAC アドレスを表します。インスタンスメタデータを取得する際にはプレースホルダーを実際の値に置き換える必要があります。

Category	説明	カテゴリがリリースされたときのバージョン
`ami-id`	インスタンスの起動に使用される AMI ID。	1.0
`ami-launch-index`	同じ `RunInstances` 呼び出しを使用して複数のインスタンスを起動する場合、この値は各インスタンスの起動順序を示します。最初に起動されたインスタンスの値は 0 です。Auto Scaling または EC2 フリートを使用してインスタンスを起動する場合、この値は常に 0 です。	1.0
`ami-manifest-path`	Amazon S3 での AMI のマニフェストファイルのパス。Amazon EBS-backed AMI を使用してインスタンスを起動した場合、返される結果は unknown です。	1.0

Category	説明	カテゴリがリリースされたときのバージョン
`ancestor-ami-ids`	この AMI を作成するために再バンドルされたあらゆるインスタンスの AMI ID。この値はAMI マニフェストファイルが `ancestor-amis` キーを含む場合にのみ存在します。	2007-10-10
`autoscaling/target-lifecycle-state`	Auto Scaling インスタンスの移行先となる、ターゲットの Auto Scaling ライフサイクルの状態を示す値。2022 年 3 月 10 日以降、インスタンスがターゲットのライフサイクル状態の 1 つに移行したときに表示されます。使用できる値: `Detached|InService |Standby|Terminated |Warmed:Hibernated |Warmed:Running |Warmed:Stopped |Warmed:Terminated` 「Amazon EC2 Auto Scaling ユーザーガイド」の「インスタンスメタデータを使用してターゲットライフサイクル状態を取得する」を参照してください。	2021-07-15
`block-device-mapping/ami`	root/boot ファイルシステムを含む仮想デバイス。	2007-12-15

Category	説明	カテゴリがリリースされたときのバージョン
block-device-mapping/ebsN	任意の Amazon EBS ボリュームに関連付けられた仮想デバイス。Amazon EBS ボリュームは起動の時点、またはインスタンスが最後に開始された時点で存在している場合にのみ、メタデータで使用できます。N はAmazon EBS ボリュームのインデックス (ebs1 や ebs2 など) を示します。	2007-12-15
block-device-mapping/ephemeral N	非 NVMe インスタンスストアボリュームの仮想デバイス。N は各ボリュームのインデックスを示します。ブロックデバイスマッピングのインスタンスストアボリュームの数はインスタンスのインスタンスストアボリュームの実際の数に一致しない場合があります。インスタンスに使用可能なインスタンスストアボリュームの数はインスタンスタイプによって決定されます。ブロックデバイスマッピングのインスタンスストアボリュームの数が、インスタンスに利用可能な数を超える場合、追加のインスタンスストアボリュームは無視されます。	2007-12-15

Category	説明	カテゴリがリリースされたときのバージョン
block-device-mapping/root	ルートデバイスに関連付けられた仮想デバイスまたはパーティション、あるいは仮想デバイス上のパーティション。ルート (/ または C:) ファイルシステムは所定のインスタンスに関連付けられています。	2007-12-15
block-device-mapping/swap	swap に関連付けられた仮想デバイス。存在しない場合もあります。	2007-12-15
events/maintenance/history	インスタンスの完了またはキャンセルされたメンテナンスイベントがある場合はイベントに関する情報を含む JSON 文字列を含みます。	2018-08-17
events/maintenance/scheduled	インスタンスがアクティブなメンテナンスイベントがある場合はイベントに関する情報を含む JSON 文字列を含みます。詳細についてはAmazon EC2 インスタンスに影響する予定されているイベントの表示を参照してください。	2018-08-17

Category	説明	カテゴリがリリースされたときのバージョン
`events/recommendations/rebalance`	EC2 インスタンスの再調整推奨通知がインスタンスに対して送信されるおおよその時間 (UTC)。このカテゴリのメタデータの例を次に示します。`{"noticeTime": "2020-11-05T08:22:00Z"}` このカテゴリは通知が発された後にのみ使用できます。詳細については「EC2 インスタンスの再調整に関する推奨事項」を参照してください。	2020-10-27
`hostname`	EC2 インスタンスが IP ベースの命名 (IPBN) を使用している場合、これはインスタンスのプライベート IPv4 DNS ホスト名です。EC2 インスタンスがリソースベースの命名 (RBN) を使用している場合、これは RBN です。複数のネットワークインターフェイスが存在する場合、これは eth0 デバイス (デバイス番号が 0 のデバイス) を示します。IPBN と RBN の詳細についてはEC2 インスタンスのホスト名とドメインを参照してください。	1.0

Category	説明	カテゴリがリリースされたときのバージョン
`iam/info`	インスタンスに関連付けられた IAM ロールがある場合、インスタンスの LastUpdated の日付、InstanceProfileArn、InstanceProfileId など、インスタンスプロファイルが更新された最終時刻に関する情報が格納されます。そうでない場合はなしになります。	2012-01-12
`iam/security-credentials/role-name`	インスタンスに関連付けられた IAM ロールがある場合、*`role-name`* はロールの名前になり、*`role-name`* に、そのロールに関連付けられた一時的なセキュリティ認証情報が格納されます (詳細については[インスタンスメタデータからのセキュリティ認証情報の取得]を参照してください)。そうでない場合はなしになります。	2012-01-12
`identity-credentials/ec2/info`	`identity-credentials/ec2/security-credentials/ec2-instance` の認証情報に関する情報。	2018-05-23

Category	説明	カテゴリがリリースされたときのバージョン
identity-credentials/ec2/security-credentials/ec2-instance	インスタンス上のソフトウェアが自身を AWS に識別し、EC2 Instance Connect や AWS Systems Manager デフォルトのホスト管理設定などの機能をサポートできるようにするインスタンスアイデンティティロール用の認証情報。これらの認証情報にはポリシーがアタッチされていないため、AWS 機能に対してインスタンスを識別する以外に追加の AWS API 許可はありません。詳細については「Amazon EC2 インスタンスのインスタンスアイデンティティロール」を参照してください。	2018-05-23
instance-action	バンドルの準備のために再起動する必要があることをインスタンスに伝えます。有効な値: none | shutdown | bundle-pending 。	2008-09-01
instance-id	このインスタンスの ID。	1.0
instance-life-cycle	このインスタンスの購入オプション。詳細については Amazon EC2 の請求および購入オプションを参照してください。	2019-10-01
instance-type	インスタンスの種類。詳細については Amazon EC2 インスタンスタイプを参照してください。	2007-08-29

Category	説明	カテゴリがリリースされたときのバージョン
ipv6	インスタンスの IPv6 アドレス。複数のネットワークインターフェイスが存在する場合、これは eth0 デバイス (デバイス番号が 0 のデバイス) のネットワークインターフェイスおよび最初に割り当てられた IPv6 アドレスを示します。ネットワークインターフェイス [0] に IPv6 アドレスが存在しない場合、この項目は設定されず、HTTP 404 応答が返されます。	2021-01-03
kernel-id	このインスタンスで起動したカーネルの ID (ある場合)。	2008-02-01
local-hostname	複数のネットワークインターフェイスが存在する場合、これは eth0 デバイス (デバイス番号が 0 のデバイス) を示します。EC2 インスタンスが IP ベースの命名 (IPBN) を使用している場合、これはインスタンスのプライベート IPv4 DNS ホスト名です。EC2 インスタンスがリソースベースの命名 (RBN) を使用している場合、これは RBN です。IPBN、RBN、および EC2 インスタンスの命名の詳細についてはEC2 インスタンスのホスト名とドメインを参照してください。	2007-01-19

Category	説明	カテゴリがリリースされたときのバージョン
local-ipv4	インスタンスのプライベート IPv4 アドレス。複数のネットワークインターフェイスが存在する場合、これは eth0 デバイス (デバイス番号が 0 のデバイス) を示します。これが IPv6 専用インスタンスの場合、この項目は設定されず、HTTP 404 応答が返されます。	1.0
mac	インスタンスのメディアアクセスコントロール (MAC) アドレス。複数のネットワークインターフェイスが存在する場合、これは eth0 デバイス (デバイス番号が 0 のデバイス) を示します。	2011-01-01
metrics/vhostmd	使用不可	2011-05-01
network/interfaces/macs/mac/device-number	そのインターフェイスに関連付けられた固有のデバイス番号。デバイス番号はデバイス名に対応します。例えば、2 という device-number は eth2 デバイスを指します。このカテゴリはAmazon EC2 API で使用される DeviceIndex フィールドと device-index フィールド、および AWS CLI の EC2 コマンドに対応します。	2011-01-01
network/interfaces/macs/mac/interface-id	ネットワークインターフェイスの ID。	2011-01-01

Category	説明	カテゴリがリリースされたときのバージョン
network/interfaces/macs/mac/ipv4-associations/public-ip	各パブリック IP アドレスに関連付けられ、そのインターフェイスに割り当てられたプライベート IPv4 アドレス。	2011-01-01
network/interfaces/macs/mac/ipv6s	インターフェイスに割り当てられた IPv6 アドレス。	2016-06-30
network/interfaces/macs/mac/ipv6-prefix	ネットワークインターフェイスに割り当てられた IPv6 プレフィクス。	
network/interfaces/macs/mac/local-hostname	インスタンスのプライベート IPv4 DNS ホスト名。複数のネットワークインターフェイスが存在する場合、これは eth0 デバイス (デバイス番号が 0 のデバイス) を示します。これが IPv6 専用インスタンスの場合、これはリソースベースの名前です。IPBN と RBN の詳細についてはEC2 インスタンスのホスト名とドメインを参照してください。	2007-01-19
network/interfaces/macs/mac/local-ipv4s	インターフェイスに関連付けられたプライベート IPv4 アドレス。これが IPv6 専用のネットワークインターフェイスである場合、この項目は設定されず、HTTP 404 応答が返されます。	2011-01-01
network/interfaces/macs/mac/mac	インスタンスの MAC アドレス。	2011-01-01

Category	説明	カテゴリがリリースされたときのバージョン
network/interfaces/macs/*mac*/network-card	ネットワークカードのインデックス。インスタンスタイプによっては複数のネットワークカードがサポートされているものもあります。	2020 年 11 月 1 日
network/interfaces/macs/mac/owner-id	ネットワークインターフェイスの所有者の ID。複数インターフェイスの環境ではインターフェイスは Elastic Load Balancing などのサードパーティによってアタッチできます。インターフェイス上のトラフィックは常にインターフェイス所有者に対して課金されます。	2011-01-01
network/interfaces/macs/mac/public-hostname	インターフェイスのパブリック DNS (IPv4)。このカテゴリは enableDnsHostnames 属性が true に設定されている場合にのみ返されます。詳細については Amazon VPC ユーザーガイドの「DNS attributes for your VPC」(VPC の DNS 属性) を参照してください。インスタンスにパブリック IPv6 アドレスのみがあり、パブリック IPv4 アドレスがない場合、この項目は設定されず、HTTP 404 応答が返されます。	2011-01-01
network/interfaces/macs/mac/public-ipv4s	インターフェイスに関連付けられたパブリック IP アドレスまたは Elastic IP アドレス。インスタンスには複数の IPv4 アドレスが存在する場合があります。	2011-01-01

Category	説明	カテゴリがリリースされたときのバージョン
network/interfaces/macs/mac/security-groups	ネットワークインターフェイスが属するセキュリティグループ。	2011-01-01
network/interfaces/macs/mac/security-group-ids	ネットワークインターフェイスが属するセキュリティグループの ID。	2011-01-01
network/interfaces/macs/mac/subnet-id	インターフェイスが存在するサブネットの ID。	2011-01-01
network/interfaces/macs/mac/subnet-ipv4-cidr-block	インターフェイスが存在するサブネットの IPv4 CIDR ブロック。	2011-01-01
network/interfaces/macs/mac/subnet-ipv6-cidr-blocks	インターフェイスが存在するサブネットの IPv6 CIDR ブロック。	2016-06-30
network/interfaces/macs/mac/vpc-id	インターフェイスが存在する VPC の ID。	2011-01-01
network/interfaces/macs/mac/vpc-ipv4-cidr-block	VPC のプライマリ IPv4 CIDR ブロック。	2011-01-01
network/interfaces/macs/mac/vpc-ipv4-cidr-blocks	VPC の IPv4 CIDR ブロック。	2016-06-30
network/interfaces/macs/mac/vpc-ipv6-cidr-blocks	インターフェイスが存在する VPC の IPv6 CIDR ブロック。	2016-06-30
placement/availability-zone	インスタンスが起動した利用可能ゾーン。	2008-02-01

Category	説明	カテゴリがリリースされたときのバージョン
placement/availability-zone-id	インスタンスが起動される静的アベイラビリティーゾーンの ID。このアベイラビリティーゾーン ID はアカウント間で一貫しています。ただし、アベイラビリティーゾーンとは異なる場合があります (アベイラビリティーゾーンはアカウントによって異なります)。	2019-10-01
placement/group-name	インスタンスが起動されるプレイスメントグループの名前。	2020-08-24
placement/host-id	インスタンスが起動されるホストの ID。専用ホスト にのみ適用されます。	2020-08-24
placement/partition-number	インスタンスが起動されるパーティションの番号。	2020-08-24
placement/region	インスタンスが起動される AWS リージョン。	2020-08-24
product-codes	インスタンスに関連付けられた AWS Marketplace 製品コード (ある場合)。	2007-03-01

Category	説明	カテゴリがリリースされたときのバージョン
public-hostname	インスタンスのパブリック DNS (IPv4)。このカテゴリは enableDnsHostnames 属性が true に設定されている場合にのみ返されます。詳細については Amazon VPC ユーザーガイドの「DNS attributes for your VPC」(VPC の DNS 属性) を参照してください。インスタンスにパブリック IPv6 アドレスのみがあり、パブリック IPv4 アドレスがない場合、この項目は設定されず、HTTP 404 応答が返されます。	2007-01-19
public-ipv4	パブリック IPv4 アドレス。インスタンスに Elastic IP アドレスが関連付けられている場合、返される値は Elastic IP アドレスです。	2007-01-19
public-keys/0/openssh-key	パブリックキー。インスタンスの起動時に指定された場合のみ返されます。	1.0
ramdisk-id	起動時に指定された RAM ディスクの ID (該当する場合)。	2007-10-10
reservation-id	予約の ID。	1.0

Category	説明	カテゴリがリリースされたときのバージョン
`security-groups`	インスタンスに適用されるセキュリティグループの名前。 起動後、インスタンスのセキュリティグループを変更できます。これらの変更はこの場所と network/interfaces/macs/*mac*/security-groups に反映されます。	1.0
`services/domain`	リージョンのAWSリソースのドメイン。	2014-02-25
`services/partition`	リソースが置かれているパーティションです。標準の AWS リージョンの場合、パーティションは `aws` です。他のパーティションにリソースがある場合、パーティションは `aws-`*`partitionname`* です。例えば、中国 (北京) リージョンにあるリソースのパーティションは`aws-cn` です。	2015-10-20
`spot/instance-action`	アクション (休止、停止、または終了) およびアクションのおよその発生時刻 (UTC)。この項目が存在するのはスポットインスタンスが休止、停止、または終了のためにマークされた場合のみです。詳細については instance-action を参照してください。	2016-11-15

Category	説明	カテゴリがリリースされたときのバージョン
spot/termination-time	スポットインスタンスで使用しているオペレーティングシステムが、シャットダウン信号を受信するおよその時刻 (UTC)。この項目はスポットインスタンスに対し Amazon EC2 による終了のマークが付けられている場合にのみ存在し、時刻値 (例えば 2015-01-05T18:02:00Z) が含まれます。ユーザー自身がスポットインスタンスを終了した場合、termination-time 項目に時刻は記述されません。詳細については termination-time を参照してください。	2014-11-05
system	インスタンスの基盤となる仮想化タイプ (ハイパーバイザー)	2022-09-24
tags/instance	インスタンスに関連付けられるインスタンスタグ。インスタンスメタデータのタグへのアクセスを明示的に許可した場合のみ使用できます。詳細については インスタンスメタデータ内のタグへのアクセスを有効にする を参照してください。	2021-03-23

動的データのカテゴリ

次の表は動的データのカテゴリをまとめたものです。

Category	説明	カテゴリがリリースされたときのバージョン
`fws/instance-monitoring`	顧客が CloudWatch で詳細な 1 分間隔のモニタリングを有効にしているかどうかを示す値。有効な値: `enabled | disabled`	2009-04-04
`instance-identity/document`	インスタンス ID、プライベート IP アドレスなど、インスタンスの属性を含む JSON。Amazon EC2 インスタンスのインスタンスアイデンティティドキュメントを参照してください。	2009-04-04
`instance-identity/pkcs7`	署名に対してドキュメントの真正性およびコンテンツを確認するために使用されます。Amazon EC2 インスタンスのインスタンスアイデンティティドキュメントを参照してください。	2009-04-04
`instance-identity/signature`	オリジンおよび権限を確認するために使用できるデータ。Amazon EC2 インスタンスのインスタンスアイデンティティドキュメントを参照してください。	2009-04-04

EC2 インスタンスのインスタンスメタデータにアクセスする

EC2 インスタンスメタデータにはインスタンス自体の内部から、または EC2 コンソール、API、SDK、または AWS CLI からアクセスできます。コンソールまたはコマンドラインからインスタンスの現在のインスタンスメタデータ設定を取得するには「既存インスタンスのインスタンスメタデータオプションのクエリ」を参照してください。

また、EBS ルートボリュームを持つインスタンスのユーザーデータを変更できます。インスタンスは停止状態である必要があります。コンソールの使用説明については「インスタンスのユーザーデータを更新する」を参照してください。AWS CLI を使用する Linux の例については「modify-instance-attribute」を参照してください。Tools for Windows PowerShell を使用する Windows の例については「the section called “ユーザーデータと Windows PowerShell用ツール”」を参照してください。

> **Note**
>
> インスタンスメタデータおよびユーザーデータの取得に使用する HTTP リクエストに対しては課金されません。

インスタンスメタデータアクセス考慮事項

インスタンスメタデータの問題を回避するには、次の点を考慮してください。

IMDSv2 の強制適用によるインスタンス起動の失敗 (**`HttpTokensEnforced=enabled`**)

IMDSv2 の強制適用を有効にする前に、インスタンス上のすべてのソフトウェアが IMDSv2 をサポートするようにする必要があります。その後、デフォルトを変更して IMDSv1 を無効 (`httpTokens=required`) にし、強制適用を有効にすることができます。詳細については、インスタンスメタデータサービスバージョン 2 の使用への移行 を参照してください。

コマンド形式

コマンド形式はインスタンスメタデータサービスバージョン 1 (IMDSv1) とインスタンスメタデータサービスバージョン 2 (IMDSv2) のどちらを使用するかによって異なります。デフォルトでは両方のバージョンのインスタンスメタデータサービスを使用できます。IMDSv2の使用を義務付けるにはインスタンスメタデータサービスを使用してインスタンスメタデータにアクセスするを参照してください。

IMDSv2 が必要な場合はIMDSv1 は動作しません。

IMDSv1 を使用していて、応答がない場合はIMDSv2 が必要になる可能性があります。IMDSv2 が必要かどうかを確認するにはインスタンスを選択して詳細を表示します。[IMDSv2] の値は[必須] (IMDSv2 を使用する必要がある) または [オプション] (IMDSv2 または IMDSv1 を使用可能) のいずれかです。

(IMDSv2) /latest/api/token を使用してトークンを取得する

バージョン固有の任意のパス (例: `/2021-03-23/api/token`) に PUT リクエストを発行した場合は、メタデータサービスから 403 Forbidden エラーが返されます。この応答は意図されたものです。

メタデータのバージョン

Amazon EC2 が新しいインスタンスメタデータビルドをリリースするたびにコードを更新する必要をなくすために、バージョン番号ではなく、パス内の `latest` を使用することが推奨されます。

IPv6 サポート

IPv6 アドレスを使用してインスタンス メタデータを取得するには、IMDS [fd00:ec2::254] の IPv4 アドレスではなく169.254.169.254IPv6 アドレスを有効にして使用するようにしてください。インスタンスはIPv6 対応サブネットで起動された Nitro ベースのインスタンスである必要があります。

(Windows) Windows Sysprep を使用してカスタム AMI を作成する

カスタム Windows AMI からインスタンスを起動したときに IMDS が動作するようにするにはAMI は Sysprep を使用して作成された標準化されたイメージである必要があります。そうでない場合、IMDS は機能しません。詳細については「Windows Sysprep を使用して Amazon EC2 AMI を作成する」を参照してください。

コンテナ環境では、再設定またはホップ制限を 2 に引き上げることを検討してください。

AWS SDK はデフォルトで IMDSv2 コールを使用します。IMDSv2 呼び出しに応答がない場合、一部の AWS SDK は呼び出しを再試行し、それでも失敗する場合はIMDSv1 を使用します。これにより、特にコンテナ環境では遅延が発生することがあります。IMDSv2 を 必要とする AWS SDK の場合、コンテナ環境でホップ制限が 1 の場合、コンテナへの移動は追加のネットワークホップとみなされ、呼び出しは応答をまったく受け取らない可能性があります。

コンテナ環境でこれらの問題を軽減するには、設定 (AWS リージョン など) をコンテナに直接渡すように設定を変更するか、ホップ制限を 2 に引き上げることを検討してください。ホップ制限の影響については、「EC2 Instance Metadata Service の拡張により、オープンファイアウォール、リバースプロキシ、および SSRF の脆弱性に対して多層防御を追加」を参照してください。ホップ制限の変更については、「PUT レスポンスホップリミットを変更する」を参照してください。

パケット/秒 (PPS) 制限

リンクローカルアドレスを使用するサービスには 1,024 パケット/秒 (PPS) の制限があります。この制限にはRoute 53 Resolver DNS クエリ、インスタンスメタデータサービス (IMDS) リクエスト、Amazon Time Service Network Time Protocol (NTP) リクエスト、および Windows Licensing Service (Microsoft Windows ベースのインスタンス向け) リクエストの総計が含まれます。

ユーザーデータアクセスに関するその他の考慮事項

- ユーザーデータは非透過的なデータとして取り扱われ、取得時には指定したものが返されます。ユーザーデータの解釈およびそれに基づくアクションはインスタンス次第です。

- ユーザーデータはbase64 でエンコードされている必要があります。使用しているツールまたは SDK によっては、base64 エンコードが実行される場合があります。次に例を示します。
  - Amazon EC2コンソールはbase64 エンコードを実行したり、base64 エンコード入力を受け入れたりできます。
  - AWS CLI バージョン 2 は、ユーザーに対しデフォルトでバイナリパラメータの base64 エンコードを実行します。AWS CLI バージョン 1 は、ユーザーに対し `--user-data` パラメータの base64 エンコードを実行します。
  - AWS SDK for Python (Boto3) は、ユーザーに対し `UserData` パラメータの base64 エンコードを実行します。
- ユーザーデータは raw 形式の 16 KB に制限されます (以前は base64 エンコード)。base64 エンコード後の 文字列の長さサイズ n はceil(n/3)*4 です。
- ユーザーデータを取得するときにユーザーデータを base64 デコードする必要があります。インスタンスのメタデータあるいはコンソールを使用してデータを取得する場合、自動的にデコードされます。
- インスタンスを停止してユーザーデータを変更した後に、インスタンスを起動した場合でも、更新されたユーザーデータは自動的には実行されません。Windows インスタンスではインスタンスを起動したとき、またはインスタンスを再起動もしくは起動するたびに、更新されたユーザーデータスクリプトが 1 回実行されるように設定を構成することができます。
- ユーザーデータはインスタンス属性です。インスタンスから AMI を作成する場合、インスタンスのユーザーデータは AMI に含まれません。

EC2 インスタンス内からインスタンスメタデータにアクセスする

インスタンスメタデータは実行中のインスタンスから取得できるため、Amazon EC2 コンソールまたは AWS CLI を使用する必要はありません。これはインスタンスから実行するスクリプトを記述しているときに便利です。例えば、インスタンスメタデータからインスタンスのローカル IP アドレスにアクセスして、外部アプリケーションへの接続を管理できます。

以下はすべてインスタンスメタデータと見なされますが、さまざまな方法でアクセスされます。詳細を表示するためにアクセスするインスタンスメタデータのタイプを表すタブを選択してください。

Metadata

インスタンスメタデータプロパティはいくつかのカテゴリに分けられます。各インスタンスメタデータカテゴリの説明についてはインスタンスメタデータのカテゴリを参照してください。

実行中のインスタンス内からインスタンスメタデータプロパティにアクセスするには次の IPv4 または IPv6 URI からデータを取得します。これらの IP アドレスはリンクローカルアドレスであり、このインスタンスからのみ有効です。詳細については「リンクローカルアドレス」を参照してください。

IPv4

```
http://169.254.169.254/latest/meta-data/
```

IPv6

```
http://[fd00:ec2::254]/latest/meta-data/
```

Dynamic data

実行中のインスタンス内から動的データを取得するには次の URI のいずれかを使用します。

IPv4

```
http://169.254.169.254/latest/dynamic/
```

IPv6

```
http://[fd00:ec2::254]/latest/dynamic/
```

例: cURL を使用したアクセス

次の例ではcURL を使用して高レベルのインスタンスアイデンティティカテゴリを取得します。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-
aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/dynamic/
instance-identity/
rsa2048
pkcs7
document
signature
```

```
dsa2048
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/dynamic/instance-identity/
rsa2048
pkcs7
document
signature
dsa2048
```

例: PowerShell を使用したアクセス

次の例ではPowerShell を使用して高レベルのインスタンスアイデンティティカテゴリを取得します。

IMDSv2

```
PS C:\> [string]$token = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token-ttl-seconds" = "21600"} -Method PUT -Uri http://169.254.169.254/latest/api/token
```

```
PS C:\> Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} -Method
 GET -Uri http://169.254.169.254/latest/dynamic/instance-identity/
document
rsa2048
pkcs7
signature
```

IMDSv1

```
PS C:\> Invoke-RestMethod -uri http://169.254.169.254/latest/dynamic/instance-identity/
document
rsa2048
pkcs7
signature
```

動的データの詳細およびその取得方法の例については[Amazon EC2 インスタンスのインスタンスアイデンティティドキュメント]を参照してください。

User data

インスタンスからユーザーデータを取得するには次の URI のいずれかを使用します。IPv6 アドレスを使用してユーザーデータを取得するにはIPv6 アドレスが有効で、インスタンスが IPv6 対応サブネット内の Nitro ベースのインスタンスである必要があります。

IPv4

```
http://169.254.169.254/latest/user-data
```

IPv6

```
http://[fd00:ec2::254]/latest/user-data
```

ユーザーデータのリクエストはデータをそのままの状態で返します (コンテンツタイプ `application/octet-stream`)。インスタンスにユーザーデータがない場合、リクエストは `404 - Not Found` を返します。

例: cURL を使用してカンマ区切りテキストを取得するアクセス

次の例ではcURL を使用して、カンマ区切りテキストとして指定されたユーザーデータを取得します。

IMDSv2

```
TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/user-data
1234,john,reboot,true | 4512,richard, | 173,,,
```

IMDSv1

```
curl http://169.254.169.254/latest/user-data
1234,john,reboot,true | 4512,richard, | 173,,,
```

例: PowerShell を使用してカンマ区切りテキストを取得する

次の例ではPowerShell を使用して、カンマ区切りテキストとして指定されたユーザーデータを取得します。

IMDSv2

```
[string]$token = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token-ttl-seconds"
 = "21600"} -Method PUT -Uri http://169.254.169.254/latest/api/token
```

```
Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} -Method GET -Uri
 http://169.254.169.254/latest/user-data
1234,john,reboot,true | 4512,richard, | 173,,,
```

IMDSv1

```
Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = Invoke-RestMethod -Headers
 @{"X-aws-ec2-metadata-token-ttl-seconds" = "21600"} `
-Method PUT -Uri http://169.254.169.254/latest/api/token} -Method GET -uri
 http://169.254.169.254/latest/user-data
1234,john,reboot,true | 4512,richard, | 173,,,
```

例: スクリプトを取得するための cURL を使用したアクセス

次の例ではcURL を使用して、スクリプトとして指定されたユーザーデータを取得します。

IMDSv2

```
TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-
token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/user-
data
#!/bin/bash
yum update -y
service httpd start
chkconfig httpd on
```

IMDSv1

```
curl http://169.254.169.254/latest/user-data
#!/bin/bash
yum update -y
service httpd start
chkconfig httpd on
```

例: スクリプトを取得するための PowerShell を使用したアクセス

次の例ではPowerShell を使用して、スクリプトとして指定されたユーザーデータを取得します。

IMDSv2

```
[string]$token = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token-ttl-seconds"
 = "21600"} -Method PUT -Uri http://169.254.169.254/latest/api/token
```

```
Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} -Method GET -Uri
 http://169.254.169.254/latest/user-data
<powershell>
$file = $env:SystemRoot + "\Temp\" + (Get-Date).ToString("MM-dd-yy-hh-mm")
New-Item $file -ItemType file
</powershell>
<persist>true</persist>
```

IMDSv1

```
Invoke-RestMethod -uri http://169.254.169.254/latest/user-data
<powershell>
$file = $env:SystemRoot + "\Temp\" + (Get-Date).ToString("MM-dd-yy-hh-mm")
New-Item $file -ItemType file
</powershell>
<persist>true</persist>
```

既存インスタンスのインスタンスメタデータオプションのクエリ

既存のインスタンスのインスタンスメタデータオプションをクエリできます。

Console

既存のインスタンスのインスタンスメタデータオプションをクエリするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択してください。
3. インスタンスを選択して次のフィールドを確認します。
   - [IMDSv2] – 値は [必須] または [オプション] のいずれかです。
   - [インスタンスメタデータのタグを許可] – 値は [有効] または [無効] のいずれかです。

4. 選択したインスタンスで、[アクション]、[インスタンスの設定]、[インスタンスメタデータのオプションを変更] の順に選択してください。

   ダイアログボックスには、選択したインスタンスでインスタンスメタデータサービスが有効か無効かが表示されます。

AWS CLI

既存のインスタンスのインスタンスメタデータオプションをクエリするには

describe-instances コマンドを使用します。

```
aws ec2 describe-instances \
    --instance-id i-1234567898abcdef0 \
    --query 'Reservations[].Instances[].MetadataOptions'
```

PowerShell

Tools for PowerShell を使用して既存のインスタンスのインスタンスメタデータオプションをクエリするには

Get-EC2Instance コマンドレットを使用します。

```
(Get-EC2Instance `
    -InstanceId i-1234567898abcdef0).Instances.MetadataOptions
```

リスポンスおよびエラーメッセージ

すべてのインスタンスメタデータがテキスト (HTTP コンテンツタイプ `text/plain`) として返されます。

特定のメタデータリソースに対するリクエストは適切な値または `404 - Not Found` HTTP エラーコード (リソースを使用できない場合) を返します。

一般的なメタデータリソースに対するリクエスト (/ で終わる URI) は使用可能なリソースのリストまたは `404 - Not Found` HTTP エラーコード (使用可能なリソースがない場合) を返します。リスト項目は個別の行に表示され、各行の末尾には改行記号 (ASCII 10) が付いています。

IMDSv1 リクエストが応答を受信しない場合は、IMDSv2 が必要になる可能性があります。

IMDSv2 を使って行われたリクエストでは、次の HTTP エラーコードが返されます。

- `400 - Missing or Invalid Parameters`–PUTリクエストが無効である。
- `401 - Unauthorized`–GETリクエストが無効なトークンを使用している。推奨されるアクションは新しいトークンを生成することです。
- `403 - Forbidden` - リクエストが許可されていないか、あるいは IMDS がオフです。
- `404 - Not Found` – リソースが利用できないか、そのようなリソースがありません。
- `503` - リクエストを完了できませんでした。リクエストを再試行します。

IMDS がエラーを返した場合、 はエラーメッセージを出力にcurl出力し、成功ステータスコードを返します。エラーメッセージは TOKEN変数に保存され、トークンを使用するcurlコマンドは失敗します。-f オプションcurlを指定して を呼び出すと、HTTP サーバーエラーが発生した場合にエラーステータスコードが返されます。エラー処理を有効にすると、シェルはエラーを検出してスクリプトを停止できます。

クエリスロットル

クエリは IMDS でインスタンスごとにスロットリングし、インスタンスから IMDS への同時接続数を制限します。

AWS セキュリティ認証情報を取得するために IMDS を使用している場合、毎回のトランザクションで、または高頻度のスレッドやプロセスから同時に認証情報をクエリしないようにします。スロットリングの原因となる可能性があります。代わりに、認証情報をキャッシュに格納して有効期限が近づくまで待つことをお勧めします。IAM ロールとロールに関連付けられたセキュリティ認証情報の詳細については「インスタンスメタデータからのセキュリティ認証情報の取得」を参照してください。

IMDS にアクセスする際にスロットリングした場合、エクスポネンシャルバックオフ戦略でクエリを再試行します。

インスタンスメタデータサービスを使用してインスタンスメタデータにアクセスする

次のいずれかのメソッドを使って、実行中のインスタンスからインスタンスメタデータにアクセスできます。

- インスタンスメタデータサービスバージョン 2 (IMDSv2) – セッション指向メソッド

  例については「IMDSv2 の例」を参照してください。
- インスタンスメタデータサービスバージョン 1 (IMDSv1) – リクエスト/レスポンスメソッド

例については「IMDSv1 の例」を参照してください。

デフォルトではIMDSv1またはIMDSv2のいずれか、あるいは両方を使用できます。

IMDSv2 呼び出しのみを受け入れるように各インスタンスでインスタンスメタデータサービス (IMDS) を設定できます。これにより、IMDSv1 呼び出しが失敗します。ユーザーに IMDSv2 を使用させるようにインスタンスを設定する方法については「インスタンスメタデータサービスのオプションを設定する」を参照してください。

PUT または GET ヘッダーは IMDSv2 に固有のものです。これらのヘッダーがリクエストに含まれている場合、そのリクエストは IMDSv2 を対象としています。ヘッダーが存在しない場合、そのリクエストは IMDSv1 を対象としているものとみなされます。

IMDSv2 の拡張のレビューの詳細については「EC2 Instance Metadata Service の拡張により、オープンファイアウォール、リバースプロキシ、および SSRF の脆弱性に対して多層防御を追加」を参照してください。

トピック

- インスタンスメタデータサービスバージョン 2 の仕組み
- サポートされる AWS SDK を使用する
- IMDSv2 の例
- IMDSv1 の例

インスタンスメタデータサービスバージョン 2 の仕組み

IMDSv2 はセッション指向リクエストを使用します。セッション指向リクエストを使用して、セッション期間 (1 秒～6 時間) を定義するセッショントークンを作成します。指定した期間中、それ以降のリクエストに同じセッショントークンを使用できます。指定した期間が期限切れになった後、将来のリクエストに使用する新しいセッショントークンを作成する必要があります。

> **Note**
>
> このセクションの例ではインスタンスメタデータサービス (IMDS) の IPv4 アドレス `169.254.169.254` を使用します。IPv6 アドレスを使用して EC2 インスタンスのインスタンスメタデータを取得する場合はIPv6 アドレスを有効にして使用してください。`[fd00:ec2::254]`。IMDS の IPv6 アドレスはIMDSv2 コマンドと互換性がありま

す。IPv6 アドレスにはNitro ベースのインスタンスと IPv6 対応サブネット (デュアルスタックまたは IPv6 のみ) でのみアクセスできます。

次の例ではシェルスクリプトと IMDSv2 を使用して、最上位インスタンスメタデータ項目を取得します。各例の操作は次のとおりです。

- PUTリクエストを使って、6 時間 (21,600 秒) のセッショントークンを作成する
- セッショントークンヘッダーを TOKEN (Linux インスタンス) または token (Windows インスタンス) という名前の変数に保管する
- トークンを使って最上位メタデータアイテムをリクエストする

Linux の例

2 つの個別のコマンドを実行することも、それらを組み合わせることもできます。

個別のコマンド

最初に、次のコマンドを使用してトークンを生成します。

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"`
```

その後、次のコマンドを使用して、トークンを使用して上位レベルのメタデータアイテムを生成します。

```
[ec2-user ~]$ curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/
```

組み合わされたコマンド

トークンを保存し、コマンドを組み合わせることができます。次の例では上記の 2 つのコマンドを組み合わせて、セッショントークンヘッダーを TOKEN という名前の変数に格納します。

> **Note**
>
> トークンの作成時にエラーが発生した場合は有効なトークンの代わりにエラーメッセージが変数に格納され、コマンドは機能しません。

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-
ec2-metadata-token-ttl-seconds: 21600"` \
 && curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/
```

トークンを作成した後、期限切れになるまで再使用することができます。次のコマンド例ではインスタンスの起動に AMI の ID が使用されていますが、前の例で $TOKENに保管されたトークンが再使用されています。

```
[ec2-user ~]$ curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/
meta-data/ami-id
```

Windows の例

```
PS C:\> [string]$token = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token-ttl-
seconds" = "21600"} -Method PUT -Uri http://169.254.169.254/latest/api/token
```

```
PS C:\> Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} -Method GET -
Uri http://169.254.169.254/latest/meta-data/
```

トークンを作成した後、期限切れになるまで再使用することができます。次のコマンド例ではインスタンスの起動に AMI の ID が使用されていますが、前の例で $tokenに保管されたトークンが再使用されています。

```
PS C:\> Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} `
 -Method GET -uri http://169.254.169.254/latest/meta-data/ami-id
```

IMDSv2を使ってインスタンスメタデータをリクエストする際はリクエストに次の項目が含まれている必要があります。

1. PUT リクエストを使って、インスタンスメタデータサービスに対してセッションを開始します。PUT リクエストはインスタンスメタデータサービスに対する後続の GET リクエストに含まれている必要のあるトークンを返します。このトークンはIMDSv2を使ってメタデータにアクセスするのに必要です。
2. トークンを IMDS に対するすべての GET リクエストに含めます。トークンの使用が `required` に設定されている場合、有効なトークンがないリクエスト、または有効期限切れのトークンを持つリクエストで `401 - Unauthorized` HTTP エラーコードが発生します。
   - トークンはインスタンス固有のキーです。トークンは他の EC2 インスタンスで有効ではなく、生成されたインスタンスの外で使用しようとすると拒否されます。

- PUTリクエストにはトークンの有効期限 (TTL) を最大 6 時間 (21,600 秒) まで秒単位で指定するヘッダーが含まれている必要があります。トークンは論理的セッションを表します。TTL はトークンが有効な時間の長さ、つまりセッションの期間を指定します。
- トークンの期限が切れた後、インスタンスメタデータにアクセスし続けるためには別の PUTを使って新しいセッションを作成する必要があります。
- 各リクエストについてトークンを再使用するか、あるいは新しいトークンを作成することを選択できます。少数のリクエストではIMDS にアクセスする必要があるたびに、トークンを生成してすぐに使用するほうが簡単である場合があります。ただし、効率を重視するなら、インスタンスメタデータをリクエストする必要があるたびにPUTリクエストを書くより、トークン期間を長く指定して再使用することができます。それぞれが独自のセッションを表すトークンを同時に使用できる数については実際的に制限がありません。ただし、IMDSv2 では通常の IMDS 接続とスロットリングの制限によって制約を受けます。詳細については「クエリスロットル」を参照してください。

HTTP GETおよびHEADメソッドはIMDSv2インスタンスメタデータリクエストで許可されています。PUT リクエストはX-Forwarded-For ヘッダーが含まれている場合、拒否されます。

デフォルトで、PUT リクエストに対するレスポンスには IP プロトコルレベルで 1 のレスポンスホップリミット (有効期限) があります。より大きなホップリミットが必要な場合はmodify-instance-metadata-options AWS CLI コマンドを使って調整できます。例えば、インスタンスで実行されているコンテナサービスとの下位互換性のために、ホップリミットを拡大する必要がある場合があります。詳細については「既存インスタンスのインスタンスメタデータオプションの変更」を参照してください。

サポートされる AWS SDK を使用する

IMDSv2 を使用するにはEC2 インスタンスが IMDSv2 の使用をサポートする AWS SDK バージョンを使用する必要があります。最新バージョンの AWS SDK はすべて IMDSv2 の使用をサポートしています。

> **Important**
>
> 最新の機能、セキュリティアップデート、および基本的な依存関係を維持するために、SDK のリリースを常に更新することをお勧めします。サポート対象外の SDK バージョンを継続して使用することはお勧めできません。お客様の判断で行ってください。詳細についてはAWS SDK とツールのリファレンスガイド」の「AWS SDK とツールのメンテナンスポリシー」を参照してください。

IMDSv2 の使用をサポートする最小バージョンは次のとおりです。

- AWS CLI – 1.16.289
- AWS Tools for Windows PowerShell – 4.0.1.0
- AWS SDK for .NET – 3.3.634.1
- AWS SDK for C++ – 1.7.229
- AWS SDK for Go – 1.25.38
- AWS SDK for Go v2 – 0.19.0
- AWS SDK for Java – 1.11.678
- AWS SDK for Java 2.x – 2.10.21
- AWS Node.js 内の SDK for JavaScript – 2.722.0
- AWS SDK for Kotlin – 1.1.4
- AWS SDK for PHP – 3.147.7
- AWS SDK for Python (Botocore) – 1.13.25
- AWS SDK for Python (Boto3) – 1.12.6
- AWS SDK for Ruby – 3.79.0

IMDSv2 の例

Amazon EC2 インスタンスで次の例を実行して、IMDSv2 のインスタンスメタデータを取得します。

Windows インスタンスでは、Windows PowerShell を使用するか、cURL または wget をインストールすることができます。Windows インスタンスにサードパーティーツールをインストールする場合は、呼び出しおよび出力がここに記載されているものとは異なる場合があるため、必ず付属のドキュメントをよく読んでください。

例

- 使用できるインスタンスメタデータのバージョンを取得する
- 上位レベルのメタデータ項目を取得する
- メタデータ項目の値を取得する
- 使用可能なパブリックキーのリストを取得する
- パブリックキー 0 が使用できるフォーマットを示す
- パブリックキー 0 を取得する (OpenSSH キーフォーマット)
- インスタンスのサブネット ID を取得する

- インスタンスのインスタンスタグを取得する

使用できるインスタンスメタデータのバージョンを取得する

次の例では使用できるインスタンスメタデータのバージョンを取得しています。各バージョンは新しいインスタンスのメタデータカテゴリがリリースされたときのインスタンスメタデータビルドを参照してください。インスタンスメタデータビルドのバージョンはAmazon EC2 API のバージョンとは相関しません。以前のバージョンに存在する構造および情報に依存するスクリプトがある場合は以前のバージョンを使用することができます。

cURL

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/
1.0
2007-01-19
2007-03-01
2007-08-29
2007-10-10
2007-12-15
2008-02-01
2008-09-01
2009-04-04
2011-01-01
2011-05-01
2012-01-12
2014-02-25
2014-11-05
2015-10-20
2016-04-19
...
latest
```

PowerShell

```
PS C:\> [string]$token = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token-ttl-seconds" = "21600"} -Method PUT -Uri http://169.254.169.254/latest/api/token
```

```
PS C:\> Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} -Method
 GET -Uri http://169.254.169.254/
```

```
1.0
2007-01-19
2007-03-01
2007-08-29
2007-10-10
2007-12-15
2008-02-01
2008-09-01
2009-04-04
2011-01-01
2011-05-01
2012-01-12
2014-02-25
2014-11-05
2015-10-20
2016-04-19
...
latest
```

上位レベルのメタデータ項目を取得する

次の例では上位レベルのメタデータ項目を取得しています。レスポンスの項目の詳細については「インスタンスメタデータのカテゴリ」を参照してください。

アクセスを許可した場合にのみ、タグがこの出力に含まれることに注意してください。詳細については「the section called “インスタンスメタデータ内のタグへのアクセスを有効にする”」を参照してください。

cURL

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/
ami-id
ami-launch-index
ami-manifest-path
block-device-mapping/
events/
hostname
iam/
instance-action
```

```
instance-id
instance-life-cycle
instance-type
local-hostname
local-ipv4
mac
metrics/
network/
placement/
profile
public-hostname
public-ipv4
public-keys/
reservation-id
security-groups
services/
tags/
```

PowerShell

```
PS C:\> [string]$token = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token-ttl-
seconds" = "21600"} -Method PUT -Uri http://169.254.169.254/latest/api/token
```

```
PS C:\> Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} -Method
 GET -Uri http://169.254.169.254/latest/meta-data/
ami-id
ami-launch-index
ami-manifest-path
block-device-mapping/
hostname
iam/
instance-action
instance-id
instance-life-cycle
instance-type
local-hostname
local-ipv4
mac
metrics/
network/
placement/
profile
public-hostname
```

```
public-ipv4
public-keys/
reservation-id
security-groups
services/
tags/
```

メタデータ項目の値を取得する

これらの例では前出の例で取得された一部の最上位メタデータ項目の値を取得しています。これらのリクエストは、前の例のコマンドを使用して作成された保管されているトークンを使用します。このトークンは有効期限が切れていない必要があります。

cURL

```
[ec2-user ~]$ curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/
latest/meta-data/ami-id
ami-0abcdef1234567890
```

```
[ec2-user ~]$ curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/
latest/meta-data/reservation-id
r-0efghijk987654321
```

```
[ec2-user ~]$ curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/
latest/meta-data/local-hostname
ip-10-251-50-12.ec2.internal
```

```
[ec2-user ~]$ curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/
latest/meta-data/public-hostname
ec2-203-0-113-25.compute-1.amazonaws.com
```

PowerShell

```
PS C:\> Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} -Method
 GET -Uri http://169.254.169.254/latest/meta-data/ami-id
ami-0abcdef1234567890
```

```
PS C:\> Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} -Method
 GET -Uri http://169.254.169.254/latest/meta-data/reservation-id
```

```
r-0efghijk987654321
```

```
PS C:\> Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} -Method
 GET -Uri http://169.254.169.254/latest/meta-data/local-hostname
ip-10-251-50-12.ec2.internal
```

```
PS C:\> Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} -Method
 GET -Uri http://169.254.169.254/latest/meta-data/public-hostname
ec2-203-0-113-25.compute-1.amazonaws.com
```

使用可能なパブリックキーのリストを取得する

次の例では使用できるパブリックキーの一覧を取得しています。

cURL

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-
aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-
data/public-keys/
0=my-public-key
```

PowerShell

```
PS C:\> [string]$token = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token-ttl-
seconds" = "21600"} -Method PUT -Uri http://169.254.169.254/latest/api/token
```

```
PS C:\> Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} -Method
 GET -Uri http://169.254.169.254/latest/meta-data/public-keys/
0=my-public-key
```

パブリックキー 0 が使用できるフォーマットを示す

次の例はパブリックキー0のフォーマットを示しています。

cURL

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-
aws-ec2-metadata-token-ttl-seconds: 21600"` \
```

```
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/public-keys/0/
openssh-key
```

PowerShell

```
PS C:\> [string]$token = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token-ttl-seconds" = "21600"} -Method PUT -Uri http://169.254.169.254/latest/api/token
```

```
PS C:\> Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} -Method
 GET -Uri http://169.254.169.254/latest/meta-data/public-keys/0/openssh-key
openssh-key
```

パブリックキー 0 を取得する (OpenSSH キーフォーマット)

次の例ではパブリックキー0を取得しています (OpenSSH キーフォーマット)。

cURL

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/public-keys/0/openssh-key
ssh-rsa MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE my-public-key
```

PowerShell

```
PS C:\> [string]$token = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token-ttl-seconds" = "21600"} -Method PUT -Uri http://169.254.169.254/latest/api/token
```

```
PS C:\> Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} -Method
 GET -Uri http://169.254.169.254/latest/meta-data/public-keys/0/openssh-key
ssh-rsa MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE my-public-key
```

インスタンスのサブネット ID を取得する

次の例ではインスタンスのサブネット ID を取得しています。

cURL

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/network/interfaces/macs/02:29:96:8f:6a:2d/subnet-id
subnet-be9b61d7
```

PowerShell

```
PS C:\> [string]$token = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token-ttl-seconds" = "21600"} -Method PUT -Uri http://169.254.169.254/latest/api/token
```

```
PS C:\> Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} -
Method GET -Uri http://169.254.169.254/latest/meta-data/network/interfaces/
macs/02:29:96:8f:6a:2d/subnet-id
subnet-be9b61d7
```

インスタンスのインスタンスタグを取得する

インスタンスメタデータ内のインスタンスタグへのアクセスが有効になっている場合はインスタンスメタデータからインスタンスのタグを取得できます。詳細については「インスタンスメタデータからタグを取得する」を参照してください。

IMDSv1 の例

Amazon EC2 インスタンスで次の例を実行して、IMDSv1 のインスタンスメタデータを取得します。

Windows インスタンスでは、Windows PowerShell を使用するか、cURL または wget をインストールすることができます。Windows インスタンスにサードパーティーツールをインストールする場合は、呼び出しおよび出力がここに記載されているものとは異なる場合があるため、必ず付属のドキュメントをよく読んでください。

例

- 使用できるインスタンスメタデータのバージョンを取得する
- 上位レベルのメタデータ項目を取得する
- メタデータ項目の値を取得する
- 使用可能なパブリックキーのリストを取得する
- パブリックキー 0 が使用できるフォーマットを示す
- パブリックキー 0 を取得する (OpenSSH キーフォーマット)
- インスタンスのサブネット ID を取得する
- インスタンスのインスタンスタグを取得する

使用できるインスタンスメタデータのバージョンを取得する

次の例では使用できるインスタンスメタデータのバージョンを取得しています。各バージョンは新しいインスタンスのメタデータカテゴリがリリースされたときのインスタンスメタデータビルドを参照してください。インスタンスメタデータビルドのバージョンはAmazon EC2 API のバージョンとは相関しません。以前のバージョンに存在する構造および情報に依存するスクリプトがある場合は以前のバージョンを使用することができます。

cURL

```
[ec2-user ~]$ curl http://169.254.169.254/
1.0
2007-01-19
2007-03-01
2007-08-29
2007-10-10
2007-12-15
2008-02-01
2008-09-01
2009-04-04
2011-01-01
2011-05-01
2012-01-12
2014-02-25
2014-11-05
2015-10-20
2016-04-19
...
latest
```

PowerShell

```
PS C:\> Invoke-RestMethod -uri http://169.254.169.254/
1.0
2007-01-19
2007-03-01
2007-08-29
2007-10-10
2007-12-15
2008-02-01
2008-09-01
2009-04-04
2011-01-01
2011-05-01
2012-01-12
2014-02-25
2014-11-05
2015-10-20
2016-04-19
...
latest
```

上位レベルのメタデータ項目を取得する

次の例では上位レベルのメタデータ項目を取得しています。レスポンスの項目の詳細については「インスタンスメタデータのカテゴリ」を参照してください。

アクセスを許可した場合にのみ、タグがこの出力に含まれることに注意してください。詳細については「the section called “インスタンスメタデータ内のタグへのアクセスを有効にする”」を参照してください。

cURL

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/
ami-id
ami-launch-index
ami-manifest-path
block-device-mapping/
events/
hostname
iam/
instance-action
instance-id
instance-type
local-hostname
local-ipv4
mac
metrics/
network/
placement/
profile
public-hostname
public-ipv4
public-keys/
reservation-id
security-groups
services/
tags/
```

PowerShell

```
PS C:\> Invoke-RestMethod -uri http://169.254.169.254/latest/meta-data/
ami-id
ami-launch-index
```

```
ami-manifest-path
block-device-mapping/
hostname
iam/
instance-action
instance-id
instance-type
local-hostname
local-ipv4
mac
metrics/
network/
placement/
profile
public-hostname
public-ipv4
public-keys/
reservation-id
security-groups
services/
tags/
```

メタデータ項目の値を取得する

これらの例では、前の例で取得した一部の最上位メタデータ項目の値を取得しています。

cURL

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/ami-id
ami-0abcdef1234567890
```

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/reservation-id
r-0efghijk987654321
```

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/local-hostname
ip-10-251-50-12.ec2.internal
```

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/public-hostname
ec2-203-0-113-25.compute-1.amazonaws.com
```

PowerShell

```
PS C:\> Invoke-RestMethod -uri http://169.254.169.254/latest/meta-data/ami-id
ami-0abcdef1234567890
```

```
PS C:\> Invoke-RestMethod -uri http://169.254.169.254/latest/meta-data/reservation-id
r-0efghijk987654321
```

```
PS C:\> Invoke-RestMethod -uri http://169.254.169.254/latest/meta-data/local-hostname
ip-10-251-50-12.ec2.internal
```

```
PS C:\> Invoke-RestMethod -uri http://169.254.169.254/latest/meta-data/public-hostname
ec2-203-0-113-25.compute-1.amazonaws.com
```

使用可能なパブリックキーのリストを取得する

次の例では使用できるパブリックキーの一覧を取得しています。

cURL

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/public-keys/
0=my-public-key
```

PowerShell

```
PS C:\> Invoke-RestMethod -uri http://169.254.169.254/latest/meta-data/public-keys/ 0=my-public-key
```

パブリックキー 0 が使用できるフォーマットを示す

次の例はパブリックキー0のフォーマットを示しています。

cURL

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/public-keys/0/
```

```
openssh-key
```

PowerShell

```
PS C:\> Invoke-RestMethod -uri http://169.254.169.254/latest/meta-data/public-
keys/0/openssh-key
openssh-key
```

パブリックキー 0 を取得する (OpenSSH キーフォーマット)

次の例ではパブリックキー0を取得しています (OpenSSH キーフォーマット)。

cURL

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/public-keys/0/openssh-key
ssh-rsa MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE my-public-key
```

PowerShell

```
PS C:\> Invoke-RestMethod -uri http://169.254.169.254/latest/meta-data/public-
keys/0/openssh-key
ssh-rsa MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
```

```
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE my-public-key
```

インスタンスのサブネット ID を取得する

次の例ではインスタンスのサブネット ID を取得しています。

cURL

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/network/interfaces/
macs/02:29:96:8f:6a:2d/subnet-id
subnet-be9b61d7
```

PowerShell

```
PS C:\> Invoke-RestMethod -uri http://169.254.169.254/latest/meta-data/network/
interfaces/macs/02:29:96:8f:6a:2d/subnet-id
subnet-be9b61d7
```

インスタンスのインスタンスタグを取得する

インスタンスメタデータ内のインスタンスタグへのアクセスが有効になっている場合はインスタンスメタデータからインスタンスのタグを取得できます。詳細については「インスタンスメタデータからタグを取得する」を参照してください。

インスタンスメタデータサービスバージョン 2 の使用への移行

インスタンスメタデータサービスバージョン 2 (IMDSv2) 呼び出しのみを許可するようにインスタンスを設定する場合は、次のツールと移行パスを使用することをお勧めします。

トピック

- IMDSv2 に移行するためのツール
- IMDSv2 を必要とする推奨パス

IMDSv2 に移行するためのツール

次のツールは、IMDSv1 から IMDSv2 へのソフトウェアの移行を特定、モニタリング、管理するのに役立ちます。これらのツールの使用方法については、「IMDSv2 を必要とする推奨パス」を参照してください。

AWS ソフトウェア

最新バージョンの AWS CLI および AWS SDK ではIMDSv2 をサポートしています。IMDSv2 を使用するには、EC2 インスタンスを更新して最新バージョンを使用してください。IMDSv2 をサポートする最低限の AWS SDK バージョンについては「サポートされる AWS SDK を使用する」を参照してください。

すべての Amazon Linux 2 と Amazon Linux 2023 ソフトウェアパッケージが IMDSv2 をサポートしています。Amazon Linux 2023 では、IMDSv1 はデフォルトで無効になっています。

IMDS パケットアナライザー

IMDS パケットアナライザーは、インスタンスの起動フェーズおよびランタイムオペレーション時の IMDSv1 呼び出しを特定してログに記録する、オープンソースツールです。これらのログを解析すると、お使いのインスタンスで IMDSv1 呼び出しを行うソフトウェアを正確に特定し、お使いのインスタンスでのみ IMDSv2 をサポートするために更新する必要があるものを特定できます。IMDS パケットアナライザーはコマンドラインから実行することも、サービスとしてインストールすることもできます。詳細についてはGitHub の AWS ImdsPacketAnalyzer のページを参照してください

CloudWatch

CloudWatch では、インスタンスのモニタリング用に次の 2 つのメトリクスが用意されています。

`MetadataNoToken` – IMDSv2 はトークンベースのセッションを使用します。ただし、IMDSv1 はこれを使用しません。`MetadataNoToken` メトリクスは IMDSv1 を使用しているインスタンスメタデータサービス (IMDS) への呼び出しの数を追跡します。このメトリクスをゼロまでトラッキングすることにより、すべてのソフトウェアが IMDSv2 を使用するようアップグレードされたかどうか、およびいつアップデートが行われたかを測定できます。

`MetadataNoTokenRejected` – IMDSv1 を無効にした後、`MetadataNoTokenRejected` メトリクスを使用して、IMDSv1 呼び出しが試行および拒否された回数を追跡できます。このメトリクスを追跡することで、IMDSv2 を使用するようにソフトウェアを更新する必要があるかどうかを確認できます。

各 EC2 インスタンスのこれらのメトリクスは相互排他的です。IMDSv1 が有効になっている場合 (`httpTokens = optional`) は、`MetadataNoToken` のみが出力されます。IMDSv1 が無効になっている場合 (`httpTokens = required`) は、`MetadataNoTokenRejected` のみが出力されます。これらのメトリクスをいつ使用すべきかについては、「IMDSv2 を必要とする推奨パス」を参照してください。

詳細については「インスタンスメトリクス」を参照してください。

API の起動

新しいインスタンス: RunInstances API を使用して、IMDSv2 の使用を必須とする新しいインスタンスを起動します。詳細については「新規インスタンスのインスタンスメタデータオプションの設定」を参照してください。

既存のインスタンス: ModifyInstanceMetadataOptions API を使用して、既存のインスタンスでの IMDSv2 の使用を必須にします。詳細については「既存インスタンスのインスタンスメタデータオプションの変更」を参照してください。

Auto Scaling グループによって起動された新しいインスタンス: Auto Scaling グループによって起動されたすべての新しいインスタンスで IMDSv2 の使用を必須とするには、Auto Scaling グループで起動テンプレートまたは起動設定を使用できます。起動テンプレートの作成時や起動設定の作成時に、IMDSv2 の使用が必須となるように `MetadataOptions` パラメータを設定する必要があります。Auto Scaling グループは新しい起動テンプレートまたは起動設定を使用して新しいインスタンスは新しいインスタンスを起動しますが、既存のインスタンスは影響を受けません。

Auto Scaling グループ内の既存のインスタンス: ModifyInstanceMetadataOptions API を使用して、既存のインスタンスで IMDSv2 の使用を必須にします。または、インスタンスを終了すると、Auto Scaling グループが新しい起動テンプレートまたは起動設定で定義されているインスタンスメタデータオプション設定を使用して新しい代替インスタンスを起動します。

AMI

`ImdsSupport` パラメータを `v2.0` に設定して構成された AMI は、デフォルトで IMDSv2 を必要とするインスタンスを起動します。Amazon Linux 2023 は `ImdsSupport = v2.0` を使用して設定されます。

新しい AMI: 新しい AMI を作成するときは、register-image CLI コマンドを使用して `ImdsSupport` パラメータを `v2.0` に設定します。

既存の AMI: 既存の AMI を変更するときは、modify-image-attribute CLI コマンドを使用して `ImdsSupport` パラメータを `v2.0` に設定しますい。

詳細については「AMI を設定する」を参照してください。

アカウントレベルのコントロール

すべてのインスタンスメタデータオプションのデフォルト値をアカウントレベルで設定できます。デフォルト値は、インスタンスの起動時に自動的に適用されます。詳細については、「IMDSv2 をアカウントのデフォルトとして設定する」を参照してください。

IMDSv2 を使用する要件をアカウントレベルで強制適用することもできます。IMDSv2 の強制適用が有効になっている場合:

- 新しいインスタンス: IMDSv1 が有効化された状態で起動するように設定されたインスタンスは起動が失敗します
- IMDSv1 が無効になっている既存のインスタンス: 既存のインスタンスで IMDSv1 を有効にする試みは阻止されます。
- IMDSv1 が有効になっている既存のインスタンス: IMDSv1 が既に有効になっている既存のインスタンスは影響を受けません。

詳細については「アカウントレベルで IMDSv2 を強制適用する」を参照してください。

IAM ポリシーおよび SCP

以下に示すように、ユーザーの管理にはIAM ポリシーを使用することも、AWS Organizations サービスコントロールポリシー (SCP) を使用することもできます。

- インスタンスが IMDSv2 を使用するように設定されていない限り、RunInstances API を使用してそのインスタンスを起動することはできません。
- ModifyInstanceMetadataOptions API を使用して実行中のインスタンスを変更し、IMDSv1 を再度有効にすることはできません。

IAM ポリシーまたは SCP には次の IAM 条件キーを含める必要があります。

- `ec2:MetadataHttpEndpoint`
- `ec2:MetadataHttpPutResponseHopLimit`
- `ec2:MetadataHttpTokens`

API および CLI 呼び出し時のパラメータが、条件キーが含まれているポリシーで指定した状態と一致しない場合、これらの API または CLI の呼び出しは失敗し `UnauthorizedOperation` レスポンスが返されます。

さらに、追加の保護レイヤーを選択して、IMDSv1からIMDSv2の変更を強制することもできます。EC2 ロールの認証情報経由で呼び出された各 API に関するアクセス管理レイヤーでは、IAM ポリシーまたは AWS Organizations サービスコントロールポリシー (SCP) で条件キーを使用

できます。具体的にはIAM ポリシーで値 `2.0` を設定した条件キー `ec2:RoleDelivery` を使用していると、IMDSv1 から取得した EC2 ロールの認証情報を使用した API コールに対して、`UnauthorizedOperation` レスポンスが返されます。同じことはSCP によって義務付けられる条件を使ってより広く達成できます。これにより、指定した条件と一致しない API コールに対しては `UnauthorizedOperation` エラーが返されるため、実際に IMDSv1 から取得した認証情報を使用して API を呼び出すことはできなくなります。

IAM ポリシーの例はインスタンスメタデータの使用を参照してください。SCP の詳細については「AWS Organizations ユーザーガイド 」の「サービスコントロールポリシー (SCPs)」を参照してください。

宣言型ポリシー

宣言型ポリシー (AWS Organizations の機能) を使用して、組織全体の IMDS アカウントデフォルト (IMDSv2 の強制適用を含む) を一元的に設定します。ポリシーの例については、「AWS Organizations ユーザーガイド」の「サポートされている宣言型ポリシー」セクションにある「インスタンスメタデータ」タブを参照してください。

IMDSv2 を必要とする推奨パス

上記のツールを使用する際、IMDSv2 への移行において次のパスを使用することを推奨します。

- ステップ 1: IMDSv2=optional を使用するインスタンスを特定し、IMDSv1 の使用状況を監査する
- ステップ 2: ソフトウェアを IMDSv2 に更新する
- ステップ 3: インスタンスで IMDSv2 を要求する
- ステップ 4: IMDSv2=required をデフォルトとして設定する
- ステップ 5: インスタンスに IMDSv2 を要求するように強制する

ステップ 1: IMDSv2=optional を使用するインスタンスを特定し、IMDSv1 の使用状況を監査する

IMDSv2 移行の範囲を評価するには、IMDSv1 または IMDSv2 のいずれかを許可するように設定されたインスタンスを特定し、IMDSv1 呼び出しを監査します。

1. IMDSv1 または IMDSv2 のいずれかを許可するように設定されたインスタンスを特定する:

   Amazon EC2 console

   1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
   2. ナビゲーションペインで、[Instances] (インスタンス) を選択してください。

3. IMDSv1 または IMDSv2 を許可するように設定されたインスタンスのみを表示するには、フィルター「IMDSv2 = optional」を追加します。
4. または、IMDSv2 がすべてのインスタンスでオプションか必須かを確認するには、[設定] ウィンドウ (歯車アイコン) を開き、[IMDSv2] をオンにして [確認] を選択します。これにより IMDSv2 列がインスタンステーブルに追加されます。

AWS CLI

describe-instances コマンドを使用して、次のように `metadata-options.http-tokens = optional` でフィルタリングします。

```
aws ec2 describe-instances --filters "Name=metadata-options.http-
tokens,Values=optional" --query "Reservations[*].Instances[*].[InstanceId]" --
output text
```

2. 各インスタンスで IMDSv1 呼び出しを監査する:

   CloudWatch メトリクス `MetadataNoToken` を使用します。このメトリクスはインスタンスの IMDS に対する IMDSv1 呼び出しの数を示します。詳細については、「インスタンスのメトリクス」を参照してください。
3. IMDSv1 呼び出しを行うインスタンス上のソフトウェアを特定する:

   オープンソースの IMDS パケットアナライザーを使用して、インスタンスの起動フェーズおよびランタイムオペレーション中に IMDSv1 呼び出しを特定し、ログに記録します。この情報を使用して、更新するソフトウェアを特定し、インスタンスが IMDSv2 のみを使用する準備を行います。IMDS パケットアナライザーはコマンドラインから実行することも、サービスとしてインストールすることもできます。

ステップ 2: ソフトウェアを IMDSv2 に更新する

インスタンスでロール認証情報を使用するすべてのSDK、CLI、ソフトウェアを IMDSv2 対応バージョンに更新します。詳細については AWS Command Line Interface ユーザーガイド」の「AWS CLI の最新バージョンをインストールまたは更新」を参照してください。

ステップ 3: インスタンスで IMDSv2 を要求する

`MetadataNoToken` メトリクスを通じて IMDSv1 呼び出しがゼロになったことを確認したら、IMDSv2 を要求するように既存のインスタンスを設定します。また、IMDSv2 を要求するように

すべての新しいインスタンスを設定します。つまり、すべての既存のインスタンスと新しいインスタンスで IMDSv1 を無効にします。

1. IMDSv2 を要求するように既存のインスタンスを設定する:

   Amazon EC2 console

   1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
   2. ナビゲーションペインで、[Instances] (インスタンス) を選択してください。
   3. インスタンスを選択してください。
   4. [アクション]、[インスタンスの設定]、[インスタンスメタデータのオプションを変更] の順に選択してください。
   5. [IMDSv2] の場合は[必須] を選択してください。
   6. [Save] を選択してください。

   AWS CLI

   IMDSv2 のみを使用するように指定するには、modify-instance-metadata-options CLI コマンドを使用します。

   > **Note**
   >
   > この設定は、実行中のインスタンスで変更できます。変更は、インスタンスを再起動しなくてもすぐに有効になります。

   詳細については「IMDSv2 の使用を要求する」を参照してください。

2. IMDSv1 を無効にした後の問題を監視する:

   1. `MetadataNoTokenRejected` CloudWatch メトリクスを使用して、IMDSv1 呼び出しが試行および拒否された回数を追跡できます。
   2. ソフトウェアの問題が発生しているインスタンスで `MetadataNoTokenRejected` メトリクスが IMDSv1 呼び出しを記録する場合、これは IMDSv2 を使用するためにソフトウェアを更新する必要があることを示します。

3. IMDSv2 を要求するように新しいインスタンスを設定する:

Amazon EC2 console

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. インスタンスを起動するには、次のステップに沿って操作します。
3. [高度な詳細] を展開し、[メタデータバージョン] で [V2 のみ (トークンが必要)] を選択します。
4. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択してください。

   詳細については「起動時にインスタンスを設定する」を参照してください。

AWS CLI

AWS CLI: run-instances CLI コマンドを使用して、IMDSv2 が必須となるように指定します。

ステップ 4: IMDSv2=required をデフォルトとして設定する

IMDSv2=required は、アカウントレベルまたは組織レベルのいずれかでデフォルト設定として設定できます。これにより、新しく起動されたすべてのインスタンスが IMDSv2 を要求するように自動的に設定されます。

1. アカウントレベルのデフォルトを設定する:

   Amazon EC2 console

   1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
   2. ナビゲーションペインで、ダッシュボードを選択してください。
   3. [アカウント属性] カードの [設定] で、[データ保護とセキュリティ] を選択します。
   4. [IMDS のデフォルト] で [管理] を選択します。
   5. [インスタンスメタデータサービス] で、[有効にする] を選択します。
   6. [メタデータバージョン] で、[V2 のみ (トークンが必要)] を選択します。
   7. [Update] (更新) を選択してください。

AWS CLI

modify-instance-metadata-defaults CLI コマンドを使用して、`--http-tokens required` と `--http-put-response-hop-limit 2` を指定します。

詳細については「IMDSv2 をアカウントのデフォルトとして設定する」を参照してください。

2. または、宣言型ポリシーを使用して組織レベルのデフォルトを設定します。

宣言型ポリシーを使用して、IMDSv2 の組織のデフォルトを必須に設定します。ポリシーの例については、「AWS Organizations ユーザーガイド」の「サポートされている宣言型ポリシー」セクションにある「インスタンスメタデータ」タブを参照してください。

ステップ 5: インスタンスに IMDSv2 を要求するように強制する

どのインスタンスにも IMDSv1 への依存がないことを確認したら、すべての新しいインスタンスに IMDSv2 を強制適用することが推奨されます。

次の 2 つのオプションのいずれかを使用して IMDSv2 を強制適用します。

1. アカウントプロパティを使用して IMDSv2 を強制適用する

各 AWS リージョンに対し、アカウントレベルで IMDSv2 の使用を強制できます。使用が強制されると、インスタンスを起動できるのは IMDSv2 が必須として設定されている場合のみになります。この強制は、インスタンスまたは AMI の設定方法を問わずに適用されます。詳細については「アカウントレベルで IMDSv2 を強制適用する」を参照してください。この設定を組織レベルで適用するには、宣言型ポリシーを設定します。ポリシーの例については、「AWS Organizations ユーザーガイド」の「サポートされている宣言型ポリシー」セクションにある「インスタンスメタデータ」タブを参照してください。

強制適用が取り消されないようにするため、IAM ポリシーを使用して ModifyInstanceMetadataDefaults API へのアクセスを防ぐ必要があります。詳細については「IAM ポリシーを使用する」を参照してください。

> **Note**
>
> この設定は既存のインスタンスの IMDS バージョンを変更しませんが、IMDSv1 が現在無効になっている既存のインスタンスでの IMDSv1 の有効化が阻止されます。

> **Warning**
>
> IMDSv2 の強制適用が有効になっており、`httpTokens` が起動時のインスタンス設定、アカウント設定、または AMI 構成のいずれかで `required` に設定されていない場合、インスタンスの起動は失敗します。トラブルシューティング情報については、「IMDSv1 が有効化されたインスタンスの起動が失敗する」を参照してください。

2. 別の手段として、次の IAM または SCP 条件キーを用いて IMDSv2 を強制適用します。
   - `ec2:MetadataHttpTokens`
   - `ec2:MetadataHttpPutResponseHopLimit`
   - `ec2:MetadataHttpEndpoint`

   これらの条件キーは RunInstances API と ModifyInstanceMetadataOptions API、および対応する CLI の使用を制御します。ポリシーを作成し、条件キーを使用してポリシーに指定した状態と API コールのパラメータが一致しない場合、API コールまたは CLI コールは失敗して `UnauthorizedOperation` レスポンスが返されます。

   IAM ポリシーの例はインスタンスメタデータの使用を参照してください。

インスタンスメタデータサービスへのアクセスを制限する

ローカルファイアウォールルールを使って、プロセスの一部またはすべてからインスタンスメタデータサービス (IMDS) へのアクセスを無効化することを検討できます。

Nitro ベースのインスタンスではVPC 内のネットワークアプライアンス (仮想ルーターなど) がパケットを IMDS アドレスに転送し、かつ、インスタンス上のデフォルトの送信元/送信先チェックが無効な場合、IMDS がユーザー自身のネットワークから到達可能になります。VPC の外側にある送信元から IMDS に到達しないようにするには送信先 IMDS の IPv4 アドレスが `169.254.169.254` (IPv6 エンドポイントを有効にしている場合はIMDS の IPv6 アドレスが `[fd00:ec2::254]`) であるパケットをドロップするように、ネットワークアプライアンスの設定を変更することをお勧めします。

Linux インスタンスの IMDS アクセスを制限する

iptables を使ったアクセス制限

次の例ではLinux iptables およびその`owner`モジュールを使って、Apache ウェブサーバーが (デフォルトインストールユーザー ID `apache`に基づいて) 169.254.169.254 にアクセスするのを防ぐことが

できます。拒否ルールを使って、そのユーザーとして実行中のプロセスからのインスタンスメタデータリクエスト (IMDSv1またはIMDSv2) をすべて拒否します。

```
$ sudo iptables --append OUTPUT --proto tcp --destination 169.254.169.254 --match owner
 --uid-owner apache --jump REJECT
```

また、ルールの許可を使うことで、特定のユーザーまたはグループへのアクセスを許可することを検討できます。ルールの許可はどのソフトウェアがインスタンスメタデータへのアクセスが必要かについてユーザーが決定しなければならないため、セキュリティ観点からみたときに管理しやすいかもしれません。ルールの許可 を使用すると、後にインスタンスのソフトウェアまたは構成を変更した場合に、誤ってソフトウェアがメタデータサービス (アクセスする意図がなかった) にアクセスするのを許可する可能性が低くなります。また、ファイアウォールのルールを変更しなくても許可されたグループにユーザーを追加/削除できるよう、グループ使用をルールの許可と組み合わせることもできます。

次の例ではユーザーアカウント `trustworthy-user` で実行中のプロセス以外のすべてのプロセスによる IMDS へのアクセスを禁止しています。

```
$ sudo iptables --append OUTPUT --proto tcp --destination 169.254.169.254 --match
 owner ! --uid-owner trustworthy-user --jump REJECT
```

> **Note**
>
> - ローカルファイアウォールルールを使用するには前の例のコマンドをニーズに合わせて変更する必要があります。
> - デフォルトではiptables ルールはシステム再起動全体で永続しません。ここには説明されていない OS 機能を使って永続的にすることができます。
> - iptables ownerモジュールはグループが所定のローカルユーザーのプライマリグループである場合にのみツールメンバーシップと一致します。他のグループは一致しません。

PF または IPFW を使ってアクセスを制限する

FreeBSD または OpenBSD を使用している場合、PF または IPFW の使用も検討できます。次の例ではIMDS へのアクセスをルートユーザーにのみ制限しています。

PF

```
$ block out inet proto tcp from any to 169.254.169.254
```

```
$ pass out inet proto tcp from any to 169.254.169.254 user root
```

IPFW

```
$ allow tcp from any to 169.254.169.254 uid root
```

```
$ deny tcp from any to 169.254.169.254
```

> **Note**
>
> PF および IPFW コマンドの順序は重要となります。PF のデフォルトは最後に一致したルールであり、IPFW のデフォルトは最初に一致したルールです。

Windows インスタンスの IMDS アクセスを制限する

Windows ファイアウォールを使ってアクセスを制限する

次の PowerShell 例では組み込み Windows ファイアウォールを使って、インターネット情報サービスウェブサーバー (デフォルトインストールユーザー ID のNT AUTHORITY\IUSRに基づいて) が 169.254.169.254 にアクセスするのを防いでいます。拒否ルールを使って、そのユーザーとして実行中のプロセスからのインスタンスメタデータリクエスト (IMDSv1またはIMDSv2) をすべて拒否します。

```
PS C:\> $blockPrincipal = New-Object -TypeName System.Security.Principal.NTAccount ("NT
 AUTHORITY\IUSR")
PS C:\> $BlockPrincipalSID =
 $blockPrincipal.Translate([System.Security.Principal.SecurityIdentifier]).Value
PS C:\> $BlockPrincipalSDDL = "D:(A;;CC;;;$BlockPrincipalSID)"
PS C:\> New-NetFirewallRule -DisplayName "Block metadata service from IIS" -Action
 block -Direction out `
-Protocol TCP -RemoteAddress 169.254.169.254 -LocalUser $BlockPrincipalSDDL
```

また、ルールの許可を使うことで、特定のユーザーまたはグループへのアクセスを許可することを検討できます。ルールの許可はどのソフトウェアがインスタンスメタデータへのアクセスが必要かにつ

いてユーザーが決定しなければならないため、セキュリティ観点からみたときに管理しやすいかもしれません。ルールの許可 を使用すると、後にインスタンスのソフトウェアまたは構成を変更した場合に、誤ってソフトウェアがメタデータサービス (アクセスする意図がなかった) にアクセスするのを許可する可能性が低くなります。また、ファイアウォールのルールを変更しなくても許可されたグループにユーザーを追加/削除できるよう、グループ使用をルールの許可と組み合わせることもできます。

次の例ではexceptionPrincipalで指定したプロセス (この例ではtrustworthy-usersと呼ばれるグループ) 以外の、変数 blockPrincipal (この例では Windows グループEveryone) で指定された OS グループとして実行中のすべてのプロセスによるインスタンスメタデータへのアクセスを禁止しています。Windows ファイアウォールはLinux iptables の！ --uid-owner trustworthy-userとは異なり、その他すべてを拒否することにより、特定のプリンシパル (ユーザーまたはグループ) のみを許可するショートカット機構を提供しないため、拒否と許可プリンシパルの両方を指定する必要があります。

```
PS C:\> $blockPrincipal = New-Object -TypeName System.Security.Principal.NTAccount
 ("Everyone")
PS C:\> $BlockPrincipalSID =
 $blockPrincipal.Translate([System.Security.Principal.SecurityIdentifier]).Value
PS C:\> $exceptionPrincipal = New-Object -TypeName System.Security.Principal.NTAccount
 ("trustworthy-users")
PS C:\> $ExceptionPrincipalSID =
 $exceptionPrincipal.Translate([System.Security.Principal.SecurityIdentifier]).Value
PS C:\> $PrincipalSDDL = "O:LSD:(D;;CC;;;$ExceptionPrincipalSID)(A;;CC;;;
$BlockPrincipalSID)"
PS C:\> New-NetFirewallRule -DisplayName "Block metadata service for
 $($blockPrincipal.Value), exception: $($exceptionPrincipal.Value)" -Action block -
Direction out `
-Protocol TCP -RemoteAddress 169.254.169.254 -LocalUser $PrincipalSDDL
```

> **Note**
>
> ローカルファイアウォールルールを使用するには前の例のコマンドを二ーズに合わせて変更する必要があります。

netsh ルールを使ってアクセスを制限する

netshルールを使ってすべてのソフトウェアをブロックすることを検討できますが、柔軟性は大幅に低下します。

```
C:\> netsh advfirewall firewall add rule name="Block metadata service altogether"
 dir=out protocol=TCP remoteip=169.254.169.254 action=block
```

Note

- ローカルファイアウォールルールを使用するには前の例のコマンドをニーズに合わせて変更する必要があります。
- netshルールは elevated コマンドプロンプトから設定する必要があり、拒否または許可の特定のプリンシパルに設定できません。

インスタンスメタデータサービスのオプションを設定する

インスタンスメタデータサービス (IMDS) はすべての EC2 インスタンスでローカルに実行されます。インスタンスメタデータオプションはEC2 インスタンス上の IMDS のアクセシビリティと動作を制御する一連の設定を参照してください。

各インスタンスで、以下のインスタンスメタデータオプションを設定できます。

[インスタンスメタデータサービス (IMDS)]: `enabled` | `disabled`

インスタンスで IMDS を有効または無効にすることができます。無効にすると、ユーザーまたはコードはインスタンスのインスタンスメタデータにアクセスできなくなります。

IMDS のインスタンスにはIPv4 (`169.254.169.254`) と IPv6 (`[fd00:ec2::254]`) という 2 つのエンドポイントがあります。IMDS を有効にすると、IPv4 エンドポイントが自動的に有効になります。IPv6 エンドポイントを有効にする場合は明示的に有効にする必要があります。

[IMDS IPv6 エンドポイント]: `enabled` | `disabled`

インスタンスで IPv6 IMDS エンドポイントを明示的に有効にできます。IPv6 エンドポイントが有効になっている場合、IPv4 エンドポイントは有効なままになります。IPv6 エンドポイントはNitro ベースのインスタンスと IPv6 対応サブネット (デュアルスタックまたは IPv6 のみ) でのみサポートされています。

[メタデータのバージョン]: `IMDSv1 or IMDSv2 (token optional)` | `IMDSv2 only (token required)`

インスタンスメタデータをリクエストするとき、IMDSv2 呼び出しはトークンを要求します。IMDSv1 呼び出はトークンを要求しません。IMDSv1 または IMDSv2 呼び出しを許可する

(トークンがオプションの場合) か、IMDSv2 呼び出しのみを許可する (トークンが必須の場合) ように、インスタンスを設定できます。

[メタデータレスポンスのホップ制限]: 1 ~ 64

ホップ制限はPUT レスポンスが実行できるネットワークホップの数です。ホップ制限は最小 1、最大 64 に設定できます。コンテナ環境では、1 のホップ制限によって問題が発生する可能性があります。これらの問題を軽減する方法については、インスタンスメタデータアクセス考慮事項でコンテナ環境に関する情報を参照してください。

[インスタンスメタデータ内のタグにアクセスする]: `enabled` | `disabled`

インスタンスのメタデータからインスタンスのタグへのアクセスを有効または無効にすることができます。詳細についてはインスタンスメタデータを使用して EC2 インスタンスのタグを表示する を参照してください。

インスタンスの現在の設定を確認するには、「既存インスタンスのインスタンスメタデータオプションのクエリ」を参照してください。

インスタンスメタデータオプションを設定する場所

インスタンスメタデータオプションは次のようにさまざまなレベルで設定できます。

- アカウント – 各 AWS リージョンのアカウントレベルで、インスタンスメタデータオプションのデフォルト値を設定できます。インスタンスが起動すると、インスタンスメタデータオプションは自動的にアカウントレベルの値に設定されます。これらの値は起動時に変更できます。アカウントレベルのデフォルト値は既存のインスタンスには影響しません。
- AMI – AMI を登録または変更するときに、`imds-support` パラメータを `v2.0` に設定できます。この AMI でインスタンスを起動すると、インスタンスメタデータバージョンは自動的に「IMDSv2」に設定され、ホップ制限は「2」に設定されます。
- インスタンス – 起動時にインスタンスのすべてのインスタンスメタデータオプションを変更し、デフォルト設定を上書きできます。実行中または停止中のインスタンスで起動した後に、インスタンスメタデータオプションを変更することもできます。変更は IAM または SCP ポリシーによって制限される可能性があることに注意してください。

詳細については新規インスタンスのインスタンスメタデータオプションの設定および既存インスタンスのインスタンスメタデータオプションの変更を参照してください。

インスタンスメタデータオプションの優先順位

各インスタンスメタデータオプションの値はインスタンスの起動時に優先順位に従って決定されます。最上位の優先順位を持つ階層は次のとおりです。

- 優先順位 1: 起動時のインスタンス設定 – 値は起動テンプレートまたはインスタンス設定のいずれかで指定できます。ここで指定された値はアカウントレベルまたは AMI で指定された値を上書きします。
- 優先順位 2: アカウント設定 – インスタンスの起動時に値が指定されていない場合はアカウントレベルの設定 (AWS リージョンごとに設定) によって値が決まります。アカウントレベルの設定では各メタデータオプションの値が含まれているか、まったく指定がないことが示されるかのどちらかです。
- 優先順位 3: AMI 設定 – インスタンスの起動時に、またはアカウントレベルで値が指定されていない場合はAMI 設定によって値が決まります。これは`HttpTokens` と `HttpPutResponseHopLimit` にのみ該当します。

各メタデータオプションは個別に評価されます。インスタンスは直接インスタンス設定、アカウントレベルのデフォルト、および AMI からの設定を組み合わせて設定できます。

IAM または SCP ポリシーによって変更が制限されていない限り、実行中または停止中のインスタンスで起動した後に、任意のメタデータオプションの値を変更できます。

> **Note**
>
> アカウントレベルでの IMDSv2 の強制適用設定の評価は、優先順位によってインスタンスの IMDS 設定が判断された後で行われます。IMDSv2 の強制適用を有効にすると、IMDSv1 が有効化されているインスタンスは失敗します。強制適用に関する詳細については、「アカウントレベルで IMDSv2 を強制適用する」を参照してください。

> **Warning**
>
> IMDSv2 の強制適用が有効になっており、`httpTokens` が起動時のインスタンス設定、アカウント設定、または AMI 構成のいずれかで `required` に設定されていない場合、起動は失敗します。

例 1 – メタデータオプションの値を判断する

この例ではEC2 インスタンスはアカウントレベルで HttpPutResponseHopLimit が 1 に設定されているリージョンで起動されます。指定された AMI ではImdsSupport が v2.0 に設定されています。起動時に、インスタンスでメタデータオプションが直接指定されることはありません。インスタンスは次のメタデータオプションを使用して起動されます。

```
"MetadataOptions": {
    ...
    "HttpTokens": "required",
    "HttpPutResponseHopLimit": 1,
    ...
```

これらの値は次のように決定されました。

- 起動時にメタデータオプションが指定されていない: インスタンスの起動時に、メタデータオプションの特定の値が、インスタンス起動パラメータにも、起動テンプレートにも指定されていませんでした。
- アカウント設定が次に優先される: 起動時に特定の値が指定されていない場合はリージョン内のアカウントレベルの設定が優先されます。つまり、アカウントレベルで設定されたデフォルト値が適用されます。この場合はHttpPutResponseHopLimit が 1 に設定されました。
- AMI 設定が最後に優先される: 起動時またはアカウントレベルで HttpTokens (インスタンスメタデータバージョン) に特定の値が指定されていない場合はAMI 設定が適用されます。この場合は AMI 設定 ImdsSupport: v2.0 により、HttpTokens が required に設定されました。AMI 設定 ImdsSupport: v2.0 は HttpPutResponseHopLimit: 2 設定するように設計されていますが、優先順位の高いアカウントレベルの設定 HttpPutResponseHopLimit: 1 によって上書きれることに注意してください。

例 2 – メタデータオプションの値を判断する

この例ではEC2 インスタンスは前の例 1 と同じ設定で起動されますが、起動時にインスタンスで直接 HttpTokens が optional に設定されています。インスタンスは次のメタデータオプションを使用して起動されます。

```
"MetadataOptions": {
    ...
    "HttpTokens": "optional",
    "HttpPutResponseHopLimit": 1,
    ...
```

HttpPutResponseHopLimit の値は例 1 と同じ方法で決定されます。ただし、HttpTokens の値は次のように決定されます。起動時にインスタンスで設定されているメタデータオプションが最優先されます。AMI が ImdsSupport: v2.0 で設定されている (つまり、HttpTokens が required に設定されている) 場合でも、起動時にインスタンスで指定されている値 (HttpTokens が optional に設定されている) が優先されます。

例 3 – HttpTokensEnforced が有効化されているメタデータオプションの値を判断する

この例では、リージョン内のアカウントに HttpTokens = required と HttpTokensEnforced = enabled があります。

次の EC2 インスタンス起動試行を検討します。

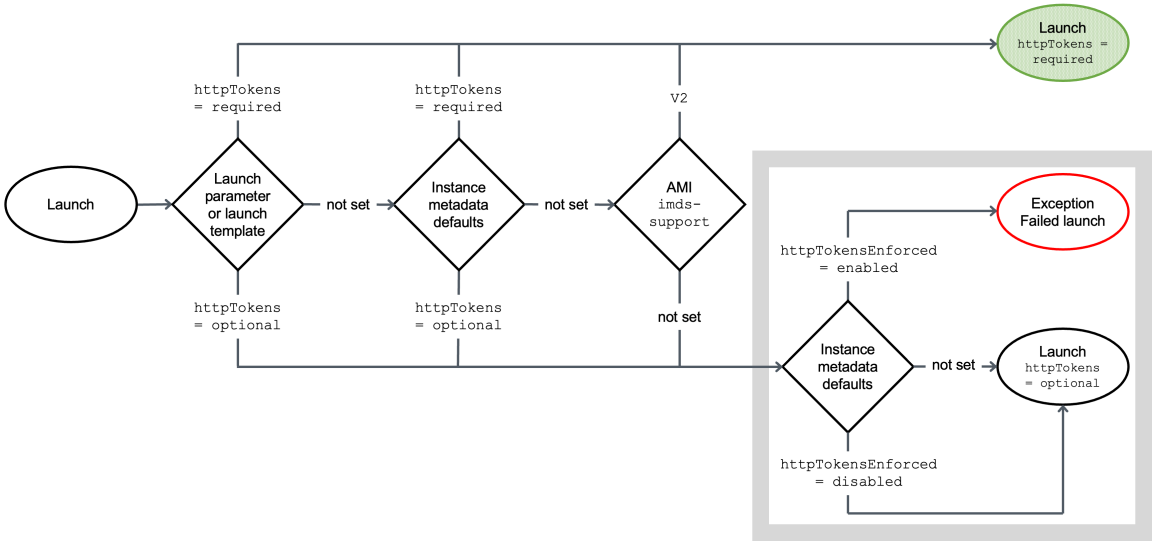
- HttpTokens を optional に設定した起動試行 – アカウントレベルでの強制適用が有効 (HttpTokensEnforced = enabled) になっており、アカウントのデフォルトよりも起動パラメータが優先されるため、起動が失敗します。
- HttpTokens を required に設定した起動試行 — アカウントレベルの強制適用に従っているため、起動は正常に行われます。
- HttpTokens 値が指定されていない起動試行 – 値はアカウント設定に基づいてデフォルトの required になるため、起動は正常に行われます。

インスタンスのメタデータバージョンを設定する

インスタンスの起動時におけるインスタンスメタデータバージョンの値は、[IMDSv1 または IMDSv2 (トークンはオプション)] (httpTokens=optional) か [IMDSv2 のみ (トークンは必須)] (httpTokens=required) のいずれかになります。

インスタンスの起動時に、メタデータバージョンの値を手動で指定するか、デフォルト値を使用できます。値を手動で指定すると、すべてのデフォルト値が上書きされます。値を手動で指定しないことを選択した場合は、デフォルト設定の組み合わせによって判断されます。

次のフローチャートは、起動時におけるインスタンスメタデータバージョンが、構成の異なるレベルでの設定と、強制適用が評価される場所によって判断される仕組みを示しています。次の表は、各レベル固有の設定を示しています。



この表は起動時のインスタンスのメタデータバージョン (列 4 の [結果として生じるインスタンス設定] で示される) が、さまざまなレベルの設定によってどのように決定されるかを示しています。優先順位は左から右で、次のように最初の列が最も優先されます。

- 列 1: [起動パラメータ] - 起動時に手動で指定するインスタンスの設定を表します。
- 列 2: [アカウントレベルのデフォルト] - アカウントの設定を表します。
- 列 3: [AMI のデフォルト] - AMI の設定を表します。

起動パラメータ	アカウントレベルのデフォルト	AMI のデフォルト	結果として生じるインスタンス設定
V2 のみ (トークンが必要)	指定なし	V2 のみ	V2 のみ
V2 のみ (トークンが必要)	V2 のみ	V2 のみ	V2 のみ
V2 のみ (トークンが必要)	V1 または V2	V2 のみ	V2 のみ
V1 または V2 (トークンはオプション)	指定なし	V2 のみ	V1 または V2

起動パラメータ	アカウントレベルのデフォルト	AMI のデフォルト	結果として生じるインスタンス設定
V1 または V2 (トークンはオプション)	V2 のみ	V2 のみ	V1 または V2
V1 または V2 (トークンはオプション)	V1 または V2	V2 のみ	V1 または V2
未設定	指定なし	V2 のみ	V2 のみ
未設定	V2 のみ	V2 のみ	V2 のみ
未設定	V1 または V2	V2 のみ	V1 または V2
V2 のみ (トークンが必要)	指定なし	null	V2 のみ
V2 のみ (トークンが必要)	V2 のみ	null	V2 のみ
V2 のみ (トークンが必要)	V1 または V2	null	V2 のみ
V1 または V2 (トークンはオプション)	指定なし	null	V1 または V2
V1 または V2 (トークンはオプション)	V2 のみ	null	V1 または V2
V1 または V2 (トークンはオプション)	V1 または V2	null	V1 または V2
未設定	指定なし	null	V1 または V2
未設定	V2 のみ	null	V2 のみ
未設定	V1 または V2	null	V1 または V2

IAM 条件キーを使用してインスタンスメタデータオプションを制限する

IAM ポリシーまたは SCP で IAM 条件キーを次のように使用できます。

- IMDSv2 の使用を要求するようにインスタンスが設定されている場合にのみ、インスタンスの起動を許可する
- ホップの許可数を制限する
- インスタンスメタデータへのアクセスを無効にする

タスク

- 新規インスタンスのインスタンスメタデータオプションの設定
- 既存インスタンスのインスタンスメタデータオプションの変更

> **Note**
>
> 注意深く実行し、変更を行う前に慎重なテストを実施する必要があります。以下の情報を記録します。
>
> - IMDSv2の使用を強制する場合、インスタンスメタデータアクセスのためにIMDSv1を使用するアプリケーションまたはエージェントは休憩します。
> - インスタンスメタデータへのアクセスをすべてオフにする場合、インスタンスメタデータアクセスに依存して機能するアプリケーションまたはエージェントは休憩します。
> - IMDSv2 でトークンを取得する際には`/latest/api/token` を使用する必要があります。
> - (Windows のみ) PowerShell のバージョンが 4.0 より前の場合はIMDSv2 の使用を要求するために Windows Management Framework 4.0 に更新する必要があります。

新規インスタンスのインスタンスメタデータオプションの設定

新規インスタンスに、以下のインスタンスメタデータオプションを設定できます。

オプション

- IMDSv2 の使用を要求する
- IMDS IPv4 および IPv6 エンドポイントを有効にする
- インスタンスメタデータへのアクセスを無効にする

- インスタンスメタデータのタグへのアクセスを許可する

> **Note**
>
> これらのオプションの設定はアカウントレベルで直接、または宣言ポリシーを使用して設定されます。これらはインスタンスメタデータオプションを設定する各 AWS リージョン で設定する必要があります。宣言型ポリシーを使用すると、複数の リージョンと複数の アカウントで同時に設定を適用できます。宣言ポリシーが使用されている場合、アカウント内で直接設定を変更することはできません。このトピックでは、アカウント内で設定を直接設定する方法について説明します。宣言ポリシーの使用の詳細についてはAWS Organizations「 ユーザーガイド」の「宣言ポリシー」を参照してください。

IMDSv2 の使用を要求する

次の方法を使用して、新しいインスタンスで IMDSv2 の使用を必須にすることができます。

IMDSv2 を必須にするには

- IMDSv2 をアカウントのデフォルトとして設定する
- アカウントレベルで IMDSv2 を強制適用する
- 起動時にインスタンスを設定する
- AMI を設定する
- IAM ポリシーを使用する

IMDSv2 をアカウントのデフォルトとして設定する

インスタンスメタデータサービス (IMDS) のデフォルトバージョンは各 AWS リージョンのアカウントレベルで設定できます。つまり、新規インスタンスを起動すると、そのインスタンスメタデータバージョンは自動的にアカウントレベルのデフォルトに設定されます。ただし、起動時または起動後に値を手動で上書きできます。アカウントレベルの設定と手動オーバーライドがインスタンスに与える影響の詳細については「インスタンスメタデータオプションの優先順位」を参照してください。

> **Note**
>
> アカウントレベルのデフォルトを設定しても、既存のインスタンスはリセットされません。例えば、アカウントレベルのデフォルトを IMDSv2 に設定しても、IMDSv1 に設定されてい

る既存のインスタンスは影響を受けません。既存のインスタンスの値を変更する場合はインスタンス自体の値を手動で変更する必要があります。

インスタンスメタデータバージョンのアカウントのデフォルトを IMDSv2 に設定すると、アカウント内のすべての新しいインスタンスを IMDSv2 で起動できます。そうすると、IMDSv1 は無効になります。このアカウントデフォルトではインスタンスを起動すると、インスタンスのデフォルト値は次のようになります。

- コンソール: [メタデータのバージョン] は [V2 のみ (トークンは必須)] に設定され、[メタデータレスポンスのホップ制限] は [2] に設定されます。
- AWS CLI: `HttpTokens` は `required` に設定され、`HttpPutResponseHopLimit` は `2` に設定されます。

> **Note**
>
> アカウントのデフォルトを IMDSv2 に設定する前に、インスタンスが IMDSv1 に依存していないことを確認してください。詳細については「IMDSv2 を必要とする推奨パス」を参照してください。

Console

指定したリージョンのアカウントのデフォルトとして IMDSv2 を設定するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. AWS リージョン を変更するにはページの右上隅にあるリージョンセレクターを使用します。
3. ナビゲーションペインで、ダッシュボードを選択してください。
4. [アカウント属性] カードの [設定] で、[データ保護とセキュリティ] を選択します。
5. [IMDS のデフォルト] の横にある [管理] を選択してください。
6. [IMDS のデフォルトを管理] ページで、次の操作を実行します。
   a. [インスタンスメタデータサービス] で、[有効にする] を選択してください。
   b. [Metadata version] (メタデータバージョン) には[V2 only (token required)] (V2 のみ (トークンが必要)) を選択してください。

c. インスタンスがコンテナをホストする場合は[メタデータレスポンスのホップ制限] で 2 を指定します。それ以外の場合は[設定なし] を選択してください。設定なしが指定されているとき、AMI に ImdsSupport: v2.0 の設定がある場合は起動時の値がデフォルトで 2 になります。それ以外の場合はデフォルトで 1 になります。

d. [Update] (更新) を選択してください。

AWS CLI

指定したリージョンのアカウントのデフォルトとして IMDSv2 を設定するには

modify-instance-metadata-defaults コマンドを使用して、IMDS アカウントレベルの設定を変更するリージョンを指定します。インスタンスがコンテナをホストする場合は--http-tokens を required に、--http-put-response-hop-limit を 2 に設定します。それ以外の場合は-1 を指定して、設定がないことを示します。-1 (設定なし) が指定されているとき、AMI に ImdsSupport: v2.0 の設定がある場合は起動時の値がデフォルトで 2 になります。それ以外の場合はデフォルトで 1 になります。

```
aws ec2 modify-instance-metadata-defaults \
    --region us-east-1 \
    --http-tokens required \
    --http-put-response-hop-limit 2
```

出力例を次に示します。

```
{
    "Return": true
}
```

指定したリージョンのインスタンスメタデータオプションのデフォルトのアカウント設定を表示するには

get-instance-metadata-defaults コマンドを使用して、リージョンを指定します。

```
aws ec2 get-instance-metadata-defaults --region us-east-1
```

出力例を次に示します。

```
{
    "AccountLevel": {
```

```
        "HttpTokens": "required",
        "HttpPutResponseHopLimit": 2
    },
    "ManagedBy": "account"
}
```

ManagedBy フィールドは設定を設定したエンティティを示します。この例では accountは設定がアカウントで直接設定されたことを示します。declarative-policyという値は宣言的ポリシーによって設定が構成されたことを意味します。詳細については「AWS OrganizationsIAM ユーザーガイド」の「 マネージドポリシー」を参照してください。

すべてのリージョンのアカウントのデフォルトとして IMDSv2 を設定するには

すべてのリージョンの IMDS アカウントレベル設定を変更するにはmodify-instance-metadata-defaults コマンドを使用します。インスタンスがコンテナをホストする場合は--http-tokens を required に、--http-put-response-hop-limit を 2 に設定します。それ以外の場合は-1 を指定して、設定がないことを示します。-1 (設定なし) が指定されているとき、AMI に ImdsSupport: v2.0 の設定がある場合は起動時の値がデフォルトで 2 になります。それ以外の場合はデフォルトで 1 になります。

```
echo -e "Region          \t Modified" ; \
echo -e "--------------  \t ---------" ; \
for region in $(
    aws ec2 describe-regions \
        --region us-east-1 \
        --query "Regions[*].[RegionName]" \
        --output text
    );
    do (output=$(
        aws ec2 modify-instance-metadata-defaults \
            --region $region \
            --http-tokens required \
            --http-put-response-hop-limit 2 \
            --output text)
        echo -e "$region        \t $output"
    );
done
```

出力例を次に示します。

```
Region                  Modified
```

```
--------------          ---------
ap-south-1              True
eu-north-1              True
eu-west-3               True
...
```

すべてのリージョンのインスタンスメタデータオプションのデフォルトアカウント設定を表示するには

get-instance-metadata-defaults コマンドを使用します。

```
echo -e "Region   \t Level          Hops    HttpTokens" ; \
echo -e "-------------- \t ------------   ----    ----------" ; \
for region in $(
    aws ec2 describe-regions \
        --region us-east-1 \
        --query "Regions[*].[RegionName]" \
        --output text
    );
    do (output=$(
        aws ec2 get-instance-metadata-defaults \
            --region $region \
            --output text)
        echo -e "$region \t $output"
    );
done
```

出力例を次に示します。

```
Region          Level          Hops    HttpTokens
--------------  ------------   ----    ----------
ap-south-1      ACCOUNTLEVEL   2       required
eu-north-1      ACCOUNTLEVEL   2       required
eu-west-3       ACCOUNTLEVEL   2       required
...
```

PowerShell

指定したリージョンのアカウントのデフォルトとして IMDSv2 を設定するには

Edit-EC2InstanceMetadataDefault コマンドレットを使用して、IMDS アカウントレベルの設定を変更するリージョンを指定します。インスタンスがコンテナをホストする場合は-HttpToken

を required に、-HttpPutResponseHopLimit を 2 に設定します。それ以外の場合は-1 を指定して、設定がないことを示します。-1 (設定なし) が指定されているとき、AMI に ImdsSupport: v2.0 の設定がある場合は起動時の値がデフォルトで 2 になります。それ以外の場合はデフォルトで 1 になります。

```
Edit-EC2InstanceMetadataDefault `
    -Region us-east-1 `
    -HttpToken required `
    -HttpPutResponseHopLimit 2
```

出力例を次に示します。

```
True
```

指定したリージョンのインスタンスメタデータオプションのデフォルトのアカウント設定を表示するには

Get-EC2InstanceMetadataDefault コマンドレットを使用して、リージョンを指定します。

```
Get-EC2InstanceMetadataDefault -Region us-east-1 | Format-List
```

出力例を次に示します。

```
HttpEndpoint            :
HttpPutResponseHopLimit : 2
HttpTokens              : required
InstanceMetadataTags    :
```

すべてのリージョンのアカウントのデフォルトとして IMDSv2 を設定するには

Edit-EC2InstanceMetadataDefault コマンドレットを使用して、すべてのリージョンの IMDS アカウントレベル設定を変更します。インスタンスがコンテナをホストする場合は-HttpToken を required に、-HttpPutResponseHopLimit を 2 に設定します。それ以外の場合は-1 を指定して、設定がないことを示します。-1 (設定なし) が指定されているとき、AMI に ImdsSupport: v2.0 の設定がある場合は起動時の値がデフォルトで 2 になります。それ以外の場合はデフォルトで 1 になります。

```
(Get-EC2Region).RegionName | `
```

```
    ForEach-Object {
    [PSCustomObject]@{
        Region   = $_
        Modified = (Edit-EC2InstanceMetadataDefault `
                -Region $_ `
                -HttpToken required `
                -HttpPutResponseHopLimit 2)
    }
} | `
Format-Table Region, Modified -AutoSize
```

正常な出力

```
Region         Modified
------         --------
ap-south-1         True
eu-north-1         True
eu-west-3          True
...
```

すべてのリージョンのインスタンスメタデータオプションのデフォルトアカウント設定を表示するには

Get-EC2InstanceMetadataDefault コマンドレットを使用します。

```
(Get-EC2Region).RegionName | `
    ForEach-Object {
    [PSCustomObject]@{
        Region = $_
        HttpPutResponseHopLimit = (Get-EC2InstanceMetadataDefault -Region
 $_).HttpPutResponseHopLimit
        HttpTokens              = (Get-EC2InstanceMetadataDefault -Region
 $_).HttpTokens
    }
} | `
Format-Table -AutoSize
```

出力の例

```
Region         HttpPutResponseHopLimit HttpTokens
------         ----------------------- ----------
```

```
ap-south-1                          2 required
eu-north-1                          2 required
eu-west-3                           2 required
...
```

アカウントレベルで IMDSv2 を強制適用する

各 AWS リージョンに対し、アカウントレベルで IMDSv2 の使用を強制できます。使用が強制されると、インスタンスを起動できるのは IMDSv2 が必須として設定されている場合のみになります。この強制は、インスタンスまたは AMI の設定方法を問わずに適用されます。

> **Note**
>
> アカウントレベルで IMDSv2 の強制適用を有効にする前に、アプリケーションと AMI が IMDSv2 をサポートしていることを確認してください。詳細については「IMDSv2 を必要とする推奨パス」を参照してください。IMDSv2 の強制適用が有効になっており、`httpTokens` が起動時のインスタンス設定、アカウント設定、または AMI 構成のいずれかで `required` に設定されていない場合、インスタンスの起動は失敗します。トラブルシューティング情報については、「IMDSv1 が有効化されたインスタンスの起動が失敗する」を参照してください。

> **Note**
>
> この設定は既存のインスタンスの IMDS バージョンを変更しませんが、IMDSv1 が現在無効になっている既存のインスタンスでの IMDSv1 の有効化を阻止します。

Console

指定されたリージョン内のアカウントに IMDSv2 を強制適用する

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. AWS リージョンを変更するには、ページの右上隅にあるリージョンセレクターを使用します。
3. ナビゲーションペインで、ダッシュボードを選択してください。
4. [アカウント属性] カードの [設定] で、[データ保護とセキュリティ] を選択します。

5. [IMDS のデフォルト] の横にある [管理] を選択してください。
6. [IMDS のデフォルトを管理] ページで、次の操作を実行します。
   a. [Metadata version] (メタデータバージョン) には[V2 only (token required)] (V2 のみ (トークンが必要)) を選択してください。
   b. [IMDSv2 を強制適用] で [有効] を選択します。
   c. [Update] (更新) を選択してください。

AWS CLI

指定されたリージョン内のアカウントに IMDSv2 を強制適用する

modify-instance-metadata-defaults コマンドを使用して、IMDSv2 を強制適用するリージョンを指定します。

```
aws ec2 modify-instance-metadata-defaults \
    --region us-east-1 \
    --http-tokens required \
    --http-tokens-enforced enabled
```

出力例を次に示します。

```
{
"Return": true
}
```

特定のリージョン内のアカウントに対する IMDSv2 の強制適用設定を表示する

get-instance-metadata-defaults コマンドを使用して、リージョンを指定します。

```
aws ec2 get-instance-metadata-defaults --region us-east-1
```

出力例を次に示します。

```
{
    "AccountLevel": {
        "HttpTokens": "required",
        "HttpTokensEnforced": "enabled"
```

```
    },
    "ManagedBy": "account"
}
```

ManagedBy フィールドは設定を設定したエンティティを示します。この例では accountは設定がアカウントで直接設定されたことを示します。declarative-policyという値は宣言的ポリシーによって設定が構成されたことを意味します。詳細については、「AWS Organizations ユーザーガイド」の「宣言型ポリシー」を参照してください。

すべてのリージョンのアカウントに IMDSv2 を強制適用する

modify-instance-metadata-defaults コマンドを使用して、すべてのリージョンで IMDSv2 を強制適用します。

```
echo -e "Region          \t Modified" ; \
echo -e "--------------  \t ---------" ; \
for region in $(
    aws ec2 describe-regions \
        --region us-east-1 \
        --query "Regions[*].[RegionName]" \
        --output text
    );
    do (output=$(
        aws ec2 modify-instance-metadata-defaults \
            --region $region \
            --http-tokens-enforced enabled \
            --output text)
        echo -e "$region        \t $output"
    );
done
```

出力例を次に示します。

```
Region                  Modified
--------------          ---------
ap-south-1              True
eu-north-1              True
eu-west-3               True
...
```

すべてのリージョン内のアカウントに対する IMDSv2 の強制適用設定を表示する

get-instance-metadata-defaults コマンドを使用します。

```
echo -e "Region   \t Level           HttpTokensEnforced" ; \
echo -e "-------------- \t ------------   ----------------" ; \
for region in $(
    aws ec2 describe-regions \
        --region us-east-1 \
        --query "Regions[*].[RegionName]" \
        --output text
    );
    do (output=$(
        aws ec2 get-instance-metadata-defaults \
            --region $region \
            --query 'AccountLevel.HttpTokensEnforced' \
            --output text)
        echo -e "$region \t ACCOUNTLEVEL $output"
    );
done
```

出力例を次に示します。

```
Region           Level          HttpTokensEnforced
--------------   ------------   ------------------
ap-south-1       ACCOUNTLEVEL   enabled
eu-north-1       ACCOUNTLEVEL   enabled
eu-west-3        ACCOUNTLEVEL   enabled
...
```

PowerShell

指定されたリージョン内のアカウントに IMDSv2 を強制適用する

Edit-EC2InstanceMetadataDefault コマンドレットを使用して、IMDSv2 を強制適用するリージョンを指定します。

```
Edit-EC2InstanceMetadataDefault `
    -Region us-east-1 `
    -HttpToken required `
    -HttpPutResponseHopLimit 2
```

出力例を次に示します。

```
@{
    Return = $true
}
```

特定のリージョン内のアカウントに対する IMDSv2 の強制適用設定を表示する

Get-EC2InstanceMetadataDefault コマンドを使用して、リージョンを指定します。

```
Get-EC2InstanceMetadataDefault -Region us-east-1
```

出力例を次に示します。

```
@{
    AccountLevel = @{
        HttpTokens = "required"
        HttpTokensEnforced = "enabled"
    }
    ManagedBy = "account"
}
```

ManagedBy フィールドは設定を設定したエンティティを示します。この例では accountは設定がアカウントで直接設定されたことを示します。declarative-policyという値は宣言的ポリシーによって設定が構成されたことを意味します。詳細については、「AWS Organizations ユーザーガイド」の「宣言型ポリシー」を参照してください。

すべてのリージョンのアカウントに IMDSv2 を強制適用する

modify-instance-metadata-defaults コマンドを使用して、すべてのリージョンで IMDSv2 を強制適用します。

```
echo -e "Region          \t Modified" ; \
echo -e "--------------  \t ---------" ; \
for region in $(
    aws ec2 describe-regions \
        --region us-east-1 \
        --query "Regions[*].[RegionName]" \
        --output text
    );
    do (output=$(
        aws ec2 modify-instance-metadata-defaults \
```

```
            --region $region \
            --http-tokens-enforced enabled \
            --output text)
        echo -e "$region        \t $output"
    );
done
```

出力例を次に示します。

```
Region                  Modified
--------------          ---------
ap-south-1              True
eu-north-1              True
eu-west-3               True
...
```

すべてのリージョンのアカウントのデフォルトとして IMDSv2 を設定するには

Edit-EC2InstanceMetadataDefault コマンドレットを使用して、すべてのリージョンの IMDS アカウントレベル設定を変更します。インスタンスがコンテナをホストする場合は-HttpToken を required に、-HttpPutResponseHopLimit を 2 に設定します。それ以外の場合は-1 を指定して、設定がないことを示します。-1 (設定なし) が指定されているとき、AMI に ImdsSupport: v2.0 の設定がある場合は起動時の値がデフォルトで 2 になります。それ以外の場合はデフォルトで 1 になります。

```
(Get-EC2Region).RegionName | `
    ForEach-Object {
    [PSCustomObject]@{
        Region   = $_
        Modified = (Edit-EC2InstanceMetadataDefault `
                -Region $_ `
                -HttpToken required `
                -HttpPutResponseHopLimit 2)
    }
} | `
Format-Table Region, Modified -AutoSize
```

正常な出力

```
Region         Modified
```

```
------          --------
ap-south-1          True
eu-north-1          True
eu-west-3           True
...
```

すべてのリージョンのインスタンスメタデータオプションのデフォルトアカウント設定を表示するには

Get-EC2InstanceMetadataDefault コマンドレットを使用します。

```
(Get-EC2Region).RegionName | `
    ForEach-Object {
    [PSCustomObject]@{
        Region = $_
        HttpPutResponseHopLimit = (Get-EC2InstanceMetadataDefault -Region
 $_).HttpPutResponseHopLimit
        HttpTokens              = (Get-EC2InstanceMetadataDefault -Region
 $_).HttpTokens
    }
} | `
Format-Table -AutoSize
```

出力の例

```
Region         HttpPutResponseHopLimit HttpTokens
------         ----------------------- ----------
ap-south-1                           2 required
eu-north-1                           2 required
eu-west-3                            2 required
...
```

起動時にインスタンスを設定する

インスタンスを起動する際に、以下のフィールドを設定しておくことで、IMDSv2 が使用されるようにそのインスタンスを構成できます。

- Amazon EC2 コンソール: [Metadata version] (メタデータバージョン) で、[V2 only (token required)] (V2 のみ (トークンが必須)) を設定します。
- AWS CLI: `HttpTokens` に `required` を設定します。

IMDSv2 が必須であることを指定する場合、[メタデータにアクセス可能] に [有効] (コンソールの場合) を設定するか、HttpEndpoint に enabled (AWS CLI の場合) を設定して、インスタンスメタデータサービス (IMDS) のエンドポイントも有効にする必要があります。

コンテナ環境ではIMDSv2 が要求されている場合、ホップ制限を 2 に設定することをお勧めします。詳細については「インスタンスメタデータアクセス考慮事項」を参照してください。

Console

新しいインスタンスで IMDSv2 の使用を要求するには

- Amazon EC2 コンソールで新しいインスタンスを起動するとき、[Advanced details] (高度な詳細) を展開し、次の操作を行います。
  - [Metadata accessible] (メタデータにアクセス可能) には[Enabled] (有効) を選択してください。
  - [Metadata version] (メタデータバージョン) には[V2 only (token required)] (V2 のみ (トークンが必要)) を選択してください。
  - (コンテナ環境) [メタデータレスポンスのホップ制限] で、2 を選択してください。

  詳細については「高度な詳細」を参照してください。

AWS CLI

新しいインスタンスで IMDSv2 の使用を要求するには

次の run-instances の例ではc6i.large を --metadata-options に設定して HttpTokens=required インスタンスを起動します。HttpTokens の値を指定する場合はHttpEndpoint も enabled に設定する必要があります。メタデータの取得リクエストではセキュリティで保護されたトークンヘッダーは required に設定されるので、インスタンスメタデータのリクエストに際してはそのインスタンスは必ず IMDSv2 を使用することになります。

コンテナ環境ではIMDSv2 が要求されている場合、HttpPutResponseHopLimit=2 を使用してホップ制限を 2 に設定することをお勧めします。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type c6i.large \
 ...
```

```
    --metadata-options
"HttpEndpoint=enabled,HttpTokens=required,HttpPutResponseHopLimit=2"
```

PowerShell

新しいインスタンスで IMDSv2 の使用を要求するには

次の New-EC2Instance コマンドレット例は、MetadataOptions_HttpEndpoint が enabled に、MetadataOptions_HttpTokens パラメータが required に設定された c6i.large インスタンスを起動します。HttpTokens の値を指定する場合はHttpEndpoint も enabled に設定する必要があります。メタデータの取得リクエストではセキュリティで保護されたトークンヘッダーは required に設定されるので、インスタンスメタデータのリクエストに際してはそのインスタンスは必ず IMDSv2 を使用することになります。

```
New-EC2Instance `
    -ImageId ami-0abcdef1234567890 `
    -InstanceType c6i.large `
    -MetadataOptions_HttpEndpoint enabled `
    -MetadataOptions_HttpTokens required
```

CloudFormation

CloudFormation を使用してインスタンスのメタデータオプションを指定するには「AWS CloudFormation ユーザーガイド」の「AWS::EC2::LaunchTemplate MetadataOptions」プロパティを参照してください。

AMI を設定する

新しい AMI を登録したり、既存の AMI を変更したりするときに、imds-support パラメータを v2.0 に設定できます。この AMI から起動されたインスタンスでは、[メタデータのバージョン] が [V2 のみ (トークンは必須)] (コンソールの場合) に設定されるか、HttpTokens が required (AWS CLI の場合) に設定されます。この設定が行われている場合、インスタンスメタデータがリクエストされる際には IMDSv2 を使用することが、インスタンスでの必須になります。

この AMI から起動されるインスタンスではimds-support に v2.0 を設定している場合、[Metadata response hop limit] (メタデータレスポンスのホップ制限) (コンソールの場合)、または http-put-response-hop-limit (AWS CLI の場合) が「2」に設定されることに注意してください。

> **Important**
>
> ご使用の AMI ソフトウェアが IMDSv2 をサポートしていない限りはこのパラメータを使用しないでください。値を v2.0 に設定すると、元に戻すことはできません。AMI を「リセット」する唯一の方法は基礎となるスナップショットから新しい AMI を作成することです。

IMDSv2 向けに AMI を新たに設定するには

IMDSv2 に新しい AMI を設定するには次のいずれかの方法を使用します。

AWS CLI

以下の register-image の例ではEBS ルートボリュームの指定されたスナップショットをデバイス /dev/xvda として使用して、AMI を登録しています。imds-support パラメータ用に v2.0 を指定し、この AMI から起動するインスタンスに対して、インスタンスメタデータのリクエスト時に IMDSv2 を使用することが、この AMI から起動されるインスタンスでの必須になります。

```
aws ec2 register-image \
    --name my-image \
    --root-device-name /dev/xvda \
    --block-device-mappings DeviceName=/dev/
xvda,Ebs={SnapshotId=snap-0123456789example} \
    --architecture x86_64 \
    --imds-support v2.0
```

PowerShell

次の Register-EC2Image コマンドレット例は、EBS ルートボリュームの指定されたスナップショットをデバイス /dev/xvda として使用して、AMI を登録します。ImdsSupport パラメータ用に v2.0 を指定し、この AMI から起動するインスタンスに対して、インスタンスメタデータのリクエスト時に IMDSv2 を使用することが、この AMI から起動されるインスタンスでの必須になります。

```
Register-EC2Image `
    -Name 'my-image' `
    -RootDeviceName /dev/xvda `
    -BlockDeviceMapping  (
    New-Object `
        -TypeName Amazon.EC2.Model.BlockDeviceMapping `
```

```
        -Property @{
        DeviceName = '/dev/xvda';
        EBS        = (New-Object -TypeName Amazon.EC2.Model.EbsBlockDevice -Property
@{
                SnapshotId = 'snap-0123456789example'
                VolumeType = 'gp3'
                } )
        }  ) `
    -Architecture X86_64 `
    -ImdsSupport v2.0
```

IMDSv2 向けに既存の AMI を設定するには

IMDSv2 向けに既存の AMI を設定するには次のいずれかの方法を使用します。

AWS CLI

次の modify-image-attribute の例ではIMDSv2 用の既存の AMI のみを変更します。imds-support パラメータ用に v2.0 を指定し、この AMI から起動するインスタンスに対して、インスタンスメタデータのリクエスト時に IMDSv2 を使用することが、この AMI から起動されるインスタンスでの必須になります。

```
aws ec2 modify-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --imds-support v2.0
```

PowerShell

次の Edit-EC2ImageAttribute コマンドレット例は、IMDSv2 用の既存の AMI のみを変更します。imds-support パラメータ用に v2.0 を指定し、この AMI から起動するインスタンスに対して、インスタンスメタデータのリクエスト時に IMDSv2 を使用することが、この AMI から起動されるインスタンスでの必須になります。

```
Edit-EC2ImageAttribute `
    -ImageId ami-0abcdef1234567890 `
    -ImdsSupport 'v2.0'
```

IAM ポリシーを使用する

次のいずれかを実行する IAM ポリシーを作成できます。

- 新しいインスタンスで IMDSv2 を必須と設定する場合を除き、ユーザーが新しいインスタンスを起動することを防ぐ。
- ユーザーが ModifyInstanceMetadataOptions API をユ呼び出して実行中のインスタンスのメタデータオプションを変更することを防ぐ。ModifyInstanceMetadataOptions httpTokens プロパティへのアクセスを制限して、実行中のインスタンスの意図しない更新が行われないようにする。
- ユーザーが ModifyInstanceMetadataDefaults API を呼び出して httpTokens と httpTokensEnforced 両方のアカウントデフォルト設定を変更することを防ぐ。これらの 2 つのプロパティへのアクセスを制限することで、承認済みのロール以外はアカウントのデフォルトを変更できないようにします。

IAM ポリシーにより、すべての新しいインスタンスでの IMDSv2 の使用を必須にするには

ユーザーが起動できるインスタンスを、メタデータのリクエスト時に IMDSv2 の使用を必須としているインスタンスに限定するには、次の手順を実行します。

- `ModifyInstanceMetadataOptions` と `ModifyInstanceMetadataDefaults` API の両方、より具体的には `httpTokens` と `httpTokensEnforced` プロパティへのアクセスを制限します。
- 次に、アカウントのデフォルトを `httpTokens = required` と `httpTokensEnforced = enabled` に設定します。

  IAM ポリシーの例についてはインスタンスメタデータの使用を参照してください。

IMDS IPv4 および IPv6 エンドポイントを有効にする

IMDS のインスタンスにはIPv4 (`169.254.169.254`) と IPv6 (`[fd00:ec2::254]`) という 2 つのエンドポイントがあります。IMDS を有効にすると、IPv4 エンドポイントが自動的に有効になります。IPv6 専用サブネットに対してインスタンスを起動しても、その IPv6 エンドポイントは無効のままになります。IPv6 エンドポイントを有効にするには明示的に有効にする必要があります。IPv6 エンドポイントを有効にしても、IPv4 エンドポイントは有効なままになります。

IPv6 エンドポイントはインスタンス起動時またはその後に有効にできます。

IPv6 エンドポイントを有効にするための要件

- 選択されているインスタンスタイプは Nitro ベースのインスタンスです。
- 選択したサブネットはそのサブネットがデュアルスタックまたは IPv6 専用である場合、IPv6 をサポートします。

IMDS IPv6 エンドポイント対応のインスタンスを起動するには以下のいずれかの方法を使用します。

Console

インスタンス起動時に IMDS IPv6 エンドポイントを有効にするには

- [Advanced details] (高度な詳細) で以下のように指定して、Amazon EC2 コンソールでインスタンスを起動します。
  - メタデータ IPv6 エンドポイント で、[有効] を選択します。

詳細については「高度な詳細」を参照してください。

AWS CLI

インスタンス起動時に IMDS IPv6 エンドポイントを有効にするには

以下の run-instances の例ではIMDS 用に IPv6 エンドポイントが有効化された、c6i.large インスタンスを起動しています。IPv6 エンドポイントを有効にするには--metadata-options パラメータに HttpProtocolIpv6=enabled を指定します。HttpProtocolIpv6 の値を指定する場合はHttpEndpoint も enabled に設定する必要があります。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type c6i.large \
    ...
    --metadata-options "HttpEndpoint=enabled,HttpProtocolIpv6=enabled"
```

PowerShell

インスタンス起動時に IMDS IPv6 エンドポイントを有効にするには

次の New-EC2Instance コマンドレット例は、IMDS 用に IPv6 エンドポイントが有効化された c6i.large インスタンスを起動します。IPv6 エンドポイントを有効にするにはMetadataOptions_HttpProtocolIpv6 を enabled に指定します。MetadataOptions_HttpProtocolIpv6 の値を指定する場合はMetadataOptions_HttpEndpoint も enabled に設定する必要があります。

```
New-EC2Instance `
    -ImageId ami-0abcdef1234567890 `
    -InstanceType c6i.large `
```

```
    -MetadataOptions_HttpEndpoint enabled `
    -MetadataOptions_HttpProtocolIpv6 enabled
```

インスタンスメタデータへのアクセスを無効にする

インスタンスを起動するときに IMDS を無効にすることで、インスタンスのメタデータへのアクセスを無効にできます。IMDS を再度有効にすると、その後でアクセスを有効にできます。詳細については「インスタンスメタデータへのアクセスを有効にする」を参照してください。

> **Important**
>
> IMDS は起動時または起動後に無効化できます。起動時に IMDS を無効にすると、以下が機能しなくなる可能性があります。
>
> - インスタンスへの SSH アクセスがない可能性があります。キーは通常 EC2 インスタンスのメタデータから提供され、アクセスされるため、インスタンスのパブリック SSH キーである `public-keys/0/openssh-key` にはアクセスできません。
> - EC2 ユーザーデータは利用できず、インスタンスの起動時には実行されません。EC2 ユーザーデータは IMDS でホストされます。IMDS を無効にすると、ユーザーデータへのアクセスは事実上無効になります。
>
> この機能にアクセスするには起動後に IMDS を再度有効にします。

Console

起動時にインスタンスメタデータへのアクセスを無効にするには

- [Advanced details] (高度な詳細) で以下のように指定して、Amazon EC2 コンソールでインスタンスを起動します。
  - [Metadata accessible] (メタデータにアクセス可能) には[Disabled] (無効) を選択してください。

詳細については「高度な詳細」を参照してください。

AWS CLI

起動時にインスタンスメタデータへのアクセスを無効にするには

--metadata-options に HttpEndpoint=disabled を設定し、インスタンスを起動します。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type c6i.large \
    ...
    --metadata-options "HttpEndpoint=disabled"
```

PowerShell

起動時にインスタンスメタデータへのアクセスを無効にするには

次の New-EC2Instance コマンドレット例は、MetadataOptions_HttpEndpoint が disabled に設定されたインスタンスを起動します。

```
New-EC2Instance `
    -ImageId ami-0abcdef1234567890 `
    -InstanceType c6i.large `
    -MetadataOptions_HttpEndpoint disabled
```

CloudFormation

CloudFormation を使用してインスタンスのメタデータオプションを指定するには「CloudFormation ユーザーガイド」の「AWS::EC2::LaunchTemplate MetadataOptions」プロパティを参照してください。

インスタンスメタデータのタグへのアクセスを許可する

デフォルトではインスタンスメタデータ内のインスタンスタグへはアクセスできません。インスタンスごとに、アクセスを明示的に許可する必要があります。アクセスが許可されている場合、インスタンスタグキーは特定の文字制限に準拠している必要があります。それ以外の場合はインスタンスの起動に失敗します。詳細については「インスタンスメタデータ内のタグへのアクセスを有効にする」を参照してください。

既存インスタンスのインスタンスメタデータオプションの変更

既存のインスタンスのインスタンスメタデータオプションを変更することが可能です。

また、既存のインスタンスでインスタンスメタデータオプションを変更することをユーザーに禁止する IAM ポリシーを作成することもできます。インスタンスメタデータオプションを変更で

きるユーザーをコントロールするには指定したロールを持つユーザー以外のすべてのユーザーに ModifyInstanceMetadataOptions API の使用を禁止するポリシーを指定できます。IAM ポリシーの例については インスタンスメタデータの使用 を参照してください。

> **Note**
>
> 宣言ポリシーを使用してインスタンスメタデータオプションを設定した場合、アカウント内で直接変更することはできません。詳細については「AWS OrganizationsIAM ユーザーガイド」の「 マネージドポリシー」を参照してください。

IMDSv2 の使用を要求する

既存のインスタンスに対して、インスタンスメタデータのリクエスト時に IMDSv2 が使用されるようにするため、既存のインスタンスメタデータオプションを変更します。IMDSv2 が必須である場合、IMDSv1 は使用できません。

> **Note**
>
> IMDSv2 の使用を要求する前に、インスタンスが IMDSv1 呼び出しを行っていないことを確認してください。`MetadataNoToken` CloudWatch メトリクスは IMDSv1 呼び出しを追跡します。あるインスタンスの `MetadataNoToken` で IMDSv1 の使用量がゼロと記録されている場合はそのインスタンスは IMDSv2 を要求できます。

Console

既存インスタンスでの IMDSv2 の使用を義務付けるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択してください。
3. インスタンスを選択してください。
4. [アクション]、[インスタンスの設定]、[インスタンスメタデータのオプションを変更] の順に選択してください。
5. [インスタンスメタデータオプションの変更] ダイアログボックスで、次の操作を行います。
   a. [インスタンスメタデータサービス] で、[有効にする] を選択してください。
   b. [IMDSv2] の場合は[必須] を選択してください。

c. [Save] を選択してください。

AWS CLI

既存インスタンスでの IMDSv2 の使用を義務付けるには

modify-instance-metadata-options CLI コマンドを使って、http-tokens パラメータを required に設定できます。http-tokens の値を指定する場合はhttp-endpoint も enabled に設定する必要があります。

```
aws ec2 modify-instance-metadata-options \
    --instance-id i-1234567890abcdef0 \
    --http-tokens required \
    --http-endpoint enabled
```

PowerShell

既存インスタンスでの IMDSv2 の使用を義務付けるには

Edit-EC2InstanceMetadataOption コマンドレットを使用して、HttpTokens パラメータを required に設定します。HttpTokens の値を指定する場合はHttpEndpoint も enabled に設定する必要があります。

```
(Edit-EC2InstanceMetadataOption `
    -InstanceId i-1234567890abcdef0 `
    -HttpTokens required `
    -HttpEndpoint enabled).InstanceMetadataOptions
```

IMDSv1 の使用を再開します

インスタンスで IMDSv2 が必須となっている場合、IMDSv1 リクエストの使用は失敗します。IMDSv2 がオプションである場合、IMDSv2 と IMDSv1 の両方が機能します。このため、IMDSv1 を復元するには、次のいずれかの方法を使用して IMDSv2 をオプション (httpTokens = optional) として設定します。

httpTokensEnforced IMDS プロパティも、既存のインスタンスで IMDSv1 を有効化する試みを阻止します。リージョン内のアカウントに対して有効化されているときに httpTokens を optional に設定しようとすると、UnsupportedOperation 例外が発生します。詳細については、「トラブルシューティング」を参照してください。

> **Important**
>
> IMDSv2 の強制適用が原因でインスタンスの起動が失敗する場合は、起動を正常に行うための 2 つのオプションがあります。
>
> - インスタンスを IMDSv2 のみとして起動する – インスタンスで実行されているソフトウェアが IMDSv2 のみを使用している (IMDSv1 に対する依存なし) 場合、インスタンスを IMDSv2 のみとして起動できます。これを実行するには、リージョン内のアカウントの起動パラメータまたはメタデータデフォルトのいずれかで `httpTokens = required` を設定することによって IMDSv2 のみに設定します。
> - 強制適用を無効にする – ソフトウェアが依然として IMDSv1 に依存している場合は、リージョン内のアカウントで `httpTokensEnforced` を `disabled` に設定します。詳細については「アカウントレベルで IMDSv2 を強制適用する」を参照してください。

Console

インスタンスで IMDSv1 の使用を復元するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択してください。
3. インスタンスを選択してください。
4. [アクション]、[インスタンスの設定]、[インスタンスメタデータのオプションを変更] の順に選択してください。
5. [インスタンスメタデータオプションの変更] ダイアログボックスで、次の操作を行います。
   a. [インスタンスメタデータサービス] で、[有効にする] が選択されていることを確認します。
   b. [IMDSv2] の場合は[オプション] を選択してください。
   c. [Save] を選択してください。

AWS CLI

インスタンスで IMDSv1 の使用を復元するには

modify-instance-metadata-options CLI コマンドを、`http-tokens` を `optional` に設定して実行すると、インスタンスメタデータのリクエスト時に IMDSv1 の使用を復元できます。

```
aws ec2 modify-instance-metadata-options \
    --instance-id i-1234567890abcdef0 \
    --http-tokens optional \
    --http-endpoint enabled
```

PowerShell

インスタンスで IMDSv1 の使用を復元するには

`HttpTokens` を `optional` に設定した Edit-EC2InstanceMetadataOption コマンドレットを使用して、インスタンスメタデータのリクエスト時に IMDSv1 の使用を復元できます。

```
(Edit-EC2InstanceMetadataOption `
    -InstanceId i-1234567890abcdef0 `
    -HttpTokens optional `
    -HttpEndpoint enabled).InstanceMetadataOptions
```

PUT レスポンスホップリミットを変更する

既存インスタンスについて、PUTリスポンスホップリミットの設定を変更することができます。

現在、PUT 応答ホップ制限の変更をサポートしているのはAWS CLI と AWS SDK のみです。

AWS CLI

PUT レスポンスホップリミットを変更するには

modify-instance-metadata-options CLI コマンドを使って、`http-put-response-hop-limit` パラメータを必要なホップ数に設定できます。以下の例ではホップリミットが3に設定されています。`http-put-response-hop-limit` の値を指定する場合は`http-endpoint` を `enabled` に設定することも必要です。

```
aws ec2 modify-instance-metadata-options \
    --instance-id i-1234567890abcdef0 \
    --http-put-response-hop-limit 3 \
    --http-endpoint enabled
```

PowerShell

PUT レスポンスホップリミットを変更するには

Edit-EC2InstanceMetadataOption コマンドレットを使用して、HttpPutResponseHopLimit パラメータを必要なホップ数に設定します。以下の例ではホップリミットが3に設定されています。HttpPutResponseHopLimit の値を指定する場合はHttpEndpoint を enabled に設定することも必要です。

```
(Edit-EC2InstanceMetadataOption `
    -InstanceId i-1234567890abcdef0 `
    -HttpPutResponseHopLimit 3 `
    -HttpEndpoint enabled).InstanceMetadataOptions
```

IMDS IPv4 および IPv6 エンドポイントを有効にする

IMDS のインスタンスにはIPv4 (169.254.169.254) と IPv6 ([fd00:ec2::254]) という 2 つのエンドポイントがあります。IMDS を有効にすると、IPv4 エンドポイントが自動的に有効になります。IPv6 専用サブネットに対してインスタンスを起動しても、その IPv6 エンドポイントは無効のままになります。IPv6 エンドポイントを有効にするには明示的に有効にする必要があります。IPv6 エンドポイントを有効にしても、IPv4 エンドポイントは有効なままになります。

IPv6 エンドポイントはインスタンス起動時またはその後に有効にできます。

IPv6 エンドポイントを有効にするための要件

- 選択されているインスタンスタイプは Nitro ベースのインスタンスです。
- 選択したサブネットはそのサブネットがデュアルスタックまたは IPv6 専用である場合、IPv6 をサポートします。

現在、AWS CLI と AWS SDK のみがインスタンス起動後の IMDS IPv6 エンドポイントの有効化をサポートします。

AWS CLI

インスタンスで IMDS IPv6 エンドポイントを有効にするには

modify-instance-metadata-options CLI コマンドを使って、http-protocol-ipv6 パラメータを enabled に設定できます。http-protocol-ipv6 の値を指定する場合はhttp-endpoint を enabled に設定することも必要です。

```
aws ec2 modify-instance-metadata-options \
```

```
--instance-id i-1234567890abcdef0 \
--http-protocol-ipv6 enabled \
--http-endpoint enabled
```

PowerShell

インスタンスで IMDS IPv6 エンドポイントを有効にするには

Edit-EC2InstanceMetadataOption コマンドレットを使用して、HttpProtocolIpv6 パラメータを enabled に設定します。HttpProtocolIpv6 の値を指定する場合はHttpEndpoint を enabled に設定することも必要です。

```
(Edit-EC2InstanceMetadataOption `
    -InstanceId i-1234567890abcdef0 `
    -HttpProtocolIpv6 enabled `
    -HttpEndpoint enabled).InstanceMetadataOptions
```

インスタンスメタデータへのアクセスを有効にする

使用中の IMDS のバージョンに関係なく、IMDS の HTTP エンドポイントを無効にすることによりインスタンスメタデータへのアクセスをオフにすることができます。HTTP エンドポイントを無効化することにより、この変更はいつでも元に戻すことができます。

次のいずれかの方法を使用して、インスタンスのインスタンスメタデータへのアクセスを有効にします。

Console

インスタンスメタデータへのアクセスを有効にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択してください。
3. インスタンスを選択してください。
4. [アクション]、[インスタンスの設定]、[インスタンスメタデータのオプションを変更] の順に選択してください。
5. [インスタンスメタデータオプションの変更] ダイアログボックスで、次の操作を行います。

    a. [インスタンスメタデータサービス] で、[有効にする] を選択してください。

b. [Save] を選択してください。

AWS CLI

インスタンスメタデータへのアクセスを有効にするには

modify-instance-metadata-options CLI コマンドを使って、`http-endpoint` パラメータを `enabled` に設定できます。

```
aws ec2 modify-instance-metadata-options \
    --instance-id i-1234567890abcdef0 \
    --http-endpoint enabled
```

PowerShell

インスタンスメタデータへのアクセスを有効にするには

Edit-EC2InstanceMetadataOption コマンドレットを使用して、`HttpEndpoint` パラメータを `enabled` に設定します。

```
(Edit-EC2InstanceMetadataOption `
    -InstanceId i-1234567890abcdef0 `
    -HttpEndpoint enabled).InstanceMetadataOptions
```

インスタンスメタデータへのアクセスを無効にする

使用中のインスタンスメタデータサービスのバージョンに関係なく、IMDS の HTTP エンドポイントを無効にすることにより IMDS へのアクセスをオフにすることができます。HTTP エンドポイントを有効化することにより、この変更はいつでも元に戻すことができます。

インスタンスのインスタンスメタデータへのアクセスを無効にするには次のいずれかの方法を使用します。

Console

インスタンスメタデータへのアクセスを無効にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択してください。

3. インスタンスを選択してください。
4. [アクション]、[インスタンスの設定]、[インスタンスメタデータのオプションを変更] の順に選択してください。
5. [インスタンスメタデータオプションの変更] ダイアログボックスで、次の操作を行います。
   a. [インスタンスメタデータサービス] では[有効にする] をオフにします。
   b. [Save] を選択してください。

AWS CLI

インスタンスメタデータへのアクセスを無効にするには

modify-instance-metadata-options CLI コマンドを使って、`http-endpoint` パラメータを `disabled` に設定できます。

```
aws ec2 modify-instance-metadata-options \
    --instance-id i-1234567890abcdef0 \
    --http-endpoint disabled
```

PowerShell

インスタンスメタデータへのアクセスを無効にするには

Edit-EC2InstanceMetadataOption コマンドレットを使用して、`HttpEndpoint` パラメータを `disabled` に設定します。

```
(Edit-EC2InstanceMetadataOption `
    -InstanceId i-1234567890abcdef0 `
    -HttpEndpoint disabled).InstanceMetadataOptions
```

インスタンスメタデータのタグへのアクセスを許可する

実行中または停止中のインスタンス上のインスタンスメタデータ内のタグへのアクセスを許可できます。インスタンスごとに、アクセスを明示的に許可する必要があります。アクセスが許可されている場合、インスタンスタグキーは特定の文字制限に準拠している必要があります。それ以外の場合は工ラーが発生します。詳細については「インスタンスメタデータ内のタグへのアクセスを有効にする」を参照してください。

トラブルシューティング

IMDSv1 が有効化されたインスタンスの変更が失敗する

説明

以下のエラーメッセージが表示されます。

```
You can't launch instances with IMDSv1 because httpTokensEnforced
is enabled for this account. Either launch the instance with
httpTokens=required or contact your account owner to disable
httpTokensEnforced using the ModifyInstanceMetadataDefaults API or the
account settings in the EC2 console.
```

原因

このエラーは、EC2 アカウント設定または AWS Organization 宣言型ポリシーが IMDSv2 の使用を強制する (`httpTokensEnforced = enabled`) アカウントで既存のインスタンスを IMDSv1 有効 (`httpTokens = optional`) に変更しようとするときにスローされます。

ソリューション

既存のインスタンスで IMDSv1 サポートが必要な場合は、リージョン内のアカウントで IMDSv2 の強制適用を無効にする必要があります。IMDSv2 の強制適用を無効にするには、`HttpTokensEnforced` を `disabled`.に設定します。詳細については、「Amazon EC2 API リファレンス」の「ModifyInstanceMetadataDefaults」を参照してください。この設定をコンソールを使用して行う場合は、「アカウントレベルで IMDSv2 を強制適用する」を参照してください。

IMDSv2 のみ (`httpTokens=required`) を使用することをお勧めします。詳細については「インスタンスメタデータサービスバージョン 2 の使用への移行」を参照してください。

ユーザーデータ入力を使用して EC2 インスタンスを起動するときにコマンドを実行する

Amazon EC2 インスタンスの起動時に、インスタンスの起動後に自動設定タスクを実行したり、スクリプトを実行したりするのに使用するインスタンスにユーザーデータを渡すことができます。

より複雑なオートメーションのシナリオに興味がある場合、CloudFormation を検討してください。詳細については、「AWS CloudFormation ユーザーガイド」の「CloudFormation を使用して Amazon EC2 でアプリケーションのデプロイ」を参照してください。

Linux インスタンスでは、シェルスクリプトと cloud-init ディレクティブという 2 つのタイプのユーザーデータを Amazon EC2 に渡すことができます。また、このデータは、プレーンテキスト、ファイル (コマンドラインツールを使用してインスタンスを起動する場合に便利です)、または base64 でエンコードされたテキスト (API コール向け) で、インスタンス起動ウィザードに渡すこともできます。

Windows インスタンスでは、起動エージェントがユーザーデータスクリプトを処理します。

考慮事項

- ユーザーデータは非透過的なデータとして取り扱われ、指定したデータがそのまま返されます。インスタンスによって解釈が異なります。
- ユーザーデータはbase64 でエンコードされている必要があります。Amazon EC2 コンソールは base64 エンコードを実行したり、base64 エンコード入力を受け入れたりできます。インスタンスのメタデータまたはコンソールを使用してユーザーデータを取得する場合、自動的に base64 でデコードされます。
- ユーザーデータは raw 形式の 16 KB に制限されます (以前は base64 エンコード)。base64 エンコード後の 文字列の長さサイズ n はceil(n/3)*4 です。
- ユーザーデータはインスタンス属性です。インスタンスから AMI を作成する場合、インスタンスのユーザーデータは AMI に含まれません。

AWS マネジメントコンソール のユーザーデータ

インスタンスの起動時のインスタンスユーザーデータを指定できます。インスタンスのルートボリュームが EBS ボリュームの場合は、インスタンスを停止してユーザーデータを更新することもできます。

Launch Wizard を使用して起動時にインスタンスのユーザーデータを指定する

EC2 コンソールで Launch Wizard を使用してインスタンスを起動するときに、ユーザーデータを指定できます。起動時にユーザーデータを指定するには、インスタンスを起動する手順に従います。[User data] (ユーザーデータ) フィールドは、インスタンス起動ウィザードの 高度な詳細 セクションにあります。PowerShell スクリプトを [ユーザーデータ] フィールドに入力し、インスタンスの起動手順を完了します。

次の [ユーザーデータ] フィールドのスクリーンショットのスクリプト例では、ファイル名に現在の日付と時刻を使用して、Windows 一時フォルダーにファイルを作成します。<persist>true</

persist> を含めると、インスタンスを再起動または起動するたびにスクリプトが実行されます。[ユーザーデータは既に base64 でエンコードされています] チェックボックスをオフのままにした場合、Amazon EC2 コンソールがユーザーに代わって base64 エンコードを実行します。

詳細については、「Launch Wizard を使用して起動時にインスタンスのユーザーデータを指定する」を参照してください。AWS CLI を使用するの Linux の例については、「the section called “ユーザーデータと AWS CLI”」を参照してください。Windows PowerShell用ツール を使用する Windows の例については、「the section called “ユーザーデータと Windows PowerShell用ツール”」を参照してください。

インスタンスユーザーデータの表示と更新

任意のインスタンスのインスタンスユーザーデータを表示し、停止したインスタンスのインスタンスユーザーデータを更新できます。

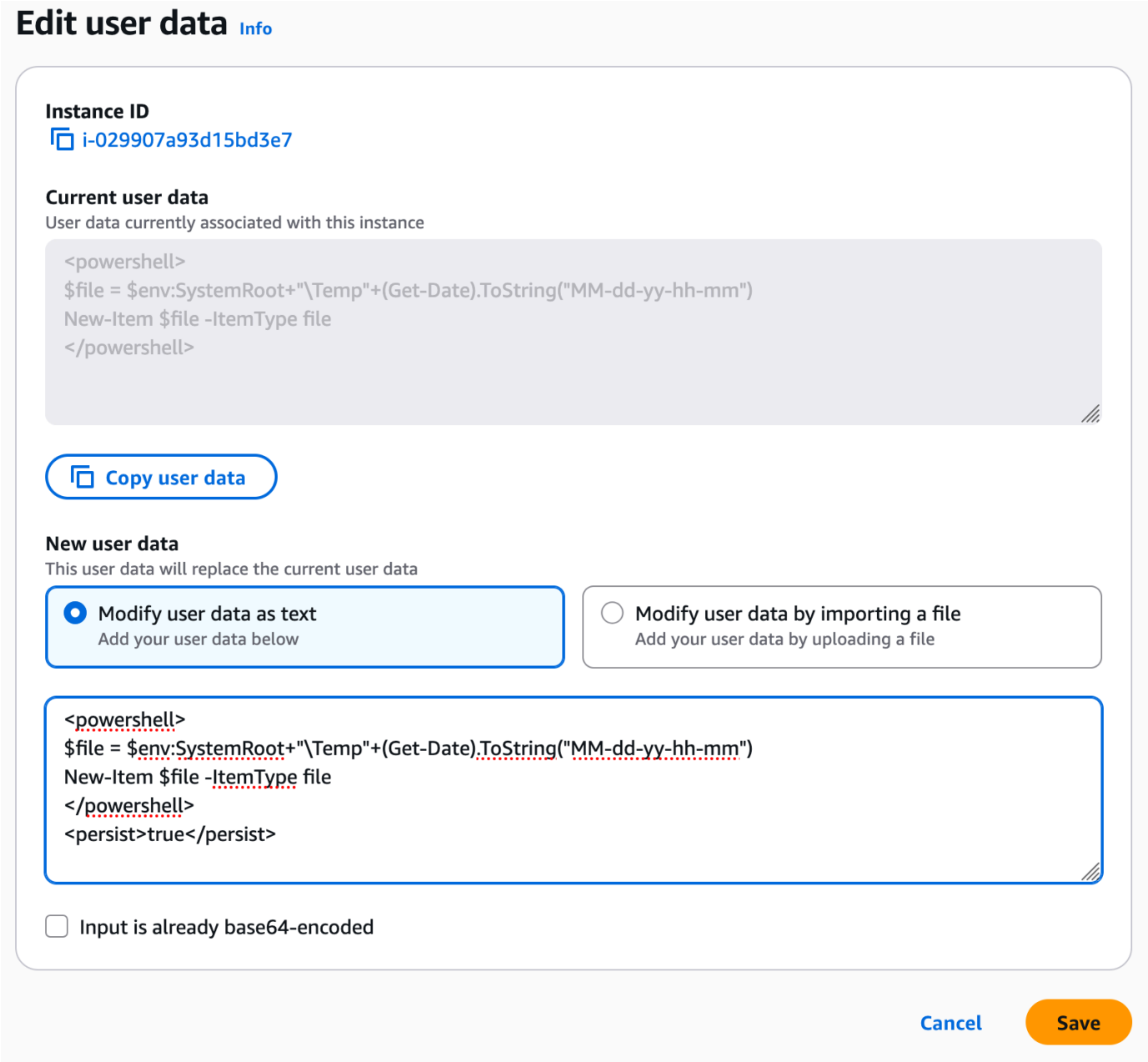
コンソールを使用してインスタンスユーザーデータを更新するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。

3. インスタンスを選択し、[Actions (アクション)]、[Instance state (インスタンスの状態)]、[Stop instance (インスタンスの停止)] の順に選択してください。

   > **Warning**
   >
   > インスタンスストアボリューム上のデータは、インスタンスを停止すると失われます。このデータを保持するには、永続ストレージにバックアップしてください。

4. 確認を求められたら、[Stop] を選択してください。インスタンスが停止するまで、数分かかる場合があります。
5. インスタンスが選択された状態のまま、[Actions (アクション)]、[Instance settings (インスタンス設定)]、[Edit user data (ユーザーデータの編集)] の順に選択してください。インスタンスの実行中はユーザーデータを変更できませんが、表示することはできます。
6. [Edit user data (ユーザーデータの編集)] ダイアログボックスで、ユーザーデータを更新し、[Save (保存)] を選択してください。インスタンスを再起動または起動するたびにユーザーデータスクリプトを実行するには、次の例に示すように `<persist>true</persist>` をユーザーデータに追加します。



7. インスタンスを起動します。後続の再起動または開始のためにユーザーデータの実行を有効にした場合、更新されたユーザーデータスクリプトはインスタンスの開始プロセスの一部として実行されます。

Amazon EC2 が Linux インスタンスのユーザーデータを処理する方法

次の例は、ユーザーデータを使用して、インスタンスの起動時に LAMP サーバーをセットアップするコマンドを実行します。各例では、次のタスクが実行されます。

- ディストリビューションソフトウェアパッケージが更新されます。

- ウェブサーバー、php、mariadb パッケージがインストールされます。
- httpd サービスが開始され、有効になります。
- ユーザー ec2-user が apache グループに追加されます。
- ウェブディレクトリとその中に含まれるファイルに対して、適切な所有権とファイル権限が設定されます。
- ウェブサーバーと PHP エンジンをテストするために、シンプルなウェブページが作成されます。

内容

- 前提条件
- ユーザーデータとシェルスクリプト
- インスタンスのユーザーデータを更新する
- ユーザーデータと cloud-init ディレクティブ
- ユーザーデータと AWS CLI
- シェルスクリプトと cloud-init ディレクティブを組み合わせる

前提条件

このトピックの例では、

- インスタンスには、インターネットからアクセス可能なパブリック DNS 名が設定されていることを前提にしています。
- インスタンスに関連付けられたセキュリティグループは、SSH (ポート 22) トラフィックを許可するように設定されているため、インスタンスに接続して出力ログファイルを表示できます。
- Amazon Linux AMI を使用してインスタンスが起動します。コマンドとディレクティブは、他の Linux ディストリビューションでは機能しないことがあります。cloud-init のサポートなど、その他のディストリビューションについての詳細は、特定のディストリビューションのドキュメントを参照してください。

ユーザーデータとシェルスクリプト

シェルスクリプトに慣れている場合は、この方法が最も簡単で完全に起動時に指示を送信する方法です。起動時にこれらのタスクを追加すると、インスタンスの起動にかかる時間が増えます。タスクが完了するまでさらに数分待ち、それからユーザースクリプトが正常に完了したことをテストしてください。

> **Important**
>
> デフォルトでは、ユーザーデータスクリプトと cloud-init ディレクティブは、インスタンスの最初の起動サイクル中にのみ実行されます。インスタンスを再起動するたびにユーザーデータスクリプトと cloud-init ディレクティブが実行されるように設定を更新できます。詳細については、AWS ナレッジセンターのAmazon EC2 Linux インスタンスを再起動する度にユーザーデータを実行して自動的にファイルを作成するにはどうすればよいですか?を参照してください。

ユーザーデータのシェルスクリプトは、#! の記号と、スクリプトを読み取るインタープリタのパス (通常は /bin/bash)) から始める必要があります。シェルスクリプトの概要については、GNU オペレーティングシステムのウェブサイトにある「Bash Reference Manual」を参照してください。

ユーザーデータとして入力されたスクリプトはルートユーザーとして実行されます。そのため、スクリプトでは sudo コマンドを使用しないでください。作成したファイルはすべてルートユーザーの所有になることを忘れないでください。ルート以外のユーザーにファイルアクセスを与える場合、スクリプトで許可を適宜変更する必要があります。また、スクリプトはインタラクティブに実行されないため、ユーザーフィードバックを必要とするコマンド (-y フラグのない yum update など) を含めることはできません。

ユーザーデータスクリプトで AWS API (AWS CLI など) を使用する場合は、インスタンスを起動するときにインスタンスプロファイルを使用する必要があります。インスタンスプロファイルは、API 呼び出しを発行するためにユーザーデータスクリプトが必要とする適切な AWS 認証情報を提供します。詳細については、IAM ユーザーガイド のインスタンスプロファイルの使用を参照してください。IAM ロールに割り当てるアクセス許可は、API で呼び出すサービスによって異なります。詳細については、「Amazon EC2 の IAM ロール」を参照してください。

cloud-init 出力ログファイルでコンソール出力がキャプチャされるため、インスタンスが意図したように動作しない場合でも、起動後、簡単にスクリプトをデバッグすることができます。ログファイルを表示するには、インスタンスに接続し、/var/log/cloud-init-output.log を開きます。

ユーザーデータスクリプトを処理すると、/var/lib/cloud/instances/*instance-id*/ にコピーされ、実行されます。実行後にスクリプトを削除することはできません。必ず /var/lib/cloud/instances/*instance-id*/ のユーザーデータスクリプトを削除してから、インスタンスに AMI を作成してください。それ以外の場合、スクリプトはこの AMI から起動されたインスタンスのこのディレクトリに存在します。

インスタンスのユーザーデータを更新する

インスタンスのユーザーデータを更新するには、まずインスタンスを停止する必要があります。インスタンスが実行されている場合は、ユーザーデータを表示できますが、変更することはできません。

> **Warning**
>
> インスタンスストアボリューム上のデータは、インスタンスを停止すると失われます。このデータを保持するには、永続ストレージにバックアップしてください。

インスタンスユーザーデータを変更するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択し、[Instance state (インスタンスの状態)]、[Stop instance (インスタンスの停止)] の順に選択してください。このオプションが無効になっている場合はインスタンスが既に停止しているか、またはルートボリュームがインスタンスストアボリュームです。
4. 確認を求められたら、[Stop] を選択してください。インスタンスが停止するまで、数分かかる場合があります。
5. インスタンスが選択された状態のまま、[Actions (アクション)]、[Instance settings (インスタンス設定)]、[Edit user data (ユーザーデータの編集)] の順に選択してください。
6. 必要に応じてユーザーデータを変更し、[Save (保存)] を選択してください。
7. インスタンスを起動します。新しいユーザーデータは、再起動後にインスタンス上に表示されますが、ユーザーデータスクリプトは実行されません。

ユーザーデータと cloud-init ディレクティブ

cloud-init パッケージは、新しい Amazon Linux インスタンスが起動したときに、特定の側面を設定します。具体的には、お客様のプライベートキーでログインできるように、ec2-user の `.ssh/authorized_keys` ファイルを設定します。Amazon Linux インスタンスに対して cloud-init パッケージが実行する設定タスクの詳細については、次のドキュメントを参照してください。

- Amazon Linux 2023 – カスタマイズされた cloud-init
- Amazon Linux 2 – Amazon Linux 2 上の cloud-init の使用

構文は異なりますが、渡されたスクリプトと同じ方法で cloud-init ユーザーディレクティブを起動時のインスタンスに渡すことができます。cloud-init の詳細については、https://cloudinit.readthedocs.org/en/latest/index.html を参照してください。

> **Important**
>
> デフォルトでは、ユーザーデータスクリプトと cloud-init ディレクティブは、インスタンスの最初の起動サイクル中にのみ実行されます。インスタンスを再起動するたびにユーザーデータスクリプトと cloud-init ディレクティブが実行されるように設定を更新できます。詳細については、AWS ナレッジセンターのAmazon EC2 Linux インスタンスを再起動する度にユーザーデータを実行して自動的にファイルを作成するにはどうすればよいですか?を参照してください。

起動時にこれらのタスクを追加すると、インスタンスの起動にかかる時間が増えます。タスクが完了するまでさらに数分待ち、それからユーザーデータディレクティブが完了したことをテストしてください。

cloud-init ディレクティブを Amazon Linux インスタンスに渡すには

1. インスタンスを起動するための手順に従います。[User data] (ユーザーデータ) フィールドは、インスタンス起動ウィザードの 高度な詳細 セクションにあります。cloud-init ディレクティブテキストを[User data] (ユーザーデータ) フィールドに入力してから、インスタンスの起動手順を完了します。

   下の例では、ディレクティブが Amazon Linux でウェブサーバーを作成して設定します。一番上の `#cloud-config` 行は、cloud-init ディレクティブとしてコマンドを識別するために必要です。

   AL2023

   ```
   #cloud-config
   package_update: true
   package_upgrade: all

   packages:
   - httpd
   - mariadb105-server
   - php8.1
   - php8.1-mysqlnd
   ```

```
runcmd:
- systemctl start httpd
- systemctl enable httpd
- [ sh, -c, "usermod -a -G apache ec2-user" ]
- [ sh, -c, "chown -R ec2-user:apache /var/www" ]
- chmod 2775 /var/www
- [ find, /var/www, -type, d, -exec, chmod, 2775, {}, \; ]
- [ find, /var/www, -type, f, -exec, chmod, 0664, {}, \; ]
- [ sh, -c, 'echo "<?php phpinfo(); ?>" > /var/www/html/phpinfo.php' ]
```

AL2

```
#cloud-config
package_update: true
package_upgrade: all

packages:
- httpd
- mariadb-server

runcmd:
- [ sh, -c, "amazon-linux-extras install -y lamp-mariadb10.2-php7.2 php7.2" ]
- systemctl start httpd
- systemctl enable httpd
- [ sh, -c, "usermod -a -G apache ec2-user" ]
- [ sh, -c, "chown -R ec2-user:apache /var/www" ]
- chmod 2775 /var/www
- [ find, /var/www, -type, d, -exec, chmod, 2775, {}, \; ]
- [ find, /var/www, -type, f, -exec, chmod, 0664, {}, \; ]
- [ sh, -c, 'echo "<?php phpinfo(); ?>" > /var/www/html/phpinfo.php' ]
```

2. インスタンスが起動し、ユーザーデータのディレクティブを実行するまで十分待ち、それから意図したタスクをディレクティブが完了したことを確認します。

   この例では、ウェブブラウザで、ディレクティブが作成した PHP テストファイルの URL を入力してください。この URL は、インスタンスのパブリック DNS アドレスにスラッシュとファイル名を追加したものです。

   ```
   http://my.public.dns.amazonaws.com/phpinfo.php
   ```

PHP 情報ページが表示されるはずです。PHP 情報ページが表示されない場合、使用しているセキュリティグループに HTTP (ポート 80) トラフィックを許可するルールが含まれていることを確認します。詳細については、「セキュリティグループルールの設定」を参照してください。

3. (オプション) ディレクティブが期待したタスクを達成しなかった場合、あるいは単にディレクティブがエラーなしで完了したことを確認する場合は、インスタンスに接続し、出力ログファイル /var/log/cloud-init-output.logを調べ、出力にエラーメッセージがないか探します。デバッグの詳細情報を取得するには、ディレクティブに次の行を追加します:

```
output : { all : '| tee -a /var/log/cloud-init-output.log' }
```

このディレクティブにより、runcmd 出力が /var/log/cloud-init-output.log に送信されます。

ユーザーデータと AWS CLI

AWS CLI を使用して、インスタンスのユーザーデータを指定、変更、表示することができます。インスタンスのメタデータを使用して、インスタンスからユーザーデータを表示する方法については、EC2 インスタンスのインスタンスメタデータにアクセスするを参照してください。

Windows では、AWS CLI を使用する代わりに AWS Tools for Windows PowerShell を使用できます。詳細については、「ユーザーデータと Windows PowerShell用ツール」を参照してください。

例: ユーザーデータは、起動時に指定します。

インスタンスの起動時にユーザーデータを指定するには、run-instances コマンドと --user-data パラメータを使用します。run-instances で、AWS CLI はユーザーデータの base64 エンコードを実行します。

次の例は、コマンドラインで文字列としてスクリプトを指定する方法を示しています。

```
aws ec2 run-instances --image-id ami-abcd1234 --count 1 --instance-type m3.medium \
    --key-name my-key-pair --subnet-id subnet-abcd1234 --security-group-ids sg-abcd1234
 \
    --user-data echo user data
```

次の例は、テキストファイルを使用してスクリプトを指定する方法を示しています。ファイルを指定するには、必ず file:// プレフィクスを使用してください。

```
aws ec2 run-instances --image-id ami-abcd1234 --count 1 --instance-type m3.medium \
    --key-name my-key-pair --subnet-id subnet-abcd1234 --security-group-ids sg-abcd1234
 \
    --user-data file://my_script.txt
```

シェルスクリプトを使用したテキストファイルの例を次に示します。

```
#!/bin/bash
yum update -y
service httpd start
chkconfig httpd on
```

例: 停止しているインスタンスのユーザーデータを変更する

停止したインスタンスのユーザーデータは、modify-instance-attribute コマンドを使用して変更できます。modify-instance-attribute では、AWS CLI はユーザーデータの base64 エンコードを実行しません。

- Linux コンピュータでは、base64 コマンドを使用してユーザーデータをエンコードします。

  ```
  base64 my_script.txt >my_script_base64.txt
  ```

- Windows コンピュータでは、certutil コマンドを使用してユーザーデータをエンコードします。このファイルを AWS CLI で使用する前に、最初の (証明書の開始) 行と最後の (証明書の終了) 行を削除する必要があります。

  ```
  certutil -encode my_script.txt my_script_base64.txt
  notepad my_script_base64.txt
  ```

--attribute および --value パラメータを使用して、エンコードされたテキストファイルを使用してユーザーデータを指定します。ファイルを指定するには、必ず file:// プレフィクスを使用してください。

```
aws ec2 modify-instance-attribute --instance-id i-1234567890abcdef0 --attribute
 userData --value file://my_script_base64.txt
```

例: 停止しているインスタンスのユーザーデータをクリアする

既存のユーザーデータを削除するには、次の modify-instance-attribute コマンドを使用します。

```
aws ec2 modify-instance-attribute --instance-id i-1234567890abcdef0 --user-data Value=
```

例: ユーザーデータの表示

インスタンスのユーザーデータを取得するには、describe-instance-attribute コマンドを使用します。describe-instance-attribute では、AWS CLI はユーザーデータの base64 デコードを実行しません。

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --attribute
 userData
```

ユーザーデータが base64 でエンコードされた出力例を次に示します。

```
{
    "UserData": {
        "Value":
 "IyEvYmluL2Jhc2gKeXVtIHVwZGF0ZSAteQpzZXJ2aWNlIGh0dHBkIHN0YXJ0CmNoa2NvbmZpZyBodHRwZCBvbg=="
    },
    "InstanceId": "i-1234567890abcdef0"
}
```

- Linux コンピュータでは、--query オプションを使用してエンコードされたユーザーデータを取得し、base64 コマンドを使用してデコードします。

  ```
  aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --attribute
   userData --output text --query "UserData.Value" | base64 --decode
  ```

- Windows コンピュータでは、--query オプションを使用してコード化されたユーザーデータを取得し、certutil コマンドを使用してコードをデコードします。エンコードされた出力はファイルに保存され、デコードされた出力は別のファイルに保存されることに注意してください。

  ```
  aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --attribute
   userData --output text --query "UserData.Value" >my_output.txt
  certutil -decode my_output.txt my_output_decoded.txt
  type my_output_decoded.txt
  ```

以下は出力の例です。

```
#!/bin/bash
```

```
yum update -y
service httpd start
chkconfig httpd on
```

シェルスクリプトと cloud-init ディレクティブを組み合わせる

デフォルトでは、ユーザーデータに含めることができるコンテンツタイプは一度に 1 つだけです。ただし、MIME マルチパートファイルの中で text/cloud-config と text/x-shellscript のコンテンツタイプを使用して、ユーザーデータにシェルスクリプトと cloud-init ディレクティブの両方を含めることは可能です。

以下に、MIME マルチパートの形式を示します。

```
Content-Type: multipart/mixed; boundary="//"
MIME-Version: 1.0

--//
Content-Type: text/cloud-config; charset="us-ascii"
MIME-Version: 1.0
Content-Transfer-Encoding: 7bit
Content-Disposition: attachment; filename="cloud-config.txt"

#cloud-config
cloud-init directives

--//
Content-Type: text/x-shellscript; charset="us-ascii"
MIME-Version: 1.0
Content-Transfer-Encoding: 7bit
Content-Disposition: attachment; filename="userdata.txt"

#!/bin/bash
shell script commands
--//--
```

例えば、次のユーザーデータには cloud-init ディレクティブと bash シェルスクリプトが含まれています。cloud-init ディレクティブはファイル /test-cloudinit/cloud-init.txtを作成し、そのファイルに Created by cloud-init を書き込みます。bash シェルスクリプトはファイル /test-userscript/userscript.txtを作成し、そのファイルに Created by bash shell script を書き込みます。

```
Content-Type: multipart/mixed; boundary="//"
```

```
MIME-Version: 1.0

--//
Content-Type: text/cloud-config; charset="us-ascii"
MIME-Version: 1.0
Content-Transfer-Encoding: 7bit
Content-Disposition: attachment; filename="cloud-config.txt"

#cloud-config
runcmd:
- [ mkdir, /test-cloudinit ]
write_files:
- path: /test-cloudinit/cloud-init.txt
content: Created by cloud-init

--//
Content-Type: text/x-shellscript; charset="us-ascii"
MIME-Version: 1.0
Content-Transfer-Encoding: 7bit
Content-Disposition: attachment; filename="userdata.txt"

#!/bin/bash
mkdir test-userscript
touch /test-userscript/userscript.txt
echo "Created by bash shell script" >> /test-userscript/userscript.txt
--//--
```

Amazon EC2 が Windows インスタンスのユーザーデータを処理する方法

Windows インスタンスでは、起動エージェントはユーザーデータに関連するタスクを実行します。詳細については次を参照してください:

- EC2Launch v2
- EC2Launch
- EC2Config サービス

`UserData` テンプレート内の CloudFormation プロパティのアセンブリ例については、UserData プロパティの Base64 エンコードおよびAccessKey と SecretKey を含む UserData プロパティの Base64 エンコードを参照してください。

ライフサイクルフックと連携する Auto Scaling グループ内のインスタンスでコマンドを実行する例については、「Amazon EC2 Auto Scaling ユーザーガイド」の「チュートリアル: インスタンスのメタデータを使用してターゲットライフサイクル状態を取得するようにユーザーデータを設定する」を参照してください。

内容

- ユーザーデータスクリプト
- 圧縮ユーザーデータ
- ユーザーデータの実行
- ユーザーデータと Windows PowerShell用ツール

ユーザーデータスクリプト

EC2Config または EC2Launch でスクリプトを実行するには、そのスクリプトをユーザーデータに追加するときにスクリプトを特別なタグで囲む必要があります。使用するタグは、コマンドをコマンドプロンプトウィンドウ (バッチコマンド) で実行するか、Windows PowerShell を使用するかによって異なります。

バッチスクリプトと Windows PowerShell スクリプトの両方を指定すると、バッチスクリプトが最初に実行され、インスタンスユーザーデータに表示される順序に関係なく、次に Windows PowerShell スクリプトが実行されます。

ユーザーデータスクリプトで AWS API (AWS CLI など) を使用する場合は、インスタンスを起動するときにインスタンスプロファイルを使用する必要があります。インスタンスプロファイルは、API 呼び出しを実行するためにユーザーデータスクリプトが必要とする適切な AWS 認証情報を提供します。詳細については、インスタンスプロファイルを参照してください。IAM ロールに割り当てるアクセス許可は、API で呼び出すサービスによって異なります。詳細については、「Amazon EC2 の IAM ロール」を参照してください。

スクリプトタイプ

- バッチスクリプトの構文
- Windows PowerShell スクリプトの構文
- YAML 設定スクリプトの構文
- Base64 エンコード

バッチスクリプトの構文

script タグを使用してバッチ スクリプトを指定します。次の例に示すように、改行を使用してコマンドを区切ります。

```
<script>
    echo Current date and time >> %SystemRoot%\Temp\test.log
    echo %DATE% %TIME% >> %SystemRoot%\Temp\test.log
</script>
```

デフォルトでは、ユーザーデータスクリプトはインスタンスを起動すると一度だけ実行されます。インスタンスを再起動または起動するたびにユーザーデータスクリプトを実行するには、<persist>true</persist> をユーザーデータに追加します。

```
<script>
    echo Current date and time >> %SystemRoot%\Temp\test.log
    echo %DATE% %TIME% >> %SystemRoot%\Temp\test.log
</script>
<persist>true</persist>
```

EC2 ローンチ v2 エージェント

XML ユーザーデータスクリプトを UserData ステージ内の EC2Launch v2 executeScript タスクでデタッチされたプロセスとして実行するには、ユーザーデータに <detach>true</detach> を追加します。

> **Note**
>
> detach タグは以前の起動エージェントではサポートされていません。

```
<script>
    echo Current date and time >> %SystemRoot%\Temp\test.log
    echo %DATE% %TIME% >> %SystemRoot%\Temp\test.log
</script>
<detach>true</detach>
```

Windows PowerShell スクリプトの構文

AWS Windows AMI には AWS Tools for Windows PowerShell が含まれており、ユーザーデータにこれらのコマンドレットを指定することができます。インスタンスに IAM ロールを関連付けている

場合は、コマンドレットに認証情報を指定する必要はありません。インスタンスで実行するアプリケーションは、ロールの認証情報を使用して AWS リソース (Simple Storage Service (Amazon S3) バケットなど) にアクセスします。

<powershell> タグを使用して Windows PowerShell スクリプトを指定します。コマンドは、改行を使って区切ります。<powershell> タグでは、大文字と小文字が区別されます。

例えば、次のようになります。

```
<powershell>
    $file = $env:SystemRoot + "\Temp\" + (Get-Date).ToString("MM-dd-yy-hh-mm")
    New-Item $file -ItemType file
</powershell>
```

デフォルトでは、ユーザーデータスクリプトはインスタンスを起動すると一度だけ実行されます。インスタンスを再起動または起動するたびにユーザーデータスクリプトを実行するには、<persist>true</persist> をユーザーデータに追加します。

```
<powershell>
    $file = $env:SystemRoot + "\Temp\" + (Get-Date).ToString("MM-dd-yy-hh-mm")
    New-Item $file -ItemType file
</powershell>
<persist>true</persist>
```

<powershellArguments> タグを使用して、1 つ以上の PowerShell 引数を指定できます。引数が渡されない場合、EC2Launch と EC2Launch v2 はデフォルトで次の引数を追加します: `-ExecutionPolicy Unrestricted`

例:

```
<powershell>
    $file = $env:SystemRoot + "\Temp" + (Get-Date).ToString("MM-dd-yy-hh-mm")
    New-Item $file -ItemType file
</powershell>
<powershellArguments>-ExecutionPolicy Unrestricted -NoProfile -NonInteractive</
powershellArguments>
```

EC2 ローンチ v2 エージェント

XML ユーザーデータスクリプトを UserData ステージ内の EC2Launch v2 executeScript タスクでデタッチされたプロセスとして実行するには、ユーザーデータに <detach>true</detach> を追加します。

> **Note**
>
> detach タグは以前の起動エージェントではサポートされていません。

```
<powershell>
    $file = $env:SystemRoot + "\Temp\" + (Get-Date).ToString("MM-dd-yy-hh-mm")
    New-Item $file -ItemType file
</powershell>
<detach>true</detach>
```

YAML 設定スクリプトの構文

EC2Launch v2 を使用してスクリプトを実行する場合は、YAML 形式を使用できます。EC2Launch v2 の設定タスク、詳細、例を表示するには、EC2Launch v2 タスクの設定を参照してください。

executeScript タスクで YAML スクリプトを指定します。

PowerShell スクリプトを実行するための YAML 構文の例

```
version: 1.0
tasks:
- task: executeScript
  inputs:
  - frequency: always
    type: powershell
    runAs: localSystem
    content: |-
      $file = $env:SystemRoot + "\Temp\" + (Get-Date).ToString("MM-dd-yy-hh-mm")
      New-Item $file -ItemType file
```

バッチスクリプトを実行するための YAML 構文の例

```
version: 1.1
tasks:
- task: executeScript
  inputs:
```

```
  - frequency: always
    type: batch
    runAs: localSystem
    content: |-
      echo Current date and time >> %SystemRoot%\Temp\test.log
      echo %DATE% %TIME% >> %SystemRoot%\Temp\test.log
```

Base64 エンコード

Amazon EC2 API または、ユーザーデータの base64 エンコーディングが実行されないツールを使用している場合、ユーザーデータを手動でエンコードする必要があります。エンコードしない場合、実行する `script` タグまたは `powershell` タグが見つからないというエラーが記録されます。Windows PowerShell を使用してエンコードする例を次に示します。

```
$UserData =
 [System.Convert]::ToBase64String([System.Text.Encoding]::ASCII.GetBytes($Script))
```

PowerShell を使用してデコードする例を次に示します。

```
$Script =
 [System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String($UserData))
```

base64 エンコードの詳細については、https://www.ietf.org/rfc/rfc4648.txtを参照してください。

圧縮ユーザーデータ

EC2Launch v2 は、IMDS が課す上限の 16 KB を超えるユーザーデータを送信する方法として、zip 形式のユーザーデータをサポートしています。この機能を使用するには、ユーザーデータスクリプトを `.zip` アーカイブに圧縮してから、EC2 インスタンスに渡します。EC2Launch v2 が圧縮されたユーザーデータを検出すると、圧縮されたユーザーデータスクリプトを自動的に解凍して実行します。

標準ユーザーデータと同様に、Amazon EC2 API、またはユーザーデータの base64 エンコーディングを実行しないツールを使用している場合は、圧縮されたユーザーデータを手動でエンコードする必要があります。ユーザーデータのサイズ制限と base64 エンコーディングの詳細については、「EC2 インスタンスのインスタンスメタデータにアクセスする」を参照してください。

ユーザーデータの実行

デフォルトでは、すべての AWS Windows AMI で初回起動時のユーザーデータの実行が有効になっています。次にインスタンスが再起動されたときにユーザーデータスクリプトが実行されるように指

定できます。また、インスタンスが再起動するたびにユーザーデータスクリプトが実行されるように指定することもできます。

> **Note**
>
> ユーザーデータは、初期起動後にデフォルトで実行するように有効化されていません。インスタンスを再起動または起動したときにユーザーデータが実行できるようにするには、「その後の再起動または起動時にスクリプトを実行する」を参照してください。

ユーザーデータスクリプトは、ランダムなパスワードが生成されたときにローカル管理者アカウントから実行されます。それ以外の場合、ユーザーデータスクリプトはシステムアカウントから実行されます。

インスタンス起動スクリプト

インスタンスユーザーデータ内のスクリプトは、インスタンスの最初の起動時に実行されます。`persist` タグが見つかった場合は、その後の再起動または開始時にユーザーデータの実行が有効になります。EC2Launch v2、EC2Launch、EC2Config のログファイルには、標準出力ストリームと標準エラーストリームの出力が含まれています。

EC2Launch v2

EC2Launch v2 のログファイルは `C:\ProgramData\Amazon\EC2Launch\log\agent.log` です。

> **Note**
>
> `C:\ProgramData` フォルダは非表示になっている場合があります。このフォルダを表示するには、非表示のファイルおよびフォルダを表示する必要があります。

ユーザーデータが実行されると、次の情報が記録されます。

- `Info: Converting user-data to yaml format` – ユーザーデータが XML 形式で提供されているか
- `Info: Initialize user-data state` – ユーザーデータ実行の開始
- `Info: Frequency is: always` – ブートのたびにユーザーデータタスクが実行されているか

- Info: Frequency is: once – ユーザーデータタスク実行は一度だけか
- Stage: postReadyUserData execution completed – ユーザーデータの実行の終了

EC2Launch

EC2Launch のログファイルは C:\ProgramData\Amazon\EC2-Windows\Launch\Log\UserdataExecution.log です。

C:\ProgramData フォルダは非表示になっている場合があります。このフォルダを表示するには、非表示のファイルおよびフォルダを表示する必要があります。

ユーザーデータが実行されると、次の情報が記録されます。

- Userdata execution begins – ユーザーデータ実行の開始
- <persist> tag was provided: true – 永続タグが見つかったか
- Running userdata on every boot – 永続タグが見つかったか
- <powershell> tag was provided.. running powershell content – PowerShell タグが見つかったか
- <script> tag was provided.. running script content – スクリプトタグが見つかったか
- Message: The output from user scripts – 実行されたユーザーデータスクリプトがあり、その出力が記録されたか

EC2Config

EC2Config のログファイルは C:\Program Files\Amazon\Ec2ConfigService\Logs\Ec2Config.log です。ユーザーデータが実行されると、次の情報が記録されます。

- Ec2HandleUserData: Message: Start running user scripts – ユーザーデータ実行の開始
- Ec2HandleUserData: Message: Re-enabled userdata execution – 永続タグが見つかったか
- Ec2HandleUserData: Message: Could not find <persist> and </persist> – 永続タグが見つからなかったのか
- Ec2HandleUserData: Message: The output from user scripts – 実行されたユーザーデータスクリプトがあり、その出力が記録されたか

その後の再起動または起動時にスクリプトを実行する

インスタンスのユーザー データを更新すると、次回インスタンスを再起動または起動したときに、更新されたユーザーデータコンテンツがインスタンスメタデータに自動的に反映されます。ただし、インストールされている起動エージェントによっては、その後の再起動または起動時にユーザー データ スクリプトを実行するように構成するための追加の設定が必要になる場合があります。

[Shutdown with Sysprep (Sysprep でシャットダウン)] オプションを選択すると、以降のインスタンスの起動時や再起動時にユーザーデータの実行を有効にしなくても、次回の起動時や再起動時にユーザーデータスクリプトが実行されます。

ユーザーデータの実行を有効にする手順については、起動エージェントに一致するタブを選択して確認してください。

EC2Launch v2

EC2Launch v1 とは異なり、EC2Launch v2 は起動のたびにユーザー データ タスクを評価します。ユーザーデータタスクを手動でスケジュールする必要はありません。ユーザーデータは、含まれている頻度または永続オプションに基づいて実行されます。

XML ユーザーデータスクリプトの場合

起動のたびにユーザーデータスクリプトを実行するには、ユーザーデータに `<persist>true</persist>` フラグを追加します。永続フラグが含まれていない場合、ユーザーデータスクリプトは最初の起動時にのみ実行されます。

YAML ユーザーデータの場合

- 最初の起動時にユーザーデータでタスクを実行するには、タスク `frequency` を `once` に設定します。
- 起動するたびにユーザーデータ内のタスクを実行するには、タスク `frequency` を `always` に設定します。

EC2Launch

1. Windows インスタンスに接続します。
2. PowerShell コマンドウィンドウを開き、次のいずれかのコマンドを実行します。

   1 回実行する

   次回の起動時にユーザーデータを 1 回実行するには、`-Schedule` フラグを使用します。

```
C:\ProgramData\Amazon\EC2-Windows\Launch\Scripts\InitializeInstance.ps1 -
Schedule
```

以降のすべての起動時に実行する

以降のすべての起動時にユーザー データを実行するには、 `-SchedulePerBoot` フラグを使用します。

```
C:\ProgramData\Amazon\EC2-Windows\Launch\Scripts\InitializeInstance.ps1 -
SchedulePerBoot
```

3. Windows インスタンスから切断します。次にインスタンスを起動するときに更新されたスクリプトを実行するには、インスタンスを停止してユーザーデータを更新します。

EC2Config

1. Windows インスタンスに接続します。
2. `C:\Program Files\Amazon\Ec2ConfigService\Ec2ConfigServiceSetting.exe` を開く。
3. [ユーザーデータ] で、[Enable UserData execution for next service start (次のサービス開始でユーザーデータの実行を有効にする)] を選択してください。
4. Windows インスタンスから切断します。次にインスタンスを起動するときに更新されたスクリプトを実行するには、インスタンスを停止してユーザーデータを更新します。

ユーザーデータと Windows PowerShell用ツール

Windows PowerShell用ツール を使用して、インスタンスのユーザーデータを指定、変更、表示することができます。インスタンスのメタデータを使用して、インスタンスからユーザーデータを表示する方法については、EC2 インスタンスのインスタンスメタデータにアクセスするを参照してください。ユーザーデータと AWS CLI の詳細については、「ユーザーデータと AWS CLI」を参照してください。

例: 起動時にインスタンスユーザーデータを指定する

インスタンスユーザーデータを含むテキストファイルを作成します。インスタンスを再起動または起動するたびにユーザーデータスクリプトを実行するには、次の例に示すように `<persist>true</persist>` をユーザーデータに追加します。

```
<powershell>
    $file = $env:SystemRoot + "\Temp\" + (Get-Date).ToString("MM-dd-yy-hh-mm")
    New-Item $file -ItemType file
</powershell>
<persist>true</persist>
```

インスタンスの起動時にインスタンスユーザーデータを指定するには、New-EC2Instance コマンドを使用します。このコマンドは、ユーザーデータの base64 エンコードを実行しません。次のコマンドを使用して、script.txt という名前のテキストファイルにユーザーデータをエンコードします。

```
PS C:\> $Script = Get-Content -Raw script.txt
PS C:\> $UserData =
 [System.Convert]::ToBase64String([System.Text.Encoding]::ASCII.GetBytes($Script))
```

-UserData パラメータを使用して、ユーザーデータを New-EC2Instance コマンドに渡します。

```
PS C:\> New-EC2Instance -ImageId ami-abcd1234 -MinCount 1 -MaxCount 1 -
InstanceType m3.medium \
    -KeyName my-key-pair -SubnetId subnet-12345678 -SecurityGroupIds sg-1a2b3c4d \
    -UserData $UserData
```

例: 停止したインスタンスのインスタンスユーザーデータを更新する

停止したインスタンスのユーザーデータは、Edit-EC2InstanceAttribute コマンドを使用して変更できます。

新しいスクリプトを使用してテキストファイルを作成します。次のコマンドを使用して、new-script.txt という名前のテキストファイルにユーザーデータをエンコードします。

```
PS C:\> $NewScript = Get-Content -Raw new-script.txt
PS C:\> $NewUserData =
 [System.Convert]::ToBase64String([System.Text.Encoding]::ASCII.GetBytes($NewScript))
```

-UserData および -Value パラメータを使用して、ユーザーデータを指定します。

```
PS C:\> Edit-EC2InstanceAttribute -InstanceId i-1234567890abcdef0 -Attribute userData -
Value $NewUserData
```

例: インスタンスユーザーデータを表示する

インスタンスのユーザーデータを取得するには、Get-EC2InstanceAttribute コマンドを使用します。

```
PS C:\> (Get-EC2InstanceAttribute -InstanceId i-1234567890abcdef0 -Attribute
 userData).UserData
```

出力例を次に示します。ユーザーデータはエンコードされていることに注意してください。

```
PHBvd2Vyc2hlbGw
+DQpSZW5hbWUtQ29tcHV0ZXIgLU5ld05hbWUgdXNlci1kYXRhLXRlc3QNCjwvcG93ZXJzaGVsbD4=
```

次のコマンドを使用して、エンコードされたユーザーデータを変数に格納し、それをデコードします。

```
PS C:\> $UserData_encoded = (Get-EC2InstanceAttribute -InstanceId i-1234567890abcdef0 -
Attribute userData).UserData
PS C:
\> [System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String($UserData_encoded)
```

出力例を次に示します。

```
<powershell>
    $file = $env:SystemRoot + "\Temp\" + (Get-Date).ToString("MM-dd-yy-hh-mm")
    New-Item $file -ItemType file
</powershell>
<persist>true</persist>
```

例: タグ値と合うようにインスタンスの名前を変更する

Get-EC2Tag コマンドを使用してタグ値を読み出し、タグ値と一致するように最初の起動時にインスタンスの名前を変更して、再起動することができます。タグ情報は API コールで取得されるため、このコマンドを適切に実行するには、ec2:DescribeTags アクセス許可が付与されているロールがインスタンスにアタッチされている必要があります。IAM ロールを使用した設定のアクセス許可の詳細については、「インスタンスへの IAM ロールのアタッチ」を参照してください。

IMDSv2

```
<powershell>
    [string]$token = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token-ttl-
seconds" = "21600"} -Method PUT -Uri 'http://169.254.169.254/latest/api/token' -
UseBasicParsing
```

```
    $instanceId = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" =
 $token} -Method GET -Uri 'http://169.254.169.254/latest/meta-data/instance-id' -
UseBasicParsing
 $nameValue = (Get-EC2Tag -Filter @{Name="resource-id";Value=
$instanceid},@{Name="key";Value="Name"}).Value
 $pattern = "^(?![0-9]{1,15}$)[a-zA-Z0-9-]{1,15}$"
 #Verify Name Value satisfies best practices for Windows hostnames
 If ($nameValue -match $pattern)
     {Try
         {Rename-Computer -NewName $nameValue -Restart -ErrorAction Stop}
     Catch
         {$ErrorMessage = $_.Exception.Message
         Write-Output "Rename failed: $ErrorMessage"}}
 Else
     {Throw "Provided name not a valid hostname. Please ensure Name value is between
 1 and 15 characters in length and contains only alphanumeric or hyphen characters"}
</powershell>
```

IMDSv1

```
<powershell>
 $instanceId = (Invoke-WebRequest http://169.254.169.254/latest/meta-data/instance-
id -UseBasicParsing).content
 $nameValue = (Get-EC2Tag -Filter @{Name="resource-id";Value=
$instanceid},@{Name="key";Value="Name"}).Value
 $pattern = "^(?![0-9]{1,15}$)[a-zA-Z0-9-]{1,15}$"
 #Verify Name Value satisfies best practices for Windows hostnames
 If ($nameValue -match $pattern)
     {Try
         {Rename-Computer -NewName $nameValue -Restart -ErrorAction Stop}
     Catch
         {$ErrorMessage = $_.Exception.Message
         Write-Output "Rename failed: $ErrorMessage"}}
 Else
     {Throw "Provided name not a valid hostname. Please ensure Name value is between
 1 and 15 characters in length and contains only alphanumeric or hyphen characters"}
</powershell>
```

インスタンスメタデータからタグにアクセスするようにインスタンスが設定されている場合は、インスタンスメタデータ内のタグを使用してインスタンスの名前を変更することもできます。詳細については、「インスタンスメタデータを使用して EC2 インスタンスのタグを表示する」を参照してください。

IMDSv2

```
<powershell>
    [string]$token = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token-ttl-
seconds" = "21600"} -Method PUT -Uri 'http://169.254.169.254/latest/api/token' -
UseBasicParsing
 $nameValue = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token}
 -Method GET -Uri 'http://169.254.169.254/latest/meta-data/tags/instance/Name' -
UseBasicParsing
 $pattern = "^(?![0-9]{1,15}$)[a-zA-Z0-9-]{1,15}$"
 #Verify Name Value satisfies best practices for Windows hostnames
 If ($nameValue -match $pattern)
     {Try
         {Rename-Computer -NewName $nameValue -Restart -ErrorAction Stop}
     Catch
         {$ErrorMessage = $_.Exception.Message
         Write-Output "Rename failed: $ErrorMessage"}}
 Else
     {Throw "Provided name not a valid hostname. Please ensure Name value is between
 1 and 15 characters in length and contains only alphanumeric or hyphen characters"}
</powershell>
```

IMDSv1

```
<powershell>
 $nameValue = Get-EC2InstanceMetadata -Path /tags/instance/Name
 $pattern = "^(?![0-9]{1,15}$)[a-zA-Z0-9-]{1,15}$"
 #Verify Name Value satisfies best practices for Windows hostnames
 If ($nameValue -match $pattern)
     {Try
         {Rename-Computer -NewName $nameValue -Restart -ErrorAction Stop}
     Catch
         {$ErrorMessage = $_.Exception.Message
         Write-Output "Rename failed: $ErrorMessage"}}
 Else
     {Throw "Provided name not a valid hostname. Please ensure Name value is between
 1 and 15 characters in length and contains only alphanumeric or hyphen characters"}
</powershell>
```

1 つのリクエストで起動された各インスタンスを特定する

この例はユーザーデータおよびインスタンスメタデータの両方を使用して Amazon EC2 インスタンスを設定する方法を示しています。

> **Note**
>
> このセクションの例ではIMDS の IPv4 アドレス 169.254.169.254 を使用します。IPv6 アドレスを使用して EC2 インスタンスのインスタンスメタデータを取得する場合はIPv6 アドレスを有効にして使用してください。[fd00:ec2::254]。IMDS の IPv6 アドレスは IMDSv2 コマンドと互換性があります。IPv6 アドレスにはNitro ベースのインスタンスと IPv6 対応サブネット (デュアルスタックまたは IPv6 のみ) でのみアクセスできます。

この例で、Alice はお気に入りのデータベース AMI の 4 つのインスタンスを起動します。最初のインスタンスを元のインスタンスとし、残りの 3 つをレプリカとします。これらのインスタンスの起動時に、レプリケーション戦略に関するユーザーデータを各レプリカに追加します。このデータはすべての 4 つのインスタンスで使用可能となるので、どの部分が各インスタンスに該当するかをそれぞれが認識できるように、ユーザーデータを構築する必要があります。この構築は各インスタンスに対して一意となる ami-launch-index インスタンスメタデータ値を使用して行うことができます。同時に複数のインスタンスを起動する場合、ami-launch-index はインスタンスが起動された順序を示します。最初に起動されたインスタンスの値は 0 で示されます。

Alice が構築したユーザーデータを次に示します。

```
replicate-every=1min | replicate-every=5min | replicate-every=10min
```

replicate-every=1min データは最初のレプリカの設定を定義し、replicate-every=5min は 2 番目のレプリカの設定を定義します。以下、同様に設定を定義します。Alice は個別のインスタンスのデータをパイプシンボル (|) で区切って、このデータを ASCII 文字列として指定することにしました。

Alice はrun-instancesコマンドを使用して 4 つのインスタンスを起動します。このとき、次のユーザーデータを指定します。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --count 4 \
```

```
    --instance-type t2.micro \
    --user-data "replicate-every=1min | replicate-every=5min | replicate-every=10min"
```

起動したすべてのインスタンスに、ユーザーデータのコピーと次に示す一般的なメタデータが含まれています。

- AMI ID: ami-0abcdef1234567890
- 予約 ID: r-1234567890abcabc0
- パブリックキー: none
- セキュリティグループ名: default
- インスタンスタイプ: t2.micro

ただし、次の表に示すように、各インスタンスには一意のメタデータがあります。

メタデータ	値
instance-id	i-1234567890abcdef0
ami-launch-index	0
public-hostname	ec2-203-0-113-25.compute-1.amazonaws.com
public-ipv4	67.202.51.223
local-hostname	ip-10-251-50-12.ec2.internal
local-ipv4	10.251.50.35

メタデータ	値
instance-id	i-0598c7d356eba48d7
ami-launch-index	1
public-hostname	ec2-67-202-51-224.compute-1.amazonaws.com
public-ipv4	67.202.51.224

メタデータ	値
local-hostname	ip-10-251-50-36.ec2.internal
local-ipv4	10.251.50.36

メタデータ	値
instance-id	i-0ee992212549ce0e7
ami-launch-index	2
public-hostname	ec2-67-202-51-225.compute-1.amazonaws.com
public-ipv4	67.202.51.225
local-hostname	ip-10-251-50-37.ec2.internal
local-ipv4	10.251.50.37

メタデータ	値
instance-id	i-1234567890abcdef0
ami-launch-index	3
public-hostname	ec2-67-202-51-226.compute-1.amazonaws.com
public-ipv4	67.202.51.226
local-hostname	ip-10-251-50-38.ec2.internal
local-ipv4	10.251.50.38

Alice は `ami-launch-index` 値を使用して、ユーザーデータのどの部分が特定のインスタンスに該当するかを判断できます。

1. いずれかのインスタンスに接続し、そのインスタンスの ami-launch-index を取得して、それがレプリカの 1 つであることを確認します。

   IMDSv2

   ```
   [ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/meta-data/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
   && curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/ami-launch-index
   2
   ```

   次のステップではIMDSv2が前のIMDSv2コマンドからの保管済みトークン (期限内であると仮定) を使用するようリクエストします。

   IMDSv1

   ```
   [ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/ami-launch-index
   2
   ```

2. 変数としてami-launch-indexを保存します。

   IMDSv2

   ```
   [ec2-user ~]$ ami_launch_index=`curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/ami-launch-index`
   ```

   IMDSv1

   ```
   [ec2-user ~]$ ami_launch_index=`curl http://169.254.169.254/latest/meta-data/ami-launch-index`
   ```

3. ユーザーデータを変数として保存します。

   IMDSv2

   ```
   [ec2-user ~]$ user_data=`curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/user-data`
   ```

   IMDSv1

   ```
   [ec2-user ~]$ user_data=`curl http://169.254.169.254/latest/user-data`
   ```

4. 最後に、Alice はcutコマンドを使用して、そのインスタンスに該当するユーザーデータの部分を抽出します。

   IMDSv2

   ```
   [ec2-user ~]$ echo $user_data | cut -d"|" -f"$ami_launch_index"
   replicate-every=5min
   ```

   IMDSv1

   ```
   [ec2-user ~]$ echo $user_data | cut -d"|" -f"$ami_launch_index"
   replicate-every=5min
   ```

ホストが EC2 インスタンスであるかどうかを検出する

特に混在コンピューティング環境がある場合は、アプリケーションまたはウェブサイトが EC2 インスタンスで実行されているかどうかを知る必要がある場合があります。次のいずれかのオプションを使用して、アプリケーションまたはウェブサイトのホストが EC2 インスタンスであるかどうかを判断できます。

オプション

- インスタンスアイデンティティドキュメント の検査
- システム UUID の検査
- システムの仮想マシン生成識別子を調べる

インスタンスアイデンティティドキュメント の検査

各インスタンスには、暗号的に検証できる署名付きインスタンス ID ドキュメントがあります。これらのドキュメントは、インスタンスメタデータサービス (IMDS) を使用して確認できます。

詳細については、「インスタンスアイデンティティドキュメント」を参照してください。

システム UUID の検査

システム UUID を取得して、EC2 (Linux では小文字の ec2 になる場合があります) の UUID の先頭オクテットを調べることができます。この方法は、EC2 インスタンスではないシステムがこれらの文字で始まる UUID を持つ可能性が低いため、速いですが不正確である可能性があります。さら

に、SMBIOS の一部のバージョンでは、UUID の先頭に EC2 がないリトルエンディアン形式を使用します。これは、Windows で SMBIOS 2.4 を使用する EC2 インスタンス、または SMBIOS の独自の実装がある Amazon Linux 以外の Linux ディストリビューションの場合に当てはまります。

Linux の例: DMI からの UUID の取得 (HVM AMI のみ)

デスクトップ管理インターフェイス (DMI) を使用して UUID を取得するには、次のコマンドを使用します。

```
[ec2-user ~]$ sudo dmidecode --string system-uuid
```

次の出力例では、UUID は「EC2」で始まりますが、これは多くの場合システムが EC2 インスタンスであることを示しています。

```
EC2E1916-9099-7CAF-FD21-012345ABCDEF
```

次の出力例では、UUID がリトルエンディアン形式で表されています。

```
45E12AEC-DCD1-B213-94ED-012345ABCDEF
```

別の方法として、Nitro システムに構築されたインスタンスの場合には、次のコマンドを使用できます。

```
[ec2-user ~]$ cat /sys/devices/virtual/dmi/id/board_asset_tag
```

次の例のように出力がインスタンス ID となる場合、そのシステムは EC2 インスタンスです。

```
i-0af01c0123456789a
```

Linux の例: ハイパーバイザーからの UUID の取得 (PV AMI のみ)

次のコマンドを使用して、ハイパーバイザーから UUID を取得します。

```
[ec2-user ~]$ cat /sys/hypervisor/uuid
```

次の出力例では、UUID は「ec2」で始まりますが、これは多くの場合システムが EC2 インスタンスであることを示しています。

```
ec2e1916-9099-7caf-fd21-012345abcdef
```

Windows の例: WMI または Windows PowerShell を使用した UUID の取得

以下のように、Windows Management Instrumentation コマンドライン (WMIC) を使用します。

```
wmic path win32_computersystemproduct get uuid
```

または、Windows PowerShell を使用している場合、次のように Get-WmiObject コマンドレットを使用します。

```
PS C:\> Get-WmiObject -query "select uuid from Win32_ComputerSystemProduct" | Select
 UUID
```

次の出力例では、UUID は「EC2」で始まりますが、これは多くの場合システムが EC2 インスタンスであることを示しています。

```
EC2AE145-D1DC-13B2-94ED-012345ABCDEF
```

SMBIOS 2.4 を使用するインスタンスの場合、UUID は次のようにリトルエンディアン形式で表されることがあります。

```
45E12AEC-DCD1-B213-94ED-012345ABCDEF
```

システムの仮想マシン生成識別子を調べる

仮想マシン生成識別子は、暗号化ランダム整数識別子として解釈される 128 ビットの一意のバッファで構成されます。仮想マシン生成識別子を取得すると、Amazon Elastic Compute Cloud インスタンスを識別できます。生成識別子は、ACPI テーブルエントリを介してインスタンスのゲストオペレーティングシステム内に公開されています。AWS にマシンをクローン、コピー、またはインポートすると、値は VM Import/Export などに変わります。

例: Linux からの仮想マシン生成識別子の取得

次のコマンドを使用して、Linux を実行しているインスタンスから仮想マシン生成識別子を取得できます。

Amazon Linux 2

1. 必要に応じて、次のコマンドを使用して既存のソフトウェアパッケージを更新します。

```
sudo yum update
```

2. 必要に応じて、次のコマンドを使用して busybox パッケージを入手します。

```
sudo curl https://www.rpmfind.net/linux/epel/next/8/Everything/x86_64/Packages/
b/busybox-1.35.0-2.el8.next.x86_64.rpm --output busybox.rpm
```

3. 必要に応じて、次のコマンドを使用して前提条件パッケージをインストールします。

```
sudo yum install busybox.rpm iasl -y
```

4. `iasl` コマンドを実行して、ACPI テーブルから出力を生成します。

```
sudo iasl -p ./SSDT2 -d /sys/firmware/acpi/tables/SSDT2
```

5. 次のコマンドを実行して、`iasl` コマンドの出力をレビューします。

```
cat SSDT2.dsl
```

仮想マシン生成識別子を取得するために必要なアドレス空間が生成されているはずです。

```
Intel ACPI Component Architecture
ASL+ Optimizing Compiler/Disassembler version 20190509
Copyright (c) 2000 - 2019 Intel Corporation

File appears to be binary: found 32 non-ASCII characters, disassembling
Binary file appears to be a valid ACPI table, disassembling
Input file /sys/firmware/acpi/tables/SSDT2, Length 0x7B (123) bytes
ACPI: SSDT 0x0000000000000000 00007B (v01 AMAZON AMZNSSDT 00000001 AMZN
 00000001)
Pass 1 parse of [SSDT]
Pass 2 parse of [SSDT]
Parsing Deferred Opcodes (Methods/Buffers/Packages/Regions)

Parsing completed
Disassembly completed
ASL Output:    ./SSDT2.dsl - 1065 bytes
$
/*
* Intel ACPI Component Architecture
* AML/ASL+ Disassembler version 20190509 (64-bit version)
```

```
* Copyright (c) 2000 - 2019 Intel Corporation
*
* Disassembling to symbolic ASL+ operators
*
* Disassembly of /sys/firmware/acpi/tables/SSDT2, Tue Mar 29 16:15:14 2022
*
* Original Table Header:
*     Signature        "SSDT"
*     Length           0x0000007B (123)
*     Revision         0x01
*     Checksum         0xB8
*     OEM ID           "AMAZON"
*     OEM Table ID     "AMZNSSDT"
*     OEM Revision     0x00000001 (1)
*     Compiler ID      "AMZN"
*     Compiler Version 0x00000001 (1)
*/
DefinitionBlock ("", "SSDT", 1, "AMAZON", "AMZNSSDT", 0x00000001)
{
Scope (\_SB)
{
    Device (VMGN)
    {
        Name (_CID, "VM_Gen_Counter")  // _CID: Compatible ID
        Name (_DDN, "VM_Gen_Counter")  // _DDN: DOS Device Name
        Name (_HID, "AMZN0000")  // _HID: Hardware ID
        Name (ADDR, Package (0x02)
        {
            0xFED01000,
            Zero
        })
    }
}
}
```

6. (オプション) 次のコマンドを使用して、残りのステップのターミナル許可を昇格させます。

```
sudo -s
```

7. 次のコマンドを使用して、以前に収集したアドレス空間を保存します。

```
VMGN_ADDR=0xFED01000
```

8. 次のコマンドを使用して、アドレス空間を介して反復処理し、仮想マシン生成識別子を作成します。

```
for offset in 0x0 0x4 0x8 0xc; do busybox devmem $(($VMGN_ADDR + $offset)) | sed
 's/0x//' | sed -z '$ s/\n$//' >> vmgenid; done
```

9. 次のコマンドを使用して、出力ファイルから仮想マシン生成識別子を取得します。

```
cat vmgenid ; echo
```

出力は次のようになります。

```
EC2F335D979132C4165896753E72BD1C
```

Ubuntu

1. 必要に応じて、次のコマンドを使用して既存のソフトウェアパッケージを更新します。

```
sudo apt update
```

2. 必要に応じて、次のコマンドを使用して前提条件パッケージをインストールします。

```
sudo apt install busybox iasl -y
```

3. `iasl` コマンドを実行して、ACPI テーブルから出力を生成します。

```
sudo iasl -p ./SSDT2 -d /sys/firmware/acpi/tables/SSDT2
```

4. 次のコマンドを実行して、`iasl` コマンドの出力をレビューします。

```
cat SSDT2.dsl
```

仮想マシン生成識別子を取得するために必要なアドレス空間が生成されているはずです。

```
Intel ACPI Component Architecture
ASL+ Optimizing Compiler/Disassembler version 20190509
Copyright (c) 2000 - 2019 Intel Corporation

File appears to be binary: found 32 non-ASCII characters, disassembling
```

```
Binary file appears to be a valid ACPI table, disassembling
Input file /sys/firmware/acpi/tables/SSDT2, Length 0x7B (123) bytes
ACPI: SSDT 0x0000000000000000 00007B (v01 AMAZON AMZNSSDT 00000001 AMZN
 00000001)
Pass 1 parse of [SSDT]
Pass 2 parse of [SSDT]
Parsing Deferred Opcodes (Methods/Buffers/Packages/Regions)

Parsing completed
Disassembly completed
ASL Output:    ./SSDT2.dsl - 1065 bytes
$
/*
* Intel ACPI Component Architecture
* AML/ASL+ Disassembler version 20190509 (64-bit version)
* Copyright (c) 2000 - 2019 Intel Corporation
*
* Disassembling to symbolic ASL+ operators
*
* Disassembly of /sys/firmware/acpi/tables/SSDT2, Tue Mar 29 16:15:14 2022
*
* Original Table Header:
*     Signature        "SSDT"
*     Length           0x0000007B (123)
*     Revision         0x01
*     Checksum         0xB8
*     OEM ID           "AMAZON"
*     OEM Table ID     "AMZNSSDT"
*     OEM Revision     0x00000001 (1)
*     Compiler ID      "AMZN"
*     Compiler Version 0x00000001 (1)
*/
DefinitionBlock ("", "SSDT", 1, "AMAZON", "AMZNSSDT", 0x00000001)
{
Scope (\_SB)
{
    Device (VMGN)
    {
        Name (_CID, "VM_Gen_Counter")  // _CID: Compatible ID
        Name (_DDN, "VM_Gen_Counter")  // _DDN: DOS Device Name
        Name (_HID, "AMZN0000")  // _HID: Hardware ID
        Name (ADDR, Package (0x02)
        {
            0xFED01000,
```

```
            Zero
        })
    }
}
}
```

5. (オプション) 次のコマンドを使用して、残りのステップのターミナル許可を昇格させます。

```
sudo -s
```

6. 次のコマンドを使用して、以前に収集したアドレス空間を保存します。

```
VMGN_ADDR=0xFED01000
```

7. 次のコマンドを使用して、アドレス空間を介して反復処理し、仮想マシン生成識別子を作成します。

```
for offset in 0x0 0x4 0x8 0xc; do busybox devmem $(($VMGN_ADDR + $offset)) | sed
 's/0x//' | sed -z '$ s/\n$//' >> vmgenid; done
```

8. 次のコマンドを使用して、出力ファイルから仮想マシン生成識別子を取得します。

```
cat vmgenid ; echo
```

出力は次のようになります。

```
EC2F335D979132C4165896753E72BD1C
```

例: Windows からの仮想マシン生成識別子の取得

Windows を実行しているインスタンスから仮想マシン生成識別子を取得するサンプルアプリケーションを作成できます。詳細については、Microsoft のドキュメントの「Obtaining the virtual machine generation identifier」(仮想マシン生成識別子の取得) を参照してください。

Amazon EC2 インスタンスのインスタンスアイデンティティドキュメント

作成する各インスタンスには、インスタンス自体に関する情報を提供するインスタンスアイデンティティドキュメントがあります。インスタンスアイデンティティドキュメントを使用して、インスタンスの属性を検証することができます。

インスタンスアイデンティティドキュメントは、インスタンスが停止して起動、再起動、起動するときに生成されます。インスタンスのインスタンスアイデンティティドキュメントには、インスタンスメタデータサービス (IMDS) からアクセスできます。手順については、「インスタンスアイデンティティドキュメントを取得する」を参照してください。

インスタンスアイデンティティドキュメントは、プレーンテキストの JSON 形式を使用します。次の情報が含まれています。

データ	説明
`accountId`	インスタンスを起動した AWS アカウントの ID。
`architecture`	インスタンスの作成に使用された AMI のアーキテクチャ (i386 | x86_64 | arm64)。
`availabilityZone`	インスタンスが実行されているアベイラビリティーゾーンの名前。例えば、`us-east-1` 。アベイラビリティーゾーン名は AWS アカウントによって異なる場合があります。
`billingProducts`	インスタンスの請求製品。
`devpayProductCodes`	廃止済み。
`imageId`	インスタンスの起動に使用される AMI の ID。
`instanceId`	インスタンスの ID。
`instanceType`	インスタンスのインスタンスタイプ。
`kernelId`	インスタンスに関連付けられているカーネルの ID (ある場合)。

データ	説明
marketplaceProductCodes	インスタンスの作成に使用された AMI の AWS Marketplace 製品コード。
pendingTime	インスタンスが作成された日時。
privateIp	インスタンスのプライベート IP アドレス。IPv4-only およびデュアルスタックインスタンスの場合、ここに IPv4 アドレスが含まれます。IPv6-onlyインスタンスの場合、ここに IPv6 アドレスが含まれます。
ramdiskId	インスタンスに関連付けられている RAM ディスクの ID (ある場合)。
region	インスタンスが実行されているリージョン。
version	インスタンスアイデンティティドキュメント形式のバージョン。

EC2 インスタンスのインスタンスアイデンティティドキュメントを取得する

Amazon EC2 インスタンスのインスタンスアイデンティティドキュメントは、プレーンテキストの JSON 形式を使用します。インスタンスアイデンティティドキュメントの内容の説明については、「the section called “インスタンスアイデンティティドキュメント”」を参照してください。

インスタンスアイデンティティドキュメントは、インスタンスのインスタンスメタデータの instance-identity/document 動的データカテゴリに保存されます。インスタンスのインスタンスアイデンティティドキュメントにアクセスするには、インスタンスに接続し、インスタンスメタデータから取得します。

IPv4 アドレス 169.254.169.254 または IPv6 アドレス fd00:ec2::254 を使用してインスタンスメタデータにアクセスできます。これらはリンクローカルアドレスです。つまりリンクローカルアドレス、インスタンスからのみアクセスできます。このページの例では、IMDS の IPv4 アドレス 169.254.169.254 を使用します。代わりに IPv6 経由で EC2 インスタンスのインスタンスメタデータを取得するには、代わりに fd00:ec2::254 を使用します。

インスタンスアイデンティティドキュメン インスタンスアイデンティティドキュメントを検証するトの取得後にその信頼性を検証するには。

IMDSv2

Linux

インスタンスアイデンティティドキュメントを取得するには、Linux インスタンスから次のコマンドを実行します。

```
TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
    && curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/dynamic/instance-identity/document
```

Windows

インスタンスアイデンティティドキュメントを取得するには、Windows インスタンスから次のコマンドレットを実行します。

```
[string]$token = (Invoke-WebRequest -Headers @{'X-aws-ec2-metadata-token-ttl-seconds' = '21600'} `
    -Method PUT -Uri 'http://169.254.169.254/latest/api/token' -UseBasicParsing).Content
```

```
(Invoke-WebRequest -Headers @{'X-aws-ec2-metadata-token' = $token} `
    -Uri 'http://169.254.169.254/latest/dynamic/instance-identity/document' -UseBasicParsing).Content
```

IMDSv1

Linux

インスタンスアイデンティティドキュメントを取得するには、Linux インスタンスから次のコマンドを実行します。

```
curl http://169.254.169.254/latest/dynamic/instance-identity/document
```

Windows

インスタンスアイデンティティドキュメントを取得するには、Windows インスタンスから次のコマンドレットを実行します。

```
(Invoke-WebRequest http://169.254.169.254/latest/dynamic/instance-identity/
document).Content
```

以下は出力の例です。

```
{
    "devpayProductCodes" : null,
    "marketplaceProductCodes" : [ "1abc2defghijklm3nopqrs4tu" ],
    "availabilityZone" : "us-west-2b",
    "privateIp" : "10.158.112.84",
    "version" : "2017-09-30",
    "instanceId" : "i-1234567890abcdef0",
    "billingProducts" : null,
    "instanceType" : "t2.micro",
    "accountId" : "123456789012",
    "imageId" : "ami-5fb8c835",
    "pendingTime" : "2016-11-19T16:32:11Z",
    "architecture" : "x86_64",
    "kernelId" : null,
    "ramdiskId" : null,
    "region" : "us-west-2"
}
```

Amazon EC2 インスタンスのインスタンスアイデンティティドキュメントを検証する

インスタンスアイデンティティドキュメントの内容を重要な用途に使用する場合は、使用前にその内容と真正性を検証する必要があります。

プレーンテキストのインスタンスアイデンティティドキュメントには、ハッシュ化および暗号化された署名が 3 つあります。これらの署名を使用して、インスタンスアイデンティティドキュメントの作成元および真正性とそれに含まれている情報を検証できます。提供されている署名は次のとおりです。

- base64 でエンコードされた署名—RSA キーペアを使用して暗号化されたインスタンスアイデンティティドキュメントの base64 でエンコードされた SHA256 ハッシュです。
- PKCS7 署名—DSA キーペアを使用して暗号化されたインスタンスアイデンティティドキュメントの SHA1 ハッシュです。

- RSA-2048 署名—RSA-2048 キーペアを使用して暗号化されたインスタンスアイデンティティドキュメントの SHA256 ハッシュです。

それぞれの署名は、インスタンスメタデータの異なるエンドポイントで取得できます。ハッシュ化と暗号化の要件に応じて、これらの署名のいずれかを使用できます。署名を検証するには、対応する AWS パブリック証明書を使用する必要があります。

オプション

- オプション 1: PKCS7 署名を使用してインスタンスアイデンティティドキュメントを検証する
- オプション 2: base64 エンコード署名を使用してインスタンスアイデンティティドキュメントを検証する
- オプション 3: RSA-2048 署名を使用してインスタンスアイデンティティドキュメントを検証する

オプション 1: PKCS7 署名を使用してインスタンスアイデンティティドキュメントを検証する

このトピックでは、PKCS7 署名と AWS DSA パブリック証明書を使用して、インスタンスアイデンティティドキュメントを検証する方法について説明します。

Linux インスタンス

PKCS7 署名と AWS DSA パブリック証明書を使用してインスタンスアイデンティティドキュメントを検証するには

1. インスタンスに接続します。
2. インスタンスメタデータから PKCS7 署名を取得し、必要なヘッダーとフッターとともにpkcs7という名前の新しいファイルに追加します。インスタンスで使用されている IMDS のバージョンに応じて、次のいずれかのコマンドを使用します。

   IMDSv2

```
$ echo "-----BEGIN PKCS7-----" >> pkcs7 \
 && TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
 && curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/dynamic/instance-identity/pkcs7 >> pkcs7 \
 && echo "" >> pkcs7 \
 && echo "-----END PKCS7-----" >> pkcs7
```

IMDSv1

```
$ echo "-----BEGIN PKCS7-----" >> pkcs7 \
 && curl -s http://169.254.169.254/latest/dynamic/instance-identity/pkcs7
 >> pkcs7 \
 && echo "" >> pkcs7 \
 && echo "-----END PKCS7-----" >> pkcs7
```

3. インスタンスアイデンティティドキュメント署名の AWS パブリック証明書でリージョン用に DSA パブリック証明書を検索し、certificate という名前の新しいファイルに追加します。
4. OpenSSL の smime コマンドを使用して、署名を検証します。署名を検証する必要があることを示す -verify オプションと証明書を検証する必要がないことを示す -noverify オプションを含めます。

```
$ openssl smime -verify -in pkcs7 -inform PEM -certfile certificate -noverify | tee
 document
```

署名が有効な場合は、Verification successful メッセージが表示されます。

また、このコマンドでは、インスタンスアイデンティティドキュメントの内容を、document という名前の新しいファイルにも書き込みます。以下のコマンドにより、このファイルの内容を、インスタンスメタデータからのインスタンスアイデンティティドキュメントの内容と比較できます。

```
$ openssl dgst -sha256 < document
```

```
$ curl -s -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/
dynamic/instance-identity/document | openssl dgst -sha256
```

署名を検証できない場合は、サポートにお問い合わせください。

Windows インスタンス

前提条件

この手順では、Microsoft .NET Core の System.Security クラスが必要です。このクラスを PowerShell セッションに追加するには、次のコマンドを実行します。

```
PS C:\> Add-Type -AssemblyName System.Security
```

> **Note**
>
> このコマンドは、現在の PowerShell セッションにのみクラスを追加します。別のセッションを開始する場合は、このコマンドをもう一度実行する必要があります。

PKCS7 署名と AWS DSA パブリック証明書を使用してインスタンスアイデンティティドキュメントを検証するには

1. インスタンスに接続します。
2. インスタンスメタデータから PKCS7 署名を取得し、バイトの配列に変換して、$Signature という名前の変数に追加します。インスタンスで使用されている IMDS のバージョンに応じて、次のいずれかのコマンドを使用します。

   IMDSv2

   ```
   PS C:\> [string]$token = (Invoke-WebRequest -Method Put -Headers @{'X-aws-ec2-
   metadata-token-ttl-seconds' = '21600'} -Uri http://169.254.169.254/latest/api/
   token).Content
   ```

   ```
   PS C:\> $Signature = [Convert]::FromBase64String((Invoke-WebRequest -Headers
    @{'X-aws-ec2-metadata-token' = $Token} -Uri http://169.254.169.254/latest/
   dynamic/instance-identity/pkcs7).Content)
   ```

   IMDSv1

   ```
   PS C:\> $Signature = [Convert]::FromBase64String((Invoke-WebRequest -Uri
    http://169.254.169.254/latest/dynamic/instance-identity/pkcs7).Content)
   ```

3. インスタンスメタデータからプレーンテキストのインスタンスアイデンティティドキュメントを取得し、バイトの配列に変換して、$Document という名前の変数に追加します。インスタンスで使用されている IMDS のバージョンに応じて、次のいずれかのコマンドを使用します。

IMDSv2

```
PS C:\> $Document = [Text.Encoding]::UTF8.GetBytes((Invoke-WebRequest -Headers
 @{'X-aws-ec2-metadata-token' = $Token} -Uri http://169.254.169.254/latest/
dynamic/instance-identity/document).Content)
```

IMDSv1

```
PS C:\> $Document =  [Text.Encoding]::UTF8.GetBytes((Invoke-WebRequest
 http://169.254.169.254/latest/dynamic/instance-identity/document).Content)
```

4. インスタンスアイデンティティドキュメント署名の AWS パブリック証明書でリージョン用に DSA パブリック証明書を検索し、certificate.pem という名前の新しいファイルに追加します。
5. 証明書ファイルから証明書を抽出し、$Store という名前の変数に格納します。

```
PS C:\> $Store =
 [Security.Cryptography.X509Certificates.X509Certificate2Collection]::new([Security.Cryptog
Path certificate.pem)))
```

6. 署名を検証します。

```
PS C:\> $SignatureDocument = [Security.Cryptography.Pkcs.SignedCms]::new()
```

```
PS C:\> $SignatureDocument.Decode($Signature)
```

```
PS C:\> $SignatureDocument.CheckSignature($Store, $true)
```

署名が有効な場合、このコマンドは出力を返しません。署名を検証できない場合、このコマンドは Exception calling "CheckSignature" with "2" argument(s): "Cannot find the original signer を返します。署名を検証できない場合は、AWS サポートにお問い合わせください。

7. インスタンスアイデンティティドキュメントの内容を検証します。

```
PS C:
\> [Linq.Enumerable]::SequenceEqual($SignatureDocument.ContentInfo.Content, $Document)
```

インスタンスアイデンティティドキュメントの内容が有効な場合、このコマンドは True を返します。インスタンスアイデンティティドキュメントを検証できない場合は、AWS サポート にお問い合わせください。

オプション 2: base64 エンコード署名を使用してインスタンスアイデンティティドキュメントを検証する

このトピックでは、base64 でエンコードされた署名と AWS RSA パブリック証明書を使用して、インスタンスアイデンティティドキュメントを検証する方法について説明します。

Linux インスタンス

base64 でエンコードされた署名と AWS RSA パブリック証明書を使用してインスタンスアイデンティティドキュメントを検証するには

1. インスタンスに接続します。
2. インスタンスメタデータから base64 でエンコードされた署名を取得し、バイナリに変換して、signature という名前のファイルに追加します。インスタンスで使用されている IMDS のバージョンに応じて、次のいずれかのコマンドを使用します。

   IMDSv2

   ```
   $ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
    && curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/dynamic/instance-identity/signature | base64 -d >> signature
   ```

   IMDSv1

   ```
   $ curl -s http://169.254.169.254/latest/dynamic/instance-identity/signature | base64 -d >> signature
   ```

3. インスタンスメタデータからプレーンテキストのインスタンスアイデンティティドキュメントを取得し、document という名前のファイルに追加します。インスタンスで使用されている IMDS のバージョンに応じて、次のいずれかのコマンドを使用します。

IMDSv2

```
$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-
metadata-token-ttl-seconds: 21600"` \
 && curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/
dynamic/instance-identity/document >> document
```

IMDSv1

```
$ curl -s http://169.254.169.254/latest/dynamic/instance-identity/document
 >> document
```

4. インスタンスアイデンティティドキュメント署名の AWS パブリック証明書でリージョン用に RSA パブリック証明書を検索し、certificate という名前の新しいファイルに追加します。
5. AWS RSA パブリック証明書からパブリックキーを抽出し、key という名前のファイルに保存します。

```
$ openssl x509 -pubkey -noout -in certificate >> key
```

6. OpenSSL の dgst コマンドを使用して、インスタンスアイデンティティドキュメントを検証します。

```
$ openssl dgst -sha256 -verify key -signature signature document
```

署名が有効な場合は、Verification successful メッセージが表示されます。

また、このコマンドでは、インスタンスアイデンティティドキュメントの内容を、document という名前の新しいファイルにも書き込みます。以下のコマンドにより、このファイルの内容を、インスタンスメタデータからのインスタンスアイデンティティドキュメントの内容と比較できます。

```
$ openssl dgst -sha256 < document
```

```
$ curl -s -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/
dynamic/instance-identity/document | openssl dgst -sha256
```

署名を検証できない場合は、サポートにお問い合わせください。

Windows インスタンス

base64 でエンコードされた署名と AWS RSA パブリック証明書を使用してインスタンスアイデンティティドキュメントを検証するには

1. インスタンスに接続します。
2. インスタンスメタデータから base64 でエンコードされた署名を取得し、バイトの配列に変換して、$Signature という名前の変数に追加します。インスタンスで使用されている IMDS のバージョンに応じて、次のいずれかのコマンドを使用します。

   IMDSv2

   ```
   PS C:\> [string]$token = (Invoke-WebRequest -Method Put -Headers @{'X-aws-
   ec2-metadata-token-ttl-seconds' = '21600'} http://169.254.169.254/latest/api/
   token).Content
   ```

   ```
   PS C:\> $Signature = [Convert]::FromBase64String((Invoke-WebRequest -Headers
    @{'X-aws-ec2-metadata-token' = $Token} http://169.254.169.254/latest/dynamic/
   instance-identity/signature).Content)
   ```

   IMDSv1

   ```
   PS C:\> $Signature = [Convert]::FromBase64String((Invoke-WebRequest
    http://169.254.169.254/latest/dynamic/instance-identity/signature).Content)
   ```

3. インスタンスメタデータからプレーンテキストのインスタンスアイデンティティドキュメントを取得し、バイトの配列に変換して、$Document という名前の変数に追加します。インスタンスで使用されている IMDS のバージョンに応じて、次のいずれかのコマンドを使用します。

   IMDSv2

   ```
   PS C:\> $Document = [Text.Encoding]::UTF8.GetBytes((Invoke-WebRequest -Headers
    @{'X-aws-ec2-metadata-token' = $Token} http://169.254.169.254/latest/dynamic/
   instance-identity/document).Content)
   ```

   IMDSv1

   ```
   PS C:\> $Document =  [Text.Encoding]::UTF8.GetBytes((Invoke-WebRequest
    http://169.254.169.254/latest/dynamic/instance-identity/document).Content)
   ```

4. インスタンスアイデンティティドキュメント署名の AWS パブリック証明書でリージョン用に RSA パブリック証明書を検索し、certificate.pem という名前の新しいファイルに追加します。
5. インスタンスアイデンティティドキュメントを検証します。

```
PS C:\> [Security.Cryptography.X509Certificates.X509Certificate2]::new((Resolve-
Path certificate.pem)).PublicKey.Key.VerifyData($Document, 'SHA256', $Signature)
```

署名が有効な場合、このコマンドは True を返します。署名を検証できない場合は、サポートにお問い合わせください。

オプション 3: RSA-2048 署名を使用してインスタンスアイデンティティドキュメントを検証する

このトピックでは、RSA-2048 署名と AWS RSA-2048 パブリック証明書を使用して、インスタンスアイデンティティドキュメントを検証する方法について説明します。

Linux インスタンス

RSA-2048 署名と AWS RSA-2048 パブリック証明書を使用してインスタンスアイデンティティドキュメントを検証するには

1. インスタンスに接続します。
2. インスタンスメタデータから RSA-2048 署名を取得し、必要なヘッダーとフッターとともにrsa2048 という名前のファイルに追加します。インスタンスで使用されている IMDS のバージョンに応じて、次のいずれかのコマンドを使用します。

   IMDSv2

```
$ echo "-----BEGIN PKCS7-----" >> rsa2048 \
 && TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-
metadata-token-ttl-seconds: 21600"` \
 && curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/
dynamic/instance-identity/rsa2048 >> rsa2048 \
 && echo "" >> rsa2048 \
 && echo "-----END PKCS7-----" >> rsa2048
```

IMDSv1

```
$ echo "-----BEGIN PKCS7-----" >> rsa2048 \
 && curl -s http://169.254.169.254/latest/dynamic/instance-identity/rsa2048
 >> rsa2048 \
 && echo "" >> rsa2048 \
 && echo "-----END PKCS7-----" >> rsa2048
```

3. インスタンスアイデンティティドキュメント署名の AWS パブリック証明書でリージョン用に RSA-2048 パブリック証明書を検索し、certificate という名前の新しいファイルに追加します。
4. OpenSSL の smime コマンドを使用して、署名を検証します。署名を検証する必要があることを示す -verify オプションと証明書を検証する必要がないことを示す -noverify オプションを含めます。

```
$ openssl smime -verify -in rsa2048 -inform PEM -certfile certificate -noverify |
 tee document
```

署名が有効な場合は、Verification successful メッセージが表示されます。署名を検証できない場合は、サポートにお問い合わせください。

Windows インスタンス

前提条件

この手順では、Microsoft .NET Core の System.Security クラスが必要です。このクラスを PowerShell セッションに追加するには、次のコマンドを実行します。

```
PS C:\> Add-Type -AssemblyName System.Security
```

> **Note**
>
> このコマンドは、現在の PowerShell セッションにのみクラスを追加します。別のセッションを開始する場合は、このコマンドをもう一度実行する必要があります。

RSA-2048 署名と AWS RSA-2048 パブリック証明書を使用してインスタンスアイデンティティドキュメントを検証するには

1. インスタンスに接続します。
2. インスタンスメタデータから RSA-2048 署名を取得し、バイトの配列に変換して、$Signature という名前の変数に追加します。インスタンスで使用されている IMDS のバージョンに応じて、次のいずれかのコマンドを使用します。

   IMDSv2

   ```
   PS C:\> [string]$token = (Invoke-WebRequest -Method Put -Headers @{'X-aws-
   ec2-metadata-token-ttl-seconds' = '21600'} http://169.254.169.254/latest/api/
   token).Content
   ```

   ```
   PS C:\> $Signature = [Convert]::FromBase64String((Invoke-WebRequest -Headers
    @{'X-aws-ec2-metadata-token' = $Token} http://169.254.169.254/latest/dynamic/
   instance-identity/rsa2048).Content)
   ```

   IMDSv1

   ```
   PS C:\> $Signature = [Convert]::FromBase64String((Invoke-WebRequest
    http://169.254.169.254/latest/dynamic/instance-identity/rsa2048).Content)
   ```

3. インスタンスメタデータからプレーンテキストのインスタンスアイデンティティドキュメントを取得し、バイトの配列に変換して、$Document という名前の変数に追加します。インスタンスで使用されている IMDS のバージョンに応じて、次のいずれかのコマンドを使用します。

   IMDSv2

   ```
   PS C:\> $Document = [Text.Encoding]::UTF8.GetBytes((Invoke-WebRequest -Headers
    @{'X-aws-ec2-metadata-token' = $Token} http://169.254.169.254/latest/dynamic/
   instance-identity/document).Content)
   ```

   IMDSv1

   ```
   PS C:\> $Document =  [Text.Encoding]::UTF8.GetBytes((Invoke-WebRequest
    http://169.254.169.254/latest/dynamic/instance-identity/document).Content)
   ```

4. インスタンスアイデンティティドキュメント署名の AWS パブリック証明書でリージョン用に RSA-2048 パブリック証明書を検索し、certificate.pem という名前の新しいファイルに追加します。
5. 証明書ファイルから証明書を抽出し、$Store という名前の変数に格納します。

```
PS C:\> $Store =
 [Security.Cryptography.X509Certificates.X509Certificate2Collection]::new([Security.Cryptog
Path certificate.pem)))
```

6. 署名を検証します。

```
PS C:\> $SignatureDocument = [Security.Cryptography.Pkcs.SignedCms]::new()
```

```
PS C:\> $SignatureDocument.Decode($Signature)
```

```
PS C:\> $SignatureDocument.CheckSignature($Store, $true)
```

署名が有効な場合、このコマンドは出力を返しません。署名を検証できない場合、このコマンドは Exception calling "CheckSignature" with "2" argument(s): "Cannot find the original signer を返します。署名を検証できない場合は、AWS サポートにお問い合わせください。

7. インスタンスアイデンティティドキュメントの内容を検証します。

```
PS C:
\> [Linq.Enumerable]::SequenceEqual($SignatureDocument.ContentInfo.Content, $Document)
```

インスタンスアイデンティティドキュメントの内容が有効な場合、このコマンドは True を返します。インスタンスアイデンティティドキュメントを検証できない場合は、AWS サポート にお問い合わせください。

インスタンスアイデンティティドキュメント署名の AWS パブリック証明書

「インスタンスアイデンティティドキュメントを検証する」で説明するように、AWS パブリック証明書を使用して、インスタンスのインスタンスアイデンティティドキュメントの内容を検証できます。

リージョンと使用している検証手順に適した証明書を使用していることを確認してください。PKCS7 の署名を検証する場合は、DSA 証明書を使用してください。base64 でエンコードされた署名を検証する場合は、RSA 証明書を使用してください。RSA-2048 署名を検証する場合は、RSA-2048 証明書を使用してください。

以下の各リージョンを展開すると、リージョン固有の証明書が表示されます。

米国東部 (バージニア北部) - us-east-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUE1y2NIKCU+Rg4uu4u32koG9QEYIwDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI0MDQyOTE3MzQwMVoXDTI5MDQyODE3MzQwMVowXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQCHvRjf/0kStpJ248khtIaN8qkDN3tkw4VjvA9nvPl2anJO+eIB
UqPfQG09kZlwpWpmyO8bGB2RWqWxCwuB/dcnIob6w420k9WY5C0IIGtDRNauN3ku
vGXkw3HEnF0EjYr0pcyWUvByWY4KswZV42X7Y7XSS13hOIcL6NLA+H94/QIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUJdbMCBXKtvCcWdwUUizvtUF2
```

```
UTgwgZkGA1UdIwSBkTCBjoAUJdbMCBXKtvCcWdwUUizvtUF2UTihYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUE1y2NIKC
U+Rg4uu4u32koG9QEYIwEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQAlxSmwcWnhT4uAeSinJuz+1BTcKhVSWb5jT8pYjQb8ZoZkXXRGb09mvYeU
NeqOBr27rvRAnaQ/9LUQf72+SahDFuS4CMI8nwowytqbmwquqFr4dxA/SDADyRiF
ea1UoMuNHTY49J/1vPomqsVn7mugTp+TbjqCfOJTpu0temHcFA==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJALFpzEAVWaQZMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTA4MTQw
ODU5MTJaGA8yMTk1MDExNzA4NTkxMlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAjS2vqZu9mEOhOq+0bRpAbCUiapbZMFNQqRg7kTlr7Cf+gDqXKpHPjsng
SfNz+JHQd8WPI+pmNs+q0Z2aTe23klmf2U52KH9/j1k8RlIbap/yFibFTSedmegX
E5r447GbJRsHUmuIIfZTZ/oRlpuIIO5/Vz7SOj22tdkdY2ADp7caZkNxhSP915fk
2jJMTBUOzyXUS2rBU/ulNHbTTeePjcEkvzVYPahD30TeQ+/A+uWUu89bHSQOJR8h
Um4cFApzZgN3aD5j2LrSMu2pctkQwf9CaWyVznqrsGYjYOY66LuFzSCXwqSnFBfv
fFBAFsjCgY24G2DoMyYkF3MyZlu+rwIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQUrynSPp4uqSECwy+PiO4qyJ8TWSkwgY4GA1UdIwSBhjCBg4AUrynS
Pp4uqSECwy+PiO4qyJ8TWSmhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJALFpzEAVWaQZMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBADW/s8lXijwdP6NkEoH1m9XLrvK4YTqkNfR6
er/uRRgTx2QjFcMNrx+g87gAml11z+D0crAZ5LbEhDMs+JtZYR3ty0HkDk6SJM85
haoJNAFF7EQ/zCp1EJRIkLLsC7bcDL/Eriv1swt78/BB4RnC9W9kSp/sxd5svJMg
N9a6FAplpNRsWAnbP8JBlAP93oJzblX2LQXgykTghMkQO7NaY5hg/H5o4dMPclTK
lYGqlFUCH6A2vdrxmpKDLmTn5//5pujdD2MN0df6sZWtxwZ0osljV4rDjm9Q3VpA
NWIsDEcp3GUB4proOR+C7PNkY+VGODitBOw09qBGosCBstwyEqY=
-----END CERTIFICATE-----
```

米国東部 (オハイオ) - us-east-2

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
```

```
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUVJTc+hOU+8Gk3JlqsX438Dk5c58wDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI0MDQyOTE3MTE0OVoXDTI5MDQyODE3MTE0OVowXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQCHvRjf/0kStpJ248khtIaN8qkDN3tkw4VjvA9nvPl2anJO+eIB
UqPfQG09kZlwpWpmyO8bGB2RWqWxCwuB/dcnIob6w420k9WY5C0IIGtDRNauN3ku
vGXkw3HEnF0EjYr0pcyWUvByWY4KswZV42X7Y7XSS13hOIcL6NLA+H94/QIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUJdbMCBXKtvCcWdwUUizvtUF2
UTgwgZkGA1UdIwSBkTCBjoAUJdbMCBXKtvCcWdwUUizvtUF2UTihYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUVJTc+hOU
+8Gk3JlqsX438Dk5c58wEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQAywJQaVNWJqW0R0T0xVOSoN1GLk9x9kKEuN67RN9CLin4dA97qa7Mr5W4P
FZ6vnh5CjOhQBRXV9xJUeYSdqVItNAUFK/fEzDdjf1nUfPlQ3OJ49u6CV01NoJ9m
usvY9kWcV46dqn2bk2MyfTTgvmeqP8fiMRPxxnVRkSzlldP5Fg==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAM07oeX4xevdMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
```

```
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNjA2MTAx
MjU4MThaGA8yMTk1MTExNDEyNTgxOFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEA6v6kGMnRmFDLxBEqXzP4npnL65OO0kmQ7w8YXQygSdmNIoScGSU5wfh9
mZdcvCxCdxgALFsFqPvH8fqiE9ttI0fEfuZvHOs8wUsIdKr0Zz0MjSx3cik4tKET
ch0EKfMnzKOgDBavraCDeX1rUDU0Rg7HFqNAOry3uqDmnqtk00XC9GenS3z/7ebJ
fIBEPAam5oYMVFpX6M6St77WdNE8wEU8SuerQughiMVx9kMB07imeVHBiELbMQ0N
lwSWRL/61fA02keGSTfSp/0m3u+lesf2VwVFhqIJs+JbsEscPxOkIRlzy8mGd/JV
ONb/DQpTedzUKLgXbw7KtO3HTG9iXQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQU2CTGYE5fTjx7gQXzdZSGPEWAJY4wgY4GA1UdIwSBhjCBg4AU2CTG
YE5fTjx7gQXzdZSGPEWAJY6hYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAM07oeX4xevdMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBANdqkIpVypr2PveqUsAKke1wKCOSuw1UmH9k
xX1/VRoHbrI/UznrXtPQOPMmHA2LKSTedwsJuorUn3cFH6qNs8ixBDrl8pZwfKOY
IBJcTFBbI1xBEFkZoO3wczzo5+8vPQ60RVqAaYb+iCa1HFJpccC3Ovajfa4GRdNb
n6FYnluIcDbmpcQePoVQwX7W3oOYLB1QLN7fE6H1j4TBIsFdO3OuKzmaifQlwLYt
DVxVCNDabpOr6Uozd5ASm4ihPPoEoKo7Ilp0fOT6fZ41U2xWA4+HF/89UoygZSo7
K+cQ90xGxJ+gmlYbLFR5rbJOLfjrgDAb2ogbFy8LzHo2ZtSe60M=
-----END CERTIFICATE-----
```

米国西部 (北カリフォルニア) - us-west-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
```

```
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUK2zmY9PUSTR7rc1k2OwPYu4+g7wwDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI0MDQyOTE3MDI0M1oXDTI5MDQyODE3MDI0M1owXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQCHvRjf/0kStpJ248khtIaN8qkDN3tkw4VjvA9nvPl2anJO+eIB
UqPfQG09kZlwpWpmyO8bGB2RWqWxCwuB/dcnIob6w420k9WY5C0IIGtDRNauN3ku
vGXkw3HEnF0EjYr0pcyWUvByWY4KswZV42X7Y7XSS13hOIcL6NLA+H94/QIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUJdbMCBXKtvCcWdwUUizvtUF2
UTgwgZkGA1UdIwSBkTCBjoAUJdbMCBXKtvCcWdwUUizvtUF2UTihYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUK2zmY9PU
STR7rc1k2OwPYu4+g7wwEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQA1Ng4QmN4n7iPh5CnadSOc0ZfM7by0dBePwZJyGvOHdaw6P6E/vEk76KsC
Q8p+akuzVzVPkU4kBK/TRqLp19wEWoVwhhTaxHjQ1tTRHqXIVlrkw4JrtFbeNM21
GlkSLonuzmNZdivn9WuQYeGe7nUD4w3q9GgiF3CPorJe+UxtbA==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJANNPkIpcyEtIMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTEwMjkw
OTAzMDdaGA8yMTk1MDQwMzA5MDMwN1owXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEApHQGvHvq3SVCzDrC7575BW7GWLzcj8CLqYcL3YY7JffupZ7OjcftO57Z
4fo5Pj0CaS8DtPzh8+8vdwUSMbiJ6cDd3ooio3MnCq6DwzmsY+pY7CiI3UVG7KcH
4TriDqr1Iii7nB5MiPJ8wTeAqX89T3SYaf6Vo+4GCb3LCDGvnkZ9TrGcz2CHkJsj
AIGwgopFpwhIjVYm7obmuIxSIUv+oNH0wXgDL029Zd98SnIYQd/njiqkzE+lvXgk
4h4Tu17xZIKBgFcTtWPky+POGu81DYFqiWVEyR2JKKm2/iR1dL1YsT39kbNg47xY
aR129sS4nB5Vw3TRQA2jL0ToTIxzhQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQUgepyiONs8j+q67dmcWu+mKKDa+gwgY4GA1UdIwSBhjCBg4AUgepy
iONs8j+q67dmcWu+mKKDa+ihYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
```

```
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJANNPkIpcyEtIMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAGLFWyutf1u0xcAc+kmnMPqtc/Q6b79VIX0E
tNoKMI2KR8lcV8ZElXDb0NC6v8UeLpe1WBKjaWQtEjL1ifKg9hdY9RJj4RXIDSK7
33qCQ8juF4vep2U5TTBd6hfWxt1Izi88xudjixmbpUU4YKr8UPbmixldYR+BEx0u
B1KJi9l1lxvuc/Igy/xeHOAZEjAXzVvHp8Bne33VVwMiMxWECZCiJxE4I7+Y6fqJ
pLLSFFJKbNaFyXlDiJ3kXyePEZSc1xiWeyRB2ZbTi5eu7vMG4i3AYWuFVLthaBgu
lPfHafJpj/JDcqt2vKUKfur5edQ6j1CGdxqqjawhOTEqcN8m7us=
-----END CERTIFICATE-----
```

米国西部 (オレゴン) - us-west-2

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUFx8PxCkbHwpD31bOyCtyz3GclbgwDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI0MDQyOTE3MjM1OVoXDTI5MDQyODE3MjM1OVowXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
```

```
A4GNADCBiQKBgQCHvRjf/0kStpJ248khtIaN8qkDN3tkw4VjvA9nvPl2anJO+eIB
UqPfQG09kZlwpWpmyO8bGB2RWqWxCwuB/dcnIob6w420k9WY5C0IIGtDRNauN3ku
vGXkw3HEnF0EjYr0pcyWUvByWY4KswZV42X7Y7XSS13hOIcL6NLA+H94/QIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUJdbMCBXKtvCcWdwUUizvtUF2
UTgwgZkGA1UdIwSBkTCBjoAUJdbMCBXKtvCcWdwUUizvtUF2UTihYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUFx8PxCkb
HwpD31bOyCtyz3GclbgwEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQBzOl+9Xy1+UsbUBI95HO9mbbdnuX+aMJXgG9uFZNjgNEbMcvx+h8P9IMko
z7PzFdheQQ1NLjsHH9mSR1SyC4m9ja6BsejH5nLBWyCdjfdP3muZM4O5+r7vUa1O
dWU+hP/T7DUrPAIVMOE7mpYa+WPWJrN6BlRwQkKQ7twm9kDalA==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJALZL3lrQCSTMMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTA4MTQw
OTAxMzJaGA8yMTk1MDExNzA5MDEzMlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEA02Y59qtAA0a6uzo7nEQcnJ26OKF+LRPwZfixBH+EbEN/Fx0gYy1jpjCP
s5+VRNg6/WbfqAsV6X2VSjUKN59ZMnMY9ALA/Ipz0n00Huxj38EBZmX/NdNqKm7C
qWu1q5kmIvYjKGiadfboU8wLwLcHo8ywvfgI6FiGGsEO9VMC56E/hL6Cohko11LW
dizyvRcvg/IidazVkJQCN/4zC9PUOVyKdhW33jXy8BTg/QH927QuNk+ZzD7HH//y
tIYxDhR6TIZsSnRjz3bOcEHxt1nsidc65mY0ejQty4hy7ioSiapw316mdbtE+RTN
fcH9FPIFKQNBpiqfAW5Ebp3Lal3/+wIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQU7coQx8Qnd75qA9XotSWT3IhvJmowgY4GA1UdIwSBhjCBg4AU7coQ
x8Qnd75qA9XotSWT3IhvJmqhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJALZL3lrQCSTMMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAFZ1e2MnzRaXCaLwEC1pW/f0oRG8nHrlPZ9W
OYZEWbh+QanRgaikBNDtVTwARQcZm3z+HWSkaIx3cyb6vM0DSkZuiwzm1LJ9rDPc
aBm03SEt5v8mcc7sXWvgFjCnUpzosmky6JheCD4O1Cf8k0olZ93FQnTrbg62OK0h
83mGCDeVKU3hLH97FYoUq+3N/IliWFDhvibAYYKFJydZLhIdlCiiB99AM6Sg53rm
oukS3csyUxZyTU2hQfdjyo1nqW9yhvFAKjnnggiwxNKTTPZzstKW8+cnYwiiTwJN
QpVoZdt0SfbuNnmwRUMi+QbuccXweav29QeQ3ADqjgB0CZdSRKk=
-----END CERTIFICATE-----
```

アフリカ (ケープタウン) – af-south-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7DCCAqwCCQCncbCtQbjuyzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xOTA2MDQxMjQ4MDVaFw00
NTA2MDQxMjQ4MDVaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbYwggErBgcqhkjOOAQBMIIBHgKBgQC12Nr1gMrHcFSZ7S/A
pQBSCMHWmn2qeoQTMVWqe50fnTd0zGFxDdIjKxUK58/8zjWG5uR4TXRzmZpGpmXB
bSufAR6BGqud2LnT/HIWGJAsnX2uOtSyNfCoJigqwhea5w+CqZ6I7iBDdnB4TtTw
qO6TlnExHFVj8LMkylZgiaE1CQIVAIhdobse4K0QnbAhCL6R2euQzloXAoGAV/21
WUuMz/79Ga0JvQcz1FNy1sT0pU9rU4TenqLQIt5iccn/7EIfNtvVO5TZKulIKq7J
gXZr0x/KIT8zsNweetLOaGehPIYRMPX0vunMMR7hN7qA7W17WZv/76adywIsnDKq
ekfe15jinaX8MsKUdyDK7Y+ifCG4PVhoM4+W2XwDgYQAAoGAIxOKbVgwLxbn6Pi2
6hBOihFv16jKxAQI0hHzXJLV0Vyv9QwnqjJJRfOCy3dB0zicLXiIxeIdYfvqJr+u
hlN8rGxEZYYJjEUKMGvsc0DW85jonXz0bNfcP0aaKH0lKKVjL+OZi5n2kn9wgdo5
F3CVnMl8BUra8A1Tr2yrrE6TVZ4wCQYHKoZIzjgEAwMvADAsAhQfa7MCJZ+/TEY5
AUr0J4wm8VzjoAIUSYZVu2NdRJ/ERPmDfhW5EsjHlCA=
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICNjCCAZ+gAwIBAgIJAKumfZiRrNvHMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xOTExMjcw
NzE0MDVaGA8yMTk5MDUwMjA3MTQwNVowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKB
gQDFd571nUzVtke3rPyRkYfvs3jh0C0EMzzG72boyUNjnfw1+m0TeFraTLKb9T6F
7TuB/ZEN+vmlYqr2+5Va8U8qLbPF0bRH+FdaKjhgWZdYXxGzQzU3ioy5W5ZM1VyB
7iUsxEAlxsybC3ziPYaHI42UiTkQNahmoroNeqVyHNnBpQIDAQABMA0GCSqGSIb3
DQEBCwUAA4GBAAJLylWyElEgOpW4B1XPyRVD4pAds8Guw2+krgqkY0HxLCdjosuH
RytGDGN+q75aAoXzW5a7SGpxLxk6Hfv0xp3RjDHsoeP0i1d8MD3hAC5ezxS4oukK
s5gbPOnokhKTMPXbTdRn5ZifCbWlx+bYN/mTYKvxho7b5SVg2o1La9aK
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
```

```
MIIDOzCCAiOgAwIBAgIJAIFI+O5A6/ZIMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xOTA2MDQx
MjQ4MDRaGA8yMTk4MTEwNzEyNDgwNFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAy7/WHBBHOrk+20aumT07g8rxrSM0UXgki3eYgKauPCG4Xx//vwQbuZwI
oeVmR9nqnhfij2wOcQdbLandh0EGtbxerete3IoXzd1KXJb11PVmzrzyu5SPBPuP
iCeV4qdjjkXo2YWM6t9YQ911hcG96YSp89TBXFYUh3KLxfqAdTVhuC0NRGhXpyii
j/czo9njofHhqhTr7UEyPun8NVS2QWctLQ86N5zWR3Q0GRoVqqMrJs0cowHTrVw2
9Qr7QBjjBOVbyYmtYxm/DtiKprYV/e6bCAVok015X1sZDd3oCOQNoGlv5XbHJe2o
JFD8GRRy2rkWO/lNwVFDcwec6zC3QwIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQCE
goqzjpCpmMgCpszFHwvRaSMbspKtK7wNImUjrSBOfBJsfFulyg1Zgn2nDCK7kQhx
jMJmNIvXbps3yMqQ2cHUkKcKf5t+WldfeT4Vk1Rz6HSA8sd0kgVcIesIaoy2aaXU
VEB/oQziRGyKdN1d4TGYVZXG44CkrzSDvlbmfiTq5tL+kAieznVF3bzHgPZW6hKP
EXC3G/IXrXicFEe6YyE1Rakl62VncYSXiGe/i2XvsiNH3Qlmnx5XS7W0SCN0oAxW
EH9twibauv82DVg1WOkQu8EwFw8hFde9X0Rkiu0qVcuU8lJgFEvPWMDFU5sGB6ZM
gkEKTzMvlZpPbBhg99Jl
-----END CERTIFICATE-----
```

アジアパシフィック (香港) – ap–east–1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7zCCAq4CCQCO7MJe5Y3VLjAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xOTAyMDMwMjIxMjFaFw00
NTAyMDMwMjIxMjFaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbgwggEsBgcqhkjOOAQBMIIBHwKBgQDvQ9RzVvf4MAwGbqfX
blCvCoVb9957OkLGn/04CowHXJ+vTBR7eyIa6AoXltsQXBOmrJswToFKKxT4gbuw
jK7s9QQX4CmTRWcEgO2RXtZSVjOhsUQMh+yf7Ht4OVL97LWnNfGsX2cwjcRWHYgI
7lvnuBNBzLQHdSEwMNq0Bk76PwIVAMan6XIEEPnwr4e6u/RNnWBGKd9FAoGBAOCG
eSNmxpW4QFu4pIlAykm6EnTZKKHT87gdXkAkfoC5fAfOxxhnE2HezZHp9Ap2tMV5
8bWNvoPHvoKCQqwfm+OUBlAxC/3vqoVkKL2mG1KgUH9+hrtpMTkwO3RREnKe7I5O
x9qDimJpOihrL4I0dYvy9xUOoz+DzFAW8+ylWVYpA4GFAAKBgQDbnBAKSxWr9QHY
6Dt+EFdGz6lAZLedeBKpaP53Z1DTO34J0C55YbJTwBTFGqPtOLxnUVDlGiD6GbmC
80f3jvogPR1mSmGsydbNbZnbUEVWrRhe+y5zJ3g9qs/DWmDW0deEFvkhWVnLJkFJ
9pdOu/ibRPH1lE2nz6pK7GbOQtLyHTAJBgcqhkjOOAQDAzAAMC0CFQCoJlwGtJQC
cLoM4p/jtVFOj26xbgIUUS4pDKyHaG/eaygLTtFpFJqzWHc=
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICSzCCAbQCCQDtQvkVxRvK9TANBgkqhkiG9w0BAQsFADBqMQswCQYDVQQGEwJV
UzETMBEGA1UECBMKV2FzaGluZ3RvbjEQMA4GA1UEBxMHU2VhdHRsZTEYMBYGA1UE
ChMPQW1hem9uLmNvbSBJbmMuMRowGAYDVQQDExFlYzIuYW1hem9uYXdzLmNvbTAe
Fw0xOTAyMDMwMzAwMDZaFw0yOTAyMDIwMzAwMDZaMGoxCzAJBgNVBAYTAlVTMRMw
EQYDVQQIEwpXYXNoaW5ndG9uMRAwDgYDVQQHEwdTZWF0dGxlMRgwFgYDVQQKEw9B
bWF6b24uY29tIEluYy4xGjAYBgNVBAMTEWVjMi5hbWF6b25hd3MuY29tMIGfMA0G
CSqGSIb3DQEBAQUAA4GNADCBiQKBgQC1kkHXYTfc7gY5Q55JJhjTieHAgacaQkiR
Pity9QPDE3b+NXDh4UdP1xdIw73JcIIG3sG9RhWiXVCHh6KkuCTqJfPUknIKk8vs
M3RXflUpBe8Pf+P92pxqPMCz1Fr2NehS3JhhpkCZVGxxwLC5gaG0Lr4rFORubjYY
Rh84dK98VwIDAQABMA0GCSqGSIb3DQEBCwUAA4GBAA6xV9f0HMqXjPHuGILDyaNN
dKcvplNFwDTydVg32MNubAGnecoEBtUPtxBsLoVYXCOb+b5/ZMDubPF9tU/vSXuo
TpYM5Bq57gJzDRaBOntQbX9bgHiUxw6XZWaTS/6xjRJDT5p3S1E0mPI3lP/eJv4o
Ezk5zb3eIf10/sqt4756
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJAMoxixvs3YssMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xODA3MjAw
ODQ0NDRaGA8yMTk3MTIyMzA4NDQ0NFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEA4T1PNsOg0FDrGlWePoHeOSmOJTA3HCRy5LSbYD33GFU2eBrOIxoU/+SM
rInKu3GghAMfH7WxPW3etIAZiyTDDU5RLcUq2Qwdr/ZpXAWpYocNc/CEmBFtfbxF
z4uwBIN3/drM0RSbe/wP9EcgmNUGQMMZWeAji8sMtwpOblNWAP9BniUG0Flcz6Dp
uPovwDTLdAYT3TyhzlohKL3f6O48TR5yTaV+3Ran2SGRhyJjfh3FRpP4VC+z5LnT
WPQHN74Kdq35UgrUxNhJraMGCzznolUuoR/tFMwR93401GsM9fVA7SW3jjCGF81z
PSzjy+ArKyQqIpLW1YGWDFk3sf08FQIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQDK
2/+C3nPMgtyOFX/I3Cyk+Pui44IgOwCsIdNGwuJysdqp5VIfnjegEu2zIMWJSKGO
lMZoQXjffkVZZ97J7RNDW06oB7kj3WVE8a7U4WEOfnO/CbMUf/x99CckNDwpjgW+
K8V8SzAsQDvYZs2KaE+18GFfLVF1TGUYK2rPSZMHyX+v/TIlc/qUceBycrIQ/kke
jDFsihUMLqgmOV2hXKUpIsmiWMGrFQV4AeV0iXP8L/ZhcepLf1t5SbsGdUA3AUY1
3If8s81uTheiQjwY5t9nM0SY/1Th/tL3+RaEI79VNEVfG1FQ8mgqCK0ar4m0oZJl
tmmEJM7xeURdpBBx36Di
-----END CERTIFICATE-----
```

アジアパシフィック (ハイデラバード) - ap-south-2

DSA

```
-----BEGIN CERTIFICATE-----
MIIC8DCCArCgAwIBAgIGAXjrQ4+XMAkGByqGSM44BAMwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgMEFdhc2hpbmd0b24g
U4EddRIpUt9KnC7s5Of2EbdSPO9EAMMeP4C2USZpRV1AIlH7WT2NWPq/
xfW6MPbLm1Vs14E7gB00b/JmYLdrmVClpJ+f6AR7ECLCT7up1/63xhv4O1fnxqimFQ8E
+4P208UewwI1VBNaFpEy9nXzrith1yrv8iIDGZ3RSAHHAhUAl2BQjxUjC8yykrmCouuEC/
BYHPUCgYEA9+GghdabPd7LvKtcNrhXuXmUr7v6OuqC+VdMCz0HgmdRWVeOutRZT
+ZxBxCBgLRJFnEj6EwoFhO3zwkyjMim4TwWeotUfI0o4KOuHiuzpnWRbqN/C/ohNWLx
+2J6ASQ7zKTxvqhRkImog9/
hWuWfBpKLZl6Ae1UlZAFMO/7PSSoDgYUAAoGBAJCKGBBoxIUxqBk94JHhwZZbgvbP0DAOoHENQWxp/98lI7/
YOfYJOVMJS22aCnHDurofmo5rvNIkgXi7Rztbhu
+lko9rK6DgpmpUwBU0WZtf34aZ2IWNBwHaVhHvWAQf9/46u18dMa2YucK1Wi+Vc+M
+KldrvgXmhym6ErNlzhJyMAkGByqGSM44BAMDLwAwLAIUaaPKxaOHoYvwz709xXpsQueIq+UCFFa/
GpzoDOSokl1057NU/2hnsiW4
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDCzCCAnSgAwIBAgIJAPw1mjt5QguhMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0yNjA1Mjgx
MDQzMjlaFw0zMTA1MjcxMDQzMjlaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQzCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEA
08B7tDQ/rBXKG6473AGJtDzUQ/H+EdYwM76C7dvqhnjkwMjtR7bY+nJrGDdwD9YT
eYmLqaRYmLJdnjT2vwg9FYNAQHwxZVea846QNI+0yWm1YeJI+yb7Q/sWDIusBEAO
MekjzjWODnmdvvKPx0vJ7nKVg1XT0TunIllJTIY5g5sCAwEAAaOB1DCB0TALBgNV
HQ8EBAMCB4AwHQYDVR0OBBYEFM4QY6Q5jpIb3pUP7EBEDUVlvGgsMIGOBgNVHSME
gYYwgYOAFM4QY6Q5jpIb3pUP7EBEDUVlvGgsoWCkXjBcMQswCQYDVQQGEwJVUzEZ
MBcGA1UECBMQV2FzaGluZ3RvbiBTdGF0ZTEQMA4GA1UEBxMHU2VhdHRsZTEgMB4G
A1UEChMXQW1hem9uIFdlYiBTZXJ2aWNlcyBMTEOCCQD8NZo7eUILoTASBgNVHRMB
Af8ECDAGAQH/AgEAMA0GCSqGSIb3DQEBCwUAA4GBAGPVdLHdjH2PVcfoK41bRQCp
vzwwOB0XpQ1AqQBYZXmm6+Ggc3nQLoOKLG1oRlc38psWtxBjGRL+ktvyAjgwk+BL
xM7Yx+wDf8m2nfrvIdaMHk0i+U7bWaiFod3cD6XICfwg/tFz50auWTtlIPcNOXAU
qPuHbYlFgMUlNuNp78nB
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAIVWfPw/X82fMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yMjA3MDQx
NDMwMjhaGA8yMjAxMTIwODE0MzAyOFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAg29QEFriG+qFEjYW/v62nN70lMJY/Hevx5TtmU/VIYBPQa3HUGTBAbbI
2Tmy8UMpa8kZeaYeI3RAfiQWt0Ws7wUrBuO2Pdp518WDPaJUH7RWEuu1BDDkyZRW
NAMNPCn3ph7Od243IFcLGku7HVekel5poqRpSfojrMasjlf+CvixUeAJbmFoxUHK
kh5unzG2sZyO4wHXcJPQkRf5a8zSTPe9YZP1kXPPEv4p/jTSggaYPxXyS6QVaTlV
zLeLFZ0fesLPMeil3KYQtV7IKLQiEA2F6dxWnxNWQlyMHtdq6PucfEmVx17i/Xza
yNBRo0azY8WUNVkEXrRhp/pU8Nh3GQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQU9AO1aZk9RLXk2ZvRVoUxYvQy9uwwgY4GA1UdIwSBhjCBg4AU9AO1
aZk9RLXk2ZvRVoUxYvQy9uyhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAIVWfPw/X82fMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBADexluMRQRftqViahCnauEWGdMvLCBr8A+Yr
6hJq0guoxEk/lahxR137DnfMPuSbi1Rx5QKo7oBrWfG/zsgQUnF2IwHTzwD+i/2m
XCane6FiS5RpK3lGdILq8ZmlhQk+6iI8yoZLr0LCfTh+CLgIKHOknfR5lFzgzAiF
SI8/Q9mm+uvYtSTZECI6Zh57QZPoETAG/yl+9jiOy2lAelqa/k1i+Qo8gMfOc+Pm
dwY7o6fV+oucgRlsdey6VM45LeyILQqv0RXtVzjuowanzmCCFMjgqiO9oZAWu4Oh
+F3unijELo01vZJs8s2N3KGlo3/jtUFTX6RTKShZlAPLwBi5GMI=
-----END CERTIFICATE-----
```

アジアパシフィック (ジャカルタ) - ap-southeast-3

DSA

```
-----BEGIN CERTIFICATE-----
MIIC8DCCArCgAwIBAgIGAXbVDEikMAkGByqGSM44BAMwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgMEFdhc2hpbmd0b24g
U4EddRIpUt9KnC7s5Of2EbdSPO9EAMMeP4C2USZpRV1AIlH7WT2NWPq/
xfW6MPbLm1Vs14E7gB00b/JmYLdrmVClpJ+f6AR7ECLCT7up1/63xhv4O1fnxqimFQ8E
+4P208UewwI1VBNaFpEy9nXzrith1yrv8iIDGZ3RSAHHAhUAl2BQjxUjC8yykrmCouuEC/
BYHPUCgYEA9+GghdabPd7LvKtcNrhXuXmUr7v6OuqC+VdMCz0HgmdRWVeOutRZT
+ZxBxCBgLRJFnEj6EwoFhO3zwkyjMim4TwWeotUfI0o4KOuHiuzpnWRbqN/C/ohNWLx
+2J6ASQ7zKTxvqhRkImog9/
hWuWfBpKLZl6Ae1UlZAFMO/7PSSoDgYUAAoGBAPjuiEx05N3JQ6cVwntJie67D8OuNo4jGRn
+crEtL7YO0jSVB9zGE1ga
+UgRPIaYETL293S8rTJTVgXAqdpBwfaHC6NUzre8U8iJ8FMNnlP9Gw1oUIlgQBjORyynVJexoB31TDZM
```

```
+/52g9O/bpq1QqNyKbeIgyBBlc1dAtr1QLnsMAkGByqGSM44BAMDLwAwLAIUK8E6RDIRtwK+9qnaTOBhvO/
njuQCFFocyT1OxK+UDR888oNsdgtif2Sf
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUMDbUg+reaqO28n34vFUuk6gFWbgwDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI2MDQyMjEyMDY0MFoXDTMxMDQyMTEyMDY0MFowXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQCnCS/Vbt0gQ1ebWcur2hSO7PnJifE4OPxQ7RgSAlc4/spJp1sD
P+ZrS0LO1ZJfKhXf1R9S3AUwLnsc7b+IuVXdY5LK9RKqu64nyXP5dx170zoL8loE
yCSuRR2fs+04i2QsWBVP+KFNAn7P5L1EHRjkgTO8kjNKviwRV+OkP9ab5wIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUsnsEPOUYPQO0Gv9t4arIC2NN
kHQwgZkGA1UdIwSBkTCBjoAUsnsEPOUYPQO0Gv9t4arIC2NNkHShYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUMDbUg+re
aqO28n34vFUuk6gFWbgwEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQCFibNMMdZF13osIpWxjb7uXrqpDYFat+196JVSdq+WMYLfcQtudZOcTclo
Zh+OXR49nRv9x/Q4XRfMWGCdptp4q/6NCScEUPBbY2+7CJyH0gyC6vYcGmLfJbUi
kMB27EWJ4tm18a/qZockIeGgJgqt5/KSZb5iN5CuxRHRDGZI+Q==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAMtdyRcH51j9MA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yMjA0MDgx
MjM5MTZaGA8yMjAxMDkxMjEyMzkxNlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAvUsKCxoH6KXRYJLeYTWAQfaBQeCwhJaR56mfUeFHJE4g8aFjWkiN4uc1
TvOyYNnIZKTHWmzmulmdinWNbwP0GiROHb/i7ro0Hhvnptyycgt8ag8affiIbx5X
7ohdwSN2KJ6G0IKflIx7f2NEI0oAMM/9k+T1eVF+MVWzpZoiDp8frLNkqp8+RAgz
ScZsbRfwv3u/if5xJAvdg2nCkIWDMSHEVPoz0lJo7v0ZuDtWWsL1LHnL5ozvsKEk
+ZJyEi23r+U1hIT1NTBdp4yoigNQexedtwCSr7q36oOdDwvZpqYlkLi3uxZ4ta+a
01pzOSTwMLgQZSbKWQrpMvsIAPrxoQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQU1GgnGdNpbnL3lLF30Jomg7Ji9hYwgY4GA1UdIwSBhjCBg4AU1Ggn
GdNpbnL3lLF30Jomg7Ji9hahYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
```

```
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAMtdyRcH51j9MBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBACVl00qQlatBKVeiWMrhpczsJroxDxlZTOba
6wTMZk7c3akb6XMOSZFbGaifkebPZqTHEhDlrClM2j9AIlYcCx6YCrTf4cuhn2mD
gcJN33143eOWSaeRY3ee4j+V9ne98y3kO2wLz95VrRgclPFR8po2iWGzGhwUi+FG
q8dXeCH3N0DZgQsSgQWwmdNQXZZej6RHLU/8In5trHKLY0ppnLBjn/UZQbeTyW5q
RJB3GaveXjfgFUWj2qOcDuRGaikdS+dYaLsi5z9cA3FolHzWxx9MOs8io8vKqQzV
XUrLTNWwuhZy88cOlqGPxnoRbw7TmifwPw/cunNrsjUUOgs6ZTk=
-----END CERTIFICATE-----
```

アジアパシフィック (マレーシア) - ap-southeast-5

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7zCCAq4CCQC5X6U+vgoLEDAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yNDAxMDMxMjU3NTRaGA8y
MDUwMDEwMzEyNTc1NFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0
b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIg
U2VydmljZXMgTExDMIIBtjCCASsGByqGSM44BAEwggEeAoGBAIzEMPCIPFfOYCg4
BCjKGyl60wOfmwHPzSOXZ3Z2wS/LYNHUtHGwtVNePSTyCu/CuZF6gC9n/wBORtQp
+SSkn+weGc/BmUA1mp/vrN7v+aSCgKJoO+sgpa1PPOgNvUaMw6O5odsZWQCMSjkU
6RTo/PL2v/tMfiCocF4ghvyRC6hvAhUA0VoObKC2IXzXgVvRRupo4qHbcm8CgYAe
bbNuawh3rAxkFvUs9FPzW5E+xllG16Z//61PENKqonmk+zBiBdiIlS1F6ZqmTqkI
z5+qfSt1m3pb3j2W0NT7lEDFvy8Gr6Y2vohCHmL+T1u1Yy4PeqbgrFwcen7y7WoO
/KCV7Y9/ODQMMyuAzT3h5wJNweT7L5MUN8JYpZSi3QOBhAACgYBqaDuG2u6V91Qj
K2wEAElxaaRaNo/ewg/wWDmHYqoeHOR0HfuFCYgASE9f7ULqYtX1VURcgcjw9XN4
BDmPilXvfi04INPTnw4IxFJKDzzC0kVH7esVas982Po8v3megH32H9Rl87r7UG1c
ZEbkSkKVX6YKYg1PR3rfjXgdwVZv/zAJBgcqhkjOOAQDAzAAMC0CFFWeRe2fYW2i
6mMd26Wzbx87Y0DXAhUAoPCnF+5hGJw0jT9aL7QsgcfLi9Y=
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICNjCCAZ+gAwIBAgIJAMuBl6rhZCJKMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yNDAxMDMx
MjU3NTRaGA8yMjAzMDYwOTEyNTc1NFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKB
```

```
gQDFuKydxZsordNH7bLwIluEGOkX7/CdLdpeqkDKEhQkFwzpRxaX4EAlkGh2/o7D
8qneC9cGQhqSG5WVVBrmZG7sfkFOM4m1AtY++kfv+MYto1VFgLk1xJbkpq1r4YeQ
Ul+ZsJYsZpyX/t+g8s7rW0OVcBsYx4L75bf34z38mwK8PQIDAQABMA0GCSqGSIb3
DQEBCwUAA4GBADD9C4pWL8RUvF1CJW8kExj35xmozlFlmrKs8Zpi8+Eg6q+W9dgd
xMdH95tgZtmVMDqlvVR+DK0iO1BNpqPjrqWkk2tTLivpS+sGzCE/jCl18Q28Rk71
/A3gLD7Rtbq5TKNvuFCHwYMjrTDHI6aBjIaAlDm4e2/j/OxVtHyZGTre
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJANC3xtbPhQ2GMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yNDAxMDMx
MjU3NTRaGA8yMjAzMDYwOTEyNTc1NFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAt3aMy7Hsp4ySG3mlfi+pdTcZw6H6XNU1Y36fNdi4c+MzinQQbnqMPyt7
QLgU+XCWmcWsVo7GQF6n9N01Rh+UXXUZU4jcXlFocQPCWf9O+IIIPXkd67kFMUV
HAxCELjfxHbC+I8e7dwOJhmdF4Bfi52Ty8zz0HdE8JDypPkTDlXuGvTgDyW7NP56
I/vlQaXLoYSbcQe5pv2a9gyBaaCM1QoeqwAHAeCNXb9Nuj9ZX3GHGJb3TuqAeKCD
5i9TscCB9XjY6Fx+zfSAobjBZwgLEtLOwJhbZnKmx4gJMaaNFipAjVT2FSS3+yev
eTYBoa1dvhk0ivQyQIPpHmihrmkWuwIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQBC
fdgyI8GjmCqiHAlh+L1bj0LdNql9zl7RXmOEzsuRdtdMumkxYXX88UtROy3fdi1i
VaEwHdAK8ThzRkesgHza/cXzqCMewaYxujSI6p6G7x99FFeGif1x0FJdj8AoeTL7
4h9bmS/6l4/NL7DJI9G7ovES/hoUA9v9TDhv+vauxXlgfrp0MPecprxBYlrc+DH2
adGCdKCp2lQ2YDKOD9TCEjYIli8XSoyevoWHUjfDYrCrCp8l4s/p7HOgYr8fJBAs
EuVy8211LVz1/X4EMBRNtNjXK9sklsxAOX14NDfBFSS0toxl3K6Tf9t/PviBl95d
hncyDAcFgDCK4w8LLlVW
-----END CERTIFICATE-----
```

アジアパシフィック (メルボルン) - ap-southeast-4

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7zCCAq
+gAwIBAgIGAXjWF7P2MAkGByqGSM44BAMwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgMEFdhc2hpbmd0b24gU3RhdGUxED
U4EddRIpUt9KnC7s5Of2EbdSPO9EAMMeP4C2USZpRV1AIlH7WT2NWPq/
xfW6MPbLm1Vs14E7gB00b/JmYLdrmVClpJ+f6AR7ECLCT7up1/63xhv4O1fnxqimFQ8E
+4P208UewwI1VBNaFpEy9nXzrith1yrv8iIDGZ3RSAHHAhUAl2BQjxUjC8yykrmCouuEC/
BYHPUCgYEA9+GghdabPd7LvKtcNrhXuXmUr7v6OuqC+VdMCz0HgmdRWVeOutRZT
```

```
+ZxBxCBgLRJFnEj6EwoFhO3zwkyjMim4TwWeotUfI0o4KOuHiuzpnWRbqN/C/ohNWLx
+2J6ASQ7zKTxvqhRkImog9/
hWuWfBpKLZl6Ae1UlZAFMO/7PSSoDgYQAAoGAPRXSsQP9E3dw8QXKlrgBgEVCprLHdK/bbrMas0XMu1EhOD
+q
+0PcTr8+iwbtoXlY5MCeatWIp1GrXQjVqsF8vQqxlEuRuYKbR3nq4mWwaeGlx9AG5EjQHRa3GQ44wWHOdof0M3NRI1MP
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUUOsUa3Aj7pphfmDjsZAFNWODCUowDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI2MDYwMzEwNDM1M1oXDTMxMDYwMjEwNDM1M1owXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQDHezwQr2VQpQSTW5TXNefiQrP+qWTGAbGsPeMX4hBMjAJUKys2
NIRcRZaLM/BCew2FIPVjNtlaj6Gwn9ipU4Mlz3zIwAMWi1AvGMSreppt+wV6MRtf
Ojh0Dvj/veJe88aEZJMozNgkJFRS+WFWsckQeL56tf6kY6QTlNo8V/0CsQIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUVCy+PqHsYEzvAHcGK+lrlknS
UvowgZkGA1UdIwSBkTCBjoAUVCy+PqHsYEzvAHcGK+lrlknSUvqhYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUUOsUa3Aj
7pphfmDjsZAFNWODCUowEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQBRI72kM+Q6PCU584L6UZ8rk6uERCeq5g8dDzlLZMaMpV4n/s2MJbFP2hO+
ovahJnxLTrs7jP3VXZeVj5Eb0dfYneg0+zNtycnHe+9jH+GTKS8+QkwGj3zmwxqG
mZ9V6XoGwtO/aEz24Yp7CRf8etCYEnVI2ChD7xClRV3DJwliEA==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAN4GTQ64zVs8MA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yMjA3MTMx
MzMzMDBaGA8yMjAxMTIxNzEzMzMwMFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEA2BYgeCr+Rk/jIAEDOHS7wJql62vc83QEwjuzk0qOFEReIZzlN1fBRNXK
g0T178Kd3gLYcE59wEFbTe/X5yOA1Lo95x1anSAo7R+Cisf9C2HQuJp+gVb+zx71
lniPF7gHziGpm0M8DdAU/IW+wkZwGbP4z7Hq9+bJ0P2ltvPJ5yxSgkFuDsI9VBHa
CLoprHsCHh2VdP8KcMgQQMmHe1NmBpyTk0ul/aLmQkCQEX6ZIRGOeq228fwlh/t+
Ho+jv87duihVKic6MrL32SlD+maX0LSDUydWdaOLLTGkh7oV7+bFuH6msrXUu+Ur
```

```
ZEP1r/MidCWMhfgrFzeTBzOHA97qxQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQUcHMd1cHqzmsQ5hpUK3EMLhHdsi4wgY4GA1UdIwSBhjCBg4AUcHMd
1cHqzmsQ5hpUK3EMLhHdsi6hYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAN4GTQ64zVs8MBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAI4PFyVN+7EGSObioiPnv0LL0f70SSzUZJ8p
XO90d4rWea7jIbgZ2AKb+ErynkU9xVg7XQQ5k6KDWgp/4jYFL2dqnt/YAY4PS0un
RSrYElawxLT0BcLn4rcSDC79vQe1xGC5//wDdV6b399COAHRAK6axWYy5w32u9PL
uw0cIp3Ch8JoNwcgTHKRRGzePmBeR4PNqhHTArG4/dJk6/aUO4OpX0WzI6L67CGY
6Nex3dau+gkLCK93dTEkrXtyXHu4wB0J9zd1w+iQ0SEa9eKc78/NjEsF/FZdGrWC
t57lIMOOXJhQ1kRgSwNeZdQWV1dRakvO6sfcvVYkfj1wAvZvvAw=
-----END CERTIFICATE-----
```

アジアパシフィック (ムンバイ) - ap-south-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUDLA+x6tTAP3LRTr0z6nOxfsozdMwDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
```

```
MB4XDTI0MDQyOTE0MTMwMVoXDTI5MDQyODE0MTMwMVowXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQCHvRjf/0kStpJ248khtIaN8qkDN3tkw4VjvA9nvPl2anJO+eIB
UqPfQG09kZlwpWpmyO8bGB2RWqWxCwuB/dcnIob6w420k9WY5C0IIGtDRNauN3ku
vGXkw3HEnF0EjYr0pcyWUvByWY4KswZV42X7Y7XSS13hOIcL6NLA+H94/QIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUJdbMCBXKtvCcWdwUUizvtUF2
UTgwgZkGA1UdIwSBkTCBjoAUJdbMCBXKtvCcWdwUUizvtUF2UTihYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUDLA+x6tT
AP3LRTr0z6nOxfsozdMwEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQAZ7rYKoAwwiiH1M5GJbrT/BEk3OO2VrEPw8ZxgpqQ/EKlzMlOs/0Cyrmp7
UYyUgYFQe5nq37Z94rOUSeMgv/WRxaMwrLlLqD78cuF9DSkXaZIX/kECtVaUnjk8
BZx0QhoIHOpQocJUSlm/dLeMuE0+0A3HNR6JVktGsUdv9ulmKw==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJAPRYyD8TtmC0MA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNjAzMDcx
MDQ1MDFaGA8yMTk1MDgxMTEwNDUwMVowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEA0LSS5I/eCT2PM0+qusorBx67QL26BIWQHd/yF6ARtHBb/1DdFLRqE5Dj
07Xw7eENC+T79mOxOAbeWg91KaODOzw6i9I/2/HpK0+NDEdD6sPKDA1d45jRra+v
CqAjI+nV9Vw91wv7HjMk3RcjWGziM8/hw+3YNIutt7aQzZRwIWlBpcqx3/AFd8Eu
2UsRMSHgkGUW6UzUF+h/U8218XfrauKNGmNKDYUhtmyBrHT+k6J0hQ4pN7fe6h+Z
w9RVHm24BGhlLxLHLmsOIxvbrF277uX9Dxu1HfKfu5D2kimTY7xSZDNLR2dt+kNY
/+iWdIeEFpPT0PLSILt52wP6stF+3QIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQBI
E6w+WWC2gCfoJO6c9HMyGLMFEpqZmz1n5IcQt1h9iyO7Vkm1wkJiZsMhXpk73zXf
TPxuXEacTX3SOEa07OIMCFwkusO5f6leOyFTynHCzBgZ3U0UkRVZA3WcpbNB6Dwy
h7ysVlqyT9WZd7EOYm5j5oue2G2xdei+6etgn5UjyWm6liZGrcOF6WPTdmzqa6WG
ApEqanpkQd/HM+hUYex/ZS6zEhd4CCDLgYkIjlrFbFb3pJ1OVLztIfSN5J4Oolpu
JVCfIq5u1NkpzL7ys/Ub8eYipbzI6P+yxXiUSuF0v9b98ymczMYjrSQXIf1e8In3
OP2CclCHoZ8XDQcvvKAh
-----END CERTIFICATE-----
```

アジアパシフィック (ニュージーランド) – ap-southeast-6

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7zCCAq4CCQCcP+D4eLFOajAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yNDEyMDMxMjEzMDZaGA8y
MDUwMTIwNDEyMTMwNlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0
b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIg
U2VydmljZXMgTExDMIIBtjCCASsGByqGSM44BAEwggEeAoGBANBMmM/LbAX13IeB
3yr2pMMpiTW1b1go/axNLfcxEmj0fThKk8y0FAT1frN/UPdI9qRInsyBdEZQLiIH
4WHCXk8IidQLh4DED3G9DvSFFC4Gb8ax+UIQY11wfHPWO4+wA9q8liYTg/FxdJMM
tZS9tZAZxjH6+8yfV8Bt0wUd2Y+lAhUAwg0CxCmZPinHtKbFnpMQfKnzjCMCgYAf
i+D7nGVFOWJ8EYOpRYbG0pHkrpwNy+h01csn6IAwQbIfeOG+SF4N3LeuOAC9vXGY
VGh9XNxuasvqkRAzgRmJ/EYM5Xa/YN62EhddRuN8wCOg66MeczUDh3RTb+Tu6HK0
UR6oTSOQ6rxA5aA1qTc7TqI0+pRZv+QYIZK/ozsZzgOBhAACgYAbVm+rOX8J38cG
LxKjUXVM0KB69SksGLEqjO4ps80EcZ/0W3g1hGLnSX/RjYfAS60hbKK1GX9PpcaO
4mDzCTm3dZ6shbae5t8bBU+mUi8BmaK+Pdu3z5J77y4zv22/yXte8ulZ5J0zKP9K
QP3p3pa7XI9NaXpKGT3yZsJ0BWY2xTAJBgcqhkjOOAQDAzAAMC0CFBs4KQmrxHjQ
OHh0r8JqB8UiwlutAhUAwgxid7wT6FNm8Zwq9hioQ2v2K0A=
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICNjCCAZ+gAwIBAgIJAMwLuv2/tk9eMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yNDEyMDMx
MjEzMDZaGA8yMjA0MDUwOTEyMTMwNlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKB
gQC9AsBNOUfT8QpguHqMa9KIXP/YgZOmfRAgEmnLawXTC+40bXUq/fwkx+cebv3x
CvUx6d07/iDlD0dm2Hf2HhPFi3OaCI7vc8c3lqVBvcn6yzmwv35aSEvtVm2NHC3j
4/q6vN2tR2d4a1Mf1QhmOVOJ7XiTEbCy2Yv+pmDbk29V6QIDAQABMA0GCSqGSIb3
DQEBCwUAA4GBAHatyQIy612kio8neI4r5VBg8pgKwW+mFGpAzOQ2NOyLCNoOeD1D
lPRaSipBN7WW6jpIozbz7JG7Kqc3N+piPGGzfjH9gZjqUqfOGrNHB5nsxvMHI2iL
e+DwsOcV7qnEVRgYbmLYOVKJv25wCsM/Tcn/6f6scEX2x+H0hAh7rwiP
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
```

```
MIIDOzCCAiOgAwIBAgIJAI/xUfWwSQOtMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yNDEyMDMx
MjEzMDZaGA8yMjA0MDUwOTEyMTMwNlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEArbid9Tnunu0pWCQUqZCkGeiF8hWixi+LBS1AOew2UsT9keAt2rXXGlU/
wSMriKHsvcrfKtuVKj5chArsOyFvagszH5lIKhI+GfWVQEd2aZiOvAJ+iRUQq1+R
gOr2gyjWlf3znZXTcyhHcJoXDhbn5e+kEa3pwS4yKNRpM3P61V4M2702xis08hjP
qMDxwXMa1Fiy4b7nAyNQifgrwb3bCA5k5prmkl/iajaWa1zMMFgLI9vA6UwUSbTK
H9iOuL/xNfMX+Xzq4CCaQYiJ8zdi56kqnB/lrA05NSjOPLskg6sUGkyiDxGK4gzC
DFrO6oygcdlBaxnyAECGdJcOM6TOmwIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQBg
8yVPNpctrvSvTazf/Ij8qC6+q6df1SptdYKQkOYWeUVZR7+3uTZ9Yiyq+Bh8Hbyf
NcR0eR6UqhgiB3p5slYNhTvLkpuMuqTpVJFvQtzZycnmKXAQ35G5zCYV0+pkF59L
ch26SukWk0+VP3kIM9VmaIZQKJEBlBLcnbDZyXlHUDnZAYPcxxKQ8uFPkW0GOFMr
ekFCGOUmFSItFXOHy5hD2xmctKNk3l12m6cKydYvONtk/3DovKNnw3c5k40WHQW/
Zdh/LWd4/Jrkan+kQnqm4rkrXF+x3aF0qrKTtHMICvpkZ8g7/588hBs4mZnCmFLa
q/g9WqIcd9SMmBkT6KZ8
-----END CERTIFICATE-----
```

アジアパシフィック (大阪) – ap-northeast-3

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUHTRhxHhBZFOGvTFKxHoy9+f5Hl8wDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI0MDQyOTE2NTQwN1oXDTI5MDQyODE2NTQwN1owXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQCHvRjf/0kStpJ248khtIaN8qkDN3tkw4VjvA9nvPl2anJO+eIB
UqPfQG09kZlwpWpmyO8bGB2RWqWxCwuB/dcnIob6w420k9WY5C0IIGtDRNauN3ku
vGXkw3HEnF0EjYr0pcyWUvByWY4KswZV42X7Y7XSS13hOIcL6NLA+H94/QIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUJdbMCBXKtvCcWdwUUizvtUF2
UTgwgZkGA1UdIwSBkTCBjoAUJdbMCBXKtvCcWdwUUizvtUF2UTihYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUHTRhxHhB
ZFOGvTFKxHoy9+f5Hl8wEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQAUZx7DcYbhWNTD4BNGhr5beruT2OUoGHH9J73UKxwdqeb9bH1LIWhIZO0X
/1mjn3bWBgCwfoS8gjZwsVB6fZbNBRy8urdBZJ87xF/4JPBjt7S9oGx/zthDUYrC
yK0Y0v4G0PgiS81CvYLg09LpmYhLSJbXENlkC04v5yxdKxZxyg==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJAMn1yPk22ditMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNzA3MTkx
MTEyNThaGA8yMTk2MTIyMjExMTI1OFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEArznEYef8IjhrJoazI0QGZkmlmHm/4rEbyQbMNifxjsDE8YWtHNwaM91z
zmyK6Sk/tKlWxcnl3g31iq305ziyFPEewe5Qbwf1iz2cMsvfNBcTh/E6u+mBPH3J
gvGanqUJt6c4IbipdEouIjjnynyVWd4D6erLl/ENijeR1OxVpaqSW5SBK7jms49E
pw3wtbchEl3qsE42Ip4IYmWxqjgaxB7vps91n4kfyzAjUmklcqTfMfPCkzmJCRgp
Vh1C79vRQhmriVKD6BXwfZ8tG3a7mijeDn7kTsQzgO07Z2SAE63PIO48JK8HcObH
tXORUQ/XF1jzi/SIaUJZT7kq3kWl8wIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQBj
ThtO9dLvU2QmKuXAhxXjsIdlQgGG3ZGh/Vke4If1ymgLx95v2Vj9Moxk+gJuUSRL
BzFte3TT6b3jPolbECgmAorjj8NxjC17N8QAAI1d0S0gI8kqkG7V8iRyPIFekv+M
pcai1+cIv5IV5qAz8QOMGYfGdYkcoBjsgiyvMJu/2N2UbZJNGWvcEGkdjGJUYYOO
NaspCAFm+6HA/K7BD9zXB1IKsprLgqhiIUgEaW3UFEbThJT+z8UfHG9fQjzzfN/J
nT6vuY/0RRu1xAZPyh2gr5okN/s6rnmh2zmBHU1n8cbCc64MVfXe2g3EZ9Glq/9n
izPrI09hMypJDP04ugQc
```

```
-----END CERTIFICATE-----
```

アジアパシフィック (ソウル) - ap-northeast-2

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUbBSn2UIO6vYk4iNWV0RPxJJtHlgwDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI0MDQyOTEzMzg0NloXDTI5MDQyODEzMzg0NlowXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQCHvRjf/0kStpJ248khtIaN8qkDN3tkw4VjvA9nvPl2anJO+eIB
UqPfQG09kZlwpWpmyO8bGB2RWqWxCwuB/dcnIob6w420k9WY5C0IIGtDRNauN3ku
vGXkw3HEnF0EjYr0pcyWUvByWY4KswZV42X7Y7XSS13hOIcL6NLA+H94/QIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUJdbMCBXKtvCcWdwUUizvtUF2
UTgwgZkGA1UdIwSBkTCBjoAUJdbMCBXKtvCcWdwUUizvtUF2UTihYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUbBSn2UIO
6vYk4iNWV0RPxJJtHlgwEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
```

```
AAOBgQAmjTjalG8MGLqWTC2uYqEM8nzI3px1eo0ArvFRsyqQ3fgmWcQpxExqUqRy
l3+2134Kv8dFab04Gut5wlfRtc2OwPKKicmv/IXGN+9bKFnQFjTqif08NIzrDZch
aFT/uvxrIiM+oN2YsHq66GUhO2+xVRXDXVxM/VObFgPERbJpyA==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJANuCgCcHtOJhMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTA5MTQx
NTU3NDRaGA8yMTk1MDIxNzE1NTc0NFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEA66iNv6pJPmGM20W8HbVYJSlKcAg2vUGx8xeAbzZIQdpGfkabVcUHGB6m
Gy59VXDMDlrJckDDk6dxUOhmcX9z785TtVZURq1fua9QosdbTzX4kAgHGdp4xQEs
mO6QZqg5qKjBP6xr3+PshfQ1rB8Bmwg0gXEm22CC7o77+7N7Mu2sWzWbiUR7vil4
9FjWS8XmMNwFTlShp4l1TDTevDWW/uYmC30RThM9S4QPvTZ0rAS18hHVam8BCTxa
LHaVCH/Yy52rsz0hM/FlghnSnK105ZKj+b+KIp3adBL8OMCjgc/Pxi0+j3HQLdYE
32+FaXWU84D2iP2gDT28evnstzuYTQIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQC1
mA4q+12pxy7By6g3nBk1s34PmWikNRJBwOqhF8ucGRv8aiNhRRye9lokcXomwo8r
KHbbqvtK85l0xUZp/Cx4sm4aTgcMvfJP29jGLclDzeqADIvkWEJ4+xncxSYVlS9x
+78TvF/+8h9U2LnSl64PXaKdxHy2IsHIVRN4GtoaP2Xhpa1S0M328Jykq/571nfN
1WRD1c/fQf1edgzRjhQ4whcAhv7WRRF+qTbfQJ/vDxy8lkiOsvU9XzUaZ0fZSfXX
wXxZamQbONvFcxVHY/0PSiM8nQoUmkkBQuKleDwRWvkoJKYKyr3jvXK7HIWtMrO4
jmXe0aMy3thyK6g5sJVg
-----END CERTIFICATE-----
```

アジアパシフィック (シンガポール) - ap-southeast-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
```

```
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUSqP6ih+++5KF07NXngrWf26mhSUwDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI0MDQyOTE0MzAxNFoXDTI5MDQyODE0MzAxNFowXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQCHvRjf/0kStpJ248khtIaN8qkDN3tkw4VjvA9nvPl2anJO+eIB
UqPfQG09kZlwpWpmyO8bGB2RWqWxCwuB/dcnIob6w420k9WY5C0IIGtDRNauN3ku
vGXkw3HEnF0EjYr0pcyWUvByWY4KswZV42X7Y7XSS13hOIcL6NLA+H94/QIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUJdbMCBXKtvCcWdwUUizvtUF2
UTgwgZkGA1UdIwSBkTCBjoAUJdbMCBXKtvCcWdwUUizvtUF2UTihYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUSqP6ih++
+5KF07NXngrWf26mhSUwEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQAw13BxW11U/JL58j//Fmk7qqtrZTqXmaz1qm2WlIpJpW750MOcP4ux1uPy
eM0RdVZ4jHSMv5gtLAv/PjExBfw9n6vNCk+5GZG4Xec5DoapBZHXmfMo93sjxBFP
4x9rWn0GuwAVO9ukjYPevq2RerilrqSVvppHtbATVNY2qecXDA==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAJVMGw5SHkcvMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTEwMjkw
ODU3MTlaGA8yMTk1MDQwMzA4NTcxOVowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAlaSSLfBl7OgmikjLReHuNhVuvM20dCsVzptUyRbut+KmIEEc24wd/xVy
2RMIrydGedkW4tUjkUyOyfET5OAyT43jTzDPHZTkRSVkYjBdcYbe9o/0Q4P7IVS3
XlvwrUu0qo9nSID0mxMnOoF1l8KAqnn10tQ0W+lNSTkasW7QVzcb+3okPEVhPAOq
```

```
MnlY3vkMQGI8zX4iOKbEcSVIzf6wuIffXMGHVC/JjwihJ2USQ8fq6oy686g54P4w
ROg415kLYcodjqThmGJPNUpAZ7MOc5Z4pymFuCHgNAZNvjhZDA842Ojecqm62zcm
Tzh/pNMNeGCRYq2EQX0aQtYOIj7bOQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQU6SSB+3qALorPMVNjToM1Bj3oJMswgY4GA1UdIwSBhjCBg4AU6SSB
+3qALorPMVNjToM1Bj3oJMuhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAJVMGw5SHkcvMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAF/0dWqkIEZKg5rca8o0P0VS+tolJJE/FRZO
atHOeaQbWzyac6NEwjYeeV2kY63skJ+QPuYbSuIBLM8p/uTRIvYM4LZYImLGUvoO
IdtJ8mAzq8CZ3ipdMs1hRqF5GRp8lg4w2QpX+PfhnW47iIOBiqSAUkIr3Y3BDaDn
EjeXF6qS4iPIvBaQQ0cvdddNh/pE33/ceghbkZNTYkrwMyBkQlRTTVKXFN7pCRUV
+L9FuQ9y8mP0BYZa5e1sdkwebydU+eqVzsil98ntkhpjvRkaJ5+Drs8TjGaJWlRw
5WuOr8unKj7YxdL1bv7//RtVYVVi296ldoRUYv4SCvJF11z0OdQ=
-----END CERTIFICATE-----
```

アジアパシフィック (シドニー) - ap-southeast-2

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUFxWyAdk4oiXIOC9PxcgjYYh71mwwDQYJKoZIhvcNAQEL
```

```
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI0MDQyOTE1MjE0M1oXDTI5MDQyODE1MjE0M1owXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQCHvRjf/0kStpJ248khtIaN8qkDN3tkw4VjvA9nvPl2anJO+eIB
UqPfQG09kZlwpWpmyO8bGB2RWqWxCwuB/dcnIob6w420k9WY5C0IIGtDRNauN3ku
vGXkw3HEnF0EjYr0pcyWUvByWY4KswZV42X7Y7XSS13hOIcL6NLA+H94/QIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUJdbMCBXKtvCcWdwUUizvtUF2
UTgwgZkGA1UdIwSBkTCBjoAUJdbMCBXKtvCcWdwUUizvtUF2UTihYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUFxWyAdk4
oiXIOC9PxcgjYYh71mwwEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQByjeQe6lr7fiIhoGdjBXYzDfkX0lGGvMIhRh57G1bbceQfaYdZd7Ptc0jl
bpycKGaTvhUdkpMOiV2Hi9dOOYawkdhyJDstmDNKu6P9+b6Kak8He5z3NU1tUR2Y
uTwcz7Ye8Nldx//ws3raErfTI7D6s9m63OX8cAJ/f8bNgikwpw==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAL2bOgb+dq9rMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTEwMjkw
OTAwNTdaGA8yMTk1MDQwMzA5MDA1N1owXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAmRcyLWraysQS8yDC1b5Abs3TUaJabjqWu7d5gHik5Icd6dKl8EYpQSeS
vz6pLhkgO4xBbCRGlgE8LS/OijcZ5HwdrxBiKbicR1YvIPaIyEQQvF5sX6UWkGYw
Ma5IRGj4YbRmJkBybw+AAV9Icb5LJNOMWPi34OWM+2tMh+8L234v/JA6ogpdPuDr
sM6YFHMZ0NWo58MQ0FnEj2D7H58Ti//vFPl0TaaPWaAIRF85zBiJtKcFJ6vPidqK
f2/SDuAvZmyHC8ZBHg1moX9bR5FsU3QazfbW+c+JzAQWHj2AaQrGSCITxCMlS9sJ
l51DeoZBjnx8cnRe+HCaC4YoRBiqIQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQU/wHIo+r5U31VIsPoWoRVsNXGxowwgY4GA1UdIwSBhjCBg4AU/wHI
o+r5U31VIsPoWoRVsNXGxoyhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAL2bOgb+dq9rMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBACobLvj8IxlQyORTz/9q7/VJL509/p4HAeve
92riHp6+Moi0/dSEYPeFTgdWB9W3YCNc34Ss9TJq2D7t/zLGGlbI4wYXU6VJjL0S
hCjWeIyBXUZOZKFCb0DSJeUElsTRSXSFuVrZ9EAwjLvHni3BaC9Ve34iP71ifr75
8Tpk6PEj0+JwiijFH8E4GhcV5chB0/iooU6ioQqJrMwFYnwo1cVZJD5v6D0mu9bS
TMIJLJKv4QQQqPsNdjiB7G9bfkB6trP8fUVYLHLsVlIy5lGx+tgwFEYkG1N8IOO/
2LCawwaWm8FYAFd3IZl04RImNs/IMG7VmH1bf4swHOBHgCN1uYo=
```

```
-----END CERTIFICATE-----
```

アジアパシフィック (台北) – ap-east-2

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7jCCAq4CCQDrb+HArMxssTAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yNDA1MjQxNDU2MDZaGA8y
MDUwMDUyNTE0NTYwNlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0
b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIg
U2VydmljZXMgTExDMIIBtjCCASsGByqGSM44BAEwggEeAoGBAPFb5R5TBUs2N0YW
aJcOjxr4UCDrpVT3x6EeqoGFNAc1yiZw9iv++ff6wsXm0VG5XV1QkWMQcjPfK/2I
TvXh9XEOhsG7vaBHoWkb2gmIxE6C2u8TfoZu0ZmiMceFK7aYz5w/q2J9CmRs38dT
I9qgTq0LYX+VkiNYOQnn7sQXQ8CjAhUAp2Lue9SfYTmubbpH+3j91Sivc+cCgYAi
DiyGU5R5VgiNV3mhuAyiHCft3LEmTIxos/hx/rHzgm+vCMH55tYejIrv9n7YGrIZ
MejOt8bGYxqSf0Ir/aoj3ZfHihxUzwyfyzwH7D8kAodXJNsPUTKi0O0pGl/e+nVD
JhTIFV3AwC8Tt0r2hljbjhOiyhdYy5hxWHa/WokLAQOBhAACgYAkGGHXbFr/JXol
itpS1qfiETWSRkXGHrdSmUkRumgti2y7c57TgXPwCyNlP1sTr1duy/kYDpwSwyqi
qEYfko0b/xWpDEQQPsWBgolpEYP7RIT85IxrTCrC9ohz85n7v6m9oF6BiBQGtZdk
GBN7iqemg85UaF6mXfHvl3tUYjsB2jAJBgcqhkjOOAQDAy8AMCwCFED9Ybn6fEVF
wLNjDvULmki5bPBWAhR64pxsHFRHz0ZQi+SYMWdIpEs5RQ==
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICNjCCAZ+gAwIBAgIJAJp99iy2CTXhMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yNDA1MjQx
NDU2MDZaGA8yMjAzMTAyOTE0NTYwNlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKB
gQDVdclgpcIIsyGG51ag8srK1UMfZ+Ioh9E4PLP+cIWQcdh31fv2kV/whKaqfmif
NefBIuHrTPl6L+mJQbsfeA+TiEx1Jamg4lwQfhgQKQCmjZVMaHtK+AY57TVN6FBf
HXxT2MXzktsKxAQRGeSTFQYrHRzSYvu+GkA+37lqyWRfcwIDAQABMA0GCSqGSIb3
DQEBCwUAA4GBABM+WL9MSb7z3bNOTMkM7ufPucvj+LQ+zEAJiuG0rcB8GI2DhGB5
8ohDPUGlq1cP7vYmDkWG/FrLqx7qgIzkbE3zfb62JpdOuX73oHgCmikMfq8ifoS1
CUxMQ36PDZBFNPzGZxv5kEORqhmGBkYCoIUCQ43jh3aALR/A/rfzfij8
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJAJt5hgMuWbpMMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yNDA1MjQx
NDU2MDZaGA8yMjAzMTAyOTE0NTYwNlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEA91FXs9Wk9kMI3lkNhjC69C6RTBpSvdLWSohPDlJKXFoaGYIDtyW1ViwO
FORAKz0koarVBx1D4fNKCxf+mhYAf//4Pxt31GzjehHNs2YVGimwhq/SNr+3AuIO
PibA+qScXamaiknE1Pxb4jisbBu2PMAG9OxU2dESNwiqBMvdJvg30YNTUITZ6r0e
MPjR1Y/EFVx/SLHW6wdvwqhNi9hz8HHdThzjCbCQPvmoLBJIKXIfosKqvauSezt1
KJSoS2jMx1OjJB2bBlQTZC8vF0WFNPoiH4jXyJ6+wMUMNSSd5b3uL5stVTMzuIz4
39Hd5YW41rmAy+6jmpGp2eXn6Td0rQIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQAs
gcgv+Xr67Cj08jKdAUT4GuCoGxveOFAjSDjs5y0cD+kXIT/Yj/+HKFOx5cwThT8F
TxNQ/oVuo3bGeuFEX799H0rT9Jupy0WiE1Vx81mk2M/1EDtns437edM4ofiWR8tR
Q1VNiRgz6Crr0nFgNoQa9HmR6/M+bImxevHJkWB3skvnxtA+lpKGwJs8jo6fA7AW
jr00opK0r731iBd1z/5hjIjkHn5NizAdiJEh3lVY/A3WS82aygW8LvYnKZeaBfZJ
J4NgxeflxocLFKBJD+hyf5u1hhvl7Mn/aGr5VI8/0F2oUqYo+xsHLT3keNghywO4
NNcyLcD9u16HVXhRhGvo
-----END CERTIFICATE-----
```

アジアパシフィック (タイ) - ap-southeast-7

DSA

```
-----BEGIN CERTIFICATE-----
MIIC8TCCAq8CCQCOEEMWiJIJpTAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yNDA0MTIxNTI4NTZaGA8y
MDUwMDQxMzE1Mjg1NlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0
b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIg
U2VydmljZXMgTExDMIIBtzCCASsGByqGSM44BAEwggEeAoGBAJWKfrIlz9+/U4wm
NDJPX00avUTdksBKX5RwHkY4o0///2G2HWwiBbxw8F1U+RzCXr80gSoe4+/SCSg3
uDuhJyzN5r4jr7c590CAgsiGdWERs7l4gLQCP504feU4TEjJoq9A8MfRZeqMj7Ug
fT5InTUW7S1/r98ddG/rBHMsJwEbAhUAneIckZ1YeyGcCeL32lujMhj+g68CgYA+
RsyTXCy3Tug5aHun51IfcG2d+pn5K/tv/N3WUR18RplVctrLhwIOUoAsDWPWtxNV
sODetezAyo759CK43JAmYGzXKbRFUHm3n46jP88tSdhuSeJhc/D4l5/0+2L7ndXp
L3+W4N05NdiKSGI8e8t452wTZv/RODivPZ3RtuCDywOBhQACgYEAjVS1Huwzdn1J
+kpd2Rcbe1BAGkmvk5sUu0KhyttqIB1kxTxeWZgsdO8REZSC5gJOwkcGFvXnb3DY
+Ms45r0e0s0rx2FFYjqMLwyRpK9wUjJfSXeJMa9iLQEXuyzBz6zPgfemXbS3zNq/
```

```
eoJ9ztIwjB9DMoKL+E1vSLsTGhehqRowCQYHKoZIzjgEAwMxADAuAhUAmljDM3c9
hf0j4XbMjjpnzrx1xhkCFQCASR9pgFNgLK8y6Kojj+P1KJkrSQ==
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICNjCCAZ+gAwIBAgIJAIuIHAhL0xWcMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yNDA0MTIx
NTI4NTZaGA8yMjAzMDkxNzE1Mjg1NlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKB
gQCajgAeOauwvqGDLrHvxujnZ1BnkMzwjrycMUTkj8jqNtWoDQWUJVNPZJILosEU
VwK2I3oNkEsx/ryl9XfXcNNceoYfVEPzkTzozrZyuOG66FWtUU1LKeJ7h9/rX0Zd
9lZEokrdr6dLPt9FsHWaK5ExlUnWBjNltcQLkkKqoeYaFwIDAQABMA0GCSqGSIb3
DQEBCwUAA4GBAE4G5G+FvKTsX3T7BEcId7f5LSCc2J9gZRDiWn2oTr40CrBMOzJT
KsWr9W89YXW3gaGWltzc0WCwYQbJZgAkuEAZItJjbhdnns87ZbsFO+NZhc6gDtjA
WC3dPlSB9b6rfVoVW9O6Xwa7iNXZo8ddYVJ/ZOIv/totUz9qJt4DmmKk
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJANaQIrYcijxaMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yNDA0MTIx
NTI4NTZaGA8yMjAzMDkxNzE1Mjg1NlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAxtMGK4r6WerVtMfpRRCY3DMP9Q/s4jnRIqolPaxeT99BAFp03HXyOrTz
WjggEHArRGGeDowFkL7Vj+QNY7ran2lnYVlAQ7Ow6OLl6WttT6lMsgsLUIcsED6G
eJ1Ko+ovT1qvmuush1U5R2pHkcCJ1IT2s6uQff85l066K7dCFpaoA1QHUk9zACHj
i0DtWuTAWBXirAZSGlmtP6uj1Aw6y5kACDTjIvZJeW/kmswfApTKaJ56eNkJsUBU
NpKwt3um1BMZduy0cjUv0Sc3dIbLNyfOdyPn8tX5u5ck0EPdtBB5WjEh71IXdZJD
oaBREV7AHg5ERPQsm0BqSvMQt09KiwIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQDG
qJyZbv/GUQnm8dEyVYgilQr3Yu35n8sfTwwZ1OJzIVce7NBPWePMfYii/tvGoKNp
KwFa/Vugbn1EwqpTEKqkPE1dOZvnUa97XxhYkoW5U7sgdKCANi1KWjywn7MiJ3Eg
j4gdqYK8wxvHi21ppqr572UO77uA8YMBoBT/CyQzWYUSmbqwKnnzaQBAZeO2iAk
VHuNU+9UsntNB676gRl9ag3Wfxq3yx5Ee1CeQf+US3HJn/pKk1H8dExXmBHvHwO6
GKUVNPN1rxfjTiaSt8wuO80uElAnyHIM3VOR8rJO7PKsobyEeJV0WI1hURO+wxpL
h3IsW7iBrFVvhX5xx7ZU
-----END CERTIFICATE-----
```

アジアパシフィック (東京) - ap-northeast-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIULgwDh7TiDrPPBJwscqDwiBHkEFQwDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI0MDQyOTEyMjMxMFoXDTI5MDQyODEyMjMxMFowXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQCHvRjf/0kStpJ248khtIaN8qkDN3tkw4VjvA9nvPl2anJO+eIB
UqPfQG09kZlwpWpmyO8bGB2RWqWxCwuB/dcnIob6w420k9WY5C0IIGtDRNauN3ku
vGXkw3HEnF0EjYr0pcyWUvByWY4KswZV42X7Y7XSS13hOIcL6NLA+H94/QIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUJdbMCBXKtvCcWdwUUizvtUF2
UTgwgZkGA1UdIwSBkTCBjoAUJdbMCBXKtvCcWdwUUizvtUF2UTihYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIEExMQ4IULgwDh7Ti
DrPPBJwscqDwiBHkEFQwEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQBtjAglBde1t4F9EHCZOj4qnY6Gigy07Ou54i+lR77MhbpzE8V28Li9l+YT
QMIn6SzJqU3/fIycIro1OVY1lHmaKYgPGSEZxBenSBHfzwDLRmC9oRp4QMe0BjOC
gepj1lUoiN7OA6PtA+ycNlsP0oJvdBjhvayLiuM3tUfLTrgHbw==
```

```
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAL9KIB7Fgvg/MA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTA4MTQw
OTAwMjVaGA8yMTk1MDExNzA5MDAyNVowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAz0djWUcmRW85C5CiCKPFiTIvj6y2OuopFxNE5d3Wtab10bm06vnXVKXu
tz3AndG+Dg0zIL0gMlU+QmrSR0PH2PfV9iejfLak9iwdm1WbwRrCEAj5VxPe0Q+I
KeznOtxzqQ5Wo5NLE9bA61sziUAFNVsTFUzphEwRohcekYyd3bBC4v/RuAjCXHVx
40z6AIksnAOGN2VABMlTeMNvPItKOCIeRLlllSqXX1gbtL1gxSW40JWdF3WPB68E
e+/1U3F7OEr7XqmNODOL6yh92QqZ8fHjG+afOL9Y2Hc4g+P1nk4w4iohQOPABqzb
MPjK7B2Rze0f9OEc51GBQu13kxkWWQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQU5DS5IFdU/QwYbikgtWvkU3fDwRgwgY4GA1UdIwSBhjCBg4AU5DS5
IFdU/QwYbikgtWvkU3fDwRihYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAL9KIB7Fgvg/MBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAG/N7ua8IE9IMyno0n5T57erBvLTOQ79fIJN
Mf+mKRM7qRRsdg/eumFft0rLOKo54pJ+Kim2cngCWNhkzctRHBV567AJNt4+ZDG5
hDgV0IxWO1+eaLE4qzqWP/9VrO+p3reuumgFZLVpvVpwXBBeBFUf2drUR14aWfI2
L/6VGINXYs7uP8v/2VBS7r6XZRnPBUy/R4hv5efYXnjwA9gq8+a3stC2ur8m5ySl
faKSwE4H320yAyaZWH4gpwUdbUlYgPHtm/ohRtiWPrN7KEG5Wq/REzMIjZCnxOfS
6KR6PNjlhxBsImQhmBvz6j5PLQxOxBZIpDoiK278e/1Wqm9LrBc=
-----END CERTIFICATE-----
```

カナダ (中部) - ca-central-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
```

```
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUIrLgixJJB5C4G8z6pZ5rB0JU2aQwDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI0MDQyOTE1MzU0M1oXDTI5MDQyODE1MzU0M1owXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQCHvRjf/0kStpJ248khtIaN8qkDN3tkw4VjvA9nvPl2anJO+eIB
UqPfQG09kZlwpWpmyO8bGB2RWqWxCwuB/dcnIob6w420k9WY5C0IIGtDRNauN3ku
vGXkw3HEnF0EjYr0pcyWUvByWY4KswZV42X7Y7XSS13hOIcL6NLA+H94/QIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUJdbMCBXKtvCcWdwUUizvtUF2
UTgwgZkGA1UdIwSBkTCBjoAUJdbMCBXKtvCcWdwUUizvtUF2UTihYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUIrLgixJJ
B5C4G8z6pZ5rB0JU2aQwEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQBHiQJmzyFAaSYs8SpiRijIDZW2RIo7qBKb/pI3rqK6yOWDlPuMr6yNI81D
IrKGGftg4Z+2KETYU4x76HSf0s//vfH3QA57qFaAwddhKYy4BhteFQl/Wex3xTlX
LiwI07kwJvJy3mS6UfQ4HcvZy219tY+0iyOWrz/jVxwq7TOkCw==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJAJNKhJhaJOuMMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNjA3Mjkx
MTM3MTdaGA8yMTk2MDEwMjExMzcxN1owXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAhDUh6j1ACSt057nSxAcwMaGr8Ez87VA2RW2HyY8l9XoHndnxmP50Cqld
+26AJtltlqHpI1YdtnZ6OrVgVhXcVtbvte0lZ3ldEzC3PMvmISBhHs6A3SWHA9ln
```

```
InHbToLX/SWqBHLOX78HkPRaG2k0COHpRy+fG9gvz8HCiQaXCbWNFDHZev9OToNI
xhXBVzIa3AgUnGMalCYZuh5AfVRCEeALG60kxMMC8IoAN7+HG+pMdqAhJxGUcMO0
LBvmTGGeWhi04MUZWfOkwn9JjQZuyLg6B1OD4Y6s0LB2P1MovmSJKGY4JcF8Qu3z
xxUbl7Bh9pvzFR5gJN1pjM2n3gJEPwIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQAJ
UNKM+gIIHNk0G0tzv6vZBT+o/vt+tIp8lEoZwaPQh1121iw/I7ZvhMLAigx7eyvf
IxUt9/nf8pxWaeGzi98RbSmbap+uxYRynqe1p5rifTamOsguuPrhVpl12OgRWLcT
rjg/K60UMXRsmg2w/cxV45pUBcyVb5h6Op5uEVAVq+CVns13ExiQL6kk3guG4+Yq
LvP1p4DZfeC33a2Rfre2IHLsJH5D4SdWcYqBsfTpf3FQThH0l0KoacGrXtsedsxs
9aRd7OzuSEJ+mBxmzxSjSwM84Ooh78DjkdpQgv967p3d+8NiSLt3/n7MgnUy6WwB
KtDujDnB+ttEHwRRngX7
-----END CERTIFICATE-----
```

カナダ西部 (カルガリー) — ca-west-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7zCCAq
+gAwIBAgIGAYPouptUMAkGByqGSM44BAMwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgMEFdhc2hpbmd0b24gU3RhdGUxED
U4EddRIpUt9KnC7s5Of2EbdSPO9EAMMeP4C2USZpRV1AIlH7WT2NWPq/
xfW6MPbLm1Vs14E7gB00b/JmYLdrmVClpJ+f6AR7ECLCT7up1/63xhv4O1fnxqimFQ8E
+4P208UewwI1VBNaFpEy9nXzrith1yrv8iIDGZ3RSAHHAhUAl2BQjxUjC8yykrmCouuEC/
BYHPUCgYEA9+GghdabPd7LvKtcNrhXuXmUr7v6OuqC+VdMCz0HgmdRWVeOutRZT
+ZxBxCBgLRJFnEj6EwoFhO3zwkyjMim4TwWeotUfI0o4KOuHiuzpnWRbqN/C/ohNWLx
+2J6ASQ7zKTxvqhRkImog9/hWuWfBpKLZl6Ae1UlZAFMO/7PSSoDgYQAAoGAMITzTJUa6cBsIfdHN69zW/
aHjUB4r1ZfKb1FMhIp9EZtEf5n+06oXjUG2+dKRSlFQeEK333ehNZsPd6uqey6TYKtHpFb5XRLS8BpqB
+7gnbAd0CBZM5o4NWesSQ1GLnTdQcGZkYG/
QESkbadoCXQTifCujJE682hTDLIVt1d4ewwCQYHKoZIzjgEAwMvADAsAhRJc4gRS/HWTkCR2MESaQEe/
jOMNQIUNoTwLvuPrmGPupPlGiHe0veZiO8=
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUCVFIhsz9lHA88TjOEChaY+KhHjkwDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI2MDYxNzA5MjE0M1oXDTMxMDYxNjA5MjE0M1owXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQDK1kIcG5Q6adBXQM75GldfTSiXl7tn54p10TnspI0ErDdb2B6q
2Ji/v4XBVH13ZCMgqlRHMqV8AWI5iO6gFn2A9sN3AZXTMqwtZeiDdebq3k6Wt7ie
```

```
YvpXTg0qvgsjQIovRZWaBDBJy9x8C2hW+w9lMQjFHkJ7Jy/PHCJ69EzebQIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUwCJ9341XAe3MNiXb/yszt4om
ccIwgZkGA1UdIwSBkTCBjoAUwCJ9341XAe3MNiXb/yszt4omccKhYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUCVFIhsz9
lHA88TjOEChaY+KhHjkwEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQCuorFtp2WcEreUMNlL44zF8JDSkju0ik27J4OdKRow6tPvqYDmyGTvybWh
N4GtOuNEhWzmj4E7eMHzRslxaHXNhxIa8l+M/GRaq9jJwAb6aU9YWcVH9FT/6Jr0
q6myzukFBNLZ9zCrrclp8dOFdGf0310RDpmGmF3qzU0MlHh+Iw==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJALyTn5IHrIZjMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yMzEyMDcx
NTM3MDFaGA8yMjAzMDUxMzE1MzcwMVowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEA1GP5os424BJmGPCKOSgOclP7lzUiB85du03M4hfjzSOszsBpmBGFDLz1
owYHtIxlq3+Vi1Lt5Qlx3id/ov1QyaBPFWXVeklHVXy9vieCcI3TdjGjTl1W/8MM
m3X26QPcsnHM/Kk2wJ7s186MrqmdSsp3SCPpxv4vEG2Q9yR2bXY41hpc2rWlW8qU
D0JGXluvmmAdFnto20l1XWZ6xFen1h60DRugek/ufCbN+ljKy0xLqPoavH0Ybjsb
UpsAsBs7phaoN+X/5hIERfbp5LfVnqq54pNG5KNu4KynfW9+kA/WS4cJ6FTTN5t+
y0P1HvcL+BL2RuDy6T2bB21xw5WqtQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQURTvu/Dd4zDnmS5G5CfVLnmUBNOswgY4GA1UdIwSBhjCBg4AURTvu
/Dd4zDnmS5G5CfVLnmUBNOuhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJALyTn5IHrIZjMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAFt523A3Aug6/F8xxyITgA8gkU0btFhlXNSP
U4U2OQ9n0tWI9WqnKNWH3KBxwY5EPitU6b3LM4xc9lDWpz7h2Pto+WhxP9LVKe6f
r8r7teTLCVZ7cfYZHzHg+flZjVpAgzE5BVfrRlj3QKpv0hYT3J1wMtI++Vorq5Nf
aPjzedehJLhmZVALwnfqfLrgv6/gmraP9Vmoa8U4D6AljNiQGYaLwyoPoRm3bUs2
v1Mh9GkEQ1b9+1pFXcqgzJJTGRuiPCyPbECI79FAnx5JM/CkGJV8HlOmjIW1qkK1
Y2qT7wzErrKLJyB53Pw15BdIM1onbDAQreZb0yZQLdoEl/tx7Uk=
-----END CERTIFICATE-----
```

中国 (北京) - cn-north-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIDNjCCAh4CCQD3yZ1w1AVkTzANBgkqhkiG9w0BAQsFADBcMQswCQYDVQQGEwJV
UzEZMBcGA1UECBMQV2FzaGluZ3RvbiBTdGF0ZTEQMA4GA1UEBxMHU2VhdHRsZTEg
MB4GA1UEChMXQW1hem9uIFdlYiBTZXJ2aWNlcyBMTEMwIBcNMTUwNTEzMDk1OTE1
WhgPMjE5NDEwMTYwOTU5MTVaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNo
aW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24g
V2ViIFNlcnZpY2VzIExMQzCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEB
AMWk9vyppSmDU3AxZ2Cy2bvKeK3F1UqNpMuyeriizi+NTsZ8tQqtNloaQcqhto/l
gsw9+QSnEJeYWnmivJWOBdn9CyDpN7cpHVmeGgNJL2fvImWyWe2f2Kq/BL9l7N7C
P2ZT52/sH9orlck1n2zO8xPi7MItgPHQwu3OxsGQsAdWucdxjHGtdchulpo1uJ31
jsTAPKZ3p1/sxPXBBAgBMatPHhRBqhwHO/Twm4J3GmTLWN7oVDds4W3bPKQfnw3r
vtBj/SM4/IgQ3xJslFcl90TZbQbgxIi88R/gWTbs7GsyT2PzstU30yLdJhKfdZKz
/aIzraHvoDTWFaOdy0+OOaECAwEAATANBgkqhkiG9w0BAQsFAAOCAQEAdSzN2+0E
V1BfR3DPWJHWRf1b7zl+1X/ZseW2hYE5r6YxrLv+1VPf/L5I6kB7GEtqhZUqteY7
zAceoLrVu/7OynRyfQetJVGichaaxLNM3lcr6kcxOowb+WQQ84cwrB3keykH4gRX
KHB2rlWSxta+2panSEO1JX2q5jhcFP90rDOtZjlpYv57N/Z9iQ+dvQPJnChdq3BK
5pZlnIDnVVxqRike7BFy8tKyPj7HzoPEF5mh9Kfnn1YoSVu+61lMVv/qRjnyKfS9
c96nE98sYFj0ZVBzXw8Sq4Gh8FiVmFHbQp1peGC19idOUqxPxWsasWxQXO0azYsP
9RyWLHKxH1dMuA==
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDCzCCAnSgAwIBAgIJALSOMbOoU2svMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0yMzA3MDQw
ODM1MzlaFw0yODA3MDIwODM1MzlaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQzCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEA
uhhUNlqAZdcWWB/OSDVDGk3OA99EFzOn/mJlmciQ/Xwu2dFJWmSCqEAE6gjufCjQ
q3voxAhC2CF+elKtJW/C0Sz/LYo60PUqd6iXF4h+upB9HkOOGuWHXsHBTsvgkgGA
1CGgel4U0Cdq+23eANr8N8m28UzljjSnTlrYCHtzN4sCAwEAAaOB1DCB0TALBgNV
HQ8EBAMCB4AwHQYDVR0OBBYEFBkZu3wT27NnYgrfH+xJz4HJaNJoMIGOBgNVHSME
gYYwgYOAFBkZu3wT27NnYgrfH+xJz4HJaNJooWCkXjBcMQswCQYDVQQGEwJVUzEZ
MBcGA1UECBMQV2FzaGluZ3RvbiBTdGF0ZTEQMA4GA1UEBxMHU2VhdHRsZTEgMB4G
A1UEChMXQW1hem9uIFdlYiBTZXJ2aWNlcyBMTEOCCQC0jjGzqFNrLzASBgNVHRMB
Af8ECDAGAQH/AgEAMA0GCSqGSIb3DQEBCwUAA4GBAECji43p+oPkYqmzll7e8Hgb
oADS0ph+YUz5P/bUCm61wFjlxaTfwKcuTR3ytj7bFLoW5Bm7Sa+TCl3lOGb2taon
```

```
2h+9NirRK6JYk87LMNvbS40HGPFumJL2NzEsGUeK+MRiWu+Oh5/lJGii3qw4YByx
SUDlRyNy1jJFstEZjOhs
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJAOtrM5XLDSjCMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTA4MTQx
MDAxNDJaGA8yMTk1MDExNzEwMDE0MlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAvVBz+WQNdPiM9S+aUULOQEriTmNDUrjLWLr7SfaOJScBzis5D5ju0jh1
+qJdkbuGKtFX5OTWTm8pWhInX+hIOoS3exC4BaANoa1A3o6quoG+Rsv72qQf8LLH
sgEi6+LMlCN9TwnRKOToEabmDKorss4zFl7VSsbQJwcBSfOcIwbdRRaW9Ab6uJHu
79L+mBR3Ea+G7vSDrVIA8goAPkae6jY9WGw9KxsOrcvNdQoEkqRVtHo4bs9fMRHU
Etphj2gh4ObXlFN92VtvzD6QBs3CcoFWgyWGvzg+dNG5VCbsiiuRdmii3kcijZ3H
Nv1wCcZoEAqH72etVhsuvNRC/xAP8wIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQA8
ezx5LRjzUU9EYWYhyYIEShFlP1qDHs7F4L46/5lc4pL8FPoQm5CZuAF31DJhYi/b
fcV7i3n++/ymQbCLC6kAg8DUB7NrcROll5ag8d/JXGzcTCnlDXLXx1905fPNa+jI
0q5quTmdmiSi0taeaKZmyUdhrB+a7ohWdSdlokEIOtbH1P+g5yll3bI2leYE6Tm8
LKbyfK/532xJPqO9abx4Ddn89ZEC6vvWVNDgTsxERg992Wi+/xoSw3XxkgAryIv1
zQ4dQ6irFmXwCWJqc6kHg/M5W+z60S/94+wGTXmp+19U6Rkq5jVMLh16XJXrXwHe
4KcgIS/aQGVgjM6wivVA
-----END CERTIFICATE-----
```

中国 (寧夏) – cn-northwest-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIDNjCCAh4CCQD3yZ1w1AVkTzANBgkqhkiG9w0BAQsFADBcMQswCQYDVQQGEwJV
UzEZMBcGA1UECBMQV2FzaGluZ3RvbiBTdGF0ZTEQMA4GA1UEBxMHU2VhdHRsZTEg
MB4GA1UEChMXQW1hem9uIFdlYiBTZXJ2aWNlcyBMTEMwIBcNMTUwNTEzMDk1OTE1
WhgPMjE5NDEwMTYwOTU5MTVaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNo
aW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24g
V2ViIFNlcnZpY2VzIExMQzCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEB
AMWk9vyppSmDU3AxZ2Cy2bvKeK3F1UqNpMuyeriizi+NTsZ8tQqtNloaQcqhto/l
gsw9+QSnEJeYWnmivJWOBdn9CyDpN7cpHVmeGgNJL2fvImWyWe2f2Kq/BL9l7N7C
P2ZT52/sH9orlck1n2zO8xPi7MItgPHQwu3OxsGQsAdWucdxjHGtdchulpo1uJ31
jsTAPKZ3p1/sxPXBBAgBMatPHhRBqhwHO/Twm4J3GmTLWN7oVDds4W3bPKQfnw3r
```

```
vtBj/SM4/IgQ3xJslFcl90TZbQbgxIi88R/gWTbs7GsyT2PzstU30yLdJhKfdZKz
/aIzraHvoDTWFaOdy0+OOaECAwEAATANBgkqhkiG9w0BAQsFAAOCAQEAdSzN2+0E
V1BfR3DPWJHWRf1b7zl+1X/ZseW2hYE5r6YxrLv+1VPf/L5I6kB7GEtqhZUqteY7
zAceoLrVu/7OynRyfQetJVGichaaxLNM3lcr6kcxOowb+WQQ84cwrB3keykH4gRX
KHB2rlWSxta+2panSEO1JX2q5jhcFP90rDOtZjlpYv57N/Z9iQ+dvQPJnChdq3BK
5pZlnIDnVVxqRike7BFy8tKyPj7HzoPEF5mh9Kfnn1YoSVu+61lMVv/qRjnyKfS9
c96nE98sYFj0ZVBzXw8Sq4Gh8FiVmFHbQp1peGC19idOUqxPxWsasWxQXO0azYsP
9RyWLHKxH1dMuA==
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDCzCCAnSgAwIBAgIJALSOMbOoU2svMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0yMzA3MDQw
ODM1MzlaFw0yODA3MDIwODM1MzlaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQzCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEA
uhhUNlqAZdcWWB/OSDVDGk3OA99EFzOn/mJlmciQ/Xwu2dFJWmSCqEAE6gjufCjQ
q3voxAhC2CF+elKtJW/C0Sz/LYo60PUqd6iXF4h+upB9HkOOGuWHXsHBTsvgkgGA
1CGgel4U0Cdq+23eANr8N8m28UzljjSnTlrYCHtzN4sCAwEAAaOB1DCB0TALBgNV
HQ8EBAMCB4AwHQYDVR0OBBYEFBkZu3wT27NnYgrfH+xJz4HJaNJoMIGOBgNVHSME
gYYwgYOAFBkZu3wT27NnYgrfH+xJz4HJaNJooWCkXjBcMQswCQYDVQQGEwJVUzEZ
MBcGA1UECBMQV2FzaGluZ3RvbiBTdGF0ZTEQMA4GA1UEBxMHU2VhdHRsZTEgMB4G
A1UEChMXQW1hem9uIFdlYiBTZXJ2aWNlcyBMTEOCCQC0jjGzqFNrLzASBgNVHRMB
Af8ECDAGAQH/AgEAMA0GCSqGSIb3DQEBCwUAA4GBAECji43p+oPkYqmzll7e8Hgb
oADS0ph+YUz5P/bUCm61wFjlxaTfwKcuTR3ytj7bFLoW5Bm7Sa+TCl3lOGb2taon
2h+9NirRK6JYk87LMNvbS40HGPFumJL2NzEsGUeK+MRiWu+Oh5/lJGii3qw4YByx
SUDlRyNy1jJFstEZjOhs
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJAPu4ssY3BlzcMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTEyMDMy
MTI5MzJaGA8yMTk1MDUwODIxMjkzMlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAsOiGi4A6+YTLzCdIyP8b8SCT2M/6PGKwzKJ5XbSBoL3gsnSWiFYqPg9c
uJPNbiy9wSA9vlyfWMd90qvTfiNrT6vewP813QdJ3EENZOx4ERcf/Wd22tV72kxD
```

```
yw1Q3I1OMH4bOItGQAxU5OtXCjBZEEUZooOkU8RoUQOU2Pql4NTiUpzWacNutAn5
HHS7MDc4lUlsJqbN+5QW6fFrcNG/0Mrib3JbwdFUNhrQ5j+Yq5h78HarnUivnX/3
Ap+oPbentv1qd7wvPJu556LZuhfqI0TohiIT1Ah+yUdN5osoaMxTHKKtf/CsSJ1F
w3qXqFJQA0VWsqjFyHXFI32I/GOupwIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQCn
Um00QHvUsJSN6KATbghowLynHn3wZSQsuS8E0COpcFJFxP2SV0NYkERbXu0n/Vhi
yq5F8v4/bRA2/xpedLWmvFs7QWlomuXhSnYFkd33Z5gnXPb9vRkLwiMSw4uXls35
qQraczUJ9EXDhrv7VmngIk9H3YsxYrlDGEqh/oz4Ze4ULOgnfkauanHikk+BUEsg
/jsTD+7e+niEzJPihHdsvKFDlud5pakEzyxovHwNJ1GS2I//yxrJFIL91mehjqEk
RLPdNse7N6UvSnuXcOokwu6l6kfzigGkJBxkcq4gre3szZFdCQcUioj7Z4xtuTL8
YMqfiDtN5cbD8R8ojw9Y
-----END CERTIFICATE-----
```

欧州 (フランクフルト) - eu-central-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUFD5GsmkxRuecttwsCG763m3u63UwDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI0MDQyOTE1NTUyOVoXDTI5MDQyODE1NTUyOVowXDELMAkGA1UEBhMCVVMx
```

```
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQCHvRjf/0kStpJ248khtIaN8qkDN3tkw4VjvA9nvPl2anJO+eIB
UqPfQG09kZlwpWpmyO8bGB2RWqWxCwuB/dcnIob6w420k9WY5C0IIGtDRNauN3ku
vGXkw3HEnF0EjYr0pcyWUvByWY4KswZV42X7Y7XSS13hOIcL6NLA+H94/QIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUJdbMCBXKtvCcWdwUUizvtUF2
UTgwgZkGA1UdIwSBkTCBjoAUJdbMCBXKtvCcWdwUUizvtUF2UTihYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUFD5Gsmkx
RuecttwsCG763m3u63UwEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQBBh0WaXlBsW56Hqk588MmJxsOrvcKfDjF57RgEDgnGnQaJcStCVWDO9UYO
JX2tdsPw+E7AjDqjsuxYaotLn3Mr3mK0sNOXq9BljBnWD4pARg89KZnZI8FN35HQ
O/LYOVHCknuPL123VmVRNs51qQA9hkPjvw21UzpDLxaUxt9Z/w==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAKD+v6LeR/WrMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTA4MTQw
OTA4MTlaGA8yMTk1MDExNzA5MDgxOVowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAka8FLhxs1cSJGK+Q+q/vTf8zVnDAPZ3U6oqppOW/cupCtpwMAQcky8DY
Yb62GF7+C6usniaq/9W6xPn/3o//wti0cNt6MLsiUeHqNl5H/4U/Q/fR+GA8pJ+L
npqZDG2tFi1WMvvGhGgIbScrjR4VO3TuKy+rZXMYvMRk1RXZ9gPhk6evFnviwHsE
jV5AEjxLz3duD+u/SjPp1vloxe2KuWnyC+EKInnka909sl4ZAUh+qIYfZK85DAjm
GJP4W036E9wTJQF2hZJrzsiB1MGyC1WI9veRISd30izZZL6VVXLXUtHwVHnVASrS
zZDVpzj+3yD5hRXsvFigGhY0FCVFnwIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQUxC2l6pvJaRflgu3MUdN6zTuP6YcwgY4GA1UdIwSBhjCBg4AUxC2l
6pvJaRflgu3MUdN6zTuP6YehYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAKD+v6LeR/WrMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAIK+DtbUPppJXFqQMv1f2Gky5/82ZwgbbfXa
HBeGSii55b3tsyC3ZW5ZlMJ7Dtnr3vUkiWbV1EUaZGOUlndUFtXUMABCb/coDndw
CAr53XTv7UwGVNe/AFO/6pQDdPxXn3xBhF0mTKPrOGdvYmjZUtQMSVb9lbMWCFfs
w+SwDLnm5NF4yZchIcTs2fdpoyZpOHDXy0xgxO1gWhKTnYbaZOxkJvEvcckxVAwJ
obF8NyJla0/pWdjhlHafEXEN8lyxyTTyOa0BGTuYOBD2cTYYynauVKY4fqHUkr3v
Z6fboaHEd4RFamShM8uvSu6eEFD+qRmvqlcodbpsSOhuGNLzhOQ=
-----END CERTIFICATE-----
```

欧州 (アイルランド) - eu-west-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUakDaQ1Zqy87Hy9ESXA1pFC116HkwDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI0MDQyOTE2MTgxMFoXDTI5MDQyODE2MTgxMFowXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQCHvRjf/0kStpJ248khtIaN8qkDN3tkw4VjvA9nvPl2anJO+eIB
UqPfQG09kZlwpWpmyO8bGB2RWqWxCwuB/dcnIob6w420k9WY5C0IIGtDRNauN3ku
vGXkw3HEnF0EjYr0pcyWUvByWY4KswZV42X7Y7XSS13hOIcL6NLA+H94/QIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUJdbMCBXKtvCcWdwUUizvtUF2
UTgwgZkGA1UdIwSBkTCBjoAUJdbMCBXKtvCcWdwUUizvtUF2UTihYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUakDaQ1Zq
y87Hy9ESXA1pFC116HkwEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQADIkn/MqaLGPuK5+prZZ5Ox4bBZLPtreO2C7r0pqU2kPM2lVPyYYydkvP0
lgSmmsErGu/oL9JNztDe2oCA+kNy17ehcsf8cw0uP861czNFKCeU8b7FgBbL+sIm
qi33rAq6owWGi/5uEcfCR+JP7W+oSYVir5r/yDmWzx+BVH5S/g==
```

```
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAOrmqHuaUt0vMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTEwMjkw
OTA2MTlaGA8yMTk1MDQwMzA5MDYxOVowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAjE7nVu+aHLtzp9FYV25Qs1mvJ1JXD7J0iQ1Gs/RirW9a5ZECCtc4ssnf
zQHq2JRVr0GRchvDrbm1HaP/avtFQR/Thvfltwu9AROVT22dUOTvERdkNzveoFCy
hf52Rqf0DMrLXG8ZmQPPXPDFAv+sVMWCDftcChxRYZ6mP9O+TpgYNT1krD5PdvJU
7HcXrkNHDYqbsg8A+Mu2hzl0QkvUET83Csg1ibeK54HP9w+FsD6F5W+6ZSHGJ88l
FI+qYKs7xsjJQYgXWfEt6bbckWs1kZIaIOyMzYdPF6ClYzEec/UhIe/uJyUUNfpT
VIsI5OltBbcPF4c7Y20jOIwwI2SgOQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQUF2DgPUZivKQR/Zl8mB/MxIkjZDUwgY4GA1UdIwSBhjCBg4AUF2Dg
PUZivKQR/Zl8mB/MxIkjZDWhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAOrmqHuaUt0vMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAGm6+57W5brzJ3+T8/XsIdLTuiBSe5ALgSqI
qnO5usUKAeQsa+kZIJPyEri5i8LEodh46DAF1RlXTMYgXXxl0YggX88XPmPtok17
l4hib/D9/lu4IaFIyLzYNSzsETYWKWoGVe7ZFz60MTRTwY2u8YgJ5dec7gQgPSGj
avB0vTIgoW41G58sfw5b+wjXCsh0nROon79RcQFFhGnvup0MZ+JbljyhZUYFzCli
31jPZiKzqWa87xh2DbAyvj2KZrZtTe2LQ48Z4G8wWytJzxEeZdREe4NoETf+Mu5G
4CqoaPR05KWkdNUdGNwXewydb3+agdCgfTs+uAjeXKNdSpbhMYg=
-----END CERTIFICATE-----
```

欧州 (ロンドン) - eu-west-2

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
```

```
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUCgCV/DPxYNND/swDgEKGiC5I+EwwDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI0MDQyOTE2MjkxNFoXDTI5MDQyODE2MjkxNFowXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQCHvRjf/0kStpJ248khtIaN8qkDN3tkw4VjvA9nvPl2anJO+eIB
UqPfQG09kZlwpWpmyO8bGB2RWqWxCwuB/dcnIob6w420k9WY5C0IIGtDRNauN3ku
vGXkw3HEnF0EjYr0pcyWUvByWY4KswZV42X7Y7XSS13hOIcL6NLA+H94/QIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUJdbMCBXKtvCcWdwUUizvtUF2
UTgwgZkGA1UdIwSBkTCBjoAUJdbMCBXKtvCcWdwUUizvtUF2UTihYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUCgCV/DPx
YNND/swDgEKGiC5I+EwwEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQATPu/sOE2esNa4+XPEGKlEJSgqzyBSQLQc+VWo6FAJhGG9fp7D97jhHeLC
5vwfmtTAfnGBxadfAOT3ASkxnOZhXtnRna460LtnNHm7ArCVgXKJo7uBn6ViXtFh
uEEw4y6p9YaLQna+VC8Xtgw6WKq2JXuKzuhuNKSFaGGw9vRcHg==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJANBx0E2bOCEPMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNjA4MTEx
NDU2NDJaGA8yMTk2MDExNTE0NTY0MlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEArYS3mJLGaMrh2DmiPLbqr4Z+xWXTzBWCjOwpsuHE9H6dWUUyl2Bgnu+Z
d8QvW306Yleec45M4F2RA3J4hWHtShzsMlOJVRt+YulGeTf9OCPr26QmIFfs5nD4
```

```
fgsJQEry2MBSGA9Fxq3Cw6qkWcrOPsCR+bHOU0XykdKl0MnIbpBf0kTfciAupQEA
dEHnM2J1L2iI0NTLBgKxy5PXLH9weX20BFauNmHH9/J07OpwL20SN5f8TxcM9+pj
Lbk8h1V4KdIwVQpdWkbDL9BCGlYjyadQJxSxz1J343NzrnDM0M4h4HtVaKOS7bQo
Bqt2ruopLRCYgcuFHck/1348iAmbRQIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQBG
wujwU1Otpi3iBgmhjMClgZyMMn0aQIxMigoFNqXMUNx1Mq/e/Tx+SNaOEAu0n2FF
aiYjvY0/hXOx75ewzZvM7/zJWIdLdsgewpUqOBH4DXFhbSk2TxggSPb0WRqTBxq5
Ed7F7+7GRIeBbRzdLqmISDnfqey8ufW0ks51XcQNomDIRG5s9XZ5KHviDCar8FgL
HngBCdFI04CMagM+pwTO9XN1Ivt+NzUj208ca3oP1IwEAd5KhIhPLcihBQA5/Lpi
h1s3170z1JQ1HZbDrH1pgp+8hSI0DwwDVb3IIH8kPR/J0Qn+hvOl2HOpaUg2Ly0E
pt1RCZe+W7/dF4zsbqwK
-----END CERTIFICATE-----
```

欧州 (ミラノ) - eu-south-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAqwCCQCMElHPdwG37jAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xOTA0MjkyMDM1MjJaFw00
NTA0MjkyMDM1MjJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbYwggErBgcqhkjOOAQBMIIBHgKBgQDAkoL4YfdMI/MrQ0oL
NPfeEk94eiCQA5xNOnU7+2eVQtEqjFbDADFENh1p3sh9Q9OoheLFH8qpSfNDWn/0
ktCS909ApTY6Esx1ExjGSeQq/U+SC2JSuuTT4WFMKJ63a/czMtFkEPPnVIjJJJmT
HJSKSsVUgpdDIRvJXuyB0zdB+wIVALQ3OLaVGdlPMNfS1nD/Yyn+32wnAoGAPBQ3
7XHg5NLOS4326eFRUT+4ornQFjJjP6dp3pOBEzpImNmZTtkCNNUKE4Go9hv5T4lh
R0pODvWv0CBupMAZVBP9ObplXPCyEIZtuDqVa7ukPOUpQNgQhLLAqkigTyXVOSmt
ECBj9tu5WNP/x3iTZTHJ+g0rhIqpgh012UwJpKADgYQAAoGAV1OEQPYQUg5/M3xf
6vE7jKTxxyFWEyjKfJK7PZCzOIGrE/swgACy4PYQW+AwcUweSlK/Hx2OaZVUKzWo
wDUbeu65DcRdw2rSwCbBTU342sitFo/iGCV/Gjf+BaiAJtxniZze7J1ob8vOBeLv
uaMQmgOYeZ5e0fl04GtqPl+lhcQwCQYHKoZIzjgEAwMwADAtAhQdoeWLrkm0K49+
AeBK+j6m2h9SKQIVAIBNhS2a8cQVABDCQXVXrc0tOmO8
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICNjCCAZ+gAwIBAgIJAOZ3GEIaDcugMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xOTEwMjQx
NTE5MDlaGA8yMTk5MDMyOTE1MTkwOVowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
```

```
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKB
gQCjiPgW3vsXRj4JoA16WQDyoPc/eh3QBARaApJEc4nPIGoUolpAXcjFhWplo2O+
ivgfCsc4AU9OpYdAPha3spLey/bhHPRi1JZHRNqScKP0hzsCNmKhfnZTIEQCFvsp
DRp4zr91/WS06/flJFBYJ6JHhp0KwM81XQG59lV6kkoW7QIDAQABMA0GCSqGSIb3
DQEBCwUAA4GBAGLLrY3P+HH6C57dYgtJkuGZGT2+rMkk2n81/abzTJvsqRqGRrWv
XRKRXlKdM/dfiuYGokDGxiC0Mg6TYy6wvsR2qRhtXW1OtZkiHWcQCnOttz+8vpew
wx8JGMvowtuKB1iMsbwyRqZkFYLcvH+Opfb/Aayi20/ChQLdI6M2R5VU
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJAO/+DgYF78KwMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xOTA0Mjky
MDM1MjJaGA8yMTk4MTAwMjIwMzUyMlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAv1ZLV+Z/P6INq+R1qLkzETBg7sFGKPiwHekbpuB6lrRxKHhj8V9vaReM
lnv1Ur5LAPpMPYDsuJ4WoUbPYAqVqyMAo7ikJHCCM1cXgZJefgN6z9bpS+uA3YVh
V/0ipHh/X2hc2S9wvxKWiSHu6Aq9GVpqL035tJQD+NJuqFd+nXrtcw4yGtmvA6wl
5Bjn8WdsP3xOTKjrByYY1BhXpP/f1ohU9jE9dstsRXLa+XTgTPWcWdCS2oRTWPGR
c5Aeh47nnDsyQfP9gLxHeYeQItV/BD9kU/2Hn6mnRg/B9/TYH8qzlRTzLapXp4/5
iNwusrTNexGl8BgvAPrfhjDpdgYuTwIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQB7
5ya11K/hKgvaRTvZwVV8GlVZt0CGPtNvOi4AR/UN6TMm51BzUB5nurB4z0R2MoYO
Uts9sLGvSFALJ4otoB77hyNpH3drttU1CVVwal/yK/RQLSon/IoUkaGEbqalu+mH
nYad5IG4tEbmepX456XXcO58MKmnczNbPyw3FRzUZQtI/sf94qBwJ1Xo6XbzPKMy
xjL57LHIZCssD+XPifXay69OFlsCIgLim11HgPkRIHEOXLSf3dsW9r+4CjoZqB/Z
jj/P4TLCxbYCLkvglwaMjgEWF40Img0fhx7yT2X92MiSrs3oncv/IqfdVTiN8OXq
jgnq1bf+EZEZKvb6UCQV
-----END CERTIFICATE-----
```

欧州 (パリ) - eu-west-3

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
```

```
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUaC9fX57UDr6u1vBvsCsECKBZQyIwDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI0MDQyOTE2MzczOFoXDTI5MDQyODE2MzczOFowXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQCHvRjf/0kStpJ248khtIaN8qkDN3tkw4VjvA9nvPl2anJO+eIB
UqPfQG09kZlwpWpmyO8bGB2RWqWxCwuB/dcnIob6w420k9WY5C0IIGtDRNauN3ku
vGXkw3HEnF0EjYr0pcyWUvByWY4KswZV42X7Y7XSS13hOIcL6NLA+H94/QIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUJdbMCBXKtvCcWdwUUizvtUF2
UTgwgZkGA1UdIwSBkTCBjoAUJdbMCBXKtvCcWdwUUizvtUF2UTihYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUaC9fX57U
Dr6u1vBvsCsECKBZQyIwEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQCARv1bQEDaMEzYI0nPlu8GHcMXgmgA94HyrXhMMcaIlQwocGBs6VILGVhM
TXP2r3JFaPEpmXSQNQHvGA13clKwAZbni8wtzv6qXb4L4muF34iQRHF0nYrEDoK7
mMPR8+oXKKuPO/mv/XKo6XAV5DDERdSYHX5kkA2R9wtvyZjPnQ==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJALWSfgHuT/ARMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNzA1MzEx
MTE4MTZaGA8yMTk2MTEwMzExMTgxNlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
```

```
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAy5V7KDqnEvF3DrSProFcgu/oL+QYD62b1U+Naq8aPuljJe127Sm9WnWA
EBdOSASkOaQ9fzjCPoG5SGgWKxYoZjsevHpmzjVv9+Ci+F57bSuMbjgUbvbRIFUB
bxQojVoXQPHgK5v433ODxkQ4sjRyUbf4YV1AFdfU7zabC698YgPVOExGhXPlTvco
8mlc631ubw2g52j0lzaozUkHPSbknTomhQIvO6kUfX0e0TDMH4jLDG2ZIrUB1L4r
OWKG4KetduFrRZyDHF6ILZu+s6ywiMicUd+2UllDFC6oas+a8D11hmO/rpWU/ieV
jj4rWAFrsebpn+Nhgy96iiVUGS2LuQIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQDE
iYv6FQ6knXCg+svlcaQG9q59xUC5z8HvJZ1+SxzPKKC4PKQdKvIIfE8GxVXqlZG1
cl5WKTFDMapnzb9RV/DTaVzWx3cMYT77vm1Hl1XGjhx611CGcENH1egI3lOTILsa
+KfopuJEQQ9TDMAIkGjhA+KieU/U5Ctv9fdej6d0GC6OEuwKkTNzPWue6UMq8d4H
2xqJboWsE1t4nybEosvZfQJcZ8jyIYcYBnsG13vCLM+ixjuU5MVVQNMY/gBJzqJB
V+U0QiGiuT5cYgY/QihxdHt99zwGaE0ZBC7213NKrlNuLSrqhDI2NLu8NsExqOFy
OmY0v/xVmQUQl26jJXaM
-----END CERTIFICATE-----
```

欧州 (スペイン) - eu-south-2

DSA

```
-----BEGIN CERTIFICATE-----
MIIC8DCCAq
+gAwIBAgIGAXjwLk46MAkGByqGSM44BAMwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgMEFdhc2hpbmd0b24gU3RhdGUxED
U4EddRIpUt9KnC7s5Of2EbdSPO9EAMMeP4C2USZpRV1AIlH7WT2NWPq/
xfW6MPbLm1Vs14E7gB00b/JmYLdrmVClpJ+f6AR7ECLCT7up1/63xhv4O1fnxqimFQ8E
+4P208UewwI1VBNaFpEy9nXzrith1yrv8iIDGZ3RSAHHAhUAl2BQjxUjC8yykrmCouuEC/
BYHPUCgYEA9+GghdabPd7LvKtcNrhXuXmUr7v6OuqC+VdMCz0HgmdRWVeOutRZT
+ZxBxCBgLRJFnEj6EwoFhO3zwkyjMim4TwWeotUfI0o4KOuHiuzpnWRbqN/C/ohNWLx
+2J6ASQ7zKTxvqhRkImog9/hWuWfBpKLZl6Ae1UlZAFMO/7PSSoDgYQAAoGAGG2m8EKmaf5qQqj3Z
+rzSaTaXE3B/R/4A2VuGqRYR7MljPtwdmU6/3CPjCACcZmTIcOAKbFiDHqadQgBZXfzGpzw8Zo
+eYmmk5fXycgnj57PYH1dIWU6I7mCbAah5MZMcmHaTmIsomGrhcnWB8d8qOU7oZ0UWK4lbiAQs1MihoUwCQYHKoZIzjg
WmbaU7YM5GwCFCvIJOesO5hZ8PHC52dAR8WWC6oe
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUbtiu7xW+c3P65VHqW/gxSFWHT8kwDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI2MDYwOTA4NDMzMVoXDTMxMDYwODA4NDMzMVowXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
```

```
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQDB/VvR1+45Aey5zn3vPk6xBm5o9grSDL6D2iAuprQnfVXn8CIb
SDbWFhA3fi5ippjKkh3sl8VyCvCOUXKdOaNrYBrPRkrdHdBuL2Tc84RO+3m/rxIU
Z2IK1fDlC6sWAjddf6sBrV2w2a78H0H8EwuwiSgttURBjwJ7KPPJCqaqrQIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUcZuHrkKFfuCOW/tfRxCPB7Ro
XQQwgZkGA1UdIwSBkTCBjoAUcZuHrkKFfuCOW/tfRxCPB7RoXQShYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUbtiu7xW+
c3P65VHqW/gxSFWHT8kwEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQANhLKjQrxeWPzqlTkLK8b44iyITsV0E+o8+gJN6tKmC0Cgapp1+rCcYDzt
4R3uglYdoRDRrEPJVLjS8XcX5cefteIJRKJGm06d+fTvgzVY11IXo77tnzJF356m
Nwunpudg2r1diQCWQKKftqbdTsQxPGPVTVOFv+tXyW1s1TYj9Q==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJALWSmO6DvSpQMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yMjA3MTgx
MzU4NDNaGA8yMjAxMTIyMjEzNTg0M1owXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAuAAhuSpsHC00/fD2zNlBDpNLRndi9qbHsNeuz3WqN7Samj2aSrM2hS+i
hUxx0BspZj0tZCOsbpPZ+i74NOEQtFeqQoEGvKhB1nJiF4y5I81HDhs5qHvoIivm
7rbbik3zgm1PqS/DmDjVQaXPcD31Rd9ILwBmWEwJqHigyNVlxYtCzTQcrlBrvNZM
dnNgCDAdX/HBEFxx9O12xeu0bSt0s+PJWZ1RTbYrNe7LIH6ntUqHxP/ziQ5trXEZ
uqy7aWk1L8uK4jmyNph0lbaqBa3Y6pYmU1nC27UE4i3fnPB0LSiAr+SrwVvX1g4z
ilo8kr+tbIF+JmcgYLBvO8Jwp+EUqQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQUwvGzKJL9A5LReJ4Fxo5K6I20xcowgY4GA1UdIwSBhjCBg4AUwvGz
KJL9A5LReJ4Fxo5K6I20xcqhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJALWSmO6DvSpQMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAJAZd31jyoTGLawAD2+v/vQsaB9vZIx5EImi
G8YGkd61uFWeNhAmtrwyE/i6FDSIphDrMHBkvw/D3BsqK+Ev/JOK/VYuaYDx/8fp
H4cwp9jC57CXzdIDREWNf6M9PsHFg2WA9XNNtClOZL5WJiJwel8eDSg+sqJUxE0l
MW+QChq/20F6niyaRK4bXrZq14as7h+F9u3A9xHEOVP7Zk9C2ehrBXzCMLSDt3GV
fEuMea2RxMhozwz34Hkdb6j18qoCfygubulovRNQjKw/cEmgPR16KfZPP5caILVt
9qkYPvePmbiVswZDee73cDymJYxLqILpOZwyXvUH8StiH42FHZQ=
-----END CERTIFICATE-----
```

欧州 (ストックホルム) - eu-north-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUN1c9U6U/xiVDFgJcYKZB4NkH1QEwDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI0MDQyOTE2MDYwM1oXDTI5MDQyODE2MDYwM1owXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQCHvRjf/0kStpJ248khtIaN8qkDN3tkw4VjvA9nvPl2anJO+eIB
UqPfQG09kZlwpWpmyO8bGB2RWqWxCwuB/dcnIob6w420k9WY5C0IIGtDRNauN3ku
vGXkw3HEnF0EjYr0pcyWUvByWY4KswZV42X7Y7XSS13hOIcL6NLA+H94/QIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUJdbMCBXKtvCcWdwUUizvtUF2
UTgwgZkGA1UdIwSBkTCBjoAUJdbMCBXKtvCcWdwUUizvtUF2UTihYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUN1c9U6U/
xiVDFgJcYKZB4NkH1QEwEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQBTIQdoFSDRHkpqNPUbZ9WXR2O5v/9bpmHojMYZb3Hw46wsaRso7STiGGX/
tRqjIkPUIXsdhZ3+7S/RmhFznmZc8e0bjU4n5vi9CJtQSt+1u4E17+V2bF+D3h/7
wcfE0l3414Q8JaTDtfEf/aF3F0uyBvr4MDMd7mFvAMmDmBPSlA==
```

```
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJALc/uRxg++EnMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xODA0MTAx
NDAwMTFaGA8yMTk3MDkxMzE0MDAxMVowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAzwCGJEJIxqtr2PD2a1mA6LhRzKhTBa1AZsg3eYfpETXIVlrpojMfvVoN
qHvGshWLgrGTT6os/3gsaADheSaJKavxwX3X6tJA8fvEGqr3a1C1MffH9hBWbQqC
LbfUTAbkwis4GdTUwOwPjT1Cm3u9R/VzilCNwkj7iQ65AFAI8Enmsw3UGldEsop4
yChKB3KW3WI0FTh0+gD0YtjrqqYJxpGOYBpJp5vwdd3fZ4t1vidmDMs7liv4f9Bx
p0oSmUobU4GUlFhBchK1DukICVQdnOVzdMonYm7s+HtpFbVHR8yf6QoixBKGdSal
mBf7+y0ixjCn0pnC0VLVooGo4mi17QIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQDG
4ONZiixgk2sjJctwbyD5WKLTH6+mxYcDw+3y/F0fWz561YORhP2FNnPOmEkf0Sl/
Jqk4svzJbCbQeMzRoyaya/46d7UioXMHRZam5IaGBhOdQbi97R4VsQjwQj0RmQsq
yDueDyuKTwWLK9KnvI+ZA6e6bRkdNGflK4N8GGKQ+fBhPwVELkbT9f16OJkezeeN
S+F/gDADGJgmPXfjogICb4Kvshq0H5Lm/xZlDULF2g/cYhyNY6EOI/eS5m1I7R8p
D/m6WoyZdpInxJfxW616OMkxQMRVsruLTNGtby3u1g6ScjmpFtvAMhYejBSdzKG4
FEyxIdEjoeO1jhTsck3R
-----END CERTIFICATE-----
```

欧州 (チューリッヒ) - eu-central-2

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7zCCAq
+gAwIBAgIGAXjXiKJnMAkGByqGSM44BAMwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgMEFdhc2hpbmd0b24gU3RhdGUxED
U4EddRIpUt9KnC7s5Of2EbdSPO9EAMMeP4C2USZpRV1AIlH7WT2NWPq/
xfW6MPbLm1Vs14E7gB00b/JmYLdrmVClpJ+f6AR7ECLCT7up1/63xhv4O1fnxqimFQ8E
+4P208UewwI1VBNaFpEy9nXzrith1yrv8iIDGZ3RSAHHAhUAl2BQjxUjC8yykrmCouuEC/
BYHPUCgYEA9+GghdabPd7LvKtcNrhXuXmUr7v6OuqC+VdMCz0HgmdRWVeOutRZT
+ZxBxCBgLRJFnEj6EwoFhO3zwkyjMim4TwWeotUfI0o4KOuHiuzpnWRbqN/C/ohNWLx
+2J6ASQ7zKTxvqhRkImog9/hWuWfBpKLZl6Ae1UlZAFMO/7PSSoDgYQAAoGAYNjaCNg/
cfgQOOl1BUj5ClUulqwZ9Q+SfDzPZh9D2C0VbiRANiZoxrV8RdgmzzC5T7VcriVwjwvta2Ch//
b+sZ86E5h0XWWr+BeEjD9cu3eDj12XB5sWEbNHNx49p5Tmtu5r2LDtlL8X/
Rpfalu2Z2OJgjFJWGf7hRwxe456n
+lowCQYHKoZIzjgEAwMvADAsAhRChsLcj4U5CVb2cp5M0RE1XbXmhAIUeGSnH+aiUQIWmPEFja+itWDufIk=
```

```
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDCzCCAnSgAwIBAgIJAK+7ZflHYRFQMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0yNjA1MjYx
MDU0MDBaFw0zMTA1MjUxMDU0MDBaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQzCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEA
tpnRnabOUc9o82HBjbIBE0xlLYSUa1akp1FiV05VWiG5BjyyjujqwH1ihadrIES
fLy7I6AVUeXBElSu3LitcCLY7ycov6KSmQFLrCcdoaYFN6VB4ZafQE4f1XlQ+fNo
03fSGPvLeTA2umd3xpYgchIl89CxB3YvDAWnC0RLtU8CAwEAAaOB1DCB0TALBgNV
HQ8EBAMCB4AwHQYDVR0OBBYEFDFYNKA6vm8m8PtCqgIoU4I3+EpSMIGOBgNVHSME
gYYwgYOAFDFYNKA6vm8m8PtCqgIoU4I3+EpSoWCkXjBcMQswCQYDVQQGEwJVUzEZ
MBcGA1UECBMQV2FzaGluZ3RvbiBTdGF0ZTEQMA4GA1UEBxMHU2VhdHRsZTEgMB4G
A1UEChMXQW1hem9uIFdlYiBTZXJ2aWNlcyBMTEOCCQCvu2X5R2ERUDASBgNVHRMB
Af8ECDAGAQH/AgEAMA0GCSqGSIb3DQEBCwUAA4GBADsVuZ+wv4qEfSnGx+MH8s7p
oBWc/w6DTMCltWl7+q6a7C7uTIymgfsbug31ClwB7O3dhy26Ae9gNzXfpRnWN7iK
chhz057HHESzeogH1rTHD2RkJE5z6KCC0F8JgEg+JQSSCuTFKzUuBkW4S8WMQ3Tj
kCniK/KM2bEKllCqvINx
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJALvT012pxTxNMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yMjA3MTgx
NTEyMDdaGA8yMjAxMTIyMjE1MTIwN1owXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAyn+Lsnq1ykrfYlZkk6aAAYNReNd9Iw8AUwCBkgOr2eBiBBepYxHwU85N
++moQ+j0EV2VaahBeTLShGZZS1HsyK8+cYT2QzpghIoamcYhrPXyIxlWiRQlaqSg
OFiE9bsqL3rCF5Vz+tOiTe5W/7ojfOFls6++g7ZpobwJlpMbuJepqyeHMPyjvO5A
age81lJewc4bxo2ntaW0HCqNksqfYB78j6X6kn3PFpX7FaYAwZA+Xx6C7UCY7rNi
UdQzfAo8htfJi4chz7frpUdQ9kl3IOQrsLshBB5fFUjl09NiFipCGBwi+8ZMeSn1
5qwBI01BWXPfG7WX6Owyjhmh6JtE1wIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQU8HN4vvJrsZgPQeksMBgJb9xR1yYwgY4GA1UdIwSBhjCBg4AU8HN4
vvJrsZgPQeksMBgJb9xR1yahYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJALvT012pxTxNMBIGA1UdEwEB/wQIMAYBAf8C
```

```
AQAwDQYJKoZIhvcNAQELBQADggEBAGlHYDtcHpfBvdHx9HeQE8HgNugJUPdEqxun
t9U33p8VFrs+uLPtrOd9HDJEGvvs5h84EUie/oGJxRt7V1Vlid1PvHf6cRmpjgqY
YdggAVkZtY/PnFVmzf2bMVlSQPrqCl7U0zaw2Kvnj4zgX0rZyCetgrRZSUSxotyp
978Wy9ccXwVSeYG/YAr5rJpS6ZH7eRQvUY0IzwFNeaOPgOTEVpcjWlV6+MQEvsEx
W85q+s6AVr49eppEx8SLJsl0C23yB+L+t32tAveQImRWtJMpzZ5cxh/sYgDVeoC0
85H1NK/7H9fAzT1cPu1oHSnB0xYzzHGOAmXmusMfwUk8fL1RQkE=
-----END CERTIFICATE-----
```

イスラエル (テルアビブ) - il-central-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7zCCAq+gAwIBAgIGAX0QPi
+9MAkGByqGSM44BAMwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgMEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcMB1NlYX
U4EddRIpUt9KnC7s5Of2EbdSPO9EAMMeP4C2USZpRV1AIlH7WT2NWPq/
xfW6MPbLm1Vs14E7gB00b/JmYLdrmVClpJ+f6AR7ECLCT7up1/63xhv4O1fnxqimFQ8E
+4P208UewwI1VBNaFpEy9nXzrith1yrv8iIDGZ3RSAHHAhUAl2BQjxUjC8yykrmCouuEC/
BYHPUCgYEA9+GghdabPd7LvKtcNrhXuXmUr7v6OuqC+VdMCz0HgmdRWVeOutRZT
+ZxBxCBgLRJFnEj6EwoFhO3zwkyjMim4TwWeotUfI0o4KOuHiuzpnWRbqN/C/ohNWLx
+2J6ASQ7zKTxvqhRkImog9/
hWuWfBpKLZl6Ae1UlZAFMO/7PSSoDgYQAAoGAbazCL5XXyPmcw3+oMYQUF5/9YogW6D0FZbYuyPgjOoUwWdl6fjlzWca
pq+l1ezuK2DFOzNTEyPEwwCQYHKoZIzjgEAwMvADAsAhRt1jkpXsvrS
+xTo2M9h2s2uLAhEQIUOZ2FcnTSrshF2EIdixZZwtNv66Q=
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUPgjnsnqT7OaGnZgGF7/43ihPh+0wDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI2MDYxMjA5MjQyOVoXDTMxMDYxMTA5MjQyOVowXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQDrc24u3AgFxnoPgzxR6yFXOamcPuxYXhYKWmapb+S8vOy5hpLo
Re4RkOrY0cM3bN07GdEMlin5mU0y1t8y3ct4YewvmkgT42kTyMM+t1K4S0xsqjXx
xS716uGYh7eWtkxrCihj8AbXN/6pa095h+7TZyl2n83keiNUzM2KoqQVMwIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQU+dSlCYS89nZaff8nPOdiQJje
2PkwgZkGA1UdIwSBkTCBjoAU+dSlCYS89nZaff8nPOdiQJje2PmhYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUPgjnsnqT
```

```
7OaGnZgGF7/43ihPh+0wEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQDOq6yTQkIXzAZffpLDBss1R8VZzRoLvRzsFa1xpOcRnPqIm0untEHsOGX/
q34FiGQBOVY6btNaZb7aQXDlZGX+sGvst6BYC2u6H6Zpyo0qy0mz6AGA3eVEonJ2
lKpvAwBkrGajRC8JW0Gx31UZDmAcn3gjQi5zMWFead9mJJJA3g==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAOVp1h2I9wW7MA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yMjA3MTUx
MjQ0MTJaGA8yMjAxMTIxOTEyNDQxMlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEA13PkyWvl61iV/SYfO1UFO76UpDfPm2SF/Rz/o33cm699X++EYPxTnoEc
vmWeS0I7eDXc4OCUiToG0sEx0klE0CX1Z1tK6qJ+zgWQLZ9SZEC9H0NsSA6LhrHu
Nq0dzeK3LjhdfcX46/4GqdiptpdTuM4m/hOQ5yx4JMQ/n1sdpv4M5VLRWwWW9Lem
ufb79Id7O9SispxgRsz1KXIjp7N9S4BY7itSXz97uSyzTqEjWZ6mDUhTu3t21GKC
6flALGTTrG2yghEhz53rkvLsvwzjPSSlT6LIf0mrRPzHaf+EdaKoasElE1SHh+ZH
9mI81HywpE+HZ+W+5hBCvjYp90Y1fwIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQU58tN2JO+yEGq5JbIXxGi4vRVPyIwgY4GA1UdIwSBhjCBg4AU58tN
2JO+yEGq5JbIXxGi4vRVPyKhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAOVp1h2I9wW7MBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBANBN0e1EqNy4+IU2yQzMJ+Wy5ZIOtTP6GSBR
7muVY1bDeAwtNTEOpwgrZVlC7/xq5Q0LC1y0Z70hHXEf8au7qStaAoUtxzvhTAZI
NC0lwoFU56UFw4N0vZII17iqEfoqRC4PpI3OxqEJHFy0VLlvAzJoKB4QLLqDAYVA
LXCiOLoVT+y9tRYsxw5MyO0Bi6fxQIIADl2bE9xkunTN1Jkkwqo3LxNy/ryz4QWR
8K7jHUItifv4h/hxBKpHEquN8CkdvM9oeGl7I8PFrSFEpGr1euDXY0euZzzYiDBV
m6GpTJgzpVsEuIX52dPcPemwQncoIfZyhWDW85MJUnby2WTEcFo=
-----END CERTIFICATE-----
```

メキシコ (中部) — mx-central-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7zCCAq8CCQD4QwfTExrgxTAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yNDExMTQwOTQ4MjRaGA8y
MDUwMTExNTA5NDgyNFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0
```

```
b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIg
U2VydmljZXMgTExDMIIBtzCCASwGByqGSM44BAEwggEfAoGBAK+pJEOtD2RFlCbf
oAzONffZa42FB8G6OA/30DT81xWvB2Hnmgw7LfiELYqgfuck3YSDffNDcmiFTJuX
AU9u0NtbbxcOytcdUq4HInpHiB85I6WWWKYB66aGbvYowUPpqBZkASb1i/pYAh5W
nNYvx408008tTzqpMfZDchJHzWqBAhUA0ogoJzTw2/4pKhz9aqTsRCzRVPsCgYEA
qklRUcuU0du/bT/M6kWxYvrTGhO9KXQe+7RtbaIq4dWRsCBnO4smDY/GmI9H8ew1
LRJ9AcLGmxDm795CvVZkHNcht7PDAREagWmz2YvhLA+ev6U0RpfdlBXCck2p1CxQ
LMtoFO7DThksHIQBtnlQdHDvguKEZQz/Iobhne6Kb3wDgYQAAoGAQSTlOqFq6RtR
Jvp406XhGl+eO9SQUpevpzG3Dzbdy6EQ8PBJDOHHJvxbebvpnQssh0CnQfTkSw26
jwPy9V5zRC0ZezwnRTyGSJwiErUKDGeVekEAoNJL1USy8jKkgBajlskZrR+0JHDN
Jv3UwYIPplc4ZS2f2E7btrtlaWt/P70wCQYHKoZIzjgEAwMvADAsAhRXljPWKWYf
61DkDVdgPPHsS2/LuwIUejcGV9WuS3uPvJ6lmn4opxUGBZw=
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICPzCCAaigAwIBAgIUCmzpTTMBQYItpMC2VDYsZfIAS7IwDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgMEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcMB1NlYXR0bGUxIDAeBgNVBAoMF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI0MDQwMjEyNTAzNloXDTI5MDQwMTEyNTAzNlowXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgMEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcMB1NlYXR0bGUxIDAe
BgNVBAoMF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQDfvOnCzm1iN58Nm7k6ehoy6v0lnnFI617D6CY3bfuq01RCdEQL
96+pYawJieTH8JAQKj02CAa3AeaqdXTE/pDhI/YKLreeMb4K68WMn24Wjjs6oxjB
bAmsKXtt9ihKHGBFNUhgFrNFYyA2i7ieJviwpHjQ/XgXiG2u1/t/4VydUwIDAQAB
MA0GCSqGSIb3DQEBCwUAA4GBAL5+vvj4lhaE+J5tuCqV3XJzDd97lsD4le2O2uGw
P0sGdUcRAdxzU3Bwq/hhtzNWnfwo0aCEQKmLM7xyd3nUa0VvKXLq+DDuayipWINr
OATnNxFRe99d38qHTR1dgkjZdKbbtnl6O4fgM57tVEuQJd/N4ILl9jaRcJ5Ip+9t
3y5t
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJALxuE00HoJomMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yNDEwMzEx
MTE0MzNaGA8yMjA0MDQwNjExMTQzM1owXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEA13xWucALo8M/TXbZJgHrqFqFON91dSLPE/eLbmeIJbP1pb9ICd33qKAx
HlKSrXI9b9YS3U1Pl0bF3ZgfeE/x4Y0KDDZwzpf07H8IgRittULJoNLYVKCJXWPq
```

```
Ky1qvDJX3653dUbUu9eAdVCTRgk7eKpPBLAmW27+pgAGzEYrVV3u2AvqNTonvfTU
sPgEVnAL1J490pNM85KtFynxFTWGigHkd3BHidxmLrTH4I4eRxNZ9q/3gsDW+zKt
jQlpM7JzZa2OqxsF5YQDh1fF52Emqsr+ufLeGqDLOgT1QWcqpz57AX8GqZpgZULo
itCRNXbQDzZY9FxIgpiFJv3y/qYYDQIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQCt
MTeH4EgqjJEjb1qm5tzXYurLprVrEVQ+PhGXJFJd3xAZyeaDVYy7kio08E2xhmHd
HtkBDty2KnOHsTQmeAJCci7d4tYXZ/1qe341wwm9OoFcO8jhIndx6FXJCgQUY4dL
AAr9HQJFWG5dMZgbi1Zuhxdio3sSoOBjL2p7QIsGNkITvCDIs/H0/szpJnyyyIqu
wmUhSel5hdy5MwOsyUKVGnAdaS5Vd9oL4kLszS9nBZ7ny6BC9odIkFAdGqQ5vM4z
vcbfOq14hjatQmJgJhksN/0Dpl78Gheq0pIhP8LTkAOEG2832nQLzCa3oxSk8otG
GJXkzzyQjse+13r8+yNJ
-----END CERTIFICATE-----
```

中東 (バーレーン) – me-south-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7jCCAq4CCQCVWIgSmP8RhTAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xOTAyMDUxMzA2MjFaFw00
NTAyMDUxMzA2MjFaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbgwggEsBgcqhkjOOAQBMIIBHwKBgQDcwojQfgWdV1QliO0B
8n6cLZ38VE7ZmrjZ9OQV//Gst6S1h7euhC23YppKXi1zovefSDwFU54zi3/oJ++q
PHlP1WGL8IZ34BUgRTtG4TVolvp0smjkMvyRu5hIdKtzjV93Ccx15gVgyk+o1IEG
fZ2Kbw/Dd8JfoPS7KaSCmJKxXQIVAIZbIaDFRGa2qcMkW2HWASyNDl7bAoGBANtz
IdhfMq+l2I5iofY2oj3HI21Kj3LtZrWEg3W+/4rVhL3lTm0Nne1rl9yGujrjQwy5
Zp9V4A/w9w2O10Lx4K6hj34Eefy/aQnZwNdNhv/FQP7Az0fju+Yl6L13OOHQrL0z
Q+9cF7zEosekEnBQx3v6psNknKgD3Shgx+GO/LpCA4GFAAKBgQCVS7m77nuNAlZ8
wvUqcooxXMPkxJFl54NxAsAul9KP9KN4svm0O3Zrb7t2FOtXRM8zU3TqMpryq1o5
mpMPsZDg6RXo9BF7Hn0DoZ6PJTamkFA6md+NyTJWJKvXC7iJ8fGDBJqTciUHuCKr
12AztQ8bFWsrTgTzPE3p6U5ckcgV1TAJBgcqhkjOOAQDAy8AMCwCFB2NZGWm5EDl
86ayV3c1PEDukgQIAhQow38rQkN/VwHVeSW9DqEshXHjuQ==
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDPDCCAqWgAwIBAgIJAMl6uIV/zqJFMA0GCSqGSIb3DQEBCwUAMHIxCzAJBgNV
BAYTAlVTMRMwEQYDVQQIDApXYXNoaW5ndG9uMRAwDgYDVQQHDAdTZWF0dGxlMSAw
HgYDVQQKDBdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzEaMBgGA1UEAwwRZWMyLmFt
YXpvbmF3cy5jb20wIBcNMTkwNDI2MTQzMjQ3WhgPMjE5ODA5MjkxNDMyNDdaMHIx
```

```
CzAJBgNVBAYTAlVTMRMwEQYDVQQIDApXYXNoaW5ndG9uMRAwDgYDVQQHDAdTZWF0
dGxlMSAwHgYDVQQKDBdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzEaMBgGA1UEAwwR
ZWMyLmFtYXpvbmF3cy5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBALVN
CDTZEnIeoX1SEYqq6k1BV0ZlpY5y3KnoOreCAE589TwS4MX5+8Fzd6AmACmugeBP
Qk7Hm6b2+g/d4tWycyxLaQlcq81DB1GmXehRkZRgGeRge1ePWd1TUA0I8P/QBT7S
gUePm/kANSFU+P7s7u1NNl+vynyi0wUUrw7/wIZTAgMBAAGjgdcwgdQwHQYDVR0O
BBYEFILtMd+T4YgH1cgc+hVsVOV+480FMIGkBgNVHSMEgZwwgZmAFILtMd+T4YgH
1cgc+hVsVOV+480FoXakdDByMQswCQYDVQQGEwJVUzETMBEGA1UECAwKV2FzaGlu
Z3RvbjEQMA4GA1UEBwwHU2VhdHRsZTEgMB4GA1UECgwXQW1hem9uIFdlYiBTZXJ2
aWNlcyBMTEMxGjAYBgNVBAMMEWVjMi5hbWF6b25hd3MuY29tggkAyXq4hX/OokUw
DAYDVR0TBAUwAwEB/zANBgkqhkiG9w0BAQsFAAOBgQBhkNTBIFgWFd+ZhC/LhRUY
4OjEiykmbEp6hlzQ79T0Tfbn5A4NYDI2icBP0+hmf6qSnIhwJF6typyd1yPK5Fqt
NTpxxcXmUKquX+pHmIkK1LKDO8rNE84jqxrxRsfDi6by82fjVYf2pgjJW8R1FAw+
mL5WQRFexbfB5aXhcMo0AA==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJANZkFlQR2rKqMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xOTAyMDUx
MzA2MjBaGA8yMTk4MDcxMTEzMDYyMFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAy4Vnit2eBpEjKgOKBmyupJzJAiT4fr74tuGJNwwa+Is2vH12jMZn9Il1
UpvvEUYTIboIgISpf6SJ5LmV5rCv4jT4a1Wm0kjfNbiIlkUi8SxZrPypcw24m6ke
BVuxQZrZDs+xDUYIZifTmdgD50u5YE+TLg+YmXKnVgxBU6WZjbuK2INohi71aPBw
2zWUR7Gr/ggIpf635JLU3KIBLNEmrkXCVSnDFlsK4eeCrB7+UNak+4BwgpuykSGG
Op9+2vsuNqFeU1l9daQeG9roHR+4rIWSPa0opmMxv5nctgypOrE6zKXx2dNXQldd
VULv+WH7s6Vm4+yBeG8ctPYH5GOo+QIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQBs
ZcViiZdFdpcXESZP/KmZNDxB/kktlIEIhsQ+MNn29jayE5oLmtGjHj5dtA3XNKlr
f6PVygVTKbtQLQqunRT83e8+7iCZMKI5ev7pITUQVvTUwI+Fc01JkYZxRFlVBuFA
WGZO+98kxCS4n6tTwVt+nSuJr9BJRVC17apfHBgSS8c5OWna0VU/Cc9ka4eAfQR4
7pYSDU3wSRE01cs30q34lXZ629IyFirSJ5TTOIc0osNL7vwMQYj8HOn4OBYqxKy8
ZJyvfXsIPh0Na76PaBIs6ZlqAOflLrjGzxBPiwRM/XrGmF8ze4KzoUqJEnK13O6A
KHKgfiigQZ1+gv5FlyXH
-----END CERTIFICATE-----
```

中東 (UAE) - me-central-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7zCCAq
+gAwIBAgIGAXjXhqnnMAkGByqGSM44BAMwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgMEFdhc2hpbmd0b24gU3RhdGUxED
U4EddRIpUt9KnC7s5Of2EbdSPO9EAMMeP4C2USZpRV1AIlH7WT2NWPq/
xfW6MPbLm1Vs14E7gB00b/JmYLdrmVClpJ+f6AR7ECLCT7up1/63xhv4O1fnxqimFQ8E
+4P208UewwI1VBNaFpEy9nXzrith1yrv8iIDGZ3RSAHHAhUAl2BQjxUjC8yykrmCouuEC/
BYHPUCgYEA9+GghdabPd7LvKtcNrhXuXmUr7v6OuqC+VdMCz0HgmdRWVeOutRZT
+ZxBxCBgLRJFnEj6EwoFhO3zwkyjMim4TwWeotUfI0o4KOuHiuzpnWRbqN/C/ohNWLx
+2J6ASQ7zKTxvqhRkImog9/hWuWfBpKLZl6Ae1UlZAFMO/7PSSoDgYQAAoGAW+csuHsWp/7/
pv8CTKFwxsYudxuR6rbWaHCykIeAydXL9AWnphK6yp1ODEMBFl68Xq8Hp23sOWyf8moOhqCom9+0+ovuUFdpvCie86bp
TOZU568Ty1ff3dDWbdRzeNQRHodRG+XEQSizMkAreeWt4kBa+PUwCQYHKoZIzjgEAwMvADAsAhQD3Z
+XGmzKmgaLgGcVX/Qf1+Tn4QIUH1cgksBSVKbWj81tovBMJeKgdYo=
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICMzCCAZygAwIBAgIGAXjRrnDjMA0GCSqGSIb3DQEBBQUAMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIDBBXYXNoaW5n
KyA6zyruJQrYy00a6wqLA7eeUzk3bMiTkLsTeDQfrkaZMfBAjGaaOymRo1C3qzE4rIenmahvUplu9ZmLwL1idWXMRX2R
+d2SeoKOKQWoc2UOFZMHYxDue7zkyk1CIRaBukTeY13/
RIrlc6X61zJ5BBtZXlHwayjQIDAQABMA0GCSqGSIb3DQEBBQUAA4GBABTqTy3R6RXKPW45FA+cgo7YZEj/
Cnz5YaoUivRRdX2A83BHuBTvJE2+WX00FTEj4hRVjameE1nENoO8Z7fUVloAFDlDo69fhkJeSvn51D1WRrPnoWGgEfr1
B+Wqm3kVEz/QNcz6npmA6
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAM4h7b1CVhqqMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yMjA0MTEx
MDE1MDNaGA8yMjAxMDkxNTEwMTUwM1owXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEApybTWfMOhSoMpqPo72eqAmnn1dXGZM+G8EoZXzwHwT/+IHEXNB4q5N6k
tudYLre1bJxuzEw+iProSHjmb9bB9YscRTofjVhBlt35Fc+i8BaMeH94SR/eE8QO
mll8gnLNW3d62lyuhzuyv1e5wVlRqzYw+X2zRH4/wRDOCOpzjKoHIgyPKsMgwsw5
aTZhNMsGxZN9dbkf0iCGeQLDytwU/JTh/HqvSr3VfUOapTJJiyAxoCtZWgp1/7wC
RvOCSMRJobpUqxZgl/VsttwNkikSFz1wGkcYeSQvk+odbnYQckA8tdddoVI56eD4
```

```
qtREQvfpMAX5v7fcqLexl5d5vH8uZQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQUOadrbTs+OhzwoAgUJ7RqQNdwufkwgY4GA1UdIwSBhjCBg4AUOadr
bTs+OhzwoAgUJ7RqQNdwufmhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAM4h7b1CVhqqMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAICTdAOGEOnII8HaGCpCB8us/hGFaLptJaAf
D5SJAyVy66/mdfjGzE1BKkKxnbxemEVUIzbRidOnyilB+pKwN3edAjTZtWdpVA0V
R/G/qQPmcVljtycBz4VC6SuOUYflGzLH1GZ6GJWbuDtFzw8r7HGdRN1wrEPe3UF2
sMpuVezqnRUdvVRoVQP4jFgNsE7kNvtn2NiPhb/CtrxpcwIQ7r6YeoHcBSheuV1Z
xZDHynC3KUprQGx1+Z9QqPrDf180MaoqAlTl4+W6Pr2NJYrVUFGS/ivYshMg574l
CPU6r4wWZSKwEUXq4BInYX6z6iclp/p/J5QnJp2mAwyi6M+I13Y=
-----END CERTIFICATE-----
```

南米 (サンパウロ) – sa-east-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUX4Bh4MQ86Roh37VDRRX1MNOB3TcwDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
```

```
MB4XDTI0MDQyOTE2NDYwOVoXDTI5MDQyODE2NDYwOVowXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQCHvRjf/0kStpJ248khtIaN8qkDN3tkw4VjvA9nvPl2anJO+eIB
UqPfQG09kZlwpWpmyO8bGB2RWqWxCwuB/dcnIob6w420k9WY5C0IIGtDRNauN3ku
vGXkw3HEnF0EjYr0pcyWUvByWY4KswZV42X7Y7XSS13hOIcL6NLA+H94/QIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQUJdbMCBXKtvCcWdwUUizvtUF2
UTgwgZkGA1UdIwSBkTCBjoAUJdbMCBXKtvCcWdwUUizvtUF2UTihYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUX4Bh4MQ8
6Roh37VDRRX1MNOB3TcwEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQBnhocfH6ZIX6F5K9+Y9V4HFk8vSaaKL5ytw/P5td1h9ej94KF3xkZ5fyjN
URvGQv3kNmNJBoNarcP9I7JIMjsNPmVzqWawyCEGCZImoARxSS3Fc5EAs2PyBfcD
9nCtzMTaKO09Xyq0wqXVYn1xJsE5d5yBDsGrzaTHKjxo61+ezQ==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAMcyoxx4U0xxMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTA4MTQw
ODU4MDJaGA8yMTk1MDExNzA4NTgwMlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAw45IhGZVbQcy1fHBqzROhO8CsrDzxj/WP4cRbJo/2DAnimVrCCDs5O86
FA39Zo1xsDuJHDlwMKqeXYXkJXHYbcPWc6EYYAnR+PlLG+aNSOGUzsy202S03hT0
B20hWPCqpPp39itIRhG4id6nbNRJOzLm6evHuepMAHR4/OV7hyGOiGaV/v9zqiNA
pMCLhbh2xk0PO35HCVBuWt3HUjsgeks2eEsu9Ws6H3JXTCfiqp0TjyRWapM29OhA
cRJfJ/d/+wBTz1fkWOZ7TF+EWRIN5ITEadlDTPnF1r8kBRuDcS/lIGFwrOOHLo4C
cKoNgXkhTqDDBDu6oNBb2rS0K+sz3QIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQUqBy7D847Ya/w321Dfr+rBJGsGTwwgY4GA1UdIwSBhjCBg4AUqBy7
D847Ya/w321Dfr+rBJGsGTyhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAMcyoxx4U0xxMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBACOoWSBf7b9AlcNrl4lr3QWWSc7k90/tUZal
PlT0G3Obl2x9T/ZiBsQpbUvs0lfotG0XqGVVHcIxF38EbVwbw9KJGXbGSCJSEJkW
vGCtc/jYMHXfhx67Szmftm/MTYNvnzsyQQ3v8y3Rdah+xe1NPdpFrwmfL6xe3pFF
cY33KdHA/3PNLdn9CaEsHmcmj3ctaaXLFIzZhQyyjtsrgGfTLvXeXRokktvsLDS/
YgKedQ+jFjzVJqgr4NjfY/Wt7/8kbbdhzaqlB5pCPjLLzv0zp/XmO6k+JvOePOGh
JzGk5t1QrSju+MqNPFk3+1O7o910Vrhqw1QRB0gr1ExrviLbyfU=
-----END CERTIFICATE-----
```

AWS GovCloud (米国東部) - us-gov-east-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIULVyrqjjwZ461qelPCiShB1KCCj4wDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI0MDUwNzE1MjIzNloXDTI5MDUwNjE1MjIzNlowXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQCpohwYUVPH9I7Vbkb3WMe/JB0Y/bmfVj3VpcK445YBRO9K80al
esjgBc2tAX4KYg4Lht4EBKccLHTzaNi51YEGX1aLNrSmxhz1+WtzNLNUsyY3zD9z
vwX/3k1+JB2dRA+m+Cpwx4mjzZyAeQtHtegVaAytkmqtxQrSCexBxvqRqQIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQU1ZXneBYnPvYXkHVlVjg7918V
gE8wgZkGA1UdIwSBkTCBjoAU1ZXneBYnPvYXkHVlVjg7918VgE+hYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IULVyrqjjw
Z461qelPCiShB1KCCj4wEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQBfAL/YZv0y3zmVbXjyxQCsDloeDCJjFKIu3ameEckeIWJbST9LMto0zViZ
puIAf05x6GQiEqfBMk+YMxJfcTmJB4Ebaj4egFlslJPSHyC2xuydHlr3B04INOH5
Z2oCM68u6GGbj0jZjg7GJonkReG9N72kDva/ukwZKgq8zErQVQ==
```

```
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJALPB6hxFhay8MA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xODA0MTAx
MjMyNDlaGA8yMTk3MDkxMzEyMzI0OVowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAva9xsI9237KYb/SPWmeCVzi7giKNron8hoRDwlwwMC9+uHPd53UxzKLb
pTgtJWAPkZVxEdl2Gdhwr3SULoKcKmkqE6ltVFrVuPT33La1UufguT9k8ZDDuO9C
hQNHUdSVEuVrK3bLjaSsMOS7Uxmnn7lYT990IReowvnBNBsBlcabfQTBV04xfUG0
/m0XUiUFjOxDBqbNzkEIblW7vK7ydSJtFMSljga54UAVXibQt9EAIF7B8k9l2iLa
mu9yEjyQy+ZQICTuAvPUEWe6va2CHVY9gYQLA31/zU0VBKZPTNExjaqK4j8bKs1/
7dOV1so39sIGBz21cUBec1o+yCS5SwIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQBt
hO2W/Lm+Nk0qsXW6mqQFsAou0cASc/vtGNCyBfoFNX6aKXsVCHxq2aq2TUKWENs+
mKmYu1lZVhBOmLshyllh3RRoL3Ohp3jCwXytkWQ7ElcGjDzNGc0FArzB8xFyQNdK
MNvXDi/ErzgrHGSpcvmGHiOhMf3UzChMWbIr6udoDlMbSIO7+8F+jUJkh4Xl1lKb
YeN5fsLZp7T/6YvbFSPpmbn1YoE2vKtuGKxObRrhU3h4JHdp1Zel1pZ6lh5iM0ec
SD11SximGIYCjfZpRqI3q50mbxCd7ckULz+UUPwLrfOds4VrVVSj+x0ZdY19Plv2
9shw5ez6Cn7E3IfzqNHO
-----END CERTIFICATE-----
```

AWS GovCloud (米国西部) - us-gov-west-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
```

```
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDITCCAoqgAwIBAgIUe5wGF3jfb7lUHzvDxmM/ktGCLwwwDQYJKoZIhvcNAQEL
BQAwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAO
BgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExD
MB4XDTI0MDUwNzE3MzAzMloXDTI5MDUwNjE3MzAzMlowXDELMAkGA1UEBhMCVVMx
GTAXBgNVBAgTEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAe
BgNVBAoTF0FtYXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQCpohwYUVPH9I7Vbkb3WMe/JB0Y/bmfVj3VpcK445YBRO9K80al
esjgBc2tAX4KYg4Lht4EBKccLHTzaNi51YEGX1aLNrSmxhz1+WtzNLNUsyY3zD9z
vwX/3k1+JB2dRA+m+Cpwx4mjzZyAeQtHtegVaAytkmqtxQrSCexBxvqRqQIDAQAB
o4HfMIHcMAsGA1UdDwQEAwIHgDAdBgNVHQ4EFgQU1ZXneBYnPvYXkHVlVjg7918V
gE8wgZkGA1UdIwSBkTCBjoAU1ZXneBYnPvYXkHVlVjg7918VgE+hYKReMFwxCzAJ
BgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdT
ZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQ4IUe5wGF3jf
b7lUHzvDxmM/ktGCLwwwEgYDVR0TAQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsF
AAOBgQCbTdpx1Iob9SwUReY4exMnlwQlmkTLyA8tYGWzchCJOJJEPfsW0ryy1A0H
YIuvyUty3rJdp9ib8h3GZR71BkZnNddHhy06kPs4p8ewF8+d8OWtOJQcI+ZnFfG4
KyM4rUsBrljpG2aOCm12iACEyrvgJJrS8VZwUDZS6mZEnn/lhA==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJANCOF0Q6ohnuMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTA5MTAx
OTQyNDdaGA8yMTk1MDIxMzE5NDI0N1owXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAzIcGTzNqie3f1olrrqcfzGfbymSM2QfbTzDIOG6xXXeFrCDAmOq0wUhi
3fRCuoeHlKOWAPu76B9os71+zgF22dIDEVkpqHCjBrGzDQZXXUwOzhm+PmBUI8Z1
qvbVD4ZYhjCujWWzrsX6Z4yEK7PEFjtf4M4W8euw0RmiNwjy+knIFa+VxK6aQv94
lW98URFP2fD84xedHp6ozZlr3+RZSIFZsOiyxYsgiwTbesRMI0Y7LnkKGCIHQ/XJ
OwSISWaCddbu59BZeADnyhl4f+pWaSQpQQ1DpXvZAVBYvCH97J1oAxLfH8xcwgSQ
/se3wtn095VBt5b7qTVjOvy6vKZazwIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQA/
```

```
S8+a9csfASkdtQUOLsBynAbsBCH9Gykq2m8JS7YE4TGvqlpnWehz78rFTzQwmz4D
fwq8byPkl6DjdF9utqZ0JUo/Fxelxom0h6oievtBlSkmZJNbgc2WYm1zi6ptViup
Y+4S2+vWZyg/X1PXD7wyRWuETmykk73uEyeWFBYKCHWsO9sI+62O4Vf8Jkuj/cie
1NSJX8fkervfLrZSHBYhxLbL+actVEo00tiyZz8GnhgWx5faCY38D/k4Y/j5Vz99
7lUX/+fWHT3+lTL8ZZK7fOQWh6NQpI0wTP9KtWqfOUwMIbgFQPoxkP00TWRmdmPz
WOwTObEf9ouTnjG9OZ20
-----END CERTIFICATE-----
```

AWS European Sovereign Cloud – eusc-de-east-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC8DCCAq8CCQDyckpYH0omIDAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yNDEwMTgwOTM5MzdaGA8y
MDUwMTAxOTA5MzkzN1owXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgTEFdhc2hpbmd0
b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0FtYXpvbiBXZWIg
U2VydmljZXMgTExDMIIBtzCCASsGByqGSM44BAEwggEeAoGBAJR1zx7FzSGhXd3m
U1PfwH33QEK4CHnFSnp/DUse0SdLwsvLkHw7Tq3M7IX4E+TZ6OzYZZeoVZttqbyq
OZ8ROUS7CzDWcQ0503de/ya51cTs4BiO4nv2awm1WEoV9XtQyA2JLn0Jv7g2uja0
317krS+Dk3kjs9RpWvkmiyTMu1YhAhUA69U+A26BWLVpNIZZ0uWMZJSNlIkCgYBw
w1cgrHFGEqJNhqN7Sjs2YwTP+jpp1GvhxKyce5aOh5ErVvufb5gVVoKahrrHEBms
TrBXxnfcPaTupwdmEZ9eM8a+9RS2vye//KU1kGIOMgafPEzBEXs9m7okmAQpYXnG
cVOTtx2gd/E/VfKRiMSAmam178qwvcbcijNqRdLjRQOBhQACgYEAiEFVGYVCntmy
va8c2oU5/owdMws+NZ6u9blat0ymPr/8Q+DR77CgUPJW+K0k41GdgWpVlrsjiiPw
iygzw4DW7Vsl018KcIFUTiSwpzv2in+tKNUkO5Ly0q6arbSxrXZffYe7aOZKtCCH
XWDMKOH2iCvZptMpG7fDDSc7Be8v4AgwCQYHKoZIzjgEAwMwADAtAhQyvCF3q437
RvdPXMFcMc6qrD0svAIVAMvsN3NwOY4KRYlFv9Dnok8D/ixI
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICNjCCAZ+gAwIBAgIJAPybReXZqUSNMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yNDEwMTgw
OTM5MzdaGA8yMjA0MDMyNDA5MzkzN1owXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKB
gQDShbskmohorgcjN7gfApw89CEsg7praCDggaUVXF6gy9JdJ5INcv7+DkwKaAxd
NQWDsWZnxIOqPuPfyAYVP3D2/exP3Rfxj31UomLduu7S83zgiJ8VVL/GzlDELv5/
```

```
K2rTcd+580NjLSa6FYDcXDip2ir6TXxRYBYHP65dwniE+wIDAQABMA0GCSqGSIb3
DQEBCwUAA4GBAKu9t3CAtZt7c+k4bCZlmxrwgps1668p24x7QRmLxgXnwj5A2MOM
qEtxA9Q3q3WxUc0nWKaCTvRrTfWW917xsLcvVL8rI6xCjg2bb5FJ73xzcfSxHr9I
Q1mThJgLaWVIblllGJK7+EogR7gfqlPhc0Ql5bYuaozdt+r69xNuJlyR
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAODRxRi5VXBbMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yNTAyMjEx
MjQ0MTRaGA8yMjA0MDcyODEyNDQxNFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAr24OkDrpRU1ffx1PAcHZbKu7fZkhfT/e/PDhSkp2FvOzJEqDUT23E2wp
K/44mhJ3d8blYtgBiRXjqRsAFu8ZbzXLc0gaQ9A5woxfx5b7iOMwneCajS9xS5qr
cE0MMEY9XwG8MCp9YBoDJqDJh74oQqcPROBq16L4k2Ys22n1khus4JpwT7VSRE1I
nnt7fGIYsUmNGM+T2E6ubG3FvkY0HbLvNCcGRMCnDYqloo2BthuSG58KbMAku7VD
hMxzzJte5yMuRpHBojQr9N0hZcSJCtSFo6FhlxCzxGzwoSKzqm99ULYL4lf+IORe
DaKoATZJvbnX0LYAKLtPr1Pz9SD9pQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQUHpdxehvJwYL2o25SXyxsSndDW48wgY4GA1UdIwSBhjCBg4AUHpdx
ehvJwYL2o25SXyxsSndDW4+hYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAODRxRi5VXBbMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAFGcArNlGB3oRBaXohpnhrnVblhJJa6U7r1/
M5TAJCKCkziSz3lOZmZZ1YgSnQC1blhUD2rhuS8a0Mw1Ml48nLjIYMMK0tkJRKCD
LPxxEZp4EzogYj6PqfSubzm59MmBenIcJPaJ9y2uY1U4Q3PMfU6M1JqqQ0Lzplq0
n9eWpJK3+7UbtuyAzigO4DRWEBMxJBUU7f2LfY+qFjskoeHopX20cmn1vc1TGyAF
Qf8xdt1SaPKC26zJsrCkiMZJNm4aa8KqvhqBHVjV4NrGeVGCcdlmSDiL71Egk9kh
744eGQxNnhZNvePW2kVcSuh0HpAo4Q4vfpjN+jz2Ves3kjFsrgc=
-----END CERTIFICATE-----
```

EC2 インスタンスの STIG コンプライアンス

セキュリティ技術実装ガイド (STIG) は、防衛情報システム局 (DISA) が情報システムとソフトウェアを保護するために作成した設定強化標準です。システムを STIG 標準に準拠させるには、さまざまなセキュリティ設定をインストール、設定、およびテストする必要があります。

STIG は、次の脆弱性分類に基づいて適用されます。STIG の強化イメージを作成するとき、あるいは既存のインスタンスに STIG 設定を適用するときは、強化適用レベルを選択できます。上位のレベ

ルには、すべての下位レベルの STIG 設定が含まれます。たとえば、高 (カテゴリ I) レベルには、中カテゴリと低カテゴリの両方の設定が含まれます。

コンプライアンスレベル

- 高 (カテゴリ I)

  最も深刻なリスクです。機密性、可用性、または整合性の損失につながる可能性のある脆弱性が含まれます。

- ミディアム (カテゴリ II)

  機密性、可用性、整合性の損失につながる可能性があるものの、そのリスクが緩和される可能性がある脆弱性が含まれます。

- 低 (カテゴリー III)

  機密性、可用性、または整合性の喪失から保護するための対策を低下させる脆弱性が含まれます。

Amazon EC2 には、STIG 強化インスタンスを作成するために以下の方法が用意されています。

- 専用のパブリック AWS Windows AMI から、STIG 強化のために事前設定された Windows インスタンスを起動できます。詳細については、「AWS Windows AMI リファレンス」の「STIG Hardened AWS Windows Server AMIs」を参照してください。
- カスタマイズされた AMI を作成することで、EC2 Image Builder STIG 強化コンポーネントを使って構築された事前設定済みのインスタンスを起動できます。詳細については、「Image Builder ユーザーガイド」の「STIG hardening components」を参照してください。
- AWSEC2-ConfigureSTIG Systems Manager コマンドドキュメントを使用すると、STIG 設定を既存の EC2 インスタンスに 適用できます。Systems Manager の STIG コマンドドキュメントは、設定ミスをスキャンし、修復スクリプトを実行して DoD 証明書をインストールおよび更新し、不要な証明書を削除して STIG コンプライアンスを維持します。STIG Systems Manager ドキュメントの使用には、追加料金はかかりません。

トピック

- EC2 インスタンスの STIG 強化設定
- STIG 強化スクリプトのダウンロード
- AWS Systems Manager を使用してインスタンスに STIG 設定を適用する

EC2 インスタンスの STIG 強化設定

STIG の設定は、インスタンスのオペレーティングシステムとユーザーが選択したコンプライアンスのレベルに応じて異なります。以下では、各コンプライアンスレベルに適用される STIG のシステム固有の詳細について説明します。

内容

- EC2 Linux インスタンスの STIG 設定
- EC2 Windows インスタンスの STIG 強化設定

EC2 Linux インスタンスの STIG 設定

このセクションでは、Amazon EC2 がサポートしている Linux STIG 強化設定について説明します。強化スクリプトは、Linux ディストリビューションに基づいて、サポートされている STIG 設定をインフラストラクチャに適用します。Linux ディストリビューションに独自の STIG 強化設定がない場合、Amazon EC2 は次のように RHEL 設定を使用します。

- Red Hat Enterprise Linux (RHEL) 7 STIG 設定
  - RHEL 7
  - CentOS 7
  - Amazon Linux 2 (AL2)
- RHEL 8 STIG 設定
  - RHEL 8
  - CentOS 8
- RHEL 9 STIG 設定
  - RHEL 9
  - CentOS Stream 9

Linux STIG Low (Category III)

次のリストには、Amazon EC2 がインフラストラクチャに対してサポートしている STIG 設定が含まれています。サポートされている設定がお客様のインフラストラクチャに適用できない場合、Amazon EC2 はその設定をスキップして次に進みます。例えば、一部の STIG 強化設定は、スタンドアロンサーバーには適用されない場合があります。組織固有のポリシーも、適用される設定に影響を与える可能性があります (例: 管理者がドキュメントの設定を確認する必要がある)。

詳細なリストについては、「STIGs Document Library」を参照してください。完全なリストを表示する方法の詳細については、「STIG 表示ツール」を参照してください。

RHEL 7 STIG バージョン 3 リリース 15

- RHEL 7/CentOS 7/AL2

  V-204452、V-204576、および V-204605

RHEL 8 STIG バージョン 2 リリース 6

- RHEL 8/CentOS 8

  V-230241, V-230269, V-230270, V-230281, V-230285, V-230346, V-230381, V-230395, V-230468, V-230469, V-230485, V-230486, V-230491, V-230494, V-230495, V-230496, V-230497, V-230498, V-230499, and V-244527

RHEL 9 STIG バージョン 2 リリース 7

- RHEL 9/CentOS Stream 9

  V-257782, V-257795, V-257796, V-257824, V-257880, V-257946, V-257947, V-258037, V-258067, V-258069, V-258076, V-258138, V-258173

RHEL 10 STIG バージョン 1 リリース 1

V-280995、V-281203、および V-281292

Amazon Linux 2023 STIG バージョン 1 リリース 1

V-274141

SLES 12 STIG バージョン 3 リリース 4

V-217108, V-217113, V-217140, V-217198, V-217209, V-217211, V-217212, V-217213, V-217214, V-217215, V-217216, V-217236, V-217237, V-217238, V-217239, V-217282, V-255915

SLES 15 STIG バージョン 2 リリース 6

V-234811、V-234850、V-234868、V-234873、V-234905、V-234907、V-234908、V-234909、V-234933、および V-255921

Ubuntu 18.04 STIG バージョン 2 リリース 15

V-219163, V-219164, V-219165, V-219172, V-219173, V-219174, V-219175, V-219178, V-219179, V-219180, V-219210, V-219301, V-219327, V-219332, V-219333

Ubuntu 20.04 STIG バージョン 2 リリース 4

V-238202, V-238203, V-238221, V-238222, V-238223, V-238224, V-238226, V-238234, V-238235, V-238237, V-238308, V-238323, V-238357, V-238362, V-238373

Ubuntu 22.04 STIG バージョン 2 リリース 7

V-260472、V-260476、V-260479、V-260480、V-260481、V-260520、V-260521、V-260549、V-260550、

Ubuntu 24.04 STIG バージョン 1 リリース 4

V-270645、V-270646、V-270664、V-270677、V-270690、V-270695、V-270706、V-270710、V-270734、

Linux STIG Medium (Category II)

次のリストには、Amazon EC2 がインフラストラクチャに対してサポートしている STIG 設定が含まれています。サポートされている設定がお客様のインフラストラクチャに適用できない場合、Amazon EC2 はその設定をスキップして次に進みます。例えば、一部の STIG 強化設定は、スタンドアロンサーバーには適用されない場合があります。組織固有のポリシーも、適用される設定に影響を与える可能性があります (例: 管理者がドキュメントの設定を確認する必要がある)。

詳細なリストについては、「STIGs Document Library」を参照してください。完全なリストを表示する方法の詳細については、「STIG 表示ツール」を参照してください。

> **Note**
>
> Linux STIG Medium カテゴリには、Amazon EC2 が Category II の脆弱性をサポートする STIG 強化設定に加えて、Linux STIG Low (Category III) に適用されるリスト済みのすべての STIG 強化設定が含まれます。

RHEL 7 STIG バージョン 3 リリース 15

Amazon EC2 が Category III (Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

- RHEL 7/CentOS 7/AL2

V-204405, V-204406, V-204407, V-204408, V-204409, V-204410, V-204411, V-204412, V-204413, V-204414, V-204415, V-204416, V-204417, V-204418, V-204420, V-204422, V-204423, V-204426, V-204427, V-204431, V-204434, V-204435, V-204437, V-204449, V-204450, V-204451, V-204457, V-204466, V-204490, V-204491, V-204503, V-204507, V-204508, V-204510, V-204511, V-204512, V-204514, V-204515, V-204516, V-204517, V-204521, V-204524, V-204531, V-204536, V-204537, V-204538, V-204539, V-204540, V-204541, V-204542, V-204543, V-204544, V-204545, V-204546, V-204547, V-204548, V-204549, V-204550, V-204551, V-204552, V-204553, V-204554, V-204555, V-204556, V-204557, V-204558, V-204559, V-204560, V-204562, V-204563, V-204564, V-204565, V-204566, V-204567, V-204568, V-204572, V-204579, V-204584, V-204585, V-204587, V-204588, V-204589, V-204590, V-204591, V-204592, V-204593, V-204596, V-204597, V-204598, V-204599, V-204600, V-204601, V-204602, V-204609, V-204610, V-204611, V-204612, V-204613, V-204614, V-204615, V-204616, V-204617, V-204619, V-204622, V-204625, V-204630, V-204631, V-204633, V-233307, V-237634, V-237635, V-251703, V-255925, V-255927, V-255928, V-256970

RHEL 8 STIG バージョン 2 リリース 6

Amazon EC2 が Category III (Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

- RHEL 8/CentOS 8

V-230222, V-230228, V-230231, V-230233, V-230236, V-230237, V-230238, V-230239, V-230240, V-230240, V-230243, V-230244, V-230245, V-230246, V-230247, V-230248, V-230249, V-230250, V-230255, V-230256, V-230257, V-230258, V-230259, V-230260, V-230261, V-230262, V-230266, V-230267, V-230268, V-230271, V-230273, V-230275, V-230276, V-230277, V-230278, V-230279, V-230280, V-230282, V-230282, V-230286, V-230287, V-230288, V-230290, V-230291, V-230296, V-230298, V-230310, V-230311, V-230312, V-230313, V-230314, V-230315, V-230316, V-230318, V-230319, V-230320, V-230321, V-230322, V-230324, V-230325, V-230326, V-230327, V-230330, V-230332, V-230333, V-230335, V-230337, V-230339, V-230341, V-230343, V-230345, V-230347, V-230348, V-230352, V-230353, V-230354, V-230356, V-230357, V-230358, V-230359, V-230360, V-230361, V-230362, V-230363, V-230365, V-230366, V-230368, V-230369, V-230370, V-230372, V-230373, V-230375, V-230376, V-230377, V-230378, V-230380, V-230382, V-230383, V-230385, V-230386, V-230387, V-230389, V-230390, V-230392, V-230393, V-230394, V-230396, V-230397, V-230398, V-230399, V-230400, V-230401, V-230402, V-230403, V-230404, V-230405, V-230406, V-230407, V-230408, V-230409, V-230410, V-230411, V-230412, V-230413, V-230418, V-230419, V-230421, V-230422, V-230423, V-230424, V-230425, V-230426, V-230427, V-230428, V-230429, V-230430, V-230431, V-230432, V-230433, V-230434, V-230435, V-230436, V-230437, V-230438,

V-230439, V-230444, V-230446, V-230447, V-230448, V-230449, V-230455, V-230456, V-230462, V-230463, V-230464, V-230465, V-230466, V-230467, V-230470, V-230471, V-230472, V-230473, V-230474, V-230475, V-230478, V-230480, V-230481, V-230482, V-230483, V-230488, V-230489, V-230493, V-230502, V-230503, V-230505, V-230506, V-230507, V-230523, V-230524, V-230525, V-230526, V-230527, V-230532, V-230535, V-230536, V-230537, V-230538, V-230539, V-230540, V-230541, V-230542, V-230543, V-230544, V-230545, V-230546, V-230547, V-230548, V-230549, V-230550, V-230555, V-230556, V-230557, V-230559, V-230560, V-230561, V-237640, V-237642, V-237643, V-244519, V-244523, V-244524, V-244525, V-244526, V-244528, V-244531, V-244533, V-244535, V-244536, V-244538, V-244539, V-244542, V-244543, V-244544, V-244545, V-244547, V-244550, V-244551, V-244552, V-244553, V-244554, V-250315, V-250315, V-250315, V-250316, V-250316, V-250317, V-251707, V-251708, V-251709, V-251710, V-251711, V-251713, V-251714, V-251715, V-251716, V-251717, V-251718, V-256974, V-257258, V-257258, V-274877, V-279929, V-279930, and V-279931

RHEL 9 STIG バージョン 2 リリース 7

Amazon EC2 が Category III (Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

- RHEL 9/CentOS Stream 9

V-257780, V-257781, V-257783, V-257786, V-257788, V-257790, V-257791, V-257792, V-257793, V-257794, V-257797, V-257798, V-257799, V-257800, V-257801, V-257802, V-257803, V-257804, V-257805, V-257806, V-257807, V-257808, V-257809, V-257810, V-257811, V-257812, V-257813, V-257814, V-257815, V-257816, V-257817, V-257818, V-257825, V-257827, V-257828, V-257829, V-257830, V-257831, V-257832, V-257833, V-257834, V-257836, V-257838, V-257839, V-257840, V-257841, V-257842, V-257849, V-257882, V-257883, V-257884, V-257885, V-257886, V-257887, V-257888, V-257889, V-257890, V-257891, V-257892, V-257893, V-257894, V-257895, V-257896, V-257897, V-257898, V-257899, V-257900, V-257901, V-257902, V-257903, V-257904, V-257905, V-257906, V-257907, V-257908, V-257909, V-257910, V-257911, V-257912, V-257913, V-257914, V-257915, V-257916, V-257917, V-257918, V-257919, V-257920, V-257921, V-257922, V-257923, V-257924, V-257925, V-257926, V-257927, V-257928, V-257929, V-257930, V-257933, V-257934, V-257935, V-257936, V-257939, V-257940, V-257942, V-257943, V-257944, V-257948, V-257951, V-257952, V-257953, V-257954, V-257957, V-257958, V-257959, V-257960, V-257961, V-257962, V-257963, V-257964, V-257965, V-257966, V-257967, V-257968, V-257969, V-257970, V-257971, V-257972, V-257973, V-257974, V-257975, V-257976, V-257977, V-257978, V-257979, V-257980, V-257982, V-257983, V-257985, V-257987, V-257988, V-257992, V-257993, V-257994, V-257995,

V-257996, V-257997, V-257998, V-257999, V-258000, V-258001, V-258002, V-258003, V-258004, V-258005, V-258006, V-258007, V-258008, V-258009, V-258010, V-258011, V-258028, V-258034, V-258035, V-258038, V-258039, V-258040, V-258041, V-258043, V-258046, V-258049, V-258052, V-258054, V-258055, V-258056, V-258057, V-258060, V-258063, V-258064, V-258065, V-258066, V-258068, V-258070, V-258071, V-258072, V-258073, V-258074, V-258075, V-258077, V-258079, V-258080, V-258081, V-258082, V-258083, V-258084, V-258085, V-258088, V-258089, V-258090, V-258091, V-258092, V-258093, V-258095, V-258097, V-258098, V-258099, V-258100, V-258101, V-258102, V-258103, V-258104, V-258105, V-258107, V-258108, V-258109, V-258110, V-258111, V-258112, V-258113, V-258114, V-258115, V-258116, V-258117, V-258118, V-258119, V-258120, V-258121, V-258122, V-258123, V-258124, V-258125, V-258126, V-258128, V-258129, V-258130, V-258133, V-258137, V-258140, V-258141, V-258142, V-258144, V-258145, V-258146, V-258147, V-258148, V-258150, V-258151, V-258152, V-258153, V-258154, V-258156, V-258157, V-258158, V-258159, V-258160, V-258161, V-258162, V-258163, V-258164, V-258165, V-258166, V-258167, V-258168, V-258169, V-258170, V-258171, V-258172, V-258175, V-258176, V-258177, V-258178, V-258179, V-258180, V-258181, V-258182, V-258183, V-258184, V-258185, V-258186, V-258187, V-258188, V-258189, V-258190, V-258191, V-258192, V-258193, V-258194, V-258195, V-258196, V-258197, V-258198, V-258199, V-258200, V-258201, V-258202, V-258203, V-258204, V-258205, V-258206, V-258207, V-258208, V-258209, V-258210, V-258211, V-258212, V-258213, V-258214, V-258215, V-258216, V-258217, V-258218, V-258219, V-258220, V-258221, V-258222, V-258223, V-258224, V-258225, V-258226, V-258227, V-258228, V-258229, V-258232, V-258233, V-258234, V-258237, V-258239, V-258240, V-270174, V-270175, V-270176, V-270177, V-272488, V-279936

RHEL 10 STIG バージョン 1 リリース 1

Amazon EC2 が Category III (Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

V-280094, V-280942, V-280943, V-280945, V-280946, V-280947, V-280948, V-280950, V-280952, V-280953, V-280954, V-280955, V-280956, V-280958, V-280959, V-280960, V-280961, V-280962, V-280963, V-280964, V-280965, V-280966, V-280967, V-280968, V-280969, V-280970, V-280972, V-280973, V-280974, V-280975, V-280976, V-280977, V-280981, V-280983, V-280984, V-280986, V-280987, V-280988, V-280989, V-280990, V-280991, V-280992, V-280993, V-280994, V-280996, V-280997, V-280999, V-281000, V-281001, V-281002, V-281003, V-281005, V-281006, V-281017, V-281018, V-281019, V-281020, V-281021, V-281022, V-281023, V-281024, V-281025, V-281026, V-281027, V-281028, V-281029, V-281030, V-281031, V-281032, V-281033, V-281034, V-281035, V-281036, V-281037, V-281038, V-281039, V-281040, V-281041, V-281042, V-281043, V-281044, V-281045, V-281047, V-281048, V-281049, V-281050, V-281051, V-281052, V-281053, V-281054,

V-281055, V-281056, V-281057, V-281058, V-281059, V-281060, V-281061, V-281062, V-281063, V-281064, V-281065, V-281066, V-281067, V-281068, V-281069, V-281070, V-281071, V-281072, V-281073, V-281074, V-281075, V-281076, V-281077, V-281078, V-281079, V-281080, V-281081, V-281082, V-281083, V-281084, V-281085, V-281086, V-281087, V-281094, V-281095, V-281096, V-281097, V-281098, V-281099, V-281100, V-281101, V-281102, V-281103, V-281104, V-281105, V-281107, V-281108, V-281109, V-281110, V-281111, V-281113, V-281114, V-281115, V-281116, V-281117, V-281118, V-281119, V-281120, V-281121, V-281122, V-281123, V-281124, V-281125, V-281126, V-281127, V-281128, V-281129, V-281130, V-281131, V-281132, V-281133, V-281134, V-281135, V-281136, V-281137, V-281138, V-281139, V-281140, V-281141, V-281142, V-281143, V-281144, V-281145, V-281146, V-281147, V-281148, V-281149, V-281150, V-281151, V-281152, V-281153, V-281154, V-281155, V-281156, V-281157, V-281158, V-281159, V-281160, V-281161, V-281162, V-281163, V-281164, V-281165, V-281168, V-281169, V-281171, V-281174, V-281175, V-281177, V-281178, V-281179, V-281180, V-281181, V-281182, V-281183, V-281184, V-281185, V-281187, V-281188, V-281189, V-281190, V-281191, V-281192, V-281193, V-281195, V-281196, V-281197, V-281198, V-281200, V-281201, V-281204, V-281205, V-281206, V-281209, V-281210, V-281212, V-281214, V-281215, V-281217, V-281218, V-281219, V-281220, V-281224, V-281225, V-281226, V-281249, V-281251, V-281252, V-281253, V-281254, V-281255, V-281256, V-281257, V-281258, V-281259, V-281260, V-281261, V-281262, V-281263, V-281264, V-281265, V-281266, V-281269, V-281270, V-281271, V-281272, V-281273, V-281274, V-281276, V-281277, V-281278, V-281279, V-281280, V-281281, V-281282, V-281284, V-281285, V-281286, V-281287, V-281288, V-281289, V-281290, V-281291, V-281293, V-281295, V-281296, V-281297, V-281300, V-281301, V-281302, V-281303, V-281304, V-281305, V-281306, V-281308, V-281309, V-281310, V-281311, V-281312, V-281313, V-281314, V-281315, V-281316, V-281317, V-281318, V-281319, V-281320, V-281322, V-281323, V-281324, V-281325, V-281327, V-281328, V-281331, V-281333, V-281334, V-281335, V-281337, V-281338, V-281340, V-281341, V-281342, V-281343, V-281344, V-281345, V-281346, V-281347, V-281348, V-281349, V-281350, V-281351, V-281352, V-281353, V-281354, V-281355, V-281356, V-281357, V-281358, V-281359, V-281360, V-281361, V-281363, V-281364, V-281365

Amazon Linux 2023 STIG バージョン 1 リリース 2

Amazon EC2 が Category III (Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

V-273995, V-274000, V-274001, V-274002, V-274003, V-274004, V-274005, V-274006, V-274008, V-274009, V-274010, V-274011, V-274012, V-274013, V-274014, V-274017, V-274018, V-274019, V-274020, V-274021, V-274022, V-274023, V-274024, V-274026, V-274027, V-274028, V-274030, V-274031, V-274032, V-274033, V-274034, V-274035, V-274036, V-274037, V-274040, V-274041,

V-274042, V-274044, V-274045, V-274047, V-274048, V-274049, V-274050, V-274051, V-274053, V-274054, V-274059, V-274061, V-274062, V-274069, V-274070, V-274071, V-274072, V-274073, V-274074, V-274075, V-274076, V-274077, V-274078, V-274079, V-274081, V-274082, V-274083, V-274084, V-274085, V-274086, V-274087, V-274088, V-274089, V-274090, V-274091, V-274092, V-274093, V-274094, V-274095, V-274096, V-274097, V-274098, V-274099, V-274100, V-274101, V-274102, V-274103, V-274104, V-274105, V-274106, V-274107, V-274108, V-274109, V-274110, V-274111, V-274112, V-274113, V-274114, V-274115, V-274116, V-274117, V-274119, V-274120, V-274121, V-274122, V-274123, V-274124, V-274125, V-274126, V-274127, V-274128, V-274129, V-274130, V-274131, V-274132, V-274133, V-274134, V-274135, V-274136, V-274137, V-274138, V-274139, V-274140, V-274142, V-274143, V-274144, V-274145, V-274147, V-274149, V-274151, V-274152, V-274154, V-274155, V-274156, V-274157, V-274160, V-274161, V-274162, V-274163, V-274164, V-274165, V-274166, V-274167, V-274168, V-274169, V-274170, V-274173, V-274177, V-274181, V-274182, V-274185, V-274187

SLES 12 STIG バージョン 3 リリース 4

Amazon EC2 が Category III (Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

V-217102, V-217105, V-217105, V-217106, V-217106, V-217110, V-217116, V-217117, V-217118, V-217119, V-217120, V-217121, V-217122, V-217124, V-217125, V-217126, V-217127, V-217128, V-217130, V-217134, V-217138, V-217143, V-217147, V-217152, V-217153, V-217154, V-217155, V-217156, V-217158, V-217161, V-217163, V-217166, V-217167, V-217168, V-217169, V-217170, V-217171, V-217182, V-217183, V-217188, V-217190, V-217191, V-217194, V-217195, V-217196, V-217197, V-217200, V-217201, V-217202, V-217203, V-217204, V-217205, V-217206, V-217207, V-217208, V-217210, V-217217, V-217218, V-217223, V-217227, V-217230, V-217240, V-217241, V-217242, V-217243, V-217244, V-217245, V-217246, V-217247, V-217248, V-217249, V-217250, V-217251, V-217252, V-217253, V-217254, V-217255, V-217257, V-217258, V-217260, V-217265, V-217266, V-217267, V-217269, V-217272, V-217273, V-217274, V-217275, V-217276, V-217277, V-217278, V-217279, V-217280, V-217283, V-217284, V-217286, V-217287, V-217288, V-217289, V-217290, V-217291, V-217292, V-217293, V-217294, V-217295, V-217296, V-217299, V-217300, V-217301, V-217302, V-233308, V-237605, V-237606, V-237607, V-237608, V-237609, V-237610, V-237611, V-237612, V-237613, V-237614, V-237615, V-237616, V-237617, V-237618, V-237619, V-237620, V-237621, V-237622, V-237623, V-251720, V-251722, V-255914, V-256981

SLES 15 STIG バージョン 2 リリース 6

Amazon EC2 が Category III (Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

V-234802, V-234807, V-234808, V-234809, V-234813, V-234815, V-234817, V-234821, V-234822, V-234823, V-234825, V-234827, V-234828, V-234829, V-234830, V-234832, V-234833, V-234834, V-234835, V-234836, V-234837, V-234838, V-234839, V-234840, V-234841, V-234842, V-234843, V-234844, V-234845, V-234848, V-234854, V-234855, V-234856, V-234857, V-234858, V-234861, V-234862, V-234863, V-234869, V-234870, V-234875, V-234878, V-234880, V-234881, V-234882, V-234883, V-234884, V-234885, V-234886, V-234887, V-234888, V-234889, V-234891, V-234895, V-234896, V-234897, V-234899, V-234900, V-234901, V-234902, V-234903, V-234904, V-234906, V-234910, V-234911, V-234912, V-234913, V-234914, V-234918, V-234924, V-234928, V-234932, V-234937, V-234938, V-234939, V-234940, V-234941, V-234942, V-234943, V-234944, V-234945, V-234946, V-234947, V-234948, V-234949, V-234950, V-234951, V-234952, V-234954, V-234956, V-234957, V-234958, V-234959, V-234961, V-234962, V-234964, V-234966, V-234969, V-234973, V-234975, V-234976, V-234977, V-234978, V-234979, V-234981, V-234982, V-234983, V-234991, V-235002, V-235003, V-235007, V-235008, V-235009, V-235010, V-235013, V-235014, V-235015, V-235016, V-235017, V-235018, V-235019, V-235020, V-235021, V-235022, V-235023, V-235024, V-235025, V-235026, V-235028, V-235029, V-235030, V-251724, V-255920, V-256983, V-274879

Ubuntu 18.04 STIG バージョン 2 リリース 15

Amazon EC2 が Category III (Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

V-219149, V-219155, V-219156, V-219160, V-219166, V-219168, V-219176, V-219181, V-219184, V-219186, V-219188, V-219189, V-219190, V-219191, V-219192, V-219193, V-219194, V-219195, V-219196, V-219197, V-219198, V-219199, V-219200, V-219201, V-219202, V-219203, V-219204, V-219205, V-219206, V-219207, V-219208, V-219209, V-219213, V-219214, V-219215, V-219216, V-219217, V-219218, V-219219, V-219220, V-219221, V-219222, V-219223, V-219224, V-219225, V-219226, V-219227, V-219228, V-219229, V-219230, V-219231, V-219232, V-219233, V-219234, V-219235, V-219236, V-219238, V-219239, V-219240, V-219241, V-219242, V-219243, V-219244, V-219250, V-219254, V-219257, V-219263, V-219264, V-219265, V-219266, V-219267, V-219268, V-219269, V-219270, V-219271, V-219272, V-219273, V-219274, V-219275, V-219276, V-219277, V-219279, V-219281, V-219287, V-219291, V-219296, V-219297, V-219298, V-219299, V-219300, V-219303, V-219304, V-219306, V-219309, V-219310, V-219311, V-219315, V-219318, V-219319, V-219323, V-219326, V-219328, V-219330, V-219331, V-219335, V-219336, V-219337, V-219338, V-219339, V-219342, V-219344, V-233779, V-233780, V-255906

Ubuntu 20.04 STIG バージョン 2 リリース 4

Amazon EC2 が Category III (Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

V-238200, V-238205, V-238207, V-238209, V-238210, V-238211, V-238212, V-238213, V-238220, V-238225, V-238227, V-238228, V-238229, V-238230, V-238231, V-238232, V-238236, V-238238, V-238239, V-238240, V-238241, V-238242, V-238244, V-238245, V-238246, V-238247, V-238248, V-238249, V-238250, V-238251, V-238252, V-238253, V-238254, V-238255, V-238256, V-238257, V-238258, V-238264, V-238268, V-238271, V-238277, V-238278, V-238279, V-238280, V-238281, V-238282, V-238283, V-238284, V-238285, V-238286, V-238287, V-238288, V-238289, V-238290, V-238291, V-238292, V-238293, V-238294, V-238295, V-238297, V-238298, V-238299, V-238300, V-238301, V-238302, V-238303, V-238304, V-238309, V-238310, V-238315, V-238316, V-238317, V-238318, V-238319, V-238320, V-238324, V-238325, V-238329, V-238330, V-238333, V-238334, V-238337, V-238338, V-238339, V-238340, V-238341, V-238342, V-238343, V-238344, V-238345, V-238346, V-238347, V-238348, V-238349, V-238350, V-238351, V-238352, V-238353, V-238355, V-238356, V-238359, V-238360, V-238369, V-238370, V-238371, V-238376, V-238377, V-238378, V-251505, V-255912, V-274852, V-274853

Ubuntu 22.04 STIG バージョン 2 リリース 7

Amazon EC2 が Category III (Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

V-260471, V-260473, V-260474, V-260475, V-260475, V-260477, V-260478, V-260485, V-260486, V-260487, V-260488, V-260489, V-260490, V-260491, V-260492, V-260493, V-260494, V-260495, V-260496, V-260497, V-260498, V-260499, V-260500, V-260505, V-260506, V-260507, V-260508, V-260509, V-260510, V-260511, V-260512, V-260513, V-260514, V-260522, V-260527, V-260528, V-260530, V-260533, V-260534, V-260535, V-260537, V-260538, V-260540, V-260542, V-260543, V-260545, V-260546, V-260547, V-260553, V-260554, V-260555, V-260556, V-260557, V-260560, V-260561, V-260562, V-260563, V-260564, V-260565, V-260566, V-260567, V-260569, V-260572, V-260573, V-260574, V-260575, V-260576, V-260582, V-260584, V-260585, V-260586, V-260588, V-260589, V-260590, V-260591, V-260594, V-260597, V-260598, V-260599, V-260600, V-260601, V-260602, V-260603, V-260604, V-260605, V-260606, V-260607, V-260608, V-260609, V-260610, V-260611, V-260612, V-260613, V-260614, V-260615, V-260616, V-260617, V-260618, V-260619, V-260620, V-260621, V-260622, V-260623, V-260624, V-260625, V-260626, V-260627, V-260628, V-260629, V-260630, V-260631, V-260632, V-260633, V-260634, V-260635, V-260636, V-260637, V-260638, V-260639, V-260640, V-260641, V-260642, V-260643, V-260644, V-260645, V-260646, V-260647, V-260648, V-260649, V-274862, V-274864, V-274866

Ubuntu 24.04 STIG バージョン 1 リリース 4

Amazon EC2 が Category III (Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

V-270649, V-270651, V-270652, V-270653, V-270654, V-270656, V-270657, V-270659, V-270660, V-270661, V-270662, V-270663, V-270669, V-270672, V-270673, V-270674, V-270676, V-270678, V-270679, V-270680, V-270681, V-270683, V-270684, V-270685, V-270686, V-270687, V-270688, V-270689, V-270692, V-270693, V-270696, V-270697, V-270698, V-270699, V-270700, V-270701, V-270702, V-270703, V-270704, V-270705, V-270709, V-270715, V-270716, V-270718, V-270720, V-270721, V-270722, V-270723, V-270724, V-270725, V-270726, V-270727, V-270728, V-270729, V-270730, V-270731, V-270732, V-270733, V-270737, V-270739, V-270740, V-270741, V-270742, V-270743, V-270746, V-270750, V-270753, V-270755, V-270756, V-270757, V-270758, V-270759, V-270760, V-270765, V-270766, V-270767, V-270768, V-270769, V-270770, V-270771, V-270772, V-270773, V-270775, V-270776, V-270777, V-270778, V-270779, V-270780, V-270781, V-270782, V-270783, V-270784, V-270785, V-270786, V-270787, V-270788, V-270789, V-270790, V-270791, V-270792, V-270793, V-270794, V-270795, V-270796, V-270797, V-270798, V-270799, V-270800, V-270801, V-270802, V-270803, V-270804, V-270805, V-270806, V-270807, V-270808, V-270809, V-270810, V-270811, V-270812, V-270813, V-270814, V-270815, V-270821, V-270822, V-270823, V-270824, V-270825, V-270826, V-270827, V-270828, V-270829, V-270830, V-270831, V-270832, V-274870, V-274871, V-274872, V-274873

Linux STIG High (Category I)

次のリストには、Amazon EC2 がインフラストラクチャに対してサポートしている STIG 設定が含まれています。サポートされている設定がお客様のインフラストラクチャに適用できない場合、Amazon EC2 はその設定をスキップして次に進みます。例えば、一部の STIG 強化設定は、スタンドアロンサーバーには適用されない場合があります。組織固有のポリシーも、適用される設定に影響を与える可能性があります (例: 管理者がドキュメントの設定を確認する必要がある)。

詳細なリストについては、「STIGs Document Library」を参照してください。完全なリストを表示する方法の詳細については、「STIG 表示ツール」を参照してください。

> **Note**
>
> Linux STIG High カテゴリには、Amazon EC2 が Category I の脆弱性をサポートする STIG 強化設定に加えて、Linux STIG Medium および Low カテゴリに適用されるリスト済みのすべての STIG 強化設定が含まれます。

RHEL 7 STIG バージョン 3 リリース 15

Amazon EC2 が Category II および III (Medium および Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

- RHEL 7/CentOS 7/AL2

  V-204424、V-204425、V-204442、V-204443、V-204447、V-204448、V-204455、V-204462、V-204497 および V-204621

RHEL 8 STIG バージョン 2 リリース 6

Amazon EC2 が Category II および III (Medium および Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

- RHEL 8/CentOS 8

  V-230223、V-230264、V-230283、V-230284、V-230487、V-230492、V-230533、V-230558、V-244540 および V-230531

RHEL 9 STIG バージョン 2 リリース 7

Amazon EC2 が Category II および III (Medium および Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

- RHEL 9/CentOS Stream 9

  V-257820、V-257821、V-257826、V-257835、V-257955、V-257956、V-258059、V-258230、V-258238 および V-257785

RHEL 10 バージョン 1 リリース 1

Amazon EC2 が Category II および III (Medium および Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

V-280932, V-280933, V-280934, V-280944, V-280949, V-280951, V-280978, V-281007, V-281014, V-281015, V-281216, V-281222, V-281223, V-281267, V-281268, V-281275, V-281298, V-281299, V-281307

Amazon Linux 2023 STIG バージョン 1 リリース 2

Amazon EC2 が Category II および III (Medium および Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

V-273996, V-273997, V-273999, V-274007, V-274038, V-274039, V-274046, V-274052, V-274057, V-274153

SLES 12 STIG バージョン 3 リリース 4

Amazon EC2 が Category II および III (Medium および Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

V-217101、V-217139、V-217141、V-217142、V-217159、V-217160、V-217164、V-217264、V-217268、

SLES 15 STIG バージョン 2 リリース 6

Amazon EC2 が Category II および III (Medium および Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

V-234800, V-234804, V-234818, V-234852, V-234859, V-234860, V-234898, V-234984, V-234985, V-234988, V-234989, V-234990, V-235031, V-235032, V-251725

Ubuntu 18.04 STIG バージョン 2 リリース 15

Amazon EC2 が Category II および III (Medium および Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

V-219157、V-219158、V-219177、V-219212、V-219308、V-219314、V-219316、および V-251507

Ubuntu 20.04 STIG バージョン 2 リリース 4

Amazon EC2 が Category II および III (Medium および Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

V-238201, V-238218, V-238219, V-238326, V-238327, V-238380, V-251504

Ubuntu 22.04 STIG バージョン 2 リリース 7

Amazon EC2 が Category II および III (Medium および Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

V-260469、V-260482、V-260483、V-260523、V-260524、V-260526、V-260529、V-260539、V-260570、

Ubuntu 24.04 STIG バージョン 1 リリース 4

Amazon EC2 が Category II および III (Medium および Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

V-270647、V-270648、V-270665、V-270666、V-270708、V-270711、V-270712、V-270713、V-270714、

EC2 Windows インスタンスの STIG 強化設定

Amazon EC2 は、次の Windows オペレーティングシステムバージョンの STIG ベースラインをサポートしています。

- Windows Server 2025
- Windows Server 2022
- Windows Server 2019
- Windows Server 2016
- Windows Server 2012 R2

このセクションでは、Amazon EC2 が Windows インフラストラクチャでサポートしている現在の STIG 設定をリストし、その後にバージョン履歴ログを示します。

低、中、高の STIG 設定を適用できます。

Windows STIG Low (Category III)

次のリストには、Amazon EC2 がインフラストラクチャに対してサポートしている STIG 設定が含まれています。サポートされている設定がお客様のインフラストラクチャに適用できない場合、Amazon EC2 はその設定をスキップして次に進みます。例えば、一部の STIG 強化設定は、スタンドアロンサーバーには適用されない場合があります。組織固有のポリシーも、適用される設定に影響を与える可能性があります (例: 管理者がドキュメントの設定を確認する必要がある)。

Windows 向け STIG の完全なリストについては、「STIG ドキュメントライブラリ」を参照してください。完全なリストを表示する方法の詳細については、「STIG 表示ツール」を参照してください。

- Windows Server 2025 STIG バージョン 1 リリース 1

  V-278082、V-278083、V-278084、V-278085、V-278098、V-278104、V-278110、および V-278231

- Windows Server 2022 STIG バージョン 2 リリース 7

  V-254335, V-254336, V-254337, V-254338, V-254351, V-254357, V-254363, および V-254481

- Windows Server 2019 STIG バージョン 3 リリース 7

V-205691、V-205819、V-205858、V-205859、V-205860、V-205870、V-205871、および V-205923

- Windows Server 2016 STIG バージョン 2 リリース 10

  V-224916、V-224917、V-224918、V-224919、V-224931、V-224942、および V-225060

- Windows Server 2012 R2 MS STIG バージョン 3 リリース 5

  V-225250, V-225318, V-225319, V-225324, V-225327, V-225328, V-225330, V-225331, V-225332, V-225333, V-225334, V-225335, V-225336, V-225342, V-225343, V-225355, V-225357, V-225358, V-225359, V-225360, V-225362, V-225363, V-225376, V-225392, V-225394, V-225412, V-225459, V-225460, V-225462, V-225468, V-225473, V-225476, V-225479, V-225480, V-225481, V-225482, V-225483, V-225484, V-225485, V-225487, V-225488, V-225489, V-225490, V-225511, V-225514, V-225525, V-225526, V-225536, V-225537

- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 7

  Microsoft .NET フレームワークには、カテゴリ III の脆弱性に対応する STIG 設定は適用されません。

- Windows Firewall STIG バージョン 2 リリース 2

  V-241994、V-241995、V-241996、V-241999、V-242000、V-242001、V-242006、V-242007、V-242008

- Internet Explorer 11 STIG バージョン 2 リリース 6

  V-223056 および V-223078

- Microsoft Edge STIG バージョン 2 リリース 4 (Windows Server 2022 および 2025 のみ)

  V-235727、V-235731、V-235751、V-235752 および V-235765

Windows STIG Medium (Category II)

次のリストには、Amazon EC2 がインフラストラクチャに対してサポートしている STIG 設定が含まれています。サポートされている設定がお客様のインフラストラクチャに適用できない場合、Amazon EC2 はその設定をスキップして次に進みます。例えば、一部の STIG 強化設定は、スタンドアロンサーバーには適用されない場合があります。組織固有のポリシーも、適用される設定に影響を与える可能性があります (例: 管理者がドキュメントの設定を確認する必要がある)。

Windows 向け STIG の完全なリストについては、「STIG ドキュメントライブラリ」を参照してください。完全なリストを表示する方法の詳細については、「STIG 表示ツール」を参照してください。

> **Note**
>
> Windows STIG Medium カテゴリには、Amazon EC2 が Category II の脆弱性をサポートする STIG 強化設定に加えて、Windows STIG Low (Category III) に適用されるリスト済みのすべての STIG 強化設定が含まれます。

- Windows Server 2025 STIG バージョン 1 リリース 1

  Amazon EC2 が Category III (Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

  V-278015, V-278016, V-278019, V-278020, V-278021, V-278022, V-278023, V-278024, V-278025, V-278026, V-278033, V-278034, V-278035, V-278036, V-278037, V-278038, V-278039, V-278047, V-278048, V-278049, V-278050, V-278051, V-278052, V-278053, V-278054, V-278055, V-278056, V-278057, V-278058, V-278059, V-278060, V-278061, V-278062, V-278063, V-278064, V-278065, V-278066, V-278067, V-278068, V-278069, V-278070, V-278071, V-278072, V-278073, V-278074, V-278075, V-278076, V-278077, V-278078, V-278079, V-278080, V-278086, V-278088, V-278089, V-278091, V-278092, V-278093, V-278094, V-278095, V-278096, V-278097, V-278102, V-278103, V-278105, V-278106, V-278107, V-278108, V-278109, V-278111, V-278112, V-278113, V-278114, V-278115, V-278116, V-278117, V-278118, V-278119, V-278120, V-278122, V-278123, V-278124, V-278126, V-278127, V-278129, V-278130, V-278131, V-278165, V-278168, V-278169, V-278170, V-278171, V-278174, V-278180, V-278181, V-278182, V-278183, V-278184, V-278185, V-278187, V-278188, V-278189, V-278192, V-278193, V-278194, V-278195, V-278198, V-278199, V-278200, V-278201, V-278202, V-278203, V-278204, V-278205, V-278206, V-278209, V-278210, V-278211, V-278212, V-278213, V-278214, V-278218, V-278220, V-278221, V-278222, V-278223, V-278226, V-278227, V-278228, V-278229, V-278230, V-278232, V-278233, V-278234, V-278235, V-278236, V-278237, V-278238, V-278239, V-278240, V-278241, V-278243, V-278244, V-278245, V-278247, V-278248, V-278249, V-278251, V-278252, V-278253, V-278254, V-278255, V-278256, V-278257, V-278258, V-278259, V-278260, V-278261, V-278262, V-279916, V-279917, V-279918, V-279919, V-279920, V-279921, V-279922, V-279923

- Windows Server 2022 STIG バージョン 2 リリース 7

  Amazon EC2 が Category III (Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

  V-254247, V-254269, V-254270, V-254271, V-254272, V-254273, V-254274, V-254275, V-254276, V-254277, V-254278, V-254285, V-254286, V-254287, V-254288, V-254289, V-254290, V-254291,

V-254292, V-254296, V-254297, V-254298, V-254299, V-254300, V-254301, V-254302, V-254303, V-254304, V-254305, V-254307, V-254309, V-254311, V-254312, V-254313, V-254314, V-254315, V-254316, V-254319, V-254320, V-254321, V-254322, V-254323, V-254324, V-254325, V-254326, V-254327, V-254328, V-254329, V-254330, V-254331, V-254332, V-254333, V-254334, V-254339, V-254341, V-254342, V-254344, V-254345, V-254346, V-254347, V-254348, V-254349, V-254350, V-254355, V-254356,V-254358, V-254359, V-254360, V-254361, V-254362, V-254364, V-254365, V-254366, V-254367, V-254368, V-254369, V-254370, V-254371, V-254372, V-254373, V-254375, V-254376, V-254377, V-254379, V-254380, V-254382, V-254383, V-254384, V-254431, V-254433, V-254434, V-254435, V-254436, V-254438, V-254439, V-254440, V-254442, V-254443, V-254444, V-254445, V-254447, V-254448, V-254449, V-254450, V-254451, V-254452, V-254453, V-254454, V-254455, V-254456, V-254459, V-254460, V-254461, V-254462, V-254463, V-254464, V-254468, V-254470, V-254471, V-254472, V-254473, V-254476, V-254477, V-254478, V-254479, V-254480, V-254482, V-254483, V-254484, V-254485, V-254486, V-254487, V-254488, V-254489, V-254491, V-254493, V-254494, V-254495, V-254497, V-254498, V-254499, V-254501, V-254502, V-254503, V-254504, V-254505, V-254506, V-254507, V-254508, V-254509, V-254510, V-254511, V-254512, V-278942, V-278943, V-278944, V-278945, V-278946, V-278947, V-278948, V-278949

- Windows Server 2019 STIG バージョン 3 リリース 7

  Amazon EC2 が Category III (Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

  V-205625, V-205626, V-205627, V-205629, V-205630, V-205633, V-205634, V-205635, V-205636, V-205637, V-205638, V-205639, V-205640, V-205641, V-205642, V-205643, V-205644, V-205648, V-205649, V-205650, V-205651, V-205652, V-205655, V-205656, V-205659, V-205660, V-205662, V-205671, V-205672, V-205673, V-205675, V-205676, V-205678, V-205679, V-205680, V-205681, V-205682, V-205683, V-205684, V-205685, V-205686, V-205687, V-205688, V-205689, V-205690, V-205692, V-205693, V-205694, V-205697, V-205698, V-205708, V-205709, V-205712, V-205714, V-205716, V-205717, V-205718, V-205719, V-205720, V-205722, V-205730, V-205731, V-205733, V-205747, V-205748,V-205749, V-205751, V-205752, V-205754, V-205755, V-205756, V-205758, V-205759, V-205760, V-205761, V-205762, V-205763, V-205764, V-205765, V-205766, V-205767, V-205768, V-205769, V-205770, V-205771, V-205772, V-205773, V-205774, V-205775, V-205776, V-205777, V-205778, V-205779, V-205780, V-205781, V-205782, V-205783, V-205784, V-205795, V-205796, V-205797, V-205798, V-205801, V-205808, V-205809, V-205810, V-205811, V-205812, V-205813, V-205814, V-205815, V-205816, V-205817, V-205821, V-205822, V-205823, V-205824, V-205825, V-205826, V-205827, V-205828, V-205830, V-205832, V-205833, V-205835, V-205836, V-205837, V-205838, V-205842, V-205861, V-205863, V-205865, V-205866, V-205867, V-205868, V-205869, V-205872, V-205873, V-205874, V-205909, V-205910, V-205911, V-205912, V-205915,

V-205916, V-205917, V-205918, V-205920, V-205921, V-205922, V-205925, V-257503, V-278934, V-278935, V-278936, V-278937, V-278938, V-278939, V-278940, V-278941

- Windows Server 2016 STIG バージョン 2 リリース 10

  Amazon EC2 が Category III (Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

  V-224850, V-224851, V-224852, V-224853, V-224854, V-224855, V-224856, V-224857, V-224858, V-224859, V-224866, V-224867, V-224868, V-224869, V-224870, V-224871, V-224872, V-224873, V-224877, V-224878, V-224879, V-224880, V-224881, V-224882, V-224883, V-224884, V-224885, V-224886, V-224888, V-224890, V-224892, V-224893, V-224894, V-224895, V-224896, V-224897, V-224900, V-224901, V-224902, V-224903, V-224904, V-224905, V-224906, V-224907, V-224908, V-224909, V-224910, V-224911, V-224912, V-224913, V-224914, V-224915, V-224920, V-224921, V-224922, V-224924, V-224925, V-224926, V-224927, V-224928, V-224929, V-224930, V-224935, V-224936, V-224937, V-224938, V-224939, V-224940, V-224941, V-224943, V-224944, V-224945, V-224946, V-224947, V-224948, V-224949, V-224951, V-224952, V-224953, V-224955, V-224956, V-224957, V-224959, V-224960, V-224962, V-224963, V-224965, V-224966, V-224967, V-224968, V-224969, V-224987, V-224988, V-224989, V-224995, V-224996, V-224997, V-224998, V-224999, V-225000, V-225001, V-225002, V-225003, V-225004, V-225005, V-225008, V-225009, V-225010, V-225011, V-225013, V-225014, V-225015, V-225016, V-225017, V-225018, V-225019, V-225020, V-225021, V-225022, V-225023, V-225024, V-225026, V-225027, V-225028, V-225029, V-225030, V-225031, V-225032, V-225033, V-225034, V-225035, V-225038, V-225039, V-225040, V-225041, V-225042, V-225043, V-225047, V-225049, V-225050, V-225051, V-225052, V-225055, V-225056, V-225057, V-225058, V-225059, V-225061, V-225062, V-225063, V-225064, V-225065, V-225066, V-225067, V-225068, V-225070, V-225072, V-225073, V-225074, V-225076, V-225077, V-225078, V-225080, V-225081, V-225082, V-225083, V-225084, V-225085, V-225086, V-225087, V-225088, V-225089, V-225092, V-225093, V-257502

- Windows Server 2012 R2 MS STIG バージョン 3 リリース 5

  Amazon EC2 が Category III (Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

  V-225239, V-225259, V-225260, V-225261, V-225263, V-225264, V-225265, V-225266, V-225267, V-225268, V-225269, V-225270, V-225271, V-225272, V-225273, V-225275, V-225276, V-225277, V-225278, V-225279, V-225280, V-225281, V-225282, V-225283, V-225284, V-225285, V-225286, V-225287, V-225288, V-225289, V-225290, V-225291, V-225292, V-225293, V-225294, V-225295, V-225296, V-225297, V-225298, V-225299, V-225300, V-225301, V-225302, V-225303, V-225304,

V-225305, V-225314, V-225315, V-225316, V-225317, V-225325, V-225326, V-225329, V-225337, V-225338, V-225339, V-225340, V-225341, V-225344, V-225345, V-225346, V-225347, V-225348, V-225349, V-225350, V-225351, V-225352, V-225353, V-225356, V-225367, V-225368, V-225369, V-225370, V-225371, V-225372, V-225373, V-225374, V-225375, V-225377, V-225378, V-225379, V-225380, V-225381, V-225382, V-225383, V-225384, V-225385, V-225386, V-225389, V-225391, V-225393, V-225395, V-225397, V-225398, V-225400, V-225401, V-225402, V-225404, V-225405, V-225406, V-225407, V-225408, V-225409, V-225410, V-225411, V-225413, V-225414, V-225415, V-225441, V-225442, V-225443, V-225448, V-225452, V-225453, V-225454, V-225455, V-225456, V-225457, V-225458, V-225461, V-225463, V-225464, V-225469, V-225470, V-225471, V-225472, V-225474, V-225475, V-225477, V-225478, V-225486, V-225494, V-225500, V-225501, V-225502, V-225503, V-225504, V-225506, V-225508, V-225509, V-225510, V-225513, V-225515, V-225516, V-225517, V-225518, V-225519, V-225520, V-225521, V-225522, V-225523, V-225524, V-225527, V-225528, V-225529, V-225530, V-225531, V-225532, V-225533, V-225534, V-225535, V-225538, V-225539, V-225540, V-225541, V-225542, V-225543, V-225544, V-225545, V-225546, V-225548, V-225549, V-225550, V-225551, V-225553, V-225554, V-225555, V-225557, V-225558, V-225559, V-225560, V-225561, V-225562, V-225563, V-225564, V-225565, V-225566, V-225567, V-225568, V-225569, V-225570, V-225571, V-225572, V-225573, V-225574

- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 7

  Amazon EC2 が Category III (Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

  V-225223, V-225230, V-225235, V-225238

- Windows Firewall STIG バージョン 2 リリース 2

  Amazon EC2 が Category III (Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

  V-241989、V-241990、V-241991、V-241993、V-241998、V-242003、V-242004、および V-242005

- Internet Explorer 11 STIG バージョン 2 リリース 6

  Amazon EC2 が Category III (Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

  V-223015, V-223017, V-223018, V-223019, V-223020, V-223021, V-223022, V-223023, V-223024, V-223025, V-223026, V-223027, V-223028, V-223029, V-223030, V-223031, V-223032, V-223033, V-223034, V-223035, V-223036, V-223037, V-223038, V-223039, V-223040, V-223041, V-223042,

V-223043, V-223044, V-223045, V-223046, V-223048, V-223049, V-223050, V-223051, V-223052, V-223053, V-223054, V-223055, V-223057, V-223058, V-223059, V-223060, V-223061, V-223062, V-223063, V-223064, V-223065, V-223066, V-223067, V-223068, V-223069, V-223070, V-223071, V-223072, V-223073, V-223074, V-223075, V-223076, V-223077, V-223079, V-223080, V-223081, V-223082, V-223083, V-223084, V-223085, V-223086, V-223087, V-223088, V-223089, V-223090, V-223091, V-223092, V-223093, V-223094, V-223095, V-223096, V-223097, V-223098, V-223099, V-223100, V-223101, V-223102, V-223103, V-223104, V-223105, V-223106, V-223107, V-223108, V-223109, V-223110, V-223111, V-223112, V-223113, V-223114, V-223115, V-223116, V-223117, V-223118, V-223119, V-223120, V-223121, V-223123, V-223125, V-223126, V-223127, V-223128, V-223129, V-223130, V-223131, V-223132, V-223133, V-223134, V-223135, V-223136, V-223137, V-223138, V-223139, V-223140, V-223141, V-223142, V-223143, V-223144, V-223145, V-223146, V-223147, V-223148, V-223149, V-250540, V-250541

- Microsoft Edge STIG バージョン 2 リリース 4 (Windows Server 2022 および 2025 のみ)

  V-235720、V-235721、V-235723、V-235724、V-235725、V-235726、V-235728、V-235729、V-235730 および V-246736

- Microsoft Defender STIG バージョン 2 リリース 7

  Amazon EC2 が Category III (Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

  V-213427, V-213429, V-213430, V-213431, V-213432, V-213433, V-213434, V-213435, V-213436, V-213437, V-213438, V-213439, V-213440, V-213441, V-213442, V-213443, V-213444, V-213445, V-213446, V-213447, V-213448, V-213449, V-213450, V-213451, V-213454, V-213455, V-213456, V-213457, V-213458, V-213459, V-213460, V-213461, V-213462, V-213463, V-213464, V-213465, V-213466, V-278647, V-278648, V-278649, V-278650, V-278651, V-278652, V-278653, V-278654, V-278655, V-278656, V-278658, V-278659, V-278660, V-278661, V-278662, V-278668, V-278669, V-278672, V-278674, V-278675, V-278676, V-278677, V-278678, V-278679, V-278680, V-278863

Windows STIG High (Category I)

次のリストには、Amazon EC2 がインフラストラクチャに対してサポートしている STIG 設定が含まれています。サポートされている設定がお客様のインフラストラクチャに適用できない場合、Amazon EC2 はその設定をスキップして次に進みます。例えば、一部の STIG 強化設定は、スタンドアロンサーバーには適用されない場合があります。組織固有のポリシーも、適用される設定に影響を与える可能性があります (例: 管理者がドキュメントの設定を確認する必要がある)。

Windows 向け STIG の完全なリストについては、「STIG ドキュメントライブラリ」を参照してください。完全なリストを表示する方法の詳細については、「STIG 表示ツール」を参照してください。

> **Note**
>
> Windows STIG High カテゴリには、Amazon EC2 が Category I の脆弱性をサポートする STIG 強化設定に加えて、Windows STIG Medium および Low カテゴリに適用されるリスト済みのすべての STIG 強化設定が含まれます。

- Windows Server 2025 STIG バージョン 1 リリース 1

  Amazon EC2 が Category II および III (Medium および Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

  V-278040、V-278099、V-278100、V-278101、V-278121、V-278125、V-278128、V-278196、V-278215 および V-278250
- Windows Server 2022 STIG バージョン 2 リリース 7

  Amazon EC2 が Category II および III (Medium および Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

  V-254250, V-254293, V-254352, V-254353, V-254354, V-254374, V-254378, V-254381, V-254446, V-254466, V-254467, V-254469, V-254474, V-254475, V-254492, V-254496, V-254500
- Windows Server 2019 STIG バージョン 3 リリース 7

  Amazon EC2 が Category II および III (Medium および Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

  V-205653、V-205654、V-205663、V-205711、V-205713、V-205724、V-205725、V-205750、V-205753
- Windows Server 2016 STIG バージョン 2 リリース 10

  Amazon EC2 が Category II および III (Medium および Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

  V-224831, V-224874, V-224932, V-224933, V-224934, V-224954, V-224958, V-224961, V-225025, V-225045, V-225046, V-225048, V-225053, V-225054, V-225071, V-225079, V-225091
- Windows Server 2012 R2 MS STIG バージョン 3 リリース 5

Amazon EC2 が Category II および III (Medium および Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

V-225274, V-225354, V-225364, V-225365, V-225366, V-225390, V-225396, V-225399, V-225444, V-225449, V-225491, V-225492, V-225493, V-225496, V-225497, V-225498, V-225505, V-225507, V-225547, V-225552, V-225556

- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 7

  Microsoft .NET Framework のカテゴリ II および III (Medium および Low) の脆弱性に対して Amazon EC2 がサポートするすべての STIG 強化設定が含まれます。カテゴリ I の脆弱性に対して、追加的な STIG 設定は適用されません。

- Windows Firewall STIG バージョン 2 リリース 2

  Amazon EC2 が Category II および III (Medium および Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

  V-241992、V-241997、および V-242002

- Internet Explorer 11 STIG バージョン 2 リリース 6

  Amazon EC2 が Category II および III (Medium および Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

  V-252910

- Microsoft Edge STIG バージョン 2 リリース 4 (Windows Server 2022 および 2025 のみ)

  Amazon EC2 が Category II および III (Medium および Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

  V-235758 および V-235759

- Microsoft Defender STIG バージョン 2 リリース 7

  Amazon EC2 が Category II および III (Medium および Low) の脆弱性をサポートするすべての STIG 強化設定に加えて、以下が含まれます。

  V-213426, V-213428, V-213452, V-213453

STIG 強化スクリプトのダウンロード

Amazon は、STIG 強化コンポーネントをリリースごとにオペレーティングシステム関連のバンドルにまとめています。バンドルは、ダウンロードして実行するターゲットオペレーティングシステムに適したアーカイブファイルです。Linux スクリプトのバンドルは、TAR ファイル (.tgz ファイル拡張子) として保存されています。Windows スクリプトのバンドルは、ZIP ファイル (.zip ファイル拡張子) として保存されています。

Amazon は、スクリプトのバンドルを各リージョン AWS リージョン の EC2 Windows S3 STIG バケットに保存しています。Linux バンドルには個別のバケットはありません。SSL/TLS を使用して AWS リソースと通信します。TLS 1.2 は必須ですが、TLS 1.3 を推奨します。

トピック

- STIG ダウンロードバンドルの詳細
- Linux STIG のバージョン履歴
- Windows STIG バージョン履歴

STIG ダウンロードバンドルの詳細

> **Important**
>
> Systems Manager ドキュメントがダウンロードする STIG 強化スクリプトは、いくつかの例外を除き、サードパーティのパッケージをインストールしません。サードパーティのパッケージがインスタンスにすでにインストールされていて、`InstallPackage` パラメータが `yes` に設定されている場合、Amazon EC2 がそのパッケージに対してサポートしている、関連する STIG が適用されます。

AWS CLI から次のコマンドを実行すると、Amazon S3 は、バケットから最新の STIG 強化スクリプトのバンドルファイルをダウンロードします。

```
aws s3 cp s3://aws-windows-downloads-region/STIG/operating system/Latest/bundle-name destination-directory
```

例: 一時ディレクトリにダウンロードする

こちらは、`/tmp` ディレクトリにダウンロードされた Linux バンドルの例です。

```
aws s3 cp s3://aws-windows-downloads-us-east-1/STIG/Linux/
Latest/LinuxAWSConfigureSTIG.tgz /tmp
```

ダウンロードファイルのストレージパスとバンドルファイル名のパターンと例は次のとおりです。

ファイルストレージパスをダウンロードする

`s3://aws-windows-downloads-<region>/STIG/<operating system>/Latest/<bundle file name>`

ダウンロードパスの変数

region

AWS リージョン (各リージョンには独自のダウンロードバケットがあります)。

operating system

STIG が適用されるインスタンスのオペレーティングシステムプラットフォーム – `Linux` または `Windows`

bundle file name

形式は *<os #####>.<#######>* です。

os bundle name

オペレーティングシステムバンドルの標準名プレフィックスは `LinuxAWSConfigureSTIG` または `AWSConfigureSTIG` です。下位互換性を維持するため、Windows 用のダウンロードにはプラットフォームプレフィックスは含まれていません。

file extension

圧縮ファイルフォーマット `tgz` (Linux) または `zip` (Windows)。

バンドルファイル名の例

- `LinuxAWSConfigureSTIG.tgz`
- `AWSConfigureSTIG.zip`

Linux STIG のバージョン履歴

このセクションでは、四半期ごとに更新される Linux スクリプトバンドルのバージョン履歴を口グに記録します。四半期ごとの変更点と公開されたバージョンを確認するには、タイトルを選択して情報を展開します。四半期中に変更がない場合、そのことがタイトルに反映されます。

2026 年 第 1 四半期の変更 - 04/07/2026:

RHEL 10 オペレーティングシステムのサポートを追加し、次の STIG バージョンを更新し、2026 年 第 1 四半期リリースの STIGS をすべてのコンプライアンスレベル (低/中/高) に対して適用しました。

STIG-Build-Linux version 1.0.x

- RHEL 8 STIG バージョン 2 リリース 6
- RHEL 9 STIG バージョン 2 リリース 7
- RHEL 10 STIG バージョン 1 リリース 1
- Amazon Linux 2023 STIG バージョン 1 リリース 2
- SLES 12 STIG バージョン 3 リリース 4
- SLES 15 STIG バージョン 2 リリース 6
- Ubuntu 20.04 STIG バージョン 2 リリース 4
- Ubuntu 22.04 STIG バージョン 2 リリース 7
- Ubuntu 24.04 STIG バージョン 1 リリース 4

2025 年第 3 四半期の変更 - 2025 年 9 月 4 日:

SUSE Linux Enterprise Server (SLES) オペレーティングシステムと Amazon Linux 2023 のサポートが追加されました。次の STIG バージョンを更新し、2025 年第 3 四半期リリースの STIG をすべてのコンプライアンスレベル (低/中/高) に対して適用しました。

STIG-Build-Linux version 1.0.x

- RHEL 7 STIG バージョン 3 リリース 15
- RHEL 8 STIG バージョン 2 リリース 4
- RHEL 9 STIG バージョン 2 リリース 5
- Amazon Linux 2023 STIG バージョン 1 リリース 1
- SLES 12 STIG バージョン 3 リリース 3
- SLES 15 STIG バージョン 2 リリース 5
- Ubuntu 18.04 STIG バージョン 2 リリース 15
- Ubuntu 20.04 STIG バージョン 2 リリース 3

- Ubuntu 22.04 STIG バージョン 2 リリース 5
- Ubuntu 24.04 STIG バージョン 1 リリース 2

2025 年第 2 四半期の変更 - 2025 年 6 月 26 日:

次の STIG のバージョンを更新し、2025 年第 2 四半期のリリースに対して STIGS を適用しました。

Linux STIG Low (Category III)

- RHEL 7 STIG バージョン 3 リリース 15
- RHEL 8 STIG バージョン 2 リリース 3
- RHEL 9 STIG バージョン 2 リリース 4
- Ubuntu 18.04 STIG バージョン 2 リリース 15
- Ubuntu 20.04 STIG バージョン 2 リリース 2
- Ubuntu 22.04 STIG バージョン 2 リリース 4
- Ubuntu 24.04 STIG バージョン 1 リリース 1

Linux STIG Medium (Category II)

- RHEL 7 STIG バージョン 3 リリース 15
- RHEL 8 STIG バージョン 2 リリース 3
- RHEL 9 STIG バージョン 2 リリース 4
- Ubuntu 18.04 STIG バージョン 2 リリース 15
- Ubuntu 20.04 STIG バージョン 2 リリース 2
- Ubuntu 22.04 STIG バージョン 2 リリース 4
- Ubuntu 24.04 STIG バージョン 1 リリース 1

Linux STIG High (Category I)

- RHEL 7 STIG バージョン 3 リリース 15
- RHEL 8 STIG バージョン 2 リリース 3
- RHEL 9 STIG バージョン 2 リリース 4
- Ubuntu 18.04 STIG バージョン 2 リリース 15

- Ubuntu 20.04 STIG バージョン 2 リリース 2
- Ubuntu 22.04 STIG バージョン 2 リリース 4
- Ubuntu 24.04 STIG バージョン 1 リリース 1

2025 年第 1 四半期の変更 - 2025 年 4 月 11 日:

次の STIG バージョンを更新し、2025 年第 1 四半期リリースに対して STIGS を適用し、Ubuntu 24.04 のサポートを追加しました。

Linux STIG Low (Category III)

- RHEL 7 STIG バージョン 3 リリース 15
- RHEL 8 STIG バージョン 2 リリース 2
- RHEL 9 STIG バージョン 2 リリース 3
- Ubuntu 18.04 STIG バージョン 2 リリース 15
- Ubuntu 20.04 STIG バージョン 2 リリース 2
- Ubuntu 22.04 STIG バージョン 2 リリース 3
- Ubuntu 24.04 STIG バージョン 1 リリース 1

Linux STIG Medium (Category II)

- RHEL 7 STIG バージョン 3 リリース 15
- RHEL 8 STIG バージョン 2 リリース 2
- RHEL 9 STIG バージョン 2 リリース 3
- Ubuntu 18.04 STIG バージョン 2 リリース 15
- Ubuntu 20.04 STIG バージョン 2 リリース 2
- Ubuntu 22.04 STIG バージョン 2 リリース 3
- Ubuntu 24.04 STIG バージョン 1 リリース 1

Linux STIG High (Category I)

- RHEL 7 STIG バージョン 3 リリース 15
- RHEL 8 STIG バージョン 2 リリース 2
- RHEL 9 STIG バージョン 2 リリース 3

- Ubuntu 18.04 STIG バージョン 2 リリース 15
- Ubuntu 20.04 STIG バージョン 2 リリース 2
- Ubuntu 22.04 STIG バージョン 2 リリース 3
- Ubuntu 24.04 STIG バージョン 1 リリース 1

2024 年第 4 四半期の変更 - 2024 年 12 月 10 日:

次の STIG バージョンを更新し、2024 年第 4 四半期リリースに対して STIGS を適用し、Linux コンポーネントの 2 つの新しい入力パラメータに関する情報を追加しました。

Linux STIG Low (Category III)

- RHEL 7 STIG バージョン 3 リリース 15
- RHEL 8 STIG バージョン 2 リリース 1
- RHEL 9 STIG バージョン 2 リリース 2
- Ubuntu 18.04 STIG バージョン 2 リリース 15
- Ubuntu 20.04 STIG バージョン 2 リリース 1
- Ubuntu 22.04 STIG バージョン 2 リリース 2

Linux STIG Medium (Category II)

- RHEL 7 STIG バージョン 3 リリース 15
- RHEL 8 STIG バージョン 2 リリース 1
- RHEL 9 STIG バージョン 2 リリース 2
- Ubuntu 18.04 STIG バージョン 2 リリース 15
- Ubuntu 20.04 STIG バージョン 2 リリース 1
- Ubuntu 22.04 STIG バージョン 2 リリース 2

Linux STIG High (Category I)

- RHEL 7 STIG バージョン 3 リリース 15
- RHEL 8 STIG バージョン 2 リリース 1
- RHEL 9 STIG バージョン 2 リリース 2
- Ubuntu 18.04 STIG バージョン 2 リリース 15

- Ubuntu 20.04 STIG バージョン 2 リリース 1
- Ubuntu 22.04 STIG バージョン 2 リリース 2

2024 年第 3 四半期の変更-2024 年 10 月 4 日 (変更なし):

2024 年第 3 四半期リリースに対する Linux コンポーネント STIGS に変更はありませんでした。

2024 年第 2 四半期の変更 - 2024 年 5 月 10 日:

STIG バージョンを更新し、2024 年第 2 四半期リリースの STIG を適用しました。また、RHEL 9、CentOS Stream 9、Ubuntu 22.04 のサポートを以下のように追加しました。

Linux STIG Low (Category III)

- RHEL 7 STIG バージョン 3 リリース 14
- RHEL 8 STIG バージョン 1 リリース 14
- RHEL 9 STIG バージョン 1 リリース 3
- Ubuntu 18.04 STIG バージョン 2 リリース 14
- Ubuntu 20.04 STIG バージョン 1 リリース 12
- Ubuntu 22.04 STIG バージョン 1 リリース 1

Linux STIG Medium (Category II)

- RHEL 7 STIG バージョン 3 リリース 14
- RHEL 8 STIG バージョン 1 リリース 14
- RHEL 9 STIG バージョン 1 リリース 3
- Ubuntu 18.04 STIG バージョン 2 リリース 14
- Ubuntu 20.04 STIG バージョン 1 リリース 12
- Ubuntu 22.04 STIG バージョン 1 リリース 1

Linux STIG High (Category I)

- RHEL 7 STIG バージョン 3 リリース 14
- RHEL 8 STIG バージョン 1 リリース 14
- RHEL 9 STIG バージョン 1 リリース 3

- Ubuntu 18.04 STIG バージョン 2 リリース 14
- Ubuntu 20.04 STIG バージョン 1 リリース 12
- Ubuntu 22.04 STIG バージョン 1 リリース 1

2024 年第 1 四半期の変更 - 2024 年 2 月 6 日:

STIG バージョンを更新し、次のように 2024 年第 1 四半期リリースの STIG を適用しました。

Linux STIG Low (Category III)

- RHEL 7 STIG バージョン 3 リリース 14
- RHEL 8 STIG バージョン 1 リリース 13
- Ubuntu 18.04 STIG バージョン 2 リリース 13
- Ubuntu 20.04 STIG バージョン 1 リリース 11

Linux STIG Medium (Category II)

- RHEL 7 STIG バージョン 3 リリース 14
- RHEL 8 STIG バージョン 1 リリース 13
- Ubuntu 18.04 STIG バージョン 2 リリース 13
- Ubuntu 20.04 STIG バージョン 1 リリース 11

Linux STIG High (Category I)

- RHEL 7 STIG バージョン 3 リリース 14
- RHEL 8 STIG バージョン 1 リリース 13
- Ubuntu 18.04 STIG バージョン 2 リリース 13
- Ubuntu 20.04 STIG バージョン 1 リリース 11

2023 年第 4 四半期の変更 - 2023 年 12 月 7 日:

STIG バージョンを更新し、次のように 2023 年第 4 四半期リリースの STIG を適用しました。

Linux STIG Low (Category III)

- RHEL 7 STIG バージョン 3 リリース 13

- RHEL 8 STIG バージョン 1 リリース 12
- Ubuntu 18.04 STIG バージョン 2 リリース 12
- Ubuntu 20.04 STIG バージョン 1 リリース 10

Linux STIG Medium (Category II)

- RHEL 7 STIG バージョン 3 リリース 13
- RHEL 8 STIG バージョン 1 リリース 12
- Ubuntu 18.04 STIG バージョン 2 リリース 12
- Ubuntu 20.04 STIG バージョン 1 リリース 10

Linux STIG High (Category I)

- RHEL 7 STIG バージョン 3 リリース 13
- RHEL 8 STIG バージョン 1 リリース 12
- Ubuntu 18.04 STIG バージョン 2 リリース 12
- Ubuntu 20.04 STIG バージョン 1 リリース 10

2023 年第 3 四半期の変更-2023 年 4 月 10 日:

STIG のバージョンを更新し、2023 年第 3 四半期のリリースに向けて以下のように STIGS を適用しました。

Linux STIG Low (Category III)

- RHEL 7 STIG バージョン 3 リリース 12
- RHEL 8 STIG バージョン 1 リリース 11
- Ubuntu 18.04 STIG バージョン 2 リリース 11
- Ubuntu 20.04 STIG バージョン 1 リリース 9

Linux STIG Medium (Category II)

- RHEL 7 STIG バージョン 3 リリース 12
- RHEL 8 STIG バージョン 1 リリース 11

- Ubuntu 18.04 STIG バージョン 2 リリース 11
- Ubuntu 20.04 STIG バージョン 1 リリース 9

Linux STIG High (Category I)

- RHEL 7 STIG バージョン 3 リリース 12
- RHEL 8 STIG バージョン 1 リリース 11
- Ubuntu 18.04 STIG バージョン 2 リリース 11
- Ubuntu 20.04 STIG バージョン 1 リリース 9

2023 第 2 四半期の変更-2023 年 5 月 3 日:

STIG のバージョンを更新し、2023 年第 2 四半期のリリースに向けて以下のように STIGS を適用しました。

Linux STIG Low (Category III)

- RHEL 7 STIG バージョン 3 リリース 11
- RHEL 8 STIG バージョン 1 リリース 10
- Ubuntu 18.04 STIG バージョン 2 リリース 11
- Ubuntu 20.04 STIG バージョン 1 リリース 8

Linux STIG Medium (Category II)

- RHEL 7 STIG バージョン 3 リリース 11
- RHEL 8 STIG バージョン 1 リリース 10
- Ubuntu 18.04 STIG バージョン 2 リリース 11
- Ubuntu 20.04 STIG バージョン 1 リリース 8

Linux STIG High (Category I)

- RHEL 7 STIG バージョン 3 リリース 11
- RHEL 8 STIG バージョン 1 リリース 10
- Ubuntu 18.04 STIG バージョン 2 リリース 11
- Ubuntu 20.04 STIG バージョン 1 リリース 8

2023 年第 1 四半期の変更-2023 年 3 月 27 日:

STIG のバージョンを更新し、2023 年第 1 四半期のリリースに向けて以下のように STIGS を適用しました。

Linux STIG Low (Category III)

- RHEL 7 STIG バージョン 3 リリース 10
- RHEL 8 STIG バージョン 1 リリース 9
- Ubuntu 18.04 STIG バージョン 2 リリース 10
- Ubuntu 20.04 STIG バージョン 1 リリース 7

Linux STIG Medium (Category II)

- RHEL 7 STIG バージョン 3 リリース 10
- RHEL 8 STIG バージョン 1 リリース 9
- Ubuntu 18.04 STIG バージョン 2 リリース 10
- Ubuntu 20.04 STIG バージョン 1 リリース 7

Linux STIG High (Category I)

- RHEL 7 STIG バージョン 3 リリース 10
- RHEL 8 STIG バージョン 1 リリース 9
- Ubuntu 18.04 STIG バージョン 2 リリース 10
- Ubuntu 20.04 STIG バージョン 1 リリース 7

2022 年第 4 四半期の変更-2023 年 2 月 1 日:

STIG のバージョンを更新し、2022 年第 4 四半期のリリースに向けて以下のように STIGS を適用しました。

Linux STIG Low (Category III)

- RHEL 7 STIG バージョン 3 リリース 9
- RHEL 8 STIG バージョン 1 リリース 8
- Ubuntu 18.04 STIG バージョン 2 リリース 9

- Ubuntu 20.04 STIG バージョン 1 リリース 6

Linux STIG Medium (Category II)

- RHEL 7 STIG バージョン 3 リリース 9
- RHEL 8 STIG バージョン 1 リリース 8
- Ubuntu 18.04 STIG バージョン 2 リリース 9
- Ubuntu 20.04 STIG バージョン 1 リリース 6

Linux STIG High (Category I)

- RHEL 7 STIG バージョン 3 リリース 9
- RHEL 8 STIG バージョン 1 リリース 8
- Ubuntu 18.04 STIG バージョン 2 リリース 9
- Ubuntu 20.04 STIG バージョン 1 リリース 6

2022 年第 3 四半期の変更-2022 年 9 月 30 日 (変更なし):

2022 年第 3 四半期リリースの Linux コンポーネント STIGS には変更はありませんでした。

2022 年第 2 四半期の変更-2022 年 8 月 2 日:

Ubuntu サポートの導入、STIG バージョンの更新、および 2022 年第 2 四半期リリース向けの STIGS の適用を以下のとおり実施しました。

Linux STIG Low (Category III)

- RHEL 7 STIG バージョン 3 リリース 7
- RHEL 8 STIG バージョン 1 リリース 6
- Ubuntu 18.04 STIG バージョン 2 リリース 6 (新規)
- Ubuntu 20.04 STIG バージョン 1 リリース 4 (新規)

Linux STIG Medium (Category II)

- RHEL 7 STIG バージョン 3 リリース 7

- RHEL 8 STIG バージョン 1 リリース 6
- Ubuntu 18.04 STIG バージョン 2 リリース 6 (新規)
- Ubuntu 20.04 STIG バージョン 1 リリース 4 (新規)

Linux STIG High (Category I)

- RHEL 7 STIG バージョン 3 リリース 7
- RHEL 8 STIG バージョン 1 リリース 6
- Ubuntu 18.04 STIG バージョン 2 リリース 6 (新規)
- Ubuntu 20.04 STIG バージョン 1 リリース 4 (新規)

2022 年第 1 四半期の変更-2022 年 4 月 26 日:

コンテナのサポートを強化するようにリファクタリングされました。以前の AL2 スクリプトと RHEL 7 を組み合わせました。STIG のバージョンを更新し、2022 年第 1 四半期のリリースに向けて以下のように STIGS を適用しました。

Linux STIG Low (Category III)

- RHEL 7 STIG バージョン 3 リリース 6
- RHEL 8 STIG バージョン 1 リリース 5

Linux STIG Medium (Category II)

- RHEL 7 STIG バージョン 3 リリース 6
- RHEL 8 STIG バージョン 1 リリース 5

Linux STIG High (Category I)

- RHEL 7 STIG バージョン 3 リリース 6
- RHEL 8 STIG バージョン 1 リリース 5

2021 年第 4 四半期の変更-2021 年 12 月 20 日:

STIG のバージョンを更新し、2021 年第 4 四半期のリリースに向けて以下のように STIGS を適用しました。

Linux STIG Low (Category III)

- RHEL 7 STIG バージョン 3 リリース 5
- RHEL 8 STIG バージョン 1 リリース 4

Linux STIG Medium (Category II)

- RHEL 7 STIG バージョン 3 リリース 5
- RHEL 8 STIG バージョン 1 リリース 4

Linux STIG High (Category I)

- RHEL 7 STIG バージョン 3 リリース 5
- RHEL 8 STIG バージョン 1 リリース 4

2021 年第 3 四半期の変更-2021 年 9 月 30 日:

STIG のバージョンを更新し、2021 年第 3 四半期のリリースに向けて以下のように STIGS を適用しました。

Linux STIG Low (Category III)

- RHEL 7 STIG バージョン 3 リリース 4
- RHEL 8 STIG バージョン 1 リリース 3

Linux STIG Medium (Category II)

- RHEL 7 STIG バージョン 3 リリース 4
- RHEL 8 STIG バージョン 1 リリース 3

Linux STIG High (Category I)

- RHEL 7 STIG バージョン 3 リリース 4
- RHEL 8 STIG バージョン 1 リリース 3

Windows STIG バージョン履歴

このセクションでは、四半期ごとに更新される Windows スクリプトバンドルのバージョン履歴を口グに記録します。四半期ごとの変更点と公開されたバージョンを確認するには、タイトルを選択して情報を展開します。四半期中に変更がない場合、そのことがタイトルに反映されます。

2026 年 第 1 四半期の変更 - 04/07/2026:

Windows Server 2025 オペレーティングシステムのサポートを追加し、次の STIG バージョンを更新し、2026 年 第 1 四半期リリースの STIGS をすべてのコンプライアンスレベル (低/中/高) に対して適用しました。

STIG-Build-Windows バージョン 1.0.x

- Windows Server 2025 STIG バージョン 1 リリース 1
- Windows Server 2022 STIG バージョン 2 リリース 7
- Windows Server 2019 STIG バージョン 3 リリース 7
- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 7
- Windows Firewall STIG バージョン 2 リリース 2
- Internet Explorer 11 STIG バージョン 2 リリース 6
- Microsoft Edge STIG バージョン 2 リリース 4 (Windows Server 2022 および 2025 のみ)

2025 年第 4 四半期の変更 - 2025 年 12 月 10 日 (変更なし):

2025 年第 4 四半期リリースの Windows コンポーネント STIGS には変更はありませんでした。

2025 年第 3 四半期の変更-2025 年 9 月 4 日 (変更なし):

2025 年第 3 四半期リリースに対する Windows コンポーネント STIGS に変更はありませんでした。

2025 年第 2 四半期の変更 - 2025 年 6 月 26 日:

STIG のバージョンを更新し、2025 年第 2 四半期のリリースに対して以下のように STIGS を適用しました。

Windows STIG Low (Category III)

- Windows Server 2022 STIG バージョン 2 リリース 4

- Windows Server 2019 STIG バージョン 3 リリース 4
- Windows Server 2016 STIG バージョン 2 リリース 10
- Windows Server 2012 R2 MS STIG バージョン 3 リリース 5
- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 6
- Windows Firewall STIG バージョン 2 リリース 2
- Internet Explorer 11 STIG バージョン 2 リリース 5
- Microsoft Edge STIG バージョン 2 リリース 2 (Windows Server 2022 のみ)

Windows STIG Medium (Category II)

- Windows Server 2022 STIG バージョン 2 リリース 4
- Windows Server 2019 STIG バージョン 3 リリース 4
- Windows Server 2016 STIG バージョン 2 リリース 10
- Windows Server 2012 R2 MS STIG バージョン 3 リリース 5
- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 6
- Windows Firewall STIG バージョン 2 リリース 2
- Internet Explorer 11 STIG バージョン 2 リリース 5
- Microsoft Edge STIG バージョン 2 リリース 2 (Windows Server 2022 のみ)
- Defender STIG バージョン 2 リリース 4

Windows STIG High (Category I)

- Windows Server 2022 STIG バージョン 2 リリース 4
- Windows Server 2019 STIG バージョン 3 リリース 4
- Windows Server 2016 STIG バージョン 2 リリース 10
- Windows Server 2012 R2 MS STIG バージョン 3 リリース 5
- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 6
- Windows Firewall STIG バージョン 2 リリース 2
- Internet Explorer 11 STIG バージョン 2 リリース 5
- Microsoft Edge STIG バージョン 2 リリース 2 (Windows Server 2022 のみ)
- Defender STIG バージョン 2 リリース 4

2025 年 Q1 四半期の変更 - 2025 年 5 月 4 日:

2025 年第 1 四半期リリースに対するすべての STIG コンポーネントで、Internet Explorer 11 STIG バージョン 2 リリース 5 の STIGS を更新しました。

2024 年第 4 四半期の変更 - 2024 年 12 月 10 日:

STIG のバージョンを更新し、2024 年第 4 四半期のリリースに対して以下のように STIGS を適用しました。

Windows STIG Low (Category III)

- Windows Server 2022 STIG バージョン 2 リリース 2
- Windows Server 2019 STIG バージョン 3 リリース 2
- Windows Server 2016 STIG バージョン 2 リリース 9
- Windows Server 2012 R2 MS STIG バージョン 3 リリース 5
- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 2
- Windows Firewall STIG バージョン 2 リリース 2
- Internet Explorer 11 STIG バージョン 2 リリース 5
- Microsoft Edge STIG バージョン 2 リリース 2 (Windows Server 2022 のみ)

Windows STIG Medium (Category II)

- Windows Server 2022 STIG バージョン 2 リリース 2
- Windows Server 2019 STIG バージョン 3 リリース 2
- Windows Server 2016 STIG バージョン 2 リリース 9
- Windows Server 2012 R2 MS STIG バージョン 3 リリース 5
- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 2
- Windows Firewall STIG バージョン 2 リリース 2
- Internet Explorer 11 STIG バージョン 2 リリース 5
- Microsoft Edge STIG バージョン 2 リリース 2 (Windows Server 2022 のみ)
- Defender STIG バージョン 2 リリース 4

Windows STIG High (Category I)

- Windows Server 2022 STIG バージョン 2 リリース 2

- Windows Server 2019 STIG バージョン 3 リリース 2
- Windows Server 2016 STIG バージョン 2 リリース 9
- Windows Server 2012 R2 MS STIG バージョン 3 リリース 5
- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 2
- Windows Firewall STIG バージョン 2 リリース 2
- Internet Explorer 11 STIG バージョン 2 リリース 5
- Microsoft Edge STIG バージョン 2 リリース 2 (Windows Server 2022 のみ)
- Defender STIG バージョン 2 リリース 4

2024 年第 3 四半期の変更-2024 年 10 月 4 日 (変更なし):

2024 年第 3 四半期リリースの Windows コンポーネント STIGS に変更はありませんでした。

2024 年第 2 四半期の変更 - 2024 年 5 月 10 日 (変更なし):

2024 年第 2 四半期リリースの Windows コンポーネント STIG には変更はありませんでした。

2024 年第 1 四半期の変更 - 2024 年 2 月 23 日 (変更なし):

2024 年第 1 四半期リリースの Windows コンポーネント STIG には変更はありませんでした。

2023 年第 4 四半期の変更 - 2023 年 12 月 7 日 (変更なし):

2023 年第 4 四半期リリースの Windows コンポーネント STIG には変更はありませんでした。

2023 年第 3 四半期の変更-2023 年 4 月 10 日 (変更なし):

2023 年第 3 四半期リリースの Windows コンポーネント STIGS には変更はありませんでした。

2023 年第 2 四半期の変更-2023 年 5 月 3 日 (変更なし):

2023 年第 2 四半期リリースの Windows コンポーネント STIGS には変更はありませんでした。

2023 年第 1 四半期の変更-2023 年 3 月 27 日 (変更なし):

2023 年第 1 四半期リリースの Windows コンポーネント STIGS には変更はありませんでした。

2022 年第 4 四半期の変更-2023 年 2 月 1 日:

STIG のバージョンを更新し、2022 年第 4 四半期のリリースに向けて以下のように STIGS を適用しました。

Windows STIG Low (Category III)

- Windows Server 2022 STIG バージョン 1 リリース 1
- Windows Server 2019 STIG バージョン 2 リリース 5
- Windows Server 2016 STIG バージョン 2 リリース 5
- Windows Server 2012 R2 MS STIG バージョン 3 リリース 5
- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 2
- Windows Firewall STIG バージョン 2 リリース 1
- Internet Explorer 11 STIG バージョン 2 リリース 3
- Microsoft Edge STIG バージョン 1 リリース 6 (Windows Server 2022 のみ)

Windows STIG Medium (Category II)

- Windows Server 2022 STIG バージョン 1 リリース 1
- Windows Server 2019 STIG バージョン 2 リリース 5
- Windows Server 2016 STIG バージョン 2 リリース 5
- Windows Server 2012 R2 MS STIG バージョン 3 リリース 5
- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 2
- Windows Firewall STIG バージョン 2 リリース 1
- Internet Explorer 11 STIG バージョン 2 リリース 3
- Microsoft Edge STIG バージョン 1 リリース 6 (Windows Server 2022 のみ)
- Defender STIG バージョン 2 リリース 4 (Windows Server 2022 のみ)

Windows STIG High (Category I)

- Windows Server 2022 STIG バージョン 1 リリース 1
- Windows Server 2019 STIG バージョン 2 リリース 5
- Windows Server 2016 STIG バージョン 2 リリース 5
- Windows Server 2012 R2 MS STIG バージョン 3 リリース 5
- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 2
- Windows Firewall STIG バージョン 2 リリース 1
- Internet Explorer 11 STIG バージョン 2 リリース 3

- Microsoft Edge STIG バージョン 1 リリース 6 (Windows Server 2022 のみ)
- Defender STIG バージョン 2 リリース 4 (Windows Server 2022 のみ)

2022 年第 3 四半期の変更-2022 年 9 月 30 日 (変更なし):

2022 年第 3 四半期リリースの Windows コンポーネント STIGS には変更はありませんでした。

2022 年第 2 四半期の変更-2022 年 8 月 2 日:

STIG のバージョンを更新し、2022 年第 2 四半期のリリースに向けて STIGS を適用しました。

Windows STIG Low (Category III)

- Windows Server 2019 STIG バージョン 2 リリース 4
- Windows Server 2016 STIG バージョン 2 リリース 4
- Windows Server 2012 R2 MS STIG バージョン 3 リリース 3
- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 1
- Windows Firewall STIG バージョン 2 リリース 1
- Internet Explorer 11 STIG バージョン 1 リリース 19

Windows STIG Medium (Category II)

- Windows Server 2019 STIG バージョン 2 リリース 4
- Windows Server 2016 STIG バージョン 2 リリース 4
- Windows Server 2012 R2 MS STIG バージョン 3 リリース 3
- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 1
- Windows Firewall STIG バージョン 2 リリース 1
- Internet Explorer 11 STIG バージョン 1 リリース 19

Windows STIG High (Category I)

- Windows Server 2019 STIG バージョン 2 リリース 4
- Windows Server 2016 STIG バージョン 2 リリース 4
- Windows Server 2012 R2 MS STIG バージョン 3 リリース 3

- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 1
- Windows Firewall STIG バージョン 2 リリース 1
- Internet Explorer 11 STIG バージョン 1 リリース 19

2022 年第 1 四半期の変更-2022 年 8 月 2 日 (変更なし):

2022 年第 1 四半期リリースの Windows コンポーネント STIGS には変更はありませんでした。

2021 年第 4 四半期の変更-2021 年 12 月 20 日:

STIG のバージョンを更新し、2021 年第 4 四半期のリリースに向けて STIGS を適用しました。

Windows STIG Low (Category III)

- Windows Server 2019 STIG バージョン 2 リリース 3
- Windows Server 2016 STIG バージョン 2 リリース 3
- Windows Server 2012 R2 MS STIG バージョン 3 リリース 3
- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 1
- Windows Firewall STIG バージョン 2 リリース 1
- Internet Explorer 11 STIG バージョン 1 リリース 19

Windows STIG Medium (Category II)

- Windows Server 2019 STIG バージョン 2 リリース 3
- Windows Server 2016 STIG バージョン 2 リリース 3
- Windows Server 2012 R2 MS STIG バージョン 3 リリース 3
- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 1
- Windows Firewall STIG バージョン 2 リリース 1
- Internet Explorer 11 STIG バージョン 1 リリース 19

Windows STIG High (Category I)

- Windows Server 2019 STIG バージョン 2 リリース 3
- Windows Server 2016 STIG バージョン 2 リリース 3
- Windows Server 2012 R2 MS STIG バージョン 3 リリース 3

- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 1
- Windows Firewall STIG バージョン 2 リリース 1
- Internet Explorer 11 STIG バージョン 1 リリース 19

2021 年第 3 四半期の変更-2021 年 9 月 30 日:

STIG のバージョンを更新し、2021 年第 3 四半期のリリースに向けて STIGS を適用しました。

Windows STIG Low (Category III)

- Windows Server 2019 STIG バージョン 2 リリース 2
- Windows Server 2016 STIG バージョン 2 リリース 2
- Windows Server 2012 R2 MS STIG バージョン 3 リリース 2
- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 1
- Windows ファイアウォール STIG バージョン 1 リリース 7
- Internet Explorer 11 STIG バージョン 1 リリース 19

Windows STIG Medium (Category II)

- Windows Server 2019 STIG バージョン 2 リリース 2
- Windows Server 2016 STIG バージョン 2 リリース 2
- Windows Server 2012 R2 MS STIG バージョン 3 リリース 2
- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 1
- Windows ファイアウォール STIG バージョン 1 リリース 7
- Internet Explorer 11 STIG バージョン 1 リリース 19

Windows STIG High (Category I)

- Windows Server 2019 STIG バージョン 2 リリース 2
- Windows Server 2016 STIG バージョン 2 リリース 2
- Windows Server 2012 R2 MS STIG バージョン 3 リリース 2
- Microsoft .NET Framework 4.0 STIG バージョン 2 リリース 1
- Windows ファイアウォール STIG バージョン 1 リリース 7

- Internet Explorer 11 STIG バージョン 1 リリース 19

AWS Systems Manager を使用してインスタンスに STIG 設定を適用する

AWSEC2-ConfigureSTIG Systems Manager コマンドドキュメントを使用すると、STIG 設定を既存の EC2 インスタンスに 適用できます。コマンドドキュメントは更新するインスタンスから実行する必要があります。コマンドドキュメントは、オペレーティングシステムと、実行されるインスタンスの設定に基づいて、適切な設定を適用します。

このページでは AWSEC2-ConfigureSTIG コマンドドキュメントの詳細について、入力パラメータや、Systems Manager コンソールでの実行方法または AWS CLI で send-command を使用して実行する方法などを含めて説明します。

AWSEC2-ConfigureSTIG 入力パラメータ

次の入力パラメータを指定すると、コマンドドキュメントがインスタンスに STIG 設定を適用する方法を指定できます。

レベル (文字列、必須)

適用する STIG 重要度のカテゴリを指定します。有効な値には次のようなものがあります。

- 高
- 中程度
- 低

値を指定しない場合、システムによるデフォルトは High になります。

InstallPackages (文字列、オプション – Linux のみ)

値が No である場合、スクリプトは追加のソフトウェアパッケージをインストールしません。値が Yes である場合、スクリプトはコンプライアンスを最大化するために必要な追加のソフトウェアパッケージをインストールします。デフォルト値は No です。

SetDoDConsentBanner (文字列、オプション – Linux のみ)

値が No である場合、Linux STIG スクリプトがインストールされているインスタンスにアタッチすると DoD の同意バナーは表示されません。値が Yes である場合、STIG Linux スクリプトのいずれかがインストールされているインスタンスに接続すると、ログインする前に DoD 同意バナーが表示されます。バナーはログインする前に確認する必要があります。デフォルト値は No です。

同意バナーの例については、DLA ドキュメントサービスのウェブサイトにアクセスしたときに表示される Disclaimer Department of Defense Privacy and Consent Notice を参照してください。

AWSEC2-ConfigureSTIG コマンドドキュメントを実行します。

AWSEC2-ConfigureSTIG ドキュメントを実行するには、ご希望の環境に応じた手順に従ってください。

Console

1. AWS Systems Manager コンソール (https://console.aws.amazon.com/systems-manager/) を開きます。
2. ナビゲーションペインから [コマンドの実行] をクリックします。該当する場合、アカウントで現在実行されているコマンドのリストが表示されます。
3. [コマンドの実行] を選択します。これにより、[コマンドの実行] ダイアログが開き、アクセスできるコマンドドキュメントのリストが表示されます。
4. コマンドドキュメントのリストから AWSEC2-ConfigureSTIG を選択します。結果を効率化するために、ドキュメント名の全体または一部を入力できます。所有者、プラットフォームタイプ、またはタグでフィルタリングすることもできます。

   コマンドドキュメントを選択すると、詳細がリストの下に表示されます。
5. [ドキュメントバージョン] リストから Default version at runtime を選択します。
6. コマンドパラメータを設定し、AWSEC2-ConfigureSTIG がスクリプトパッケージをインストールして実行し、インスタンスを更新する方法を定義します。パラメータの詳細については、「AWSEC2-ConfigureSTIG 入力パラメータ」を参照してください。
7. [ターゲットの選択] で、タグを指定するか、手動でインスタンスを選択して、この操作を実行するインスタンスを特定します。

   > **Note**
   >
   > インスタンスを手動で選択することにしたが、そのインスタンスがリストに表示されない場合は、「インスタンスの場所」でトラブルシューティングのヒントを参照してください。
8. [レート制御] などの Systems Manager Run Command の動作を定義する追加パラメータについては、「コンソールからのコマンドの実行」の説明に従って値を入力します。

9. [Run] (実行) を選択します。

   正常に実行されると、コマンドドキュメントはスクリプトをインストールし、インスタンスを設定します。コマンドの実行が失敗した場合、Systems Manager コマンド出力で、実行が失敗した理由の詳細を確認してください。

AWS CLI

例 1: デフォルト値で実行する

次のコマンドを実行して STIG スクリプトをインストールし、デフォルト値でこれを実行します。入力パラメータの詳細については、「AWSEC2-ConfigureSTIG 入力パラメータ」を参照してください。

```
aws ssm send-command \
 --document-name "AWSEC2-ConfigureSTIG" \
 --instance-ids "i-1234567890abcdef0"'
```

例 2: インスタンスで中レベルの STIG 設定を構成する

次のコマンドを実行して STIG スクリプトをインストールし、`Level` 入力パラメータを `Medium` に設定して実行します。入力パラメータの詳細については、「AWSEC2-ConfigureSTIG 入力パラメータ」を参照してください。

```
aws ssm send-command \
 --document-name "AWSEC2-ConfigureSTIG" \
 --instance-ids "i-1234567890abcdef0"
 --parameters '{"Level":"Medium"}'
```

正常に実行されると、コマンドドキュメントはスクリプトをインストールし、インスタンスを設定します。コマンドの実行が失敗した場合、 コマンド出力で、実行が失敗した理由の詳細を確認してください。

PowerShell

例 1: デフォルト値で実行する

次のコマンドを実行して STIG スクリプトをインストールし、デフォルト値でこれを実行します。入力パラメータの詳細については、「AWSEC2-ConfigureSTIG 入力パラメータ」を参照してください。

```
Send-SSMCommand -DocumentName "AWSEC2-ConfigureSTIG" -InstanceId
 "i-1234567890abcdef0"}
```

例 2: インスタンスで中レベルの STIG 設定を構成する

次のコマンドを実行して STIG スクリプトをインストールし、Level 入力パラメータを Medium に設定して実行します。入力パラメータの詳細については、「AWSEC2-ConfigureSTIG 入力パラメータ」を参照してください。

```
Send-SSMCommand -DocumentName "AWSEC2-ConfigureSTIG" -InstanceId
 "i-1234567890abcdef0" -Parameter @{'Level'='Medium'}
```

正常に実行されると、コマンドドキュメントはスクリプトをインストールし、インスタンスを設定します。コマンドの実行が失敗した場合、 コマンド出力で、実行が失敗した理由の詳細を確認してください。

EC2 インスタンス上の正確なクロックと時刻の同期

多くのサーバータスクとプロセスにとって、Amazon EC2 インスタンスでの一貫して正確な時刻のリファレンスが不可欠です。システムログのタイムスタンプは問題が発生した時期やイベントの時系列を特定する上で重要な役割を果たします。AWS CLI または AWS SDK を使用してインスタンスからリクエストを行う際に、これらのツールによって自動的にリクエストに署名されます。インスタンスの日時設定が不正確な場合、署名の日付とリクエストの日付が一致しないことがあり、その場合は AWS がリクエストを却下します。

これに対処することが重要なため、Amazon は Amazon Time Sync Service を提供しています。このサービスはすべての EC2 インスタンスからアクセスでき、さまざまな AWS のサービス で利用されます。このサービスは各 AWS リージョン で衛星接続された基準となる原子時計のフリートを使用して、世界標準時 (UTC) の正確な現在時刻表示を配信します。

最高のパフォーマンスを得るにはEC2 インスタンスでローカル Amazon Time Sync Service を使用することをお勧めします。インスタンスのローカル Amazon Time Sync Service へのバックアップのために、または Amazon EC2 外部のリソースの Amazon Time Sync Service に接続するために、time.aws.com にあるパブリック Amazon Time Sync Service を使用できます。パブリック Amazon Time Sync Service はローカル Amazon Time Sync Service と同様に UTC に追加されたうるう秒の Leap Smearing を自動的に行います。パブリック Amazon Time Sync Service は各 AWS リージョン で衛星接続された基準となる原子時計のフリートにより、世界中でサポートされています。

ハードウェアパケットタイムスタンプ

インスタンスでハードウェアパケットタイムスタンプを有効にして、すべての受信ネットワークパケットに 64 ビットのナノ秒精度のタイムスタンプを追加できます。ハードウェアパケットのタイムスタンプは、パケットがカーネル、ソケット、またはアプリケーションレイヤーに到達する前にハードウェアレベルで発生するため、ソフトウェアのタイムスタンプによって追加される遅延が回避されます。ハードウェアタイムスタンプの基盤となるリファレンスクロックは、Amazon Time Sync Service PTP ハードウェアクロックです。

利点

ハードウェアパケットのタイムスタンプには、次の利点があります。

- イベントの順序付けが改善されます。これは、パケットが EC2 インスタンスに到着する実際の順序を決定して公平なパケット処理を確保するためにも使用できます。
- 一方向ネットワークレイテンシーを測定します。
- ほとんどのオンプレミスソリューションと比較して、より高い精度と正確性で分散トランザクションの速度が向上します。

設定の前提条件

ハードウェアパケットのタイムスタンプを有効にするには、インスタンスが次の前提条件を満たす必要があります。

- Linux インスタンスである必要があります。
- PTP ハードウェアクロックをサポートする要件を満たす必要があります。

設定手順については、GitHub の「Linux kernel driver for Elastic Network Adapter (ENA) family」ページの「Hardware Packet Timestamping」を参照してください。

うるう秒

1972 年に導入されたうるう秒は国際原子時 (TAI) と太陽時 (Ut1) の違いに対応するため、地球の自転の不規則性を考慮して UTC 時刻をときどき 1 秒調整するものです。お客様に代わってうるう秒を管理するために、当社は Amazon Time Sync Service 内での Leap Smearing を設計しました。詳細については「うるう秒に備える — 迫り来るうるう秒と AWS」を参照してください。

うるう秒はなくなりつつあります。当社は2035 年までにうるう秒を廃止するという第 27 回国際度量衡総会で採択された決議を全面的に支持しています。

この移行をサポートするために、ローカル NTP 接続または当社のパブリック NTP プール (`time.aws.com`) 経由で Amazon Time Sync Service にアクセスする場合、うるう秒の発生中に Leap Smearing を引き続き計画しています。ただし、PTP ハードウェアクロックには Leap Smearing のオプションはありません。うるう秒が発生した場合、PTP ハードウェアクロックは UTC 標準に従ってうるう秒を追加します。Leap Smearing を行う時刻の供給元とうるう秒を挿入する時刻の供給元はほとんどの場合同様です。ただし、うるう秒の発生中は両者が異なるため、うるう秒の発生中はタイムクライアントの設定で Leap Smearing を行う時刻の供給元と行わない時刻の供給元の両方を使用することはお勧めしません。

トピック

- EC2 インスタンスのタイムリファレンスを、ローカル Amazon Time Sync Service を使用するように設定します。
- EC2 インスタンスまたはインターネットに接続されたデバイスのタイムリファレンスを、パブリック Amazon Time Sync Service を使用するように設定する
- Linux インスタンスのタイムスタンプを比較する
- インスタンスのタイムゾーンを変更する

関連リソース

- AWS コンピューティングブログ: 今がそのとき: Amazon EC2 インスタンスでマイクロ秒精度のクロック
- AWS クラウドの運用と移行のブログ: Manage Amazon EC2 instance clock accuracy using Amazon Time Sync Service and Amazon CloudWatch – Part 1
- (Linux) https://chrony-project.org/

EC2 インスタンスのタイムリファレンスを、ローカル Amazon Time Sync Service を使用するように設定します。

Amazon Time Sync Service には、Amazon EC2 インスタンスがローカルタイムソースと同期するためのいくつかの方法が用意されています。まず、どの Amazon EC2 インスタンスも Network Time Protocol (NTP) を介してローカルタイムソースにアクセスできます。さらに、拡張 Amazon Time Sync Service は、サポートされている Amazon EC2 インスタンスに対して、より高精度のローカルタイムソースを提供します。より高精度の NTP ソースにアクセスする `precision-time` 戦略を使用して、プレイスメントグループでサポートされているインスタンスを起動します。最後に、高精度タイムプレイスメントグループで起動された Linux インスタンスは、PTP ハードウェアクロック (PHC) デバイスにアクセスし、ハードウェアパケットタイムスタンプを取得できます。

すべての Amazon EC2 インスタンスは、ローカル NTP ソースにアクセスできます。リンクローカル IP アドレスを介して NTP ソースにアクセスできます。これにより、特定の VPC 設定を変更することなく、このトラフィックをお使いの VPC 内に制限できます。AMI は、デフォルトでローカル NTP ソースを使用するようにクロック同期デーモンを設定済みである可能性があります。この NTP ソースは、次の IP アドレスで使用できます。

- IPv4:`169.254.169.123`
- IPv6: `fd00:ec2::123` (Nitro ベースのインスタンスでのみアクセス可能。)

サポートされている Amazon EC2 インスタンスは、拡張 Amazon Time Sync Service にアクセスできます。拡張 Amazon Time Sync Service にアクセスするには、`precision-time` 戦略を使用してプレイスメントグループでサポートされているインスタンスを起動します。NTP リンクローカル IP アドレスを使用する場合、拡張機能のメリットを享受するためにインスタンスを設定する必要はありません。どのオペレーティングシステムでもこの拡張機能を使用できます。選択した NTP クライアントを使用して、拡張 NTP ソースのメリットを享受できているかどうかを確認できます。

サポートされているインスタンスファミリーで実行される Linux ベースの AMI には、PHC デバイスからタイムソースを取得する追加オプションがあります。ENA ドライバーはこのデバイスを使用可能にします。拡張 NTP ソースと PHC デバイスの両方が、同じ高精度のタイムソースを使用します。これにより、PHC タイムソースへのアクセスが最適化され、Amazon EC2 インスタンスをより正確に同期できます。

考慮事項

- NTP タイムソースは UTC タイムスケールの Leap Smearing (うるう秒の調整) ビューを提供しますが、PHC はうるう秒の調整を行いません。詳細については、「うるう秒」を参照してください。
- Linux インスタンスのみがローカル PTP ハードウェアクロックにアクセスできます。Windows インスタンスが拡張 Amazon Time Sync Service にアクセスするには、 NTP を使用する必要があります。
- リンクローカルアドレスを使用するサービスには 1,024 パケット/秒 (PPS) の制限があります。この制限には、Route 53 Resolver DNS クエリ、インスタンスメタデータサービス (IMDS) リクエスト、Amazon Time Sync Service Network Time Protocol (NTP) リクエスト、および Windows Licensing Service (Microsoft Windows ベースのインスタンス向け) リクエストの総計が含まれます。

内容

- Amazon Time Sync Service の IPv4 エンドポイントにアクセスする
- Amazon Time Sync Service の IPv6 エンドポイントにアクセスする
- 拡張 Amazon Time Sync Service にアクセスする
- PTP ハードウェアクロック (PHC) にアクセスする

Amazon Time Sync Service の IPv4 エンドポイントにアクセスする

AMI では、Amazon Time Sync Service がデフォルトですでに設定されている可能性があります。それ以外の場合は、次の手順に従って、IPv4 エンドポイントを介してローカル Amazon Time Sync Service を使用するようにインスタンスを設定します。

問題のトラブルシューティングについては、「Amazon EC2 Linux サーバーの NTP 同期の問題をトラブルシューティングする方法を教えてください。」または「EC2 Windows インスタンスの時間に関する問題のトラブルシューティング方法を教えてください。」を参照してください。

Amazon Linux

AL2023 および Amazon Linux 2 の最新バージョンは、デフォルトで Amazon Time Sync Service の IPv4 エンドポイントを使用するように設定されています。インスタンスが既に設定されていることを確認したら、次の手順をスキップできます。

chrony が IPv4 エンドポイントを使用するように設定されていることを確認するには

以下のコマンドを実行してください。出力では、^* から始まる行は優先時刻ソースが示されます。

```
[ec2-user ~]$ chronyc sources -v | grep -F ^*
```

```
^* 169.254.169.123               3   4   377    13  -4325ns[-9201ns] +/-  401us
```

古いバージョンの Amazon Linux 2 で IPv4 エンドポイントに接続するように chrony を設定するには

1. インスタンスに接続し、NTP サービスをアンインストールします。

   ```
   [ec2-user ~]$ sudo yum erase 'ntp*'
   ```

2. chrony パッケージをインストールします。

   ```
   [ec2-user ~]$ sudo yum install chrony
   ```

3. 任意のテキストエディタ (例: /etc/chrony.conf または vim など) を使って nano ファイルを開きます。ファイルに既に存在する他の server ステートメントや pool ステートメントの前に次の行を追加し、変更を保存します。

   ```
   server 169.254.169.123 prefer iburst minpoll 4 maxpoll 4
   ```

4. chrony デーモン (chronyd) を再起動します。

   ```
   [ec2-user ~]$ sudo service chronyd restart
   ```

   ```
   Starting chronyd:                                          [  OK  ]
   ```

   > **Note**
   >
   > RHEL と CentOS (バージョン 6 まで) ではサービス名は chrony ではなく chronyd です。

5. システムがブートするたびに起動するように chronyd を設定するにはchkconfig を使用します。

```
[ec2-user ~]$ sudo chkconfig chronyd on
```

6. chrony が IPv4 エンドポイント 169.254.169.123 を使用して時刻を同期させていることを確認します。

```
[ec2-user ~]$ chronyc sources -v | grep -F ^*
```

出力では、^* が優先時刻ソースを示します。

```
^* 169.254.169.123               3   6    17    43    -30us[ -226us] +/-  287us
```

7. chrony で報告された時刻同期メトリクスを確認します。

```
[ec2-user ~]$ chronyc tracking
```

```
Reference ID    : A9FEA97B (169.254.169.123)
Stratum         : 4
Ref time (UTC)  : Wed May 06 00:39:14 2026
System time     : 0.000002191 seconds fast of NTP time
Last offset     : +0.000002164 seconds
RMS offset      : 0.000082968 seconds
Frequency       : 3.710 ppm slow
Residual freq   : +0.002 ppm
Skew            : 0.504 ppm
Root delay      : 0.000362541 seconds
Root dispersion : 0.000225028 seconds
Update interval : 16.1 seconds
Leap status     : Normal
```

Ubuntu

Ubuntu の IPv4 エンドポイントに接続するように chrony を設定するには

1. インスタンスに接続し、apt を使用して chrony パッケージをインストールします。

```
ubuntu:~$ sudo apt install chrony
```

> **Note**
>
> 必要に応じて、sudo apt update を実行してインスタンスを最初に更新します。

2. 任意のテキストエディタ (例: /etc/chrony/chrony.conf または vim など) を使って nano ファイルを開きます。ファイルに既に存在する他の server ステートメントや pool ステートメントの前に次の行を追加し、変更を保存します。

```
server 169.254.169.123 prefer iburst minpoll 4 maxpoll 4
```

3. chrony サービスを再起動します。

```
ubuntu:~$ sudo /etc/init.d/chrony restart
```

```
Restarting chrony (via systemctl): chrony.service.
```

4. chrony が IPv4 エンドポイント 169.254.169.123 を使用して時刻を同期させていることを確認します。

```
ubuntu:~$ chronyc sources -v | grep -F ^*
```

出力では、^* から始まる行は優先時刻ソースが示されます。

```
^* 169.254.169.123               3   6    17    12    +15us[  +57us] +/-  320us
```

5. chrony で報告された時刻同期メトリクスを確認します。

```
ubuntu:~$ chronyc tracking
```

```
Reference ID    : A9FEA97B (169.254.169.123)
Stratum         : 4
Ref time (UTC)  : Wed May 06 00:39:14 2026
System time     : 0.000002191 seconds fast of NTP time
Last offset     : +0.000002164 seconds
RMS offset      : 0.000082968 seconds
Frequency       : 3.710 ppm slow
Residual freq   : +0.002 ppm
Skew            : 0.504 ppm
```

```
Root delay      : 0.000362541 seconds
Root dispersion : 0.000225028 seconds
Update interval : 16.1 seconds
Leap status     : Normal
```

SUSE Linux

SUSE Linux Enterprise Server 15 以降、chrony は NTP にデフォルトで実装されています。

SUSE Linux で IPv4 エンドポイントに接続するように chrony を設定するには

1. 任意のテキストエディタ (例: /etc/chrony.conf または vim など) を使って nano ファイルを開きます。
2. ファイルに次の行が含まれていることを確認します。

   ```
   server 169.254.169.123 prefer iburst minpoll 4 maxpoll 4
   ```

   この行が存在しない場合は追加します。
3. 他のサーバーまたはプールの行はすべてコメントアウトします。
4. YaST を開き、chrony サービスを有効にします。

Windows

2018 年 8 月のリリースから、Windows AMI はデフォルトで Amazon Time Sync Service を使用します。これらの AMI から起動されるインスタンスにはこれ以上の設定は不要で、以下の手順はスキップできます。

Amazon Time Sync Service がデフォルトで設定されていない AMI を使用している場合はまず現在の NTP 設定を確認します。インスタンスが既に Amazon Time Sync Service の IPv4 エンドポイントを使用している場合はそれ以上設定する必要はありません。インスタンスが Amazon Time Sync Service を使用していない場合はAmazon Time Sync Service を使用するように NTP サーバーを変更する手順を完了してください。

NTP 設定を確認するには

1. インスタンスで、コマンドプロンプトウィンドウを開きます。
2. 次のコマンドを入力して、現在の NTP 設定を取得します。

```
w32tm /query /configuration
```

このコマンドはWindows インスタンスの現在の設定を返し、Amazon Time Sync Service に接続しているかどうかを表示します。

3. (オプション) 次のコマンドを入力して、現在の設定のステータスを取得します。

```
w32tm /query /status
```

このコマンドはインスタンスと NTP サーバーを同期した最終時刻やポーリング間隔などの情報を返します。

NTP サーバーが Amazon Time Sync Service を使用するよう変更するには

1. コマンドプロンプトウィンドウで、次のコマンドを実行します。

```
w32tm /config /manualpeerlist:169.254.169.123 /syncfromflags:manual /update
```

2. 次のコマンドを使用して新しい設定を確認します。

```
w32tm /query /configuration
```

返される出力で、NtpServer が 169.254.169.123 IPv4 エンドポイントを表示することを確認します。

Amazon Windows AMI のデフォルトの NTP 設定

Amazon マシンイメージ (AMI) は一般的に、EC2 インフラストラクチャで機能させるために変更が必要な場合を除き、初期状態のデフォルトに準拠しています。以下の設定は仮想化環境で適切に動作するとともに、クロック同期ずれを 1 秒以内の精度に保持するように定められています。

- 更新間隔 — タイムサービスがシステム時刻を正しくなるように調整する頻度を管理します。AWS は更新間隔を 2 分に 1 回になるように設定します。
- NTP サーバー — 2018 年 8 月のリリースから、AMI はデフォルトで Amazon Time Sync Service を使用することになりました。このタイムサービスは169.254.169.123 IPv4 エンドポイントにあるすべての AWS リージョン からアクセス可能です。さらに 0x9 フラグはタイム

サービスがクライアントとして機能しており、設定されたタイムサーバーを確認する頻度を `SpecialPollInterval` を使用して決定することを示しています。

- タイプ – 「NTP」とはサービスがドメインの一部としてではなく、スタンドアロン NTP クライアントとして機能することを意味します。
- Enabled および InputProvider – タイムサービスが有効で、オペレーティングシステムに時刻が提供されます。
- 特別なポーリング間隔 – 設定された NTP サーバーを 900 秒 (15 分) ごとに確認します。

> **Note**
>
> Windows Server 2025 AMIの場合、`SpecialPollInterval` 値は 900 秒ではなく 1024 秒になります。

レジストリパス	キー名	データ
HKLM:\System\CurrentControlSet\services\w32time\Config	UpdateInterval	120
HKLM:\System\CurrentControlSet\services\w32time\Parameters	NtpServer	169.254.169.123,0x9
HKLM:\System\CurrentControlSet\services\w32time\Parameters	タイプ	NTP
HKLM:\System\CurrentControlSet\services\w32time\TimeProviders\NtpClient	有効	1
HKLM:\System\CurrentControlSet\services\w32time\TimeProviders\NtpClient	InputProvider	1

レジストリパス	キー名	データ
HKLM:\System\CurrentControlSet\services\w32time\TimeProviders\NtpClient	SpecialPollInterval	900 (Windows Server 2016、2019、および 2022) または 1024 (Windows Server 2025)

Amazon Time Sync Service の IPv6 エンドポイントにアクセスする

このセクションではIPv6 エンドポイントを通じてローカル Amazon Time Sync Service を使用するようにインスタンスを設定する場合、Amazon Time Sync Service の IPv4 エンドポイントにアクセスする で説明した手順と異なる点について説明します。Amazon Time Sync Service の設定プロセス全体について説明しているわけではありません。

IPv6 エンドポイントはNitro ベースのインスタンスでのみアクセス可能です。

IPv4 と IPv6 の両方のエンドポイントエントリを同時に使用することはお勧めしません。IPv4 および IPv6 NTP パケットはインスタンスの同じローカルサーバーから取得されます。IPv4 と IPv6 の両方のエンドポイントを設定する必要はなく、そうしてもインスタンスの時刻の精度は向上しません。

Linux

使用している Linux ディストリビューションに応じて、`chrony.conf` ファイルを編集するステップに到達すると、IPv4 エンドポイント (`169.254.169.123`) ではなく、Amazon Time Sync Service の IPv6 エンドポイント (`fd00:ec2::123`) を使用することになります。

```
server fd00:ec2::123 prefer iburst minpoll 4 maxpoll 4
```

ファイルを保存してchrony が `fd00:ec2::123` IPv6 エンドポイントを使用して時刻を同期させていることを確認します。

```
[ec2-user ~]$ chronyc sources -v
```

出力で、`fd00:ec2::123` IPv6 エンドポイントが表示されているのを確認したら、設定は完了しています。

Windows

Amazon Time Sync Service を使用するように NTP サーバーを変更するステップに到達すると、IPv4 エンドポイント (169.254.169.123) ではなく、Amazon Time Sync Service の IPv6 エンドポイント (fd00:ec2::123) を使用することになります。

```
w32tm /config /manualpeerlist:fd00:ec2::123 /syncfromflags:manual /update
```

新しい設定で fd00:ec2::123 IPv6 エンドポイントが使用されて時刻が同期されていることを確認します。

```
w32tm /query /configuration
```

出力で、NtpServer に fd00:ec2::123 IPv6 エンドポイントが表示されることを確認します。

拡張 Amazon Time Sync Service にアクセスする

拡張 Amazon Time Sync Service は、サポートされている Amazon EC2 インスタンスに対して、より正確なローカルタイムソースを提供します。precision-time 戦略を使用してプレイスメントグループで起動されたインスタンスは、これらのローカルソースにアクセスできます。リンクローカル NTP ソースまたは Linux インスタンス上の PTP ハードウェアクロック (PHC) デバイスからのより正確な時間を必要とするアプリケーションには、高精度タイムプレイスメントグループの使用をお勧めします。インスタンスを高精度タイムプレイスメントグループ内で起動すると、AWS により、AWS インフラストラクチャ内の高精度タイムソースに直接アクセス可能なサポートされているハードウェア上にインスタンスが配置されます。

主な利点

- デフォルトでの NTP ソースの強化 – 前のセクションの説明に従い NTP リンクローカル IP アドレスを使用すると、インスタンスは拡張ローカル NTP タイムソースにすぐにアクセスできます。
- マイクロ秒精度クロック同期 – Amazon EC2 Linux インスタンスの設定により、PTP ハードウェアクロックデバイスを使用してマイクロ秒精度クロック同期を達成できます。
- デプロイの簡素化 – 単一プレイスメント戦略により、すべてのインスタンスで正確な時間機能を保証できます。
- ハードウェアパケットタイムスタンプ – ネットワーク測定値の低レベルパケットタイムスタンプにアクセスできます。

- 追加コストなし – 高精度タイムプレイスメントグループは追加料金なしで利用できます。

ルールと制限

- 高精度タイムプレイスメントグループは、すべての AWS 商用リージョンで利用できます。
- 高精度タイムプレイスメントグループは、Gen7 以降の以下の Amazon EC2 インスタンスファミリーをサポートしています。
  - 汎用: M7a、M7g、M7g-flex、M7gd、M7i、M7i-flex、M8a、M8g、M8g-flex
  - コンピューティング最適化: C7a、C7gd、C7i、C7i-flex、C8g、C8g-flex、C8gd
  - メモリ最適化:
    R7a、R7g、R7i、R7id、R8g、X8adez、X8adz-3tb、X8adz-6tb、X8adzs、X8aedez、X8aedz-3tb、X8
  - ストレージ最適化: I8g、I8ge
- 高精度タイムプレイスメントグループでインスタンスを開始または起動した時に、拡張 Amazon Time Sync Service へのアクセスを許可するハードウェアが不十分な場合、リクエストは失敗します。
- 高精度タイムプレイスメントグループ内のインスタンスをいったん停止して再起動しても、そのインスタンスは同じプレイスメントグループ内で実行されます。ただし、拡張 Amazon Time Sync Service へのアクセスを許可するハードウェアが不十分な場合、開始が失敗することがあります。
- Amazon EC2 は、拡張 Amazon Time Sync Service をサポートするハードウェアを継続的に追加しています。容量不足が原因でリクエストが失敗した場合は、後で再試行するか、別のアベイラビリティーゾーンを試してください。詳細については、「Amazon EC2 インスタンスの起動に関する問題のトラブルシューティング」を参照してください。
- プレイスメントグループのルールと制限が適用されます。詳細については、Amazon EC2 インスタンスのプレイスメントグループを参照してください。

高精度タイムプレイスメントグループを作成する

AWS CLI、AWS マネジメントコンソール、または AWS SDK を使用して `precision-time` 戦略を指定することで、高精度タイムプレイスメントグループを作成できます。

AWS CLI を使用する

次のコマンドを実行します。

```
aws ec2 create-placement-group \
    --group-name my-precision-time-pg \
```

```
    --strategy precision-time
```

コマンドは、キャパシティ予約の作成時に使用するプレイスメントグループ ARN と、インスタンスの起動時およびプレイスメントグループのリンク時に使用するグループ名またはグループ ID を返します。

インスタンスの起動

高精度タイムプレイスメントグループを作成したら、インスタンスを起動するときにそのグループを指定して、拡張 Amazon Time Sync Service にアクセスします。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type r7g.2xlarge \
    --placement GroupId=pg-0aaa1111111111111
```

拡張 Amazon Time Sync Service へのアクセスを確認する

高精度タイムプレイスメントグループでインスタンスを起動すると、拡張 NTP タイムソースのメリットを享受できます。

例えば、インスタンスで chronyd デーモンを使用すると NTP タイムソースが Stratum 1 と表示され、クロック精度メトリクスが改善されたことを確認できます。

```
[ec2-user ~]$ chronyc sources
```

```
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
^* 169.254.169.123               1   4   377     3  +3477ns[+4689ns] +/-   91us
```

chrony で報告された時刻同期メトリクスを確認します。

```
[ec2-user ~]$ chronyc tracking
```

```
Reference ID    : A9FEA97B (169.254.169.123)
Stratum         : 2
Ref time (UTC)  : Wed May 06 01:33:43 2026
System time     : 0.000000276 seconds fast of NTP time
Last offset     : +0.000000331 seconds
RMS offset      : 0.000001929 seconds
```

```
Frequency       : 2.870 ppm fast
Residual freq   : +0.000 ppm
Skew            : 0.031 ppm
Root delay      : 0.000107584 seconds
Root dispersion : 0.000036476 seconds
Update interval : 16.2 seconds
Leap status     : Normal
```

PTP ハードウェアクロック (PHC) にアクセスする

PTP ハードウェアクロック (PHC) は AWS Nitro System の一部です。高精度タイムプレイスメントグループで起動された、サポートされているベアメタルおよび仮想化された Amazon EC2 インスタンスから直接アクセスできます。PHC デバイスには、現在 Linux インスタンスでのみアクセスできます。以下のセクションでは、Linux インスタンスで PHC デバイスを設定および確認する方法について説明します。

要件

- ENA ドライバーバージョン 2.10.0 以降がサポート対象のオペレーティングシステムにインストールされている。
- Linux を実行する Amazon EC2 インスタンス。サポート対象のオペレーティングシステムの詳細についてはGitHub でドライバーの「前提条件」を参照してください。
- 高精度タイムプレイスメントグループで起動された Amazon EC2 インスタンス (「拡張 Amazon Time Sync Service にアクセスする」を参照)。

> **Note**
>
> 拡張 Amazon Time Sync Service と PHC デバイスは、次のリージョンおよび特定のインスタンスファミリーでは、高精度タイムプレイスメントグループなしでアクセス可能です。ただし、最良のエクスペリエンスを得るには、高精度プレイスメントグループで Amazon EC2 インスタンスを起動することをお勧めします。
>
> - レガシーアクセスでサポートされる AWS リージョン: 米国東部 (バージニア北部)、米国東部 (オハイオ)、アジアパシフィック (マレーシア)、アジアパシフィック (タイ)、アジアパシフィック (東京)、欧州 (ストックホルム)
> - レガシーアクセスでサポートされるローカルゾーン: 米国東部 (ニューヨークシティ)
> - レガシーアクセスでサポートされるインスタンスファミリー:
>   - 汎用: M7a、M7g、M7i

- メモリ最適化: R7a、R7g、R7i
- ストレージ最適化: I8g、I8ge

PHC サポート付きで ENA ドライバーをコンパイルして有効にする

お使いのオペレーティングシステム固有の情報については、GitHub の「ENA ドライバードキュメント」内の最新 Elastic Network Adapter (ENA) ドライバーバージョン向け最新手順を必ず確認してください。以下が Amazon Linux 向け手順の概要になります。

開始する前に、インスタンスが 要件 を満たしていることを確認してください。

PHC サポート付きで ENA ドライバーをコンパイルして有効にするには

1. インストール条件

```
[ec2-user ~]$ sudo yum update
[ec2-user ~]$ sudo yum install kernel-devel-$(uname -r) git
[ec2-user ~]$ sudo reboot
```

2. 必要な ptp モジュールをロードします。

```
[ec2-user ~]$ sudo modprobe ptp
```

3. 最新の ENA ドライバーバージョン (バージョン 2.10.0 以降) を取得します。

```
[ec2-user ~]$ git clone https://github.com/amzn/amzn-drivers.git /tmp/amzn-drivers
```

4. PHC サポート付きで ENA ドライバー ena.ko を構築します。

```
[ec2-user ~]$ cd /tmp/amzn-drivers/kernel/linux/ena
[ec2-user ~]$ ENA_PHC_INCLUDE=1 make
```

5. ENA ドライバーを再ロードし、PHC デバイスを有効にします。

```
[ec2-user ~]$ sudo rmmod ena && sudo insmod ena.ko phc_enable=1
```

6. ロードされた ENA ドライバーが PHC をサポートしていることを確認します。

```
[ec2-user ~]$ modinfo ena | grep -E "phc_enable"
```

```
parm:           phc_enable:Enable PHC.
```

7. ENA ドライバーで PHC サポートが有効になっていることを確認します。

```
[ec2-user ~]$ cat /sys/module/ena/parameters/phc_enable
```

```
1
```

ドライバーをインストールし、再起動時に phc_enable オプションを有効にする手順については、「GitHub」の「ENA ドライバーに関する README」を参照してください。

PTP デバイス設定を確認する

ENA PTP ハードウェアクロックデバイスがインスタンスに表示されていることを確認します。

```
[ec2-user ~]$ for file in /sys/class/ptp/*; do echo -n "$file: "; cat "$file/
clock_name"; done
```

正常な出力

```
/sys/class/ptp/ptp<index>: ena-ptp-<PCI slot>
```

説明は以下のとおりです。

- *index* は、カーネルに登録されている PTP ハードウェアクロックインデックスです。
- *PCI slot* は ENA イーサネットコントローラー PCI スロットです。これは、「`lspci | grep ENA`」に示されているのと同じスロットです。

出力の例

```
/sys/class/ptp/ptp0: ena-ptp-05
```

出力に ena-ptp-*<PCI slot>* が表示されない場合はENA ドライバーが正しくインストールされていません。PHC サポート付きで ENA ドライバーをコンパイルして有効にする のステップを確認します。

PTP シンボリックリンクを設定する

PTP デバイスの名前は通常 /dev/ptp0 や /dev/ptp1 などで、ハードウェアの初期化順序に応じてインデックスが付けられます。シンボリックリンクを作成することで、chrony などのアプリケーションは、インデックスの変更に関係なく、常に正しいデバイスを参照できます。

最新の Amazon Linux 2023 AMI には、ENA ホストに関連付けられた正しい /dev/ptp エントリを指す /dev/ptp_ena シンボリックリンクを作成する udev ルールが含まれています。

まず、次のコマンドを実行して、シンボリックリンクが存在するかどうかをチェックします。

```
[ec2-user ~]$ ls -l /dev/ptp*
```

出力の例

```
crw------- 1 root root 245, 0 Jan 31 2025 /dev/ptp0
lrwxrwxrwx 1 root root      4 Jan 31 2025 /dev/ptp_ena -> ptp0
```

説明は以下のとおりです。

- /dev/ptp*<index>* は PTP デバイスへのパスです。
- /dev/ptp_ena は、同じ PTP デバイスを指す定数シンボリックリンクです。

/dev/ptp_ena シンボリックリンクが存在する場合は、「PHC デバイスを使用するように chronyd デーモンを設定する」に進みます。シンボリックリンクが欠落している場合は、以下を実行します。

PTP シンボリックリンクを作成するには

1. udev ルールを追加します。

   ```
   [ec2-user ~]$ echo "SUBSYSTEM==\"ptp\", ATTR{clock_name}==\"ena-ptp-*\", SYMLINK +=
    \"ptp_ena\"" | sudo tee -a /etc/udev/rules.d/53-ec2-network-interfaces.rules
   ```

2. インスタンスを再起動するか、次のコマンドを実行して、udev ルールを再ロードします。

   ```
   [ec2-user ~]$ sudo udevadm control --reload-rules && sudo udevadm trigger
   ```

PHC デバイスを使用するように chronyd デーモンを設定する

chronyd クロック同期デーモンは、PHC デバイスを、/dev/ptp_ena シンボリックリンクを使用してデバイスを識別する追加のタイムソースとして使用するように設定する必要があります。

PHC デバイスを使用するように chronyd デーモンを設定するには

1. テキストエディタを使用して /etc/chrony.conf を編集し、次の行を追加します。

```
refclock PHC /dev/ptp_ena poll 0 delay 0.000010 prefer
```

2. chrony を再起動します。

```
[ec2-user ~]$ sudo systemctl restart chronyd
```

3. chrony が PTP ハードウェアクロックを使用していることを確認します。PHC0 デバイスは、Stratum 0 値を持つ優先タイムソースである必要があります。ローカル NTP タイムソース (設定されている場合) には、Stratum 1 値が必要です。

```
[ec2-user ~]$ chronyc sources
```

```
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
#* PHC0                          0   0   377     0   +184ns[ +198ns] +/- 5032ns
^- 169.254.169.123             1   4   377     8    -18us[  -18us] +/-  115us
```

ハードウェアパケットタイムスタンプを確認する

PHC サポート付きでロードされた ENA ドライバーは、ハードウェアパケットタイムスタンプへのアクセスを提供します。ethtool -T *interface* コマンドを使用して、この機能のサポートを確認できます。

ハードウェアパケットタイムスタンプの使用の詳細については、「Linux のパケットタイムスタンプに関するドキュメント」を参照してください。

```
[ec2-user ~]$ sudo ethtool -T ens5
```

```
Time stamping parameters for ens5:
Capabilities:
```

```
        software-transmit
        hardware-receive
        software-receive
        software-system-clock
PTP Hardware Clock: 0
Hardware Transmit Timestamp Modes: none
Hardware Receive Filter Modes:
        none
        all
```

出力が機能リストの hardware-receive に表示されている場合、ハードウェアパケットタイムスタンプはお使いのインスタンスで使用できます。

EC2 インスタンスまたはインターネットに接続されたデバイスのタイムリファレンスを、パブリック Amazon Time Sync Service を使用するように設定する

インスタンス、またはローカルコンピュータやオンプレミスサーバーなどのインターネットに接続されたデバイスを、インターネット上の time.aws.com でアクセスできるパブリック Amazon Time Sync Service を使用するように設定できます。ローカル Amazon Time Sync Service のバックアップとして、そして AWS 外部のリソースを Amazon Time Sync Service に接続するために、パブリック Amazon Time Sync Service を使用できます。

> **Note**
>
> 最高のパフォーマンスを得るにはインスタンスでローカル Amazon Time Sync Service を使用し、パブリック Amazon Time Sync Service はバックアップとしてのみ使用することをお勧めします。

インスタンスまたはデバイスのオペレーティングシステムの説明に従ってください。

Linux

chrony または ntpd を使用してパブリック Amazon Time Sync Service を使用するように Linux インスタンスまたはデバイスを設定するには

1. 次のように、テキストエディタを使用して /etc/chrony.conf (chrony を使用する場合) または /etc/ntp.conf (ntpd を使用する場合) を編集します。

a. インスタンスまたはデバイスが、Leap Smearing を行うサーバーと行わないサーバーを混在させようとしないように、ローカル Amazon Time Sync Service への既存の接続以外の `server` で始まる行を削除またはコメントアウトします。

> **Important**
>
> パブリック Amazon Time Sync Service に接続するように EC2 インスタンスを設定する場合は次の行でローカル Amazon Time Sync Service に接続するようにインスタンスを設定しているので、この行を削除しないでください。ローカル Amazon Time Sync Service の方が、直接的な接続でクロックが正確です。パブリック Amazon Time Sync Service はバックアップとしてのみ使用してください。
>
> ```
> server 169.254.169.123 prefer iburst minpoll 4 maxpoll 4
> ```

b. パブリック Amazon Time Sync Service に接続するには次の行を追加します。

```
pool time.aws.com iburst
```

2. 以下のコマンドのいずれかを使用してデーモンを再起動します。

- chrony

```
sudo service chronyd force-reload
```

- ntpd

```
sudo service ntp reload
```

macOS

パブリック Amazon Time Sync Service を使用するように macOS インスタンスまたはデバイスを設定するには

1. [システム環境設定] を開きます。
2. [Date & Time] (日付と時刻) を選択し、[Date & Time] (日付と時刻) タブを選択してください。
3. 変更するにはロックアイコンを選択し、プロンプトが表示されたらパスワードを入力してください。

4. [Set date and time automatically] (日付と時刻を自動的に設定) に、**time.aws.com** と入力してください。

Windows

パブリック Amazon Time Sync Service を使用するように Windows インスタンスまたはデバイスを設定するには

1. [Control Panel] (コントロールパネル を開いてください。
2. [Date and Time] (日付と時刻) アイコンを選択してください。
3. [Internet Time] (インターネット時刻) タブを選択してください。お使いの PC がドメインの一部である場合、このタブは使用できません。その場合はドメインコントローラと時刻が同期されます。Amazon Time Sync Service を使用するようにコントローラを設定できます。
4. [Change settings] (設定を変更) を選択してください。
5. [インターネットタイムサーバーと同期] のチェックボックスをオンにします。
6. [Server] (サーバー) の横に、**time.aws.com** と入力してください。

パブリック Amazon Time Sync Service を使用するように Windows Server インスタンスまたはデバイスを設定するには

- Microsoft の手順に従ってレジストリを更新してください。

Linux インスタンスのタイムスタンプを比較する

Amazon Time Sync Service を使用している場合はAmazon EC2 Linux インスタンスのタイムスタンプと ClockBound を比較して、イベントの実際の時刻を判断できます。ClockBound は EC2 インスタンスのクロック精度を測定し、インスタンスの現在のクロックに関して、特定のタイムスタンプが過去または将来にあるかどうかを確認できます。この情報は各インスタンスの地理的位置に関係なく、EC2 インスタンス間のイベントとトランザクションの順序と一貫性を判断するのに役立ちます。

ClockBound はオープンソースのデーモンとライブラリです。ClockBound の詳細についてはGitHub のClockBoundを参照してください。

ClockBound は Linux インスタンスでのみサポートされています。

PTP ハードウェアクロックへの直接 PTP 接続を使用している場合、chrony などのタイムデーモンはクロック誤差範囲を過小評価します。これはPTP ハードウェアクロックが NTP と異なり誤差範囲の正しい情報を chrony に渡さないためです。その結果、クロック同期デーモンはクロックが UTC に対して正確であると想定しているため、0 という誤差範囲になります。誤差範囲全体を測定するために、Nitro System は PTP ハードウェアクロックの誤差範囲を計算し、ENA ドライバー sysfs ファイルシステムを介して EC2 インスタンスで使用できるようにします。これをナノ秒単位の値として直接読み取ることができます。

PTP ハードウェアクロックエラーバウンドを取得するには

1. まず、以下のコマンドのいずれかを使用して PTP ハードウェアクロックデバイスの正しい位置を取得します。コマンドのパスはインスタンスの起動に使用される AMI によって異なります。

   - 複数 Amazon Linux 2:

     ```
     cat /sys/class/net/eth0/device/uevent | grep PCI_SLOT_NAME
     ```

   - 複数 Amazon Linux 2023:

     ```
     cat /sys/class/net/ens5/device/uevent | grep PCI_SLOT_NAME
     ```

   出力は PTP ハードウェアクロックデバイスの場所である PCI スロット名です。この例では場所は 0000:00:03.0 です。

   ```
   PCI_SLOT_NAME=0000:00:03.0
   ```

2. PTP ハードウェアクロックのエラー範囲を取得するには次のコマンドを実行します。前の手順で取得した PCI スロット名を指定します。

   ```
   cat /sys/bus/pci/devices/0000:00:03.0/phc_error_bound
   ```

   出力は PTP ハードウェアクロックのクロック誤差範囲 (ナノ秒単位) です。

PTP ハードウェアクロックへの直接 PTP 接続を使用するときに、特定の時点における正しいクロック誤差範囲を計算するにはPTP ハードウェアクロックをポーリングする時点の「範囲からのクロックエラーバウンド」chrony または「ClockBound」を追加する必要があります。chronyクロック精度

の測定とモニタリングの詳細については「Amazon Time Sync Service と Amazon CloudWatch を使用して Amazon EC2 インスタンスのクロック精度を管理する — パート 1」を参照してください。

インスタンスのタイムゾーンを変更する

Amazon EC2 インスタンスはデフォルトで UTC (協定世界時) タイムゾーンに設定されています。インスタンスの時刻をローカルのタイムゾーンまたはネットワーク内の別のタイムゾーンに変更できます。

インスタンスのオペレーティングシステムの説明を使用してください。

Linux

> **Important**
>
> この情報はAmazon Linux に適用されます。その他のディストリビューションの情報については各ドキュメントを参照してください。

Amazon Linux での タイムゾーンを変更するには

1. システムの現在のタイムゾーン設定を表示します。

```
[ec2-user ~]$ timedatectl
```

2. 使用可能なタイムゾーンを一覧表示します。

```
[ec2-user ~]$ timedatectl list-timezones
```

3. 選択したタイムゾーンを設定します。

```
[ec2-user ~]$ sudo timedatectl set-timezone America/Vancouver
```

4. (オプション) timedatectl コマンドをもう一度実行して、現在のタイムゾーンが新しいタイムゾーンに更新されていることを確認します。

```
[ec2-user ~]$ timedatectl
```

Windows

Windows インスタンスの時間帯を変更するには

1. インスタンスで、コマンドプロンプトウィンドウを開きます。
2. インスタンスで使用する時間帯を特定します。タイムゾーンの一覧を取得するには次のコマンドを使用します。

```
tzutil /l
```

このコマンドは利用可能なすべてのタイムゾーンのリストを次の形式で返します。

```
display name
time zone ID
```

3. インスタンスに割り当てるタイムゾーン ID を見つけます。
4. 例: UTC タイムゾーンの割り当て:

```
tzutil /s "UTC"
```

例: 太平洋標準時の割り当て:

```
tzutil /s "Pacific Standard Time"
```

Windows インスタンスのタイムゾーンを変更するときはシステムを再起動してもそのタイムゾーンが維持されるようにする必要があります。そうでない場合、インスタンスを再起動すると、再び UTC 時間が使用されます。RealTimeIsUniversal レジストリキーを追加することでタイムゾーン設定を維持できます。このキーはすべての現世代のインスタンスでデフォルトで設定されます。RealTimeIsUniversal レジストリキーが設定されているかどうかを確認するには以下の手順のステップ 3 を参照してください。キーが設定されていない場合は以下の手順を最初から実行します。

RealTimeIsUniversal レジストリキーを設定するには

1. インスタンスで、コマンドプロンプトウィンドウを開きます。
2. 次のコマンドを使用してレジストリキーを追加します。

```
reg add "HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\TimeZoneInformation" /
v RealTimeIsUniversal /d 1 /t REG_DWORD /f
```

3. (オプション) 次のコマンドを使用して、インスタンスでキーが正常に保存されたことを確認します。

```
reg query "HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control
\TimeZoneInformation" /s
```

このコマンドは TimeZoneInformation レジストリキーのサブキーを返します。ジョブが実行されると、次のような RealTimeIsUniversal キーがリストの一番下に表示されます。

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\TimeZoneInformation
    Bias                            REG_DWORD     0x1e0
    DaylightBias                    REG_DWORD     0xffffffc4
    DaylightName                    REG_SZ        @tzres.dll,-211
    DaylightStart                   REG_BINARY    00000300020002000000000000000000
    StandardBias                    REG_DWORD     0x0
    StandardName                    REG_SZ        @tzres.dll,-212
    StandardStart                   REG_BINARY    00000B0001000200000000000000000
    TimeZoneKeyName                 REG_SZ        Pacific Standard Time
    DynamicDaylightTimeDisabled     REG_DWORD     0x0
    ActiveTimeBias                  REG_DWORD     0x1a4
    RealTimeIsUniversal             REG_DWORD     0x1
```

EC2 Capacity Manager

Amazon EC2 Capacity Manager を使用すると、オンデマンドインスタンス、スポットインスタンス、キャパシティ予約のキャパシティ使用状況を 1 か所から監視、分析、管理できます。Capacity Manager は、1 時間ごとの更新と最適化機能で使用状況データを集計する統合ダッシュボードを通じて、キャパシティの追跡を簡素化します。

大規模なエンタープライズ企業やマルチアカウント組織向けに設計された Capacity Manager により、複数のソースから手動でデータを収集する必要がなくなります。このツールは、AWS リージョン全体の深いインサイトを提供します。詳しいキャパシティメトリクスをドリルダウンし、クラウドリソースを最適化するためのアクションを即座に実行できます。

AWS Organizations 統合を有効にすると、組織内にあるすべてのメンバーアカウント全体のキャパシティデータを単一のダッシュボードから表示して分析できます。Organizations 統合を有効にしない場合、Capacity Manager が監視するのは有効化されている個々の AWS アカウント内のリソースのみになります。

> **Note**
>
> - Capacity Manager を有効にできるのは、アカウントあたり 1 つの AWS リージョンのみです。
> - Capacity Manager に表示されるすべてのコストは、公開されているオンデマンド料金に基づいた推定コストです。Savings Plans やリザーブドインスタンスなどの割引は見積もりに含まれていません。実際のコストは、これらの見積もりとは異なる場合があります。

主な機能

- ダッシュボード – すべてのオンデマンドキャパシティ予約、オンデマンド、およびスポットの使用量のおおまかな概要を提供するもので、キャパシティ体制の改善に役立つ主要メトリクスが含まれています。
- クロスアカウント集約とクロスリージョン集約 – 組織内のすべてのメンバーアカウントのキャパシティ使用状況を単一のダッシュボードで表示します。Capacity Manager は、タグデータを含むパーティション内のすべてのリージョンのデータを、Capacity Manager を有効にした単一のリージョンに集約します。クロスアカウントデータでは、Capacity Manager はアカウント名をディメンションとして提供します。
- タグベースのグループ化とフィルタリング — Amazon EC2 リソース (`environment` や `team` など) からタグキーをアクティブ化して、キャパシティメトリクスをグループ化およびフィルタリングするときにディメンションとして使用します。EC2 Auto Scaling グループ、EKS クラスター名、EKS Kubernetes ノードプール、および Karpenter ノードプールの Capacity Manager 提供のタグは、デフォルトで含まれています。詳細については、「モニタリングされているタグキーの管理」を参照してください。
- データエクスポート — キャパシティデータを CSV 形式または Parquet 形式で Amazon S3 にエクスポートして、詳細な分析とカスタムレポートを行います。
- API — `GetCapacityManagerMetricData` と `GetCapacityManagerMetricDimensions` を使用して、キャパシティメトリクスをプログラムでクエリします。
- 日付セレクタ — 1 時間から 90 日までの日付範囲にわたってキャパシティデータを分析します。

- 30 を超えるメトリクス — 30 を超えるメトリクスを使用して、キャパシティ予約、オンデマンドインスタンス、およびスポットインスタンス全体でキャパシティを追跡します。

EC2 Capacity Manager の有効化

Capacity Manager は、要件に応じて 2 つの異なるレベルで有効にできます。単一の AWS アカウントのために有効にして、そのアカウント内のリソースのみを監視する、または、組織内のすべてのメンバーアカウントの一元的な可視化のために AWS Organizations と統合することができます。

- 組織レベル: 単一のダッシュボードから組織内にあるすべてのメンバーアカウントのキャパシティデータを表示し、分析します。
- アカウントレベル: 単一の AWS アカウント内のキャパシティ使用状況を監視します。

トピック

- AWS Organizations での EC2 Capacity Manager の有効化
- アカウントレベルでの EC2 Capacity Manager の有効化
- 委任管理者の登録
- EC2 Capacity Manager 用のサービスリンクロールの使用

AWS Organizations での EC2 Capacity Manager の有効化

AWS Organizations で EC2 Capacity Manager を有効にして、すべてのメンバーアカウントのキャパシティを組織レベルで可視化して管理することができます。この統合により、キャパシティ使用状況を一元的な場所から監視、分析、管理できます。

組織レベルのアクセスを有効化し、組織全体のキャパシティを管理する責任は、管理アカウントが担います。

AWS Organizations で Capacity Manager を有効化することで、以下のメリットが得られます。

- 一元的なキャパシティの可視性 – クロスアカウント集約とクロスリージョン集約を使用して、組織内のすべてのメンバーアカウントのキャパシティ使用状況を単一のダッシュボードで表示します。
- 組織全体での最適化 – 組織内のすべてのアカウントで未使用のキャパシティ予約と最適化機会を特定します。

- 委任管理者 – 適切なアクセスコントロールを維持しながら、特定のメンバーアカウントが組織の Capacity Manager の管理をできるようにします。

AWS Organizations との統合を有効にしない場合は、Capacity Manager を有効にした個々の AWS アカウント内のリソースのみを監視できます。

前提条件

- 1 つの管理アカウントと、1 つ以上のメンバーアカウントで設定された AWS Organizations が必要です。アカウントタイプの詳細については、AWS ユーザーガイドの用語とコンセプトを参照してください。
- 管理アカウントには、以下の IAM アクションに対する許可が必要です。
  - `organizations:EnableAwsServiceAccess`
  - `organizations:RegisterDelegatedAdministrator` (委任管理を使用している場合)
  - `iam:CreateServiceLinkedRole`
- AWS Organization にアクセスを許可するには、AWSEC2CapacityManagerServiceRolePolicy ユースケースを使用してサービスリンクロールを作成する必要があります。詳細については、「Creating a service-linked role for Capacity Manager」を参照してください。

AWS Organizations での Capacity Manager の有効化

管理アカウントを使用して、Capacity Manager で組織アクセスを有効にします。

Console

Capacity Manager で組織アクセスを有効にする

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Capacity Manager] を選択します。
3. [設定] タブを選択します。
4. [信頼できるアクセス] セクションで、[信頼できるアクセスを管理] を選択します。
5. 表示されたプロンプトで、[信頼できるアクセスを有効にする] を選択します。次に、[保存] を選択します。

AWS CLI

Capacity Manager で組織のアクセスを有効にする

1. サービスにリンクされたロールの作成

```
aws iam create-service-linked-role --aws-service-name
 ec2.capacitymanager.amazonaws.com
```

2. AWS 組織のアクセスを有効にする

```
aws organizations enable-aws-service-access --service-principal
 ec2.capacitymanager.amazonaws.com
```

3. AWS Organization で Capacity Manager を有効にする

```
aws ec2 enable-capacity-manager --organizations-access
```

既存の Capacity Manager の組織アクセスを更新するには、次のコマンドを実行します。

```
aws ec2 update-capacity-manager-organizations-access --organizations-access
```

PowerShell

Capacity Manager で組織のアクセスを有効にする

1. New-IAMServiceLinkedRole コマンドレットを使用して、サービスリンクロールを作成します。

```
New-IAMServiceLinkedRole -AWSServiceName "ec2.capacitymanager.amazonaws.com"
```

2. Enable-ORGAWSServiceAccess コマンドレットを使用して、AWS 組織アクセスを有効にします。

```
Enable-ORGAWSServiceAccess -ServicePrincipal "ec2.capacitymanager.amazonaws.com"
```

3. Enable-EC2CapacityManager コマンドレットを使用して、AWS 組織で Capacity Manager を有効にします。

```
Enable-EC2CapacityManager -OrganizationsAccess $true
```

既存の Capacity Manager の組織アクセスを更新するには、次の Update-EC2CapacityManagerOrganizationsAccess コマンドレットを実行します。

```
Update-EC2CapacityManagerOrganizationsAccess -OrganizationsAccess $true
```

組織で Capacity Manager が有効化されていることの確認

Console

組織で Capacity Manager が有効化されていることを確認する

1. Capacity Manager コンソールで、[設定] を選択します。
2. [信頼できるアクセス] セクションで、[組織のアクセス] が [有効] として表示されていることを確認します。
3. [組織 ID] に組織の ID が表示されていることを確認します。

AWS CLI

組織で Capacity Manager が有効化されていることを確認する

次のコマンドを実行します。

```
aws ec2 get-capacity-manager-attributes
```

出力には以下が表示されます。

```
{
```

```
    "CapacityManagerStatus": "enabled",
    "OrganizationsAccess": true,
    "IngestionStatus": "initial-ingestion-pending",
    "IngestionStatusMessage": "Capacity Manager is collecting historical data from
 2025-10-01T00:00:00Z. Data collection is in progress and may take several hours to
 complete."
}
```

PowerShell

組織で Capacity Manager が有効化されていることを確認する

Get-EC2CapacityManagerAttribute コマンドレットを使用します。

```
Get-EC2CapacityManagerAttribute
```

出力には以下が表示されます。

```
CapacityManagerStatus      : enabled
DataExportCount            : 0
EarliestDatapointTimestamp :
IngestionStatus            : initial-ingestion-in-progress
IngestionStatusMessage     : Capacity Manager is collecting historical data from
                             2026-03-17T16:00:00Z. Data collection is in progress
 and may take
                             several hours to complete.
LatestDatapointTimestamp   :
OrganizationsAccess        : True
```

考慮事項

- サービスリンクロールの作成: コンソールを使用して組織のアクセスを有効にすると、Capacity Manager が自動的にすべてのメンバーアカウントで AWSServiceRoleForEC2CapacityManager サービスリンクロールを作成します。AWS CLI を使用して有効にする場合は、`createServiceLinkedRole` を手動で呼び出す必要があります。
- データ集約: 組織のアクセスを有効にすると、Capacity Manager がすべてのメンバーアカウントから 14 日間の履歴データを埋め戻します。通常、このプロセスの完了には数分かかります。
- リージョンの制限: Capacity Manager を有効にできるのは組織あたり 1 リージョンのみですが、すべての商用リージョンのデータが集約されます。

- 許可: メンバーアカウントが実行する必要のあるアクションはありません。Capacity Manager がサービスリンクロールを使用して、すべてのアカウントのリソースを自動的に検出します。

アカウントレベルでの EC2 Capacity Manager の有効化

アカウントレベルで Capacity Manager を有効にして、単一の AWS アカウント内の EC2 キャパシティ使用状況を監視し、分析します。有効にすると、Capacity Manager がオンデマンドインスタンス、スポットインスタンス、キャパシティ予約に関するデータを収集して、最適化機会を特定し、使用パターンを追跡できるようにします。

アカウントレベルで Capacity Manager を有効にする

Console

アカウントの Capacity Manager を有効にする

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Capacity Manager] を選択します。
3. Capacity Manager ページで [リージョンで有効化] を選択します。

AWS CLI

アカウントの Capacity Manager を有効にする

次のコマンドを実行します。

```
aws ec2 enable-capacity-manager
```

PowerShell

アカウントの Capacity Manager を有効にする

Enable-EC2CapacityManager コマンドレットを使用します。

```
Enable-EC2CapacityManager
```

> **Note**
>
> - Capacity Manager を有効にすると、14 日間の履歴データが収集および集計されます。このプロセスは数時間かかる場合があります。
> - 履歴データの収集中は、`initial-ingestion-in-progress` 状態が表示されます。この収集期間中、履歴データにギャップが発生する可能性があります。データ収集が完了すると、 `ingestion-complete` 状態が表示されます。

委任管理者の登録

Capacity Manager 用の委任管理者を登録できます。そうすることで、メンバーアカウントが AWS 組織の Capacity Manager を管理できるようになります。組織内の委任管理者を登録または削除できるのは、管理アカウントのみです。

> **Note**
>
> 登録済みの委任管理者が存在する間は、組織の Capacity Manager を無効にすることができません。

トピック

- 前提条件
- 委任された管理者の登録
- 委任管理者を削除する

前提条件

管理アカウントの AWS Organizations で Capacity Manager が有効になっている必要があります。詳細については、AWS Organizations での EC2 Capacity Manager の有効化を参照してください。

委任された管理者の登録

委任管理者の登録には、Amazon EC2 コンソール、AWS CLI、または PowerShell を使用できます。

Console

委任管理者を登録する手順

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Capacity Manager] を選択します。
3. [設定] タブを選択します。
4. [委任管理者] セクションで、[追加] を選択します。
5. 表示されるプロンプトで、委任管理者として追加する組織メンバーのアカウント ID を入力します。
6. [Add delegated administrator (代理管理者の追加)] を選択します。

AWS CLI

委任管理者を登録する手順

次のコマンドを実行します。

```
aws organizations register-delegated-administrator \
    --account-id 123456789012 \
    --service-principal ec2.capacitymanager.amazonaws.com
```

PowerShell

委任管理者を登録する手順

Register-ORGDelegatedAdministrator コマンドレットを使用します。

```
Register-ORGDelegatedAdministrator `
    -AccountId "123456789012" `
    -ServicePrincipal "ec2.capacitymanager.amazonaws.com"
```

委任管理者を削除する

委任管理者は、Amazon EC2 コンソール、AWS CLI、または PowerShell を削除できます。

Console

委任管理者を削除する

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Capacity Manager] を選択します。
3. [設定] タブを選択します。
4. [委任管理者] セクションで [管理] を選択します。
5. 表示されるプロンプトで、[委任管理者を削除する] を選択します。

AWS CLI

委任管理者を削除する

次のコマンドを実行します。

```
aws organizations deregister-delegated-administrator \
    --account-id 123456789012 \
    --service-principal ec2.capacitymanager.amazonaws.com
```

PowerShell

委任管理者を削除する

Unregister-ORGDelegatedAdministrator コマンドレットを使用します。

```
Unregister-ORGDelegatedAdministrator `
    -AccountId "123456789012" `
    -ServicePrincipal "ec2.capacitymanager.amazonaws.com"
```

EC2 Capacity Manager 用のサービスリンクロールの使用

EC2 Capacity Manager は AWS Identity and Access Management (IAM) サービスリンクロールを使用します。サービスリンクロールは、Capacity Manager に直接リンクされる一意の IAM ロールタイプです。サービスリンクロールは Capacity Manager によって事前定義されており、当サービスがユーザーに代わって他の AWS サービスを呼び出すために必要な許可のすべてが含まれています。

サービスリンクロールを使用すると必要な許可を手動で追加する必要がなくなるため、Capacity Manager の設定が容易になります。Capacity Manager のサービスリンクロールの許可は Capacity Manager が定義し、そのロールを引き受けることができるのは Capacity Manager のみです (別途定義されている場合を除く)。定義される許可は信頼ポリシーと許可ポリシーに含まれており、その許可ポリシーを他の IAM エンティティにアタッチすることはできません。

サービスリンクロールを削除するには、最初に関連リソースを削除する必要があります。これにより、リソースにアクセスする許可が誤って削除されることがなくなるため、Capacity Manager のリソースが保護されます。

サービスリンクロールをサポートする他のサービスについては、「IAM と連携する AWS のサービス」を参照して、サービスリンクロール列にはいと表記されているサービスを確認してください。サービスリンクロールに関するドキュメントをサービスで表示するには、リンクで [はい] を選択します。

Capacity Manager 用のサービスリンクロール許可

Capacity Manager は、ユーザーがキャパシティリソースを管理することを可能にし、ユーザーに代わって AWS Organizations と統合するために、AWSServiceRoleForEC2CapacityManager という名前のサービスリンクロールを使用します。

AWSServiceRoleForEC2CapacityManager サービスリンクロールは、以下のサービスを信頼してロールを引き受けます。

- `ec2.capacitymanager.amazonaws.com`

AWSEC2CapacityManagerServiceRolePolicy という名前のロール許可ポリシーは、Capacity Manager が以下のアクションを完了できるようにします。

- `organizations:DescribeOrganization`
- `organizations:ListAccounts`
- `organizations:ListChildren`
- `organizations:ListAWSServiceAccessForOrganization`
- `organizations:ListDelegatedAdministrators`

ユーザー、グループ、またはロールにサービスリンクロールの作成、編集、または削除を許可するには、アクセス許可を設定する必要があります。詳細については、IAM ユーザーガイド の「サービスにリンクされた役割のアクセス許可」を参照してください。

Capacity Manager 用のサービスリンクロールの作成

AWSEC2CapacityManagerServiceRolePolicy ユースケースでサービスリンクロールを作成するには、IAM コンソールを使用できます。AWS CLI または AWS API では、`ec2.capacitymanager.amazonaws.com` サービス名を使用してサービスリンクロールを作成します。詳細については、IAM ユーザーガイドの「サービスリンクロールの作成」を参照してください。このサービスリンクロールを削除しても、同じ方法でロールを再作成できます。

Capacity Manager 用のサービスリンクロールの編集

Capacity Manager では、AWSServiceRoleForEC2CapacityManager サービスリンクロールを編集できません。サービスリンクロールの作成後は、さまざまなエンティティがロールを参照する可能性があるため、ロール名を変更することはできません。ただし、IAM を使用してロールの説明を編集することはできます。詳細については、「IAM ユーザーガイド」の「サービスリンクロールの編集」を参照してください。

Capacity Manager 用のサービスリンクロールの削除

サービスリンクロールを必要とする機能やサービスが不要になった場合は、ロールを削除することをお勧めします。そうすることで、使用していないエンティティがアクティブにモニタリングされたり、メンテナンスされたりすることがなくなります。ただし、手動で削除する前に、サービスリンクロールのリソースをクリーンアップする必要があります。

> **Note**
>
> リソースを削除しようとするときに Capacity Manager サービスがそのロールが使用していると、削除が失敗する場合があります。失敗した場合は数分待ってから操作を再試行してください。

AWSServiceRoleForEC2CapacityManager が使用している Capacity Manager リソースを削除する

1. すべての委任管理者は、組織のアクセスを削除する前に Capacity Manager を無効にしておく必要があります。
2. Capacity Manager を無効にする前に、アクティブなデータエクスポートを削除する必要があります。

サービスリンクロールを IAM で手動削除するには

AWSServiceRoleForEC2CapacityManager サービスリンクロールは、IAM コンソール、AWS CLI、または AWS API を使用して削除します。詳細については、「IAM ユーザーガイド」の「サービスにリンクされたロールの削除」を参照してください。

Capacity Manager サービスリンクロールがサポートされるリージョン

Capacity Manager は、このサービスを利用できるすべてのリージョンでサービスリンクロールの使用をサポートしています。詳細については、「AWS リージョンとエンドポイント」を参照してください。

Capacity Manager は、このサービスを利用できるすべてのリージョンでサービスリンクロールの使用をサポートしているわけではありません。AWSServiceRoleForEC2CapacityManager ロールは、以下のリージョンで使用できます。

リージョン名	リージョン識別子	Capacity Manager でのサポート
米国東部 (バージニア北部)	us-east-1	はい
米国東部 (オハイオ)	us-east-2	はい
米国西部 (北カリフォルニア)	us-west-1	はい
米国西部 (オレゴン)	us-west-2	はい
アフリカ (ケープタウン)	af-south-1	いいえ
アジアパシフィック (香港)	ap-east-1	いいえ
アジアパシフィック (ジャカルタ)	ap-southeast-3	いいえ
アジアパシフィック (ムンバイ)	ap-south-1	はい
アジアパシフィック (大阪)	ap-northeast-3	はい
アジアパシフィック (ソウル)	ap-northeast-2	はい
アジアパシフィック (シンガポール)	ap-southeast-1	はい
アジアパシフィック (シドニー)	ap-southeast-2	はい

リージョン名	リージョン識別子	Capacity Manager でのサポート
アジアパシフィック (東京)	ap-northeast-1	はい
カナダ (中部)	ca-central-1	はい
欧州 (フランクフルト)	eu-central-1	はい
欧州 (アイルランド)	eu-west-1	はい
欧州 (ロンドン)	eu-west-2	はい
欧州 (ミラノ)	eu-south-1	いいえ
欧州 (パリ)	eu-west-3	はい
欧州 (ストックホルム)	eu-north-1	はい
中東 (バーレーン)	me-south-1	いいえ
中東 (アラブ首長国連邦)	me-central-1	いいえ
南米 (サンパウロ)	sa-east-1	はい
AWS GovCloud (US-East)	us-gov-east-1	いいえ
AWS GovCloud (米国西部)	us-gov-west-1	いいえ

Capacity Manager でのデータの整理

Capacity Manager は、メトリクス、データポイント、ディメンション、日付範囲、期間の組み合わせを使用してキャパシティデータを整理します。これは、使用状況パターンを分析し、リソースに関して情報に基づいた判断を行うために役立ちます。

メトリクスとデータポイント

メトリクスとは、時間順に並べられた一連のデータポイントです。例えば、vCPU でのスポット使用状況を監視する場合は、`SpotTotalUsageHrsVcpu` メトリクスを使用します。

このメトリクスは、vCPU 時間単位でのスポット使用状況とタイムスタンプ付きのデータポイントを 1 時間ごとに生成します。例えば、午前 10 時に 100 vCPU を使用した場合、Capacity Manager はタイムスタンプが午前 10 時のデータポイントと 100 の値を作成します。

Capacity Manager が分析するメトリクスの完全なリストについては、「EC2 Capacity Manager のメトリクス」を参照してください。

ディメンション

ディメンションは、メトリクスのさまざまな側面を分類して特定するために役立つ名前と値のペアです。例えば、Capacity Manager にある 1 つのディメンションの名前は AccountID で、値は実際のアカウント ID です。Capacity Manager は、インスタンスファミリー、予約 ARN、予約タイプ、テナンシーなどのデータをセグメント化およびグループ化するためのディメンションを提供します。

組み込みディメンションに加えて、Amazon EC2 リソースからタグキーをアクティブ化して、カスタムディメンションとして使用できます。例えば、インスタンスに `environment` または `team` のタグが付けられている場合、それらのタグキーをアクティブ化し、その値でキャパシティメトリクスをグループ化してフィルタリングできます。

Capacity Manager は、EC2 Auto Scaling グループ (`aws:autoscaling:groupName`)、EKS クラスター名 (`aws:eks:cluster-name`)、EKS Kubernetes ノードプール (`eks:kubernetes-node-pool-name`)、Karpenter ノードプール (`karpenter.sh/nodepool`) などの Capacity Manager が提供するタグも提供します。これらのタグは、タグキーの制限にカウントされずにデフォルトで使用できます。

ディメンションの完全なリストについては、「EC2 Capacity Manager のメトリクス」を参照してください。タグディメンションのアクティブ化と管理の詳細については、「モニタリングされているタグキーの管理」を参照してください。

日付範囲と期間

日付範囲は、分析する時間を 1 時間から 90 日間の範囲で指定します。期間は、Capacity Manager が時間全体でデータを集計する方法と、返すデータポイントの数を決定します。例えば、日付範囲が 1 週間で期間が 1 日の場合、Capacity Manager は 7 個のデータポイントを返します。各データポイントは、1 日分の集計データを表します。期間は 1 時間間隔とし、日付範囲に均等に分割する必要があります。

トピック

- EC2 Capacity Manager のメトリクス

- データのグループ化とフィルタリング
- モニタリングされているタグキーの管理

EC2 Capacity Manager のメトリクス

Capacity Manager は、さまざまなリソースタイプ全体のキャパシティを追跡するための広範なメトリクス選択肢を提供します。メトリクスは、分析ニーズに応じて異なる単位を使用して測定できます。

Capacity Manager のメトリクス名には、測定されるキャパシティのタイプを分類するために 4 つの異なるプレフィックスが使用されています。

- `Reservation` – リザーブドキャパシティの合計、使用率、未使用のキャパシティ、予約数を含めたキャパシティ予約そのもののメトリクス。
- `Reserved` – キャパシティ予約が適用されるオンデマンドインスタンスの使用状況。
- `Unreserved` – キャパシティ予約外で実行されるオンデマンドインスタンスの使用状況。
- `Spot` – ランタイムや推定コストを含めたスポットインスタンス使用状況専用のメトリクス。これらのメトリクスは、予約ベースのキャパシティとは別のものです。

以下の表には、各メトリクスで利用できるディメンションも記載されています。ディメンションのカテゴリーは次のように分類されます。

- 一般的なキャパシティディメンション – アカウント ID、アカウント名、リージョン、インスタンスファミリー、アベイラビリティーゾーン、インスタンスタイプ、プラットフォーム、テナンシー
- キャパシティ予約ディメンション – 予約 ID (キャパシティ予約 ID)、予約 ARN、未使用の財務所有者、予約タイプ (ODCR/キャパシティブロック)、作成タイムスタンプ、開始タイムスタンプ、終了タイムスタンプ、状態、インスタンス一致基準。
- 予約使用状況ディメンション – 予約 ID (CRID)、予約 ARN、予約タイプ
- タグディメンション — カスタマーマネージドタグキーと Capacity Manager が提供するタグ。タグキーをディメンションとして使用する前に、タグキーをアクティブ化する必要があります。詳細については、「モニタリングされているタグキーの管理」を参照してください。

メトリクス	説明	利用可能なディメンション	利用可能な単位
ReservationAvgCommittedSize	コミットメントが設定されたアクティブまたはスケジュール済み状態のキャパシティの平均合計量。サイズはディメンション全体で合計され、時間全体で平均化されます。	一般的なキャパシティとキャパシティ予約のディメンション	vCPU、インスタンス
ReservationAvgFutureSize	将来起動することがスケジュールされているが、選択した期間中はまだアクティブ化されていないキャパシティ予約の平均量。サイズはディメンション全体で合計され、時間全体で平均化されます。	一般的なキャパシティとキャパシティ予約のディメンション	vCPU、インスタンス
ReservationAvgUtilization	選択した期間中に使用されたリザーブドキャパシティの平均パーセンテージ。	一般的なキャパシティとキャパシティ予約のディメンション	vCPU、インスタンス
ReservationMaxCommittedSize	コミットメントが設定されたアクティブまたはスケジュール済み状態の最大合計キャパシティ。サイズはディメンション全体で合計され、その期間の最大値が使用されます。	一般的なキャパシティとキャパシティ予約のディメンション	vCPU、インスタンス
ReservationMaxFutureSize	将来起動することがスケジュールされているが、選択した期間中はまだアクティブ化されていないキャパシティ予約の最大量。サイズはディメンション全体で合計され、期間の最大値が使用されます。	一般的なキャパシティとキャパシティ予約のディメンション	vCPU、インスタンス

メトリクス	説明	利用可能なディメンション	利用可能な単位
ReservationMaxSize	選択した期間中の任意の時点で到達したキャパシティ予約の最大サイズ。予約 ID が必要です。	一般的なキャパシティとキャパシティ予約のディメンション	vCPU、インスタンス
ReservationMaxUnusedSize	選択した期間中の任意の時点におけるキャパシティ予約の未使用キャパシティの最大量。予約 ID が必要です。	一般的なキャパシティとキャパシティ予約のディメンション	vCPU、インスタンス
ReservationMaxUtilization	選択した期間中の任意の時点で達成したキャパシティ予約の最大使用パーセンテージ。予約 ID が必要です。	一般的なキャパシティとキャパシティ予約のディメンション	
ReservationMinCommittedSize	コミットメントが設定されたアクティブまたはスケジュール済み状態のキャパシティの最大合計量。サイズはディメンション全体で合計され、その期間の最小値が使用されます。	一般的なキャパシティとキャパシティ予約のディメンション	vCPU、インスタンス
ReservationMinFutureSize	将来起動することが予定されているが、選択した期間のどの時点でもまだアクティブ化されていないキャパシティ予約の最大量。サイズはディメンション全体で合計され、その期間の最小値が使用されます。	一般的なキャパシティとキャパシティ予約のディメンション	vCPU、インスタンス
ReservationMinSize	選択した期間中の任意の時点で到達したキャパシティ予約の最小サイズ。予約 ID が必要です。	一般的なキャパシティとキャパシティ予約のディメンション	vCPU、インスタンス

メトリクス	説明	利用可能なディメンション	利用可能な単位
ReservationMinUnusedSize	選択した期間中の任意の時点におけるキャパシティ予約の未使用キャパシティの最小量。予約 ID は必須です。	一般的なキャパシティとキャパシティ予約のディメンション	vCPU、インスタンス
ReservationMinUtilization	選択した期間中の任意の時点で達成したキャパシティ予約の最小使用パーセンテージ。予約 ID が必要です。	一般的なキャパシティとキャパシティ予約のディメンション	
ReservationTotalCapacityHrs	選択した期間中にキャパシティ予約を使用して予約したキャパシティの合計量。	一般的なキャパシティとキャパシティ予約のディメンション	vCPU、インスタンス
ReservationTotalCount	選択した期間中に所有していた各種キャパシティ予約の総数。	一般的なキャパシティとキャパシティ予約のディメンション	
ReservationTotalEstimatedCost	選択した期間中に予約されたキャパシティ合計時間の推定コスト。	一般的なキャパシティとキャパシティ予約のディメンション	
ReservationUnusedTotalCapacityHrs	選択した期間中に所有していたが使用しなかったリザーブドキャパシティの合計量。	一般的なキャパシティとキャパシティ予約のディメンション	vCPU、インスタンス
ReservationUnusedTotalEstimatedCost	選択した期間中に所有していたが使用しなかったリザーブドキャパシティの推定コスト (オンデマンド料金を使用して算出)。	一般的なキャパシティとキャパシティ予約のディメンション	

メトリクス	説明	利用可能なディメンション	利用可能な単位
ReservedTotalEstimatedCost	選択した期間中にキャパシティ予約が適用されたオンデマンドインスタンスの推定使用コスト。スポットの使用は対象外です。	一般的なキャパシティと予約使用のディメンション	
ReservedTotalUsageHrs	選択した期間中にキャパシティ予約が適用されたオンデマンドインスタンスの合計使用時間。スポットの使用は対象外です。	一般的なキャパシティと予約使用のディメンション	vCPU、インスタンス
SpotAvgRunTimeBeforeInterruption	選択した期間に中断されたインスタンスの時間単位での平均ランタイム。	リージョン、AZ、およびアカウント ID ディメンションのみ	インスタンス
SpotInterruptionRate	選択した期間に中断された実行中のスポットインスタンスの割合。	リージョン、AZ、およびアカウント ID ディメンションのみ	vCPU、インスタンス
SpotMaxRunTimeBeforeInterruption	選択した期間に中断されたインスタンスの最大ランタイム。	リージョン、AZ、およびアカウント ID ディメンションのみ	インスタンス
SpotMinRunTimeBeforeInterruption	選択した期間に中断されたインスタンスの時間単位での最小ランタイム。	リージョン、AZ、およびアカウント ID ディメンションのみ	インスタンス

メトリクス	説明	利用可能なディメンション	利用可能な単位
SpotTotalCount	選択した期間中に実行されたスポットインスタンスまたは vCPU の数。	リージョン、AZ、およびアカウント ID ディメンションのみ	vCPU、インスタンス
SpotTotal EstimatedCost	選択した期間中のスポットインスタンスの推定使用コスト (公開されているスポット料金を使用して算出)。	一般的なキャパシティディメンション	
SpotTotal Interruptions	選択した期間中に中断されたスポットインスタンスまたは vCPU の数。	リージョン、AZ、およびアカウント ID ディメンションのみ	vCPU、インスタンス
SpotTotal UsageHrs	選択した期間中のスポットインスタンスの合計使用時間。	一般的なキャパシティディメンション	vCPU、インスタンス
UnreservedTotalEstimatedCost	選択した期間中にキャパシティ予約が適用されなかったオンデマンドインスタンスの推定使用コスト。スポットの使用は対象外です。	一般的なキャパシティディメンション	
UnreservedTotalUsageHrs	選択した期間中にキャパシティ予約が適用されなかったオンデマンドインスタンスの合計使用時間。スポットの使用は対象外です。	一般的なキャパシティディメンション	vCPU、インスタンス

> **Note**
>
> 単位にインスタンスを含める場合は、ディメンションにインスタンスタイプを含めることが推奨されます。

タグディメンション

前のセクションの組み込みディメンションに加えて、Capacity Manager はタグディメンションをサポートしています。タグディメンションを使用すると、Amazon EC2 リソースのタグキーを使用してメトリクスをグループ化およびフィルタリングできます。

カスタマーマネージドタグディメンション

ディメンションとして使用するタグキーは最大 5 つまでアクティブ化できます。アクティブ化後、`activated` ステータスで、タグディメンションは、一般的なキャパシティディメンションをサポートするメトリクスで使用できます。ディメンションカテゴリの完全なリストについては、「EC2 Capacity Manager のメトリクス」を参照してください。

ディメンション	説明	値の例
タグキー名	Amazon EC2 リソースからのカスタマーマネージドタグキー。	`environment` , `team`, `cost-center`

Capacity Manager 提供のタグディメンション

Capacity Manager は、デフォルトで次の Capacity Manager 提供のタグを提供します。Capacity Manager 提供のタグは常に利用可能であり、タグキーの制限にはカウントされません。

ディメンション	説明
`tag:aws:autoscaling:groupName`	インスタンスに関連付けられた EC2 Auto Scaling グループの名前。
`tag:aws:eks:cluster-name`	インスタンスに関連付けられた EKS クラスターの名前。

ディメンション	説明
tag:eks:kubernetes-node-pool-name	インスタンスに関連付けられた EKS Kubernetes ノードプール。
tag:karpenter.sh/nodepool	インスタンスに関連付けられた Karpenter ノードプール。

> **Note**
>
> タグディメンションでグループ化すると、そのタグの値を持たないリソースは、空の文字列値を持つ別のバケットに含まれます。これにより、合計がすべてのリソースに対して考慮されるようになります。

データのグループ化とフィルタリング

Capacity Manager は、ユーザーが選択したディメンションと期間に基づいてメトリクスを集計します。ディメンションが選択されていない場合、Capacity Manager は日付範囲内のデータを集約し、期間ごとに 1 つのデータポイントを返します。Amazon EC2 リソースからタグディメンションを含むディメンションを追加することで、データをより小さな集計にグループ化できます。

- グループ化 – リージョン、インスタンスファミリー、アカウント ID、またはタグキーなどのディメンション別にキャパシティデータを分類します。複数のディメンションでメトリクスをグループ化して、データをさらに分類することができます。例えば、リージョンとアベイラビリティーゾーンでグループ化すると、使用状況があるリージョンとアベイラビリティーゾーンの組み合わせに関するデータポイントを取得できます。
- フィルタリング – 選択したディメンションの特定のサブセットのみを表示します。例えば、インスタンスファミリーでグループ化すると、使用状況があるすべてのファミリーに関するデータポイントを取得できます。ただし、p5 でのフィルタリングも行う場合は、p5 インスタンスファミリーに関するデータポイントのみが表示されます。タグ値でフィルタリングすることもできます。例えば、environment=prod でフィルタリングして本番稼働用リソースのみを表示できます。選択したタグの値を持たないリソースをフィルタリングするには、空の値でフィルタリングします。
- メトリクス単位 – vCPU、インスタンス、推定コストといった異なる単位で結果を表示します。例えば、リージョン別にグループ化し、特定のインスタンスファミリーでフィルタリングした後、使

用する vCPU の合計数、実行中のインスタンス数、または推定コストを切り替えることができます。

コンソールでデータのグループ化とフィルタリングを実行する方法

Capacity Manager でデータのグループ化とフィルタリングを実行する

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Capacity Manager] を選択します。
3. [使用量]、[予約]、または [スポット] から、分析するリソースタイプのタブを選択します。
4. [日付フィルター] セクションで、[日付範囲]、[タイムゾーン]、[間隔] を選択します。
5. [ディメンションフィルター] セクションの [ディメンションを選択] ドロップダウンからディメンションを選択します。ディメンションのオプションはリソースタイプごとに異なります。詳細については、「EC2 Capacity Manager のメトリクス」を参照してください。

   タグディメンションでグループ化するには、ドロップダウンからタグキーを選択します (例: `environment`)。`activated` ステータスのタグキーのみが利用可能です。タグキーのアクティブ化の詳細については、「モニタリングされているタグキーの管理」を参照してください。

   複数のタグディメンションを含む複数のディメンションを追加して、より詳細なグループを作成できます。
6. 選択したディメンションでフィルタリングするには、[ディメンションでフィルタリング] ドロップダウンからフィルターオプションを選択します。

   タグディメンションでは、フィルターにはデータに存在するタグ値が表示されます。選択したタグの値がないリソースをフィルタリングするには、空の値オプションを選択します。
7. [集計] セクションで単位を選択して、vCPU、インスタンス、または推定コストごとの結果を表示します。

> **Note**
>
> タグディメンションでグループ化すると、そのタグの値を持たないリソースは、空の値を持つ別のグループに含まれます。これにより、合計が、タグ付けされたリソースだけでなく、すべてのリソースに対して考慮されるようになります。

モニタリングされているタグキーの管理

Amazon EC2 Capacity Manager では、Amazon EC2 リソース (`environment` や `team` など) からタグキーを選択して、キャパシティデータを分析するときにディメンションとして使用できます。タグキーがアクティブ化されたら、リージョン、インスタンスタイプ、またはアベイラビリティーゾーンで行えるのと同様に、そのタグの値でメトリクスをグループ化してフィルタリングできます。

各アカウントは、最大 5 つのタグキーをモニタリングできます。Capacity Manager が提供するタグは、この制限にはカウントされません。

トピック

- Capacity Manager 提供のタグ
- タグのライフサイクル
- モニタリングされているタグキーをアクティブ化または非アクティブ化する
- モニタリングされているタグキーを表示する
- タグディメンションを使用してメトリクスをクエリする
- データエクスポートのタグ
- 組織および委任管理者
- 考慮事項

Capacity Manager 提供のタグ

Capacity Manager には、アカウントごとに一連のデフォルトタグが含まれています。これらはタグキーの制限にはカウントされません。Capacity Manager 提供のタグは、一般的に使用されるグループ化ディメンションを表し、以下が含まれます:

- `aws:autoscaling:groupName` — EC2 Auto Scaling グループ
- `aws:eks:cluster-name` — EKS クラスター名
- `eks:kubernetes-node-pool-name` — EKS Kubernetes ノードプール
- `karpenter.sh/nodepool` — Karpenter ノードプール

Capacity Manager 提供のタグは、`CapacityManagerProvided` が `true` に 設定されている状態で `GetCapacityManagerMonitoredTagKeys` に表示され、お客様が有効化または無効化することはできません。Capacity Manager が最初に有効になると、Capacity Manager 提供のタグは

activating ステータスで始まり、Capacity Manager が Capacity Manager 提供のタグを含む最初のデータポイントを受け取った後、通常は 1～2 時間以内に activated に移行します。

タグのライフサイクル

モニタリングされているタグキーは、次のステータスで進行します:

ステータス	説明
activating	タグキーが登録されています。Capacity Manager は、このタグのデータを収集する準備をしています。このステータスのタグを使用してメトリクスをクエリすることはできません。
activated	タグデータは取り込み中であり、メトリクス API およびデータエクスポートを介してクエリできます。
suspended	タグキーが 100,000 個の一意のタグ値のしきい値を超えています。タグは引き続き制限にカウントされますが、Capacity Manager はそのタグのデータを取り込みません。タグ値の使用量が一定期間しきい値を下回ると、タグは自動的に再アクティブ化されます。
deactivating	タグキーは削除中です。非アクティブ化が完了すると、GetCapacityManagerMonitoredTagKeys に表示されなくなります。

タグが suspended ステータスの場合、GetCapacityManagerMonitoredTagKeys は次のステータスメッセージを返します:「タグ値が多すぎるためタグが中断されました。タグの使用を減らすか、非アクティブ化してください。」

> **Note**
>
> タグキーを非アクティブ化し、後で同じキーを再アクティブ化した場合、再アクティブ化後に取り込まれたデータのみがクエリ可能です。前回のアクティブ化の履歴データにはアクセスできません。EarliestDatapointTimestamp は、新しいアクティブ化のたびにリセットされます。

モニタリングされているタグキーをアクティブ化または非アクティブ化する

タグキーをアクティブ化してディメンションとしてモニタリングを開始するか、不要になったタグキーを非アクティブ化できます。アクティベーションが非同期です — そのため、タグは `activating` 状態になり、Capacity Manager がそのタグのデータを受け取り始めた後に `activated` に移行します。非アクティブ化すると、モニタリングされているセットからタグキーが削除されます。

> **Note**
>
> タグキーをアクティブ化するときは、キー名 (例: `environment`) のみを入力します。Capacity Manager は、それをグループ化およびフィルタリングするためのディメンションとして自動的に利用できるようにします。

Console

モニタリングされているタグキーをアクティブ化または非アクティブ化するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Capacity Manager] を選択します。
3. [設定] タブを選択します。
4. [モニタリングされているタグキー] セクションで、[タグキーを管理] を選択します。
5. タグキーをアクティブ化するには、タグキー名を入力し、[追加] を選択します。タグキーを非アクティブ化するには、タグキーを選択し、[削除] を選択します。
6. [Save changes] (変更の保存) をクリックします。

AWS CLI

タグキーをアクティブ化するには

次のコマンドを使用して、1 つまたは複数のタグキーをアクティブ化します。

```
aws ec2 update-capacity-manager-monitored-tag-keys \
    --activate-tag-keys "environment" "teamId"
```

出力には、タグキーとその初期ステータスが表示されます:

```
{
    "CapacityManagerTagKeys": [
        {
            "TagKey": "environment",
            "Status": "activating"
        },
        {
            "TagKey": "teamId",
            "Status": "activating"
        }
    ]
}
```

タグキーを非アクティブ化するには

次のコマンドを使用して、1 つまたは複数のタグキーを非アクティブ化します。

```
aws ec2 update-capacity-manager-monitored-tag-keys \
    --deactivate-tag-keys "project"
```

出力には、更新されたステータスが表示されます。

```
{
    "CapacityManagerTagKeys": [
        {
            "TagKey": "project",
            "Status": "deactivating"
        }
    ]
}
```

同じリクエストでタグキーをアクティブ化または非アクティブ化できます。

```
aws ec2 update-capacity-manager-monitored-tag-keys \
    --activate-tag-keys "environment" "teamId" \
    --deactivate-tag-keys "project"
```

モニタリングされているタグキーを表示する

現在のステータスやデータが利用可能になっている最も早いタイムスタンプなど、アカウントのすべてのモニタリングされているタグキーを表示できます。

Console

モニタリングされているタグキーを表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Capacity Manager] を選択します。
3. [設定] タブを選択します。
4. [モニタリングされているタグキー] セクションで、タグキー、ステータス、およびそれらが Capacity Manager 提供のタグであるかどうかを表示します。

AWS CLI

モニタリングされているタグキーを表示するには

次のコマンドを実行します。

```
aws ec2 get-capacity-manager-monitored-tag-keys
```

出力には、Capacity Manager 提供のタグやカスタマーマネージドタグなど、すべてのタグキーが一覧表示されます。

```
{
    "CapacityManagerTagKeys": [
        {
            "TagKey": "aws:autoscaling:groupName",
            "Status": "activated",
            "CapacityManagerProvided": true,
            "earliestDatapointTimestamp": "2026-04-08T00:00:00"
        },
        {
            "TagKey": "environment",
            "Status": "activated",
            "CapacityManagerProvided": false,
```

```
            "earliestDatapointTimestamp": "2025-08-11T22:00:00"
        }
    ]
}
```

CapacityManagerProvided フィールドは、タグが Capacity Manager 提供のタグ (true) かカスタマーマネージドタグ (false) かを示します。Capacity Manager 提供のタグは、タグキーの制限にはカウントされません。EarliestDatapointTimestamp は、そのタグキーのデータが利用可能になった最も早い時点を示します。

タグディメンションを使用してメトリクスをクエリする

タグキーが activated ステータスに達したら、それを GetCapacityManagerMetricDimensions および GetCapacityManagerMetricData のディメンションとして使用できます。

タグディメンション別にグループ化されたメトリクスをクエリするには

以下のコマンドを使用します。

```
aws ec2 get-capacity-manager-metric-dimensions \
    --group-by tag:environment account-id \
    --filter-by
 'DimensionCondition={Dimension=tag:environment,Comparison=equals,Values=[prod]}'
```

タグディメンションでグループ化すると、結果には、タグを持つリソースだけでなく、アカウント内のすべてのリソースが含まれます。タグの値を持たないリソースは、空の文字列値を持つ別のバケットにグループ化されます。例えば、アカウントが特定の期間に 800 vCPU 時間を使用し、それらのリソースの一部のみが environment タグを持っている場合、environment タグキーでグループ化すると以下が返される場合があります:

- prod — 300 vCPU 時間
- staging — 200 vCPU 時間
- "" (空の文字列) — environment タグのないリソースから 300 vCPU 時間

これにより、すべてのバケット全体で合計がフル使用量を占めるようになります。空の文字列をフィルター値として渡すことで、タグ付けされていないリソースを明示的にフィルタリングできます。

```
--filter-by
 'DimensionCondition={Dimension=tag:environment,Comparison=equals,Values=[""]}'
```

> **Note**
>
> まだ activating ステータスのタグキーを使用してクエリを実行すると、クエリは 400 エラーで拒否されます。タグのステータスが activated に変わるのを待ってからクエリを実行します。GetCapacityManagerMonitoredTagKeys を使用してステータスを確認できます。

> **Note**
>
> 指定されたタグディメンションの EarliestDatapointTimestamp より前の開始時刻のクエリは拒否されます。GetCapacityManagerMonitoredTagKeys を使用して、各タグでデータがいつ利用可能になったかを確認します。

データエクスポートのタグ

タグモニタリングを有効にすると、データエクスポートには、アクティブ化されたタグキーと Capacity Manager 提供のタグが追加の列として含まれます。タグ列は、tag:environment や tag:team などのヘッダーを持つすべての標準列の後に表示されます。タグ列はアルファベット順にソートされます。

エクスポートには、activated ステータスのタグのみが含まれます。Capacity Manager は、activating、deactivating、または suspended ステータスのタグを除外します。

> **Note**
>
> 新しいタグキーをアクティブ化した場合、既存のデータエクスポートには新しいタグが自動的に含まれることはありません。新しくアクティブ化されたタグキーを列として含めるには、新しいデータエクスポートを作成する必要があります。

組織および委任管理者

アカウントが組織レベルの Capacity Manager が有効になっている AWS 組織の一部である場合、各アカウント (組織管理者と委任管理者) はタグキーを個別にアクティブ化、非アクティブ化、およびクエリできます。各アカウントは、独自のタグステータス、EarliestDatapointTimestamp、およびタグキーの制限を維持します。

アカウントは、アカウント自体がアクティブ化したタグキーのメトリクスデータのみをクエリできます。組織管理者と委任管理者の両方が同じタグキー (environment など) をアクティブ化すると、各アカウントは独自のアクティブ化ステータスとデータの可用性を個別に追跡します。

委任管理者がタグキーを非アクティブ化すると、組織管理者が同じタグキーをアクティブ化している場合でも、委任管理者はそのタグのデータをクエリできなくなります。

考慮事項

- タグ値の更新: 新しいリソースのタグ値と、既存のリソースに新しく適用されたタグは、数時間以内に利用可能になります。リソースの既存のタグの値を変更すると、更新された値が Capacity Manager に反映されるまでに最大 24 時間かかることがあります。
- アクティブ化時間: タグキーをアクティブ化すると、タグが activated ステータスに移行し、データがクエリ可能になるまでに最大 24 時間かかることがあります。EarliestDatapointTimestamp は、タグがアクティブ化されたときではなく、データが利用可能になったときを表します。
- タグキーの制限: 各アカウントは、最大 5 つのタグキーをモニタリングできます。Capacity Manager が提供するタグは、この制限にはカウントされません。
- タグキー文字の要件: タグキーには、Unicode 文字、数字、空白、および次の文字を含めることができます: _ . : / = + @ -。タグキーは 128 文字を超えてはなりません。
- 再アクティブ化: 同じタグキーを非アクティブ化して再アクティブ化すると、新しいデータのみが利用可能になります。EarliestDatapointTimestamp はアクティブ化のたびにリセットされます。
- 停止されたタグ: 各タグキーは、最大 100,000 個の一意のタグ値をサポートしています。タグキーがこのしきい値を超えると、suspended ステータスに移行します。タグは引き続き制限にカウントされますが、データは取り込まれなくなります。タグの一意の値の数を減らすか、非アクティブ化して、別のタグキーのスペースを解放します。

AWS コンソールでの Capacity Manager のナビゲーション

Capacity Manager コンソールは、キャパシティデータのさまざまなビューを提供するタブで構成されています。

- ダッシュボード – すべてのオンデマンドキャパシティ予約、オンデマンド、およびスポットの使用量のおおまかな概要を提供するもので、キャパシティ体制の改善に役立つ主要メトリクスが含まれています。
- 使用量 – オンデマンドインスタンスとスポットインスタンスに関するインスタンス使用パターンの概要を提供します。柔軟なグループ化とフィルタリングを通じてキャパシティ予約の適用範囲を分析し、最適化機会を特定します。
- 予約 – アカウントとリージョン全体のキャパシティ予約使用率、管理機能、詳細な予約メトリクスの分析を提供します。
- スポット – アカウントとリージョン全体のスポット使用パターンを監視し、スポットインスタンスのコスト分析を提供します。
- データエクスポート – スケジューリング、フォーマット、テンプレートの選択など、Amazon S3 へのデータエクスポートの設定を管理します。
- 設定 – 組織のアクセス、リージョン設定、モニタリングされているタグキーの管理などのサービス設定オプションを提供します。Amazon EC2 リソースからタグキーをアクティブ化して、キャパシティメトリクスをグループ化およびフィルタリングするときにディメンションとして使用できます。詳細については、「モニタリングされているタグキーの管理」を参照してください。

[使用量] タブと [予約] タブ内では、Capacity Manager が階層的なナビゲーション構造を提供します。この構造は、おおまかな概要を詳細なリソース情報にドリルダウンすることを可能にします。このナビゲーションパターンを理解することは、キャパシティデータを効率的に分析して最適化機会を特定するために役立ちます。

トピック

- ナビゲーション階層
- 使用量と予約の内訳と詳細を表示する

ナビゲーション階層

Capacity Manager は、使用状況と予約に関する 3 レベルのナビゲーション構造を使用します。

1. 概要ページ – 集計されたメトリクスによるおおまかな概要。

2. 内訳ページ – フィルタリングとグループ化のオプションによる詳細な分析
3. 詳細ページ:
   - 使用状況の詳細 – 選択したディメンションの組み合わせに関する情報。使用パターンを理解するために役立つ統計と時系列データを提供します。
   - 予約の詳細 – 特定のキャパシティ予約に関する情報。使用率統計、経時的な使用パターン、設定の詳細などが含まれます。

> **Note**
>
> スポットは、概要ページのみを使用する簡素化された構造に従います。

使用量と予約の内訳と詳細を表示する

[使用量] タブと [予約] タブはどちらも同じ 3 レベルのナビゲーション構造に従うため、概要ページから内訳ページ、詳細ページへと進むことができます。内訳ページと詳細ページにアクセスするプロセスは似ていますが、各タブ内のナビゲーションリンクの場所にわずかな違いがあります。

リソースの内訳ページにアクセスする

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Capacity Manager] を選択します。
3. [使用量] または [予約] から、分析するリソースタイプのタブを選択します。
4. [集計] セクションで、内訳リンクを見つけます。
   - [使用量]: [アクション] 列で [内訳を表示] を選択します。

     > **Note**
     >
     > 画面のサイズまたは適用したディメンションの数によっては、ページを水平方向にスクロールして [内訳を表示] リンクを見つける必要がある場合があります。

   - [予約]: [予約] 列で、表示する予約の番号 (番号はリンク) を選択します。

詳細ページにアクセスする

1. 内訳ページで関連するセクションに移動します。

- [使用量]: 一意のディメンション組み合わせセクション。
- [予約]: 予約セクション。

2. 選択したリソースの関連セクションで詳細ページにアクセスします。

- [使用量]: [アクション] 列で [詳細を表示] を選択します。

> **Note**
>
> 画面のサイズまたは適用したディメンションの数によっては、ページを水平方向にスクロールして [詳細を表示] リンクを見つける必要がある場合があります。

- [予約]: [予約 ID] 列で表示する予約を選択します。

Capacity Manager データのエクスポート

EC2 Capacity Manager から Amazon S3 にキャパシティデータをエクスポートして、さらなる分析、カスタムレポートの作成、他の AWS サービスとの統合を行えるようにします。データは、CSV 形式または Parquet 形式でエクスポートできます。以下のセクションには、Capacity Manager データをエクスポートする方法に関する情報が記載されています。

> **Note**
>
> Capacity Manager は、AWS アカウントごとに 1 件のデータエクスポートのみを許可します。

> **Note**
>
> まれに、データの問題が原因でデータエクスポートを再処理する必要がある場合、新しいファイルはその時間の既存のファイルを上書きします。

トピック

- Capacity Manager データエクスポート用の Amazon S3 バケットのセットアップ
- Capacity Manager データ用のデータエクスポートの作成

Capacity Manager データエクスポート用の Amazon S3 バケットのセットアップ

Capacity Manager データエクスポートを受信するには、AWS アカウント アカウントにエクスポートファイルを受信して保存するための Amazon S3 バケットが必要です。所有している既存の Amazon S3 バケットの選択や、新しいバケットの作成は、Capacity Manager コンソールでデータエクスポートを作成するときに行えます。

いずれの場合も、Capacity Manager によるエクスポートファイルの配信を許可するために必要なバケットポリシーを適用する必要があります。データエクスポートを作成した後で Amazon S3 コンソールでこのポリシーを編集したり、S3 バケットの所有者を変更したりすると、Capacity Manager がエクスポートを配信できなくなります。

Amazon S3 バケットを作成するには、「Amazon Simple Storage Service ユーザーガイド」の「S3 バケットの作成」を参照してください。

Capacity Manager がデータエクスポートを配信できるようにするには、以下のポリシーを S3 バケットに適用する必要があります。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ec2.capacitymanager.amazonaws.com"
            },
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:ec2:us-east-1:111122223333:capacity-
manager-data-export/*"
```

```
                }
            }
        }
    ]
}
```

このバケットポリシーは、Capacity Manager のデータエクスポートファイルをバケットにセキュアに配信できることを確実にするために役立ちます。具体的には次のとおりです。

- Capacity Manager のデータエクスポートが配信されるたびに、バケットの所有者がエクスポートを設定したアカウントのまま変わっていないことを AWS が最初に確認します。バケットの所有権が変更されていた場合、エクスポートは配信されません。これは、Capacity Manager データのセキュリティの確保に役立ちます。このバケットポリシーは、AWS (`"Effect": "Allow"`) が、どのアカウントがバケットを所有するかを確認すること (`"Action": ["s3:ListBucket"]`) を許可します。
- ポリシーは、エクスポートファイルを書き込み (`"Action": "s3:PutObject"`)、オブジェクトを読み込んで (`"Action": "s3:GetObject"`) バケットにデータをコピーする許可を Capacity Manager サービス (`"Service": "ec2.capacitymanager.amazonaws.com"`) に付与します。

Capacity Manager データ用のデータエクスポートの作成

データエクスポートの作成には、Capacity Manager コンソールのデータエクスポートページ、または AWS CLI を使用できます。

前提条件

Amazon Simple Storage Service (Amazon S3) バケットを作成する必要があります。以下の条件を確認する必要があります。

- Capacity Manager を有効にした AWS リージョンと同じリージョン内に S3 バケットがあること。
- Capacity Manager サービスがファイルを配信するために必要な許可ポリシーが S3 バケットにあること。

詳細については、Capacity Manager データエクスポート用の Amazon S3 バケットのセットアップを参照してください。

手順

Capacity Manager データは、AWS コンソール、AWS CLI、または PowerShell を使用してエクスポートできます。

Console

データエクスポートを作成する

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Capacity Manager] を選択します。
3. [データエクスポート] タブを選択します。
4. [データエクスポートを作成] を選択します。
5. エクスポートのプロパティ、配信先、タグ (オプション) を設定します。
6. [作成] を選択します。

AWS CLI

データエクスポートを作成する

以下のコマンドを使用して、指定された設定でデータエクスポートを作成します。

```
aws ec2 create-capacity-manager-data-export \
    --s3-bucket-name my-exports-bucket \
    --s3-bucket-prefix capacity-data-exports \
    --schedule hourly \
    --output-format parquet/CSV \
    --tag-specifications 'ResourceType=capacity-manager-data-
export,Tags=[{Key=environment,Value=production}]'
```

> **Note**
>
> 上記のコマンドの `--tag-specifications` パラメータは、データエクスポートリソース自体にリソースタグを適用します (コスト配分や組織など)。これらは、エクスポートされたデータに含まれるタグ列を決定するモニタリングされているタグキーとは別です。モ

ニタリングされているタグキーの詳細については、「モニタリングされているタグキーの管理」を参照してください。

PowerShell

データエクスポートを作成する

New-EC2CapacityManagerDataExport コマンドレットを使用します。

```
New-EC2CapacityManagerDataExport `
    -S3BucketName "my-exports-bucket" `
    -S3BucketPrefix "capacity-data-exports" `
    -Schedule "hourly" `
    -OutputFormat "parquet" `
    -TagSpecification @([Amazon.EC2.Model.TagSpecification]@{
        ResourceType = "capacity-manager-data-export"
        Tags = @([Amazon.EC2.Model.Tag]@{
            Key   = "environment"
            Value = "production"
        })
    })
```

データエクスポートのタグ列

新しく作成されたデータエクスポートには、アクティブ化されたモニタリングされているタグキーと Capacity Manager 提供のタグの追加列としてタグ値が含まれます 詳細については、「モニタリングされているタグキーの管理」の データエクスポートのタグ を参照してください。

> **Note**
>
> モニタリングされているタグキーをアクティブ化していない場合でも、エクスポートには Capacity Manager 提供のタグ (aws:autoscaling:groupName、aws:eks:cluster-name、eks:kubernetes-node-pool-name、karpenter.sh/nodepool など) の列が含まれます。

EC2 インスタンスのデバイスドライバーを管理する

デバイスドライバーは、Amazon EC2 インスタンスの仮想ハードウェアと通信するソフトウェアコンポーネントです。システムエラー、パフォーマンスの問題、その他の予期しない動作を防ぐためには、ドライバーを最新の状態に保つことが重要です。このことは、ネットワーキングドライバー、グラフィックスドライバー、ストレージデバイスドライバーなど、使用状況に応じてシステムパフォーマンスに大きな影響を与える可能性のあるドライバーに特に当てはまります。新しいドライバーリリースには、不具合の修正が含まれていたり、現在実行中のインスタンスで活用できる機能拡張が導入されたりする場合があります。

ネットワークドライバー

Linux ディストリビューションでは、Elastic Network Adapter (ENA) や Elastic Fabric Adapter (EFA) などのネットワーク機能をカーネル内に組み込むことができます。ただし、カーネルドライバー機能の実装のタイミングは、ディストリビューションによって異なる場合があります。

ENA および EFA Linux カーネルドライバーは、Amazon Drivers GitHub リポジトリから入手できます。使用可能なドライバーの詳細とリンクについては、GitHub の「Amazon Drivers」を参照してください。

ENA ドライバーの詳細については、「EC2 インスタンスで ENA による拡張ネットワーキングを有効にする」を参照してください。EFA ドライバーの詳細については、このガイドの「Amazon EC2 の AI/ML および HPC ワークロード用の Elastic Fabric Adapter」セクションの「開始方法」のトピックを参照してください。

Windows インスタンスでネットワーキングドライバーをインストールまたは更新するには、以下のトピックを参照してください。

- Windows に ENA ドライバーをインストールする
- 最新の AWS PV ドライバーのインストール

  詳細については、「Windows インスタンス用 Paravirtual ドライバー」を参照してください。

> **Note**
>
> EFA は Windows インスタンスではサポートされていません。

グラフィックスドライバー

グラフィックスドライバーをインストールまたは更新するには、以下のトピックを参照してください。

- EC2 インスタンス用の AMD ドライバー
- Amazon EC2 インスタンス用の NVIDIA ドライバー

ストレージデバイスドライバー

ストレージドライバーをインストールまたは更新するには、以下のトピックを参照してください。

- Linux インスタンスの場合は、「Amazon EBS ユーザーガイド」の「NVMe ドライバのインストールまたはアップグレード」を参照してください。
- Windows インスタンスの場合は、「NVMe ドライバー」を参照してください。

その他の Windows デバイスドライバー

その他の Windows デバイスドライバーをインストールまたは更新するには、以下のトピックを参照してください。

- その他の Windows デバイスドライバー

EC2 インスタンス用の AMD ドライバー

AMD GPU がアタッチされたインスタンス (G4ad インスタンスなど) には、適切な AMD ドライバーがインストールされている必要があります。要件に応じて、ドライバーをプリインストールした AMI を使用するか、Amazon S3 からドライバーをダウンロードできます。

NVIDIA GPU がアタッチされたインスタンス (G4dn インスタンスなど) に NVIDIA ドライバーをインストールするには、代わりに「NVIDIA ドライバー」を参照してください。

目次

- エンタープライズドライバー向け AMD Radeon Pro ソフトウェア
- AMDドライバーをインストールした AMI
- AMD ドライバーのダウンロード

エンタープライズドライバー向け AMD Radeon Pro ソフトウェア

エンタープライズドライバー向け AMD Radeon Pro ソフトウェアは、プロフェッショナルグレードのグラフィックスのユースケースをサポートするために構築されています。ドライバーを使用して、GPU ごとに 2 つの 4K ディスプレイでインスタンスを設定できます。

サポートされている API

- OpenGL、OpenCL
- Vulkan
- AMD Advanced Media Framework
- Video Acceleration API
- DirectX 9 以降
- Microsoft Hardware Media Foundation Transform

AMDドライバーをインストールした AMI

AWS では、AMD ドライバーがインストールされたさまざまな Amazon マシンイメージ (AMI) を提供しています。AMD ドライバーで Marketplace 製品を開きます

AMD ドライバーのダウンロード

AMD ドライバーがインストールされた AMI を使用していない場合は、AMD ドライバーをダウンロードしてインスタンスにインストールできます。次のオペレーティングシステムのバージョンのみが AMD ドライバーをサポートしています。

- カーネルバージョン 5.4 搭載の Amazon Linux 2
- Ubuntu 20.04
- Ubuntu 22.04
- Ubuntu 24.04
- Windows Server 2016
- Windows Server 2019
- Windows Server 2022

これらのダウンロードは、AWS カスタマーのみが利用できます。ダウンロードすることで、AMD Radeon Pro V520 ハードウェアの使用において、ダウンロードしたソフトウェアを AMIs の開

発用のみで使用することに同意したことになります。このソフトウェアをインストールすることは、AMD End User License Agreement の規約の遵守に同意したものと見なされます。

Amazon Linux 2 Linux インスタンスに AMD ドライバーをインストールする

1. Linux インスタンスに接続します。
2. Linux インスタンスに AWS CLI をインストールし、デフォルトの認証情報を設定します。詳細については、AWS CLI ユーザーガイドのAWS Command Line Interface のインストールを参照してください。

   > **Important**
   >
   > ユーザーまたはロールは、[AmazonS3ReadOnlyAccess] ポリシーを含む許可を持っている必要があります。詳細については、「Amazon Simple Storage Service ユーザーガイド」の「AWS 管理されたポリシー: AmazonS3ReadOnlyAccess」を参照してください。

3. カーネル 5.4 をインストールする

   ```
   $ sudo amazon-linux-extras disable kernel-5.10
   $ sudo amazon-linux-extras enable kernel-5.4
   $ sudo yum install -y kernel
   ```

4. gcc および make をインストールします (まだインストールされていない場合)。

   ```
   $ sudo yum install gcc make
   ```

5. パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

   ```
   $ sudo amazon-linux-extras install epel -y
   $ sudo yum update -y
   ```

6. インスタンスを再起動します。

   ```
   $ sudo reboot
   ```

7. 再起動後にインスタンスに再接続します。
8. 最新の AMD ドライバーをダウンロードします。

   ```
   $ aws s3 cp --recursive s3://ec2-amd-linux-drivers/latest/ .
   ```

9. ファイルを抽出します。

```
$ tar -xf amdgpu-pro-*rhel*.tar.xz
```

10. 抽出されたドライバーのフォルダに変更します。
11. 自己インストールスクリプトを実行して、完全なグラフィックススタックをインストールします。

```
$ ./amdgpu-pro-install -y --opencl=pal,legacy
```

12. インスタンスを再起動します。

```
$ sudo reboot
```

13. ドライバーが機能していることを確認します。

```
$ sudo dmesg | grep amdgpu
```

レスポンスは次のようになります。

```
Initialized amdgpu
```

Ubuntu Linux インスタンスに AMD ドライバーをインストールする

1. Linux インスタンスに接続します。
2. パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
$ sudo apt-get update --fix-missing && sudo apt-get upgrade -y
```

3. gcc および make をインストールします (まだインストールされていない場合)。

```
$ sudo apt install build-essential -y
```

4. Linux ファームウェアとカーネルモジュールをインストールする

```
$ sudo apt install linux-firmware linux-modules-extra-aws -y
```

5. インスタンスを再起動する

```
$ sudo reboot
```

6. 再起動後にインスタンスに再接続します。
7. AMD Linux ドライバーパッケージをインストールする

   - Ubuntu 20.04 の場合:

   ```
   $ wget https://repo.radeon.com/.preview/afe3e25b8f1beff0bb312e27924d63b5/amdgpu-install/5.4.02.01/ubuntu/focal/amdgpu-install_5.4.02.01.50402-1_all.deb
   $ sudo dpkg --add-architecture i386
   $ sudo apt install ./amdgpu-install_5.4.02.01.50402-1_all.deb
   ```

   - これ以降の Ubuntu バージョンについては、「Linux® Drivers for AMD Radeon™ Graphics」にアクセスして最新の Ubuntu パッケージをダウンロードし、インストールしてください。

   ```
   $ sudo apt install ./amdgpu-install_{version-you-downloaded}.deb
   ```

8. 自己インストールスクリプトを実行して、完全なグラフィックススタックをインストールします。

   ```
   $ amdgpu-install --usecase=workstation --vulkan=pro -y
   ```

9. インスタンスを再起動します。

   ```
   $ sudo reboot
   ```

10. ドライバーが機能していることを確認します。

    ```
    $ sudo dmesg | grep amdgpu
    ```

    レスポンスは次のようになります。

    ```
    Initialized amdgpu
    ```

Windows インスタンスに AMD ドライバーをインストールする

1. Windows インスタンスに接続し、PowerShell ウィンドウを開きます。

2. Windows インスタンスで AWS Tools for Windows PowerShell のデフォルトの認証情報を設定します。詳細については、AWS Tools for Windows PowerShell ユーザーガイドの「AWS Tools for PowerShell の使用開始」を参照してください。

> **Important**
>
> ユーザーまたはロールは、[AmazonS3ReadOnlyAccess] ポリシーを含む許可を持っている必要があります。詳細については、「Amazon Simple Storage Service ユーザーガイド」の「AWS 管理されたポリシー: AmazonS3ReadOnlyAccess」を参照してください。

3. Windows のバージョンに従ってキープレフィックスを設定します。

   - Windows 10 および Windows 11

   ```
   $KeyPrefix = "latest/AMD_GPU_WINDOWS10"
   ```

   - Windows Server 2016

   ```
   $KeyPrefix = "archives"
   ```

   - Windows Server 2019

   ```
   $KeyPrefix = "latest/AMD_GPU_WINDOWS_2K19" # use "archives" for Windows Server
    2016
   ```

   - Windows Server 2022

   ```
   $KeyPrefix = "latest/AMD_GPU_WINDOWS_2K22"
   ```

4. 次の PowerShell コマンドを使用して、Amazon S3 からデスクトップにドライバーをダウンロードします。

```
$Bucket = "ec2-amd-windows-drivers"
$LocalPath = "$home\Desktop\AMD"
$Objects = Get-S3Object -BucketName $Bucket -KeyPrefix $KeyPrefix -Region us-east-1
foreach ($Object in $Objects) {
$LocalFileName = $Object.Key
if ($LocalFileName -ne '' -and $Object.Size -ne 0) {
    $LocalFilePath = Join-Path $LocalPath $LocalFileName
```

```
    Copy-S3Object -BucketName $Bucket -Key $Object.Key -LocalFile $LocalFilePath -
Region us-east-1
    }
}
```

5. ダウンロードしたドライバファイルを解凍し、次の PowerShell コマンドを使用してインストーラを実行します。

```
Expand-Archive $LocalFilePath -DestinationPath "$home\Desktop\AMD\$KeyPrefix" -
Verbose
```

ここで、新しいディレクトリの内容を確認します。ディレクトリ名は、Get-ChildItem PowerShell コマンドを使用して取得できます。

```
Get-ChildItem "$home\Desktop\AMD\$KeyPrefix"
```

出力は次の例のようになります:

```
Directory: C:\Users\Administrator\Desktop\AMD\latest

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----       10/13/2021  12:52 AM                210414a-365562C-Retail_End_User.2
```

ドライバーをインストール :

```
pnputil /add-driver $home\Desktop\AMD\$KeyPrefix\*.inf /install /subdirs
```

6. ドライバーをインストールする手順にしたがい、必要に応じてインスタンスを再起動します。
7. GPU が正しく動作していることを確認するには、デバイスマネージャーをチェックします。ディスプレイアダプタとして「AMD Radeon Pro V520 MxGPU」が表示されます。
8. 最大 4K の解像度のディスプレイを 4 台活用するには、高性能ディスプレイプロトコル Amazon DCV を設定します。

Amazon EC2 インスタンス用の NVIDIA ドライバー

NVIDIA GPU がアタッチされたインスタンス (P- や G- シリーズのインスタンスタイプなど) には、適切な NVIDIA ドライバーがインストールされている必要があります。インスタンスタイプに応じ

て、公開された NVIDIA ドライバーまたは AWS カスタマーのみが使用できる Amazon S3 のドライバーをダウンロードするか、ドライバーが事前インストールされた AWS AMI を使用します。

AMD GPU がアタッチされたインスタンス (G4ad インスタンスなど) に AMD ドライバーをインストールするには、代わりに「AMD ドライバー」を参照してください。

トピック

- NVIDIA ドライバーの種類
- インスタンスタイプ別の使用可能なドライバー
- インストールオプション
- NVIDIA ドライバーが付属する AMI を使用する
- NVIDIA パブリックドライバーのインストール
- NVIDIA GRID ドライバー (G7e、G6、Gr6、G6e、G6f、Gr6f、G5、G4dn、G3 インスタンス)のインストール
- NVIDIA ゲームドライバー (G7e、G6、G6e、G5、G4dn インスタンス)のインストール

NVIDIA ドライバーの種類

GPU ベースのインスタンスで使用できる主な種類の NVIDIA ドライバーを次に示します。

Tesla ドライバー

これらのドライバーは主に、機械学習用の並列浮動小数点計算、ハイパフォーマンスコンピューティングアプリケーション用の高速フーリエ変換などの計算タスクに GPU を使用するコンピューティングワークロードを対象としています。

GRID ドライバー

これらのドライバーは、3D モデルや高解像度動画などのコンテンツをレンダリングするプロフェッショナルな視覚化アプリケーションに最適なパフォーマンスを提供することが認定されています。GRID ドライバーを構成すると、2 つのモードをサポートできます。Quadro Virtual Workstation は、GPU あたり 4 台の 4K ディスプレイへのアクセスを提供します。GRID vApps は、RDSH アプリのホスティング機能を提供します。

ゲームドライバー

これらのドライバーはゲーム用に最適化されており、パフォーマンスを向上させるために頻繁に更新されます。これらは GPU あたり単一の 4K ディスプレイをサポートします。

設定モード

Windows では、Tesla ドライバーは Tesla Compute Cluster (TCC) モードで実行するように設定されています。GRID ドライバーとゲームドライバーは、Windows Display Driver Model (WDDM) モードで実行するように設定されています。TCC モードでは、カードはコンピューティングワークロード専用です。WDDM モードでは、カードはコンピューティングワークロードとグラフィックスワークロードの両方をサポートします。

NVIDIA コントロールパネル

NVIDIA コントロールパネルは、GRID およびゲームドライバーでサポートされています。Tesla ドライバーではサポートされていません。

Tesla、GRID、ゲームドライバーでサポートされている API

- OpenCL、OpenGL、Vulkan
- NVIDIA CUDA および関連ライブラリ (cuDNN、TensorRT、nvJPEG、cuBLAS など)
- 動画エンコード用の NVENC と動画デコード用の NVDEC
- Windows 専用 API: DirectX、Direct2D、DirectX Video Acceleration、DirectX Raytracing

インスタンスタイプ別の使用可能なドライバー

次の表は、GPU インスタンスタイプごとにサポートされている NVIDIA ドライバーをまとめたものです。

インスタンスタイプ	Tesla ドライバー	GRID ドライバー	ゲームドライバー
G3	はい	はい	いいえ
G4dn	はい	はい	はい
G5	はい	はい	はい
G5g	はい [1]	いいえ	いいえ
G6	はい	はい	はい
G6e	はい	はい	はい
G6f	いいえ	はい	いいえ

インスタンスタイプ	Tesla ドライバー	GRID ドライバー	ゲームドライバー
Gr6	はい	はい	いいえ
Gr6f	いいえ	はい	いいえ
G7e	はい	はい	はい
P3	はい	いいえ	いいえ
P4d	はい	いいえ	いいえ
P4de	はい	いいえ	いいえ
P5	はい	いいえ	いいえ
P5e	はい	いいえ	いいえ
P5en	はい	いいえ	いいえ
P6-B200	はい	いいえ	いいえ
P6e-GB200	はい	いいえ	いいえ
P6-B300	はい	いいえ	いいえ

[1] この Tesla ドライバーは、ARM64 プラットフォーム固有の最適化されたグラフィックスアプリケーションもサポートしています。

インストールオプション

次のいずれかのオプションを使用して、GPU インスタンスに必要な NVIDIA ドライバーを取得します。

オプション

1. NVIDIA ドライバーが付属する AMI を使用する
2. NVIDIA パブリックドライバーのインストール
3. NVIDIA GRID ドライバー (G7e、G6、Gr6、G6e、G6f、Gr6f、G5、G4dn、G3 インスタンス)のインストール

4. NVIDIA ゲームドライバー (G7e、G6、G6e、G5、G4dn インスタンス)のインストール

NVIDIA ドライバーが付属する AMI を使用する

AWS と NVIDIA では、NVIDIA ドライバーがインストールされたさまざまな Amazon マシンイメージ (AMI) を提供しています。

- Tesla パブリックドライバーを使用した Marketplace 製品
- GRID ドライバーを使用した Marketplace 製品
- ゲームドライバーを使用した Marketplace 製品

オペレーティングシステム (OS) プラットフォームに依存する考慮事項を確認するには、ご自分の AMI が該当するタブを選択してください。

Linux

これらの AMI のいずれかを使用してインストールされたドライバーのバージョンを更新するには、バージョンの競合を避けるために、インスタンスから NVIDIA パッケージをアンインストールする必要があります。次のコマンドを使用して、NVIDIA パッケージをアンインストールします。

```
[ec2-user ~]$ sudo yum erase nvidia cuda
```

CUDA ツールキットパッケージは、NVIDIA ドライバーに依存します。NVIDIA パッケージをアンインストールすると、CUDA ツールキットが消去されます。NVIDIA ドライバーをインストールした後に、CUDA ツールキットを再インストールする必要があります。

Windows

いずれかの AWS Marketplace サービスを使用してカスタム Windows AMI を作成する場合、GRID ドライバーを動作させるには、AMI が Windows Sysprep を使用して作成された標準化されたイメージである必要があります。詳細については、「Windows Sysprep を使用して Amazon EC2 AMI を作成する」を参照してください。

NVIDIA パブリックドライバーのインストール

「NVIDIA ドライバーが付属する AMI を使用する」に記載されている AWS Marketplace AMI がユースケースに適さない場合、パブリックドライバーをインストールして独自のライセンスを使用できます。インストールオプションは以下のとおりです。

- オプション 1: ドライバーのみのインストール
- オプション 2: CUDA ツールキットを使用して をインストールする (Linux ディストリビューションに推奨)

P6-B200 および P6-B300 インスタンスタイプに関する考慮事項

P6-B200 および P6-B300 プラットフォームは、Mellanox ConnectX ネットワークインターフェイスカード (NIC) を PCIe デバイスとしてインスタンスに公開するという点で独特です。これらの NIC は一般的なネットワークインターフェイスとして機能しませんが、GPU 相互接続の NVLink トポロジである NVFabric を初期化および設定するための制御パスを提供する NVSwitch ブリッジとして機能します。

システムを完全に初期化するには、NVIDIA Fabric Manager が `NVFabric` を設定し、NVSwitch トポロジを確立する必要があります。これにより、InfiniBand カーネルモジュールは Mellanox ConnectX NIC と通信できるようになります。

NVIDIA Fabric Manager は CUDA ツールキットに含まれています。このインスタンスタイプには「オプション 2: CUDA ツールキットを使用して をインストールする」をお勧めします。

オプション 1: ドライバーのみのインストール

特定のドライバーをインストールするには、お使いのインスタンスにログオンし、使用するインスタンスタイプに適した 64 ビット NVIDIA パブリックドライバーを http://www.nvidia.com/Download/Find.aspx からダウンロードします。[製品タイプ]、[製品シリーズ]、[製品] の順にクリックし、次の表に示すオプションを使用します。

次に、「NVIDIA Driver Installation Guide」の「Local Repository Installation」の手順に沿って操作します。

> **Note**
>
> P6-B200 および P6-B300 インスタンスタイプの場合、NVIDIA CUDA Toolkit にバンドルされた追加のパッケージのインストールと設定が必要です。詳細については、「オプション 2: CUDA ツールキットを使用して をインストールする」のLinux ディストリビューションの手順を参照してください。

インスタンス	製品タイプ	製品シリーズ	製品	最小ドライバーバージョン
G3	Tesla	M-Class	M60	--
G4dn	Tesla	T シリーズ	T4	--
G5	Tesla	A シリーズ	A10	470.00 以降
G5g[1]	Tesla	T シリーズ	T4G	470.82.01 以降
G6	Tesla	L シリーズ	L4	525.0 以降
G6e	Tesla	L シリーズ	L40S	535.0 以降
Gr6	Tesla	L シリーズ	L4	525.0 以降
G7e	Tesla	HGX シリーズ	RTX PRO 6000 Blackwell	575.0 以降
P3	Tesla	V シリーズ	V100	--
P4d	Tesla	A シリーズ	A100	--
P4de	Tesla	A シリーズ	A100	--
P5	Tesla	H シリーズ	H100	530 以降
P5e	Tesla	H シリーズ	H200	550 以降
P5en	Tesla	H シリーズ	H200	550 以降

インスタンス	製品タイプ	製品シリーズ	製品	最小ドライバーバージョン
P6-B200[2]	Tesla	HGX シリーズ	B200	570 以降
P6e-GB200	Tesla	HGX シリーズ	B200	570 以降
P6-B300[2]	Tesla	HGX シリーズ	B300	580 以降

[1] G5g インスタンスのオペレーティングシステムは Linux aarch64 です。

[2] P6-B200 および P6-B300 インスタンスタイプの場合、NVIDIA Fabric Manager を設定するための追加のインストール要件があります。

オプション 2: CUDA ツールキットを使用して をインストールする

インストール手順はオペレーティングシステムによって若干異なります。NVIDIA CUDA ツールキットを使用してインスタンスにパブリックドライバーをインストールするには、インスタンスオペレーティングシステムの手順に従います。ここに示されていないインスタンスオペレーティングシステムの場合は、NVIDIA Developer ウェブサイトに掲載されているオペレーティングシステムとインスタンスタイプのアーキテクチャの手順に従ってください。詳細については、「CUDA Toolkit Downloads」を参照してください。

インスタンスタイプのアーキテクチャやその他の仕様については、「Amazon EC2 インスタンスタイプ」リファレンスの「Accelerated computing specifications」を参照してください。

Amazon Linux 2023

このセクションでは、Amazon Linux 2023 インスタンスへの NVIDIA CUDA ツールキットのインストールについて説明します。このセクションのコマンド例は、x86_64 アーキテクチャに基づいています。

`arm64-sbsa` コマンドについては、「CUDA Toolkit Downloads」を参照して、ディストリビューションに適用するオプションを選択してください。手順は、最終的な選択を行った後に表示されます。

前提条件

ツールキットとドライバーをインストールする前に、以下のコマンドを実行して、カーネルヘッダーと開発パッケージのバージョンが正しいことを確認します。

```
[ec2-user ~]$ sudo dnf install kernel-devel-$(uname -r) kernel-headers-$(uname -r) -y
```

ツールキットとドライバーをダウンロードする

インスタンスに使用するインストールのタイプを選択し、関連する手順に従います。

AL2023 NVIDIA repository

Amazon Linux 2023 は、AWS が維持する専用リポジトリを通じて NVIDIA GPU ドライバーと CUDA ツールキットパッケージを提供します。AWS は、このリポジトリを AL2023 リリース候補で認定し、Amazon Linux セキュリティセンターを通じてセキュリティアドバイザリを提供します。AL2023 インスタンスでは、インストールを簡素化し、標準 `dnf update` ワークフローを通じてドライバーを更新するため、このオプションをお勧めします。

手順については、「Amazon Linux 2023 ユーザーガイド」の「NVIDIA ドライバー」を参照してください。

RPM local installation

以下の手順に従って、CUDA ツールキットインストーラリポジトリバンドルをインスタンスにダウンロードし、指定されたバンドルを抽出して登録できます。

NVIDIA Developer ウェブサイトで手順を確認するには、「CUDA Toolkit Downloads」を参照してください。

```
[ec2-user ~]$ wget https://developer.download.nvidia.com/compute/cuda/13.0.0/
local_installers/cuda-repo-amzn2023-13-0-local-13.0.0_580.65.06-1.x86_64.rpm
[ec2-user ~]$ sudo rpm -i cuda-repo-amzn2023-13-0-
local-13.0.0_580.65.06-1.x86_64.rpm
```

RPM network installation

これらの手順に従って、CUDA リポジトリをインスタンスのパッケージマネージャーに登録できます。インストール手順を実行すると、パッケージマネージャーによって必要なパッケージのみがダウンロードされます。

NVIDIA Developer ウェブサイトで手順を確認するには、「CUDA Toolkit Downloads」を参照してください。

```
[ec2-user ~]$ sudo dnf config-manager --add-repo https://
developer.download.nvidia.com/compute/cuda/repos/amzn2023/x86_64/cuda-amzn2023.repo
```

残りの手順は、RPM ローカルインストールと RPM ネットワークインストールの両方で同じです。

1. CUDA ツールキットのインストールを完了します

```
[ec2-user ~]$ sudo dnf clean all
[ec2-user ~]$ sudo dnf install cuda-toolkit -y
```

2. ドライバーのオープンカーネルモジュールバリアントをインストールします

```
[ec2-user ~]$ sudo dnf module install nvidia-driver:open-dkms -y
```

3. GPUDirect Storage と Fabric Manager をインストールします

```
[ec2-user ~]$ sudo dnf install nvidia-gds -y
[ec2-user ~]$ sudo dnf install nvidia-fabric-manager -y
```

4. Fabric Manager とドライバーの永続性を有効にします

```
[ec2-user ~]$ sudo systemctl enable nvidia-fabricmanager
[ec2-user ~]$ sudo systemctl enable nvidia-persistenced
```

5. (P6-B200 および P6-B300 のみ) これらのインスタンスタイプの場合、NVIDIA CUDA Toolkit にバンドルされた追加のパッケージのインストールと設定が必要です。

   1. NVIDIA Link Subnet Manager と `ibstat` をインストールします。

      ```
      [ec2-user ~]$ sudo dnf install nvlink5
      ```

   2. 起動時の Infiniband モジュールの自動ロードを有効にします。

      ```
      [ec2-user ~]$ echo "ib_umad" | sudo tee -a /etc/modules-load.d/modules.conf
      ```

6. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

Ubuntu 24.04

このセクションでは、Ubuntu 24.04 インスタンスへの NVIDIA CUDA ツールキットのインストールについて説明します。このセクションのコマンド例は、x86_64 アーキテクチャに基づいています。

arm64-sbsa コマンドについては、「CUDA Toolkit Downloads」を参照して、ディストリビューションに適用するオプションを選択してください。手順は、最終的な選択を行った後に表示されます。

前提条件

ツールキットとドライバーをインストールする前に、以下のコマンドを実行して、カーネルヘッダーと開発パッケージのバージョンが正しいことを確認します。

```
$ apt install linux-headers-$(uname -r)
```

ツールキットとドライバーをダウンロードする

インスタンスに使用するインストールのタイプを選択し、関連する手順に従います。

Deb local installation

以下の手順に従って、CUDA ツールキットインストーラリポジトリバンドルをインスタンスにダウンロードし、指定されたバンドルを抽出して登録できます。

NVIDIA Developer ウェブサイトで手順を確認するには、「CUDA Toolkit Downloads」を参照してください。

```
$ wget https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2404/
x86_64/cuda-ubuntu2404.pin
$ sudo mv cuda-ubuntu2404.pin /etc/apt/preferences.d/cuda-repository-pin-600
$ wget https://developer.download.nvidia.com/compute/cuda/13.0.0/
local_installers/cuda-repo-ubuntu2404-13-0-local_13.0.0-580.65.06-1_amd64.deb
$ sudo dpkg -i cuda-repo-ubuntu2404-13-0-local_13.0.0-580.65.06-1_amd64.deb
$ sudo cp /var/cuda-repo-ubuntu2404-13-0-local/cuda-*-keyring.gpg /usr/share/
keyrings/
```

Deb network installation

これらの手順に従って、CUDA リポジトリをインスタンスのパッケージマネージャーに登録できます。インストール手順を実行すると、パッケージマネージャーによって必要なパッケージのみがダウンロードされます。

NVIDIA Developer ウェブサイトで手順を確認するには、「CUDA Toolkit Downloads」を参照してください。

```
$ wget https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2404/x86_64/
cuda-keyring_1.1-1_all.deb
$ sudo dpkg -i cuda-keyring_1.1-1_all.deb
```

残りの手順は、ローカルインストールとネットワークインストールの両方で同じです。

1. CUDA ツールキットのインストールを完了します

```
$ sudo apt update
$ sudo apt install cuda-toolkit -y
```

2. ドライバーのオープンカーネルモジュールバリアントをインストールします

```
$ sudo apt install nvidia-open -y
```

3. GPUDirect Storage と Fabric Manager をインストールします

```
$ sudo apt install nvidia-gds -y
$ sudo apt install nvidia-fabricmanager -y
```

4. Fabric Manager とドライバーの永続性を有効にします

```
$ sudo systemctl enable nvidia-fabricmanager
$ sudo systemctl enable nvidia-persistenced
```

5. (P6-B200 および P6-B300 のみ) これらのインスタンスタイプの場合、NVIDIA CUDA Toolkit にバンドルされた追加のパッケージのインストールと設定が必要です。

   1. 最新の InfiniBand 固有のデバイスドライバーと診断ユーティリティをインストールします。

      ```
      $ sudo apt install linux-modules-extra-$(uname -r) -y
      $ sudo apt install infiniband-diags -y
      ```

   2. NVIDIA Link Subnet Manager をインストールします。

      ```
      $ sudo apt install nvlsm -y
      ```

6. インスタンスを再起動します。

```
sudo reboot
```

7. パスを更新し、以下の環境変数を追加します。

```
$ export PATH=${PATH}:/usr/local/cuda-13.0/bin
$ export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:/usr/local/cuda-13.0/lib64
```

Windows オペレーティングシステム

Windows に NVIDIA ドライバーをインストールするには、次のステップに従ってください。

1. ドライバーをダウンロードしたフォルダを開き、インストールファイルを起動します。ドライバーをインストールする手順にしたがい、必要に応じてインスタンスを再起動します。
2. デバイスマネージャを使用して、警告アイコンが表示されている [Microsoft 基本ディスプレイアダプター] という名前のディスプレイアダプターを無効にします。Windows の機能である、Media Foundation および Quality Windows Audio Video Experience をインストールします。

   > **Important**
   >
   > [Microsoft リモートディスプレイアダプター] という名前のディスプレイアダプターを無効にしないでください。[Microsoft リモートディスプレイアダプター] が無効になっていると、接続が中断され、再起動後にインスタンスに接続しようとすると失敗する可能性があります。

3. デバイスマネージャーで、GPU が正しく動作していることを確認します。
4. GPU の最善のパフォーマンスを実現するには、「Amazon EC2 インスタンスの GPU 設定を最適化する」の最適化ステップを完了します。

NVIDIA GRID ドライバー (G7e、G6、Gr6、G6e、G6f、Gr6f、G5、G4dn、G3 インスタンス)のインストール

これらのダウンロードは、AWS カスタマーのみが利用できます。ダウンロードすると、NVIDIA GRID Cloud のエンドユーザーライセンス契約 (EULA) で記述されている AWS ソリューションの要件を遵守するには、ダウンロードしたソフトは NVIDIA L4、NVIDIA L40S、NVIDIA A10G、NVIDIA Tesla T4、NVIDIA Tesla M60 のハードウェアに使用する AMI のみを開発するために使用することに同意します。GRID ドライバーを使用して、AWS 環境内で AMI の作成および使用の両方ができます。このソフトウェアをインストールすることは、NVIDIA GRID Cloud End User License Agreement の規約の遵守に同意したものと見なされます。ご使用のオペレーティングシステムの

NVIDIA GRID ドライバーのバージョンについては、NVIDIA ウェブサイトのNVIDIA® 仮想 GPU (vGPU) ソフトウェアマニュアル」を参照してください。

考慮事項

- G7e インスタンスには、Linux の場合は GRID 19.1 以降、Windows の場合は 19.4 (582.16) 以降が必要です。
- G6f および Gr6f インスタンスには GRID 18.4〜GRID 19.5 が必要です。
- G6e インスタンスには GRID 17.4 以降が必要です。
- G6 および Gr6 インスタンスには GRID 17.1 以降が必要です。
- G5 インスタンスには GRID 13.1 以降 (または GRID 12.4 以降) が必要です。
- G3 インスタンスで GRID ライセンスを機能させるには、AWS が提供する DNS 解決が必要です。
- IMDSv2 は、NVIDIA ドライバーのバージョン 14.0 以降でのみサポートされています。
- Windows インスタンスでは、カスタム Windows AMI からインスタンスを起動する場合、GRID ドライバーを動作させるには、AMI が Windows Sysprep を使用して作成された標準化されたイメージである必要があります。詳細については、「Windows Sysprep を使用して Amazon EC2 AMI を作成する」を参照してください。
- GRID 17.0 以降では、Windows Server 2019 はサポートされていません。
- GRID 14.2 以降では、Windows Server 2016 はサポートされていません。
- GRID 17.0 以降は G3 インスタンスではサポートされていません。
- Linux インスタンスの場合、NVIDIA インストーラが失敗しエラーメッセージが表示された場合、gcc などのパッケージをインストールまたは更新する必要がある場合があります。詳細は、オペレーティングシステムのバージョンとカーネルによって異なります。詳細については、NVIDIA エンタープライズサポートポータルを参照してください。

前提条件

- (Linux) AWS CLI がインスタンスにインストールされ、デフォルトの認証情報で構成されていることを確認します。詳細については、「AWS Command Line Interface ユーザーガイド」の「Installing the AWS CLI」を参照してください。
- インスタンスで AWS Tools for Windows PowerShell のデフォルトの認証情報を設定します。詳細については、AWS Tools for PowerShell ユーザーガイドのAWS Tools for Windows PowerShell の開始を参照してください。
- ユーザーまたはロールは、[AmazonS3ReadOnlyAccess] ポリシーを含む許可を持っている必要があります。

Amazon Linux 2023

インスタンスに NVIDIA GRID ドライバーをインストールするには

1. インスタンスに接続します。パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
[ec2-user ~]$ sudo dnf update -y
```

2. gcc および make をインストールします (まだインストールされていない場合)。

```
[ec2-user ~]$ sudo dnf install gcc make
```

3. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

```
[ec2-user ~]$ sudo reboot
```

4. 再起動後にインスタンスに再接続します。
5. カーネルヘッダーパッケージをインストールします。

```
[ec2-user ~]$ sudo dnf install -y kernel-devel kernel-modules-extra
```

6. 次のコマンドを使用して、GRID ドライバーインストールユーティリティをダウンロードします。

```
[ec2-user ~]$ aws s3 cp --recursive s3://ec2-linux-nvidia-drivers/latest/ .
```

GRID ドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

```
[ec2-user ~]$ aws s3 ls --recursive s3://ec2-linux-nvidia-drivers/
```

7. 次のコマンドを使用して、ドライバーのインストールを実行するアクセス権限を追加します。

```
[ec2-user ~]$ chmod +x NVIDIA-Linux-x86_64*.run
```

8. 次のようにセルフインストールスクリプトを実行して、ダウンロードした GRID ドライバーをインストールします。次に例を示します。

```
[ec2-user ~]$ sudo /bin/sh ./NVIDIA-Linux-x86_64*.run
```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

9. ドライバーが機能していることを確認します。次のコマンドのレスポンスに、インストールされた NVIDIA ドライバーバージョンおよび GPU に関する詳細が表示されます。

```
[ec2-user ~]$ nvidia-smi -q | head
```

10. G4dn、G5 または G5g インスタンスで NVIDIA vGPU ソフトウェアバージョン 14.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
[ec2-user ~]$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
[ec2-user ~]$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /
etc/modprobe.d/nvidia.conf
```

11. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

12. (オプション) ユースケースによっては、以下のオプションのステップを実行できます。この機能が必要ない場合は、以下のステップを実行しないでください。

    a. 最大 4K の解像度のディスプレイを 4 台活用するには、高性能ディスプレイプロトコル Amazon DCV を設定します。

    b. NVIDIA Quadro 仮想ワークステーションモードはデフォルトで有効になっています。RDSH アプリケーションホスティング機能用に GRID 仮想アプリケーションをアクティブ化するには、Amazon EC2 GPU ベースのインスタンスで NVIDIA GRID 仮想アプリケーションをアクティブ化するの GRID 仮想アプリケーションのアクティブ化手順を完了します。

Amazon Linux 2

インスタンスに NVIDIA GRID ドライバーをインストールするには

1. インスタンスに接続します。パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
[ec2-user ~]$ sudo yum update -y
```

2. gcc および make をインストールします (まだインストールされていない場合)。

```
[ec2-user ~]$ sudo yum install gcc make
```

3. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

```
[ec2-user ~]$ sudo reboot
```

4. 再起動後にインスタンスに再接続します。
5. 実行中のカーネルのバージョン用のカーネルヘッダーパッケージをインストールします。

```
[ec2-user ~]$ sudo yum install -y kernel-devel-$(uname -r)
```

6. 次のコマンドを使用して、GRID ドライバーインストールユーティリティをダウンロードします。

```
[ec2-user ~]$ aws s3 cp --recursive s3://ec2-linux-nvidia-drivers/latest/ .
```

GRID ドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

```
[ec2-user ~]$ aws s3 ls --recursive s3://ec2-linux-nvidia-drivers/
```

7. 次のコマンドを使用して、ドライバーのインストールを実行するアクセス権限を追加します。

```
[ec2-user ~]$ chmod +x NVIDIA-Linux-x86_64*.run
```

8. 次のようにセルフインストールスクリプトを実行して、ダウンロードした GRID ドライバーをインストールします。次に例を示します。

```
[ec2-user ~]$ sudo /bin/sh ./NVIDIA-Linux-x86_64*.run
```

Amazon Linux 2 をカーネルバージョン 5.10 で使用している場合は、次のコマンドを使用して GRID ドライバーをインストールしてください。

```
[ec2-user ~]$ sudo CC=/usr/bin/gcc10-cc ./NVIDIA-Linux-x86_64*.run
```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

9. ドライバーが機能していることを確認します。次のコマンドのレスポンスに、インストールされた NVIDIA ドライバーバージョンおよび GPU に関する詳細が表示されます。

```
[ec2-user ~]$ nvidia-smi -q | head
```

10. G4dn、G5 または G5g インスタンスで NVIDIA vGPU ソフトウェアバージョン 14.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
[ec2-user ~]$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
[ec2-user ~]$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /
etc/modprobe.d/nvidia.conf
```

11. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

12. (オプション) ユースケースによっては、以下のオプションのステップを実行できます。この機能が必要ない場合は、以下のステップを実行しないでください。

    a. 最大 4K の解像度のディスプレイを 4 台活用するには、高性能ディスプレイプロトコル Amazon DCV を設定します。

    b. NVIDIA Quadro 仮想ワークステーションモードはデフォルトで有効になっています。RDSH アプリケーションホスティング機能用に GRID 仮想アプリケーションをアクティブ化するには、Amazon EC2 GPU ベースのインスタンスで NVIDIA GRID 仮想アプリケーションをアクティブ化するの GRID 仮想アプリケーションのアクティブ化手順を完了します。

CentOS 7 と Red Hat Enterprise Linux 7

インスタンスに NVIDIA GRID ドライバーをインストールするには

1. インスタンスに接続します。パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
[ec2-user ~]$ sudo yum update -y
```

2. gcc および make をインストールします (まだインストールされていない場合)。

```
[ec2-user ~]$ sudo yum install -y gcc make
```

3. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

```
[ec2-user ~]$ sudo reboot
```

4. 再起動後にインスタンスに再接続します。
5. 現在実行しているカーネルのバージョン用のカーネルヘッダーパッケージをインストールします。

```
[ec2-user ~]$ sudo yum install -y kernel-devel-$(uname -r)
```

6. NVIDIA グラフィックスカード用の nouveau オープンソースドライバーを無効にします。

   a. nouveau ブラックリストファイルに /etc/modprobe.d/blacklist.conf を追加します。次のコードブロックをコピーして、ターミナルに貼り付けます。

   ```
   [ec2-user ~]$ cat << EOF | sudo tee --append /etc/modprobe.d/blacklist.conf
   blacklist vga16fb
   blacklist nouveau
   blacklist rivafb
   blacklist nvidiafb
   blacklist rivatv
   EOF
   ```

   b. /etc/default/grub ファイルを編集して、次の行を追加します。

   ```
   GRUB_CMDLINE_LINUX="rdblacklist=nouveau"
   ```

   c. Grub 設定を再構築します。

   ```
   [ec2-user ~]$ sudo grub2-mkconfig -o /boot/grub2/grub.cfg
   ```

7. 次のコマンドを使用して、GRID ドライバーインストールユーティリティをダウンロードします。

```
[ec2-user ~]$ aws s3 cp --recursive s3://ec2-linux-nvidia-drivers/latest/ .
```

GRID ドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

```
[ec2-user ~]$ aws s3 ls --recursive s3://ec2-linux-nvidia-drivers/
```

8. 次のコマンドを使用して、ドライバーのインストールを実行するアクセス権限を追加します。

```
[ec2-user ~]$ chmod +x NVIDIA-Linux-x86_64*.run
```

9. 次のようにセルフインストールスクリプトを実行して、ダウンロードした GRID ドライバーをインストールします。次に例を示します。

```
[ec2-user ~]$ sudo /bin/sh ./NVIDIA-Linux-x86_64*.run
```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

10. ドライバーが機能していることを確認します。次のコマンドのレスポンスに、インストールされた NVIDIA ドライバーバージョンおよび GPU に関する詳細が表示されます。

```
[ec2-user ~]$ nvidia-smi -q | head
```

11. G4dn、G5 または G5g インスタンスで NVIDIA vGPU ソフトウェアバージョン 14.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
[ec2-user ~]$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
[ec2-user ~]$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /
etc/modprobe.d/nvidia.conf
```

12. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

13. (オプション) ユースケースによっては、以下のオプションのステップを実行できます。この機能が必要ない場合は、以下のステップを実行しないでください。

a. 最大 4K の解像度のディスプレイを 4 台活用するには、高性能ディスプレイプロトコル Amazon DCV を設定します。

b. NVIDIA Quadro 仮想ワークステーションモードはデフォルトで有効になっています。RDSH アプリケーションホスティング機能用に GRID 仮想アプリケーションをアクティブ化するには、Amazon EC2 GPU ベースのインスタンスで NVIDIA GRID 仮想アプリケーションをアクティブ化するの GRID 仮想アプリケーションのアクティブ化手順を完了します。

c. GUI デスクトップ/ワークステーションパッケージをインストールします。

```
[ec2-user ~]$ sudo yum groupinstall -y "Server with GUI"
```

CentOS Stream 8 と Red Hat Enterprise Linux 8

インスタンスに NVIDIA GRID ドライバーをインストールするには

1. インスタンスに接続します。パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
[ec2-user ~]$ sudo yum update -y
```

2. gcc および make をインストールします (まだインストールされていない場合)。

```
[ec2-user ~]$ sudo yum install -y gcc make
```

3. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

```
[ec2-user ~]$ sudo reboot
```

4. 再起動後にインスタンスに再接続します。
5. 現在実行しているカーネルのバージョン用のカーネルヘッダーパッケージをインストールします。

```
[ec2-user ~]$ sudo dnf install -y elfutils-libelf-devel libglvnd-devel kernel-
devel-$(uname -r)
```

6. 次のコマンドを使用して、GRID ドライバーインストールユーティリティをダウンロードします。

```
[ec2-user ~]$ aws s3 cp --recursive s3://ec2-linux-nvidia-drivers/latest/ .
```

GRID ドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

```
[ec2-user ~]$ aws s3 ls --recursive s3://ec2-linux-nvidia-drivers/
```

7. 次のコマンドを使用して、ドライバーのインストールを実行するアクセス権限を追加します。

```
[ec2-user ~]$ chmod +x NVIDIA-Linux-x86_64*.run
```

8. 次のようにセルフインストールスクリプトを実行して、ダウンロードした GRID ドライバーをインストールします。次に例を示します。

```
[ec2-user ~]$ sudo /bin/sh ./NVIDIA-Linux-x86_64*.run
```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

9. ドライバーが機能していることを確認します。次のコマンドのレスポンスに、インストールされた NVIDIA ドライバーバージョンおよび GPU に関する詳細が表示されます。

```
[ec2-user ~]$ nvidia-smi -q | head
```

10. G4dn、G5 または G5g インスタンスで NVIDIA vGPU ソフトウェアバージョン 14.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
[ec2-user ~]$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
[ec2-user ~]$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /
etc/modprobe.d/nvidia.conf
```

11. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

12. (オプション) ユースケースによっては、以下のオプションのステップを実行できます。この機能が必要ない場合は、以下のステップを実行しないでください。

a. 最大 4K の解像度のディスプレイを 4 台活用するには、高性能ディスプレイプロトコル Amazon DCV を設定します。

b. NVIDIA Quadro 仮想ワークステーションモードはデフォルトで有効になっています。RDSH アプリケーションホスティング機能用に GRID 仮想アプリケーションをアクティブ化するには、Amazon EC2 GPU ベースのインスタンスで NVIDIA GRID 仮想アプリケーションをアクティブ化するの GRID 仮想アプリケーションのアクティブ化手順を完了します。

c. GUI ワークステーションパッケージをインストールします。

```
[ec2-user ~]$ sudo dnf groupinstall -y workstation
```

Rocky Linux 8

Linux インスタンスに NVIDIA GRID ドライバーをインストールするには

1. インスタンスに接続します。パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
[ec2-user ~]$ sudo yum update -y
```

2. gcc および make をインストールします (まだインストールされていない場合)。

```
[ec2-user ~]$ sudo yum install -y gcc make
```

3. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

```
[ec2-user ~]$ sudo reboot
```

4. 再起動後にインスタンスに再接続します。
5. 現在実行しているカーネルのバージョン用のカーネルヘッダーパッケージをインストールします。

```
[ec2-user ~]$ sudo dnf install -y elfutils-libelf-devel libglvnd-devel kernel-
devel-$(uname -r)
```

6. 次のコマンドを使用して、GRID ドライバーインストールユーティリティをダウンロードします。

```
[ec2-user ~]$ aws s3 cp --recursive s3://ec2-linux-nvidia-drivers/latest/ .
```

GRID ドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

```
[ec2-user ~]$ aws s3 ls --recursive s3://ec2-linux-nvidia-drivers/
```

7. 次のコマンドを使用して、ドライバーのインストールを実行するアクセス権限を追加します。

```
[ec2-user ~]$ chmod +x NVIDIA-Linux-x86_64*.run
```

8. 次のようにセルフインストールスクリプトを実行して、ダウンロードした GRID ドライバーをインストールします。次に例を示します。

```
[ec2-user ~]$ sudo /bin/sh ./NVIDIA-Linux-x86_64*.run
```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

9. ドライバーが機能していることを確認します。次のコマンドのレスポンスに、インストールされた NVIDIA ドライバーバージョンおよび GPU に関する詳細が表示されます。

```
[ec2-user ~]$ nvidia-smi -q | head
```

10. G4dn、G5 または G5g インスタンスで NVIDIA vGPU ソフトウェアバージョン 14.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
[ec2-user ~]$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
[ec2-user ~]$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /
etc/modprobe.d/nvidia.conf
```

11. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

12. (オプション) ユースケースによっては、以下のオプションのステップを実行できます。この機能が必要ない場合は、以下のステップを実行しないでください。

a. 最大 4K の解像度のディスプレイを 4 台活用するには、高性能ディスプレイプロトコル Amazon DCV を設定します。

b. NVIDIA Quadro 仮想ワークステーションモードはデフォルトで有効になっています。RDSH アプリケーションホスティング機能用に GRID 仮想アプリケーションをアクティブ化するには、Amazon EC2 GPU ベースのインスタンスで NVIDIA GRID 仮想アプリケーションをアクティブ化するの GRID 仮想アプリケーションのアクティブ化手順を完了します。

Ubuntu と Debian

インスタンスに NVIDIA GRID ドライバーをインストールするには

1. インスタンスに接続します。パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
$ sudo apt-get update -y
```

2. gcc および make をインストールします (まだインストールされていない場合)。

```
$ sudo apt-get install -y gcc make
```

3. (Ubuntu) 最新バージョンを受け取るには、`linux-aws` パッケージをアップグレードします。

```
$ sudo apt-get upgrade -y linux-aws
```

(Debian) 最新バージョンを受け取るには、パッケージをアップグレードします。

```
$ sudo apt-get upgrade -y
```

4. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

```
$ sudo reboot
```

5. 再起動後にインスタンスに再接続します。

6. 現在実行しているカーネルのバージョン用のカーネルヘッダーパッケージをインストールします。

```
$ sudo apt-get install -y linux-headers-$(uname -r) linux-modules-extra-$(uname -r)
```

7. NVIDIA グラフィックスカード用の nouveau オープンソースドライバーを無効にします。

   a. nouveau ブラックリストファイルに /etc/modprobe.d/blacklist.conf を追加します。次のコードブロックをコピーして、ターミナルに貼り付けます。

   ```
   $ cat << EOF | sudo tee --append /etc/modprobe.d/blacklist.conf
   blacklist vga16fb
   blacklist nouveau
   blacklist rivafb
   blacklist nvidiafb
   blacklist rivatv
   EOF
   ```

   b. /etc/default/grub ファイルを編集して、次の行を追加します。

   ```
   GRUB_CMDLINE_LINUX="rdblacklist=nouveau"
   ```

   c. Grub 設定を再構築します。

   ```
   $ sudo update-grub
   ```

8. 次のコマンドを使用して、GRID ドライバーインストールユーティリティをダウンロードします。

   ```
   $ aws s3 cp --recursive s3://ec2-linux-nvidia-drivers/latest/ .
   ```

   GRID ドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

   ```
   $ aws s3 ls --recursive s3://ec2-linux-nvidia-drivers/
   ```

9. 次のコマンドを使用して、ドライバーのインストールを実行するアクセス権限を追加します。

   ```
   $ chmod +x NVIDIA-Linux-x86_64*.run
   ```

10. 次のようにセルフインストールスクリプトを実行して、ダウンロードした GRID ドライバーをインストールします。次に例を示します。

```
$ sudo /bin/sh ./NVIDIA-Linux-x86_64*.run
```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

11. ドライバーが機能していることを確認します。次のコマンドのレスポンスに、インストールされた NVIDIA ドライバーバージョンおよび GPU に関する詳細が表示されます。

```
$ nvidia-smi -q | head
```

12. G4dn、G5 または G5g インスタンスで NVIDIA vGPU ソフトウェアバージョン 14.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /etc/
modprobe.d/nvidia.conf
```

13. インスタンスを再起動します。

```
$ sudo reboot
```

14. (オプション) ユースケースによっては、以下のオプションのステップを実行できます。この機能が必要ない場合は、以下のステップを実行しないでください。

    a. 最大 4K の解像度のディスプレイを 4 台活用するには、高性能ディスプレイプロトコル Amazon DCV を設定します。

    b. NVIDIA Quadro 仮想ワークステーションモードはデフォルトで有効になっています。RDSH アプリケーションホスティング機能用に GRID 仮想アプリケーションをアクティブ化するには、Amazon EC2 GPU ベースのインスタンスで NVIDIA GRID 仮想アプリケーションをアクティブ化するの GRID 仮想アプリケーションのアクティブ化手順を完了します。

    c. GUI デスクトップ/ワークステーションパッケージをインストールします。

```
$ sudo apt-get install -y lightdm ubuntu-desktop
```

Windows オペレーティングシステム

Windows インスタンスに NVIDIA GRID ドライバーをインストールするには

1. Windows インスタンスに接続し、PowerShell ウィンドウを開きます。
2. 次の PowerShell コマンドを使用して、ドライバーと NVIDIA GRID Cloud End User License Agreement を Amazon S3 からデスクトップにダウンロードします。

```
$Bucket = "ec2-windows-nvidia-drivers"
$KeyPrefix = "latest"
$LocalPath = "$home\Desktop\NVIDIA"
$Objects = Get-S3Object -BucketName $Bucket -KeyPrefix $KeyPrefix -Region us-east-1
foreach ($Object in $Objects) {
$LocalFileName = $Object.Key
if ($LocalFileName -ne '' -and $Object.Size -ne 0) {
$LocalFilePath = Join-Path $LocalPath $LocalFileName
Copy-S3Object -BucketName $Bucket -Key $Object.Key -LocalFile $LocalFilePath -
Region us-east-1
}
}
```

   NVIDIA GRID ドライバの複数のバージョンがこのバケットに保存されます。`-KeyPrefix $KeyPrefix` オプションを削除すると、使用可能なすべての Windows バージョンをバケットにダウンロードできます。ご使用のオペレーティングシステムの NVIDIA GRID ドライバーのバージョンについては、NVIDIA ウェブサイトのNVIDIA® 仮想 GPU (vGPU) ソフトウェアマニュアル」を参照してください。

   GRID バージョン 11.0 以降では、G3 インスタンスと G4dn インスタンスの両方で `latest` のドライバーを使用できます。11.0 より後のバージョンは `g4/latest` に追加されませんが、バージョン 11.0 および G4dn に固有の以前のバージョンは `g4/latest` に保持されます。

   G5 インスタンスには GRID 13.1 以降 (または GRID 12.4 以降) が必要です。
3. デスクトップに移動し、インストールファイルをダブルクリックして起動します (インスタンスの OS バージョンに該当するドライバーバージョンを選択してください)。ドライバーをインストールする手順にしたがい、必要に応じてインスタンスを再起動します。GPU が正しく動作していることを確認するには、デバイスマネージャーをチェックします。
4. (オプション) 次のコマンドを使用してコントロールパネルでライセンスページを無効化し、ユーザーが誤ってプロダクトキーを変更することを回避します (NVIDIA GRID 仮想ワークステーショ

ンはデフォルトで有効になっています)。詳細については、GRID Licensing User Guide を参照してください。

PowerShell

次の PowerShell コマンドを実行してレジストリ値を作成し、コントロールパネルのライセンスページを無効にします。デフォルトでは、AWS Windows AMI の AWS Tools for PowerShell は 32 ビットバージョンであり、このコマンドは失敗します。代わりに、オペレーティングシステムに付属する 64 ビットバージョンの PowerShell を使用してください。

```
New-Item -Path "HKLM:\SOFTWARE\NVIDIA Corporation\Global" -Name GridLicensing
New-ItemProperty -Path "HKLM:\SOFTWARE\NVIDIA Corporation\Global\GridLicensing" -
Name "NvCplDisableManageLicensePage" -PropertyType "DWord" -Value "1"
```

コマンドプロンプト

次のレジストリコマンドを実行してレジストリ値を作成し、コントロールパネルのライセンスページを無効にします。コマンドプロンプトウィンドウまたは 64 ビットバージョンの PowerShell を使用してコマンドを実行できます。

```
reg add "HKLM\SOFTWARE\NVIDIA Corporation\Global\GridLicensing" /v
 NvCplDisableManageLicensePage /t REG_DWORD /d 1
```

5. (オプション) ユースケースによっては、以下のオプションのステップを実行できます。この機能が必要ない場合は、以下のステップを実行しないでください。

   a. 最大 4K の解像度のディスプレイを 4 台活用するには、高性能ディスプレイプロトコル Amazon DCV を設定します。

   b. NVIDIA Quadro 仮想ワークステーションモードはデフォルトで有効になっています。RDSH アプリケーションホスティング機能用に GRID 仮想アプリケーションをアクティブ化するには、Amazon EC2 GPU ベースのインスタンスで NVIDIA GRID 仮想アプリケーションをアクティブ化するの GRID 仮想アプリケーションのアクティブ化手順を完了します。

NVIDIA ゲームドライバー (G7e、G6、G6e、G5、G4dn インスタンス)のインストール

これらのドライバーは、AWS カスタマーのみが利用できます。ダウンロードすることにより、ダウンロードしたソフトは RTX PRO 6000 Blackwell、NVIDIA L4、NVIDIA L40S、NVIDIA A10G、NVIDIA Tesla T4、NVIDIA Tesla M60 のハードウェアに使用する AMI のみを開発するために使用することに同意することになります。GRID ドライバーを使用して、AWS 環境内で AMI の作成および使用の両方ができます。このソフトウェアをインストールすることは、NVIDIA GRID Cloud End User License Agreement の規約の遵守に同意したものと見なされます。

考慮事項

- G3 インスタンスで GRID ライセンスを機能させるには、AWS が提供する DNS 解決が必要です。
- IMDSv2は、NVIDIA ドライバーのバージョン 495.x 以降でのみサポートされています。

前提条件

- (Linux) AWS CLI がインスタンスにインストールされ、デフォルトの認証情報で構成されていることを確認します。詳細については、「AWS Command Line Interface ユーザーガイド」の「Installing the AWS CLI」を参照してください。
- ユーザーまたはロールは、[AmazonS3ReadOnlyAccess] ポリシーを含む許可を持っている必要があります。

Amazon Linux 2023

インスタンスに NVIDIA ゲームドライバーをインストールするには

1. インスタンスに接続します。パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
[ec2-user ~]$ sudo dnf update -y
```

2. gcc および make をインストールします (まだインストールされていない場合)。

```
[ec2-user ~]$ sudo dnf install gcc make
```

3. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

```
[ec2-user ~]$ sudo reboot
```

4. 再起動後にインスタンスに再接続します。
5. カーネルヘッダーパッケージをインストールします。

```
[ec2-user ~]$ sudo dnf install -y kernel-devel kernel-modules-extra kernel-devel-
$(uname -r) kernel-headers-$(uname -r) dkms
```

6. 次のコマンドを使用して、ゲームドライバーインストールユーティリティをダウンロードします。

```
[ec2-user ~]$ aws s3 cp --recursive s3://nvidia-gaming/linux/latest/ .
```

ゲームドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

```
[ec2-user ~]$ aws s3 ls --recursive s3://nvidia-gaming/linux/
```

7. ダウンロードした .zip アーカイブから、ゲームドライバーのインストールユーティリティを抽出します。

```
[ec2-user ~]$ unzip latest-driver-name.zip -d nvidia-drivers
```

8. 次のコマンドを使用して、ドライバーのインストールユーティリティを実行するためのアクセス許可を追加します。

```
[ec2-user ~]$ chmod +x nvidia-drivers/NVIDIA-Linux-x86_64*-grid.run
```

9. 次のコマンドを使用してインストーラを実行します。

```
[ec2-user ~]$ sudo ./nvidia-drivers/NVIDIA-Linux-x86_64*.run
```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

10. 次のコマンドを使用して必要な設定ファイルを作成します。

```
[ec2-user ~]$ cat << EOF | sudo tee -a /etc/nvidia/gridd.conf
vGamingMarketplace=2
```

```
EOF
```

11. 次のコマンドを使用して認証ファイルをダウンロードし、名前を変更します。
    - バージョン 590.48 以降:

      ```
      [ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert_2026_03_02.cert"
      ```

    - バージョン 575.64 から 580.105.08:

      ```
      [ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCertLinux_2025_07_01.cert"
      ```

    - バージョン 460.39 から 575.57:

      ```
      [ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCertLinux_2024_02_22.cert"
      ```

    - バージョン 440.68 から 445.48:

      ```
      [ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2020_04.cert"
      ```

    - それより前のバージョン:

      ```
      [ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2019_09.cert"
      ```

12. G4dn、G5 または G5g インスタンスで NVIDIA ドライバーバージョン 510.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

    ```
    [ec2-user ~]$ sudo touch /etc/modprobe.d/nvidia.conf
    ```

    ```
    [ec2-user ~]$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /etc/modprobe.d/nvidia.conf
    ```

13. インスタンスを再起動します。

    ```
    [ec2-user ~]$ sudo reboot
    ```

14. 次のコマンドを使用して、NVIDIA のゲーミングライセンスを検証します。

```
[ec2-user ~]$ nvidia-smi.exe -q
```

出力内で vGPU Software Licensed Product を検索します。

15. (オプション) 最大 4K 解像度の 1 台のディスプレイを活用するには、高性能ディスプレイプロトコル Amazon DCV を設定します。

Amazon Linux 2

インスタンスに NVIDIA ゲームドライバーをインストールするには

1. インスタンスに接続します。パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
[ec2-user ~]$ sudo yum update -y
```

2. gcc および make をインストールします (まだインストールされていない場合)。

```
[ec2-user ~]$ sudo yum install gcc make
```

3. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

```
[ec2-user ~]$ sudo reboot
```

4. 再起動後にインスタンスに再接続します。
5. 現在実行しているカーネルのバージョン用のカーネルヘッダーパッケージをインストールします。

```
[ec2-user ~]$ sudo yum install -y kernel-devel-$(uname -r)
```

6. 次のコマンドを使用して、ゲームドライバーインストールユーティリティをダウンロードします。

```
[ec2-user ~]$ aws s3 cp --recursive s3://nvidia-gaming/linux/latest/ .
```

ゲームドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

```
[ec2-user ~]$ aws s3 ls --recursive s3://nvidia-gaming/linux/
```

7. ダウンロードした .zip アーカイブから、ゲームドライバーのインストールユーティリティを抽出します。

```
[ec2-user ~]$ unzip latest-driver-name.zip -d nvidia-drivers
```

8. 次のコマンドを使用して、ドライバーのインストールユーティリティを実行するためのアクセス許可を追加します。

```
[ec2-user ~]$ chmod +x nvidia-drivers/NVIDIA-Linux-x86_64*-grid.run
```

9. 次のコマンドを使用してインストーラを実行します。

```
[ec2-user ~]$ sudo ./nvidia-drivers/NVIDIA-Linux-x86_64*.run
```

Amazon Linux 2 をカーネルバージョン 5.10 で使用している場合は、次のコマンドを使用して NVIDIA ゲームドライバーをインストールします。

```
[ec2-user ~]$ sudo CC=/usr/bin/gcc10-cc ./NVIDIA-Linux-x86_64*.run
```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

10. 次のコマンドを使用して必要な設定ファイルを作成します。

```
[ec2-user ~]$ cat << EOF | sudo tee -a /etc/nvidia/gridd.conf
vGamingMarketplace=2
EOF
```

11. 次のコマンドを使用して認証ファイルをダウンロードし、名前を変更します。

- バージョン 590.48 以降:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert_2026_03_02.cert"
```

- バージョン 575.64 から 580.105.08:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCertLinux_2025_07_01.cert"
```

- バージョン 460.39 から 575.57:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCertLinux_2024_02_22.cert"
```

- バージョン 440.68 から 445.48:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2020_04.cert"
```

- それより前のバージョン:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2019_09.cert"
```

12. G4dn、G5 または G5g インスタンスで NVIDIA ドライバーバージョン 510.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
[ec2-user ~]$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
[ec2-user ~]$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /
etc/modprobe.d/nvidia.conf
```

13. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

14. 次のコマンドを使用して、NVIDIA のゲーミングライセンスを検証します。

```
[ec2-user ~]$ nvidia-smi.exe -q
```

出力内で `vGPU Software Licensed Product` を検索します。

15. (オプション) 最大 4K 解像度の 1 台のディスプレイを活用するには、高性能ディスプレイプロトコル Amazon DCV を設定します。

CentOS 7 と Red Hat Enterprise Linux 7

インスタンスに NVIDIA ゲームドライバーをインストールするには

1. Linux インスタンスに接続します。gcc および make をインストールします (まだインストールされていない場合)。

```
[ec2-user ~]$ sudo yum install -y gcc make
```

2. パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
[ec2-user ~]$ sudo yum update -y
```

3. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

```
[ec2-user ~]$ sudo reboot
```

4. 再起動後にインスタンスに再接続します。
5. 現在実行しているカーネルのバージョン用のカーネルヘッダーパッケージをインストールします。

```
[ec2-user ~]$ sudo yum install -y unzip kernel-devel-$(uname -r)
```

6. NVIDIA グラフィックスカード用の nouveau オープンソースドライバーを無効にします。

   a. nouveau ブラックリストファイルに /etc/modprobe.d/blacklist.conf を追加します。次のコードブロックをコピーして、ターミナルに貼り付けます。

```
[ec2-user ~]$ cat << EOF | sudo tee --append /etc/modprobe.d/blacklist.conf
blacklist vga16fb
blacklist nouveau
blacklist rivafb
blacklist nvidiafb
blacklist rivatv
EOF
```

   b. /etc/default/grub ファイルを編集して、次の行を追加します。

```
GRUB_CMDLINE_LINUX="rdblacklist=nouveau"
```

   c. Grub 設定を再構築します。

```
[ec2-user ~]$ sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

7. 次のコマンドを使用して、ゲームドライバーインストールユーティリティをダウンロードします。

   ```
   [ec2-user ~]$ aws s3 cp --recursive s3://nvidia-gaming/linux/latest/ .
   ```

   ゲームドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

   ```
   [ec2-user ~]$ aws s3 ls --recursive s3://nvidia-gaming/linux/
   ```

8. ダウンロードした .zip アーカイブから、ゲームドライバーのインストールユーティリティを抽出します。

   ```
   [ec2-user ~]$ unzip *Gaming-Linux-Guest-Drivers.zip -d nvidia-drivers
   ```

9. 次のコマンドを使用して、ドライバーのインストールユーティリティを実行するためのアクセス許可を追加します。

   ```
   [ec2-user ~]$ chmod +x nvidia-drivers/NVIDIA-Linux-x86_64*-grid.run
   ```

10. 次のコマンドを使用してインストーラを実行します。

    ```
    [ec2-user ~]$ sudo nvidia-drivers/NVIDIA-Linux-x86_64*.run
    ```

    プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

11. 次のコマンドを使用して必要な設定ファイルを作成します。

    ```
    [ec2-user ~]$ cat << EOF | sudo tee -a /etc/nvidia/gridd.conf
    vGamingMarketplace=2
    EOF
    ```

12. 次のコマンドを使用して認証ファイルをダウンロードし、名前を変更します。

    - バージョン 590.48 以降:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert_2026_03_02.cert"
```

- バージョン 575.64 から 580.105.08:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCertLinux_2025_07_01.cert"
```

- バージョン 460.39 から 575.57:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCertLinux_2024_02_22.cert"
```

- バージョン 440.68 から 445.48:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2020_04.cert"
```

- それより前のバージョン:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2019_09.cert"
```

13. G4dn、G5 または G5g インスタンスで NVIDIA ドライバーバージョン 510.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
[ec2-user ~]$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
[ec2-user ~]$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /
etc/modprobe.d/nvidia.conf
```

14. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

15. (オプション) 最大 4K 解像度の 1 台のディスプレイを活用するには、高性能ディスプレイプロトコル Amazon DCV を設定します。この機能が必要ない場合は、このステップを実行しないでください。

CentOS Stream 8 と Red Hat Enterprise Linux 8

インスタンスに NVIDIA ゲームドライバーをインストールするには

1. Linux インスタンスに接続します。gcc および make をインストールします (まだインストールされていない場合)。

```
[ec2-user ~]$ sudo yum install -y gcc make
```

2. パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
[ec2-user ~]$ sudo yum update -y
```

3. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

```
[ec2-user ~]$ sudo reboot
```

4. 再起動後にインスタンスに再接続します。
5. 現在実行しているカーネルのバージョン用のカーネルヘッダーパッケージをインストールします。

```
[ec2-user ~]$ sudo yum install -y unzip kernel-devel-$(uname -r)
```

6. 次のコマンドを使用して、ゲームドライバーインストールユーティリティをダウンロードします。

```
[ec2-user ~]$ aws s3 cp --recursive s3://nvidia-gaming/linux/latest/ .
```

ゲームドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

```
[ec2-user ~]$ aws s3 ls --recursive s3://nvidia-gaming/linux/
```

7. ダウンロードした .zip アーカイブから、ゲームドライバーのインストールユーティリティを抽出します。

```
[ec2-user ~]$ unzip *Gaming-Linux-Guest-Drivers.zip -d nvidia-drivers
```

8. 次のコマンドを使用して、ドライバーのインストールユーティリティを実行するためのアクセス許可を追加します。

```
[ec2-user ~]$ chmod +x nvidia-drivers/NVIDIA-Linux-x86_64*-grid.run
```

9. 次のコマンドを使用してインストーラを実行します。

```
[ec2-user ~]$ sudo nvidia-drivers/NVIDIA-Linux-x86_64*.run
```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

10. 次のコマンドを使用して必要な設定ファイルを作成します。

```
[ec2-user ~]$ cat << EOF | sudo tee -a /etc/nvidia/gridd.conf
vGamingMarketplace=2
EOF
```

11. 次のコマンドを使用して認証ファイルをダウンロードし、名前を変更します。

- バージョン 590.48 以降:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert_2026_03_02.cert"
```

- バージョン 575.64 から 580.105.08:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCertLinux_2025_07_01.cert"
```

- バージョン 460.39 から 575.57:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCertLinux_2024_02_22.cert"
```

- バージョン 440.68 から 445.48:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2020_04.cert"
```

- それより前のバージョン:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2019_09.cert"
```

12. G4dn、G5 または G5g インスタンスで NVIDIA ドライバーバージョン 510.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
[ec2-user ~]$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
[ec2-user ~]$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /
etc/modprobe.d/nvidia.conf
```

13. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

14. (オプション) 最大 4K 解像度の 1 台のディスプレイを活用するには、高性能ディスプレイプロトコル Amazon DCV を設定します。

Rocky Linux 8

インスタンスに NVIDIA ゲームドライバーをインストールするには

1. Linux インスタンスに接続します。gcc および make をインストールします (まだインストールされていない場合)。

```
[ec2-user ~]$ sudo yum install -y gcc make
```

2. パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
[ec2-user ~]$ sudo yum update -y
```

3. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

```
[ec2-user ~]$ sudo reboot
```

4. 再起動後にインスタンスに再接続します。
5. 現在実行しているカーネルのバージョン用のカーネルヘッダーパッケージをインストールします。

```
[ec2-user ~]$ sudo dnf install -y unzip elfutils-libelf-devel libglvnd-devel
 kernel-devel-$(uname -r)
```

6. 次のコマンドを使用して、ゲームドライバーインストールユーティリティをダウンロードします。

```
[ec2-user ~]$ aws s3 cp --recursive s3://nvidia-gaming/linux/latest/ .
```

ゲームドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

```
[ec2-user ~]$ aws s3 ls --recursive s3://nvidia-gaming/linux/
```

7. ダウンロードした .zip アーカイブから、ゲームドライバーのインストールユーティリティを抽出します。

```
[ec2-user ~]$ unzip *Gaming-Linux-Guest-Drivers.zip -d nvidia-drivers
```

8. 次のコマンドを使用して、ドライバーのインストールユーティリティを実行するためのアクセス許可を追加します。

```
[ec2-user ~]$ chmod +x nvidia-drivers/NVIDIA-Linux-x86_64*-grid.run
```

9. 次のコマンドを使用してインストーラを実行します。

```
[ec2-user ~]$ sudo nvidia-drivers/NVIDIA-Linux-x86_64*.run
```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

10. 次のコマンドを使用して必要な設定ファイルを作成します。

```
[ec2-user ~]$ cat << EOF | sudo tee -a /etc/nvidia/gridd.conf
vGamingMarketplace=2
EOF
```

11. 次のコマンドを使用して認証ファイルをダウンロードし、名前を変更します。

- バージョン 590.48 以降:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert_2026_03_02.cert"
```

- バージョン 575.64 から 580.105.08:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCertLinux_2025_07_01.cert"
```

- バージョン 460.39 から 575.57:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCertLinux_2024_02_22.cert"
```

- バージョン 440.68 から 445.48:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2020_04.cert"
```

- それより前のバージョン:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2019_09.cert"
```

12. G4dn、G5 または G5g インスタンスで NVIDIA ドライバーバージョン 510.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
[ec2-user ~]$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
[ec2-user ~]$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /
etc/modprobe.d/nvidia.conf
```

13. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

14. (オプション) 最大 4K 解像度の 1 台のディスプレイを活用するには、高性能ディスプレイプロトコル Amazon DCV を設定します。

Ubuntu と Debian

インスタンスに NVIDIA ゲームドライバーをインストールするには

1. Linux インスタンスに接続します。gcc および make をインストールします (まだインストールされていない場合)。

```
$ sudo apt-get install -y gcc make build-essential
```

2. パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
$ sudo apt-get update -y
```

3. 最新バージョンにするには、linux-aws パッケージにアップグレードします。

```
$ sudo apt-get upgrade -y linux-aws
```

4. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

```
$ sudo reboot
```

5. 再起動後にインスタンスに再接続します。
6. 現在実行しているカーネルのバージョン用のカーネルヘッダーパッケージをインストールします。

```
$ sudo apt install -y unzip dkms linux-headers-$(uname -r)
```

7. NVIDIA グラフィックスカード用の nouveau オープンソースドライバーを無効にします。

   a. nouveau ブラックリストファイルに /etc/modprobe.d/blacklist.conf を追加します。次のコードブロックをコピーして、ターミナルに貼り付けます。

```
$ cat << EOF | sudo tee --append /etc/modprobe.d/blacklist.conf
blacklist vga16fb
blacklist nouveau
blacklist rivafb
blacklist nvidiafb
blacklist rivatv
EOF
```

   b. /etc/default/grub ファイルを編集して、次の行を追加します。

```
GRUB_CMDLINE_LINUX="rdblacklist=nouveau"
```

c. Grub 設定を再構築します。

```
$ sudo update-grub
```

8. 次のコマンドを使用して、ゲームドライバーインストールユーティリティをダウンロードします。

```
$ aws s3 cp --recursive s3://nvidia-gaming/linux/latest/ .
```

ゲームドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

```
$ aws s3 ls --recursive s3://nvidia-gaming/linux/
```

9. ダウンロードした .zip アーカイブから、ゲームドライバーのインストールユーティリティを抽出します。

```
$ unzip *Gaming-Linux-Guest-Drivers.zip -d nvidia-drivers
```

10. 次のコマンドを使用して、ドライバーのインストールユーティリティを実行するためのアクセス許可を追加します。

```
$ chmod +x nvidia-drivers/NVIDIA-Linux-x86_64*-grid.run
```

11. 次のコマンドを使用してインストーラを実行します。

```
$ sudo nvidia-drivers/NVIDIA-Linux-x86_64*.run
```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

12. 次のコマンドを使用して必要な設定ファイルを作成します。

```
$ cat << EOF | sudo tee -a /etc/nvidia/gridd.conf
vGamingMarketplace=2
EOF
```

13. 次のコマンドを使用して認証ファイルをダウンロードし、名前を変更します。

- バージョン 590.48 以降:

```
$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert_2026_03_02.cert"
```

- バージョン 575.64 から 580.105.08:

```
$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCertLinux_2025_07_01.cert"
```

- バージョン 460.39 から 575.57:

```
$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCertLinux_2024_02_22.cert"
```

- バージョン 440.68 から 445.48:

```
$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2020_04.cert"
```

- それより前のバージョン:

```
$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2019_09.cert"
```

14. G4dn、G5 または G5g インスタンスで NVIDIA ドライバーバージョン 510.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /etc/
modprobe.d/nvidia.conf
```

15. インスタンスを再起動します。

```
$ sudo reboot
```

16. (オプション) 最大 4K 解像度の 1 台のディスプレイを活用するには、高性能ディスプレイプロトコル Amazon DCV を設定します。この機能が必要ない場合は、このステップを実行しないでください。

Windows オペレーティングシステム

インスタンスに NVIDIA ゲームドライバーをインストールする前に、すべてのゲームドライバーについて説明した考慮事項に加えて、次の前提条件が満たされていることを確認する必要があります。

- カスタム Windows AMI を使用して Windows インスタンスを起動する場合、ゲームドライバーを動作させるには、AMI が Windows Sysprep を使用して作成された標準化されたイメージである必要があります。詳細については、「Windows Sysprep を使用して Amazon EC2 AMI を作成する」を参照してください。
- Windows インスタンスで AWS Tools for Windows PowerShell のデフォルトの認証情報を設定します。詳細については、AWS Tools for Windows PowerShell ユーザーガイドのAWS Tools for PowerShell の使用開始を参照してください。

Windows インスタンスに NVIDIA ゲームドライバーをインストールするには

1. Windows インスタンスに接続し、PowerShell ウィンドウを開きます。
2. 次の PowerShell コマンドを使用して、ゲームドライバーをダウンロードしてインストールします。

```
$Bucket = "nvidia-gaming"
$KeyPrefix = "windows/latest"
$LocalPath = "$home\Desktop\NVIDIA"
$Objects = Get-S3Object -BucketName $Bucket -KeyPrefix $KeyPrefix -Region us-east-1
foreach ($Object in $Objects) {
$LocalFileName = $Object.Key
if ($LocalFileName -ne '' -and $Object.Size -ne 0) {
$LocalFilePath = Join-Path $LocalPath $LocalFileName
Copy-S3Object -BucketName $Bucket -Key $Object.Key -LocalFile $LocalFilePath -
Region us-east-1
}
}
```

NVIDIA GRID ドライバーの複数のバージョンがこの S3 バケットに保存されます。$KeyPrefix 変数の値を「windows/latest」から「windows」に変更すると、バケット内の使用可能なすべてのバージョンをダウンロードできます。

3. デスクトップに移動し、インストールファイルをダブルクリックして起動します (インスタンスの OS バージョンに該当するドライバーバージョンを選択してください)。ドライバーをインストールする手順にしたがい、必要に応じてインスタンスを再起動します。GPU が正しく動作していることを確認するには、デバイスマネージャーをチェックします。
4. ドライバーを登録するには、以下のいずれかの方法を使用します。

   Version 527.27 or above

   64 ビットバージョンの PowerShell または コマンドプロンプトウィンドウで次のレジストリキーを作成します。

   キー: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\nvlddmkm\Global

   名前: vGamingMarketplace

   タイプ: DWord

   値: 2

   PowerShell

   このレジストリ値を作成するには、次の PowerShell コマンドを実行します。デフォルトでは、AWS Windows AMI の AWS Tools for PowerShell は 32 ビットバージョンであり、このコマンドは失敗します。代わりに、オペレーティングシステムに付属する 64 ビットバージョンの PowerShell を使用してください。

   ```
   New-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Services\nvlddmkm\Global"
    -Name "vGamingMarketplace" -PropertyType "DWord" -Value "2"
   ```

   コマンドプロンプト

   このレジストリ値を作成するには、次のレジストリコマンドを実行します。コマンドプロンプトウィンドウまたは 64 ビットバージョンの PowerShell を使用してコマンドを実行できます。

```
reg add "HKLM\SYSTEM\CurrentControlSet\Services\nvlddmkm\Global" /v
 vGamingMarketplace /t REG_DWORD /d 2
```

Earlier versions

64 ビットバージョンの PowerShell または コマンドプロンプトウィンドウで次のレジストリキーを作成します。

キー: HKEY_LOCAL_MACHINE\SOFTWARE\NVIDIA Corporation\Global

名前: vGamingMarketplace

タイプ: DWord

値: 2

PowerShell

このレジストリ値を作成するには、次の PowerShell コマンドを実行します。デフォルトでは、AWS Windows AMI の AWS Tools for PowerShell は 32 ビットバージョンであり、このコマンドは失敗します。代わりに、オペレーティングシステムに付属する 64 ビットバージョンの PowerShell を使用してください。

```
New-ItemProperty -Path "HKLM:\SOFTWARE\NVIDIA Corporation\Global" -Name
 "vGamingMarketplace" -PropertyType "DWord" -Value "2"
```

コマンドプロンプト

次のレジストリコマンドを実行して、コマンドプロンプトウィンドウでこのレジストリキーを作成します。このコマンドは 64 ビットバージョンの PowerShell でも使用できます。

```
reg add "HKLM\SOFTWARE\NVIDIA Corporation\Global" /v vGamingMarketplace /t
 REG_DWORD /d 2
```

5. PowerShell で、次のコマンドを実行します。これにより、証明書ファイルがダウンロードされ、ファイル GridSwCert.txt の名前が変更され、ファイルがシステムドライブのパブリックドキュメントフォルダに移動されます。通常、フォルダパスは C:\Users\Public\Documents です。

   - バージョン 591.59 以降:

```
Invoke-WebRequest -Uri "https://nvidia-gaming.s3.amazonaws.com/GridSwCert-
Archive/GridSwCert_2026_03_02.cert" -OutFile "$Env:PUBLIC\Documents
\GridSwCert.txt"
```

- バージョン 576.80 から 581.80:

```
Invoke-WebRequest -Uri "https://nvidia-gaming.s3.amazonaws.com/GridSwCert-
Archive/GridSwCertWindows_2025_07_01.cert" -OutFile "$Env:PUBLIC\Documents
\GridSwCert.txt"
```

- バージョン 460.39 から 576.72:

```
Invoke-WebRequest -Uri "https://nvidia-gaming.s3.amazonaws.com/GridSwCert-
Archive/GridSwCertWindows_2024_02_22.cert" -OutFile "$Env:PUBLIC\Documents
\GridSwCert.txt"
```

- バージョン 445.87:

```
Invoke-WebRequest -Uri "https://nvidia-gaming.s3.amazonaws.com/GridSwCert-
Archive/GridSwCert-Windows_2020_04.cert" -OutFile "$Env:PUBLIC\Documents
\GridSwCert.txt"
```

- それより前のバージョン:

```
Invoke-WebRequest -Uri "https://nvidia-gaming.s3.amazonaws.com/GridSwCert-
Archive/GridSwCert-Windows_2019_09.cert" -OutFile "$Env:PUBLIC\Documents
\GridSwCert.txt"
```

ファイルのダウンロード時にエラーが表示され、Windows Server 2016 以前のバージョンを使用している場合は、PowerShell ターミナルで TLS 1.2 を有効にする必要がある場合があります。次のコマンドで現在の PowerShell セッションの TLS 1.2 を有効にしてから、もう一度試してください。

```
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
```

6. インスタンスを再起動します。
7. インスタンス上で `nvidia-smi.exe` ファイルを見つけます。

```
Get-ChildItem -Path C:\ -Recurse -Filter "nvidia-smi.exe"
```

次のコマンドを使用して、NVIDIA のゲーミングライセンスを検証します。*##*を、前のコマンドの出力内のフォルダーの名前に置き換えます。

```
C:\Windows\System32\DriverStore\FileRepository\path\nvidia-smi.exe -q
```

出力は以下のようになります。

```
vGPU Software Licensed Product
Product Name                : NVIDIA Cloud Gaming
License Status              : Licensed (Expiry: N/A)
```

8. (オプション) 最大 4K 解像度の 1 台のディスプレイを活用するには、高性能ディスプレイプロトコル、Amazon DCV を設定します。この機能が必要ない場合は、このステップを実行しないでください。

EC2 Windows インスタンスに ENA ドライバーをインストールする

インスタンスが Amazon 提供の最新の Windows Amazon マシンイメージ (AMI) に基づいていない場合は、次の手順を実行して現在の ENA ドライバーをインスタンスにインストールします。この更新は、インスタンスを再起動できる時に実行する必要があります。インストールスクリプトがインスタンスを自動的に再起動しない場合は、最後のステップとしてインスタンスを再起動することをお勧めします。

インスタンスの実行中にインスタンスストアボリュームを使用してデータを保存した場合、そのデータはインスタンスを停止すると消去されます。インスタンスを停止する前に、必要なデータをインスタンスストアボリュームから永続的ストレージ (Amazon EBS や Amazon S3 など) にコピーしていることを確認します。

前提条件

ENA ドライバーをインストールまたはアップグレードするには、Windows インスタンスが次の前提条件を満たしている必要があります。

- PowerShell バージョン 3.0 以降がインストールされている
- このセクションに示すコマンドは、PowerShell の 64 ビットバージョンで実行する必要があります。x86 バージョンの PowerShell を使用しないでください。これはシェルの 32 ビットバージョンで、これらのコマンドではサポートされていません。

ステップ 1: データをバックアップする

[デバイスマネージャー] を通じて変更をロールバックできない場合に備えて、バックアップ AMI を作成することをお勧めします。AWS マネジメントコンソール でバックアップ AMI を作成するには、次のステップを実行します。

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. ドライバーのアップグレードが必要なインスタンスを選択し、[インスタンスの状態] メニューから [インスタンスを停止] を選択してください。
4. インスタンスを停止した後、インスタンスを再度選択してください。バックアップを作成するには、[アクション] メニューから [イメージとテンプレート] を選択し、[イメージを作成] を選択してください。
5. インスタンスを再起動するには、[インスタンスの状態] メニューから [インスタンスを開始] を選択してください。

ステップ 2: ENA ドライバーをインストールまたはアップグレードする

ENA ドライバーは、AWS Systems Manager Distributor または PowerShell コマンドレットを使用してインストールまたはアップグレードできます。詳細な手順については、使用する方法に一致するタブを選択してください。

Systems Manager Distributor

Systems Manager Distributor 機能を使用して、Systems Manager 管理されたノードにパッケージをデプロイできます。Systems Manager Distributor を使用すると、ENA ドライバーパッケージを 1 回インストールすることも、スケジュールされた更新を使用してインストールすることもできます。システムマネージャー ディストリビューター を使用して ENA ドライバーパッケージ `AwsEnaNetworkDriver`をインストールする方法の詳細については、「AWS Systems Manager ユーザーガイド」の「パッケージのインストールまたは更新」を参照してください。

PowerShell

このセクションでは、PowerShell コマンドレットを使用して ENA ドライバーパッケージをインスタンスにダウンロードしてインストールする方法について説明します。

オプション 1: 最新バージョンをダウンロードして抽出する

1. インスタンスに接続してローカル管理者としてログインします。

2. invoke-webrequest コマンドレットを使用して、最新のドライバーパッケージをダウンロードします。

```
PS C:\> invoke-webrequest https://ec2-windows-drivers-
downloads.s3.amazonaws.com/ENA/Latest/AwsEnaNetworkDriver.zip -
outfile $env:USERPROFILE\AwsEnaNetworkDriver.zip
```

> **Note**
>
> ファイルのダウンロード時にエラーが表示され、Windows Server 2016 以前のバージョンを使用している場合は、PowerShell ターミナルで TLS 1.2 を有効にする必要がある場合があります。次のコマンドで現在の PowerShell セッションの TLS 1.2 を有効にしてから、もう一度試してください。
>
> ```
> [Net.ServicePointManager]::SecurityProtocol =
>  [Net.SecurityProtocolType]::Tls12
> ```
>
> あるいは、インスタンスのブラウザウィンドウから最新のドライバーパッケージをダウンロードすることもできます。

3. expand-archive コマンドレットを使用して、インスタンスにダウンロードした zip アーカイブを抽出します。

```
PS C:\> expand-archive $env:userprofile\AwsEnaNetworkDriver.zip -
DestinationPath $env:userprofile\AwsEnaNetworkDriver
```

オプション 2: 特定のバージョンをダウンロードして抽出する

1. インスタンスに接続してローカル管理者としてログインします。
2. ENA Windows ドライバーのバージョン履歴 テーブル内のバージョンのリンクから、必要な特定のバージョンの ENA ドライバーパッケージをダウンロードします。
3. zip アーカイブをインスタンスに抽出します。

PowerShell を使用して ENA ドライバーをインストールする

ダウンロードしたのが最新のドライバーと特定のバージョンのいずれであっても、インストールのステップは同じです。ENA ドライバーをインストールするには、次のステップを実行します。

1. ドライバーをインストールするには、インスタンス上の `AwsEnaNetworkDriver` ディレクトリから `install.ps1` PowerShell スクリプトを実行します。エラーが発生した場合は、PowerShell 3.0 以降を使用していることを確認してください。
2. インストーラーがインスタンスを自動的に再起動しない場合は、Restart-Computer PowerShell コマンドレットを実行します。

```
PS C:\> Restart-Computer
```

ステップ 3 (オプション): インストール後に ENA ドライバーのバージョンを確認する

ENA ドライバーパッケージがインスタンスに正常にインストールされたことを確認するために、次のように新しいバージョンを確認できます。

1. インスタンスに接続してローカル管理者としてログインします。
2. Windows デバイスマネージャーを開くには、[実行] ボックスに `devmgmt.msc` と入力してください。
3. [OK] を選択してください。デバイスマネージャーウィンドウが開きます。
4. [ネットワークアダプター] の左側にある矢印を選択してリストを展開します。
5. 名前を選択するか、[Amazonエラスティックネットワークアダプター] のコンテキストメニューを開き、[プロパティ] を選択してください。これにより、[Amazon Elastic Network Adaptor のプロパティ] ダイアログが開きます。

> **Note**
>
> ENA アダプターはすべて同じドライバーを使用します。複数の ENA アダプターがある場合は、そのうちのいずれかを選択して、すべての ENA アダプターのドライバーを更新できます。

6. インストールされている現在のバージョンを確認するには、[ドライバー] タブを開いて [ドライバーのバージョン] を確認します。現在のバージョンがターゲットバージョンと一致しない場合は、「エラスティックネットワークアダプタ Windows ドライバーのトラブルシューティング」を参照してください。

ENA ドライバーのインストールをロールバックする

インストールで問題が発生した場合は、ドライバーをロールバックする必要がある場合があります。インスタンスにインストールされている ENA ドライバーの以前のバージョンにロールバックするには、次のステップを実行します。

1. インスタンスに接続してローカル管理者としてログインします。
2. Windows デバイスマネージャーを開くには、[実行] ボックスに `devmgmt.msc` と入力してください。
3. [OK] を選択してください。デバイスマネージャーウィンドウが開きます。
4. [ネットワークアダプター] の左側にある矢印を選択してリストを展開します。
5. 名前を選択するか、[Amazonエラスティックネットワークアダプター] のコンテキストメニューを開き、[プロパティ] を選択してください。これにより、[Amazon Elastic Network Adaptor のプロパティ] ダイアログが開きます。

   > **Note**
   >
   > ENA アダプターはすべて同じドライバーを使用します。複数の ENA アダプターがある場合は、そのうちのいずれかを選択して、すべての ENA アダプターのドライバーを更新できます。

6. ドライバーをロールバックするには、[ドライバー] タブを開き、[ドライバーをロールバック] を選択してください。これにより、[ドライバーパッケージのロールバック] ウィンドウが開きます。

   > **Note**
   >
   > [ドライバー] タブに [ドライバーをロールバック] アクションが表示されない場合、またはアクションが使用できない場合は、インスタンス上のドライバーストアに、以前にインストールされたドライバーパッケージが含まれていないことを意味します。この問題をトラブルシューティングするには、「トラブルシューティングシナリオ」を参照し、「予期しない ENA ドライバのバージョンがインストールされている」セクションを展開します。デバイスドライバーパッケージの選択プロセスの詳細については、Microsoft ドキュメントウェブサイトの「Windows がデバイスのドライバーパッケージを選択する方法」を参照してください。

ENA Windows ドライバーのバージョンリリースを追跡する

Windows AMI には、拡張ネットワーキングを有効にできる ENA Windows ドライバーが含まれています。

Windows Server バージョン 2016 以降の場合、最新のドライバーバージョンを使用することをお勧めします。Windows Server の以前のバージョンの場合、次のテーブルを参照して使用する ENA ドライバーのバージョンを確認してください。

Windows サーバー バージョン	ENA ドライバーバージョン
Windows Server 2012 R2	2.6.0 以前
Windows Server 2012	2.6.0 以前
Windows サーバー 2008 R2	2.2.3 以前

ENA Windows ドライバーのバージョン履歴

次の表は、各リリースの変更をまとめたものです。

ドライバーのバージョン	詳細	リリース日
2.11.0	新機能 • 最大 64 個の I/O キューをスケーリングするダイナミックキューの割り当てに対するサポートが追加されましたが、インスタンスタイプの制限が適用されます。 バグ修正 • デバイスから物理的な DMA アドレスの幅を取得するときのエラー条件の処理が欠落していた問題が修正されました。	2025 年 8 月 1 日

ドライバーのバージョン	詳細	リリース日
2.10.0	バグ修正 • 予期しない再起動を防ぐために、キューの初期化中にメモリ割り当てエラーが不適切に処理される問題を修正しました。 • ネットワークオフロードの設定ミスでデバイス機能が無視される問題を修正しました。	2025 年 6 月 24 日
2.9.0	新機能 • デバイスによって開始された非同期リセットリクエストのサポートが追加されました。 • デバイスによって提供される最大大きな LLQ 深度値を処理するためのサポートが追加されました。 • Windows イベントビューワー58001にイベント ID を追加し、デバイスの設定ミスによる予期しない電源状態遷移の可視性を強化しました。 バグ修正 • 予期しない再起動を防ぐために、デバイスの初期化中にメモリ割り当てエラーが不適切に処理される問題を修正しました。 • 割り込みサービスルーチンで、デバイスの停止中に DPC をキューに入れ、予期しない再起動を防ぐ問題を修正しました。	2024 年 12 月 12 日

ドライバーのバージョン	詳細	リリース日
2.8.0	バグ修正 • エグレスネットワークバッファリスト (NBL) 処理の全体フローにおいて、既にリリースされた NBL をリリースしようとするとメモリが破損する可能性があった競合条件を修正しました。 • すべての LSO オフロードとチェックサムオフロードを無効にすると予期しない動作につながる可能性があった L3 プロトコルの誤検出の問題を修正しました。	2024 年 9 月 30 日

ドライバーのバージョン	詳細	リリース日
2.7.0	新機能 • Windows サーバー 2012 (Windows 8) および Windows サーバー 2012 R2 (Windows 8.1) のサポートが削除されました。これらのオペレーティングシステムのバージョンは、AWS からのサポートが終了しました。Windows Server 2012 以前のバージョンでは、ドライバーのインストールが失敗します。 • デバイスへの IPv6 Tx チェックサム計算のオフロードのサポートが追加されました。 • 低レイテンシーキューイング (LLQ) の幅広いサポートが追加されました。これは、デバイスのレコメンデーションに基づいて動的に有効になります。この設定は、新しい「WideLLQ」レジストリキーで上書きできます。 • 受信パケットの Rx リングの容量が不足していることを示す、Rx オーバーランによるパケットドロップのレポート機能が追加されました。 • デバイスからの準最適な設定通知のサポートが追加されました。Windows イベントビューアーのイベント ID 59000 を参照してください。 バグ修正 • 低レイテンシーキューイング (LLQ) ヘッダーの最大サイズを超えるヘッダーを持つ Tx パケットによる不要なデバイスリセットは回避されます。	2024 年 5 月 1 日

ドライバーのバージョン	詳細	リリース日

2.6.0	新機能
	• ENA Express をサポートするインスタンスタイプ用に、次のネットワークパフォーマンスメトリクスを追加しました。
	  • ena_srd_mode
	  • ena_srd_tx_pkts
	  • ena_srd_eligible_tx_pkts
	  • ena_srd_rx_pkts
	  • ena_srd_resource_utilization
	• Nitro ベースのインスタンスタイプ用に、conntrack_allowance_available ネットワークパフォーマンスメトリクスを追加しました。
	• RX データ破損の検出による新しいアダプタのリセット理由を追加しました。
	• ドライバーのログ記録におけるインフラストラクチャを更新しました。
	
	バグ修正
	• CPU 不足によりネットワークパフォーマンスメトリクスの更新が失敗した場合、アダプターがリセットを防止します。
	•
		2023 年 6 月 20 日

ドライバーのバージョン	詳細	リリース日
	デバイスのハートビートの中断の誤検出を防止できます。 • ダウングレードの操作をサポートするようにドライバーインストールスクリプトを修正しました。 • 受信エラー数の統計情報を修正しました。	
2.5.0	発表 ENA Windows ドライバーのバージョン 2.5.0 は、Windows ドメインコントローラーで初期化できなかったため、ロールバックされました。Windows クライアントと Windows Server は影響を受けません。	2023 年 2 月 17 日

ドライバーのバージョン	詳細	リリース日
2.4.0	新機能 • Windows Server 2022 のサポートの追加 • Windows Server 2008 R2 のサポートの終了 • 第 6 世代の Amazon EC2 インスタンスのパフォーマンスを向上させるため、低レイテンシーキューイング (LLQ) を常にオンに設定します。 バグ修正 • Windows (PCW) のパフォーマンスカウンター (PCW) システムにネットワークパフォーマンスメトリクスを公開できない問題を修正しました。 • レジストリキーの読み取り操作に発生するメモリリークを修正しました。 • アダプターのリセット処理中に回復不能なエラーが発生した際に生じる、無限のリセットループを防止しました。	2022 年 4 月 28 日

ドライバーのバージョン	詳細	リリース日
2.2.4	発表 ENA Windows ドライバーのバージョン 2.2.4 は、第 6 世代 EC2 インスタンスでパフォーマンスが低下する可能性があるため、ロールバックされました。次のいずれかの方法を使用して、ドライバーをダウングレードすることをお勧めします。 • 以前のバージョンをインストールする 1. この表のリンクから、以前のバージョンのパッケージをダウンロードします (バージョン 2.2.3)。 2. install.ps1 PowerShell インストールスクリプトを実行します。 インストール前とインストール後の手順の詳細については、「Windows の拡張ネットワーキングの有効化」を参照してください。 • Amazon EC2 Systems Manager を使用して一括更新する • 次のパラメータを使用して、SSM ドキュメント `AWS-ConfigureAWSPackage` を介して一括更新を実行します。 • [Name] (名前): AwsEnaNetworkDriver • バージョン: 2.2.3	2021 年 10 月 26 日

ドライバーのバージョン	詳細	リリース日
2.2.3	新機能 • 最大 400 Gbps インスタンスネットワーキングを備えた新しい Nitro Card のサポートを追加します。 バグ修正 • ハードウェア応答不能の誤検出発生の原因となっていた、ENA ドライバによるシステム時刻の変更とシステム時刻クエリとの競合状態を修正します。 発表 Windows ENA ドライババージョン 2.2.3 は、Windows Server 2008 R2 をサポートする最後のバージョンです。現時点で ENA を利用可能なインスタンスタイプは、引き続き Windows Server 2008 R2 でサポートされ、ドライバをダウンロードすることができます。今後、Windows Server 2008 R2 をサポートするインスタンスタイプはリリースされません。また、将来のインスタンスタイプに対し、Windows Server 2008 R2 イメージをインポートおよび移行して、起動することはできません。	2021 年 3 月 25 日

ドライバーのバージョン	詳細	リリース日
2.2.2	新機能 • CloudWatch と、Windows コンシューマー用のパフォーマンスカウンターを使用したネットワークアダプターのパフォーマンスメトリクスのクエリに対するサポートを追加します。 バグ修正 • ベアメタルインスタンスでのパフォーマンスの問題が修正されました。	2020 年 12 月 21 日
2.2.1	新機能 • ホストが Elastic Network Adapter にネットワークパフォーマンスメトリクスをクエリできるようにするメソッドを追加します。	2020 年 10 月 1 日

ドライバーのバージョン	詳細	リリース日
2.2.0	新機能 • 次世代ハードウェアタイプのサポートを追加しました。 • stop-hibernate から再開後のインスタンスの開始時間を短縮し、誤検出の ENA エラーメッセージを排除しました。 パフォーマンスの最適化 • 受信トラフィックの処理を最適化しました。 • 低リソース環境での共有メモリ管理を改良しました。 バグ修正 • ドライバーのリセットに失敗するまれなシナリオで、ENA デバイスの取り外し時のシステムクラッシュが解消されました。	2020 年 8 月 12 日
2.1.5	バグ修正 • ベアメタルインスタンスでときどき発生するネットワークアダプタの初期化エラーを修正します。	2020 年 6 月 23 日

ドライバーのバージョン	詳細	リリース日
2.1.4	バグ修正 • ネットワークスタックから届く破損 LSO パケットメタデータに起因する接続問題が解消されました。 • すでにリリースされているパケットメモリへのアクセスを生じさせる、まれな競合条件に起因するシステムクラッシュが解消されました。	2019 年 11 月 25 日
2.1.2	新機能 • OS が MAC ベースの UUID を生成できるように、ベンダー ID レポートのサポートを追加しました。 バグ修正 • 初期化中の DHCP ネットワーク設定パフォーマンスを改善しました。 • 最大送信単位 (MTU) が 4K を超える場合、インバウンド IPv6 トラフィックの L4 チェックサムを適切に計算します。 • ドライバの安定性の全般的な改善と軽微なバグの修正。	2019 年 11 月 4 日

ドライバーのバージョン	詳細	リリース日
2.1.1	バグ修正 • オペレーティングシステムから着信する高度にフラグメント化された TCP LSO パケットの削除を防ぎます。 • IPv6 ネットワークの IPSec 内で、カプセル化セキュリティペイロード (ESP) プロトコルを適切に処理します。	2019 年 9 月 16 日

ドライバーのバージョン	詳細	リリース日

2.1.0	ENA Windows ドライバー v2.1 は、新しい ENA デバイス機能を提供し、パフォーマンスの向上をもたらします。また、複数の新機能を加え、安定性に関する複数の機能強化を行っています。
	
	新機能
	
	• Jumbo Frames 設定で標準化された Windows レジストリキーを使用。
	• ENA ドライバープロパティ GUI を介した VLAN ID 設定が可能。
	• 復旧フローの改善
	  • 失敗を識別する機構を改善。
	  • 調整可能な復旧パラメータのサポートを追加。
	• vCPU 数が 8 台を超える新しい EC2 インスタンスで I/O キューを最大 32 個までサポート。
	• ドライバーのメモリ占有領域を最大 90% 削減。
	
	パフォーマンスの最適化
	
	• 転送パスのレイテンシーの短縮。
	• 受信チェックサムオフロードのサポート。
	• 負荷が大きいシステムのパフォーマンスの最適化 (ロック機構の使用の最適化)。
	•
		2019 年 7 月 1 日

ドライバーのバージョン	詳細	リリース日
	CPU 使用率を低減させて負荷時のシステム応答を改善するための機能強化。 バグ修正 • 連続しない Tx ヘッダーの無効な解析に伴うクラッシュを修正。 • ベアメタルインスタンスにおける Elastic Network Interface デタッチ中のドライバー v1.5 のクラッシュを修正。 • IPv6 での LSO 疑似ヘッダーチェックサム計算エラーを修正。 • 初期化の失敗時に起こり得るメモリリソースリークの修正。 • IPv4 フラグメントの TCP/UDP チェックサムオフロードを無効化。 • VLAN 設定の修正。VLAN の優先度のみを無効化すべきときに VLAN が間違って無効化されていました。 • イベントビューアーによるカスタムドライバーメッセージの正しい解析を有効化。 • 無効なタイムスタンプ処理に伴うドライバーの初期化エラーを修正。 • データ処理と ENA デバイス無効化との間の競合状態を修正。	

ドライバーのバージョン	詳細	リリース日
1.5.0	更新 • 安定性の向上およびパフォーマンス修正。 • [Receive Buffers] は、ENA NIC の [Advanced Properties] で最大 8192 に設定できるようになりました。 • デフォルトの [Receive Buffers] は 1000 です。	2018 年 10 月 4 日
1.2.3	信頼性の修正が含まれ、Windows Server 2016 から Windows Server 2008 R2 のサポート内容を統合します。	2018 年 2 月 13 日
1.0.8	初回リリース。Windows Server 2008 R2、Windows Server 2012 RTM、Windows Server 2012 R2、および Windows Server 2016 用の AMI が含まれています。	2016年7月日

Amazon SNS から ENA Windows ドライバーのリリースの通知を定期的に受け取る

EC2 Windows ドライバーの新しいバージョンがリリースされたときには、Amazon SNS から通知を受け取ることができます。このような通知をサブスクライブするには、以下の手順を使用します。

EC2 の通知をサブスクライブする

1. Amazon SNS コンソールのhttps://console.aws.amazon.com/sns/v3/homeを開いてください。
2. ナビゲーションバーで、必要に応じて、リージョンを [米国東部 (バージニア北部)] に変更します。購読する SNS 通知がこのリージョンにあるため、このリージョンを選択する必要があります。
3. ナビゲーションペインで [Subscriptions] を選択してください。
4. [Create subscription] を選択してください。
5. [Create subscription] ダイアログボックスで、次の操作を行います。
   a. [TopicARN] では、次の Amazon リソースネーム (ARN) をコピーします。

arn:aws:sns:us-east-1:801119661308:ec2-windows-drivers

b. [プロトコル] で `Email` を選択してください。

c. [エンドポイント] には、通知を送信する E メールアドレスを入力してください。

d. [Create subscription] を選択してください。

6. 確認メールが送信されます。E メールを開き、指示に従ってサブスクリプションを完了します。

サブスクライバには、EC2 Windows ドライバーの新しいバージョンがリリースされるたびに、通知が送信されます。通知が不要になった場合は、次の手順で受信登録を解除します。

Amazon EC2 Windows ドライバー通知のサブスクリプションを解除する

1. Amazon SNS コンソールのhttps://console.aws.amazon.com/sns/v3/homeを開いてください。
2. ナビゲーションペインで [Subscriptions] を選択してください。
3. 任意のサブスクリプションのチェックボックスを選択し、[アクション]、[サブスクリプションの削除] を選択してください。確認を求めるメッセージが表示されたら、[削除] を選択してください。

Windows インスタンス用 Paravirtual ドライバー

Windows AMI には、仮想ハードウェアにアクセスできるようにするためのドライバー一式が含まれています。Amazon EC2 は、これらのドライバーを使用して、インスタンスストアボリュームと Amazon EKS ボリュームをデバイスにマップします。次の表に、さまざまなドライバの主な相違点を示します。

	Red Hat PV	Citrix PV	AWS PV
インスタンスタイプ	すべてのインスタンスタイプでサポートされているわけではありません。サポートされていないインスタンスタイプを指定すると、インスタンスの機能が低下します。	Xen インスタンスタイプでサポートされています。	Xen インスタンスタイプでサポートされています。

	Red Hat PV	Citrix PV	AWS PV
アタッチ済みボリューム	最大 16 個のアタッチされたボリュームをサポートします。	16 個を超えるアタッチされたボリュームをサポートします。	16 個を超えるアタッチされたボリュームをサポートします。

	Red Hat PV	Citrix PV	AWS PV
Network	このドライバーには、高速 FTP ファイル転送など、負荷が高いときにネットワーク接続がリセットされるという既知の問題があります。		このドライバーは、互換性のあるインスタンスタイプで使用された場合、ネットワークアダプタで自動的にジャンボフレームを設定します。インスタンスがクラスタープレイスメントグループ内にある場合、この結果クラスタープレイスメントグループ内にあるインスタンス間のネットワークパフォーマンスが向上します。詳細については、「Amazon EC2 インスタンスの

	Red Hat PV	Citrix PV	AWS PV
			プレイスメントグループ」を参照してください。

以下の表では、Amazon EC2 で Windows Server の各バージョンで実行する必要のある PV ドライバーを示しています。

Windows Server バージョン	PV ドライバーバージョン
Windows Server 2025	サポートされていません
Windows Server 2022	AWS PV 最新バージョン
Windows Server 2019	AWS PV 最新バージョン
Windows Server 2016	AWS PV 最新バージョン
Windows Server 2012 R2	AWS PV バージョン 8.4.3
Windows Server 2012	AWS PV バージョン 8.4.3
Windows サーバー 2008 R2	AWS PV バージョン 8.3.5
Windows Server 2008	Citrix PV 5.9
Windows Server 2003	Citrix PV 5.9

コンテンツ

- AWS PV ドライバー
- Citrix PV ドライバー
- RedHat PV ドライバー
- の通知のサブスクライブ
- EC2 Windows インスタンスでの PV ドライバーのアップグレード

- Windows インスタンスでの PV ドライバーのトラブルシューティング

AWS PV ドライバー

AWS PV ドライバーは %ProgramFiles%\Amazon\Xentools ディレクトリに格納されます。このディレクトリには、パブリックシンボルと、XenStore のエントリにアクセスできるようにするコマンドラインツール xenstore_client.exe も含まれます。例えば、次の PowerShell コマンドは、Hypervisor から現在時刻を返します。

```
PS C:\> [DateTime]::FromFileTimeUTC((gwmi -n root\wmi -cl
 AWSXenStoreBase).XenTime).ToString("hh:mm:ss")
11:17:00
```

AWS PV ドライバーコンポーネントは Windows レジストリの HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services の下にリストされています。これらのドライバーコンポーネントは、xenbus、xeniface、xennet、xenvbd、および xenvif です。

AWS PV ドライバーには、ユーザーモードで実行される、LiteAgent という Windows サービスもあります。これは、Xen 世代のインスタンスの AWS API からのシャットダウンイベントや再起動イベントなどのタスクを処理するサービスです。コマンドラインでServices.mscと入力することで、サービスにアクセスしてサービスを管理できます。Nitro 世代のインスタンスで実行している場合、AWS PV ドライバーは使用されず、ドライバーバージョン 8.2.4 以降では LiteAgent サービスは自動停止します。最新の AWS PV ドライバーに更新すると、LiteAgent も更新され、すべてのインスタンス世代で信頼性が向上します。

最新の AWS PV ドライバーのインストール

Amazon Windows AMI には、仮想ハードウェアにアクセスできるようにするためのドライバー一式が含まれています。Amazon EC2 は、これらのドライバーを使用して、インスタンスストアボリュームと Amazon EBS ボリュームをデバイスにマップします。EC2 Windows インスタンスの安定性とパフォーマンスを向上させるため、最新のドライバーをインストールすることをお勧めします。

インストールオプション

- AWS Systems Manager を使用して、自動的に PV ドライバーを更新します。詳細については、AWS Systems Manager ユーザーガイドのチュートリアル: Windows の EC2 インスタンスで PV ドライバーを自動的に更新する (コンソール)を参照してください。

- インストールプログラムを手動で実行するために、ドライバーパッケージをダウンロードします。システム要件については、必ず `readme.txt` ファイルを確認してください。AWS PV ドライバーのダウンロードとインストール、またはドメインコントローラーのアップグレードに関する情報については、手動による Windows Server インスタンスのアップグレード (AWS PV アップグレード)を参照してください。

AWS PV ドライバーパッケージの履歴

次の表は、各ドライバーリリースでの AWS PV ドライバーの変更点を示しています。

パッケージバージョン	詳細	リリース日
8.6.0	• XenStore インタラクションの安定性の修正。	2025 年 5 月 27 日
8.5.0	• ネットワークデバイスのデタッチ中にクラッシュが発生するまれなケースに対処する安定性の修正。 • EBS ボリュームのデタッチ中にクラッシュが発生するまれなケースに対処するための安定性の修正。 • パッケージインストーラのバグを修正しました。 • `Pnputil` を使用するように PV インストーラを更新しました。	2024 年 10 月 31 日
8.4.3	アップグレード体験が向上するよう、パッケージインストーラのバグを修正しました。これは、Windows Server 2012 および 2012 R2 で実行できる最後のバージョンです。このバージョンはダウンロード可能ですが、Windows Server 2012 と 2012 R2 がサポートを終了したため、現在はサポートされていません。	2023 年 1 月 24 日
8.4.2	競合状態に対処するための安定性向上。	2022 年 4 月 13 日
8.4.1	パッケージインストーラが改善されました。	2022 年 1 月 7 日
8.4.0	• ディスク IO がスタックするまれなケースに対処するための安定性の修正。	2021 年 3 月 2 日

パッケージバージョン	詳細	リリース日
	• EBS ボリュームのデタッチ中にクラッシュが発生するまれなケースに対処するための安定性の修正。 • 20,000 IOPS 以上が使用され、ボトルネックによりパフォーマンスが低下しているワークロードに対して、複数のコア間で負荷を分散する機能を追加しました。この機能を有効にするには、20,000 以上のディスク IOPS を使用するワークロードでは、CPU のボトルネックによりパフォーマンスが低下しますを参照してください。	
8.3.5	パッケージインストーラが改善されました。 これは、Windows Server 2008 R2 で実行できる最後のバージョンです。このバージョンはダウンロード可能ですが、サポートが終了しています。Windows Server 2008 R2 はサポートライフサイクルが終了し、Microsoft によるサポートが終了しています。	2022 年 1 月 7 日
8.3.4	ネットワークデバイスのアタッチメントの信頼性が向上しました。	2020 年 8 月 4 日
8.3.3	• エラー処理パス中のバグチェックを防ぐために、XenStore に面するコンポーネントを更新しました。 • 無効な SRB が送信されたときにクラッシュしないように、ストレージコンポーネントを更新しました。 Windows Server 2008 R2 インスタンスでこのドライバーを更新するには、まず、次の Microsoft セキュリティアドバイザリ Microsoft Security Advisory 3033929 に対応する適切なパッチがインストールされていることを確認する必要があります。	2020 年 2 月 4 日
8.3.2	ネットワーキングコンポーネントの信頼性を強化。	2019 年 7 月 30 日
8.3.1	ストレージコンポーネントのパフォーマンスと堅牢性の改善。	2019 年 6 月 12 日

パッケージバージョン	詳細	リリース日
8.2.7	最新世代のインスタンスタイプへの移行をサポートする効率が向上しました。	2019 年 5 月 20 日
8.2.6	クラッシュダンプパスの効率が向上しました。	2019 年 1 月 15 日
8.2.5	追加のセキュリティ拡張。 パッケージで PowerShell インストーラーを使用できるようになりました。	2018 年 12 月 12 日
8.2.4	信頼性の向上。	2018 年 10 月 2 日
8.2.3	パフォーマンス向上とバグ修正が行われています。 EBS ボリューム ID を EBS ボリュームのディスクシリアル番号として報告しました。これにより、S2D などのクラスターシナリオが可能になります。	2018 年 5 月 29 日
8.2.1	ネットワークとストレージのパフォーマンス向上および複数の堅牢性の修正。 このバージョンがインストールされていることを確認するには、次に示す Windows のレジストリ値を参照してください: `HKLM\Software\Amazon\PVDriver\Version` `8.2.1`。	2018 年 3 月 8 日
7.4.3	Windows Server 2016 のサポートが追加されました。 サポートされているすべての Windows OS バージョンの安定性の修正。 *AWS PV ドライバーのバージョン 7.4.3 の署名は、2019 年 3 月 29 日に期限が切れます。最新の AWS PV ドライバーにアップグレードすることをお勧めします。	2016 年 11 月 18 日

パッケージバージョン	詳細	リリース日
7.4.2	X1 インスタンスタイプのサポートの安定性の修正。	2016 年 8 月 2 日
7.4.1	• AWS PV ストレージドライバーのパフォーマンス向上。 • AWS PV ストレージドライバーの安定性の修正: インスタンスがバグチェックコード 0x0000DEAD でシステムクラッシュを発生させる問題を修正しました。 • AWS PV ネットワークドライバーの安定性の修正。 • Windows Server 2008R2 のサポートが追加されました。	2016 年 7 月 12 日
7.3.2	• ログ記録と診断を強化しました。 • AWS PV ストレージドライバーの安定性の修正。ディスクをインスタンスに再アタッチした後で、Windows に表示されない場合があります。 • Windows Server 2012 のサポートが追加されました。	2015 年 6 月 24 日
7.3.1	TRIM 更新: TRIM リクエストに関する修正。この修正によって、多数の TRIM リクエストを管理する場合にインスタンスが安定し、インスタンスのパフォーマンスが向上します。	
7.3.0	TRIM のサポート: AWS PV ドライバーはハイパーバイザーに TRIM リクエストを送信するようになりました。エフェメラルディスクは、基になるストレージが TRIM (SSD) をサポートしていれば、TRIM リクエストを適切に処理します。2015 年 3 月現在、EBS ベースのストレージで TRIM はサポートされていません。	

パッケージバージョン	詳細	リリース日
7.2.5	• AWS PV ストレージドライバーの安定性を修正: 場合によって、AWS PV ドライバーが無効なメモリを間接参照することで、システム障害が発生していました。 • クラッシュダンプ生成中の安定性を修正: 場合によって、クラッシュダンプの書き込み中に AWS PV ドライバーが競合状態でスタックすることがありました。このリリースの前、この問題を解決するには、ドライバーを強制的に停止して再起動するしかなかったため、メモリダンプが失われていました。	
7.2.4	デバイス ID の永続性: このドライバーの修正により、プラットフォームの PCI デバイス ID がマスクされ、システムによって常に同じデバイス ID が公開されます。この ID はインスタンスが移動されても変更されません。より一般的にこの修正は、ハイパーバイザーが仮想デバイスを公開する方法に影響を与えます。この修正には、AWS PV ドライバーの共同インストーラーの修正も含まれており、システムはマッピングされた仮想デバイスを保持するようになります。	
7.2.2	• AWS PV ドライバーをディレクトリサービス復元モード (DSRM) でロード: ディレクトリサービス復元モードは、Windows Server ドメインコントローラーのセーフモード起動オプションです。 • 仮想ネットワークアダプタデバイスが再接続されたときにデバイス ID を保持: この修正により、システムは MAC アドレスのマッピングをチェックし、デバイス ID を保持するようになります。また、アダプターは再アタッチされた場合にその静的な設定を保持するようになります。	

パッケージバージョン	詳細	リリース日
7.2.1	• セーフモードでの実行: ドライバーがセーフモードでロードされない問題を修正しました。これまで AWS PV ドライバーは正常に実行中のシステムでのみインスタンス化されていました。 • Microsoft Windows ストレージプールにディスクを追加: これまでページ 83 クエリを合成していました。この修正によりページ 83 のサポートが無効になりました。PV ディスクは有効なクラスターディスクでないため、クラスター環境で使用されているストレージプールに影響を与えません。	
7.2.0	ベース: AWS PV ベースのバージョン。	

Citrix PV ドライバー

Citrix PV ドライバは `%ProgramFiles%\Citrix\XenTools` (32 ビットインスタンス) または `%ProgramFiles(x86)%\Citrix\XenTools` (64 ビットインスタンス) ディレクトリに格納されます。

Citrix PV ドライバーコンポーネントは Windows レジストリの `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services` の下にリストされています。これらのドライバー コンポーネントは、xenevtchn、xeniface、xennet、Xennet6、xensvc、xenvbd、および xenvif です。

Citrix には、Windows サービスとして実行する、XenGuestAgent というドライバーコンポーネントも付属しています。これは、API からのシャットダウンイベントや再起動イベントなどのタスクを処理します。コマンドラインで`Services.msc`と入力することで、サービスにアクセスしてサービスを管理できます。

特定のワークロードの実行中にネットワークエラーが発生した場合、Citrix PV ドライバーの TCP オフロード機能を無効にすることが必要になる場合があります。詳細については、「TCP オフロード」を参照してください。

RedHat PV ドライバー

RedHat ドライバーはレガシーインスタンスでサポートされていますが、ドライバーの制限のため、RAM が 12 GB を超える新しいインスタンスはお勧めしません。RAM が 12 GB を超えてい

て、RedHat ドライバーを実行しているインスタンスは、起動に失敗し、アクセスできなくなることがあります。Citrix PV ドライバーに RedHat ドライバーをアップグレードしてから、Citrix PV ドライバーを AWS PV ドライバーにアップグレードすることをお勧めします。

RedHat ドライバー用のソースファイルは、`%ProgramFiles%\RedHat` (32 ビットインスタンス) または `%ProgramFiles(x86)%\RedHat` (64 ビットインスタンス) ディレクトリにあります。ドライバーには、RedHat Paravirtualized ネットワークドライバーである `rhelnet` と、RedHat SCSI ミニドライバーである `rhelscsi` の 2 つがあります。

の通知のサブスクライブ

EC2 Windows ドライバーの新しいバージョンがリリースされたときには、Amazon SNS から通知を受け取ることができます。これらの通知をサブスクライブできます。

> **Note**
>
> SNS トピックが作成されたリージョンを指定する必要があります。

サブスクライバには、EC2 Windows ドライバーの新しいバージョンがリリースされるたびに、通知が送信されます。通知が不要になった場合は、サブスクライブを解除できます。詳細については、「Amazon SNS トピックおよびサブスクリプションを削除する」を参照してください。

Console

通知をサブスクライブするには

1. Amazon SNS コンソールのhttps://console.aws.amazon.com/sns/v3/homeを開いてください。
2. ナビゲーションバーで、必要に応じて、リージョンを [米国東部 (バージニア北部)] に変更します。購読する SNS 通知がこのリージョンにあるため、このリージョンを選択する必要があります。
3. ナビゲーションペインで [Subscriptions] を選択してください。
4. [Create subscription] を選択してください。
5. [Create subscription] ダイアログボックスで、次の操作を行います。

   a. [TopicARN] では、次の Amazon リソースネーム (ARN) をコピーします。

      arn:aws:sns:us-east-1:801119661308:ec2-windows-drivers

b. [プロトコル] で Email を選択してください。

c. [エンドポイント] では、通知を受信するために使用できる E メールアドレスを入力してください。

d. [Create subscription] を選択してください。

6. 確認メールが送信されます。E メールを開き、指示に従ってサブスクリプションを完了します。

AWS CLI

通知をサブスクライブするには

次のコマンドを使用します。

```
aws sns subscribe \
    --topic-arn arn:aws:sns:us-east-1:801119661308:ec2-windows-drivers \
    --region us-east-1 \
    --protocol email \
    --notification-endpoint YourUserName@YourDomainName.ext
```

PowerShell

通知をサブスクライブするには

次のコマンドを使用します。

```
Connect-SNSNotification `
    -TopicArn 'arn:aws:sns:us-east-1:801119661308:ec2-windows-drivers' `
    -Region us-east-1 `
    -Protocol email `
    -Endpoint "YourUserName@YourDomainName.ext"
```

EC2 Windows インスタンスでの PV ドライバーのアップグレード

EC2 Windows インスタンスの安定性とパフォーマンスを向上させるため、最新の PV ドライバーをインストールすることをお勧めします。このページの指示は、ドライバーパッケージをダウンロードしてインストールプログラムを実行するのに役立ちます。

Windows インスタンスが使用しているドライバーを確認するには

[デバイスマネージャー] を開き、[ネットワークアダプタ] を表示します。PV ドライバーが、次のいずれかであるかどうかを確認してください。

- AWS PV Network Device
- Citrix PV Ethernet Adapter
- RedHat PV NIC Driver

システム要件

システム要件については、ダウンロード中の `readme.txt` ファイルを確認してください。

内容

- ディストリビューターによる Windows Server インスタンスのアップグレード (AWS PV アップグレード)
- 手動による Windows Server インスタンスのアップグレード (AWS PV アップグレード)
- ドメインコントローラーのアップグレード (AWS PV アップグレード)
- Windows Server 2008 および 2008 R2 インスタンスのアップグレード (Redhat から Citrix PV へのアップグレード)
- Citrix Xen ゲストエージェントサービスのアップグレード

ディストリビューターによる Windows Server インスタンスのアップグレード (AWS PV アップグレード)

ディストリビューター (AWS Systems Manager の機能) を使用して、AWS PV ドライバーパッケージをインストールまたはアップグレードすることができます。インストールまたはアップグレードを 1 回実行することも、スケジュールに従ってインストールまたは更新することもできます。[インストールタイプ] の `In-place update` オプションは、このディストリビューターパッケージではサポートされていません。

> **Important**
>
> インスタンスがドメインコントローラーである場合は、ドメインコントローラーのアップグレード (AWS PV アップグレード)を参照してください。ドメインコントローラーのインスタ

ンスのアップグレードプロセスは、Windows のスタンダードエディションでのプロセスとは異なります。

1. 変更を元に戻す必要がある場合に備えて、バックアップを作成することをお勧めします。

   > **Tip**
   >
   > Amazon EC2 コンソールから AMI を作成する代わりに、Systems Manager Automation を使用し、`AWS-CreateImage` ランブックを使用して AMI を作成できます。詳細については、「AWS Systems Manager オートメーションランブックリファレンス」の「ユーザーガイド」の「AWS-CreateImage」を参照してください。

   a. インスタンスを停止すると、インスタンスストアボリューム上のデータは消去されます。インスタンスを停止する前に、必要なデータをインスタンスストアボリュームから永続的ストレージ (Amazon EBS や Amazon S3 など) にコピーしていることを確認します。

   b. ナビゲーションペインで、[インスタンス] を選択してください。

   c. ドライバーのアップグレードが必要なインスタンスを選択し、[インスタンスの状態]、[インスタンスの停止] の順に選択してください。

   d. インスタンスを停止したら、インスタンスを選択し、[アクション]、[イメージとテンプレート]、[イメージの作成] の順に選択してください。

   e. [Instance state (インスタンスの状態)]、[Start instance (インスタンスの開始)] の順に選択してください。

2. リモートデスクトップを使用してインスタンスに接続します。詳細については、「the section called “RDP クライアントを使用して接続する”」を参照してください。

3. このアップグレードを実行する前に、システム以外のすべてのディスクをオフラインにし、ディスクの管理でセカンダリディスクへのドライブ文字のマッピングをメモすることをお勧めします。AWS PV ドライバーのインプレースアップグレードを実行している場合は、このステップは必要ありません。また、サービスコンソールで、不可欠でないサービスを [Manual] 起動に設定することをお勧めします。

4. ディストリビューターを使用して AWS PV ドライバーパッケージをインストールまたはアップグレードする方法については、「AWS Systems Manager ユーザーガイド」の「パッケージのインストールまたは更新」の手順を参照してください。

5. [名前] で、AWSPVDriver を選択してください。

6. [インストールタイプ] で、[アンインストールと再インストール] を選択してください。
7. 必要に応じてパッケージの他のパラメータを設定し、Step 4 の参照手順を使用してインストールまたはアップグレードを実行します。

   ディストリビューターパッケージの実行後、インスタンスは自動的に再起動され、ドライバーがアップグレードされます。インスタンスは最大 15 分間、使用できなくなります。
8. アップグレードが完了し、インスタンスが Amazon EC2 コンソールの両方のヘルスチェックに合格した後、リモートデスクトップを使用してインスタンスに接続して、新しいドライバーがインストールされたことを確認します。
9. 接続後、次の PowerShell コマンドを実行します。

   ```
   Get-ItemProperty HKLM:\SOFTWARE\Amazon\PVDriver
   ```
10. ドライバーバージョンがドライバーのバージョン履歴の表に掲載されている最新バージョンと同じであることを確認します。詳細については、「AWS PV ドライバーパッケージの履歴 ディスク管理を開く」を参照して、オフラインセカンダリボリュームを確認し、Step 3 でメモしたドライブ文字に対応してオンラインにします。

以前に Netsh を使用して Citrix PV ドライバーに対して TCP オフロード を無効にしている場合は、AWS PV ドライバーにアップグレード後、この機能を再度有効にすることをお勧めします。Citrix ドライバーにおける TCP オフロードの問題は AWS PV ドライバーでは存在しません。そのため、AWS PV ドライバーを使用することで TCP オフロードはより高いパフォーマンスを提供します。

ネットワークインターフェイスに静的 IP アドレスまたは DNS 設定を既に適用している場合、AWS PV ドライバーをアップグレードした後、静的 IP アドレスまたは DNS 設定を再適用することが必要な場合があります。

手動による Windows Server インスタンスのアップグレード (AWS PV アップグレード)

次の手順に従って、Windows Server 2008 R2、Windows Server 2012、Windows Server 2012 R2、Windows Server 2016、Windows Server 2019、または Windows Server 2022 で、AWS PV ドライバーのインプレースアップグレードを実行するか、Citrix PV ドライバーから AWS PV ドライバーにアップグレードします。このアップグレードは、RedHat ドライバーまたは Windows Server の他のバージョンでは使用できません。

Windows Server の古いバージョンの一部では、最新のドライバーを使用することができません。ご使用のオペレーティングシステムで、どのドライバーバージョンを使用すべきかを確認するには、

「Windows インスタンス用 Paravirtual ドライバー」(Windows インスタンス用 Paravirtual ドライバー) のドライバーバージョン表を参照してください。

> **Important**
>
> インスタンスがドメインコントローラーである場合は、ドメインコントローラーのアップグレード (AWS PV アップグレード)を参照してください。ドメインコントローラーのインスタンスのアップグレードプロセスは、Windows のスタンダードエディションでのプロセスとは異なります。

AWS PV ドライバーを手動でアップグレードするには

1. 変更を元に戻す必要がある場合に備えて、バックアップを作成することをお勧めします。

   > **Tip**
   >
   > Amazon EC2 コンソールから AMI を作成する代わりに、Systems Manager Automation を使用し、`AWS-CreateImage` ランブックを使用して AMI を作成できます。詳細については、「AWS Systems Manager オートメーションランブックリファレンス」の「ユーザーガイド」の「AWS-CreateImage」を参照してください。

   a. インスタンスを停止すると、インスタンスストアボリューム上のデータは消去されます。インスタンスを停止する前に、必要なデータをインスタンスストアボリュームから永続的ストレージ (Amazon EBS や Amazon S3 など) にコピーしていることを確認します。
   b. ナビゲーションペインで、[インスタンス] を選択してください。
   c. ドライバーのアップグレードが必要なインスタンスを選択し、[インスタンスの状態]、[インスタンスの停止] の順に選択してください。
   d. インスタンスを停止したら、インスタンスを選択し、[アクション]、[イメージとテンプレート]、[イメージの作成] の順に選択してください。
   e. [Instance state (インスタンスの状態)]、[Start instance (インスタンスの開始)] の順に選択してください。
2. リモートデスクトップを使用してインスタンスに接続します。
3. このアップグレードを実行する前に、システム以外のすべてのディスクをオフラインにし、ディスクの管理でセカンダリディスクへのドライブ文字のマッピングをメモすることをお勧めします。AWS PV ドライバーのインプレースアップグレードを実行している場合は、このステップ

は必要ありません。また、サービスコンソールで、不可欠でないサービスを [Manual] 起動に設定することをお勧めします。

4. 次のいずれかのオプションを使用して、インスタンスにドライバをダウンロードします。
   - ブラウザ - zip アーカイブを抽出するには、最新のドライバーパッケージをインスタンスにダウンロードします。
   - PowerShell – 次のコマンドを実行します。

     ```
     Invoke-WebRequest https://s3.amazonaws.com/ec2-windows-drivers-downloads/AWSPV/Latest/AWSPVDriver.zip -outfile $env:USERPROFILE\pv_driver.zip
     Expand-Archive $env:userprofile\pv_driver.zip -DestinationPath $env:userprofile\pv_drivers
     ```

     ファイルのダウンロード時にエラーが表示され、Windows Server 2016 以前のバージョンを使用している場合は、PowerShell ターミナルで TLS 1.2 を有効にする必要がある場合があります。次のコマンドで現在の PowerShell セッションの TLS 1.2 を有効にしてから、もう一度試してください。

     ```
     [Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
     ```

5. `AWSPVDriverSetup.msi` を実行します。

MSI の実行後、インスタンスは自動的に再起動され、ドライバーがアップグレードされます。インスタンスは最大 15 分間、使用できなくなります。アップグレードが完了し、インスタンスが Amazon EC2 コンソールの両方のヘルスチェックに合格したら、リモートデスクトップを使用してインスタンスに接続後、次の PowerShell コマンドを実行することで、新しいドライバーがインストールされたことを確認できます。

```
Get-ItemProperty HKLM:\SOFTWARE\Amazon\PVDriver
```

ドライバーバージョンがドライバーのバージョン履歴の表に掲載されている最新バージョンと同じであることを確認します。詳細については、「AWS PV ドライバーパッケージの履歴 ディスク管理を開く」を参照して、オフラインセカンダリボリュームを確認し、Step 3 でメモしたドライブ文字に対応してオンラインにします。

以前に Netsh を使用して Citrix PV ドライバーに対して TCP オフロード を無効にしている場合は、AWS PV ドライバーにアップグレード後、この機能を再度有効にすることをお勧めします。Citrix ドライバーにおける TCP オフロードの問題は AWS PV ドライバーでは存在しません。そ

のため、AWS PV ドライバーを使用することで TCP オフロードはより高いパフォーマンスを提供します。

ネットワークインターフェイスに静的 IP アドレスまたは DNS 設定を既に適用している場合、AWS PV ドライバーをアップグレードした後、静的 IP アドレスまたは DNS 設定を再適用することが必要な場合があります。

ドメインコントローラーのアップグレード (AWS PV アップグレード)

ドメインコントローラーで次の手順を使用して、AWS PV ドライバーのインプレースアップグレード、または Citrix PV ドライバーから AWS PV ドライバーへのアップグレードを実行します。アップグレード中に FSMO ロールが動作し続けるように、アップグレードを開始する前に、それらのロールを他のドメインコントローラーに移行することをお勧めします。詳細については、Microsoft 学習ウェブサイトの「How to view and transfer FSMO roles」を参照してください。

ドメインコントローラーをアップグレードするには

1. 変更を元に戻す必要がある場合に備えて、ドメインコントローラーのバックアップを作成することをお勧めします。AMI をバックアップとして使用することはサポートされていません。詳細については、Microsoft のドキュメントの仮想化ドメインコントローラーのバックアップと復元の考慮事項を参照してください。
2. Windows がディレクトリサービス復元モード (DSRM) で起動するように設定するには、次のコマンドを実行します。

   > **Warning**
   >
   > このコマンドを実行する前に、DSRM パスワードを知っていることを確認してください。この情報は、アップグレードが完了した後インスタンスにログインするために必要で、インスタンスは自動的に再起動します。

   ```
   bcdedit /set {default} safeboot dsrepair
   ```

   PowerShell:

   ```
   PS C:\> bcdedit /set "{default}" safeboot dsrepair
   ```

アップグレードユーティリティでは、AWS PV ドライバーをインストールできるように Citrix PV ストレージドライバーが削除されるため、システムは DSRM で起動する必要があります。したがって、ディスク管理のセカンダリディスクへのドライブ文字とフォルダのマッピングに注意することをお勧めします。Citrix PV ストレージドライバーが存在しない場合、セカンダリドライブは検出されません。セカンダリドライブで NTDS フォルダを使用するドメインコントローラーは、セカンダリディスクが検出されないため起動しません。

> **⚠ Warning**
>
> このコマンドの実行後、手動でシステムを再起動しないでください。Citrix PV ドライバーは DSRM をサポートしないため、システムにアクセスできなくなります。

3. 次のコマンドを実行して、**`DisableDCCheck`** をレジストリに追加します。

```
reg add HKLM\SOFTWARE\Wow6432Node\Amazon\AWSPVDriverSetup /v DisableDCCheck /t
 REG_SZ /d true
```

4. zip アーカイブを抽出するには、最新のドライバーパッケージをインスタンスにダウンロードします。
5. `AWSPVDriverSetup.msi` を実行します。

   MSI の実行後、インスタンスは自動的に再起動され、ドライバーがアップグレードされます。インスタンスは最大 15 分間、使用できなくなります。
6. アップグレードが完了し、インスタンスが Amazon EC2 コンソールの両方のヘルスチェックに合格した後、リモートデスクトップを使用してインスタンスに接続します。ディスク管理を開くを参照して、オフラインセカンダリボリュームを確認し、先にメモしたドライブ文字とフォルダマッピングに対応してオンラインにします。

   ユーザー名を形式 hostname\administrator で指定してインスタンスに接続する必要があります。例えば、Win2k12TestBox\administrator のようにします。
7. DSRM 起動設定を削除するには、次のコマンドを実行します。

```
bcdedit /deletevalue safeboot
```

8. インスタンスを再起動します。
9. アップグレードプロセスを完了するには、新しいドライバーがインストールされたことを確認します。デバイスマネージャーの [Storage Controllers] で、[AWS PV Storage Host Adapter] を見

つけます。ドライバーバージョンがドライバーのバージョン履歴の表に掲載されている最新バージョンと同じであることを確認します。詳細については、AWS PV ドライバーパッケージの履歴を参照してください。

10. レジストリから **DisableDCCheck** を削除するには、次のコマンドを実行します。

```
reg delete HKLM\SOFTWARE\Wow6432Node\Amazon\AWSPVDriverSetup /v DisableDCCheck
```

> **Note**
>
> 以前に Netsh を使用して Citrix PV ドライバーに対して TCP オフロード を無効にしている場合は、AWS PV ドライバーにアップグレード後、この機能を再度有効にすることをお勧めします。Citrix ドライバーにおける TCP オフロードの問題は AWS PV ドライバーでは存在しません。そのため、AWS PV ドライバーを使用することで TCP オフロードはより高いパフォーマンスを提供します。

Windows Server 2008 および 2008 R2 インスタンスのアップグレード (Redhat から Citrix PV へのアップグレード)

RedHat ドライバーを Citrix PV ドライバーにアップグレードする前に、次のことを実行してください。

- EC2Config サービスの最新バージョンをインストールします。詳細については、EC2Config の最新バージョンのインストールを参照してください。
- Windows PowerShell 3.0 がインストールされていることを確認します。インストールされているバージョンを確認するには、PowerShell ウィンドウで、次のコマンドを実行します。

  ```
  PS C:\> $PSVersionTable.PSVersion
  ```

  Windows PowerShell 3.0 は、Windows Management Framework (WMF) バージョン 3.0 のインストールパッケージに同梱されています。Windows PowerShell 3.0 をインストールする必要がある場合は、Microsoft Download Center のWindows Management Framework 3.0を参照してください。
- インスタンスにある重要な情報のバックアップを作成するか、インスタンスから AMI を作成します。AMI の作成の詳細については、Amazon EBS-backed AMI を作成するを参照してください。

> **Tip**
>
> Amazon EC2 コンソールから AMI を作成する代わりに、Systems Manager Automation を使用し、`AWS-CreateImage` ランブックを使用して AMI を作成できます。詳細については、「AWS Systems Manager オートメーションランブックリファレンス」の「ユーザーガイド」の「AWS-CreateImage」を参照してください。

AMI を作成する場合は、事前に次のことを実行してください。

- パスワードを書き留める。
- Sysprep ツールを手動でまたは EC2Config サービスを使用して実行しないでください。
- DHCP を使用して IP アドレスを自動的に取得するようにイーサネットアダプタを設定します。

RedHat ドライバーをアップグレードするには

1. インスタンスに接続してローカル管理者としてログインします。インスタンスへの接続の詳細についてはRDP を使用した Windows インスタンスへの接続を参照してください。
2. インスタンスで、Citrix PV アップグレードパッケージをダウンロードします。
3. アップグレードパッケージを好きな場所に展開します。
4. Upgrade.bat ファイルをダブルクリックしてください。セキュリティ警告が表示された場合は、[実行] を選択してください。
5. [Upgrade Drivers] (ドライバのアップグレード) ダイアログボックスの内容を確認し、アップグレードを開始する場合は、[はい] を選択してください。
6. [Red Hat Paravirtualized Xen Drivers for Windows uninstaller] ダイアログボックスで [はい] を選択して RedHat ソフトウェアを削除します。インスタンスが再起動されます。

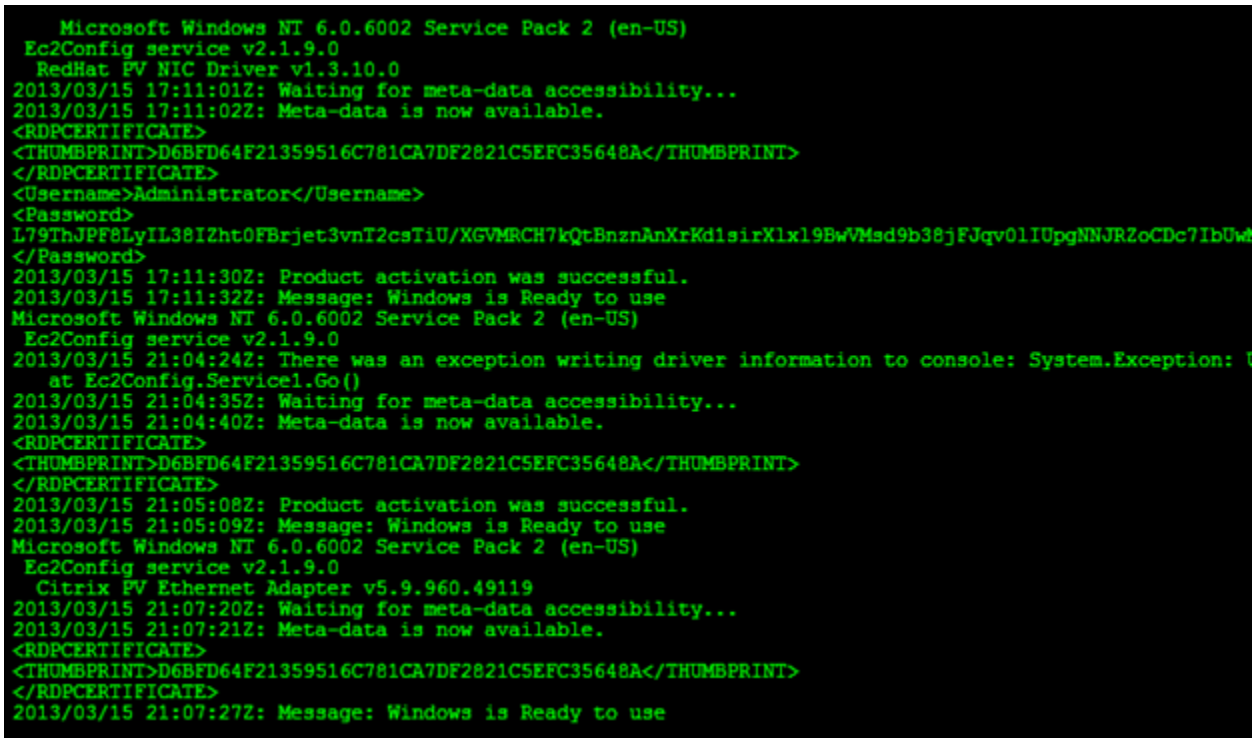
> **Note**
>
> アンインストーラのダイアログボックスが表示されない場合は、Windows タスクバーの [Red Hat Paravirtualize] を選択してください。

7. インスタンスが再起動して使用できる状態にあることを確認します。

a. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

b. [インスタンス] ページで、[アクション]、[モニタリングおよびトラブルシューティング] の順に選択し、[システムログの取得] をクリックしてください。

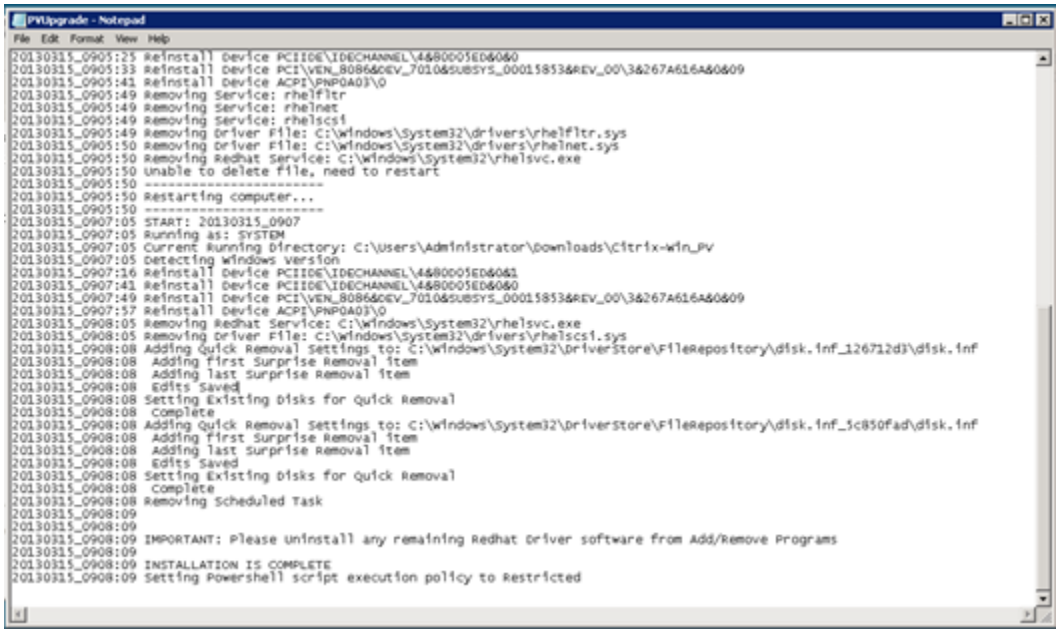
c. アップグレード操作では、サーバーが 3～4 回再起動します。何度再起動されたかは、ログで`Windows is Ready to use`が表示された回数で確認できます。



8. インスタンスに接続してローカル管理者としてログインします。

9. [Red Hat Paravirtualized Xen Drivers for Windows uninstaller] ダイアログボックスを閉じます。

10. インストールが完了したことを確認します。先ほどファイルを抽出した`Citrix-WIN_PV` フォルダに移動して、`PVUpgrade.log` ファイルを開き、`INSTALLATION IS COMPLETE`という文字列を確認します。



Citrix Xen ゲストエージェントサービスのアップグレード

Windows Server で Citrix PV ドライバーを使用している場合、Citrix Xen ゲストエージェントサービスをアップグレードできます。この Windows サービスは、API からのシャットダウンイベントや再起動イベントなどのタスクを処理します。インスタンスが Citrix PV ドライバーを実行している限り、いずれのバージョンの Windows Server でもこのアップグレードパッケージを実行できます。

> **Important**
>
> Windows Server 2008 R2 以降の場合は、Guest Agent の更新を含む AWS PV ドライバーにアップグレードすることをお勧めします。

ドライバーのアップグレードを始める前に、インスタンスにある重要な情報のバックアップを作成するか、インスタンスから AMI を作成します。AMI の作成の詳細については、Amazon EBS-backed AMI を作成するを参照してください。

> **Tip**
>
> Amazon EC2 コンソールから AMI を作成する代わりに、Systems Manager Automation を使用し、`AWS-CreateImage` ランブックを使用して AMI を作成できます。詳細については、

「AWS Systems Manager オートメーションランブックリファレンス」の「ユーザーガイド」の「AWS-CreateImage」を参照してください。

AMI を作成する場合は、事前に次のことを実行してください。

- EC2Config サービスで Sysprep ツールを有効にしていないことを確認する。
- パスワードを書き留める。
- イーサネットアダプタを DHCP に設定する。

Citrix Xen ゲストエージェントサービスをアップグレードするには

1. インスタンスに接続してローカル管理者としてログインします。インスタンスへの接続の詳細についてはRDP を使用した Windows インスタンスへの接続を参照してください。
2. インスタンスで、Citrix アップグレードパッケージをダウンロードします。
3. アップグレードパッケージを好きな場所に展開します。
4. Upgrade.bat ファイルをダブルクリックしてください。セキュリティ警告が表示された場合は、[実行] を選択してください。
5. [Upgrade Drivers] (ドライバのアップグレード) ダイアログボックスの内容を確認し、アップグレードを開始する場合は、[はい] を選択してください。
6. アップグレードが完了すると、PVUpgrade.log ファイルが開きます。UPGRADE IS COMPLETEという文字列が含まれているはずです。
7. インスタンスを再起動します。

Windows インスタンスでの PV ドライバーのトラブルシューティング

以下は、古い Amazon EC2 イメージと PV ドライバで発生する可能性のある問題の解決策です。

コンテンツ

- Windows Server 2012 R2 でインスタンスの再起動後にネットワークおよびストレージとの接続が失われる
- TCP オフロード
- 時刻同期
- 20,000 以上のディスク IOPS を使用するワークロードでは、CPU のボトルネックによりパフォーマンスが低下します

Windows Server 2012 R2 でインスタンスの再起動後にネットワークおよびストレージとの接続が失われる

> **Important**
>
> この問題は、2014 年 9 月より前に利用可能になった AMI でのみ発生します。

2014 年 9 月 10 日より前に使用可能になった Windows Server 2012 R2 Amazon Machine Image (AMI) では、インスタンスの再起動後にネットワークおよびストレージとの接続が失われることがあります。AWS マネジメントコンソール のシステムログに 「Difficulty detecting PV driver details for Console Output「 というエラーが見つかります。この接続損失は、プラグアンドプレイクリーンアップ機能によって発生します。これは、30 日ごとに非アクティブなシステムデバイスをスキャンして無効にする機能です。この機能は EC2 ネットワークデバイスを非アクティブと誤認して、システムから削除します。この状況になると、インスタンスの再起動後にネットワーク接続が失われます。

この問題の影響を受ける可能性のあるシステムに対して、インプレースドライバーアップグレードをダウンロードして実行できます。インプレースドライバーアップグレードを実行できない場合は、ヘルパースクリプトを実行できます。スクリプトはインスタンスがこの問題の影響を受けているかどうかを調べます。影響を受けていて、Amazon EC2 ネットワークデバイスが削除されていない場合、スクリプトはプラグアンドプレイクリーンアップスキャンを無効にします。ネットワークデバイスが削除されている場合、スクリプトはデバイスを修復し、プラグアンドプレイクリーンアップスキャンを無効にしてから、インスタンスを再起動して、ネットワーク接続を有効にします。

コンテンツ

- 問題の解決方法の選択
- 方法 1 - 拡張ネットワーキング
- 方法 2 - レジストリ設定
- 修復スクリプトの実行

問題の解決方法の選択

この問題の影響を受けたインスタンスへのネットワークおよびストレージ接続を復元する方法は 2 つあります。次のいずれかの方法を選択してください。

方法	前提条件	手順概要
方法 1 - 拡張ネットワーキング	拡張ネットワーキングは、C3 インスタンスタイプを必要とする Virtual Private Cloud (VPC) でのみ利用できます。サーバーが現在 C3 インスタンスタイプを使用していない場合は、一時的に変更する必要があります。	サーバーインスタンスタイプを C3 インスタンスに変更します。拡張ネットワーキングにより、影響を受けたインスタンスに接続して問題を修正できるようになります。問題を修正した後、インスタンスを元のインスタンスタイプに戻します。この方法は、通常は方法 2 よりすぐに終わり、ユーザーエラーが発生する可能性が低くなります。C3 インスタンスが実行されている限り追加料金が発生します。
方法 2 - レジストリ設定	2 番目のサーバーを作成したりアクセスしたりする知識。レジストリ設定を変更する知識。	影響を受けたインスタンスからルートボリュームをデタッチして別のインスタンスにアタッチし、接続してレジストリに変更を加えます。追加のサーバーが実行されている限り追加料金が発生します。この方法は、方法 1 より時間がかかりますが、この方法は方法 1 で問題が解決されない場合も効果がありました。

方法 1 - 拡張ネットワーキング

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. 影響のあるインスタンスを特定します。インスタンスを選択し、[インスタンスの状態] をクリックし、[インスタンスの停止] を選択してください。

> **Warning**
>
> インスタンスストアボリューム上のデータは、インスタンスを停止すると失われます。このデータを保持するには、永続ストレージにバックアップしてください。

4. インスタンスの停止後、バックアップを作成します。インスタンスを選択し、[アクション]、[イメージとテンプレート] の順に選択し、[イメージの作成] をクリックしてください。
5. インスタンスタイプを C3 インスタンスタイプに変更します。
6. インスタンスを起動します。
7. リモートデスクトップを使用してインスタンスに接続し、AWS PV ドライバーアップグレードパッケージをインスタンスにダウンロードします。
8. フォルダの内容を抽出し、`AWSPVDriverSetup.msi` を実行します。

   MSI の実行後、インスタンスは自動的に再起動され、ドライバーがアップグレードされます。インスタンスは最大 15 分間、使用できなくなります。
9. アップグレードが完了し、インスタンスが Amazon EC2 コンソールの両方のヘルスチェックに合格した後、リモートデスクトップを使用してインスタンスに接続し、新しいドライバーがインストールされたことを確認します。デバイスマネージャーの [Storage Controllers] で、[AWS PV Storage Host Adapter] を見つけます。ドライバーバージョンがドライバーのバージョン履歴の表に掲載されている最新バージョンと同じであることを確認します。詳細については、AWS PV ドライバーパッケージの履歴を参照してください。
10. インスタンスを停止し、インスタンスを元のインスタンスタイプに戻します。
11. インスタンスを起動し、標準の使用を再開します。

方法 2 - レジストリ設定

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. 影響のあるインスタンスを特定します。インスタンスを選択し、[インスタンスの状態]、[インスタンスの停止] の順に選択してください。

> **Warning**
>
> インスタンスストアボリューム上のデータは、インスタンスを停止すると失われます。このデータを保持するには、永続ストレージにバックアップしてください。

4. [インスタンスの起動] を選択し、影響のあるインスタンスと同じアベイラビリティーゾーンに Windows Server 2008 または Windows Server 2012 の一時インスタンスを作成します。Windows Server 2012 R2 インスタンスは作成しないでください。

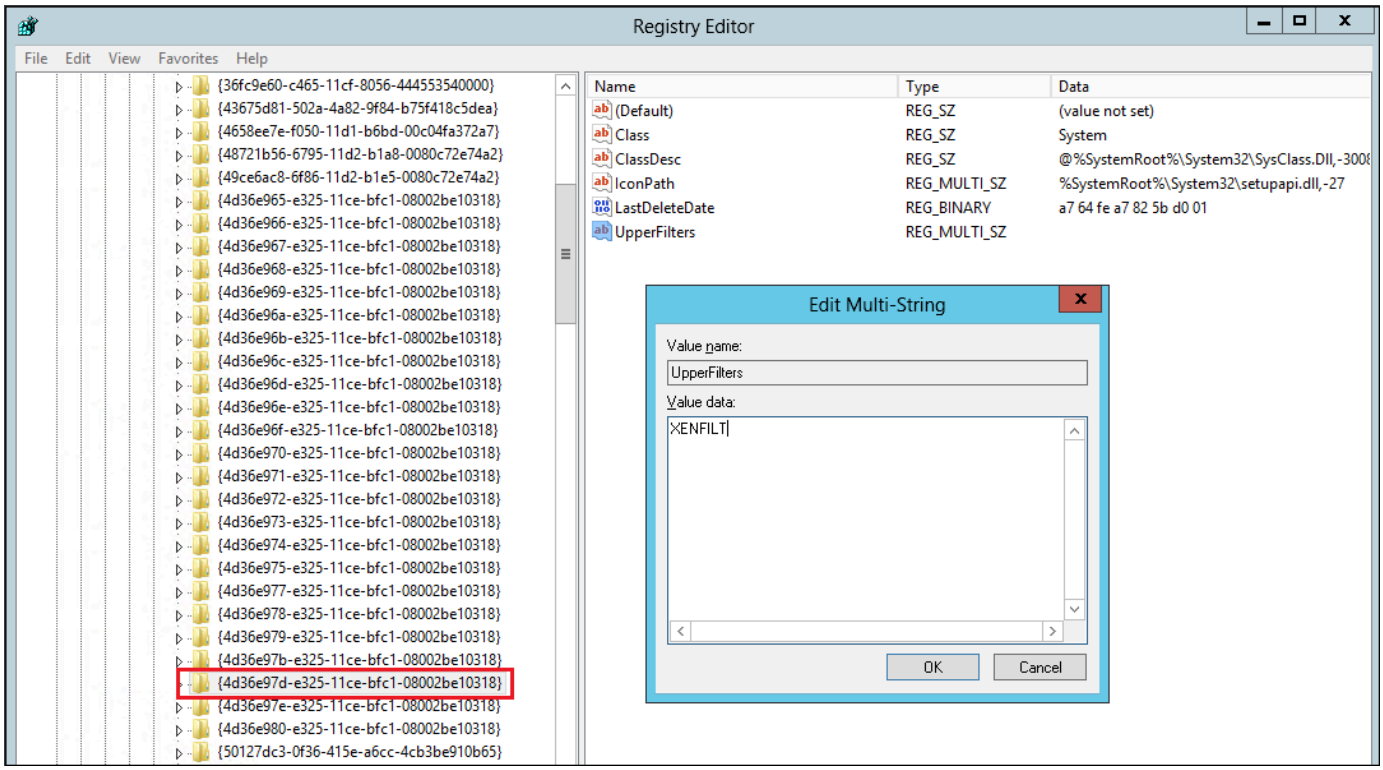
> **Important**
>
> 影響のあるインスタンスと同じアベイラビリティーゾーンにインスタンスを作成しない場合、影響のあるインスタンスのルートボリュームを新しいインスタンスにアタッチできません。

5. ナビゲーションペインの [Volumes] を選択してください。
6. 影響のあるインスタンスのルートボリュームを見つけます。ボリュームをデタッチし、先ほど作成した一時インスタンスにボリュームをアタッチします。デフォルトのデバイス名 (xvdf) でアタッチしてください。
7. リモートデスクトップを使用して一時インスタンスに接続したら、Disk Management ユーティリティを使用してボリュームを有効にします。
8. 一時インスタンスで、[実行] ダイアログボックスを開き、**`regedit`** と入力して、Enter キーを押します。
9. レジストリエディターのナビゲーションペインで、[HKEY_Local_Machine] を選択し、[File] メニューの [Load Hive] を選択してください。
10. [Load Hive] ダイアログボックスで、Affected Volume\Windows\System32\config\System に移動し、[Key Name] ダイアログボックスに一時的な名前を入力してください。例えば、OldSysと入力してください。
11. レジストリエディターのナビゲーションペインで、以下のキーを見つけます。

    HKEY_LOCAL_MACHINE***`your_temporary_key_name`**\ControlSet001\Control\Class\4d36e97d-e325-11ce-bfc1-08002be10318

    HKEY_LOCAL_MACHINE***`your_temporary_key_name`**\ControlSet001\Control\Class\4d36e96a-e325-11ce-bfc1-08002be10318

12. キーごとに、[UpperFilters] をダブルクリックし、XENFILT の値を入力して、[OK] を選択してください。



13. 以下のキーを見つけます。

    HKEY_LOCAL_MACHINE***your_temporary_key_name***\ControlSet001\Services\XENBUS\Parameters

14. ActiveDevice という名前と以下の値で新しい文字列 (REG_SZ) を作成します。

    PCI\VEN_5853&DEV_0001&SUBSYS_00015853&REV_01

15. 以下のキーを見つけます。

    HKEY_LOCAL_MACHINE***your_temporary_key_name***\ControlSet001\Services\XENBUS

16. [Count] を 0 から 1 に変更します。
17. 以下のキーを見つけ、削除します。

    HKEY_LOCAL_MACHINE***your_temporary_key_name***\ControlSet001\Services\xenvbd\StartOverride

    HKEY_LOCAL_MACHINE ***your_temporary_key_name***\ControlSet001\Services\xenfilt\StartOverride

18. レジストリエディターのナビゲーションペインで、最初にレジストリエディターを開いたときに作成した一時キーを選択してください。
19. [File] メニューの [Unload Hive] を選択してください。
20. Disk Management ユーティリティで、先ほどアタッチしたドライブを選択し、右クリックコンテキストメニューを開いて、[オフライン] を選択してください。
21. Amazon EC2 コンソールで、影響のあるボリュームを一時インスタンスからデタッチし、/dev/sda1 というデバイス名で Windows Server 2012 R2 インスタンスに再アタッチします。ボリュームをルートボリュームとして指定するには、このデバイス名を指定する必要があります。
22. インスタンスを起動します。
23. リモートデスクトップを使用してインスタンスに接続し、AWS PV ドライバーアップグレードパッケージをインスタンスにダウンロードします。
24. フォルダの内容を抽出し、`AWSPVDriverSetup.msi` を実行します。

    MSI の実行後、インスタンスは自動的に再起動され、ドライバーがアップグレードされます。インスタンスは最大 15 分間、使用できなくなります。
25. アップグレードが完了し、インスタンスが Amazon EC2 コンソールの両方のヘルスチェックに合格した後、リモートデスクトップを使用してインスタンスに接続し、新しいドライバーがインストールされたことを確認します。デバイスマネージャーの [Storage Controllers] で、[AWS PV Storage Host Adapter] を見つけます。ドライバーバージョンがドライバーのバージョン履歴の表に掲載されている最新バージョンと同じであることを確認します。詳細については、AWS PV ドライバーパッケージの履歴を参照してください。
26. この手順で作成した一時インスタンスを削除するか停止します。

修復スクリプトの実行

インプレースドライバアップグレードを実行できないか、新しいインスタンスに移行できない場合は、修正スクリプトを実行して、プラグアンドプレイクリーンアップタスクによって発生する問題を修正できます。

修復スクリプトを実行するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. 修復スクリプトを実行するインスタンスを選択してください。[インスタンスの状態] を選択し、[インスタンスの停止] をクリックしてください。

> **Warning**
>
> インスタンスストアボリューム上のデータは、インスタンスを停止すると失われます。このデータを保持するには、永続ストレージにバックアップしてください。

4. インスタンスの停止後、バックアップを作成します。インスタンスを選択し、[アクション]、[イメージとテンプレート] の順に選択し、[イメージの作成] をクリックしてください。
5. [インスタンスの状態] を選択し、[インスタンスの起動] をクリックしてください。
6. リモートデスクトップを使用してインスタンスに接続し、RemediateDriverIssue.zip フォルダをそのインスタンスにダウンロードします。
7. フォルダの内容を展開します。
8. Readme.txt ファイルの指示に従って、修復スクリプトを実行します。このファイルは、RemediateDriverIssue.zip を抽出したフォルダにあります。

TCP オフロード

> **Important**
>
> この問題は、AWS PV または Intel ネットワークドライバーを実行しているインスタンスには該当しません。

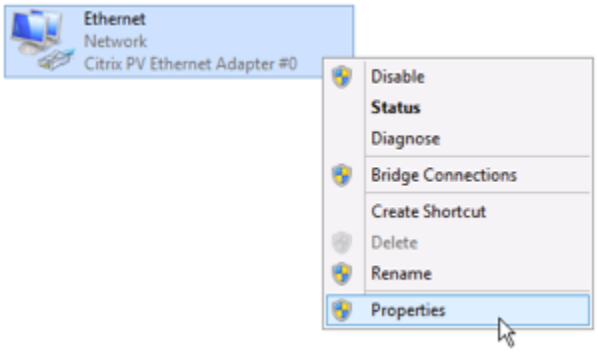
Windows AMI の Citrix PV ドライバーでは、TCP オフロードがデフォルトで有効になっています。例えば、特定の SQL ワークロードを実行しているときに、トランスポートレベルのエラーまたはパケット送信エラー (Windows パフォーマンス モニターで表示される) が発生した場合、この機能を無効にすることが必要になる場合があります。

> **Warning**
>
> TCP オフロードを無効にすると、インスタンスのネットワークパフォーマンスが低下することがあります。

Windows Server 2012 および 2008 の TCP オフロードを無効にするには

1. インスタンスに接続してローカル管理者としてログインします。

2. Windows Server 2012 を使用している場合は、[Ctrl+Esc] を押して [スタート] 画面にアクセスし、[コントロールパネル] を選択してください。Windows Server 2008 を使用している場合は、[スタート]、[コントロールパネル] の順に選択してください。
3. [Network and Internet] (ネットワークとインターネット)、[Network and Sharing Center] (ネットワークと共有センター) の順に選択してください。
4. [Change adapter settings] (アダプター設定の変更) を選択してください。
5. [Citrix PV Ethernet Adapter #0] を右クリックし、[プロパティ] を選択してください。



6. [Local Area Connection Properties] (ローカルエリア接続のプロパティ) ダイアログボックスで [設定] を選択すると、[Citrix PV Ethernet Adapter #0 Properties] ダイアログボックスが開きます。
7. [Advanced] (詳細) タブでプロパティをひとつずつ無効にします。ただし、[Correct TCP/UDP Checksum Value] は除きます。プロパティを無効にするには、[プロパティ] で選択してから、[値] の [無効] を選択してください。
8. [OK] を選択してください。
9. コマンドプロンプトウィンドウから次のコマンドを実行します。

```
netsh int ip set global taskoffload=disabled
netsh int tcp set global chimney=disabled
netsh int tcp set global rss=disabled
netsh int tcp set global netdma=disabled
```

10. インスタンスを再起動します。

時刻同期

2013.02.13 の Windows AMI のリリース以前、Citrix Xen ゲストエージェントでシステム時刻が誤って設定される場合がありました。これにより、DHCP リースが有効期限切れになる可能性がありま

す。インスタンスに接続できない問題が発生している場合、エージェントの更新が必要である可能性があります。

更新された Citrix Xen ゲストエージェントがあるかどうかを判断するには、`C:\Program Files \Citrix\XenGuestAgent.exe` ファイルが 2013 年 3 月のものであるかどうかを確認します。このファイルの日付がこれよりも前である場合は、Citrix Xen ゲストエージェントサービスを更新してください。詳細については、Citrix Xen ゲストエージェントサービスのアップグレードを参照してください。

20,000 以上のディスク IOPS を使用するワークロードでは、CPU のボトルネックによりパフォーマンスが低下します

この問題の影響を受けるのは、20,000 超の IOPS を利用する AWS PV ドライバーを実行している Windows インスタンスを使用しており、バグチェックコード `0x9E: USER_MODE_HEALTH_MONITOR` が発生した場合です。

AWS PV ドライバーでのディスクの読み取りと書き込み (IO) は、次の 2 つのフェーズで行われます: IO の準備および IO の完了です。デフォルトでは、準備フェーズは単一の任意のコアで実行されます。完了フェーズはコア `0` で実行されます。IO の処理に必要な計算量は、そのサイズやその他のプロパティによって異なります。準備フェーズでより多くの計算を使用する IO もあれば、完了フェーズで多くの計算を使用する IO もあります。インスタンスが 20,000 IOPS 以上の場合、準備フェーズまたは完了フェーズでボトルネックが発生し、インスタンスを実行する CPU の容量が 100% になる可能性があります。準備フェーズまたは完了フェーズがボトルネックになるかどうかは、アプリケーションが使用する IO のプロパティによって異なります。

AWS PV ドライバー 8.4.0 以降、準備フェーズと完了フェーズの負荷を複数のコアに分散し、ボトルネックを排除できます。各アプリケーションは、異なる IO プロパティを使用します。したがって、次の設定のいずれかを適用することで、アプリケーションのパフォーマンスを向上、低下させたり、または影響しないようにすることができます。これらの設定のいずれかを適用したら、アプリケーションを監視して、目的のパフォーマンスを満たしていることを確認します。

1. 前提条件

   このトラブルシューティング手順を開始する前に、以下の前提条件を確認してください。

   - インスタンスで AWS PV ドライバーのバージョン 8.4.0 以降を使用しています。アップグレードするには、EC2 Windows インスタンスでの PV ドライバーのアップグレードを参照してください。

- インスタンスへの RDP アクセス権があります。RDP を使用して Windows インスタンスに接続する手順については、RDP クライアントを使用して Windows インスタンスに接続するを参照してください。
- インスタンスに対する管理者アクセス権があります。

2. インスタンスの CPU 負荷を監視する

   Windows タスクマネージャーを使用して、各 CPU の負荷を表示し、ディスク IO の潜在的なボトルネックを特定できます。

   1. アプリケーションが実行中で、本番ワークロードと同様のトラフィックを処理していることを確認します。
   2. RDP を使用してインスタンスに接続します。
   3. インスタンスで [スタート] メニューを選択してください。
   4. [スタート] メニュー の `Task Manager` で、タスクマネージャーを開きます。
   5. タスクマネージャーに概要ビューが表示される場合は、[詳細] を選択して詳細ビューを展開します。
   6. [パフォーマンス] タブを選択してください。
   7. 左ペインの CPU を選択してください。
   8. メインペインでグラフを右クリックし、グラフをに変更する、論理プロセッサの順に選択して、個々のコアを表示します。
   9. インスタンスにあるコアの数により、時間の経過に伴い CPU 負荷の詳細が表示される場合と、数字のみが表示される場合があります。
      - 時間の経過に伴う負荷を示すグラフが表示される場合は、ボックスがほぼ完全に網掛けされている CPU を探します。
      - 各コアに数字が表示されている場合は、一貫して 95% 以上を示すコアを探します。

   10コア `0` または別のコアで、高い負荷がかかっているかどうかに注意してください。

3. 適用する設定を選択する

設定名	この設定を適用する場合	コメント
Default configuration	ワークロードが 20,000 IOPS 未満であるか、他の設定ではパフォーマンスや安定性が向上しませんでした。	この設定では、IO は少数のコアで発生するため、キャッシュの局所性を向上させ、コンテキストの切り替えを減ら

設定名	この設定を適用する場合	コメント
		すことで、より小さなワークロードで恩恵を受けることができます。
Allow driver to choose whether to distribute completion	ワークロードが 20,000 IOPS 以上で、コア 0 で中程度または高い負荷がかかっていることが確認できます。	この設定は、問題が発生したかどうかにかかわらず、PV 8.4.0 以降で 20,000 IOPS 以上を使用するすべての Xen インスタンスに推奨されます。
Distribute both preparation and completion	ワークロードが 20,000 IOPS 以上で、ドライバーがディストリビューションを選択できるようにしてもパフォーマンスが向上しなかったか、0 で高い負荷がかかっています。	この設定では、IO の準備および完了の両方を分散できます。

> **Note**
>
> IO の準備を分散せずに、IO の完了も分散しないことをお勧めします (NotifierDistributed を設定せずに DpcRedirection を設定します)。準備フェーズが並行して動作している場合、完了フェーズは準備フェーズによる過負荷に敏感になるためです。

レジストリキーの値

- NotifierDistributed

  値 0 または存在しない — 完了フェーズはコア 0 で実行されます。

  値 1 — ドライバーは、接続されたディスクごとに、完了フェーズ、コア 0、追加されたもう 1 つのコアのいずれかの実行を選択してください。

値 2 — ドライバーは、接続されたディスクごとに、追加された 1 つのコアで完了フェーズを実行します。

- DpcRedirection

  値 0 または存在しない — 準備フェーズは、単一の任意のコアで実行されます。

  値 1 — 準備フェーズは、複数のコアに分散されます。

デフォルト設定

AWS PV ドライバーのバージョンが 8.4.0 以前であるか、このセクションの他のいずれかの設定を適用した後にパフォーマンスまたは安定性の低下が見られる場合は、デフォルト設定を適用します。

1. RDP を使用してインスタンスに接続します。
2. 管理者として新しい PowerShell コマンドプロンプトを開きます。
3. 以下のコマンドを実行して、NotifierDistributed および DpcRedirection レジストリキーを削除します。

```
Remove-ItemProperty -Path HKLM:\System\CurrentControlSet\Services\xenvbd
\Parameters -Name NotifierDistributed
```

```
Remove-ItemProperty -Path HKLM:\System\CurrentControlSet\Services\xenvbd
\Parameters -Name DpcRedirection
```

4. インスタンスを再起動します。

完了を分散するかどうかをドライバーが選択できるようにする

NotiferDistributed レジストリキーを設定して、PV ストレージドライバーが IO の完了を分散するかどうかを選択できるようにします。

1. RDP を使用してインスタンスに接続します。
2. 管理者として新しい PowerShell コマンドプロンプトを開きます。

3. 以下のコマンドを実行して NotiferDistributed レジストリキーを設定します。

```
Set-ItemProperty -Type DWORD -Path HKLM:\System\CurrentControlSet\Services\xenvbd
\Parameters -Value 0x00000001 -Name NotifierDistributed
```

4. インスタンスを再起動します。

準備と完了の両方を分散する

NotifierDistributed および DpcRedirection レジストリキーを設定して、準備フェーズと完了フェーズの両方を常に分散します。

1. RDP を使用してインスタンスに接続します。
2. 管理者として新しい PowerShell コマンドプロンプトを開きます。
3. 以下のコマンドを実行して、NotifierDistributed および DpcRedirection レジストリキーを設定します。

```
Set-ItemProperty -Type DWORD -Path HKLM:\System\CurrentControlSet\Services\xenvbd
\Parameters -Value 0x00000002 -Name NotifierDistributed
```

```
Set-ItemProperty -Type DWORD -Path HKLM:\System\CurrentControlSet\Services\xenvbd
\Parameters -Value 0x00000001 -Name DpcRedirection
```

4. インスタンスを再起動します。

NVMe ドライバー

Amazon EBS ボリュームおよびインスタンスストアボリュームは、Nitro ベースのインスタンスで NVMe ブロックデバイスとして公開されます。NVMe ブロックデバイスとして公開されるボリュームで Amazon EBS 機能のパフォーマンスと機能を最大限に活用するには、インスタンスに NVMe AWS ドライバーがインストールされている必要があります。現行世代のすべての AWS Windows AMI には、デフォルトで NVMe ドライバーがインストールされています。AWS

EBS および NVMe の詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS および NVMe」を参照してください。SSD インスタンスストアと NVMe の詳細については、「EC2 インスタンスの SSD インスタンスストアボリューム」を参照してください。

Linux インスタンス

以下の AMI には、必要な NVMe ドライバーが含まれています。

- Amazon Linux 2
- Amazon Linux AMI 2018.03
- `linux-aws` カーネルを搭載した Ubuntu 14.04 以降

> **Note**
>
> AWS Graviton ベースのインスタンスタイプには`linux-aws` カーネル搭載の Ubuntu 18.04 以降が必要です

- Red Hat Enterprise Linux 7.4 以降
- SUSE Linux Enterprise Server 12 SP2 以降
- CentOS 7.4.1708 以降
- FreeBSD 11.1 以降
- Debian GNU/Linux 9 以降

インスタンスに NVMe ドライバーがあることを確認するには

次のコマンドを使用して、インスタンスに NVMe ドライバーがあることを確認できます。

- Amazon Linux、RHEL、CentOS、SUSE Linux Enterprise Server

```
$ modinfo nvme
```

  インスタンスに NVMe ドライバーがある場合、コマンドはそのドライバーに関する情報を返します。

- Amazon Linux 2、Ubuntu

```
$ ls /sys/module/ | grep nvme
```

  インスタンスに NVMe ドライバーがある場合、コマンドはインストールされたドライバーを返します。

NVMe ドライバーを更新するには

インスタンスに NVMe ドライバーがある場合は、次の手順を使用してドライバーを最新バージョンに更新できます。

1. インスタンスに接続します。
2. パッケージのキャッシュを更新し、必要なパッケージの更新を次のように取得します。

   - Amazon Linux 2、Amazon Linux、CentOS、Red Hat Enterprise Linux の場合:

     ```
     [ec2-user ~]$ sudo yum update -y
     ```

   - Ubuntu と Debian の場合:

     ```
     [ec2-user ~]$ sudo apt-get update -y
     ```

3. Ubuntu 16.04 以降には、`linux-aws` パッケージが含まれます。このパッケージには、Nitro ベースのインスタンスで必要な NVMe および ENA ドライバーが含まれます。最新バージョンにするには、次のように `linux-aws` パッケージにアップグレードします。

   ```
   [ec2-user ~]$ sudo apt-get install --only-upgrade -y linux-aws
   ```

   Ubuntu 14.04 の場合は、次のように最新の `linux-aws` パッケージをインストールできます。

   ```
   [ec2-user ~]$ sudo apt-get install linux-aws
   ```

4. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

   ```
   sudo reboot
   ```

5. 再起動後にインスタンスに再接続します。

Windows インスタンス

PowerShell

Amazon が提供する最新の AWS Windows AMI のいずれかからインスタンスを起動しなかった場合は、次の手順を使用してインスタンスに現在の AWS NVMe ドライバーをインストールします。このインストールには再始動が必要です。インストールスクリプトを入力するとインスタンスが再起動されますが、再起動されない場合には、最終段階で再起動する必要があります。

前提条件

- PowerShell バージョン 3.0 以降がインストールされている
- このセクションに示すコマンドは、PowerShell の 64 ビットバージョンで実行する必要があります。x86 バージョンの PowerShell を使用しないでください。これはシェルの 32 ビットバージョンで、これらのコマンドではサポートされていません。

最新の AWS NVMe ドライバーをダウンロードしてインストールするには

1. 変更を元に戻す必要がある場合に備えて、AMI をバックアップとして作成することをお勧めします。

   a. インスタンスを停止すると、インスタンスストアボリューム上のデータは消去されます。インスタンスを停止する前に、必要なデータをインスタンスストアボリュームから永続的ストレージ (Amazon EBS や Amazon S3 など) にコピーしていることを確認します。

   b. ナビゲーションペインで、[インスタンス] を選択してください。

   c. ドライバーのアップグレードが必要なインスタンスを選択し、[インスタンスの状態]、[インスタンスの停止] の順に選択してください。

   d. インスタンスを停止したら、インスタンスを選択し、[アクション]、[イメージとテンプレート]、[イメージの作成] の順に選択してください。

   e. [Instance state (インスタンスの状態)]、[Start instance (インスタンスの開始)] の順に選択してください。

2. インスタンスに接続してローカル管理者としてログインします。
3. 次のいずれかのオプションを使用して、インスタンスにドライバをダウンロードします。

   - ブラウザ - zip アーカイブを抽出するには、最新のドライバーパッケージをインスタンスにダウンロードします。
   - PowerShell – 次のコマンドを実行します。

     ```
     Invoke-WebRequest https://s3.amazonaws.com/ec2-windows-drivers-downloads/
     NVMe/Latest/AWSNVMe.zip -outfile $env:USERPROFILE\nvme_driver.zip
     Expand-Archive $env:userprofile\nvme_driver.zip -DestinationPath
      $env:userprofile\nvme_driver
     ```

     ファイルのダウンロード時にエラーが表示され、Windows Server 2016 以前のバージョンを使用している場合は、PowerShell ターミナルで TLS 1.2 を有効にする必要がある場

合があります。次のコマンドで現在の PowerShell セッションの TLS 1.2 を有効にしてから、もう一度試してください。

```
[Net.ServicePointManager]::SecurityProtocol =
 [Net.SecurityProtocolType]::Tls12
```

4. install.ps1 ディレクトリ nvme_driverから .\install.ps1 PowerShell スクリプトを実行し、ドライバーをインスタンスにインストールします。エラーが発生した場合は、PowerShell 3.0 以降を使用していることを確認してください。

   a. (オプション) AWS NVMe バージョン 1.5.0 以降、Windows Server 2016 以降ではスモールコンピュータシステムインターフェイス (SCSI) の永続予約がサポートされます。この機能により、共有 Amazon EBS ストレージによる Windows Server フェイルオーバークラスタリングのサポートが追加されました。デフォルトでは、この機能はインストール中には有効になっていません。

   install.ps1 スクリプトを実行してドライバーをインストールするときに、EnableSCSIPersistentReservations パラメータに $true という値を指定すると、この機能を有効にできます。

   ```
   PS C:\> .\install.ps1 -EnableSCSIPersistentReservations $true
   ```

   install.ps1 スクリプトを実行してドライバーをインストールするときに、EnableSCSIPersistentReservations パラメータに $false という値を指定すると、この機能を無効にできます。

   ```
   PS C:\> .\install.ps1 -EnableSCSIPersistentReservations $false
   ```

   b. AWS NVMe 1.5.0 以降、install.ps1 スクリプトは常に ebsnvme-id ツールをドライバーとともにインストールします。

   (オプション) バージョン 1.4.0、1.4.1、および 1.4.2 の場合、install.ps1 スクリプトを使用して ebsnvme-id ツールをドライバーとともにインストールするかどうかを指定できます。

      i. ebsnvme-id ツールをインストールするには、InstallEBSNVMeIdTool 'Yes' を指定します。

      ii. ツールをインストールしない場合は、InstallEBSNVMeIdTool 'No' を指定します。

InstallEBSNVMeIdTool を指定しない場合で、かつツールが既に C:\ProgramData\Amazon\Tools に存在している場合、パッケージはデフォルトでツールをアップグレードします。ツールが存在しない場合は、install.ps1 は、デフォルトではツールをアップグレードしません。

ツールをパッケージの一部としてインストールせず、後でインストールする場合は、ドライバーパッケージにツールの最新バージョンがあります。または、Amazon S3 からバージョン 1.0.0 をダウンロードすることもできます。

ebsnvme-id ツールをダウンロードします。

5. インストーラがインスタンスを再起動しない場合には、手動でインスタンスを再起動します。

Distributor

ディストリビューター (AWS Systems Manager の機能) を使用して、NVMe ドライバーパッケージを 1 回インストールすることも、スケジュールされた更新を使用してインストールすることもできます。

最新の AWS NVMe ドライバーをインストールする

1. ディストリビューターを使用して NVMe ドライバーパッケージをインストールする方法については、「Amazon EC2 Systems Manager ユーザーガイド」の「パッケージのインストールまたは更新」を参照してください。
2. [インストールタイプ] で、[アンインストールと再インストール] を選択します。
3. [名前] で、AWSNVMe を選択してください。
4. (任意) [追加の引数] では、値を指定することでインストールをカスタマイズできます。値は有効な JSON 構文を使用した形式にする必要があります。aws configure パッケージの追加の引数を渡す方法の例については、「コマンドドキュメントプラグインリファレンス」を参照してください。

   a. AWS NVMe 1.5.0 以降、このドライバーは Windows Server 2016 以降の SCSI 永続予約をサポートしています。デフォルトでは、この機能はインストール中には有効になっていません。

      - この機能を有効にするには、{"SSM_EnableSCSIPersistentReservations": "true"} を指定します。

- この機能を有効にしない場合は、{"SSM_EnableSCSIPersistentReservations": "false"} を指定します。

b. AWS NVMe 1.5.0 以降、install.ps1 スクリプトは常に ebsnvme-id ツールをインストールします。

(オプション) バージョン 1.4.0、1.4.1、および 1.4.2 の場合、install.ps1 スクリプトを使用して ebsnvme-id ツールをドライバーとともにインストールするかどうかを指定できます。

- ebsnvme-id ツールをインストールするには、{"SSM_InstallEBSNVMeIdTool": "Yes"} を指定します。
- ツールをインストールしない場合は、{"SSM_InstallEBSNVMeIdTool": "No"} を指定します。

もし[Additional Arguments] (追加の引数)で SSM_InstallEBSNVMeIdTool が指定されていない場合、かつツールがすでに C:\ProgramData\Amazon\Tools に存在している場合、パッケージはデフォルトでツールをアップグレードします。ツールが存在しない場合、パッケージはデフォルトでツールをアップグレードしません。

ツールをパッケージの一部としてインストールせず、後でインストールする場合は、ドライバーパッケージにツールの最新バージョンがあります。または、Amazon S3 からバージョン 1.0.0 をダウンロードすることもできます。

ebsnvme-id ツールをダウンロードします。

5. インストーラがインスタンスを再起動しない場合には、手動でインスタンスを再起動します。

Windows インスタンスの SCSI 永続予約を設定する

AWS NVMe ドライバーバージョン 1.5.0 以降をインストールした後は、Windows Server 2016 以降の Windows レジストリを使用して SCSI 永続予約を有効または無効にできます。これらのレジストリ変更を反映するにはインスタンスを再起動する必要があります。

SCSI 永続予約は、次のコマンドで EnableSCSIPersistentReservations の値を 1 に設定すると有効にできます。

```
PS C:\> $registryPath = "HKLM:\SYSTEM\CurrentControlSet\Services\AWSNVMe\Parameters
\Device"
Set-ItemProperty -Path $registryPath -Name EnableSCSIPersistentReservations -Value 1
```

SCSI 永続予約は、次のコマンドで EnableSCSIPersistentReservations の値を 0 に設定すると無効にできます。

```
PS C:\> $registryPath = "HKLM:\SYSTEM\CurrentControlSet\Services\AWSNVMe\Parameters
\Device"
Set-ItemProperty -Path $registryPath -Name EnableSCSIPersistentReservations -Value 0
```

AWS NVMe ドライバーのバージョン履歴

以下の表では、Amazon EC2 の Windows Server の各バージョンで実行する AWS NVMe ドライバーを示しています。

Windows サーバー バージョン	AWS NVMe ドライバーのバージョン
Windows Server 2025	最新バージョン
Windows Server 2022	最新バージョン
Windows Server 2019	最新バージョン
Windows Server 2016	最新バージョン
Windows Server 2012 R2	バージョン 1.5.1 以前
Windows Server 2012	バージョン 1.5.1 以前
Windows サーバー 2008 R2	バージョン 1.3.2 以前
Windows Server 2008	バージョン 1.3.2 以前

次の表に、AWS NVMe ドライバーのリリース済みバージョンを示します。

パッケージバージョン	ドライバーのバージョン	詳細	リリース日
1.8.2	1.8.2	• 割り込み処理を改善することで、起動時のバグチェックを修正しました。	2026 年 6 月 15 日
1.8.1	1.8.1	• ディスクシリアル番号生成での下位互換性問題を修正しました。	2026 年 2 月 24 日
1.8.0	1.8.0	• 動的名前空間管理を実装するデバイスで動作する IOCTL のサポートが追加されました。詳細については、Microsoft ドキュメントの「IOCTL_SCSI_MINIPORT IOCTL」を参照してください。 • Get Log Page コマンドとシリアル番号生成のバグ修正。	2026 年 1 月 16 日
1.7.0	1.7.0	• NVMe Get Log Page コマンドのサポートが追加されました。 • EBS および EC2 インスタンスストアボリュームの詳細なパフォーマンス統計のサポートが追加されました。 • Identify Namespace コマンドのバグが修正されました。	2025 年 9 月 17 日
1.6.0	1.6.0	• PnPUtil を使用するようにインストールスクリプトを更新しました。 • NVMe IOCTL を使用するように ebsnvme-id.exe を更新しました。	2024 年 10 月 25 日

パッケージバージョン	ドライバーのバージョン	詳細	リリース日
1.5.1	1.5.0	`ebsnvme-id` ツール用のフォルダーが存在しない場合に作成するようにインストールスクリプトを修正しました。	2023 年 11 月 17 日
1.5.0	1.5.0	Windows Server 2016 以降を実行しているインスタンス向けのスモールコンピュータシステムインターフェイス (SCSI) 永続予約のサポートが追加されました。ebsnvme-id ツール `ebsnvme-id.exe` がデフォルトでインストールされるようになりました。	2023 年 8 月 31 日
1.4.2	1.4.2	AWS NVMe ドライバー が D3 インスタンスのインスタンスストアボリュームをサポートしていなかったバグを修正しました。	2023 年 3 月 16 日
1.4.1	1.4.1	このオプションの NVMe 機能をサポートする EBS ボリュームの Namespace Preferred Write Granularity (NPGW) を報告します。詳細については、「NVMe Base Specification, version 1.4」のセクション 8.25「Improving Performance through I/O Size and Alignment Adherence」(I/O サイズおよびアライメント遵守によるパフォーマンスの向上) を参照してください。	2022 年 5 月 20 日

パッケージバージョン	ドライバーのバージョン	詳細	リリース日
1.4.0	1.4.0	• アプリケーションが NVMe デバイスとコミュニケーションを取るようにする IOCTL のサポートが追加されました。このサポートにより、アプリケーションは NVMe デバイスから `IdentifyController` 、`IdentifyNamespace` 、および `NameSpace` のリストを取得します。詳細については、Microsoft ドキュメントのプロトコル固有のクエリを参照してください。 • ドライバーバージョン 1.4.0 と最新の `ebsnvme-id` ツール `ebsnvme-id.exe` は、単一のパッケージにまとめられています。この組み合わせにより、ドライバとツールの両方を単一のパッケージからインストールできます。詳細については、NVMe ドライバーを参照してください。 • バグ修正と信頼性の向上。	2021 年 11 月 23 日
1.3.2	1.3.2	IO をアクティブに処理する EBS ボリュームを変更すると、データが破損する可能性がある問題を修正しました。オンライン EBS ボリューム (サイズ変更やタイプの変更など) を変更しないお客様は影響を受けません。 これは、Windows Server 2008 および 2008 R2 で実行できる最後のバージョンです。このバージョンはダウンロード可能ですが、サポートが終了しています。Windows Server 2008と2008 R2 はサポートライフサイクルが終了し、Microsoft によるサポートが終了しています。	2019 年 9 月 10 日
1.3.1	1.3.1	信頼性の向上。	2019 年 5 月 21 日

パッケージバージョン	ドライバーのバージョン	詳細	リリース日
1.3.0	1.3.0	デバイス最適化の向上。	2018 年 8 月 31 日
1.2.0	1.2.0	サポートされているすべてのインスタンス (例: ベアメタルインスタンス) での AWS NVMe デバイスのパフォーマンスと信頼性の向上	2018 年 6 月 13 日
>1.0.0	>1.0.0	Windows Server を実行するサポート対象インスタンスタイプ用の AWS NVMe ドライバー	2018 年 2 月 12 日

の通知のサブスクライブ

EC2 Windows ドライバーの新しいバージョンがリリースされたときには、Amazon SNS から通知を受け取ることができます。このような通知をサブスクライブするには、以下の手順を使用します。

コンソールから EC2 の通知にサブスクライブするには

1. Amazon SNS コンソールのhttps://console.aws.amazon.com/sns/v3/homeを開いてください。
2. ナビゲーションバーで、必要に応じて、リージョンを [米国東部 (バージニア北部)] に変更します。購読する SNS 通知がこのリージョンにあるため、このリージョンを選択する必要があります。
3. ナビゲーションペインで [Subscriptions] を選択してください。
4. [Create subscription] を選択してください。
5. [Create subscription] ダイアログボックスで、次の操作を行います。
   a. [TopicARN] では、次の Amazon リソースネーム (ARN) をコピーします。

      arn:aws:sns:us-east-1:801119661308:ec2-windows-drivers
   b. [プロトコル] で `Email` を選択してください。
   c. [エンドポイント] では、通知を受信するために使用できる E メールアドレスを入力してください。
   d. [Create subscription] を選択してください。
6. 確認メールが送信されます。E メールを開き、指示に従ってサブスクリプションを完了します。

サブスクライバには、EC2 Windows ドライバーの新しいバージョンがリリースされるたびに、通知が送信されます。通知が不要になった場合は、次の手順で受信登録を解除します。

Amazon EC2 Windows ドライバー通知から受信登録を解除するには

1. Amazon SNS コンソールのhttps://console.aws.amazon.com/sns/v3/homeを開いてください。
2. ナビゲーションペインで [Subscriptions] を選択してください。
3. 任意のサブスクリプションのチェックボックスを選択し、[アクション]、[サブスクリプションの削除] を選択してください。確認を求めるメッセージが表示されたら、[削除] を選択してください。

AWS CLI を使用して EC2 の通知をサブスクライブするには

AWS CLI で EC2 の通知をサブスクライブするには、次のコマンドを使用します。

```
aws sns subscribe --topic-arn arn:aws:sns:us-east-1:801119661308:ec2-windows-drivers --
protocol email --notification-endpoint YourUserName@YourDomainName.ext
```

AWS Tools for Windows PowerShell を使用して EC2 の通知をサブスクライブするには

AWS Tools for Windows PowerShell で EC2 の通知をサブスクライブするには、次のコマンドを使用します。

```
Connect-SNSNotification -TopicArn 'arn:aws:sns:us-east-1:801119661308:ec2-windows-
drivers' -Protocol email -Region us-east-1 -Endpoint 'YourUserName@YourDomainName.ext'
```

その他の Windows デバイスドライバー

AWS VMClock デバイスは、Nitro ベースのインスタンスに存在する仮想クロックデバイスです。デバイスは、[システムデバイス] の下のデバイスマネージャーに `AWS VMClock Device` として表示されます。AWS は、オペレーティングシステムがデバイスを認識するための Windows 用の null (機能なし) ドライバーパッケージを提供します。

サポートされているインスタンス

AWS VMClock デバイスは Nitro ベースのインスタンスタイプで使用できます。Nitro ベースのインスタンスのリストについては、「Nitro ベースのインスタンス」を参照してください。

サポートされるオペレーティングシステム

ドライバーパッケージは、次のバージョンの Windows Server をサポートしています。

- Windows Server 2016
- Windows Server 2019
- Windows Server 2022
- Windows Server 2025

ドライバーのインストール

ドライバーがインストールされていないことを示す警告アイコンが AWS VMClock デバイスのデバイスマネージャーに表示される場合、null ドライバーパッケージを手動でインストールできます。

AWS VMClock ドライバーをインストールするには

1. Amazon S3 からドライバーパッケージをダウンロードして抽出します。

```
Invoke-WebRequest https://s3.amazonaws.com/ec2-windows-drivers-downloads/
AWSVMClock/Latest/AWSVMClock.zip -OutFile $env:USERPROFILE\AWSVMClock.zip
Expand-Archive $env:USERPROFILE\AWSVMClock.zip -DestinationPath $env:USERPROFILE
\AWSVMClock
```

2. pnputil を使用してドライバーをインストールします。

```
pnputil /add-driver $env:USERPROFILE\AWSVMClock\aws_vmclock.inf /install
```

3. デバイスマネージャーでデバイスのステータスを確認します。インスタンスを再起動する必要はありません。

Amazon EC2 Windows インスタンスの設定

Windows インスタンスの起動後は、管理者として ログインして Windows の機能とシステム設定の追加設定を実行できます。EC2 Windows トラブルシューティングユーティリティ は、インスタンス上の問題のトラブルシューティングに役立ちます。

Windows 起動エージェントおよびその他の Windows 固有の機能は、次のように設定できます。

Windows 起動エージェント

各 AWS Windows AMI (および AWS Marketplace で使用可能な他の多くの AMI) には、デフォルト設定で事前設定された Windows 起動エージェントが含まれています。起動エージェントにより、インスタンスのスタートアップ時にタスクが実行されます。インスタンスが停止してその後起動された場合、または再起動された場合も、タスクが実行されます。

Windows の EC2 Fast Launch

すべての Amazon EC2 Windows インスタンスで、Windows オペレーティングシステム (OS) の標準の起動手順を実行する必要があります。この手順では再起動が複数回行われ、完了までに 15 分以上かかることがあります。EC2 Fast Launch 機能が有効になっている Amazon EC2 Windows Server AMI では、インスタンスの起動にかかる時間を短縮するために、これらのステップの一部を完了して事前に再起動します。

Windows 固有のシステム設定

次のリストには、Windows オペレーティングシステムにのみ適用されるいくつかのシステム設定が含まれています。

Windows 管理者のパスワードの変更

Windows インスタンスに接続するときは、そのインスタンスにアクセスする権限を持つユーザーアカウントとパスワードを指定する必要があります。インスタンスに初めて接続するときは、管理者アカウントを使用し、デフォルトのパスワードを入力する必要があります。初めてインスタンスに接続するときは、管理者パスワードをデフォルトの値から変更することをお勧めします。

Windows システムコンポーネントの追加

Windows Server オペレーティングシステムには、多数のオプションコンポーネントが含まれます。それぞれの AWS Windows Server AMI にすべてのオプションコンポーネントを含めることは現実的ではありません。代わりに、Windows インスタンスにコンポーネントを設定、またはインストールするのに必要なファイルを持つインストールメディアの EBS スナップショットを提供します。

Windows に WSL をインストールする

Windows Subsystem for Linux (WSL) は無料でダウンロードでき、Windows インスタンスにインストールできます。WSL をインストールすると、ネイティブ Linux コマンドラインツールを Windows インスタンス上で直接実行し、従来の Windows デスクトップと並行して、スクリプ

ト作成のために Linux ツールを使用できるようになります。単一の Windows インスタンス上で Linux と Windows を簡単に切り替えることができるため、開発環境で役立つ場合があります。

AWS Windows インスタンス用 NVMe ドライバー

Windows インスタンスの AWS デバイスドライバーが更新できます。詳細については、「the section called “デバイスドライバーの管理”」を参照してください。

次の表は、ニトロベースのインスタンスでサポートされているドライバーを Windows バージョン別にまとめたものです。

バージョン	ストレージドライバー	拡張ネットワーキングドライバー
Windows Server 2025	AWS 最新バージョン	ENA 最新バージョン
Windows Server 2022	AWS 最新バージョン	ENA 最新バージョン
Windows Server 2019	AWS 最新バージョン	ENA 最新バージョン
Windows Server 2016	AWS 最新バージョン	ENA 最新バージョン
Windows Server 2012 R2	AWSNVMe バージョン 1.5.1	ENA バージョン 2.6.0
Windows サーバー 2008 R2	AWSNVMe バージョン 1.3.2	ENA バージョン 2.2.3

次の表は、Xen ベースのインスタンスでサポートされているドライバーを Windows バージョン別にまとめたものです。

バージョン	ストレージドライバー	拡張ネットワーキングドライバー
Windows Server 2022	AWS PV 最新バージョン	• ENA 最新バージョン [1] • Intel VF [2] • AWS PV 最新バージョン[3]
Windows Server 2019	AWS PV 最新バージョン	

バージョン	ストレージドライバー	拡張ネットワーキングドライバー
		• ENA 最新バージョン [1] • Intel VF [2] • AWS PV 最新バージョン[3]
Windows Server 2016	AWS PV 最新バージョン	• ENA 最新バージョン [1] • Intel VF [2] • AWS PV 最新バージョン[3]
Windows Server 2012 R2	AWS PV バージョン 8.4.3	• ENA バージョン 2.6.0 [1] • Intel VF [2] • AWSPV バージョン 8.4.3 [3]
Windows サーバー 2008 R2	AWS PV バージョン 8.3.5	• ENA バージョン 2.2.3 [1] • Intel VF [2] • AWS PV バージョン 8.3.5 [3]

[1] インスタンスタイプ G3、H1、I3、`m4.16xlarge`、 P3, P3dn、および R4 の場合。

[2] インスタンスタイプ C3、C4、D2、I2、M4 (`m4.16xlarge` を除く)、および R3 の場合。

[3] インスタンスタイプ C1、M1、M2、M3、T1、T2、X1、X1e の場合。

Amazon EC2 Windows インスタンス上の Windows 起動エージェント

各 AWS Windows AMI には、デフォルト設定で事前設定された Windows 起動エージェントが含まれています。起動エージェントにより、インスタンスのスタートアップ時にタスクが実行されます。インスタンスが停止してその後起動された場合、または再起動された場合も、タスクが実行されます。特定のエージェントの詳細については、次のリストの詳細ページを参照してください。

AWS Windows AMI の詳細については、AWS Windows AMI リファレンスを参照してください。

- EC2 Windows インスタンスの起動時に EC2Launch v2 エージェントを使用してタスクを実行する

- EC2 Windows インスタンスの起動時に EC2Launch v1 エージェントを使用してタスクを実行する
- EC2Config サービスを使用して、EC2 レガシー Windows オペレーティングシステムインスタンスの起動時にタスクを実行する

コンテンツ

- Amazon EC2 起動エージェントを比較する
- EC2 Windows 起動エージェントの DNS サフィックスを設定する
- EC2 Windows 起動エージェント通知をサブスクライブする
- EC2Launch v2 および EC2Config エージェントの Windows サービス管理

Amazon EC2 起動エージェントを比較する

次の表に、EC2Config、EC2Launch v1、EC2Launch v2 の機能の主な相違点を示します。

機能	EC2Config	EC2Launch v1	EC2Launch v2
として実行	Windows サービス	PowerShell スクリプト	Windows サービス
サポート対象	レガシー OS のみ	Windows Server バージョン • 2016 • 2019 (LTSC および SAC)	Windows Server バージョン • 2016 • 2019 (LTSC および SAC) • 2022 • 2025
設定ファイル	XML	JSON	JSON/YAML
管理者のユーザー名を設定	いいえ	なし	あり
圧縮ユーザーデータ	なし	なし	あり

機能	EC2Config	EC2Launch v1	EC2Launch v2
AMI で作成されたローカルユーザーデータ	いいえ	いいえ	はい、設定可能です
ユーザーデータでのタスク設定	いいえ	なし	はい
設定可能な壁紙	いいえ	なし	あり
タスクの実行順序をカスタマイズ	なし	なし	はい
設定可能なタスク	15	9	20 (起動時)
Windows イベントビューワーのサポート	はい	なし	はい
イベントビューワーのイベントタイプの数	2	0	30

> **Note**
>
> EC2Config のドキュメントは、履歴を参照するためにのみ提供されています。実行されているオペレーティングシステムのバージョンは、Microsoft ではサポートされなくなりました。最新の起動サービスにアップグレードすることを強くお勧めします。

EC2 Windows 起動エージェントの DNS サフィックスを設定する

Amazon EC2 起動エージェントを使用すると、Windows インスタンスがドメイン名の解決に使用する DNS サフィックスのリストを設定できます。起動エージェントは、DNS サフィックスの検索リストに次の値を追加することにより、`System\CurrentControlSet\Services\Tcpip\Parameters\SearchList` レジストリキーの標準 Windows 設定を上書きします。

- インスタンスのドメイン
- インスタンスドメインの継承に起因するサフィックス
- NV ドメイン
- 各ネットワークインターフェイスカードにより指定されたドメイン

DNS サフィックス設定は、すべての起動エージェントでサポートされています。詳細については、特定の起動エージェントのバージョンを参照してください。

- `setDnsSuffix` タスク、および EC2Launch v2 で DNS サフィックスを設定する方法の詳細については、「setDnsSuffix」を参照してください。
- DNS サフィックスのリストの設定、および EC2Launch v1 の継承を有効または無効にする方法については、「Windows インスタンスで EC2Launch v1 エージェントを設定する」を参照してください。
- DNS サフィックスのリストの設定、および EC2Config の継承を有効または無効にする方法については、「EC2Config の設定ファイル」を参照してください。

ドメイン名の継承

ドメイン名の継承は Active Directory の動作であり、子ドメインのコンピュータが、完全修飾ドメイン名を使用せずに親ドメインのリソースにアクセスすることを可能にします。デフォルトでは、ドメイン名の継承は、ドメイン名の進行でノードが残り 2 つになるまで続きます。

インスタンスがドメインに接続されている場合、起動エージェントはドメイン名の継承を実行し、その結果を、**`System\CurrentControlSet\Services\Tcpip\Parameters\SearchList`**レジストリキーに保持されている DNS サフィックスの検索リストに結果を追加します。エージェントは、次のレジストリキーの設定を使用して継承の動作を判定します。

- **`System\CurrentControlSet\Services\Tcpip\Parameters\UseDomainNameDevolution`**
  - 設定されていない場合、継承は無効になります
  - `1` に設定されている場合、継承は有効になります (デフォルト)
  - `0` に設定されている場合、継承は無効になります
- **`System\CurrentControlSet\Services\Dnscache\Parameters\DomainNameDevolutionLevel`**
  - 設定されていない場合、2 のレベルを使用します (デフォルト)

- 3 以上に設定されている場合、値を使用してレベルを設定します

継承を無効にするか、継承の設定をより高いレベルに変更すると、System\CurrentControlSet\Services\Tcpip\Parameters\SearchList レジストリキースタイルには、以前に追加されたサフィックスが含まれます。これらは自動的には削除されません。リストは手動で更新できます。また、削除して、新しいリストを設定するプロセスをエージェントに実行させることもできます。

> **Note**
>
> レジストリから DNS サフィックスリストを削除するには、次のコマンドを実行します。
>
> ```
> PS C:\> Invoke-CimMethod -ClassName Win32_NetworkAdapterConfiguration -
> MethodName "SetDNSSuffixSearchOrder" -Arguments @{ DNSDomainSuffixSearchOrder =
>  $null } | Out-Null
> ```

継承の例

次の例は、継承のプロセスにおけるドメイン名の進行状況を示したものです。

corp.example.com

- example.com への進行

locale.region.corp.example.com

1. region.corp.example.com への進行
2. corp.example.com への進行
3. example.com への進行

locale.region.corp.example.com、設定は DomainNameDevolutionLevel=3

1. region.corp.example.com への進行
2. corp.example.com への進行 レベル設定により、進行はここで停止します。

EC2 Windows 起動エージェント通知をサブスクライブする

EC2 起動エージェントの新しいバージョンがリリースされた際に、Amazon SNS から通知を受け取ることができます。このような通知をサブスクライブするには、以下の手順を使用します。

EC2Config の通知をサブスクライブする

1. Amazon SNS コンソールのhttps://console.aws.amazon.com/sns/v3/homeを開いてください。
2. ナビゲーションバーで、必要に応じて、リージョンを [米国東部 (バージニア北部)] に変更します。サブスクライブする SNS 通知がこのリージョンで作成されているため、このリージョンを選択する必要があります。
3. ナビゲーションペインで [Subscriptions] を選択してください。
4. [Create subscription] を選択してください。
5. [Create subscription] ダイアログボックスで、次の操作を行います。
   a. [トピック ARN] には、通知を受信するエージェントと一致する次の Amazon リソースネーム (ARN) を使用します。
      - EC2Launch v2:

        ```
        arn:aws:sns:us-east-1:309726204594:amazon-ec2launch-v2
        ```

      - EC2Launch または EC2Config:

        ```
        arn:aws:sns:us-east-1:801119661308:ec2-windows-ec2config
        ```

   b. [プロトコル] で [Email] を選択してください。
   c. [エンドポイント] には、通知を受信する E メールアドレスを入力してください。
   d. [Create subscription] を選択してください。
6. サブスクリプションの確認を求めるメールが届きます。E メールを開き、指示に従ってサブスクリプションを完了します。

サブスクライバーには、起動エージェントの新しいバージョンがリリースされるたびに、通知が送信されます。通知が不要になった場合は、次の手順で受信登録を解除します。

起動エージェントの通知のサブスクリプションを解除する

1. Amazon SNS コンソールを開きます。

2. ナビゲーションペインで [Subscriptions] を選択してください。
3. サブスクリプションを選択し、[アクション]、[サブスクリプションの削除] を選択してください。確認を求めるメッセージが表示されたら、[削除] を選択してください。

EC2Launch v2 および EC2Config エージェントの Windows サービス管理

管理者権限を持つユーザーとしてインスタンスにログインしている場合は、他の Windows サービスと同様に EC2Launch v2 および EC2Config 起動エージェントを管理できます。EC2Launch v1 は、デフォルトでスケジュールされたタスクによって管理される一連の PowerShell スクリプトです。このセクションでは、EC2Launch v2 および EC2Config のサービス管理について説明します。

インスタンスに更新された設定を適用するには、サービス用の Microsoft 管理コンソールの MMC) インターフェイスから EC2Launch v2 エージェントまたは EC2Config サービス起動エージェントを停止して再起動します。同様に、新しいバージョンの起動エージェントをインストールする場合は、まずエージェントを停止し、インストールが完了したら再起動する必要があります。

> **Note**
>
> これらのアクションを選択するには、管理者として MMC サービスインターフェイスを開く必要があります。これを行うには、コンテキストメニューから [管理者として実行] を選択してください。または、キーボードを使用してインターフェイスを開く場合は、次の手順に従います。
>
> 1. [Tab] キーまたは矢印キーを使用して、[管理ツール] メニューから [サービス] メニュー項目を選択してください。
> 2. 管理者として開くには、キーボードの組み合わせ [`Ctrl` + `Shift` + `Enter`] を使用します。

次の手順では、インスタンスで起動エージェントを停止および開始するステップを示します。

起動エージェントを停止する

1. Windows インスタンスを起動して接続します。
2. Windows の [スタート] メニューから [管理ツール] を選択してください。
3. このセクションの冒頭で説明されているように、管理者として [サービス] コンソールを開きます。

4. サービスのリストで、インスタンスで実行されているエージェント ([EC2Launch] または [EC2Config]) を選択し、[アクション] メニューから [停止] を選択してください。または、コンテキストメニューを使用してエージェントを停止することもできます。

起動エージェントを再起動する

1. Windows インスタンスを起動して接続します。
2. Windows の [スタート] メニューから [管理ツール] を選択してください。
3. このセクションの冒頭で説明されているように、管理者として [サービス] コンソールを開きます。
4. サービスのリストで、インスタンスで実行されているエージェント ([EC2Launch] または [EC2Config]) を選択し、[アクション] メニューから [開始] または [再起動] を選択してください。または、コンテキストメニューを使用してエージェントを再起動することもできます。

構成設定を更新したり、独自の AMI を作成したり、AWS Systems Manager を使用する必要がなければ、起動エージェントは削除またはアンインストールできます。

[Delete] (削除)

サービスを削除するとレジストリのサブキーも削除されます。

アンインストール

サービスをアンインストールすると、ファイル、レジストリのサブキー、サービスへのショートカット (ある場合) が削除されます。

起動エージェントを削除する

1. Windows インスタンスを起動して接続します。
2. Windows コマンドプロンプトウィンドウを起動します。
3. 次のいずれかのコマンドを実行して起動エージェントを削除します。

   - 次のコマンドを実行して、EC2Launch または EC2Launch v2 を削除します。

     ```
     sc delete ec2launch
     ```

   - EC2Config サービスを削除するには、次のコマンドを実行します。

```
sc delete ec2config
```

起動エージェントをアンインストールする

1. Windows インスタンスを起動して接続します。
2. [Windows システム] を選択し、Windows の [スタート] メニューから [コントロールパネル] を選択してください。
3. [プログラムと機能] を選択して、インスタンスにインストールされているプログラムのリストを開きます。
4. リストから起動エージェント ([Amazon EC2Launch] または [EC2ConfigService]) を選択し、[ファイル] メニューから [アンインストール] を選択してください。または、コンテキストメニューを使用できます。

> **Note**
>
> インストールされている起動エージェントのバージョンは、[バージョン] 列で確認できます。

EC2 Windows インスタンスの起動時に EC2Launch v2 エージェントを使用してタスクを実行する

AWS Windows Server 2022 および Windows Server 2025 AMI から起動される Amazon EC2 のサポートされたすべてのインスタンスには、EC2Launch v2 起動エージェント `EC2Launch.exe`がデフォルトで含まれています。また、デフォルトの起動エージェントとして EC2Launch v2 がインストールされた、Windows Server 2016 および 2019 AMI も提供されています。これらの AMI は、EC2Launch v1 を含む Windows Server 2016 および 2019 AMI に追加して提供されています。デフォルトで EC2Launch v2 を含む Windows AMI を検索するには、Amazon EC2 コンソールの AMI ページで、次のプレフィクスを入力して検索を行います。`EC2LaunchV2-Windows_Server-*`

起動エージェントのバージョン機能を比較するには、「Amazon EC2 起動エージェントを比較する」を参照してください。

EC2Launch v2 により、インスタンスの起動時にタスクが実行されます。加えて、インスタンスが停止後に起動された場合、または再起動された場合にもタスクが実行されます。EC2Launch v2 では、オンデマンドでタスクを実行させることもできます。タスクには自動的に有効化されるものもありま

すが、手動で有効化しなければならないものもあります。EC2Launch v2 サービスは、EC2Config と EC2Launch のすべての機能をサポートしています。

このサービスは、設定ファイルを使用してオペレションを制御します。設定ファイルを更新するには、グラフィカルツールを使用するか、これを単一の .yml ファイル `agent-config.yml`として直接編集できます。ファイルロケーションの詳細については、「EC2Launch v2 のディレクトリ構造」を参照してください。

EC2Launch v2 から発行される Windows イベントログを、エラーのトラブルシューティングやトリガーの設定に使用できます。詳細については、「Windows イベントログ」を参照してください。

サポートされる OS のバージョン

EC2Launch v2 エージェントでは、次の Windows Server オペレーティングシステム (OS) バージョンがサポートされます。

- Windows Server 2025
- Windows Server 2022
- Windows Server 2019 (長期サービスチャネルおよび半期チャネル)
- Windows Server 2016

デフォルトで実行されるタスク

EC2Launch v2 エージェントは、インスタンスの初回開始時に、デフォルトで次のタスクを 1 回だけ実行します。タスクは、開始ステージ内で実行される順序に基づいて整理されます。

`Boot` ステージ

- extendRootPartition

`PreReady` ステージ

- activateWindows
- setDnsSuffix
- setAdminAccount
- setWallpaper

`PostReady` ステージ

- startSsm

EC2Launch v2 の概念

次に示す各概念は、EC2Launch v2 の使用を考慮する際の理解に役立ちます。

agent-config

`agent-config` ファイルは、EC2Launch v2 の設定フォルダに置かれています。これには、Boot、Network、PreReady、PostReady の各ステージの設定が含まれます。このファイルを使用して、AMI の初回起動時または後続の起動時に実行するタスクのインスタンスの設定を指定します。

デフォルトでは、EC2Launch v2 のインストール時にインストールされる `agent-config` ファイルに、標準の Amazon Windows AMI で使用される推奨設定が含まれています。設定ファイルを修正すると、EC2Launch v2 が指定する AMI での、デフォルトのブート処理を変更できます。ファイルロケーションの詳細については、「EC2Launch v2 のディレクトリ構造」を参照してください。

頻度

タスク頻度は、ブートコンテキストに応じて、いつタスクを実行するか決定されます。ほとんどのタスクで許可される頻度は 1 つだけです。`executeScript` タスクの頻度を指定できます。

EC2Launch v2 タスクの設定 には次の頻度が表示されます。

- 1 回 — タスクは、AMI の初回起動時 (Sysprep の終了時) に 1 回実行されます。
- 常時 — タスクは起動エージェントが実行されるたびに実行されます。起動エージェントは、以下の場合に実行されます。
  - インスタンスの起動または再起動
  - EC2Launch サービスの実行
  - `EC2Launch.exe run` の呼び出し

ステージ

ステージとは、EC2Launch v2 エージェントが実行するタスクを論理的にグループ化したものです。一部のタスクは、特定のステージでのみ実行できます。その他は複数のステージで実行できます。`agent-config.yml` を使用するときは、ステージのリストと各ステージ内のタスクのリストを指定する必要があります。

このサービスは、次の順序で実行されます。

ステージ 1: ブート

ステージ 2: ネットワーク

ステージ 3: PreReady

Windowsの準備ができました

PreReady ステージが完了すると、サービスは Amazon EC2 コンソールに `Windows is ready` メッセージを送信します。

ステージ 4: PostReady

ユーザーデータは PostReady ステージで実行されます。スクリプトバージョンには、次のように `agent-config.yml` file PostReady ステージの前に実行されるものと後に実行されるものがあります。

`agent-config.yml` 前

- YAML ユーザーデータバージョン 1.1
- XML ユーザーデータ

`agent-config.yml` 後

- YAML ユーザーデータバージョン 1.0 (後方互換性のためのレガシーバージョン)

ステージとタスクの例については、「例:`agent-config.yml`」を参照してください。

ユーザーデータを使用する場合は、起動エージェントが実行するタスクのリストを指定する必要があります。ステージは暗黙に示しています。例については、「例: ユーザーデータ」を参照してください。

EC2Launch v2は、`agent-config.yml` およびユーザーデータで指定した順序でタスクのリストを実行します。ステージは順番に実行されます。次のステージは、前のステージが完了した後に開始されます。タスクも順番に実行されます。

タスク

インスタンスに対してアクションを実行するために、タスクを呼び出すことができます。タスクは、`agent-config.yml` ファイルまたはユーザーデータで設定できます。EC2Launch v2 で使用可能なタスクのリストについては、「EC2Launch v2 のタスク」を参照してください。タスク設定スキーマの詳細については、「EC2Launch v2 タスクの設定」を参照してください。

ユーザーデータ

ユーザーデータは、インスタンスの起動時に設定できるデータです。ユーザーデータを更新して、カスタム AMI やクイックスタート AMI の設定を動的に変更できます。EC2Launch v2 は、60 KB のユーザーデータの入力長をサポートします。ユーザーデータに含まれるのは UserData ステージのみであるため、ユーザーデータは `agent-config` ファイルの後に実行されます。インスタンスの起動ウィザードを使用してインスタンスを起動するときにユーザーデータを入力することも、EC2 コンソールからユーザーデータを変更することもできます。ユーザーデータの操作方法の詳細については、Amazon EC2 が Windows インスタンスのユーザーデータを処理する方法を参照してください。

EC2Launch v2 タスクの概要

EC2Launch v2 では、ブートが行われるたびに、次のタスクを実行できます。

- インスタンスに関する情報をレンダリングする新しい壁紙を設定し、必要に応じてカスタマイズします。
- ローカルマシンに作成される管理者アカウントの属性を設定します。
- 検索サフィックスのリストに DNS サフィックスを追加します。まだ存在しないサフィックスのみがリストに追加されます。
- 追加ボリュームのドライブ文字を設定し、これらを拡張して使用可能な領域を使用します。
- 設定からファイルをディスクに書き込みます。
- EC2Launch v2 設定ファイルでまたは `user-data` から指定されたスクリプトを実行します。`user-data` からのスクリプトは、プレーンテキストでも、圧縮して base64 形式での提供でも対応可能です。
- 指定された引数でプログラムを実行します。
- コンピュータ名を設定します。
- インスタンス情報を Amazon EC2 コンソールに送信します。
- RDP 証明書のサムプリントを Amazon EC2 コンソールに送信します。
- オペレーティングシステムパーティションを動的に拡張して、未使用の領域が含まれるようにします。
- ユーザーデータを実行します。ユーザーデータを指定する方法については、EC2Launch v2 タスクの設定を参照してください。
- メタデータサービスと AWS KMS サーバーに到達するように、非永続的な静的ルートを設定します。

- 非ブートパーティションを mbr または gpt に設定します。
- Sysprep 後に Systems Manager (SSM) サービスを開始します。
- ENA 設定を最適化します。
- 新しい Windows バージョンで OpenSSH を有効にします。
- ジャンボフレームを有効にします。
- EC2Launch v2 で実行するように Sysprep を設定します。
- Windows イベントログを発行します。

EC2Launch v2 のディレクトリ構造

EC2Launch v2 は、次のような構成のディレクトリにインストールする必要があります。

- サービスバイナリ: `%ProgramFiles%\Amazon\EC2Launch`
- サービスデータ (設定、ログファイル、状態ファイル): `%ProgramData%\Amazon\EC2Launch`

> **Note**
>
> Windows では、デフォルトで `C:\ProgramData` 以下のファイルとフォルダは非表示になります。EC2Launch v2 のディレクトリとファイルを表示するには、Windows エクスプローラーにパスを入力するか、フォルダのプロパティを変更して非表示のファイルとフォルダを表示する必要があります。

`%ProgramFiles%\Amazon\EC2Launch` ディレクトリには、バイナリとサポートライブラリが含まれます。次のサブディレクトリが含まれます。

- `settings`
  - `EC2LaunchSettingsUI.exe` - `agent-config.yml` ファイルを修正するためのユーザーインターフェース
  - `YamlDotNet.dll` - ユーザーインターフェイスでいくつかのオペレーションをサポートするための DLL
- `tools`
  - `ebsnvme-id.exe` - インスタンスで EBS ボリュームのメタデータを調べるためのツール
  - `AWSAcpiSpcrReader.exe` - 使用する正しい COM ポートを決定するためのツール

  - EC2LaunchEventMessage.dll - EC2Launch の Windows イベントログ記録をサポートするための DLL。
- service
  - EC2LaunchService.exe — 起動エージェントがサービスとして実行されたときに起動する Windows サービスの実行可能ファイル。
- EC2AgentTelemetry.dll – EC2 エージェントテレメトリをサポートするための DLL
- EC2Launch.exe - メインの EC2Launch 実行可能ファイル
- EC2LaunchAgentAttribution.txt - EC2 Launch 内で使用されるコードの属性

%ProgramData%\Amazon\EC2Launch ディレクトリには以下のサブディレクトリがあります。ログ、設定、状態など、サービスによって生成されたすべてのデータは、このディレクトリに保存されます。

- config – 設定

  サービス設定ファイルは、このディレクトリに agent-config.yml として保存されます。このファイルを更新し、サービスによってデフォルトで実行されるタスクを変更、追加、削除できます。このディレクトリにファイルを作成する権限は、権限の昇格を防ぐために、管理者アカウントに制限されています。
- log – インスタンスログ

  サービス (agent.log) およびテレメトリ (telemetry.log) のログはこのディレクトリに保存されます。agent.log は 1 MB のサイズに達すると、自動的にローテーションされ、タイムスタンプ形式 (agent-2026-03-02T18-56-39.188.log など) でバックアップファイルが作成されます。一度に保持できるバックアップログファイルは 1 つだけです。
- state – サービスの状態データ

  実行するタスクを決定するためにサービスで使用する状態がここに保存されます。Sysprep 後にサービスが実行済みであるかどうかを示す .run-once ファイルがあり、これにより、頻度が 1 回のタスクは次の実行でスキップされます。このサブディレクトリには、各タスクのステータスを追跡するための previous-state.json と state.json があります。
- sysprep – Sysprep

  このディレクトリ内のファイルを使用して、再利用可能なカスタマイズされた Windows AMI の作成時に Sysprep で実行するオペレーションを決定します。
- wallpaper – 壁紙

この壁紙イメージはこのディレクトリに保存されます。

テレメトリ

テレメトリは、AWSを使用して、要件の理解を深め、問題を診断し、AWS のサービスのサービスのユーザーエクスペリエンスを向上するのに役立つ追加情報です。

EC2Launch v2 バージョン2.1.592 およびそれ以降のバージョンは、使用状況指標やエラーなどのテレメトリを収集します。このデータは、EC2Launch v2 が実行される Amazon EC2 インスタンスから収集されます。これには、AWSによって所有されるすべての Windows AMI が含まれます。

EC2Launch v2 では、以下のテレメトリを収集しています。

- 使用状況の情報— エージェントのコマンド、インストール方法、スケジュールされた実行頻度。
- エラーと診断情報 — エージェントのインストールエラーコード、実行エラーコード、およびエラーコールスタック。

バージョン 2.0.592 から 2.1.1 まで収集されたデータの例：

```
2025/07/18 22:38:52Z: EC2LaunchTelemetry: IsTelemetryEnabled=true
2025/07/18 22:38:52Z: EC2LaunchTelemetry: AgentOsArch=windows_amd64
2025/07/18 22:38:52Z: EC2LaunchTelemetry: IsAgentScheduledPerBoot=true
2025/07/18 22:38:52Z: EC2LaunchTelemetry: AgentCommandErrorCode=0
2025/07/18 22:38:52Z: EC2LaunchTelemetry: AdminPasswordTypeCode=0
2025/07/18 22:38:52Z: EC2LaunchTelemetry: IpConflictDetectionCode=0
2025/07/18 22:38:52Z: EC2LaunchTelemetry: AgentErrorLocation=addroutes.go:49
```

バージョン 2.2.63 以降、EC2 エージェントテレメトリデータは JSON オブジェクトとしてフォーマットされます。

```
{"type":"EC2AgentTelemetry","agentId":"WindowsLaunchAgentV2" ... }
```

テレメトリーはデフォルトでは有効になっています。テレメトリ収集はいつでも無効にできます。

インスタンスでのテレメトリの無効化

1 つのインスタンスのテレメトリを無効にするには、システム環境変数を設定するか、MSI を使用してインストールを変更します。

システム環境変数を設定してテレメトリを無効にするには、管理者として次のコマンドを実行します。

```
setx /M EC2LAUNCH_TELEMETRY 0
```

MSI を使用してテレメトリを無効にするには、MSIをダウンロードした後で、以下のコマンドを実行します：

```
msiexec /i ".\AmazonEC2Launch.msi" Remove="Telemetry" /q
```

EC2Launch v2 のその他のトピック

- EC2Launch v2 の最新バージョンのインストール
- EC2Launch v2 設定を Windows インスタンスに構成する
- EC2Launch v2 起動タスクのタスク定義
- EC2Launch v2 エージェントの問題をトラブルシューティングする
- EC2Launch v2 のバージョン履歴

EC2Launch v2 の最新バージョンのインストール

EC2Launch v2 エージェントを EC2 インスタンスにインストールするには次のいずれかの方法を使用できます。

- Amazon S3 からエージェントをダウンロードし、Windows PowerShell を使用してインストールします。ダウンロード URL については「EC2Launch v2 を Amazon S3 でダウンロード」を参照してください。
- SSM ディストリビューターでインストールする
- カスタムイメージの作成時に EC2 Image Builder コンポーネントからインストールします。
- EC2Launch v2 がプリインストールされている AMI からインスタンスを起動する

> **Warning**
>
> AmazonEC2Launch.msi はEC2Launch (v1) や EC2Config などの以前のバージョンの EC2 起動サービスをアンインストールします。

インストール手順についてはお好みの方法に合ったタブを選択してください。

PowerShell

Windows PowerShell で EC2Launch v2 エージェントの最新バージョンをインストールするには以下の手順に従ってください。

1. ローカルディレクトリを作成する

```
New-Item -Path "$env:USERPROFILE\Desktop\EC2Launchv2" -ItemType Directory
```

2. ダウンロード場所の URL を設定します。使用する Amazon S3 URL を使用して次のコマンドを実行します。ダウンロード URL については「EC2Launch v2 を Amazon S3 でダウンロード」を参照してください。

```
$Url = "Amazon S3 URL/AmazonEC2Launch.msi"
```

3. 次の複合コマンドを使用して、エージェントをダウンロードしてインストールします。

```
$DownloadFile = "$env:USERPROFILE\Desktop\EC2Launchv2\" + $(Split-Path -Path
 $Url -Leaf)
Invoke-WebRequest -Uri $Url -OutFile $DownloadFile
msiexec /i "$DownloadFile"
```

> **Note**
>
> ファイルのダウンロード時にエラーが表示され、Windows Server 2016 以前のバージョンを使用している場合は、PowerShell ターミナルで TLS 1.2 を有効にする必要がある場合があります。次のコマンドで現在の PowerShell セッションの TLS 1.2 を有効にしてから、もう一度試してください。
>
> ```
> [Net.ServicePointManager]::SecurityProtocol =
>  [Net.SecurityProtocolType]::Tls12
> ```

4. msiexec コマンドはWindows Server インスタンスの `%ProgramFiles%\Amazon\EC2Launch` に EC2Launch v2 をインストールします。インストールが実行されたことを検証するにはインスタンスのローカルファイルシステムを確認できます。

AWS Systems Manager Distributor

AWS Systems Manager Quick Setup で EC2Launch v2 の自動更新を設定する方法については「ディストリビューター Quick Setup で自動的にインストールおよび更新する」を参照してください。

AWS Systems Manager ディストリビューターから `AWSEC2Launch-Agent` パッケージを 1 回だけインストールすることもできます。Systems Manager Distributor からパッケージをインストールする方法については「AWS Systems Manager ユーザーガイド」の「パッケージをインストールまたは更新する」をご参照ください。

EC2 Image Builder component

EC2 Image Builder でカスタムイメージをビルドするときに、`ec2launch-v2-windows` コンポーネントをインストールできます。EC2 Image Builder でカスタムイメージをビルドする方法についてはEC2 Image Builder ユーザーガイドのCreate an image pipeline using the EC2 Image Builder console wizardを参照してください。

AMI

EC2Launch v2 は次の Windows Server 2022 以降の OS の AMI にデフォルトでプリインストールされています。

- Windows_Server-#####-English-Full-Base
- Windows_Server-#####-English-Core-Base
- Windows_Server-#####-English-Core-EKS_Optimized
- 他のすべての言語による Windows Server ##### AMI
- SQL がインストールされた Windows Server ##### AMI

EC2Launch v2 は次の Windows Server AMI にもプリインストールされています。これらの AMI は Amazon EC2 コンソールから、または AWS CLI の `EC2LaunchV2-` の検索プレフィックスを使用して検索できます。

- EC2LaunchV2-Windows_Server-2019-English-Core-Base
- EC2LaunchV2-Windows_Server-2019-English-Full-Base
- EC2LaunchV2-Windows_Server-2016-English-Core-Base
- EC2LaunchV2-Windows_Server-2016-English-Full-Base

AWS Systems Manager ディストリビューター Quick Setup で EC2Launch v2 を自動的にインストールおよび更新する

AWS Systems Manager ディストリビューター Quick Setup で、EC2Launch v2 の自動アップデートを設定できます。以下のプロセスではSystems Manager 関連付けをインスタンスに設定して、指定した頻度で EC2Launch v2 エージェントを自動的に更新します。ディストリビューター Quick Setup が作成する関連付けにはAWS アカウント 内かつリージョン内のインスタンス、または AWS 組織内のインスタンスが含まれる場合があります。組織の設定の詳細については「AWS Organizations ユーザーガイド」の「チュートリアル: 組織の作成と設定」を参照してください。

開始する前に、インスタンスがすべての前提条件を満たしていることを確認してください。

前提条件

ディストリビューター Quick Setup で自動更新を設定するにはインスタンスが次の前提条件を満たす必要があります。

- EC2Launch v2 をサポートするインスタンスを 1 つ以上実行している。EC2Launch v2 がサポートされているオペレーティングシステムを参照してください。
- インスタンスで Systems Manager の設定タスクを実行した。詳細については「AWS Systems Manager ユーザーガイド」の「Systems Manager のセットアップ」を参照してください。
- インスタンスにインストールされている起動エージェントは EC2Launch v2 のみである必要があります。複数の起動エージェントがインストールされている場合、ディストリビューター Quick Setup 設定は失敗します。ディストリビューター Quick Setup で EC2Launch v2 を設定する前に、EC2Config または EC2Launch v1 起動エージェントが存在する場合はアンインストールします。

EC2Launch v2 のディストリビューター Quick Setup の設定

ディストリビューター Quick Setup で EC2Launch v2 の設定を作成するにはディストリビューターパッケージのデプロイの手順を完了するときに以下の設定を使用してください。

- ソフトウェアパッケージ:Amazon EC2Launch v2 エージェント。
- 更新頻度:リストから頻度を選択してください。
- ターゲット:利用可能なデプロイオプションから選択してください。

設定のステータスを確認するにはAWS マネジメントコンソール で、Systems Manager Quick Setup の [設定] タブに移動します。

1. AWS Systems Manager コンソール (https://console.aws.amazon.com/systems-manager/) を開きます。
2. ナビゲーションペインで、[Quick Setup] を選択してください。
3. [設定] タブで、作成した設定に関連付けられている行を選択してください。[設定] タブには設定が一覧表示され、リージョン、デプロイステータス、関連付けステータスなどの重要な詳細情報の概要が表示されます。

   > **Note**
   >
   > すべての EC2Launch v2 ディストリビューター設定の関連付け名は`AWS-QuickSetup-Distributor-EC2Launch-Agent-` のプレフィックスで始まります。

4. 詳細を表示するには設定を選択して [詳細を表示] を選択してください。

詳細とトラブルシューティング手順については「AWS Systems Managerユーザーガイド」の「Quick Setup の結果のトラブルシューティング」を参照してください。

EC2Launch v2 を Amazon S3 でダウンロード

最新バージョンの EC2Launch v2 をインストールするには以下の場所からインストーラーをダウンロードします。

- https://s3.amazonaws.com/amazon-ec2launch-v2/windows/amd64/latest/AmazonEC2Launch.msi

インストールオプションを設定する

EC2Launch v2 をインストールまたはアップグレードする場合、EC2Launch v2 インストールダイアログまたはコマンドラインシェルのコマンド msiexec を使用してインストールオプションを設定できます。

EC2Launch v2 インストーラーをインスタンスで初めて実行すると、インスタンスの起動エージェント設定が次のように初期化されます。

- ローカルパスを作成し、そこに起動エージェントファイルを書き込みます。これはクリーンインストールと呼ばれることもあります。
- `EC2LAUNCH_TELEMETRY` 環境変数が存在しない場合は作成され、設定に基づいて設定されます。

設定の詳細については使用する設定方法に合ったタブを選択してください。

Amazon EC2Launch Setup dialog

EC2Launch v2 をインストールまたはアップグレードする場合、EC2Launch v2 インストールダイアログから次のインストールオプションを設定できます。

ベーシックインストールオプション

テレメトリを送信する

この機能をセットアップダイアログに含めると、インストーラーはEC2LAUNCH_TELEMETRY環境変数を次の値に1。[テレメトリを送信] を無効にした場合、インストーラーは環境変数を 0 の値に設定します。

EC2Launch v2 エージェントが実行されると、EC2LAUNCH_TELEMETRY 環境変数を読み取り、テレメトリデータをアップロードするかどうかを決定します。値が 1 の場合、データをアップロードします。そうしないと、アップロードされません。

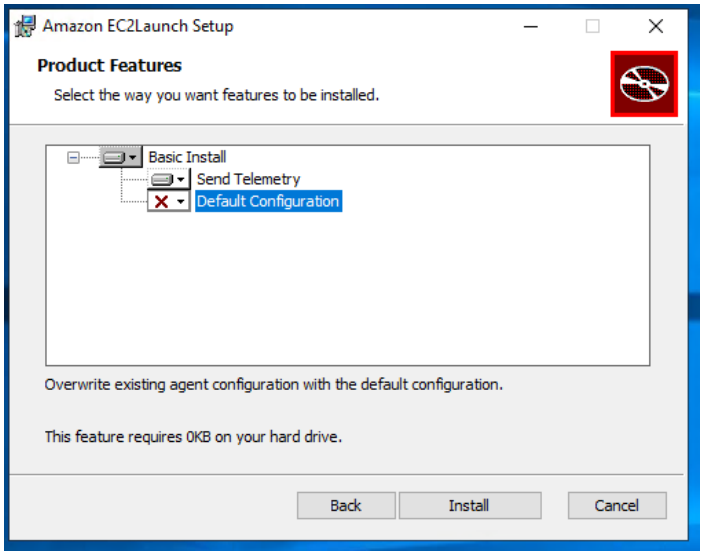
デフォルト設定

EC2Launch v2 のデフォルト設定ではローカル起動エージェントがすでに存在する場合はそれを上書きします。インスタンスで初めてインストールを実行すると、デフォルト設定でクリーンインストールが実行されます。初回インストール時にデフォルト設定を無効にすると、インストールは失敗します。

インスタンスでインストールを再度実行する場合はデフォルト設定を無効にして、%ProgramData%/Amazon/EC2Launch/config/agent-config.yml ファイルを置き換えないアップグレードを実行できます。

例: テレメトリによる EC2Launch v2 のアップグレード

次の例は現在のインストールをアップグレードしてテレメトリを有効にするように設定された EC2Launch v2 セットアップダイアログを示しています。この設定ではエージェント設定ファイルを置き換えずにインストールを実行し、EC2LAUNCH_TELEMETRY 環境変数を 1 の値に設定します。



Command line

EC2Launch v2 をインストールまたはアップグレードする場合、コマンドラインシェルの msiexec コマンドを使用して次のインストールオプションを設定できます。

ADDLOCAL パラメータ値

Basic(必須)

起動エージェントをインストールします。この値が ADDLOCAL パラメータに存在しない場合、インストールは終了します。

Clean

ADDLOCAL パラメータに Clean の値を含めると、インストーラはエージェント設定ファイルを %ProgramData%/Amazon/EC2Launch/config/agent-config.yml の場所に書き込みます。エージェント設定ファイルがすでに存在する場合、ファイルは上書きされます。

ADDLOCAL パラメータの Clean 値を省略した場合、インストーラはエージェント設定ファイルを置き換えないアップグレードを実行します。

Telemetry

ADDLOCAL パラメータに Telemetry の値を含めると、インストーラは EC2LAUNCH_TELEMETRY 環境変数を 1 の値に設定します。

ADDLOCAL パラメータから Telemetry の値を省略した場合、インストーラは環境変数を 0 の値に設定します。

EC2Launch v2 エージェントが実行されると、EC2LAUNCH_TELEMETRY 環境変数を読み取り、テレメトリデータをアップロードするかどうかを決定します。値が 1 の場合、データをアップロードします。そうしないと、アップロードされません。

例: テレメトリを使用した EC2Launch v2 のインストール

```
& msiexec /i "C:\Users\Administrator\Desktop\EC2Launchv2\AmazonEC2Launch.msi"
 ADDLOCAL="Basic,Clean,Telemetry" /q
```

EC2Launch v2 のバージョンを検証する

インスタンスにインストールされている EC2Launch v2 のバージョンを検証するには次のいずれかの手順を使用します。

PowerShell

次のように、Windows PowerShell を使用して、インストールされている EC2Launch v2 のバージョンを確認します。

1. AMI からインスタンスを起動して接続します。
2. PowerShell で以下のコマンドを実行して、インストールされている EC2Launch v2 のバージョンを検証します。

   ```
   & "C:\Program Files\Amazon\EC2Launch\EC2Launch.exe" version
   ```

Windows Control Panel

Windows のコントロール パネルで、インストールされている EC2Launch v2 のバージョンを検証するには以下に従います。

1. AMI からインスタンスを起動して接続します。
2. Windows の [コントロールパネル] を開き、[プログラムと機能] を選択してください。
3. インストールされたプログラムのリストで Amazon EC2Launch を探します。バージョン番号は [Version] 列に表示されています。

AWS Windows AMI の最新のアップデートを確認するにはAWS Windows AMI リファレンスの「Windows AMI のバージョン履歴」を参照してください。

EC2Launch v2 の最新バージョンはEC2Launch v2 のバージョン履歴でご確認ください。

EC2Launch v2 サービスの新しいバージョンがリリースされた際には通知を受け取ることができます。詳細については「EC2 Windows 起動エージェント通知をサブスクライブする」を参照してください。

EC2Launch v2 設定を Windows インスタンスに構成する

このセクションでは、EC2Launch v2 の設定を構成する方法について説明します。

トピックは以下のとおりです。

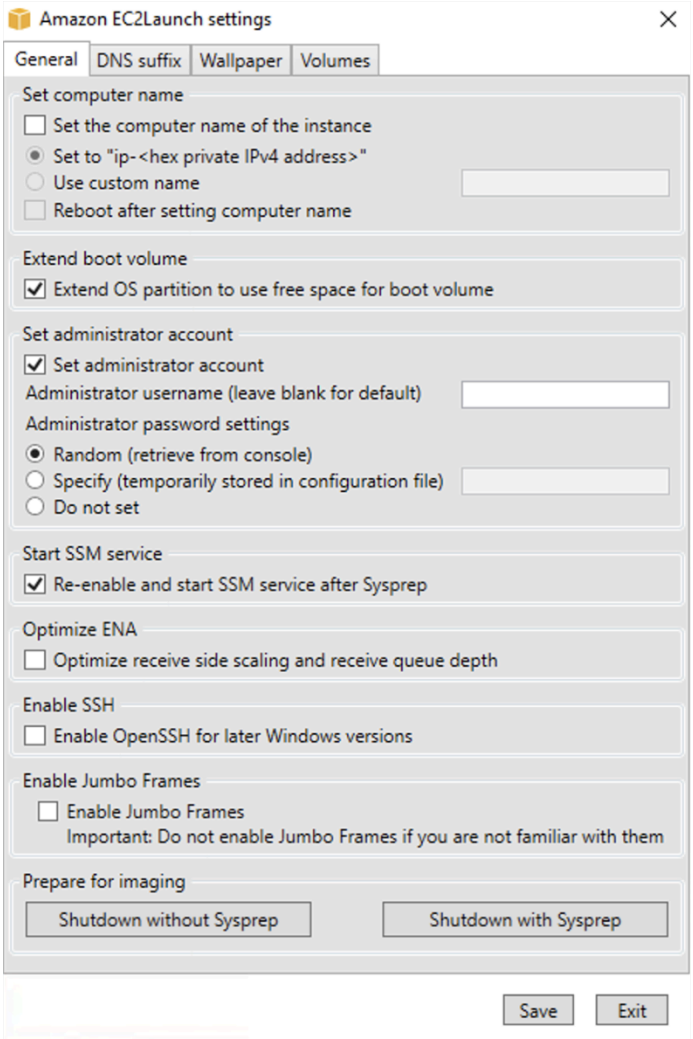
- EC2Launch v2 設定ダイアログボックスを使用して設定を変更する
- CLI を使用した EC2Launch v2 の設定
- EC2Launch v2 タスクの設定
- EC2Launch v2 の終了コードと再起動
- EC2Launch v2 とSysprep

EC2Launch v2 設定ダイアログボックスを使用して設定を変更する

次の手順では、EC2Launch v2 設定ダイアログボックスを使用して、設定を有効または無効にする方法を示します。

> **Note**
>
> agent-config.yml ファイルでカスタムタスクを不適切に設定した場合、Amazon EC2Launch 設定ダイアログボックスを開こうとすると、エラーが発生します。スキーマの例については、「例:agent-config.yml」を参照してください。

1. Windows インスタンスを起動して接続します。
2. [スタート] メニューから、[すべてのプログラム] を選択し、[EC2Launch 設定] に移動します。[Sysprep を使用してシャットダウン] または [Sysprep を使用せずにシャットダウン] を選択する前に、シャットダウンの実行時に適用する変更を必ず保存してください。



3. [EC2Launch 設定] ダイアログボックスの [全般] タブで、次の設定を有効または無効にすることができます。

   a. コンピュータ名の設定

      この設定を有効にすると (デフォルトでは無効になっています)、ブートごとに現在のホスト名が希望するホスト名と比較されます。ホスト名が一致しない場合、ホスト名はリセットされ、システムは必要に応じて再起動して、新しいホスト名を取得します。カスタムホスト名が指定されていない場合は、16 進形式のプライベート IPv4 アドレスを使用して生成されます。例えば、`ip-AC1F4E6` などです。既存のホスト名が変更されないようにするには、この設定を有効にしないでください。

   b. ブートボリュームの拡張

      この設定は、`Disk 0/Volume 0` を動的に拡張し、未使用の領域を含めます。独自のサイズを指定したルートボリュームからインスタンスを起動するときに便利です。

c. 管理者アカウントの設定

有効にすると、ローカルマシンに作成される管理者アカウントのユーザー名とパスワードの属性を設定できます。この機能を有効にしないと、Sysprep 後に管理者アカウントがシステムに作成されません。adminPasswordtype が Specify である場合のみ、adminPasswordにパスワードを入力してください。

パスワードの種類は次のとおりです。

i. Random

EC2Launch は、ユーザーのキーを使用してパスワードを生成し、暗号化します。この設定はインスタンス起動後に無効になるため、インスタンスを再起動したり、停止して起動した場合でもパスワードは保持されます。

ii. Specify

EC2Launch は、adminPassword で指定したパスワードを使用します。指定したパスワードがシステム要件を満たさない場合は、代わりに EC2Launch によってランダムなパスワードが生成されます。このパスワードはクリアテキストとして agent-config.yml に保存され、Sysprep で管理者パスワードが設定されると削除されます。EC2Launch は、ユーザーのキーを使用してパスワードを暗号化します。

iii. Do not set

EC2Launch は、unattend.xml ファイルで指定したパスワードを使用します。unattend.xml でパスワードを指定しないと、管理者アカウントは無効になります。

d. SSM サービスの開始

選択された Systems Manager サービスの起動が Sysprep の後に有効化されます。EC2Launch v2 は前述のすべてのタスクを実行し、SSM Agent は Run Command やステートマネージャーなどの Systems Manager 機能に対するリクエストを処理します。

Run Command を使用して既存のインスタンスをアップグレードすることで、最新バージョンの EC2Launch v2 サービスや SSM Agent を使用できるようになります。詳細については、AWS Systems Manager ユーザーガイドのRun Command を使用した SSM エージェントの更新を参照してください。

e. ENA の最適化

選択すると、ENA の受信側のスケーリングおよび受信キューの深さの設定を AWS 用に最適化するように ENA 設定が設定されます。詳細については、受信側スケーリング CPU アフィニティを設定するを参照してください。

f. SSH の有効化

この設定では、より新しいバージョンの Windows で OpenSSH を有効にし、リモートシステム管理を許可できます。

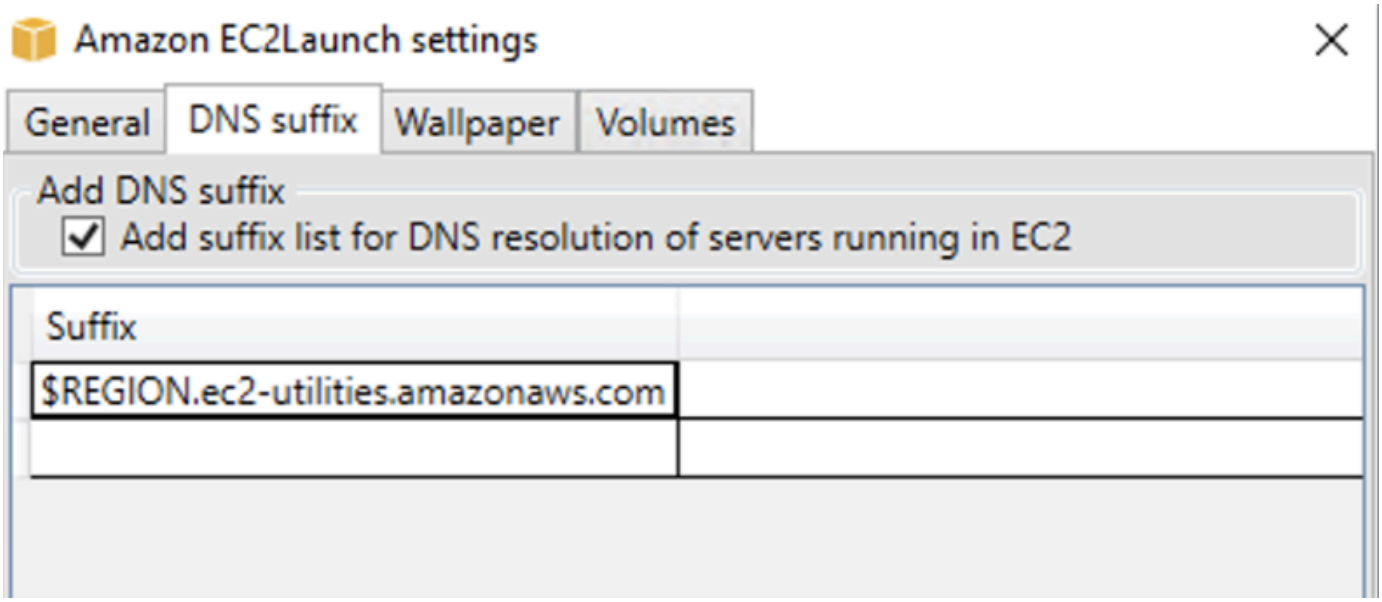
g. ジャンボフレームの有効化

ジャンボフレームを有効にする場合は、これを選択してください。ジャンボフレームは、ネットワーク通信に意図しない影響を及ぼす可能性があるため、ジャンボフレームがシステムに与える影響をよく理解した上で有効にしてください。ジャンボフレームの詳細については、ジャンボフレーム (9001 MTU)を参照してください。

h. イメージングの準備

EC2 インスタンスのシャットダウンに Sysprep を使用するかしないかを選択してください。EC2Launch v2 で Sysprep を実行する場合は、[Sysprep でシャットダウン] を選択してください。

4. [DNS サフィックス] タブで、完全修飾ドメイン名を指定せずに、EC2 で実行されているサーバーの DNS 解決用に DNS サフィックスのリストを追加するかどうかを選択できます。DNS サフィックスには、`$REGION` 変数と `$AZ` 変数を含めることができます。まだ存在しないサフィックスのみがリストに追加されます。

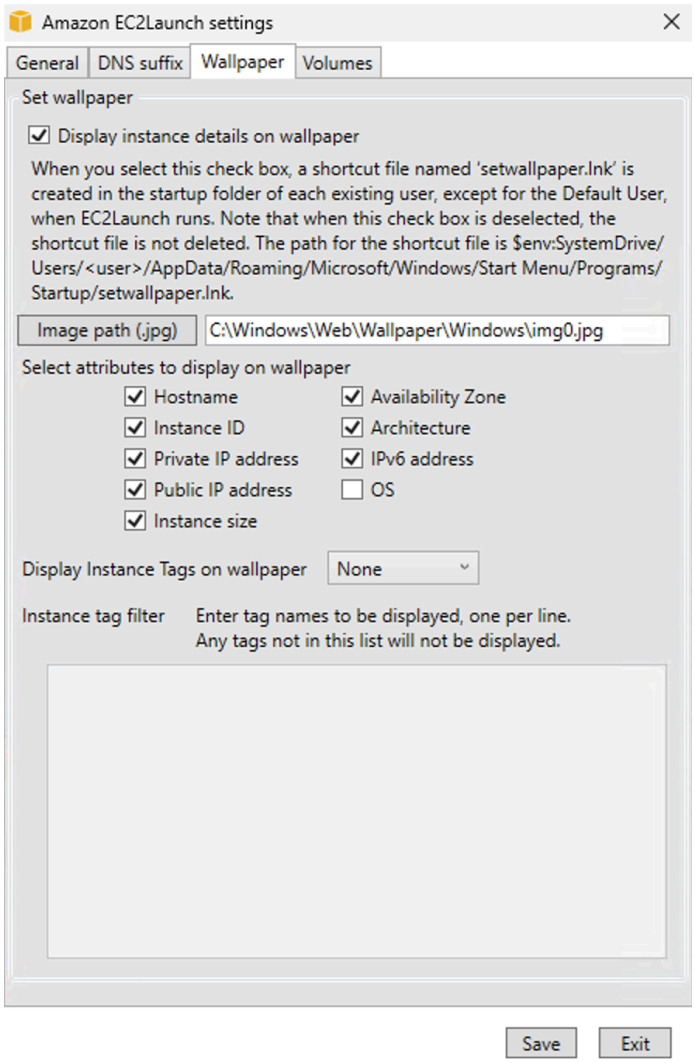


5. [壁紙] タブでは、背景画像を使用してインスタンスの壁紙を設定し、表示する壁紙のインスタンスの詳細を指定できます。Amazon EC2 は、ログインするたびに詳細を生成します。

   次のコントロールで壁紙を設定できます。

   - [壁紙にインスタンスの詳細を表示] - このチェックボックスは、壁紙のインスタンス詳細表示を有効または無効にします。
   - [画像パス (.jpg)] - 壁紙の背景として使用する画像へのパスを指定します。
   - [壁紙に表示する属性を選択] - 壁紙に表示するインスタンスの詳細のチェックボックスを選択してください。壁紙からインスタンスの詳細を削除するには、以前に選択したチェックボックスをオフにします。
   - [壁紙にインスタンスタグを表示] - 壁紙にインスタンスタグを表示するには、次のいずれかの設定を選択してください。
     - [なし] - 壁紙にインスタンスタグを表示しないでください。
     - [すべて表示] - 壁紙にすべてのインスタンスタグを表示します。
     - [フィルター済みを表示] - 指定したインスタンスタグを壁紙に表示します。この設定を選択すると、[インスタンスタグフィルター] ボックスが表示され、壁紙に表示するインスタンスタグを追加できます。

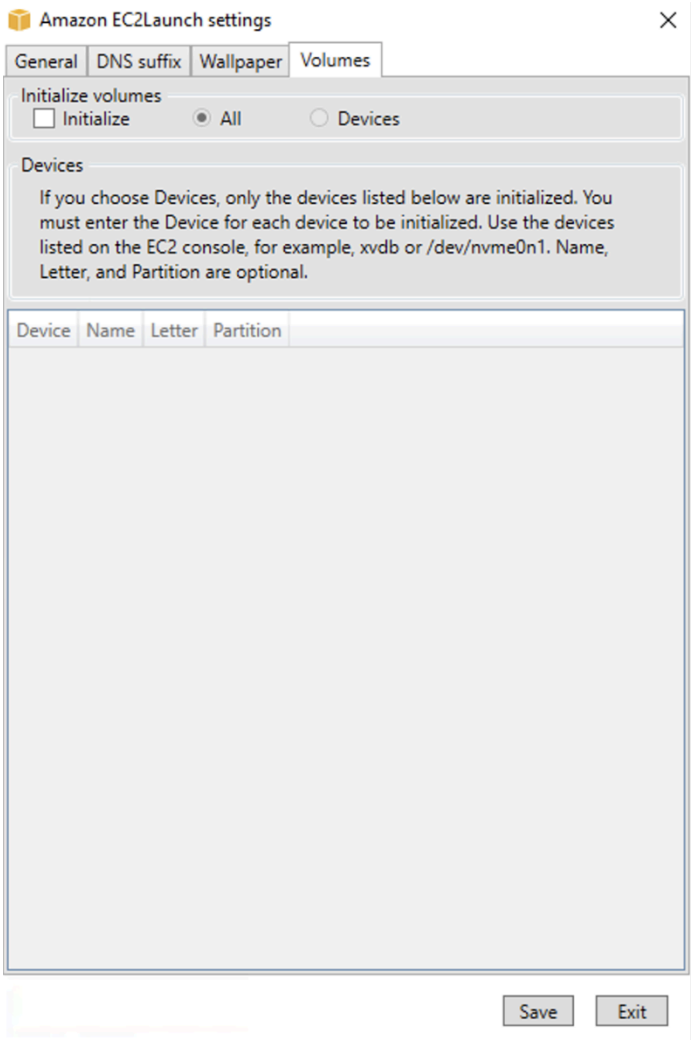
     > **Note**
     >
     > 壁紙にタグを表示するには、メタデータのタグを有効にする必要があります。インスタンスのタグおよびメタデータの詳細については、「インスタンスメタデータを使用して EC2 インスタンスのタグを表示する」を参照してください。



6. [ボリューム] タブで、インスタンスにアタッチされているボリュームを初期化するかどうかを選択してください。有効にすると、追加ボリュームのドライブ文字が設定され、使用可能な領域を使用するようにドライブ文字が拡張されます。[すべて] を選択すると、すべてのストレージボリュームが初期化されます。[デバイス] を選択すると、リストで指定されているデバイスのみが初期化されます。初期化するデバイスごとに、デバイスを入力する必要があります。EC2 コンソールにリストされているデバイス (xvdb や /dev/nvme0n1 など) を使用します。ドロップダウンリストには、インスタンスにアタッチされているストレージボリュームが表示されます。インスタンスにアタッチされていないデバイスを入力するには、テキストフィールドに入力してください。

   [名前]、[文字]、[パーティション] は、オプションのフィールドです。[パーティション] に値を指定しなかった場合、2 TB より大きいストレージボリュームは gpt パーティションタイプで初期化され、2 TB より小さいストレージボリュームは mbr パーティションタイプで初期化されま

す。デバイスが設定済みで、NTFS 以外のデバイスにパーティションテーブルが含まれているか、ディスクの最初の 4 KB にデータが含まれている場合、ディスクはスキップされ、アクションがログに記録されます。



CLI を使用した EC2Launch v2 の設定

コマンドラインインターフェイス (CLI) を使用して、EC2Launch の設定を構成し、サービスを管理できます。以下のセクションでは、EC2Launch v2 の管理に使用できる CLI コマンドを説明し、その使用方法を示します。

コマンド

- collect-logs
- get-agent-config
- list-volumes

- reset
- 実行
- ステータス
- sysprep
- validate
- バージョン
- 壁紙

collect-logs

EC2Launch のログファイルを収集し、これらのファイルを圧縮して、指定先のディレクトリに配置します。

例

```
ec2launch collect-logs -o C:\Mylogs.zip
```

使用方法

`ec2launch collect-logs [flags]`

Flags

`-h, --help`

`collect-logs` に関するヘルプ

`-o, --output string`

圧縮された出力ログファイルへのパス

get-agent-config

`agent-config.yml` を指定された形式 (JSON または YAML) で印刷します。書式が指定されていない場合、`agent-config.yml` は以前に指定された書式で印刷されます。

例

```
ec2launch get-agent-config -f json
```

使用方法

`ec2launch get-agent-config [flags]`

Flags

`-h, --help`

`get-agent-config` に関するヘルプ

`-f, --format string`

`agent-config` ファイルの出力形式: `json`、`yaml`

list-volumes

エフェメラルボリュームや EBS ボリュームなど、インスタンスにアタッチされているすべてのストレージボリュームを一覧表示します。

例

```
ec2launch list-volumes
```

使用方法

`ec2launch list-volumes`

Flags

`-h, --help`

`list-volumes` に関するヘルプ

reset

このタスクの主な目的は、次回実行時にエージェントをリセットすることです。そのために、resetコマンドは EC2Launch v2 EC2Launch のすべてのエージェント状態データをローカルディレクトリから削除します (を参照) 。EC2Launch v2 のディレクトリ構造リセットすると、オプションでサービスログと Sysprep ログが削除されます。

スクリプトの動作は、エージェントがスクリプトを実行するモード (インラインかデタッチモード) によって異なります。

インライン (デフォルト)

EC2Launch v2 エージェントはスクリプトを 1 つずつ実行します (`detach: false`)。これはデフォルトの設定です。

> **Note**
>
> インラインスクリプトが reset または sysprep コマンドを発行すると、すぐに実行され、エージェントがリセットされます。現在のタスクが終了すると、エージェントはそれ以上のタスクを実行せずにシャットダウンします。
> 例えば、コマンドを発行するタスクの後にタスク (ユーザーデータの実行後にデフォルトで含まれる) が続く場合、その `startSsm` タスクは実行されず、Systems Manager サービスは開始されません。

デタッチ済み

EC2Launch v2 エージェントは、他のタスクと同時にスクリプトを実行します (`detach: true`)。

> **Note**
>
> デタッチされたスクリプトが reset または sysprep コマンドを発行すると、それらのコマンドはエージェントが終了するのを待ってから実行します。ExecuteScript の後のタスクは引き続き実行されます。

例

```
ec2launch reset -c
```

使用方法

`ec2launch reset [flags]`

Flags

`-c, --clean`

`reset` 前にインスタンスログを消去する

`-h, --help`

`reset` に関するヘルプ

実行

EC2Launch v2 を実行します。

例

```
ec2launch run
```

使用方法

`ec2launch run [flags]`

Flags

`-h, --help`

`run` に関するヘルプ

ステータス

EC2Launch v2 エージェントのステータスを取得します。オプションで、エージェントが終了するまでプロセスをブロックします。プロセスの終了コードは、エージェントの状態を決定します:

- `0` — エージェントは実行され、成功しました。
- `1` — エージェントは実行されましたが、失敗しました。
- `2` — エージェントはまだ実行中です。
- `3` — エージェントの状態が不明です。エージェントの状態が実行されていないか、停止していません。
- `4` — エージェントの状態を取得しようとしたときにエラーが発生しました。
- `5` — エージェントが実行されておらず、最後に既知の実行の状態が不明です。これは、次のいずれかの 1 つを意味します。
  - `state.json`および`previous-state.json`の両方がが削除されます。
  - `previous-state.json`は破損している。

  これは、`reset` コマンドを実行した後のエージェントの状態です。

例:

```
ec2launch status -b
```

使用方法

`ec2launch status [flags]`

Flags

`-b,--block`

エージェントの実行が終了するまでプロセスをブロックします

`-h,--help`

`status` に関するヘルプ

sysprep

このタスクの主な目的は、次回実行時にエージェントをリセットすることです。そのために、sysprepコマンドはエージェントの状態をリセットし、`unattend.xml`ファイルを更新し、RDPを無効にして、Sysprepを実行します。

スクリプトの動作は、エージェントがスクリプトを実行するモード (インラインかデタッチモード) によって異なります。

インライン (デフォルト)

EC2Launch v2 エージェントはスクリプトを 1 つずつ実行します (`detach: false`)。これはデフォルトの設定です。

> **Note**
>
> インラインスクリプトが reset または sysprep コマンドを発行すると、すぐに実行され、エージェントがリセットされます。現在のタスクが終了すると、エージェントはそれ以上のタスクを実行せずにシャットダウンします。
> 例えば、コマンドを発行するタスクの後にタスク (ユーザーデータの実行後にデフォルトで含まれる) が続く場合、その `startSsm` タスクは実行されず、Systems Manager サービスは開始されません。

デタッチ済み

EC2Launch v2 エージェントは、他のタスクと同時にスクリプトを実行します (`detach: true`)。

> **Note**
>
> デタッチされたスクリプトが reset または sysprep コマンドを発行すると、それらのコマンドはエージェントが終了するのを待ってから実行します。ExecuteScript の後のタスクは引き続き実行されます。

例:

```
ec2launch sysprep
```

使用方法

`ec2launch sysprep [flags]`

Flags

`-c,--clean`

`sysprep` 前にインスタンスログを消去する

`-h,--help`

Sysprep に関するヘルプ

`-s,--shutdown`

`sysprep` の後にインスタンスをシャットダウンする

validate

`agent-config` ファイル `C:\ProgramData\Amazon\EC2Launch\config\agent-config.yml` を検証します。

例

```
ec2launch validate
```

使用方法

`ec2launch validate [flags]`

Flags

`-h 、--help`

`validate` に関するヘルプ

バージョン

実行可能なバージョンを取得します。

例

```
ec2launch version
```

使用方法

`ec2launch version [flags]`

Flags

`-h, --help`

`version` に関するヘルプ

壁紙

指定した壁紙パス (.jpg ファイル) に新しい壁紙を設定し、選択したインスタンスの詳細を表示します。

構文

```
ec2launch wallpaper ^
--path="C:\ProgramData\Amazon\EC2Launch\wallpaper\Ec2Wallpaper.jpg" ^
--all-tags ^
--
attributes=hostName,instanceId,privateIpAddress,publicIpAddress,ipv6Address,instanceSize,availa
```

入力

パラメータ

--allowed-tags [*tag-name-1*, *tag-name-n*]

(オプション) 壁紙に表示するインスタンスタグ名の Base64 エンコードされた JSON 配列。このタグまたは `--all-tags` を使用できますが、両方は使用できません。

--attributes *attribute-string-1*, *attribute-string-n*

(オプション) 壁紙に設定を適用する `wallpaper` 属性文字列のコンマ区切りのリスト。

[--path | -p] *path-string*

(必須) `wallpaper` 背景画像ファイルのパスを指定します。

フラグ

--all-tags

(オプション) 壁紙にすべてのインスタンスタグを表示します。このタグまたは `--allowed-tags` を使用できますが、両方は使用できません。

[--help | -h]

wallpaper コマンドに関するヘルプを表示します。

EC2Launch v2 タスクの設定

このセクションでは、`agent-config.yml`とユーザーデータの設定スキーマ、タスク、詳細、および例が含まれます。

タスクと例

- スキーマ：agent-config.yml
- 起動または再起動時に実行される EC2Launch v2 ユーザーデータスクリプトを設定する

スキーマ：**agent-config.yml**

`agent-config.yml` ファイルの構造を以下に示します。同じステージでタスクを繰り返すことはできないことに注意してください。タスクのプロパティについては、次のタスクの説明を参照してください。

ドキュメント構造: agent-config.yml

JSON

```
{
 "version": "1.1",
 "config": [
  {
   "stage": "string",
   "tasks": [
    {
     "task": "string",
     "inputs": {
      ...
     }
    },
    ...
   ]
  },
  ...
 ]
}
```

YAML

```
version: 1.1
config:
- stage: string
  tasks:
  - task: string
 inputs:
   ...
  ...
...
```

例:**agent-config.yml**

次の例は、agent-config.yml 設定ファイルの設定を示しています。

```
version: 1.1
config:
- stage: boot
```

```
  tasks:
  - task: extendRootPartition
- stage: preReady
  tasks:
  - task: activateWindows
    inputs:
      activation:
        type: amazon
  - task: setDnsSuffix
    inputs:
      suffixes:
      - $REGION.ec2-utilities.amazonaws.com
  - task: setAdminAccount
    inputs:
      password:
        type: random
  - task: setWallpaper
    inputs:
      path: C:\ProgramData\Amazon\EC2Launch\wallpaper\Ec2Wallpaper.jpg
      attributes:
      - hostName
      - instanceId
      - privateIpAddress
      - publicIpAddress
      - instanceSize
      - availabilityZone
      - architecture
- stage: postReady
  tasks:
  - task: startSsm
```

起動または再起動時に実行される EC2Launch v2 ユーザーデータスクリプトを設定する

次の JSON と YAML の例は、ユーザーデータのドキュメント構造を示しています。Amazon EC2 は、ドキュメントで指定した tasks 配列で指定された各タスクを解析します。各タスクには、独自のプロパティと要件があります。詳細については、「EC2Launch v2 起動タスクのタスク定義」を参照してください。

> **Note**
>
> タスクは、ユーザーデータタスク配列で重複しないようにします。

文書構造: ユーザーデータ

JSON

```
{
 "version": "1.1",
 "tasks": [
  {
   "task": "string",
   "inputs": {
    ...
   },
  },
  ...
 ]
}
```

YAML

```
version: 1.1
tasks:
- task: string
  inputs:
    ...
...
```

例: ユーザーデータ

ユーザーデータの詳細については、Amazon EC2 が Windows インスタンスのユーザーデータを処理する方法を参照してください。

次の YAML ドキュメントの例は、EC2Launch v2 がファイルを作成するためのユーザーデータとして実行する PowerShell スクリプトを示しています。

```
version: 1.1
tasks:
- task: executeScript
  inputs:
  - frequency: always
    type: powershell
    runAs: localSystem
    content: |-
```

```
      New-Item -Path 'C:\PowerShellTest.txt' -ItemType File
```

ユーザーデータには、以前のバージョンの起動エージェントと互換性のある XML 形式を使用できます。EC2Launch v2 はスクリプトを UserData ステージ内の executeScript タスクとして実行します。EC2Launch v1 と EC2Config の動作に合わせて、ユーザーデータスクリプトはデフォルトでアタッチ/インラインプロセスとして実行されます。

オプションのタグを追加して、スクリプトの実行方法をカスタマイズできます。例えば、インスタンスの起動時だけでなく、インスタンスの再起動時にユーザーデータスクリプトを実行するには、次のタグを使用できます。

<persist>true</persist>

例:

```
<powershell>
  $file = $env:SystemRoot + "\Temp" + (Get-Date).ToString("MM-dd-yy-hh-mm")
  New-Item $file -ItemType file
</powershell>
<persist>true</persist>
```

<powershellArguments> タグを使用して、1 つ以上の PowerShell 引数を指定できます。引数が渡されない場合、EC2Launch v2 はデフォルトで次の引数を追加します: -ExecutionPolicy Unrestricted

例:

```
<powershell>
  $file = $env:SystemRoot + "\Temp" + (Get-Date).ToString("MM-dd-yy-hh-mm")
  New-Item $file -ItemType file
</powershell>
<powershellArguments>-ExecutionPolicy Unrestricted -NoProfile -NonInteractive</
powershellArguments>
```

XML ユーザーデータスクリプトをデタッチプロセスとして実行するには、ユーザーデータに次のタグを追加します。

<detach>true</detach>

例:

```
<powershell>
  $file = $env:SystemRoot + "\Temp" + (Get-Date).ToString("MM-dd-yy-hh-mm")
  New-Item $file -ItemType file
</powershell>
<detach>true</detach>
```

> **Note**
>
> デタッチタグは以前の起動エージェントではサポートされていません。

変更ログ: ユーザーデータ

次の表は、ユーザーデータの変更点と、該当する EC2Launch v2 エージェントバージョンとの相互参照を示しています。

ユーザーデータバージョン	詳細	ご紹介
1.1	• ユーザーデータタスクは、エージェント設定ファイルの `PostReady` ステージの前に実行されます。 • システムマネージャーエージェントを開始する前にユーザーデータを実行します (EC2Launch v1 および EC2Config と同じ動作)。*	EC2Launch v2 バージョン 2.0.1245
1.0	• 廃止されます。 • ユーザーデータタスクは、エージェント設定ファイルの `PostReady` ステージ後に実行されます。これには EC2Launch v1 との下位互換性はありません。 • システムマネージャーエージェントの起動とユーザーデータタスクの間の競合状態による影響を受けます。	EC2Launch v2 バージョン 2.0.0

* デフォルトの `agent-config.yml` ファイルで使用した場合。

EC2Launch v2 の終了コードと再起動

EC2Launch v2 を使用して、スクリプトによる終了コードの処理方法を定義できます。デフォルトでは、スクリプトで最後に実行されたコマンドの終了コードは、スクリプト全体の終了コードとしてレポートされます。例えば、スクリプトに 3 つのコマンドが含まれており、最初のコマンドが失敗したが、次のコマンドが成功した場合、実行ステータスは、最後のコマンドが成功したために `success` として報告されます。

スクリプトでインスタンスを再起動する場合、再起動がスクリプトの最後のステップで実行されるようになっていても、そのスクリプト内で `exit 3010` を指定する必要があります。`exit 3010` は、インスタンスを再起動し、さらに `3010` 以外の終了コードが返されるか再起動が最大回数に達するまで、スクリプトを再度呼び出すように EC2Launch v2 に指示します。EC2Launch v2 では、タスクごとに最大 5 回の再起動が許可されます。`Restart-Computer` などの別のメカニズムを使用してスクリプトからインスタンスを再起動しようとすると、スクリプトの実行ステータスは矛盾します。例えば、再起動ループで停止したり、再起動を実行しなかったりすることがあります。

古いエージェントと互換性のある XML ユーザーデータ形式を使用している場合、ユーザーデータは意図した回数よりも多く実行されることがあります。詳細については、トラブルシューティングセクションのサービスはユーザーデータを複数回実行するを参照してください。

EC2Launch v2 とSysprep

EC2Launch v2 サービスは Sysprep という Microsoft ツールを実行します。このツールを利用すると、再利用可能でカスタマイズされた Windows AMI を作成できます。EC2Launch v2 は、Sysprep を呼び出す際、`%ProgramData%\Amazon\EC2Launch` にあるファイルを使用して実行する操作を決定します。(オプション) フィルターセットをデフォルトフィルターとして設定するには、[EC2 サービスプロパティ] チェックボックスをオンにします。ただし、一部の高度な設定は [EC2Launch 設定] ダイアログボックスで利用できないため、これらのエントリは直接編集する必要があります。

インスタンスの設定を更新した後で、そのインスタンスから AMI を作成した場合、その AMI から起動されるすべてのインスタンスには、更新後の新しい設定が適用されます。AMI の作成の詳細については、Amazon EBS-backed AMI を作成するを参照してください。

EC2Launch v2 起動タスクのタスク定義

EC2Launch v2 が起動時またはスタートアップ時に実行する各タスクには、独自のプロパティと要件のセットがあります。タスクの詳細には、タスクの実行頻度 (1 回のみ、または毎回) に関する設定、タスクが実行されるエージェントのブートプロセスのステージ、構文、および YAML ドキュメントの例が含まれます。詳細については、このリファレンスに示されているタスクの詳細を確認してください。

EC2Launch v2 のタスク

- activateWindows
- enableJumboFrames
- enableOpenSsh
- executeProgram
- executeScript
- extendRootPartition
- initializeVolume
- optimizeEna
- setAdminAccount
- setDnsSuffix
- setHostName
- setWallpaper
- startSsm
- sysprep
- writeFile

activateWindows

AWS KMS サーバーのセットに対して Windows をアクティブ化します。インスタンスが Bring-Your-Own-License (BYOL) であることが検出された場合、アクティベーションはスキップされます。

Frequency - 1 回

AllowedStages - `[PreReady]`

Inputs -

`activation`: (マップ)

`type`: (文字列) 使用するアクティベーションタイプ、`amazon` に設定

例

```
task: activateWindows
```

```
  inputs:
    activation:
    type: amazon
```

enableJumboFrames

ジャンボフレームを有効にします。これにより、ネットワークアダプターの最大送信単位 (MTU) が増加します。詳細については、ジャンボフレーム (9001 MTU)を参照してください。

Frequency - 常に

AllowedStages - `[PostReady, UserData]`

Inputs - なし

例

```
task: enableJumboFrames
```

enableOpenSsh

Windows OpenSSH を有効にし、インスタンスのパブリックキーを認証済みキーフォルダに追加します。

Frequency - 1 回

AllowedStages - `[PreReady, UserData]`

Inputs - なし

例

次の例は、インスタンスで OpenSSH を有効にし、インスタンスのパブリックキーを認証済みキーフォルダーに追加する方法を示しています。この設定は、Windows Server 2019 以降のバージョンを実行しているインスタンスでのみ機能します。

```
task: enableOpenSsh
```

executeProgram

オプションの引数と指定された頻度でプログラムを実行します。

ステージ: PreReady、PostReady、および UserData のステージ中に executeProgram タスクを実行できます。

頻度: 設定可能です。「入力」を参照してください。

入力

このセクションには、executeProgram タスクを実行するための 1 つ以上のプログラム (入力) が含まれます。各入力には、次の設定を含めることができます:

frequency (文字列)

(必須) 次の値のいずれかを正確に指定します。

- once
- always

path (文字列)

(必須) 実行する実行可能ファイルのファイルパス。

arguments (文字列のリスト)

(オプション) 入力としてプログラムに提供する引数のカンマ区切りのリスト。

runAs (文字列)

(必須) localSystem に設定する必要があります

Output

すべてのタスクはログファイルエントリを agent.log ファイルに書き込みます。executeProgram タスクからの追加出力は、次のように動的に名前が付けられたフォルダに個別に保存されます。

%LocalAppData%\Temp*EC2Launch#########**outputfilename.tmp*

出力ファイルへの正確なパスが agent.log ファイルに含まれています。例:

```
Program file is created at: C:\Windows\system32\config\systemprofile\AppData\Local
\Temp\EC2Launch123456789\ExecuteProgramInputs.tmp
Output file is created at: C:\Windows\system32\config\systemprofile\AppData\Local
\Temp\EC2Launch123456789\Output.tmp
Error file is created at: C:\Windows\system32\config\systemprofile\AppData\Local
\Temp\EC2Launch123456789\Err.tmp
```

executeProgram タスクの出力ファイル

ExecuteProgramInputs.tmp

実行可能ファイルのパスと、executeProgram タスクが実行時に渡すすべての入力パラメータが含まれます。

Output.tmp

executeProgram タスクが実行するプログラムからのランタイム出力が含まれます。

Err.tmp

executeProgram タスクが実行するプログラムからのランタイムエラーメッセージが含まれます。

例

次の例は、executeProgram タスクを使用してインスタンス上のローカルディレクトリから実行可能ファイルを実行する方法を示しています。

例 1: 1 つの引数を使用する設定実行可能ファイル

この例は、設定実行可能ファイルを Quiet モードで実行する executeProgram タスクを示しています。

```
task: executeProgram
  inputs:
    - frequency: always
      path: C:\Users\Administrator\Desktop\setup.exe
      arguments: ['-quiet']
```

例 2: 2 つの引数を使用する VLC 実行可能ファイル

この例は、入力パラメータとして渡された 2 つの引数を使用して VLC 実行可能ファイルを実行する executeProgram タスクを示しています。

```
task: executeProgram
  inputs:
    - frequency: always
      path: C:\vlc-3.0.11-win64.exe
      arguments: ['/L=1033','/S']
```

```
      runAs: localSystem
```

executeScript

オプションの引数と指定された頻度でスクリプトを実行します。スクリプトの動作は、エージェントがスクリプトを実行するモード (インラインかデタッチモード) によって異なります。

インライン (デフォルト)

EC2Launch v2 エージェントはスクリプトを 1 つずつ実行します (`detach: false`)。これはデフォルトの設定です。

> **Note**
>
> インラインスクリプトが reset または sysprep コマンドを発行すると、すぐに実行され、エージェントがリセットされます。現在のタスクが終了すると、エージェントはそれ以上のタスクを実行せずにシャットダウンします。
> 例えば、コマンドを発行するタスクの後にタスク (ユーザーデータの実行後にデフォルトで含まれる) が続く場合、その `startSsm` タスクは実行されず、Systems Manager サービスは開始されません。

デタッチ済み

EC2Launch v2 エージェントは、他のタスクと同時にスクリプトを実行します (`detach: true`)。

> **Note**
>
> デタッチされたスクリプトが reset または sysprep コマンドを発行すると、それらのコマンドはエージェントが終了するのを待ってから実行します。ExecuteScript の後のタスクは引き続き実行されます。

ステージ: `PreReady`、`PostReady`、および `UserData` のステージ中に `executeScript` タスクを実行できます。

頻度: 設定可能です。「入力」を参照してください。

入力

このセクションには、executeScript タスクを実行するための 1 つ以上のスクリプト (入力) が含まれます。各入力には、次の設定を含めることができます:

frequency (文字列)

(必須) 次の値のいずれかを正確に指定します。

- `once`
- `always`

type (文字列)

(必須) 次の値のいずれかを正確に指定します。

- `batch`
- `powershell`

arguments (文字列のリスト)

(オプション) (PowerShell スクリプトではなく) シェルに渡す文字列引数のリスト。このパラメータは、`type: batch` ではサポートされません。引数が渡されない場合、EC2Launch v2 はデフォルトで次の引数を追加します: `-ExecutionPolicy Unrestricted`

content (文字列)

(必須) インラインスクリプトのコンテンツ。

runAs (文字列)

(必須) 次の値のいずれかを正確に指定します。

- `admin`
- `localSystem`

デタッチ (ブール値)

(オプション) EC2Launch v2 エージェントは、スクリプトを一度に 1 つずつ実行するようデフォルトで設定されています (`detach: false`)。スクリプトを他のタスクと同時に実行するには、値を `true` (`detach: true`) に設定します。

> **Note**
>
> `detach` に `true` が設定されている場合、スクリプトの終了コード (`3010` など) は効果がなくなります。

Output

すべてのタスクはログファイルエントリを agent.log ファイルに書き込みます。executeScript タスクが実行するスクリプトからの追加出力は、次のように動的に名前が付けられたフォルダに個別に保存されます。

%LocalAppData%\Temp*EC2Launch#########**outputfilename.ext*

出力ファイルへの正確なパスが agent.log ファイルに含まれています。例:

```
Program file is created at: C:\Windows\system32\config\systemprofile\AppData\Local
\Temp\EC2Launch123456789\UserScript.ps1
Output file is created at: C:\Windows\system32\config\systemprofile\AppData\Local
\Temp\EC2Launch123456789\Output.tmp
Error file is created at: C:\Windows\system32\config\systemprofile\AppData\Local
\Temp\EC2Launch123456789\Err.tmp
```

executeScript タスクの出力ファイル

UserScript.*ext*

executeScript タスクが実行したスクリプトが含まれます。ファイルの拡張子は、次のように、executeScript タスクの type パラメータで指定したスクリプトのタイプによって異なります。

- タイプが batch の場合、ファイルの拡張子は .bat です。
- タイプが powershell の場合、ファイルの拡張子は .ps1 です。

Output.tmp

executeScript タスクが実行するスクリプトからのランタイム出力が含まれます。

Err.tmp

executeScript タスクが実行するスクリプトからのランタイムエラーメッセージが含まれます。

例

次の例は、executeScript タスクでインラインスクリプトを実行する方法を示しています。

例 1: Hello world 出力テキストファイル

この例は、PowerShell スクリプトを実行して C: ドライブ上に「Hello world」テキストファイルを作成する executeScript タスクを示しています。

```
task: executeScript
  inputs:
    - frequency: always
      type: powershell
      runAs: admin
      content: |-
        New-Item -Path 'C:\PowerShellTest.txt' -ItemType File
        Set-Content 'C:\PowerShellTest.txt' "Hello world"
```

例 2: 2 つのスクリプトを実行する

この例は、executeScript タスクが複数のスクリプトを実行できること、およびスクリプトの種類が必ずしも一致する必要がないことを示しています。

最初のスクリプト (type: powershell) は、インスタンス上で現在実行されているプロセスの概要を C: ドライブ上のテキストファイルに書き込みます。

2 番目のスクリプト (batch) は、システム情報を Output.tmp ファイルに書き込みます。

```
task: executeScript
  inputs:
    - frequency: always
      type: powershell
      runAs: localSystem
      content: |
        Get-Process | Out-File -FilePath C:\Process.txt
    - frequency: always
      type: batch
      runAs: localSystem
      content: |
        systeminfo
```

例 3: 再起動を伴うべき等システム設定

この例は、べき等スクリプトを実行して、各ステップの間に再起動しながら、次のシステム設定を実行する executeScript タスクを示しています。

- コンピュータの名前を変更します。
- コンピュータをドメインに参加させます。

- Telnet を有効にします。

スクリプトは、各オペレーションが 1 回だけ実行されるようにします。これにより、再起動ループが防止され、スクリプトがべき等になります。

```
task: executeScript
  inputs:
    - frequency: always
      type: powershell
      runAs: localSystem
      content: |-
        $name = $env:ComputerName
        if ($name -ne $desiredName) {
          Rename-Computer -NewName $desiredName
          exit 3010
        }
        $domain = Get-ADDomain
        if ($domain -ne $desiredDomain)
        {
          Add-Computer -DomainName $desiredDomain
          exit 3010
        }
        $telnet = Get-WindowsFeature -Name Telnet-Client
        if (-not $telnet.Installed)
        {
          Install-WindowsFeature -Name "Telnet-Client"
          exit 3010
        }
```

extendRootPartition

ルートボリュームを拡張して、ディスクのすべての使用可能な領域を使用します。

Frequency - 1 回

AllowedStages - [Boot]

Inputs - なし

例

```
task: extendRootPartition
```

initializeVolume

インスタンスにアタッチされた空のボリュームを初期化して、アクティブ化およびパーティション化できるようにします。起動エージェントは、ボリュームが空でないことを確認した場合、初期化をスキップします。ボリュームの最初の 4 KiB が空の場合、またはボリュームに Windows で認識可能なドライブレイアウトがない場合、そのボリュームは空とみなされます。

`letter` 入力パラメータは、ドライブがすでに初期化されているかどうかを問わず、このタスクの実行時に常に適用されます。

`initializeVolume` タスクは以下のアクションを実行します。

- ディスク属性 `offline` と `readonly` を false に設定します。
- パーティションを作成します。`partition` 入力パラメータでパーティションタイプが指定されていない場合、以下のデフォルトが適用されます。
  - ディスクサイズが 2 TB 未満の場合、パーティションタイプを `mbr` に設定します。
  - ディスクサイズが 2 TB 以上の場合、パーティションタイプを `gpt` に設定します。
- ボリュームを NTFS としてフォーマットします。
- ボリュームラベルを次のように設定します。
  - 指定されている場合、`name` 入力パラメータの値を使用します。
  - ボリュームが エフェメラルで名前が指定されていない場合、ボリュームラベルを `Temporary Storage Z` に設定します。
- ボリュームがエフェメラル (Amazon EBS ではなく SSD または HDD) である場合は、ボリュームのルートに次の内容を持つ `Important.txt` ファイルを作成します。

```
This is an 'Instance Store' disk and is provided at no additional charge.

*This disk offers increased performance since it is local to the host
*The number of Instance Store disks available to an instance vary by instance type
*DATA ON THIS DRIVE WILL BE LOST IN CASES OF IMPAIRMENT OR STOPPING THE INSTANCE.
 PLEASE ENSURE THAT ANY IMPORTANT DATA IS BACKED UP FREQUENTLY

For more information, please refer to: EC2 ##############################.
```

- ドライブ文字を `letter` 入力パラメータで指定された値に設定します。

Stages:: `PostReady` および `UserData` のステージ中に `initializeVolume` タスクを実行できます。

Frequency: 常時。

入力

ランタイムパラメータは次のように設定できます。

devices (マップのリスト)

(条件付き) 起動エージェントが初期化する各デバイスの設定です。initialize 入力パラメータが devices に設定されている場合に必須です。

- device (文字列、必須) — インスタンスの作成中にデバイスを識別します。例えば、xvdb、xvdf、または \dev\nvme0n1 です。
- letter (文字列、オプション) — 1 文字です。割り当てるドライブ文字です。
- name (文字列、オプション) — 割り当てるボリューム名です。
- partition (文字列、オプション) — 作成するパーティションのタイプに次のいずれかの値を指定するか、ボリュームサイズに基づいて起動エージェントをデフォルトにします。
  - mbr
  - gpt

initialize (文字列)

(必須) 次の値のいずれかを正確に指定します。

- all
- devices

例

以下は、initializeVolume タスクの入力構成の例です。

例 1: インスタンス上の 2 つのボリュームを初期化

こちらは、インスタンス上の 2 つのセカンダリボリュームを初期化する initializeVolume タスクの例です。この例の DataVolume2 という名前のデバイスはエフェメラルです。

```
task: initializeVolume
inputs:
  initialize: devices
  devices:
  - device: xvdb
```

```
    name: DataVolume1
    letter: D
    partition: mbr
  - device: /dev/nvme0n1
    name: DataVolume2
    letter: E
    partition: gpt
```

例 2: インスタンスにアタッチされた EBS ボリュームを初期化

こちらは、インスタンスにアタッチされた空の EBS ボリュームをすべて初期化する `initializeVolume` タスクの例です。

```
task: initializeVolume
inputs:
  initialize: all
```

optimizeEna

現在のインスタンスタイプに基づいて ENA 設定を最適化します。インスタンスは再起動される場合があります。

Frequency - 常に

AllowedStages - `[PostReady, UserData]`

Inputs - なし

例

```
task: optimizeEna
```

setAdminAccount

ローカルマシンに作成されるデフォルトの管理者アカウントの属性を設定します。

Frequency - 1 回

AllowedStages - `[PreReady]`

Inputs -

name: (文字列) 管理者アカウントの名前

password: (マップ)

type: (文字列) パスワードを設定する戦略 (static、random、doNothing のいずれかとして設定)

data: (文字列) type フィールドが静的な場合にデータを保存

例

```
task: setAdminAccount
inputs:
  name: Administrator
  password:
  type: random
```

setDnsSuffix

検索サフィックスのリストに DNS サフィックスを追加します。まだ存在しないサフィックスのみがリストに追加されます。起動エージェントによる DNS サフィックスの設定方法の詳細については、「EC2 Windows 起動エージェントの DNS サフィックスを設定する」を参照してください。

Frequency - 常に

AllowedStages - [PreReady]

Inputs -

suffixes: (文字列のリスト) 1 つ以上の有効な DNS サフィックスのリスト (有効な代替変数は $REGION と $AZ)

例

```
task: setDnsSuffix
inputs:
  suffixes:
  - $REGION.ec2-utilities.amazonaws.com
```

setHostName

コンピュータのホスト名を、カスタム文字列に設定します。また、hostName が指定されていない場合は、プライベート IPv4 アドレスに設定します。

Frequency - 常に

AllowedStages - [PostReady, UserData]

Inputs -

hostName: (文字列) オプションのホスト名。次のようにフォーマットする必要があります。

- 15 文字以下にする必要があります
- 英数字 (a～z、A～Z、0～9) とハイフン (-) のみを使用する必要があります。
- 数字だけで構成することはできません。

reboot: (ブール値) ホスト名の変更時に再起動を許可するかどうかを示す

例

```
task: setHostName
inputs:
  reboot: true
```

setWallpaper

既存の (Default User を除く) 各ユーザーのスタートアップフォルダ内に、ショートカットファイル setwallpaper.lnk を作成します。このショートカットファイルは、ユーザーがインスタンスの起動後に初めてログインしたときに実行されます。インスタンス属性を表示するカスタム壁紙があるインスタンスが設定されます。

ショートカットパスは次のとおりです。

```
$env:SystemDrive/Users/<user>/AppData/Roaming/Microsoft/Windows/Start Menu/Programs/
Startup/setwallpaper.lnk
```

> **Note**
>
> この setWallpaper タスクを削除しても、このショートカットファイルは削除されません。詳細については、「setWallpaper タスクは有効になっていないものの、再起動時にウォールペーパーがリセットされる」を参照してください。

Stages: PreReady および UserData の各ステージ中にウォールペーパーを設定できます。

頻度: always

壁紙設定

次の設定を使用して壁紙を設定できます。

入力

指定した入力パラメータと、壁紙を設定するために設定できる属性:

path (文字列)

(必須) 壁紙画像に使用するローカル .jpg 形式の画像ファイルのファイル名パス。

属性 (文字列のリスト)

(オプション) 壁紙には、次の属性を 1 つ以上追加できます。

- architecture
- availabilityZone
- hostName
- instanceId
- instanceSize
- privateIpAddress
- publicIpAddress
- ipv6Address

InstanceTags

(オプション) この設定には、次のオプションを 1 つだけ使用できます。

- AllTags (文字列) - すべてのインスタンスタグを壁紙に追加します。

```
instanceTags: AllTags
```

- instanceTags (文字列のリスト) - 壁紙に追加するインスタンスタグ名のリストを指定します。例えば、次のようになります。

```
instanceTags:
  - Tag 1
```

```
  - Tag 2
```

例

次の例は、壁紙背景画像のファイルパス、Tag 1 および Tag 2 という名前のインスタンスタグ、およびインスタンスのホスト名、インスタンス ID、プライベート IP アドレスとパブリック IP アドレスを含む属性を設定する壁紙設定入力を示しています。

```
task: setWallpaper
inputs:
  path: C:\ProgramData\Amazon\EC2Launch\wallpaper\Ec2Wallpaper.jpg
  attributes:
  - hostName
  - instanceId
  - privateIpAddress
  - publicIpAddress
  instanceTags:
  - Tag 1
  - Tag 2
```

> **Note**
>
> 壁紙にタグを表示するには、メタデータのタグを有効にする必要があります。インスタンスのタグおよびメタデータの詳細については、「インスタンスメタデータを使用して EC2 インスタンスのタグを表示する」を参照してください。

startSsm

Sysprep に続けて Systems Manager (SSM) サービスを開始します。

Frequency - 常に

AllowedStages - [PostReady, UserData]

Inputs - なし

例

```
task: startSsm
```

sysprep

サービスの状態のリセット、unattend.xml の更新、RDP の無効化、Sysprep の実行を行います。このタスクは、他のすべてのタスクが完了した後にのみ実行されます。

Frequency - 1 回

AllowedStages - [UserData]

Inputs -

clean: (ブール) Sysprep を実行する前にインスタンスログを消去する

shutdown: (ブール) Sysprep を実行した後にインスタンスをシャットダウンする

例

```
task: sysprep
inputs:
clean: true
shutdown: true
```

writeFile

ファイルを送信先に書き込みます。

Frequency - Inputs を参照

AllowedStages - [PostReady, UserData]

Inputs -

frequency: (文字列) once または always のいずれか

destination: (文字列) コンテンツを書き込む先のパス

content: (文字列) 送信先に書き込むテキスト

例

```
task: writeFile
inputs:
  - frequency: once
  destination: C:\Users\Administrator\Desktop\booted.txt
  content: Windows Has Booted
```

EC2Launch v2 エージェントの問題をトラブルシューティングする

このセクションでは、EC2Launch v2 での一般的なトラブルシューティングシナリオ、Windows イベントログの表示に関する情報、およびコンソールログの出力とメッセージについて説明します。

トラブルシューティングのトピック

- 一般的なトラブルシューティングのシナリオ
- Windows イベントログ
- EC2Launch v2 コンソールログ出力

一般的なトラブルシューティングのシナリオ

このセクションでは、一般的なトラブルシューティングのシナリオと解決手順を示します。

シナリオ

- サービスで壁紙を設定できない
- サービスでユーザーデータを実行できない
- サービスはタスクを 1 回だけ実行する
- サービスでタスクを実行できない
- サービスはユーザーデータを複数回実行する
- EC2Launch v2 への移行後、EC2Launch v1 のスケジュールされたタスクが実行されない
- 空ではない EBS ボリュームがサービスによって初期化される
- setWallpaper タスクは有効になっていないものの、再起動時にウォールペーパーがリセットされる
- サービスは実行中ステータスでスタックしています
- 無効なagent-config.ymlは EC2Launch v2 設定ダイアログボックスが開くことを防止します
- task:executeScript should be unique and only invoked once

サービスで壁紙を設定できない

解決策

1. `%AppData%\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\setwallpaper.lnk` が存在することを確認します。

2. %ProgramData%\Amazon\EC2Launch\log\agent.log をチェックし、エラーが発生したかどうかを確認します。

サービスでユーザーデータを実行できない

考えられる原因: ユーザーデータを実行する前にサービスが失敗した可能性があります。

解決策

1. %ProgramData%\Amazon\EC2Launch\state\previous-state.json をチェックします。
2. boot、network、preReady、postReadyLocalData のすべてが成功とマークされているかどうかを確認します。
3. いずれかのステージが失敗した場合は、特定のエラーの %ProgramData%\Amazon\EC2Launch\log\agent.log を確認します。

サービスはタスクを 1 回だけ実行する

解決策

1. タスクの頻度を確認します。
2. サービスが Sysprep 後に実行済みであり、タスクの頻度が once に設定されている場合、タスクは再度実行されません。
3. EC2Launch v2 が実行されるたびにタスクを実行する場合は、タスクの頻度を always に設定します。

サービスでタスクを実行できない

解決策

1. %ProgramData%\Amazon\EC2Launch\log\agent.log の最新のエントリを確認します。
2. エラーが発生しなかった場合は、"%ProgramFiles%\Amazon\EC2Launch\EC2Launch.exe" run からサービスを手動で実行し、タスクが成功するかどうかを確認します。

サービスはユーザーデータを複数回実行する

解決策

ユーザーデータは、EC2Launch v1 と EC2Launch v2 では異なる方法で処理されます。persist が true に設定されている場合、EC2Launch v1 はユーザーデータをインスタンスでスケジュールされたタスクとして実行します。persist が false に設定されている場合、タスクが再起動して終了したり、実行中に中断された場合でも、タスクはスケジュールされません。

EC2Launch v2 は、ユーザーデータをエージェントタスクとして実行し、実行状態を追跡します。ユーザーデータがコンピュータの再起動を発行した場合、または実行中にユーザーデータが中断された場合、実行状態は pending として持続し、ユーザーデータは次回のインスタンスの起動時に再び実行されます。ユーザーデータスクリプトが複数回実行されないようにするには、スクリプトをべき等にします。

次のべき等スクリプト例では、コンピュータ名を設定し、ドメインに参加します。

```
<powershell>
  $name = $env:computername
  if ($name -ne $desiredName) {
 Rename-Computer -NewName $desiredName
  }
  $domain = Get-ADDomain
  if ($domain -ne $desiredDomain)
  {
 Add-Computer -DomainName $desiredDomain
  }
  $telnet = Get-WindowsFeature -Name Telnet-Client
  if (-not $telnet.Installed)
  {
 Install-WindowsFeature -Name "Telnet-Client"
  }
</powershell>
<persist>false</persist>
```

EC2Launch v2 への移行後、EC2Launch v1 のスケジュールされたタスクが実行されない

解決策

移行ツールは、EC2Launch v1 スクリプトにリンクされているスケジュールされたタスクを検出しないため、EC2Launch v2 のタスクは自動的にセットアップされません。タスクを設定するには、agent-config.yml ファイルを編集するか、EC2Launch v2 設定ダイアログボックスを使用します。例えば、インスタンスのタスクが InitializeDisks.ps1 を実行するようにスケジュールされている場合は、移行ツールを実行した後、EC2Launch v2 の設定ダイアログボックスで、初期化す

るボリュームを指定する必要があります。EC2Launch v2 設定ダイアログボックスを使用して設定を変更する の手順については、ステップ 6 を参照してください。

空ではない EBS ボリュームがサービスによって初期化される

解決策

ボリュームが初期化される前に、EC2Launch v2 は、そのボリュームにデータがないこと確認します。ボリュームが空でない場合、初期化はスキップされます。空でないことが検出されたボリュームは初期化されません。ボリュームの最初の 4 KiB が空の場合、またはボリュームに Windows を認識できるドライブレイアウトがない場合、ボリュームは空と見なされます。Linux システムで初期化およびフォーマットされたボリュームには、Windows を認識できるドライブレイアウト (MBR や GPT など) がありません。したがって、それは空とみなされ、初期化されます。このデータを保持する場合、EC2Launch v2 による空ドライブの検出には頼らないでください。代わりに、EC2Launch v2 設定ダイアログボックス (ステップ 6 を参照) または `agent-config.yml` で、初期化するボリュームを指定します。

`setWallpaper` タスクは有効になっていないものの、再起動時にウォールペーパーがリセットされる

`setWallpaper` タスクにより、すべての既存ユーザーのスタートアップフォルダ内に `setwallpaper.lnk` ショートカットファイルが作成されます。このショートカットファイルは、ユーザーがインスタンスの起動後に初めてログインしたときに実行されます。インスタンス属性を表示するカスタム壁紙があるインスタンスが設定されます。この `setWallpaper` タスクを削除しても、このショートカットファイルは削除されません。このファイルは手動で削除するか、スクリプトを使用して削除する必要があります。

ショートカットパスは次のとおりです。

`$env:SystemDrive/Users/<user>/AppData/Roaming/Microsoft/Windows/Start Menu/Programs/Startup/setwallpaper.lnk`

解決策

このファイルを手動で削除するか、スクリプトを使用して削除します。

ショートカットファイルを削除する PowerShell スクリプトの例

```
foreach ($userDir in (Get-ChildItem "C:\Users" -Force -Directory).FullName)
{
```

```
 $startupPath = Join-Path $userDir -ChildPath "AppData\Roaming\Microsoft\Windows\Start
 Menu\Programs\Startup"
 if (Test-Path $startupPath)
 {
  $wallpaperSetupPath = Join-Path $startupPath -ChildPath "setwallpaper.lnk"
  if (Test-Path $wallpaperSetupPath)
  {
   Remove-Item $wallpaperSetupPath -Force -Confirm:$false
  }
 }
}
```

サービスは実行中ステータスでスタックしています

説明

EC2Launch v2 は、次のようなログ (agent.log) でブロックされます。

```
2022-02-24 08:08:58 Info:
 ********************************************************************************
2022-02-24 08:08:58 Info: EC2Launch Service starting
2022-02-24 08:08:58 Info: Windows event custom log exists: Amazon EC2Launch
2022-02-24 08:08:58 Info: ACPI SPCR table not supported. Bailing Out
2022-02-24 08:08:58 Info: Serial port is in use. Waiting for Serial Port...
2022-02-24 08:09:00 Info: ACPI SPCR table not supported. Use default console port.
2022-02-24 08:09:02 Info: ACPI SPCR table not supported. Use default console port.
2022-02-24 08:09:04 Info: ACPI SPCR table not supported. Use default console port.
2022-02-24 08:09:06 Info: ACPI SPCR table not supported. Use default console port.
```

考えられる原因

SAC が有効化され、シリアルポートを使用しています。詳細については、「SAC を使用して Windows インスタンスをトラブルシューティングする」を参照してください。

解決策

この問題を解決するには、以下の手順を実行します。

- このシリアルポートを使用しているサービスを、無効にします。
- サービスでこのシリアルポートを引き続き使用する場合は、起動エージェントタスクを実行するカスタムスクリプトを作成し、スケジュールされたタスクとして起動します。

無効な**agent-config.yml**は EC2Launch v2 設定ダイアログボックスが開くことを防止します

説明

EC2Launch v2 設定は、ダイアログボックスを開く前に agent-config.yml ファイルの解析を試みます。YAML 設定ファイルがサポートされているスキーマに準拠しない場合、ダイアログボックスに次のエラーが表示されます。

```
Unable to parse configuration file agent-config.yml. Review configuration file. Exiting application.
```

解決策

1. 設定ファイルが [supported schema] (サポートされているスキーマ) に準拠していることを検証します。
2. 最初から始めたい場合は、デフォルトの設定ファイルを agent-config.yml にコピーします。「タスク設定セクション」に提供されている サンプル agent-config.yml を使用できます。
3. agent-config.yml を削除して最初からやり直すこともできます。EC2Launch v2 設定は空の設定ファイルを生成します。

task:executeScript should be unique and only invoked once

説明

同じステージでタスクを繰り返すことはできません。

解決策

一部のタスクは、executeScript と executeProgram などの配列として入力する必要があります。スクリプトを配列として記述する方法の例については、「executeScript」を参照してください。

Windows イベントログ

EC2Launch v2 は、サービスの開始、Windows の準備完了、さらにタスクの成否などの重要なイベントに関する Windows イベントログを発行します。イベント識別子は、特定のイベントを一意に識別します。各イベントには、ステージ、タスク、レベル情報、説明が含まれます。イベント識別子を使用して、特定のイベントのトリガーを設定できます。

イベント識別子は、イベントに関する情報を提供し、いくつかのイベントを一意に識別します。イベント ID の最下位桁は、イベントの重大度を示します。

イベント	最下位桁
Success	. . .0
Informational	. . .1
Warning	. . .2
Error	. . .3

サービスの開始時または停止時に生成されるサービスに関するイベントには、1 桁のイベント識別子が含まれます。

イベント	1 桁の識別子
Success	0
Informational	1
Warning	2
Error	3

EC2LaunchService.exe イベントのイベントメッセージは、Service: から始まります。EC2Launch.exe イベントのイベントメッセージは、Service: で始まるものではありません。

4 桁のイベント ID には、イベントのステージ、タスク、重大度に関する情報が含まれます。

トピック

- イベント ID 形式
- イベント ID の例
- Windows イベントログのスキーマ

イベント ID 形式

次の表に EC2Launch v2 でのイベント識別子の形式を示します。

3	2 1	0
S	T	L

表内の文字と数字は、次のイベントタイプと定義を表します。

イベントタイプ	定義
S (ステージ)	0 - サービスレベルのメッセージ 1 - Boot 2 - Network 3 - PreReady 5 - Windows is Ready 6 - PostReady 7 - User Data
T (タスク)	対応する 2 つの値で表されるタスクは、ステージごとに異なります。イベントの詳細なリストを表示するには、Windows イベントログのスキーマを参照してください。
L (イベントのレベル)	0 - 成功 1 - 情報 2 - 警告 3 - エラー

イベント ID の例

イベント ID の例を次に示します。

- 5000 - Windows の使用準備完了
- 3010 - PreReady ステージの Windows のアクティブ化タスクが成功しました
- 6013 - PostReady ローカルデータステージの壁紙の設定タスクでエラーが発生しました

Windows イベントログのスキーマ

メッセージ ID/イベント ID	イベントメッセージ
. . .0	Success
. . .1	Informational
. . .2	Warning
. . .3	Error
x	EC2Launch service-level logs
0	EC2Launch service exited successfully
1	EC2Launch service informational logs
2	EC2Launch service warning logs
3	EC2Launch service error logs
10	Replace state.json with previous-state.json
100	Serial Port
200	Sysprep
300	PrimaryNic

メッセージ ID/イベント ID	イベントメッセージ
400	Metadata
x000	Stage (1 digit), Task (2 digits), Status (1 digit)
1000	Boot
1010	Boot - extend_root_partition
2000	Network
2010	Network - add_routes
3000	PreReady
3010	PreReady - activate_windows
3020	PreReady - install_egpu_manager
3030	PreReady - set_monitor_on
3040	PreReady - set_hibernation
3050	PreReady - set_admin_account
3060	PreReady - set_dns_suffix
3070	PreReady - set_wallpaper
3080	PreReady - set_update_schedule
3090	PreReady - output_log
3100	PreReady - enable_open_ssh
5000	Windows is Ready to use
6000	PostReadyLocalData
7000	PostReadyUserData

メッセージ ID/イベント ID	イベントメッセージ
6010/7010	PostReadyLocal/UserData - set_wallpaper
6020/7020	PostReadyLocal/UserData - set_update_schedule
6030/7030	PostReadyLocal/UserData - set_hostname
6040/7040	PostReadyLocal/UserData - execute_program
6050/7050	PostReadyLocal/UserData - execute_script
6060/7060	PostReadyLocal/UserData - manage_package
6070/7070	PostReadyLocal/UserData - initialize_volume
6080/7080	PostReadyLocal/UserData - write_file
6090/7090	PostReadyLocal/UserData - start_ssm
7100	PostReadyUserData - enable_open_ssh
6110/7110	PostReadyLocal/UserData - enable_jumbo_frames

EC2Launch v2 コンソールログ出力

このセクションには、サンプルの EC2Launch v2 コンソールログ出力が含まれており、EC2Launch v2 コンソールログエラーメッセージのうち、問題のトラブルシューティングに役立つものを一覧表

示しています。インスタンスコンソール出力とそのアクセス方法の詳細については、「the section called “インスタンスコンソール出力”」を参照してください。

アウトプット

- EC2Launch v2 コンソールログ出力
- EC2Launch v2 コンソールログメッセージ

EC2Launch v2 コンソールログ出力

以下に示しているのは、サンプルの EC2Launch v2 コンソールログ出力です。この例の一部の値は、中括弧で囲まれた代表的なテキストに置き換えられます。

```
2025/07/22 21:26:53Z: Windows sysprep configuration complete.
2025/07/22 21:26:53Z: Message: Waiting for access to metadata...
2025/07/22 21:26:53Z: Message: Meta-data is now available.
2025/07/22 21:26:53Z: AMI Origin Version: 2024.12.13
2025/07/22 21:26:53Z: AMI Origin Name: Windows_Server-2022-English-Full-Base
2025/07/22 21:26:53Z: OS: Microsoft Windows NT 10.0.20348
2025/07/22 21:26:53Z: OsVersion: 10.0
2025/07/22 21:26:53Z: OsProductName: Windows Server 2022 Datacenter
2025/07/22 21:26:53Z: OsBuildLabEx: 20348.1.amd64fre.fe_release.210507-1500
2025/07/22 21:26:53Z: OsCurrentBuild: 20348
2025/07/22 21:26:53Z: OsReleaseId: 2009
2025/07/22 21:26:53Z: Language: en-US
2025/07/22 21:26:53Z: TimeZone: UTC
2025/07/22 21:26:53Z: Offset: UTC +0000
2025/07/22 21:26:53Z: Launch: EC2 Launch v2.2.63
2025/07/22 21:26:53Z: AMI-ID: ami-1234567890abcdef1
2025/07/22 21:26:53Z: Instance-ID: i-1234567890abcdef0
2025/07/22 21:26:54Z: Instance Type: t3.xlarge
2025/07/22 21:26:54Z: Driver: AWS NVMe Driver v1.6.0.35
2025/07/22 21:26:54Z: SubComponent: 1.6.0.35; EnableSCSIPersistentReservations: 0
2025/07/22 21:26:54Z: Driver: AWS PV Driver Package v8.5.0
2025/07/22 21:26:55Z: Driver: Amazon Elastic Network Adapter v2.8.0.0
2025/07/22 21:26:55Z: Driver: EC2 Windows Utility Device v3.1.0.25
2025/07/22 21:26:55Z: HOSTNAME: EC2AMAZ-9FJG5CC
2025/07/22 21:26:55Z: RDPCERTIFICATE-SUBJECTNAME: {certificate subject name}
2025/07/22 21:26:55Z: RDPCERTIFICATE-THUMBPRINT: {thumbprint hash}
2025/07/22 21:26:56Z: SSM: Amazon SSM Agent v3.3.2746.0
2025/07/22 21:26:57Z: User data format: no_user_data
2025/07/22 21:26:57Z: EC2LaunchTelemetry: IsTelemetryEnabled=true
2025/07/22 21:26:57Z: EC2LaunchTelemetry: AgentOsArch=windows_amd64
```

```
2025/07/22 21:26:57Z: EC2LaunchTelemetry: IsAgentScheduledPerBoot=true
2025/07/22 21:26:57Z: EC2LaunchTelemetry: AgentCommandErrorCode=1
2025/07/22 21:26:57Z: EC2LaunchTelemetry: AdminPasswordTypeCode=0
2025/07/22 21:26:57Z: EC2LaunchTelemetry: AgentErrorLocation=execute_windows.go:410
2025/07/22 21:26:57Z: EC2LaunchTelemetry: IpConflictDetectionCode=0
2025/07/22 21:26:57Z: Message: Windows is Ready to use
{"type":"EC2AgentTelemetry","agentId":"WindowsLaunchAgentV2", ...}
{"type":"EC2AgentTelemetry","agentId":"WindowsLaunchAgentV2", ...}
```

EC2Launch v2 コンソールログメッセージ

以下に示しているのは、すべての EC2Launch v2 コンソールログメッセージのリストです。

```
Error EC2Launch service is stopping. {error message}
```

停止したサービスエラーの詳細：

- `Error setting up EC2Launch agent folders`
- `See instance logs for detail`
- `Error stopping service`
- `Error initializing service`

```
Windows sysprep configuration complete
```

```
Invalid administrator username: {invalid username}
```

```
Invalid administrator password
Username: {username}
Password: <Password>{encrypted password}</Password>
```

次のメッセージは、AMI の詳細を含む情報ブロックです。

```
AMI Origin Version: {amiVersion}
AMI Origin Name: {amiName}
Microsoft Windows NT {currentVersion}.{currentBuildNumber}
OsVersion: {currentVersion}
OsProductName: {productName}
OsBuildLabEx: {buildLabEx}
OsCurrentBuild: {currentBuild}
```

```
OsReleaseId: {releaseId}
Language: {language}
TimeZone: {timeZone}
Offset: UTC {offset}
Launch agent: EC2Launch {BuildVersion}
AMI-ID: {amiId}
Instance-ID: {instanceId}
Instance Type: {instanceType}
HOSTNAME: {computer name}
RDPCERTIFICATE-SUBJECTNAME: {certificate subject name}
RDPCERTIFICATE-THUMBPRINT: {thumbprint hash}
SqlServerBilling: {sql billing}
SqlServerInstall: {sql patch leve, edition type}
Driver: AWS NVMe Driver {version}
Driver: Inbox NVMe Driver {version}
Driver: AWS PV Driver Package {version}
Driver: EC2 Windows Utility Device {version}
SSM: Amazon SSM Agent {version}
AWS VSS Version: {version}
```

```
Windows sysprep configuration complete.
Windows is being configured. 'SysprepState is {state}'
Windows is still being configured. 'SysprepState is {state}'
Windows is Ready to use
Waiting for access to metadata...
Meta-data is now available.
Metadata is not available for this instance.
Timed out waiting for access to metadata.
User data format: {format}
```

EC2Launch v2 テレメトリメッセージには、起動テレメトリプロパティ値が含まれます。バージョン 2.2.63 以降、EC2 エージェントテレメトリデータは JSON オブジェクトとしてフォーマットされます。

```
EC2LaunchTelemetry: {telemetry property}
```

```
{"type":"EC2AgentTelemetry","agentId":"WindowsLaunchAgentV2" ... }
```

EC2Launch v2 のバージョン履歴

バージョン履歴

- EC2Launch v2 のバージョン履歴

EC2Launch v2 のバージョン履歴

最新の起動エージェントがインストールされていることを確認するには、「EC2Launch v2 の最新バージョンのインストール」を参照してください。EC2Launch v2 エージェントの新しいバージョンがリリースされた際に、通知を受け取ることができます。詳細については「EC2 Windows 起動エージェント通知をサブスクライブする」を参照してください。

次の表で、EC2Launch v2 のリリース済みバージョンについて説明します。

バージョン	詳細	リリース日
2.5.1	• 休止を有効にする際のエラー検出が改善されました。	2026 年 6 月 3 日
2.5.0	• シリアルポートが利用できない場合でもエージェントタスクを継続できるよう、シリアルポートのロジックを更新しました。 • インスタンスのコンソール出力に、EC2 Windows ユーティリティ デバイス ドライバーのバージョンを追加しました。 • 使用されていない bench.log ファイルを削除しました。 • インスタンスのコンソール出力への書き込み時のドライバー検索を最適化しました。	2026 年 5 月 4 日
2.4.0	• 冗長な `console.log` 、 `err.log` ファイルを削除することで、エージェントのログ記録を簡素化しました。 • ログローテーションを追加し、ログサイズを 1 MB に制限し、時間精度をミリ秒に改善することで、エージェントのログ記録機能を強化しました。 • Hyper-V のステータス照会および関連するインスタンスコンソールの出力を削除しました。	2026 年 3 月 5 日

バージョン	詳細	リリース日
2.3.237	• 特定の Visual C++ ランタイムバージョンとエージェントテレメトリ関数間における非互換性が原因でエージェントがクラッシュする問題を修正しました。	2026 年 2 月 25 日
2.3.108	• 壁紙に IPv6 アドレス表示を追加しました。メモリ属性とネットワーク属性については、AWS ドキュメントを参照してください。	2025 年 12 月 29 日
2.3.56	• EC2Launch v2 で事前設定済みの永続的ネットワークルートが適切に処理されない問題を修正しました。	2025 年 11 月 4 日
2.3.5	• ルートを追加する際の IMDS を使用したプライマリネットワークインターフェイス検出が改善されました。	2025 年 9 月 15 日
2.2.63	• 起動エージェントを更新して テレメトリ を改善しました。	2025 年 7 月 29 日
2.1.1	• IPv6 に対する完全なサポートを追加しました。	2025 年 5 月 14 日
2.0.2107	• IPv4 アドレスまたは IPv6 アドレスがインターフェイスで使用できないシナリオを処理するためのルートの追加を強化しました。	2025 年 3 月 27 日
2.0.2081	• RDP 証明書情報が適切に取得または検証されなかった問題を修正しました。必要に応じてリモートデスクトップサービスを自動的に開始する機能を追加しました。 • EC2Launch v2 サービスのアクセス許可を調整して、サービスステータスのクエリするときに発生する問題を修正しました。	2025 年 2 月 4 日

バージョン	詳細	リリース日
2.0.2046	• デフォルトのオペレーティングシステム壁紙パスを使用するように agent-config.yml ファイルの壁紙パスを更新しました。 • エージェントエラーの発生場所をモニタリングするためのテレメトリを追加しました。 • エージェントログメッセージングを更新しました。	2024 年 10 月 3 日
2.0.1981	• 管理者以外のユーザーに対する EC2Launch.exe CLI コマンドエラーメッセージが更新されました。	2024 年 8 月 6 日
2.0.1948	• 管理者パスワードのオプションの使用状況をモニタリングするためのテレメトリを追加しました。 • EC2Launch のアクセス許可を修正しました。	2024 年 7 月 1 日
2.0.1924	• EC2Launch 設定 UI を更新しました。 • 壁紙 CLI コマンドを更新しました。 • EC2Launch インストーラを更新しました。	2024 年 6 月 10 日
2.0.1914	• ゲートウェイアドレスが指定されていないルートが追加されます (IPv4 では 0.0.0.0、IPv6では ::)。 • IPv4 ルートと IPv6 ルートの両方が必ず追加されます。 • Administrator ユーザー名が指定されていないにもかかわらず agent-config.yml ファイルに追加される問題が修正されました。 • EC2Launch v2 のアクセス許可を修正しました。	2024 年 6 月 5 日

バージョン	詳細	リリース日
2.0.1881	• 暗号化されたパスワードオプションを `setAdminAccount` タスクに追加しました。 • agent-config.yml で静的パスワードを暗号化する CLI コマンドを追加しました。 • XML ユーザーデータが、管理者アクセス許可での実行時に PowerShell 引数を追加しない問題を修正しました。詳細については、Amazon EC2 が Windows インスタンスのユーザーデータを処理する方法を参照してください。 • `LocalSystem` アクセス許可で実行された `executeScript` タスクおよびユーザーデータスクリプトの PowerShell 引数を調整しました。引数が空の場合、エージェントは次のデフォルト値を使用します: `-ExecutionPolicy Unrestricted` 。 • 重複するドライバーバージョンをコンソールログに出力しないようにしました。	2024 年 5 月 8 日
2.0.1815	• sysprep の前に重大なセットアップの問題で失敗するようにエラー処理を調整しました。 • プライマリネットワークインターフェイスに複数の IP アドレスが割り当てられているインスタンスで、壁紙とホスト名のタスクが間違った IP アドレスが使用する可能性がある問題を修正しました。 • 壁紙とホスト名のタスクは、まず IMDS からプライベート IP を取得し、IMDS が無効になっている場合は WMI にフェイルバックするように変更されました。 • 一時的なエラーが原因で `sc1` ボリュームの初期化に失敗する `initializeVolume` タスクの問題を修正しました。	2024 年 3 月 6 日

バージョン	詳細	リリース日
2.0.1739	• Windows Administrator ユーザーとして実行された `executeScript` タスクによって終了コードがキャプチャされない問題を修正しました。	2024 年 1 月 17 日
2.0.1702	• 標準ユーザーに関して、`Telemetry.log` 権限を `read-execute` のみに制限しました。 • EC2Launch Windows サービスが起動に失敗したときに再起動するように設定しました。 • `route.exe` `stderr` 出力をログに記録することで `add-routes` の失敗に対処できるようにしました。 • ルートメトリックが [1, 9999] の範囲外の場合に発生する問題を修正しました。 • いくつかの新しいインスタンスタイプに壁紙のサポートが追加されました。 • Windows 管理者ユーザーとして実行され、出力を `stderr` に送信するユーザーデータスクリプトが原因で発生する問題を修正しました。	2024 年 1 月 4 日

バージョン	詳細	リリース日
2.0.1643	• ebsnvme-id.exe ツールがバージョン 1.1.0.7 に更新されました。 • metal-48x1 などの 'metal-*' で始まるメタルインスタンスタイプでの受信側スケーリング (RSS) と受信キューの深さ設定に関する問題を修正しました。 • エージェントをブロックする XML ユーザーデータコマンドを報告するテレメトリイベントを削除しました。 • レジストリエントリに基づいてドメイン名の権限委譲を制限するように setDnsSuffix タスクを更新しました: HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Dnscache\Parameters\DomainNameDevolutionLevel 。 • ネットワークルートを追加するパブリックタスクとCLI を追加しました。 • 注 — これは Windows Server 2012 を公式にサポートする最後のバージョンです。 • 注 — 32 ビットオペレーティングシステムを公式にサポートする最後のバージョンです。	2023 年 10 月 4 日
2.0.1580	• ログファイルのアクセス許可を変更するときの起動エージェントのエラー処理方法を変更しました。 • シリアルポートに接続するためのタイムアウトを追加しました。このタイムアウトにより、シリアルポートが使用中であっても起動エージェントは実行を継続できます。	2023 年 9 月 5 日

バージョン	詳細	リリース日

2.0.1521	• –blockEC2Launch.exe resetsysprepおよびコマンドのフラグは廃止されました。
	• EC2Launch.exe resetsysprepexecuteScript インラインタスクで使用されるおよびコマンドを検出して処理するように更新されました。これらのコマンドにより、executeScript タスクが実行された後にエージェントの実行が停止します。
	• XML ユーザーデータスクリプトがデフォルトでインラインで実行されるように更新されました。
	• XML detach ユーザーデータスクリプトを新しいタグでデタッチして実行できるようにします。詳細については、ユーザーデータスクリプトを参照してください。
	• エージェントログに次の変更を加えました。
	  • エージェントログメッセージを更新しました。
	  • executeScript エージェントログからコンテンツと出力を削除しました。
	  • executeProgram エージェントログから引数と出力を削除しました。
	• コンソールログに次の変更を加えました。
	  • EnableSCSIPersistentReservations コンソールログに値を追加しました。
		2023 年 7 月 3 日

バージョン	詳細	リリース日
2.0.1303	• ネットワークルートの追加時における追加のエラー処理と口グ行が追加されました。 • PreReady ステージで `executeScript` および `executeProgram` タスクが許可されました。 • executeScript タスクからの出力と類似の出力ファイルを生成する `executeProgram` タスクを更新しました。詳細については「executeProgram」を参照してください。 • XML ユーザーデータ内のブロッキングエージェントコマンドの使用状況をモニタリングするためのテレメトリを追加しました。	2023 年 5 月 3 日
2.0.1245	• クラッシュコールスタックをクリアテキストで記録することで、クラッシュの可視性が向上しました。 • Amazon EC2Launch サービスが EventLog サービスよりも速く起動したときに発生するクラッシュを修正するために、スタートアップ依存関係として EventLog サービスを追加しました。 • PostReady ステージの前にエージェント設定ファイル (EC2Launch v1 や EC2Config など) から XML ユーザーデータを実行するようになりました。 • YAML ユーザーデータバージョン 1.1 を追加して、ユーザーデータをエージェント設定ファイルから PostReady ステージの前に実行するようになりました (YAML ユーザーデータバージョン 1.0 は、エージェント設定ファイルの PostReady ステージの後に実行されます)。	2023 年 3 月 8 日

バージョン	詳細	リリース日
2.0.1173	• インスタンスタグを壁紙に表示するオプション機能を追加します。詳細については「setWallpaper」を参照してください。 • Elastic Graphics のセキュリティグループが正しく設定されていない場合のエラー処理を追加します。 • インスタンスメタデータサービスが有効になっていない場合のタイムアウトを修正します。	2023 年 2 月 6 日
2.0.1121	• パブリック IPv4 アドレスが割り当てられていない場合に 404 エラーが壁紙に印刷される問題を修正しました。 • デバイスのドライブ文字が D に設定されている際、ボリュームのファイルシステムが、NTFS ではなく RAW でフォーマットされる問題を修正しました。 • NVMe SSD ボリュームが誤って EBS ボリュームとして識別される問題を修正しました。 • IMDS が無効になっているときに Windows をアクティベーションする際のエラーを修正しました。	2023 年 1 月 4 日

バージョン	詳細	リリース日
2.0.1082	• IMDS が無効になっているときに `setWallpaper : privateIpAddress` フィールドが空白になる問題を修正しました。 • IMDS が無効になっているときにホスト名がプライベート IPv4 アドレスに設定される問題を修正しました。 • Windows Server 2012 でボリュームを初期化する際の問題が修正されました。 • ジャンボフレームの設定に関する問題を修正しました。 • インスタンスの起動時に SSH キーが指定されていない場合のエラーを修正しました。 • Windows に「ReleaseId」レジストリキーがない場合に発生する Windows Server 2012 のエラーを修正しました。	2022 年 12 月 7 日
2.0.1011	• PnPDeviceID が空の場合にネットワークアダプタを検索するロジックが修正されました。	2022 年 11 月 11 日
2.0.1009	• PCI セグメント情報を使用してコンソールポートを選択します。	2022 年 11 月 8 日

バージョン	詳細	リリース日
2.0.982	• RDP 情報を取得するために再試行ロジックを追加します。 • d2.8xlarge インスタンスのボリューム初期化中のエラーを修正します。 • 再起動後に間違ったネットワークアダプターを選択できる問題を修正します。 • ACPI SPCR が使用できない場合の誤ったアラームエラーメッセージを削除します。	2022 年 10 月 31 日
2.0.863	• IMDS 待機ロジックを、IMDSv2 リクエストのみを行うように更新しています。 • 既に初期化されているが、マウントされていないボリュームに、ドライブ文字を割り当てるロジックを追加します。 • キーペアのタイプがサポートされていない場合は、より具体的なエラーメッセージを出力します。 • 3010 リブートコードのバグを修正しました。 • base64 エンコード後の、無効なユーザーデータのチェックを追加します。	2022 年 7 月 6 日
2.0.698	• スクリプト実行時のログ出力での入力ミスを修正しました。	2022 年 1 月 30 日
2.0.674	• テレメトリは、有効/無効のプライバシーコントロールをアップロードします。 • index out of bounds バグの修正。 • sysprep 中に壁紙のショートカットを削除します。	2021 年 11 月 15 日

バージョン	詳細	リリース日
2.0.651	• EC2Launch v2 のインストール中にレガシーエージェントをアンインストールするための、ロジックを追加。 • ルートボリュームがボリューム 0 としてリストされていない場合に `list-volume` (CLI) で発生する問題を修正。	2021 年 10 月 7 日
2.0.592	• ステージのステータスを正しく報告するようにバグを修正しました。 • ログファイルが閉じられたときに、誤ったエラーメッセージを削除します。 • テレメトリーを追加します。	2021 年 8 月 31 日
2.0.548	• 16 進数の IP ホスト名の先頭にゼロを追加します。 • `enableOpenSsh` タスクのファイル許可を修正しました。 • sysprep コマンドのクラッシュが修正されました。	2021 年 8 月 4 日

バージョン	詳細	リリース日
2.0.470	• ネットワークステージで、DHCP がインスタンスに IP を割り当てるのを待つようバグを修正しました。 • `SearchList` のレジストリキーが存在しない場合の `setDnsSuffix` のバグを修正しました。 • `setDnsSuffix` の DNS デボリューションロジックのバグを修正しました。 • 中間リブート後にネットワークルートを追加します。 • 既存のボリュームを再レターすることを `initializeVolume` に許可します。 • バージョンサブコマンドから余分な情報を削除します。	2021 年 7 月 20 日
2.0.285	• デタッチ済みのプロセスでユーザースクリプトを実行するオプションを追加します。 • レガシーユーザーデータ (XML ユーザーデータ) はここでデタッチ済みのプロセスで実行し、これは以前の起動エージェントと同様の挙動です。 • CLI フラグを `sysprep` および `reset` コマンドに追加します。これにより、サービスが停止するまでブロックを許可できます。 • config フォルダのアクセス許可を制限します。	2021 年 3 月 8 日

バージョン	詳細	リリース日
2.0.207	• `setHostName` タスクにオプションの `hostName` フィールドを追加します。 • 再起動のバグを修正します。`executeScript` タスクを再起動し、`executeProgram` が実行中としてマークされます。 • status コマンドにリターンコードを追加します。 • `t2.nano` インスタンスタイプで実行する場合、スタートアップの問題を修正するブートストラップサービスを追加します。 • クリーンインストールモードを修正し、インストーラによって追跡されないファイルを削除します。	2021 年 2 月 2 日
2.0.160	• 無効なステージ名を検出する `validate` コマンドを修正しました。 • `addroutes` タスクに `w32tm resync` コマンドを追加しました。 • DNS サフィックスの検索順序が変わる問題を修正しました。 • チェック条件を追加して、無効なユーザーデータをより適切に報告できるようにしました。	2020 年 12 月 4 日
2.0.153	UserData に Sysprep 機能を追加しました。	2020 年 11 月 3 日

バージョン	詳細	リリース日
2.0.146	• 英語以外の AMI での RootExtend の問題を修正します。 • ユーザーグループにログファイルへの書き込みアクセス許可を付与します。 • GPT ボリューム用の MS 予約済みパーティションを作成します。 • Amazon EC2Launch 設定に list-volumes コマンドとボリュームドロップダウンを追加します。 • agent-config.yml ファイルを yaml または json 形式で出力する get-agent-config コマンドを追加します。 • パブリックキーが検出されない場合に静的パスワードを消去します。	2020 年 10 月 6 日
2.0.124	• 壁紙に OS バージョンを表示するオプションを追加します。 • 暗号化された EBS ボリュームを初期化します。 • ローカル DNS 名のない VPC のルートを追加します。	2020 年 9 月 10 日
2.0.104	• DNS サフィックス検索リストが存在しなければ作成します。 • 休止が必要なければスキップします。	2020 年 8 月 12 日
2.0.0	初回リリース。	2020 年 6 月 30 日

EC2 Windows インスタンスの起動時に EC2Launch v1 エージェントを使用してタスクを実行する

Windows Server 2016 および 2019 用の Amazon マネージド AMI には、EC2Launch と呼ばれる一連の Windows Powershell スクリプトが含まれています。EC2Launch は最初のインスタンスの起動中にタスクを実行します。AWS Windows AMI に含まれる EC2Launch のバージョンについては、「AWS Windows AMI リファレンス」を参照してください。

> **Note**
>
> AWS Windows Server 2016 以降の OS バージョンの最新起動エージェントは EC2Launch v2 であり、EC2Config および EC2Launch の両方を置き換えるもので、`EC2LaunchV2-Windows_Server-*` で始まる名前で Windows Server 2016 および 2019 AMI にプレインストールされています。Windows Server 2016 および 2019 にエージェントを手動でインストールして設定することもできます。詳細については、「EC2Launch v2 をインストールする」を参照してください。
> IMDSv2 で EC2Launch を使用するには、バージョンは 1.3.2002730 以降でなければなりません。

インストールされている EC2Launch のバージョンを確認するには、次の Windows PowerShell コマンドを使用します。

```
Test-ModuleManifest -Path "C:\ProgramData\Amazon\EC2-Windows\Launch\Module
\Ec2Launch.psd1" | Select Version
```

EC2Launch タスク

EC2Launch は最初のインスタンスの起動中に、デフォルトで以下のタスクを実行します。

- インスタンスに関する情報をレンダリングする新しい壁紙を設定します。
- コンピュータ名をインスタンスのプライベート IPv4 アドレスに設定します。
- Amazon EC2 コンソールにインスタンス情報を送信します。
- EC2 コンソールに RDP 証明書のサムプリントを送信します。
- 管理者アカウントに、ランダムなパスワードを設定します。
- DNS サフィックスを追加します。

- オペレーティングシステムパーティションを動的に拡張して、未使用の領域が含まれるようにします。
- ユーザーデータを実行します (指定された場合)。ユーザーデータを指定する方法については、ユーザーデータ入力を使用して EC2 インスタンスを起動するときにコマンドを実行するを参照してください。
- メタデータサービスと AWS KMS サーバーに到達するために永続的な静的ルートを設定します。

> **Important**
>
> このインスタンスからカスタム AMI を作成している場合、これらのルートは OS 設定の一部としてキャプチャされます。また、AMI から起動された新しいインスタンスは、サブネットの位置に関係なく、同じルートを保持します。ルートを更新するには、カスタム AMI の起動時に Server 2016 以降のメタデータ/KMS ルートを更新するを参照してください。

以下のタスクは、EC2Config サービスとの下位互換性を維持するために役立ちます。また、起動中にこれらのタスクを実行するように EC2Launch を設定することもできます。

- セカンダリ EBS ボリュームを初期化します。
- EC2 コンソールのログに Windows イベントログを送信します。
- Windows の使用準備ができている旨のメッセージを EC2 コンソールに送信します。

EC2Launch のディレクトリ構造

EC2Launch はデフォルトでは、Windows Server 2016 以降の AMI のルートディレクトリ `C:\ProgramData\Amazon\EC2-Windows\Launch` にインストールされます。

> **Note**
>
> Windows では、デフォルトで `C:\ProgramData` 以下のファイルとフォルダは非表示になります。EC2Launch のディレクトリとファイルを表示するには、Windows エクスプローラーにパスを入力するか、フォルダのプロパティを変更して、非表示のファイルとフォルダを表示する必要があります。

Launch ディレクトリには以下のサブディレクトリがあります。

- `Scripts` — EC2Launch を構成する PowerShell スクリプトが格納されます。
- `Module` — Amazon EC2 関連するスクリプトを構築するためのモジュールが格納されます。
- `Config` — カスタマイズ可能なスクリプト設定ファイルが格納されます。
- `Sysprep` — Sysprep リソースが格納されます。
- `Settings` — Sysprep グラフィカルユーザーインターフェイスのアプリケーションが格納されます。
- `Library` - EC2 起動エージェントの共有ライブラリが含まれています。
- `Log` – スクリプトのサブディレクトリと、スクリプトによって生成されたログファイルが含まれます。

テレメトリ

テレメトリは、AWSを使用して、要件の理解を深め、問題を診断し、AWSのサービスのユーザーエクスペリエンスを向上するのに役立つ追加情報です。

EC2Launch v2 バージョン `1.3.2003498` およびそれ以降のバージョンは、使用状況メトリクスやエラーなどのテレメトリを収集します。このデータは、EC2Launch が実行される Amazon EC2 インスタンスから収集されます。これには、AWSによって所有されるすべての Windows AMI が含まれます。

EC2Launch では、以下のテレメトリを収集しています。

- 使用状況の情報— エージェントのコマンド、インストール方法、スケジュールされた実行頻度。
- エラーと診断情報— エージェントのインストールと実行エラーコード。

収集されるデータの例：

```
2021/07/15 21:44:12Z: EC2LaunchTelemetry: IsAgentScheduledPerBoot=true
2021/07/15 21:44:12Z: EC2LaunchTelemetry: IsUserDataScheduledPerBoot=true
2021/07/15 21:44:12Z: EC2LaunchTelemetry: AgentCommandCode=1
2021/07/15 21:44:12Z: EC2LaunchTelemetry: AgentCommandErrorCode=5
2021/07/15 21:44:12Z: EC2LaunchTelemetry: AgentInstallCode=2
2021/07/15 21:44:12Z: EC2LaunchTelemetry: AgentInstallErrorCode=0
```

テレメトリーはデフォルトでは有効になっています。テレメトリ収集はいつでも無効にできます。テレメトリが有効な場合、EC2Launch は、追加の顧客通知なしでテレメトリデータを送信します。

テレメトリを有効または無効にするかの選択が収集されます。

テレメトリの収集は、オプトインまたはオプトアウトできます。テレメトリのオプトインまたはオプトアウトの選択は、ユーザーのテレメトリオプションを確実に遵守するために収集されます。

テレメトリー可視性

テレメトリが有効な場合、Amazon EC2 コンソールの出力に次のように表示されます。

```
2021/07/15 21:44:12Z: Telemetry: <Data>
```

インスタンスでのテレメトリの無効化

システム環境変数を設定してテレメトリを無効にするには、管理者として次のコマンドを実行します。

```
setx /M EC2LAUNCH_TELEMETRY 0
```

インストール中にテレメトリを無効にするには、次のように `install.ps1` を実行します。

```
. .\install.ps1 -EnableTelemetry:$false
```

EC2Launch のその他のトピック

- EC2Launch の最新バージョンのインストール
- Windows インスタンスで EC2Launch v1 エージェントを設定する
- EC2Launch バージョン履歴

EC2Launch の最新バージョンのインストール

インスタンスで最新バージョンの EC2Launch をダウンロードしてインストールするには、次の手順を使用します。

最新バージョンの EC2Launch をダウンロードしてインストールするには

1. インスタンスで EC2Launch をすでにインストールして設定している場合はEC2Launch 設定ファイルのバックアップを作成します。インストールプロセスではこのファイルに変更内容が保存されません。デフォルトではこのファイルは `C:\ProgramData\Amazon\EC2-Windows\Launch\Config` ディレクトリにあります。

2. EC2-Windows-Launch.zip をインスタンス上のディレクトリにダウンロードします。
3. install.ps1 を EC2-Windows-Launch.zip のダウンロード先と同じディレクトリにダウンロードします。
4. install.ps1 を実行する
5. EC2Launch 設定ファイルのバックアップを作成する場合は同ファイルを C:\ProgramData\Amazon\EC2-Windows\Launch\Config ディレクトリにコピーします。

PowerShell を使用して最新バージョンの EC2Launch をダウンロードしてインストールするには

インスタンスで EC2Launch をすでにインストールして設定している場合はEC2Launch 設定ファイルのバックアップを作成します。インストールプロセスではこのファイルに変更内容が保存されません。デフォルトではこのファイルは C:\ProgramData\Amazon\EC2-Windows\Launch\Config ディレクトリにあります。

PowerShell を使用して EC2Launch の最新バージョンをインストールするには、管理者として PowerShell ウィンドウから次のコマンドを実行します。

```
mkdir $env:USERPROFILE\Desktop\EC2Launch
$Url = "https://s3.amazonaws.com/ec2-downloads-windows/EC2Launch/latest/EC2-Windows-Launch.zip"
$DownloadZipFile = "$env:USERPROFILE\Desktop\EC2Launch\" + $(Split-Path -Path $Url -Leaf)
Invoke-WebRequest -Uri $Url -OutFile $DownloadZipFile
$Url = "https://s3.amazonaws.com/ec2-downloads-windows/EC2Launch/latest/install.ps1"
$DownloadZipFile = "$env:USERPROFILE\Desktop\EC2Launch\" + $(Split-Path -Path $Url -Leaf)
Invoke-WebRequest -Uri $Url -OutFile $DownloadZipFile
& $env:USERPROFILE\Desktop\EC2Launch\install.ps1
```

> **Note**
>
> ファイルのダウンロード時にエラーが表示され、Windows Server 2016 を使用している場合は、PowerShell ターミナルで TLS 1.2 を有効にする必要がある場合があります。次のコマンドで現在の PowerShell セッションの TLS 1.2 を有効にしてから、もう一度試してください。
>
> ```
> [Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
> ```

起動エージェントをチェックしてインストールを確認します。管理者として PowerShell ウィンドウから次のコマンドを実行します。

```
Import-Module C:\ProgramData\Amazon\EC2-Windows\Launch\Module\Ec2Launch.psm1
Import-LocalizedData -BaseDirectory C:\ProgramData\Amazon\EC2-Windows\Launch\Module\ -
FileName 'Ec2Launch.psd1' -BindingVariable moduleManifest
$moduleManifest.Get_Item('ModuleVersion')
```

Windows インスタンスで EC2Launch v1 エージェントを設定する

インスタンスが最初に初期化された後で、EC2Launch を再実行し、別の起動タスクを実行するよう EC2Launch を設定できます。

タスク

- 初期化タスクの設定
- EC2Launch が起動ごとに実行されるようスケジュールします
- ドライブの初期化とドライブ文字のマッピング
- EC2 コンソールへの Windows イベントログの送信
- 正常に起動した後の Windows の準備が完了した旨のメッセージの送信

初期化タスクの設定

次の初期化タスクを有効または無効にするには、`LaunchConfig.json` ファイルで設定を指定します。

- コンピュータ名をインスタンスのプライベート IPv4 アドレスに設定します。
- モニターを常にオンにするように設定します。
- 新しい壁紙を設定します。
- DNS サフィックス一覧を追加します。

> **Note**
>
> これにより、次のドメインの DNS サフィックス検索が追加され、他の標準サフィックスが設定されます。起動エージェントによる DNS サフィックスの設定方法の詳細については、「EC2 Windows 起動エージェントの DNS サフィックスを設定する」を参照してください。

```
region.ec2-utilities.amazonaws.com
```

- ブートボリュームサイズを拡大します。
- 管理者パスワードを設定します。

初期設定を行うには

1. 設定するインスタンスで、C:\ProgramData\Amazon\EC2-Windows\Launch\Config\LaunchConfig.json ファイルをテキストエディタで開きます。
2. 必要に応じて次の設定を更新し、変更を保存します。adminPasswordtype が Specify である場合のみ、adminPasswordにパスワードを入力してください。

```
{
 "setComputerName": false,
 "setMonitorAlwaysOn": true,
 "setWallpaper": true,
 "addDnsSuffixList": true,
 "extendBootVolumeSize": true,
 "handleUserData": true,
 "adminPasswordType": "Random | Specify | DoNothing",
 "adminPassword": "password that adheres to your security policy (optional)"
}
```

パスワードの種類は次のとおりです。

Random

EC2Launch は、ユーザーのキーを使用してパスワードを生成し、暗号化します。この設定はインスタンス起動後に無効になるため、インスタンスを再起動したり、停止して起動した場合でもパスワードは保持されます。

Specify

EC2Launch は、adminPassword で指定したパスワードを使用します。指定したパスワードがシステム要件を満たさない場合は、代わりに EC2Launch によってランダムなパスワードが生成されます。このパスワードはクリアテキストとして LaunchConfig.json に保存され、Sysprep で管理者パスワードが設定されると削除されます。EC2Launch は、ユーザーのキーを使用してパスワードを暗号化します。

`DoNothing`

EC2Launch は、`unattend.xml` ファイルで指定したパスワードを使用します。`unattend.xml` でパスワードを指定しない場合、管理者アカウントは無効になります。

3. Windows PowerShell で次のコマンドを実行し、Windows のスケジュールされたタスクとしてスクリプトの実行をスケジュールします。スクリプトは次回の起動時に 1 回のみ実行され、以降、これらのタスクは再実行されなくなります。

```
C:\ProgramData\Amazon\EC2-Windows\Launch\Scripts\InitializeInstance.ps1 -Schedule
```

EC2Launch が起動ごとに実行されるようスケジュールします

最初の起動だけでなく、起動ごとに実行されるように EC2Launch をスケジュールすることができます。

EC2Launch を起動ごとに実行するには。

1. Windows PowerShell を開き、次のコマンドを実行します。

```
C:\ProgramData\Amazon\EC2-Windows\Launch\Scripts\InitializeInstance.ps1 -
SchedulePerBoot
```

2. または、次のコマンドを使用して、実行可能ファイルを実行します。

```
C:\ProgramData\Amazon\EC2-Windows\Launch\Settings\Ec2LaunchSettings.exe
```

次に、`Run EC2Launch on every boot` を選択してください。You can specify that your EC2 インスタンスを `Shutdown without Sysprep` または `Shutdown with Sysprep` に指定できます。

> **Note**
>
> 起動のたびに EC2Launch が実行されるように設定した場合は、次回 EC2Launch が実行される際に以下のことが行われます。
>
> - 依然として `AdminPasswordType` に `Random` が設定されている場合、EC2Launch は次の起動時に新しいパスワードを生成します。起動後、`AdminPasswordType` は自動的に `DoNothing` に設定されます。これにより、以後の起動時に EC2Launch が新しいパスワー

ドを生成しないようにします。EC2Launch が初回起動時に新しいパスワードを生成しないようにするには、再起動する前に、AdminPasswordType に DoNothing を手動で設定します。

- ユーザーデータの HandleUserData が false に設定されていない限り、persist は true に設定されます。詳細については、「the section called “ユーザーデータスクリプト”」を参照してください。

ドライブの初期化とドライブ文字のマッピング

ドライブ文字を EC2 インスタンスのボリュームにマッピングするには、DriveLetterMappingConfig.json ファイルで設定を指定します。このスクリプトは、初期化およびパーティション化されていないドライブを初期化します。Windows でボリュームの詳細を取得する方法について詳しくは、Microsoft のドキュメントの「Get-Volume」をご参照ください。

ドライブ文字をボリュームにマッピングするには

1. テキストエディタで C:\ProgramData\Amazon\EC2-Windows\Launch\Config\DriveLetterMappingConfig.json ファイルを開きます。
2. 次のボリューム設定を指定し、変更を保存します。

```
{
 "driveLetterMapping": [
  {
   "volumeName": "sample volume",
   "driveLetter": "H"
  }
 ]
}
```

3. Windows PowerShell を開き、次のコマンドを使用して、ディスクを初期化する EC2Launch スクリプトを実行します。

```
C:\ProgramData\Amazon\EC2-Windows\Launch\Scripts\InitializeDisks.ps1
```

インスタンスが起動するたびにディスクを初期化するには、-Schedule フラグを次のように追加します。

```
C:\ProgramData\Amazon\EC2-Windows\Launch\Scripts\InitializeDisks.ps1 -Schedule
```

EC2 コンソールへの Windows イベントログの送信

EC2 コンソールログに Windows イベントログを送信するには、EventLogConfig.json ファイルで設定を指定します。

Windows イベントログを送信するよう設定を指定するには

1. インスタンスで、C:\ProgramData\Amazon\EC2-Windows\Launch\Config\EventLogConfig.json ファイルをテキストエディタで開きます。
2. 次のログ設定を行い、変更を保存します。

```
{
 "events": [
  {
   "logName": "System",
   "source": "An event source (optional)",
   "level": "Error | Warning | Information",
   "numEntries": 3
  }
 ]
}
```

3. Windows PowerShell で次のコマンドを実行し、Windows のスケジュールされたタスクとして、インスタンスが起動されるたびにスクリプトの実行がスケジュールされるようにします。

```
C:\ProgramData\Amazon\EC2-Windows\Launch\Scripts\SendEventLogs.ps1 -Schedule
```

ログが EC2 コンソールログに表示されるまでには、3 分以上かかる場合があります。

正常に起動した後の Windows の準備が完了した旨のメッセージの送信

EC2Config サービスは、起動するたびに、「Windows の準備が完了」した旨のメッセージを EC2 コンソールに送信しました。EC2Launch は最初の起動後に、このメッセージを送信します。EC2Config サービスとの下位互換性のため、毎回の起動後にこのメッセージを送信するよう EC2Launch をスケジュールできます。インスタンスで Windows PowerShell を開き、次のコマンドを実行します。システムは、Windows のスケジュールされたタスクとして実行するようスクリプトをスケジュールします。

```
C:\ProgramData\Amazon\EC2-Windows\Launch\Scripts\SendWindowsIsReady.ps1 -Schedule
```

EC2Launch バージョン履歴

最新バージョンの EC2Launch をダウンロードしてインストールするには、EC2Launch の最新バージョンのインストールを参照してください。

次の表は、EC2Launch のリリース済みバージョンについて説明しています。

バージョン	詳細	リリース日
1.4.299	• EC2WinUtil ドライバーバージョンを取得するためのロジックを最適化しました。 • Hyper-V ステータスクエリを削除することで、Windows is Ready が表示されるまでの時間を短縮しました。	2026 年 3 月 3 日
1.4.183	• Windows Server 2022 でサポートされていないインストールが行われないようにするためにインストーラーロジックを更新しました。 • EC2WinUtil ドライバーバージョンをインスタンスコンソールログに出力するための更新を行いました。	2026 年 2 月 4 日
1.4.6	• エージェントログメッセージングを更新しました。	2025 年 10 月 13 日
1.3.2005119	• パラメータなしで呼び出されると `Invoke-Userdata` が失敗する問題を修正しました。	2025 年 2 月 11 日
1.3.2005065	• RDP 証明書情報が適切に取得または検証されなかった問題を修正しました。必要に応じてリモートデスクトップサービスを自動的に開始する機能を追加しました。	2024 年 10 月 22 日
1.3.2005008	• デフォルトの壁紙イメージが見つからない場合は、`Set-Wallpaper` が単色の背景にフォールバックするように更新されました。	2024 年 8 月 6 日

バージョン	詳細	リリース日
1.3.2004959	• Windows Server 2025 以降でサポートされていないインストールを防止するためにインストーラーロジックを更新しました。	2024 年 7 月 2 日
1.3.2004891	• `HandleUserData` が想定どおり `false` に設定されない問題を修正しました。 • `LaunchConfig.json` に `Encrypted` パスワードオプションを追加しました。 • ユーザー指定のパスワードをデフォルトで暗号化するように `Settings` UI の動作を変更しました。 • エージェント設定ファイルの `Specify` パスワードオプションを `Encrypted` パスワードオプションに変換するために `SetAdminPasswordConfig.ps1` を追加しました。	2024 年 5 月 31 日
1.3.2004617	• 壁紙を設定する際のエラーを修正しました。	2024 年 1 月 15 日

バージョン	詳細	リリース日
1.3.2004592	• install.ps1 によって %ProgramData%\Amazon\EC2-Windows\Launch に設定されたアクセス権限が更新されました。 • EC2Launch のフォルダ/ファイルへのアクセスを、標準ユーザーアカウントに対してのみ読み取り/実行に制限しました。 • インスタンスのインスタンスメタデータサービス (IMDS) が有効化されていない場合に、IMDS が初期化されるのを待たないようにエージェントを変更しました。 • IMDS が初期化されるのを待つ際に 5 分のタイムアウトを追加しました。 • Windows is Ready メッセージの後ではなく前にインスタンスのコンソールログにテレメトリを書き込むようにエージェントを変更しました。 • いくつかの新しいインスタンスタイプに壁紙のサポートが追加されました。 EC2Launch ディレクトリのアクセス許可とユーザーアカウント許可の詳細については、「the section called “EC2Launch のディレクトリ構造”」を参照してください。	2024 年 1 月 2 日
1.3.2004491	• [管理者パスワードの指定] オプションの使用状況をモニタリングするためのテレメトリを追加しました。	2023 年 11 月 9 日
1.3.2004462	• シリアルコンソールへの毎回の書き込み後にフラッシュが追加されました。	2023 年 10 月 18 日

バージョン	詳細	リリース日
1.3.2004438	• レジストリエントリに基づいてドメイン名の権限委譲を制限: HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Dnscache\Parameters\DomainNameDevolutionLevel 。 • UserdataExecution.log の許可は Administrators のみに制限されています。 • ログの初期化に失敗したときの Windows イベントログにエラーメッセージを追加しました。	2023 年 10 月 4 日
1.3.2004256	• EnableSCSIPersistentReservations コンソールログに値を追加しました。 • の再試行機能が追加されました。Get-ConsolePort	2023 年 7 月 7 日
1.3.2004052	• インスタンスの起動時に SSH キーが指定されていない場合のエラーを修正しました。 • 障害発生時に AmazonSSMAgent Windows サービスの開始が再試行されるように更新されました。 • BeforeSysprep.cmd がゼロ以外の終了コードで失敗した場合に SysprepInstance.ps1 が失敗するように更新されました。	2023 年 3 月 8 日
1.3.2003975	• SysprepInstance.ps1 が 1 の $LastErrorCode を返す Packer AMI ビルドに影響する問題を修正しました。	2022 年 12 月 24 日

バージョン	詳細	リリース日
1.3.2003961	• 高速起動されたインスタンスで、明示的に指定された管理者パスワードがランダムなパスワードで上書きされる問題を修正しました。 • SSM エージェントが小さいインスタンスタイプで起動できない問題を修正しました。 • インスタンスコンソールログに、有効な RDP 証明書のサムプリント値の代わりに `RDPCERTIFICATE-THUMBPRINT: 0000000000000000000000000` が含まれる問題を修正しました。	2022 年 12 月 6 日
1.3.2003923	• PnPDeviceID が空の場合にネットワークアダプタを検索するロジックが修正されました。	2022 年 11 月 9 日
1.3.2003919	• PCI セグメント情報を使用するように Get-ConsolePort を更新しました。 • 再起動後に間違ったネットワークアダプターを選択できる問題を修正します。 • Start-SSM-Agent のタイムアウトロジックを修正しました。 • Send-AdminCredentials 関数エイリアスの下位互換性が修正されました。	2022 年 11 月 8 日
1.3.2003857	• プライマリネットワークアダプタが選択された場合に、デフォルトゲートウェイのあるアダプタを優先します。 • インメモリのパスワード暗号化を拡張しました。	2022 年 10 月 3 日

バージョン	詳細	リリース日
1.3.2003824	• `setComputerName` 中のエラーを修正しました。 • BYOL 請求コードが検出された場合に Windows のアクティベーションをスキップするロジックを追加しました。 • メモリ内パスワードの暗号化を追加しました。 • `m6id.4xlarge` でのボリューム初期化中のエラーを修正しました。	2022 年 8 月 30 日
1.3.2003691	• IMDSv2 リクエストのみを行うように IMDS 待機ロジックを更新しました。 • eGPU のインストールに影響するバグを修正しました。	2022 年 6 月 21 日
1.3.2003639	• 初期化前の使用を防ぐため、ネットワークアダプタの待機ロジックを追加しました。 • 小さな問題を修正しました。	2022 年 5 月 10 日
1.3.2003498	• テレメトリが追加されました。 • 設定 UI へのショートカットを追加しました。 • PowerShell スクリプトをフォーマットしました。 • BeforeSysprep.cmd が完了する前にシャットダウンが発生する問題を修正しました。	2022 年 1 月 31 日
1.3.2003411	パスワード生成ロジックを変更して、複雑さの低いパスワードを除外するようにしました。	2021 年 8 月 4 日
1.3.2003364	IMDSv2 サポート付の Install-EgpuManager を更新。	2021 年 6 月 7 日

バージョン	詳細	リリース日
1.3.2003312	• `setMonitorAlwaysOn` 設定の前後にログ行を追加。 • コンソールログに AWS Nitro Enclaves パッケージバージョンを追加。	2021 年 5 月 4 日
1.3.2003284	ユーザデータを保存する場所を `LocalAppData` に更新することで、アクセス許可モデルを改善しました。	2021 年 3 月 23 日
1.3.2003236	• `Set-AdminAccount` および `Randomize-LocalAdminPassword` でのユーザーパスワードの設定方法を更新しました。 • 書き込み可能に設定する前に、ディスクが読み取り専用に設定されているかどうかをチェックする `InitializeDisks` を修正しました。	2021 年 2 月 11 日
1.3.2003210	`install.ps1` のローカライゼーションを修正しました。	2021 年 1 月 7 日
1.3.2003205	`%ProgramData%AmazonEC2-Windows\Launch\ModuleScripts` ディレクトリのアクセス許可を更新するための `install.ps1` に対するセキュリティ修正。	2020 年 12 月 28 日
1.3.2003189	ルートを追加した後に `w32tm resync` が追加されました。	2020 年 12 月 4 日
1.3.2003155	インスタンスタイプ情報を更新しました。	2020 年 8 月 25 日
1.3.2003150	コンソール出力に `OsCurrentBuild` と `OsReleaseId` を追加しました。	2020 年 4 月 22 日
1.3.2003040	IMDS バージョン 1 のフォールバックロジックを修正しました。	2020 年 4 月 7 日

バージョン	詳細	リリース日
1.3.2002730	IMDS V2 のサポートを追加しました。	2020 年 3 月 3 日
1.3.2002240	小さな問題を修正しました。	2019 年 10 月 31 日
1.3.2001660	Sysprep の初回実行後にパスワードなしのユーザーの自動ログイン問題を修正しました。	2019 年 7 月 2 日
1.3.2001360	小さな問題を修正しました。	2019 年 3 月 27 日
1.3.2001220	すべての PowerShell スクリプトが署名されています。	2019 年 2 月 28 日
1.3.2001200	Windows Server フェイルオーバークラスター内のノードでスクリプトを実行すると、ドライブ文字がローカルドライブ文字と一致するリモートノード上のドライブがフォーマットされるという InitializeDisks.ps1 の問題を修正しました。	2019 年 2 月 27 日
1.3.2001160	Windows 2019 の壁紙が表示されない問題を修正しました。	2019 年 2 月 22 日
1.3.2001040	• ACPI の問題を解決するためにモニターをオフにしないように設定するためのプラグインを追加しました。 • SQL Server のエディションとバージョンがコンソールに書き込まれます。	2019 年 1 月 21 日
1.3.2000930	ipv6 有効 ENI でメタデータへのルートの追加を修正しました。	2019 年 1 月 2 日
1.3.2000760	• ENA デバイス用の RSS および受信キュー設定のデフォルト設定を追加しました。 • Sysprep 中の休止状態を無効化しました。	2018 年 12 月 5 日

バージョン	詳細	リリース日
1.3.2000630	• DNS サーバーに ルート 169.254.169.253/32 を追加しました。 • 管理者ユーザー設定のフィルターを追加しました。 • インスタンスを休止状態にするための改善。 • 起動ごとに EC2Launch を実行するようにスケジュールオプションが追加されました。	2018 年 11 月 9 日
1.3.2000430.0	• AMZN タイムサービスにルート 169.254.169.123/32 を追加 • GRID ライセンスにルート 169.254.169.249/32 を追加 • Systems Manager を開始しようとするときのタイムアウト 25 秒を追加。	2018 年 9 月 19 日
1.3.200039.0	• EBS NVMe ボリュームが一時的なものとしてマークされる問題を修正。 • NVME ドライバーのバージョンの追加ログを追加。	2018 年 8 月 15 日
1.3.2000080	小さな問題を修正しました。	
1.3.610	出力およびエラーがユーザーデータからファイルにリダイレクトされる問題を修正しました。	
1.3.590	• 表示されないインスタンスタイプを壁紙に追加しました。 • ドライブ文字のマッピングとディスクの設置に関する問題を修正しました。	

バージョン	詳細	リリース日
1.3.580	• ウェブのリクエストに対してデフォルトのシステムプロキシ設定を使用するように `Get-Metadata` を修正しました。 • ディスクの初期化に NVMe の特殊なケースを追加しました。 • 小さな問題を修正しました。	
1.3.550	シャットダウンなしで Sysprep を有効化する `-NoShutdown` オプションを追加しました。	
1.3.540	小さな問題を修正しました。	
1.3.530	小さな問題を修正しました。	
1.3.521	小さな問題を修正しました。	
1.3.0	• コンピュータ名の変更に伴う 16 進数の長さの問題を修正しました。 • コンピュータ名の変更に伴って起こる場合がある再起動ループを修正しました。 • 壁紙設定の問題を修正しました。	
1.2.0	• インストールされているオペレーティングシステム (OS) に関する情報を EC2 システムログに情報を表示するための更新。 • EC2Launch および SSM Agent のバージョンを EC2 システムログに表示するための更新。 • 小さな問題を修正しました。	

バージョン	詳細	リリース日
1.1.2	• ENA ドライバー情報を EC2 システムログに表示するための更新。 • プライマリ NIC フィルタロジックから Hyper-V を除外するための更新。 • KMS のアクティブ化のレジストリキー内に AWS KMS サーバーおよびポートを追加しました。 • 複数のユーザー用に壁紙設定を改善しました。 • 永続的なストアからのルートをクリアするための更新。 • DNS サフィックスリストのアベイラビリティーゾーンから z を削除するための更新。 • ユーザーデータの <runAsLocalSystem> タグの問題に対処するための更新。	
1.1.1	初回リリース。	

EC2Config サービスを使用して、EC2 レガシー Windows オペレーティングシステムインスタンスの起動時にタスクを実行する

> **Note**
>
> EC2Config のサポートが終了しました。実行されているオペレーティングシステムのバージョンは、Microsoft のサポートが終了しました。最新の起動エージェントにアップグレードすることを強くお勧めします。
> Windows Server 2022 以降の OS バージョンの最新起動エージェントは EC2Launch v2 であり、EC2Config および EC2Launch の両方を置き換えて、AWS Windows Server 2022 および 2025 AMI にプリインストールされた状態になります。Windows Server 2016 および 2019 にエージェントを手動でインストールして設定することもできます。詳細については、「EC2Launch v2 をインストールする」を参照してください。

Windows Server 2016 より前の Windows AMI for Windows Server バージョンには、EC2Config サービス EC2Config.exeというオプションのサービスが含まれています。EC2Config は、インスタンスが起動し、起動時にタスクを実行したとき、およびインスタンスを停止または開始するたびに起動します。オンデマンドでタスクを実行させることもできます。タスクには自動的に有効化されるものもありますが、手動で有効化しなければならないものもあります。使用は任意ですが、このサービスは他の手段では利用できない高度な機能を提供します。このサービスは LocalSystem アカウントで実行されます。

EC2Config サービスは Sysprep という Microsoft ツールを実行します。このツールを利用すると、再利用可能なカスタマイズされた Windows AMI を作成できます。EC2Config は、Sysprep を呼び出す際、%ProgramFiles%\Amazon\EC2ConfigService\Settings のファイルを使用して、実行する操作を決定します。これらのファイルを編集するには、[EC2 サービスプロパティ] システムダイアログで間接的に行うか、XML エディタまたはテキストエディタで直接行います。ただし一部の高度な設定は [EC2 サービスプロパティ] システムダイアログで利用できないため、それらのエントリを直接編集する必要があります。

インスタンスの設定を更新した後で、そのインスタンスから AMI を作成した場合、その AMI から起動されるすべてのインスタンスには、更新後の新しい設定が適用されます。AMI の作成の詳細については、Amazon EBS-backed AMI を作成するを参照してください。

EC2Config は設定ファイル群を使って操作を制御します。設定ファイル群に変更を加えるには、GUI ツールを使うか、XML ファイルを直接編集します。サービスのバイナリおよびその他のファイルは、%ProgramFiles%\Amazon\EC2ConfigService ディレクトリに格納されています。

内容

- EC2Config および AWS Systems Manager
- EC2Config タスク
- EC2Config の設定ファイル
- EC2Config の最新バージョンのインストール
- EC2Config サービスの .NET プロキシ設定を構成する
- EC2 Windows インスタンスのシステムダイアログから EC2Config サービスプロパティを設定する
- EC2Config 起動エージェントに関する問題のトラブルシューティング
- EC2Config バージョン履歴

EC2Config および AWS Systems Manager

EC2Config サービスは、2016 年 11 月以前に発行された Windows Server 2016 より前のバージョンの Windows Server で AMI から作成されたインスタンスに対する Systems Manager リクエストを処理します。

2016 年 11 月以降に発行された Windows Server 2016 より前のバージョンの Windows Server で AMI から作成されたインスタンスには、EC2Config サービスおよび SSM Agent が含まれます。EC2Config は前述のすべてのタスクを実行し、SSM Agent は Run Command やステートマネージャーなどの Systems Manager 機能に対するリクエストを処理します。

Run Command を使用して既存のインスタンスをアップグレードし、最新バージョンの EC2Config サービスおよび SSM Agent を使用できます。詳細については、AWS Systems Manager ユーザーガイドのRun Command を使用した SSM エージェントの更新を参照してください。

EC2Config タスク

EC2Config は、インスタンスの初回起動時に複数の初期起動タスクを実行し、その後、それらを無効にします。これらのタスクを再び実行するには、ユーザーが明示的に有効化した後でインスタンスをシャットダウンするか、手動で Sysprep を実行する必要があります。初回起動時のタスクには以下のものがあります。

- 管理者アカウントに、ランダムに生成した暗号化パスワードを設定する
- リモートデスクトップに使用されるホスト証明書を生成しインストールする
- オペレーティングシステムパーティションを動的に拡張して、未使用の領域が含まれるようにします。
- 指定されたユーザーデータ (および、インストールされていれば Cloud-Init) を実行します。ユーザーデータを指定する方法については、ユーザーデータ入力を使用して EC2 インスタンスを起動するときにコマンドを実行するを参照してください。

EC2Config は、インスタンスが起動するたびに次のタスクを実行します。

- 16 進数表記のプライベート IP アドレスと一致するようにコンピュータのホスト名を変更する (このタスクはデフォルトでは無効になっているので、このタスクを有効にしてインスタンスの起動時に実行する必要があります)。
- key management server AWS KMSを設定し、Windows アクティベーションのステータスを確認して、必要に応じて Windows のアクティベーションを行う。

- すべての Amazon EBS ボリュームおよびインスタンスストアボリュームをマウントし、ボリューム名をドライブ文字にマップします。
- イベントログエントリをコンソールに出力し、トラブルシューティングに役立てる (このタスクはデフォルトでは無効になっているので、このタスクを有効にしてインスタンスの起動時に実行する必要があります)。
- コンソールに Windows の準備が完了した旨の通知を出力する
- 1 つ、または複数の NIC がアタッチされているとき、プライマリネットワークアダプターにカスタムルートを追加して、IP アドレス `169.254.169.250`、`169.254.169.251`、および `169.254.169.254` を有効にします。これらのアドレスは Windows ライセンス認証が使用し、またユーザーがインスタンスのメタデータにアクセスする際にも使用します。

> **Note**
>
> Windows OS が IPv4 を使用するように設定されている場合は、これらの IPv4 リンクローカルアドレスを使用できます。Windows OS が IPv4 ネットワークプロトコルスタックを無効にし、代わりに IPv6 を使用する場合は、`[fd00:ec2::250]` と `169.254.169.250` の代わりに `169.254.169.251` を追加します。次に、`[fd00:ec2::254]` の代わりに `169.254.169.254` を追加します。

EC2Config は、ユーザーがログインするたびに以下のタスクを実行します。

- デスクトップ背景に壁紙情報を表示する

インスタンスの実行中、ユーザーは EC2Config にリクエストを送信して以下のタスクをオンデマンドで実行させることができます。

- Sysprep を実行し、インスタンスをシャットダウンして、ユーザーがそこから AMI を作成できるようにする 詳細については、「Windows Sysprep を使用して Amazon EC2 AMI を作成する」を参照してください。

EC2Config の設定ファイル

設定ファイルは、EC2Config サービスの操作をコントロールします。これらのファイルは、`C:\Program Files\Amazon\Ec2ConfigService\Settings` ディレクトリにあります。

- `ActivationSettings.xml` - キーマネージメントサーバー AWS KMSを使用した製品のアクティブ化を制御します。
- `AWS.EC2.Windows.CloudWatch.json` — CloudWatch に送信するパフォーマンスカウンタと CloudWatch Logs に送信するログを制御します。
- `BundleConfig.xml` — EC2Config が AMI 作成のために instance store-backed インスタンスを準備する方法を制御します。
- `Config.xml` — 主要な設定を制御します。
- `DriveLetterConfig.xml` — ドライブ文字のマッピングを制御します。
- `EventLogConfig.xml` — インスタンスの起動中、コンソールに表示されるイベントログ情報を制御します。
- `WallpaperSettings.xml` — デスクトップの背景に表示される情報を制御します。

ActivationSettings.xml

このファイルは製品のアクティベーションを制御する設定を含みます。Windows の起動時に、EC2Config サービスは Windows がライセンス認証済みかどうかを確認します。Windows が既にアクティブ化されている場合、指定された AWS KMS サーバーを探すことで Windows のアクティブ化を試みます。

- `SetAutodiscover` – AWS KMS を自動検出するかどうかを示します。
- `TargetKMSServer` - AWS KMS のプライベート IP アドレスを保存します。AWS KMS はユーザーのインスタンスと同じリージョンに存在する必要があります。
- `DiscoverFromZone` — 指定された DNS ゾーンで AWS KMS サーバーを探します。
- `ReadFromUserData` — UserData から AWS KMS サーバーを取得します。
- `LegacySearchZones` — 指定された DNS ゾーンで AWS KMS サーバーを探します。
- `DoActivate` — このセクションで指定された設定を用いてアクティベーションを試みます。この値は `true` または `false` となります。
- `LogResultToConsole` — 結果をコンソールに出力します。

BundleConfig.xml

このファイルは、EC2Config が AMI 作成のインスタンスを準備する方法を制御する設定を含みます。

- AutoSysprep — Sysprep を自動で使用するかどうかを示します。Sysprep を使用には、値を Yes に変更します。
- SetRDPCertificate — リモートデスクトップサーバーに自己署名証明書を設定します。これにより、RDP で安全にインスタンスに接続できます。新しいインスタンスに証明書が必要な場合は、値を Yes に変更します。

  この設定は、Windows Server 2016 より前のバージョンのオペレーティングシステムのインスタンスでは使用されません。これらのオペレーティングシステムは独自の証明書を生成できるためです。
- SetPasswordAfterSysprep — 新しく起動したインスタンスにランダムなパスワードを設定し、ユーザー起動キーで暗号化し、暗号化されたパスワードをコンソールに出力します。新しいインスタンスに暗号化されたランダムなパスワードを自動的に設定しない場合は、この設定の値を No に変更します。

Config.xml

プラグイン

- Ec2SetPassword — 暗号化されたランダムなパスワードを、インスタンスを起動するたびに新しく生成します。この機能は、最初の起動以後デフォルトで無効化されますので、同じインスタンスを再起動してもユーザーが設定したパスワードが変更されることはありません。引き続きインスタンスを起動するたびにパスワードを生成するには、この設定を Enabled に変更します。

  インスタンスから AMI を作成する予定がある場合、この設定は重要になります。
- Ec2SetComputerName — インスタンスのホスト名を、インスタンスの IP アドレスに基づく一意の名前に設定した後、インスタンスを再起動します。独自のホスト名を設定するには、また既存のホスト名が変更されないようにするには、この設定を無効にする必要があります。
- Ec2InitializeDrives — 起動時にすべてのボリュームの初期化とフォーマットを行います。この機能は、デフォルトでご利用になれます。
- Ec2EventLog — コンソールにイベントログのエントリを表示します。デフォルトでは、System イベントログから 3 つの最新エラーエントリが表示されます。表示するイベントログのエントリを指定するには、EventLogConfig.xml ディレクトリにある EC2ConfigService\Settings ファイルを編集します。このファイル内の設定について詳しくは、MSDN ライブラリの Eventlog Key を参照してください。
- Ec2ConfigureRDP — ユーザーがリモートデスクトップを使ってインスタンスに安全にアクセスできるよう、自己署名証明書を設定します。この設定は、Windows Server 2016 より前のバー

ジョンのオペレーティングシステムのインスタンスでは使用されません。これらのオペレーティングシステムは独自の証明書を生成できるためです。

- Ec2OutputRDPCert — ユーザーがサムプリントと照合できるよう、リモートデスクトップの証明書情報をコンソールに表示します。
- Ec2SetDriveLetter — ユーザーが定義した設定に基づき、ドライブ文字をマウントされたボリュームに割り当てます。デフォルトでは、Amazon EBS ボリュームがインスタンスにアタッチされている場合、ドライブ文字を使ってそのインスタンスにマウントできます。ドライブ文字のマッピングを指定するには、DriveLetterConfig.xml ディレクトリにある EC2ConfigService\Settings ファイルを編集します。
- Ec2WindowsActivate — プラグインは Windows のライセンス認証を処理します。このプラグインは Windows がライセンス認証されたかどうかをチェックします。アクティブ化されていない場合は、AWS KMS クライアントの設定を更新し、Windows をアクティブ化します。

  AWS KMS 設定を変更するには、EC2ConfigService\Settings ディレクトリにある ActivationSettings.xml ファイルを編集します。
- Ec2DynamicBootVolumeSize — ディスク 0/ボリューム 0 が未使用の領域を含むように拡張します。
- Ec2HandleUserData— Sysprep が実行された後初めてインスタンスが起動するときに、ユーザーが作成したスクリプトを作成し実行します。script タグでラップされたコマンドはバッチファイルに保存され、PowerShell タグでラップされたコマンドは .ps1 ファイルに保存されます ([EC2 サービスプロパティ] システムダイアログの [ユーザーデータ] チェックボックスに対応)。
- Ec2ElasticGpuSetup — インスタンスが Elastic GPU に関連付けられている場合は、Elastic GPU ソフトウェアパッケージをインストールします。
- Ec2FeatureLogging — Windows の機能のインストールとそのサービスの状態をコンソールに送信します。Microsoft Hyper-V 機能、およびその vmms サービスでのみサポートされます。

グローバル設定

- ManageShutdown — Sysprep の実行中に、Amazon S3-backed AMI から起動したインスタンスが終了しないようにします。
- SetDnsSuffixList — ネットワークアダプタの DNS サフィックスを Amazon EC2 に設定します。これにより、完全修飾ドメイン名がなくても、Amazon EC2 で実行中のサーバーの DNS 解決が可能になります。

> **Note**
>
> これにより、次のドメインの DNS サフィックス検索が追加され、他の標準サフィックスが設定されます。起動エージェントによる DNS サフィックスの設定方法の詳細については、「EC2 Windows 起動エージェントの DNS サフィックスを設定する」を参照してください。
>
> ```
> region.ec2-utilities.amazonaws.com
> ```

- `WaitForMetaDataAvailable` — メタデータにアクセスできるようになり、ネットワークが利用可能になるまで、EC2Config サービスが起動処理を続行しないようにします。これにより、EC2Config はアクティベーションのメタデータや他のプラグインから情報を取得できるようになります。
- `ShouldAddRoutes` — 複数の NIC がアタッチされているとき、プライマリネットワークアダプタにカスタムルートを追加して、IP アドレス 169.254.169.250、169.254.169.251、および 169.254.169.254 を有効にする。これらのアドレスは Windows ライセンス認証が使用し、またユーザーがインスタンスのメタデータにアクセスする際にも使用します。
- `RemoveCredentialsfromSyspreponStartup` — 次回のサービスの開始時に `Sysprep.xml` から管理者パスワードを削除します。パスワードを保存しておくためには、この設定を編集します。

DriveLetterConfig.xml

このファイルは、ドライブ文字のマッピングを制御する設定を含みます。デフォルトでは、ボリュームには使用可能な任意のドライブ文字がマッピングされる可能性があります。次のようにして、ボリュームに特定のドライブ文字をマウントできます。

```
<?xml version="1.0" standalone="yes"?>
<DriveLetterMapping>
  <Mapping>
    <VolumeName></VolumeName>
    <DriveLetter></DriveLetter>
  </Mapping>
  . . .
  <Mapping>
    <VolumeName></VolumeName>
    <DriveLetter></DriveLetter>
  </Mapping>
```

```
</DriveLetterMapping>
```

- VolumeName — ボリュームラベル。例えば、*My Volume* と指定します。インスタンスストレージボリュームのマッピングを指定するには、Temporary Storage X というラベルを使用します。このとき、X は 0 ~ 25 の数字です。
- DriveLetter — ドライブ文字。例えば、*M:* と指定します。ドライブ文字がすでに使用されている場合はマッピングが失敗します。

EventLogConfig.xml

このファイルは、インスタンスの起動中、コンソールに表示されるイベントログ情報を制御する設定を含みます。デフォルトでは、System イベントログから 3 つの最新エラーエントリが表示されます。

- Category — 監視するイベントログキー。
- ErrorType — イベントタイプ (Error、Warning、Information など)。
- NumEntries — このカテゴリに格納されるイベントの数。
- LastMessageTime — 同じメッセージが何度もプッシュされることを防ぐため、サービスがメッセージをプッシュするたびにこの値が更新されます。
- AppName — イベントログを記録したイベントソースまたはアプリケーション。

WallpaperSettings.xml

このファイルは、デスクトップの背景に表示される情報を制御する設定を含みます。デフォルトでは、次の情報が表示されます。

- Hostname — コンピュータ名が表示されます。
- Instance ID — インスタンスの ID を表示します。
- Public IP Address — インスタンスのパブリック IP アドレスを表示します。
- Private IP Address — インスタンスのプライベート IP アドレスを表示します。
- Availability Zone — インスタンスが実行しているアベイラビリティーゾーンを表示します。
- Instance Size — インスタンスのタイプを表示します。
- Architecture — PROCESSOR_ARCHITECTURE 環境変数の設定を表示します。

エントリを削除すると、デフォルトで表示された任意の情報を削除できます。次のようにして、表示する追加インスタンスメタデータを指定できます。

```
<WallpaperInformation>
  <name>display_name</name>
  <source>metadata</source>
  <identifier>meta-data/path</identifier>
</WallpaperInformation>
```

次のようにして、表示する追加システム環境変数を指定できます。

```
<WallpaperInformation>
  <name>display_name</name>
  <source>EnvironmentVariable</source>
  <identifier>variable-name</identifier>
</WallpaperInformation>
```

InitializeDrivesSettings.xml

このファイルには、EC2Config によるドライブの初期化方法を制御する設定が含まれています。

デフォルトで、EC2Config はオペレーティングシステムと共にオンライン状態にならなかったドライブを初期化します。プラグインは次のようにカスタマイズできます。

```
<InitializeDrivesSettings>
    <SettingsGroup>setting</SettingsGroup>
</InitializeDrivesSettings>
```

ドライブの初期化方法を指定するには、設定グループを使用してください。

FormatWithTRIM

ドライブのフォーマット中に TRIM コマンドが有効になります。ドライブのフォーマットと初期化が完了した後、システムは TRIM 設定を復元します。

EC2Config バージョン 3.18 以降、TRIM コマンドはデフォルトでディスクフォーマットの操作中に無効になります。これにより、フォーマット時間が短縮されます。EC2Config バージョン 3.18 以降のディスクフォーマット操作中に TRIM を有効にするには、この設定を使用します。

FormatWithoutTRIM

ドライブのフォーマット時に TRIMを コマンドを無効にして、Windows でのフォーマット時間を短縮します。ドライブのフォーマットと初期化が完了した後、システムは TRIM 設定を復元します。

DisableInitializeDrives

新しいドライブのフォーマットを無効にします。ドライブを手動で初期化するには、この設定を使用します。

EC2Config の最新バージョンのインストール

> **Note**
>
> Windows Server 2022 以降の OS バージョンの最新の起動エージェントは EC2Launch v2 であり、EC2Config および EC2Launch の両方を置き換えます。EC2Launch v2 は、AWS Windows Server 2022 および 2025 AMI にプリインストールされています。Windows Server 2016 および 2019 にエージェントを手動でインストールして設定することもできます。詳細については、「EC2Launch v2 をインストールする」を参照してください。

EC2Config 更新の通知を受け取る方法については、EC2 Windows 起動エージェント通知をサブスクライブするを参照してください。各バージョンの変更については、EC2Config バージョン履歴を参照してください。

開始する前に

- .NET Framework 3.5 SP1以上を使用していることを確認します。
- デフォルトでは、セットアップによってインストール時に設定ファイルがデフォルト設定ファイルに置き換えられ、インストールが完了すると EC2Config サービスが再開されます。EC2Config サービス設定を変更した場合は、`config.xml` ディレクトリの `%Program Files%\Amazon\Ec2ConfigService\Settings` ファイルをコピーします。EC2Config サービスを更新したら、このファイルを復元して設定の変更を維持することができます。

EC2Config のバージョンの確認

インストールされている EC2Config をインスタンスで確認するには、次の手順を使用します。

インストールされている EC2Config のバージョンを確認するには

1. AMI からインスタンスを起動して接続します。
2. コントロールパネルから [Program and Features] を選択してください。
3. インストールされたプログラムのリストで `Ec2ConfigService` を探します。バージョン番号は [Version] 列に表示されています。

EC2Config の更新

インスタンスで最新バージョンの EC2Config をダウンロードしてインストールするには、次の手順を使用します。

EC2Config の最新バージョンをダウンロードしてインストールするには

1. EC2Config インストーラをダウンロードして解凍します。
2. `EC2Install.exe` を実行します。オプションの完全なリストについては`EC2Install` オプションを指定して `/?` を実行してください。デフォルトではセットアップによってプロンプトが表示されます。プロンプトを表示せずにコマンドを実行するには、`/quiet` オプションを使用します。

   > **Important**
   >
   > 保存した `config.xml` ファイルのカスタム設定を保持するには、`EC2Install` オプションを指定して `/norestart` を実行し、設定を復元した後、手動で EC2Config サービスを再開します。

3. EC2Config バージョン 4.0 以降を実行している場合は、Microsoft サービススナップインからインスタンスの SSM Agent を再起動する必要があります。

   > **Note**
   >
   > 更新された EC2Config バージョン情報は、インスタンスを再起動または停止して開始するまで、インスタンスのシステムログまたは Trusted Advisor チェックに表示されません。

PowerShell を使用して最新バージョンの EC2Config のダウンロードとインストールを行うには

PowerShell を使用して EC2Config の最新バージョンをダウンロード、解凍、インストールするには、PowerShell ウィンドウから次のコマンドを実行します。

```
$Url = "https://s3.amazonaws.com/ec2-downloads-windows/EC2Config/EC2Install.zip"
$DownloadZipFile = "$env:USERPROFILE\Desktop\" + $(Split-Path -Path $Url -Leaf)
$ExtractPath = "$env:USERPROFILE\Desktop\"
Invoke-WebRequest -Uri $Url -OutFile $DownloadZipFile
$ExtractShell = New-Object -ComObject Shell.Application
$ExtractFiles = $ExtractShell.Namespace($DownloadZipFile).Items()
$ExtractShell.NameSpace($ExtractPath).CopyHere($ExtractFiles)
Start-Process $ExtractPath
Start-Process `
    -FilePath $env:USERPROFILE\Desktop\EC2Install.exe `
    -ArgumentList "/S"
```

> **Note**
>
> ファイルのダウンロード時にエラーが表示され、Windows Server 2016 以前のバージョンを使用している場合は、PowerShell ターミナルで TLS 1.2 を有効にする必要がある場合があります。次のコマンドで現在の PowerShell セッションの TLS 1.2 を有効にしてから、もう一度試してください。
>
> ```
> [Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
> ```

`Ec2ConfigService` ディレクトリの `C:\Program Files\Amazon\` をチェックしてインストールを確認します。

EC2Config サービスの .NET プロキシ設定を構成する

EC2Config サービス は、AWS SDK for .NET、`system.net` 要素、または Microsoft グループポリシーと Internet Explorer のいずれかを使用してプロキシを介して通信するように設定できます。サインイン認証情報を指定できるという点で、.NET 対応 AWS SDK を使用する方法が推奨されます。

方法

- AWS SDK for .NET(優先) を使用したプロキシ設定の構成
- system.net エレメントを使用したプロキシ設定の構成
- Microsoft Group Policy と Microsoft Internet Explorer を使用したプロキシ設定の構成

AWS SDK for .NET(優先) を使用したプロキシ設定の構成

ファイルに proxy エレメントを指定することで、EC2Config サービスのプロキシ設定を構成できます。Ec2Config.exe.config詳細については、AWS SDK for .NET の設定ファイルリファレンスをご参照ください。

Ec2Config.exe.config の proxy エレメントを指定するには

1. プロキシを介して通信するように EC2Config サービスを構成するインスタンスの Ec2Config.exe.config ファイルを編集します。デフォルトでは、このファイルは %ProgramFiles%\Amazon\Ec2ConfigService ディレクトリにあります。
2. 次の aws エレメントを configSections に追加します。これを既存の sectionGroups には追加しないでください。

   EC2Config バージョン 3.17 以前の場合

   ```
   <configSections>
      <section name="aws" type="Amazon.AWSSection, AWSSDK"/>
   </configSections>
   ```

   EC2Config バージョン 3.18 以降の場合

   ```
   <configSections>
        <section name="aws" type="Amazon.AWSSection, AWSSDK.Core"/>
   </configSections>
   ```

3. 次の aws エレメントを Ec2Config.exe.config ファイルに追加します。

   ```
   <aws>
      <proxy
        host="string value"
        port="string value"
        username="string value"
        password="string value" />
   </aws>
   ```

4. 変更を保存します。

system.net エレメントを使用したプロキシ設定の構成

ファイルの system.net エレメントにプロキシ設定を指定できます。Ec2Config.exe.config詳細については、MSDN のdefaultProxy 要素 (ネットワーク設定)を参照してください。

Ec2Config.exe.config の system.net エレメントを指定するには

1. プロキシを介して通信するように EC2Config サービスを構成するインスタンスの Ec2Config.exe.config ファイルを編集します。デフォルトでは、このファイルは %ProgramFiles%\Amazon\Ec2ConfigService ディレクトリにあります。
2. defaultProxy エントリを system.net に追加します。詳細については、MSDN のdefaultProxy 要素 (ネットワーク設定)を参照してください。

   例えば、次の設定では、プロキシを迂回するメタデータとライセンストラフィックを除くすべてのトラフィックが、現在 Internet Explorer 用に設定されているプロキシを使用するようにルーティングされています。

   ```
   <defaultProxy>
       <proxy usesystemdefault="true" />
       <bypasslist>
           <add address="169.254.169.250" />
           <add address="169.254.169.251" />
           <add address="169.254.169.254" />
           <add address="[fd00:ec2::250]" />
           <add address="[fd00:ec2::254]" />
       </bypasslist>
   </defaultProxy>
   ```
3. 変更を保存します。

Microsoft Group Policy と Microsoft Internet Explorer を使用したプロキシ設定の構成

EC2Config サービスは、ローカルシステムのユーザーアカウントで動作します。インスタンスのグループポリシー設定を変更した後で、このアカウントのインスタンス全体のプロキシ設定を Internet Explorer で指定できます。

グループポリシーと Internet Explorer を使用してプロキシ設定を構成するには

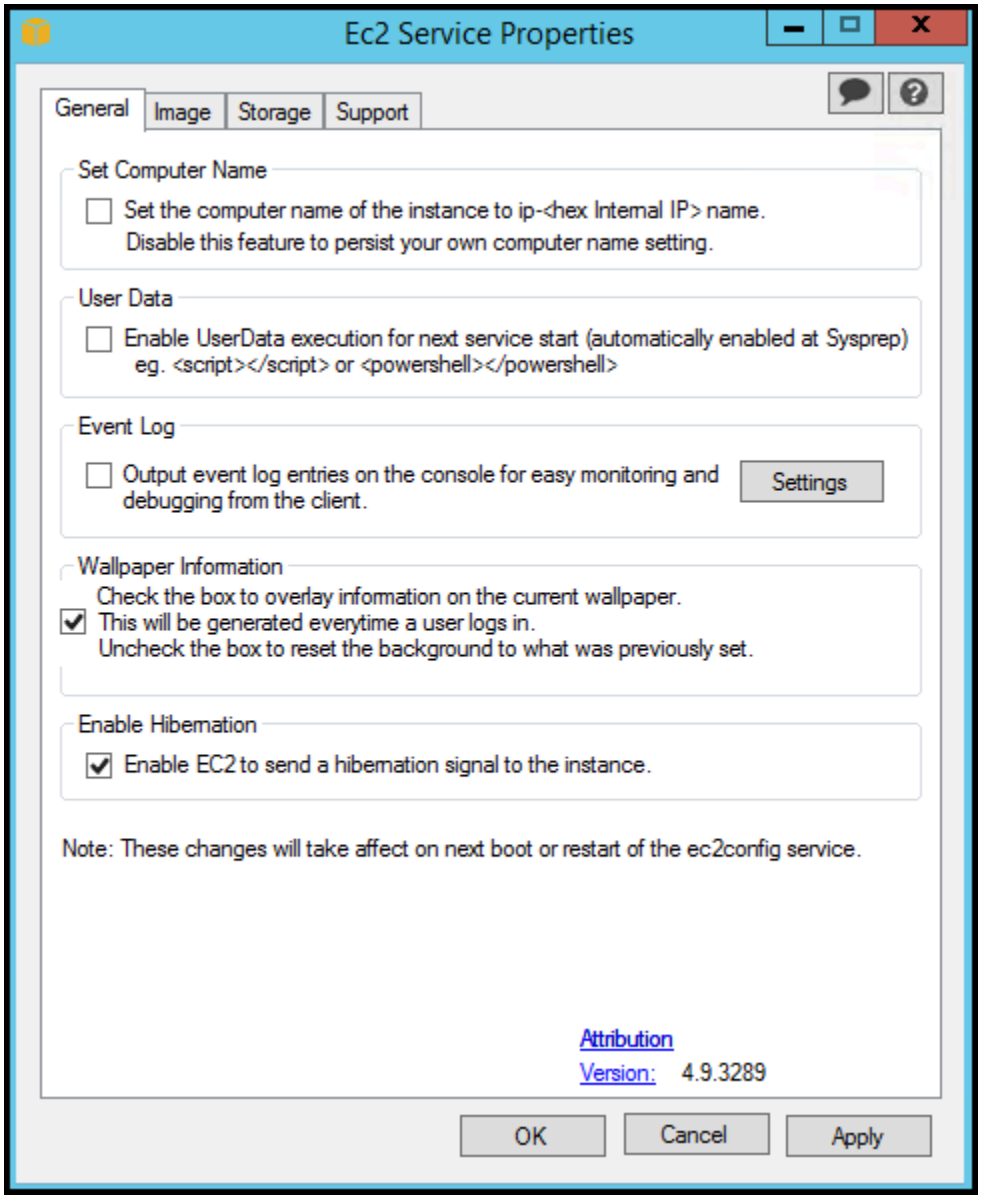
1. プロキシを介して通信するように EC2Config サービスを構成するインスタンスで、管理者としてコマンドプロンプトを開き、**gpedit.msc**と入力して Enter キーを押します。

2. ローカル グループ ポリシー エディターで、[ローカル コンピュータ ポリシー] の [コンピュータの構成]、[管理用テンプレート]、[Windows コンポーネント]、[Internet Explorer] の順に選択してください。
3. 右のペインで、[コンピュータ別にプロキシを設定する (ユーザー別ではなく)] を選択し、[ポリシー設定の編集] を選択してください。
4. [有効] を選択し、[適用] を選択してください。
5. Internet Explorer を開き、[ツール] ボタンを選択してください。
6. [インターネット オプション] を選択し、[接続] タブを選択してください。
7. [LAN の設定] を選択してください。
8. [プロキシ サーバー] の [LAN にプロキシ サーバーを使用する] を選択してください。
9. アドレスとポート情報を指定し、[OK] を選択してください。

EC2 Windows インスタンスのシステムダイアログから EC2Config サービスプロパティを設定する

[EC2 サービスプロパティ] システムダイアログを使用して設定を有効化または無効化する手順を以下に説明します。

1. Windows インスタンスを起動して接続します。
2. [Start] メニューから [All Programs] をクリックし、次に [EC2ConfigService Settings] をクリックしてください。



3. [EC2 サービスプロパティ] システムダイアログの [一般] タブで、次の設定を有効化または無効化できます。

   コンピュータ名の設定

   この設定が有効になっている場合 (デフォルトでは無効になっている)、起動のたびにホスト名が現在の内部 IP アドレスと比較されます。ホスト名と内部 IP アドレスが一致しない場合は、ホスト名がリセットされ、ホスト名に内部 IP アドレスが含まれます。その後、システムが再起動して新しいホスト名が取得されます。独自のホスト名を設定するには、また既存のホスト名が変更されないようにするには、この設定を有効にしないでください。

[User Data]

ユーザーデータの実行により、スクリプトをインスタンスのメタデータに指定できます。デフォルトでは、これらのスクリプトは最初の起動時に実行されます。また、次回の再起動時やインスタンスの起動時、またはインスタンスの再起動時やインスタンスの起動時に実行するように設定することもできます。

大きいスクリプトがある場合は、ユーザーデータを使ってスクリプトをダウンロードして実行することをお勧めします。

詳細については、ユーザーデータの実行を参照してください。

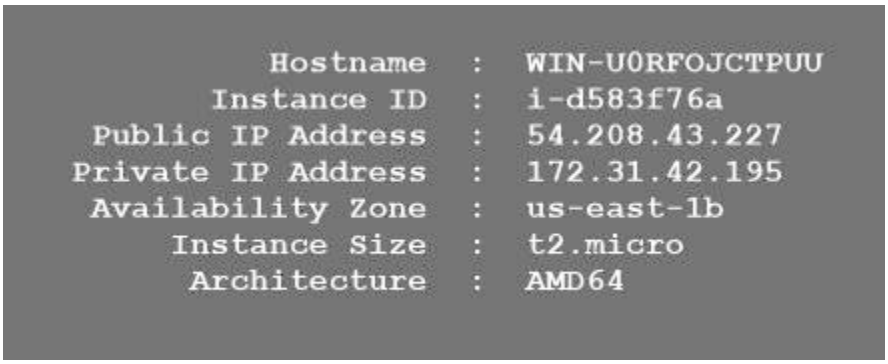
[Event Log]

この設定を使用して、起動時にイベントログエントリをコンソールに表示し、監視とデバッグを容易にします。

[Settings] をクリックすると、コンソールに出力するログエントリにフィルタを指定できます。デフォルトのフィルタは、システムイベントログから 3 つの最新エラーエントリをコンソールに出力します。

[Wallpaper Information]

この設定を使用して、システム情報をデスクトップの背景に表示します。以下はデスクトップの背景に表示される情報のサンプルです。

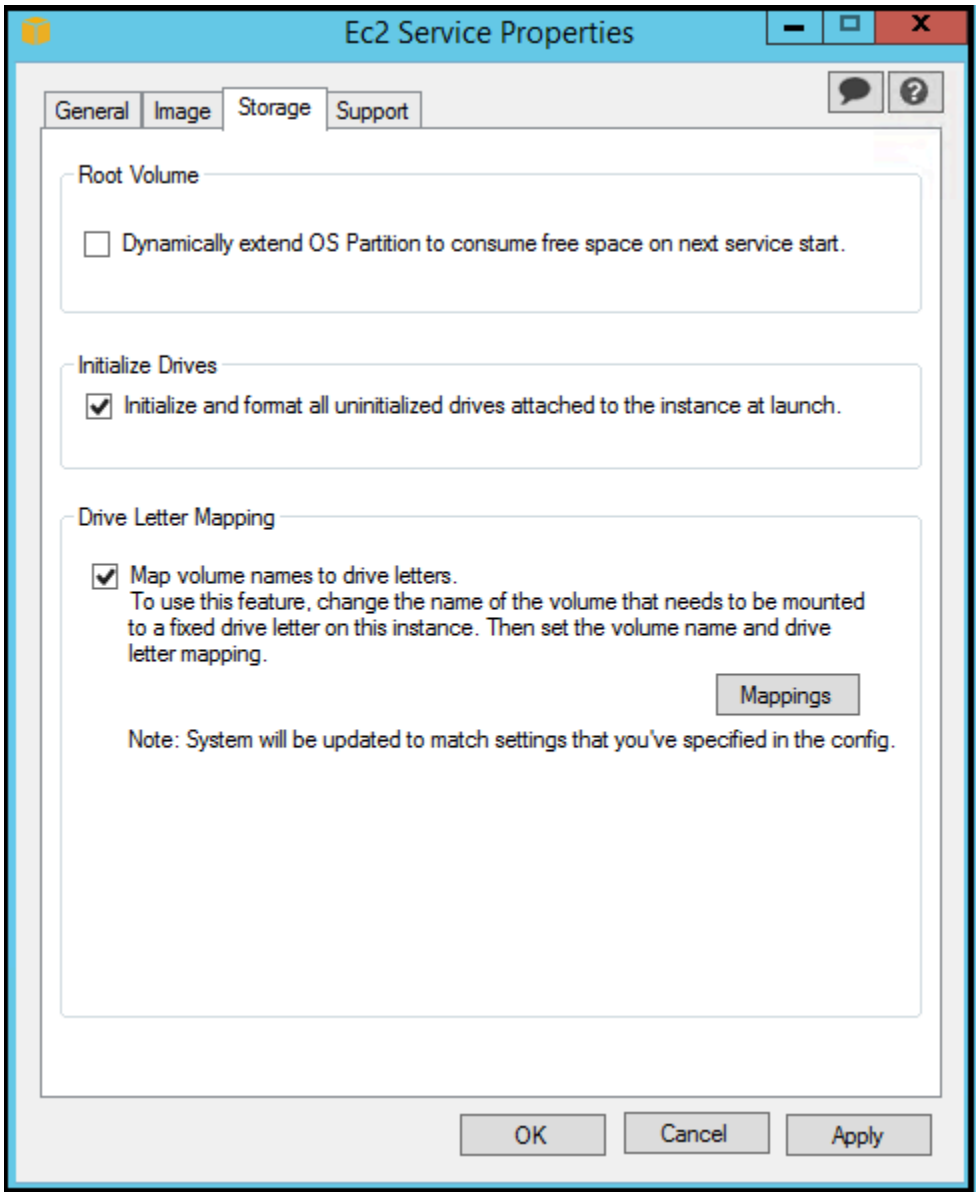


デスクトップの背景に表示される情報は、設定ファイル `EC2ConfigService\Settings\WallpaperSettings.xml`で制御されます。

休止を有効にする

この設定を使用して、EC2 からオペレーティングシステムに休止を実行するように通知することができます。

4. [Storage] タブをクリックしてください。有効化または無効化できる設定は以下のとおりです。



[Root Volume]

この設定は、ディスク 0 / ボリューム 0 が未使用の領域を含むように、動的に拡張します。独自のサイズを指定したルートボリュームからインスタンスを起動するときに便利です。

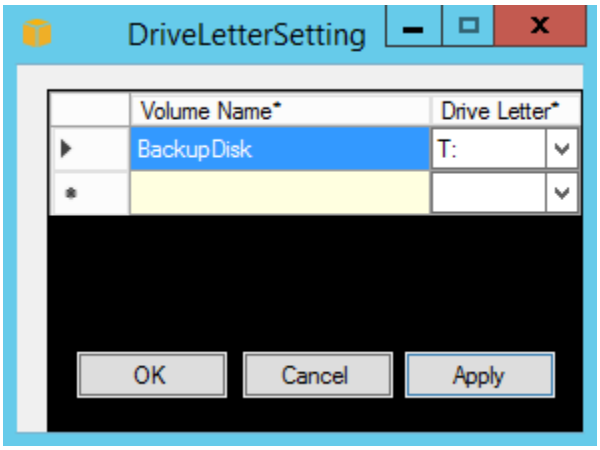
[Initialize Drives]

この設定は、インスタンスの起動中に、インスタンスにアタッチされた全ボリュームのフォーマットとマウントを行います。

[Drive Letter Mapping]

システムは、インスタンスにアタッチされたボリュームにドライブ文字をマッピングします。Amazon EBS ボリュームの場合、デフォルトでは D: から Z: のドライブ文字が割り当てられます。インスタンスストアボリュームの場合、デフォルトはドライバーにより異なります。AWSPV ドライバーと Citrix PV ドライバーはインスタンスストアボリュームに Z: から A: の順番でドライブ文字を割り当てます。Red Hat ドライバーは、インスタンスストアボリュームに D: から Z: の順番でドライブ文字を割り当てます。

ボリュームのドライブ文字を選択するには、[Mappings] をクリックしてください。[DriveLetterSetting] ダイアログボックスで、[ボリューム名] と [ドライブ文字] の値をボリュームごとに指定した後、[適用] をクリックし、[OK] をクリックしてください。中間付近のアルファベットのドライブ文字など、使用されている可能性が高いドライブ文字と競合しないドライブ文字を選択することをお勧めします。



ドライブ文字のマッピングを指定して、指定したボリューム名の 1 つと同じラベルを持つボリュームをアタッチすると、EC2Config は指定したドライブ文字をそのボリュームに自動的に割り当てます。ただし、ドライブ文字がすでに使用されている場合、ドライブ文字のマッピングは失敗します。EC2Config は、ドライブ文字のマッピングを指定したときにマウント済みだったボリュームのドライブ文字を変更しません。

5. 設定をいったん保存して後で作業を再開するには、[OK] をクリックして [EC2 サービスプロパティ] システムダイアログを閉じます。インスタンスの設定が終了し、そのインスタンスから AMI を作成する場合は、Windows Sysprep を使用して Amazon EC2 AMI を作成する を参照してください。

EC2Config 起動エージェントに関する問題のトラブルシューティング

次の情報は、EC2Config サービスの問題のトラブルシューティングに役立ちます。

接続できないインスタンスの EC2Config の更新

リモートデスクトップを使用して接続できない Windows Server インスタンスの EC2Config サービスを更新するには、以下の手順を使用します。

接続できない Amazon EBS-Backed Windows インスタンスで EC2Config を更新するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. 影響のあるインスタンスを特定します。インスタンスを選択し、[インスタンスの状態] をクリックし、[インスタンスの停止] を選択してください。

   > **Warning**
   >
   > インスタンスストアボリューム上のデータは、インスタンスを停止すると失われます。このデータを保持するには、永続ストレージにバックアップしてください。

4. [インスタンスの起動] を選択し、影響のあるインスタンスと同じアベイラビリティーゾーンに一時 `t2.micro` インスタンスを作成します。影響のあるインスタンスの起動には使用していない、別の AMI を使用します。

   > **Important**
   >
   > 影響のあるインスタンスと同じアベイラビリティーゾーンにインスタンスを作成しない場合、影響のあるインスタンスのルートボリュームを新しいインスタンスにアタッチできません。

5. EC2 コンソールで、[Volumes] を選択してください。
6. 影響のあるインスタンスのルートボリュームを見つけます。ボリュームをデタッチし、先ほど作成した一時インスタンスにボリュームをアタッチします。デフォルトのデバイス名 (xvdf) でアタッチしてください。
7. リモートデスクトップを使用して一時インスタンスに接続したら、Disk Management ユーティリティを使用してボリュームを有効にします。
8. EC2Config サービスの最新バージョンをダウンロードします。`.zip` ファイルを、アタッチしたドライブの `Temp` ディレクトリに展開します。
9. 一時インスタンスで、[実行] ダイアログボックスを開き、**`regedit`** と入力して、Enter キーを押します。

10. HKEY_LOCAL_MACHINE を選択してください。[File] メニューの [Load Hive] を選択してください。ドライブを選択した後、次のファイル Windows\System32\config\SOFTWAREに移動して開きます。プロンプトが表示されたら、キー名を指定します。
11. ロードしたキーを選択し、Microsoft\Windows\CurrentVersion に移動します。RunOnce キーを選択してください。このキーが存在しない場合は、コンテキスト (右クリック) メニューから CurrentVersion を選択し、[New] を選択した後、[Key] を選択してください。キーに RunOnce と名付けます。
12. コンテキスト (右クリック) メニューから RunOnce キーを選択し、[New] を選択した後、[String Value] を選択してください。名前に Ec2Install、データに C:\Temp\Ec2Install.exe /quiet と入力してください。
13. HKEY_LOCAL_MACHINE*specified key name*\Microsoft\Windows NT \CurrentVersion\Winlogon キーを選択します。コンテキスト (右クリック) メニューから [New] を選択した後、[String Value] を選択してください。名前に **AutoAdminLogon**、値データに **1** と入力してください。
14. HKEY_LOCAL_MACHINE*specified key name*\Microsoft\Windows NT \CurrentVersion\Winlogon> キーを選択してください。コンテキスト (右クリック) メニューから [New] を選択した後、[String Value] を選択してください。名前に **DefaultUserName**、値データに **Administrator** と入力してください。
15. HKEY_LOCAL_MACHINE*specified key name*\Microsoft\Windows NT \CurrentVersion\Winlogon キーを選択してください。コンテキスト (右クリック) メニューから [New] を選択した後、[String Value] を選択してください。名前に **DefaultPassword**、値データにパスワードを入力してください。
16. レジストリエディターのナビゲーションペインで、最初にレジストリエディターを開いたときに作成した一時キーを選択してください。
17. [File] メニューの [Unload Hive] を選択してください。
18. Disk Management ユーティリティで、先ほどアタッチしたドライブを選択し、右クリックコンテキストメニューを開いて、[Offline] を選択してください。
19. Amazon EC2 コンソールで、影響のあるボリュームを一時インスタンスからデタッチし、/dev/sda1 というデバイス名でインスタンスに再アタッチします。ボリュームをルートボリュームとして指定するには、このデバイス名を指定する必要があります。
20. Amazon EC2 インスタンスの停止と開始 インスタンス。
21. インスタンスが起動したら、システムログをチェックして、Windows is ready to use というメッセージが表示されることを確認します。

22. レジストリエディタを開いて、HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon を選択してください。前に作成した文字列値キーを削除します。AutoAdminLogon、DefaultUserName、および DefaultPassword です。
23. この手順で作成した一時インスタンスを削除するか停止します。

EC2Config バージョン履歴

次の表は、EC2Config のリリース済みバージョンについて説明しています。SSM Agentの更新の詳細については、Systems Manager SSM Agent のリリースノートを参照してください。

> **Important**
>
> EC2Config のサポートが終了しました。最新バージョンの EC2Config エージェントのみをダウンロードできます。以前のバージョンは非公開になります。

バージョン	詳細	リリース日
4.9.5777	• 一部のインスタンスタイプで RSS 構成が正しく設定されない問題を修正しました。 • 新しいバージョンの SSM Agent 3.3.484.0 。	2024 年 6 月 17 日
4.9.5554	• レジストリエントリに基づいてドメイン名の権限委譲を制限: HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Dnscache\Parameters\DomainNameDevolutionLevel 。 • 新しいバージョンの SSM Agent 3.2.1630.0 。	2023 年 10 月 4 日
4.9.5467	• コンソールポートを検出するための再試行機能を追加しました。 • 新しいバージョンの SSM Agent 3.1.2282.0 。	2023 年 8 月 1 日

バージョン	詳細	リリース日
4.9.5288	• AWS Core SDK をバージョン 3.7.103.23 に更新しました。 • IMDSv2 のみを有効にしたインスタンスで AWS-UpdateEC2Config SSM ドキュメントが EC2Config の更新に失敗する問題を修正しました。 • 新しいバージョンの SSM Agent 3.1.2144.0 。	2023 年 3 月 8 日
4.9.5231	• SSM Agent 3.1.1927.0 の新しいバージョン。	2023 年 2 月 14 日
4.9.5103	• r5d および i4i インスタンスファミリーでエフェメラルボリュームが誤って識別される問題を修正しました。 • SSM エージェント 3.1.1856.0 の新しいバージョン。	2022 年 12 月 5 日
4.9.5064	• PCI セグメント情報を使用してコンソールポートを選択するように更新されました。 • PowerShell スクリプトに署名し、著作権ヘッダーを追加しました。 • プライマリネットワークアダプタの選択ロジックを修正しました。 • SSM エージェント 3.1.1732.0 の新しいバージョン。	2022 年 11 月 16 日
4.9.4588	• IMDSv2 リクエストのみを行うように IMDS 待機ロジックを更新しました。 • libec2launch.dll 起動エージェント共有ライブラリを追加しました。 • SSM エージェント 3.1.1188.0 の新しいバージョン	2022 年 5 月 31 日

バージョン	詳細	リリース日
4.9.4556	• 使用前の NIC に対し完全な初期化を行うための待機ロジックを追加しました。 • 新しいバージョンの Log4Net 2.0.14.0 では、セキュリティパッチが取得されます。 • 新しいバージョンの SSM Agent 3.1.1045.0 では、セキュリティパッチが取得されます。	2022 年 3 月 1 日
4.9.4536	• Temp フォルダがない場合にユーザーデータがクラッシュする問題を修正しました。 • SSM Agent の新しいバージョン 3.1.804.0。	2022 年 1 月 31 日
4.9.4508	• diskpart スクリプトのパスを正しく計算できるように問題を修正。 • SSM Agent の新しいバージョン 3.1.338.0。	2021 年 10 月 6 日
4.9.4500	• IMDS v2 のサポートによって`Install-EgpuManagerConfig` を更新します。 • httpsを使用するようにWebリンクを更新しました。 • 新しいバージョンの SSM Agent (3.1.282.0)	2021 年 9 月 7 日
4.9.4419	• IMDS バージョン 1 のフォールバックロジックを修正しました • Windows の一時ディレクトリのすべての使用状況を EC2Config 一時ディレクトリに更新 • SSM Agent の新しいバージョン 3.0.1124.0	2021 年 6 月 2 日

バージョン	詳細	リリース日
4.9.4381	• EC2ConfigUpdater で SSM ドキュメントスキーマバージョン 2.2 のサポートを追加しました • コンソールログに AWS Nitro Enclaves パッケージバージョンを追加。 • SSM Agent の新しいバージョン 3.0.529.0	2021 年 5 月 4 日
4.9.4326	• 設定 UI のすべてのリンクを削除しました • Windows Server 2008 をサポートする最後の EC2Config バージョンです。	2021 年 3 月 3 日
4.9.4279	• `Ec2ConfigMonitor` スケジュールされたタスクに関連するセキュリティ問題の修正 • ドライブ文字マッピングの問題と不正確な一時ディスク数の修正 • コンソール出力への `OsCurrentBuild` と `OsReleaseId` の追加 • 新しいバージョンの SSM Agent (2.3.871.0)	2020 年 12 月 11 日
4.9.4222	• IMDS バージョン 1 のフォールバックロジックを修正しました • 新しいバージョンの SSM Agent (2.3.842.0)	2020 年 4 月 7 日
4.9.4122	• IMDS v2 のサポートを追加しました • 新しいバージョンの SSM Agent (2.3.814.0)	2020 年 3 月 4 日
4.9.3865	• メタルインスタンスの Windows Server 2008 R2 の COM ポートを検出する問題を修正しました • 新しいバージョンの SSM Agent (2.3.722.0)	2019 年 10 月 31 日
4.9.3519	• 新しいバージョンの SSM Agent (2.3.634.0)	2019 年 6 月 18 日

バージョン	詳細	リリース日
4.9.3429	• 新しいバージョンの SSM Agent 2.3.542.0	2019 年 4 月 25 日
4.9.3289	• 新しいバージョンの SSM Agent 2.3.444.0	2019 年 2 月 11 日
4.9.3270	• ACPI の問題を解決するためにモニターをオフにしないように設定するためのプラグインを追加しました • SQL Server のエディションとバージョンがコンソールに書き込まれます • 新しいバージョンの SSM Agent 2.3.415.0	2019 年 1 月 22 日
4.9.3230	• 機能に合わせてドライブ文字のマッピングの説明が更新されました • 新しいバージョンの SSM Agent 2.3.372.0	2019 年 1 月 10 日
4.9.3160	• プライマリ NIC の待機時間の増加 • ENA デバイス用の RSS および受信キュー設定のデフォルト設定を追加 • Sysprep 中の休止状態を無効化 • 新しいバージョンの SSM Agent 2.3.344.0 • AWS SDK を 3.3.29.13 にアップグレードしました	2018 年 12 月 15 日
4.9.3067	• インスタンスを休止状態にするための改善 • 新しいバージョンの SSM Agent 2.3.235.0	2018 年 11 月 8 日
4.9.3034	• DNS サーバーに ルート 169.254.169.253/32 を追加しました • 新しいバージョンの SSM Agent 2.3.193.0	2018 年 10 月 24 日
4.9.2986	• すべての EC2Config 関連バイナリに追加された署名 • 新しいバージョンの SSM Agent 2.3.136.0	2018 年 10 月 11 日
4.9.2953	新しいバージョンの SSM Agent (2.3.117.0)	2018 年 10 月 2 日

バージョン	詳細	リリース日
4.9.2926	新しいバージョンの SSM Agent (2.3.68.0)	2018 年 9 月 18 日
4.9.2905	• 新しいバージョンの SSM Agent (2.3.50.0) • AMZN タイムサービスにルート 169.254.169.123/32 を追加 • GRID ライセンスにルート 169.254.169.249/32 を追加 • EBS NVMe ボリュームが一時的なものとしてマークされる問題を修正	2018 年 9 月 17 日
4.9.2854	新しいバージョンの SSM Agent (2.3.13.0)	2018 年 8 月 17 日
4.9.2831	新しいバージョンの SSM Agent (2.2.916.0)	2018 年 8 月 7 日
4.9.2818	新しいバージョンの SSM Agent (2.2.902.0)	2018 年 7 月 31 日
4.9.2756	新しいバージョンの SSM Agent (2.2.800.0)	2018 年 6 月 27 日
4.9.2688	新しいバージョンの SSM Agent (2.2.607.0)	2018 年 5 月 25 日
4.9.2660	新しいバージョンの SSM Agent (2.2.546.0)	2018 年 5 月 11 日
4.9.2644	新しいバージョンの SSM Agent (2.2.493.0)	2018 年 4 月 26 日
4.9.2586	新しいバージョンの SSM Agent (2.2.392.0)	2018 年 3 月 28 日

バージョン	詳細	リリース日
4.9.2565	• 新しいバージョンの SSM Agent (2.2.355.0) • M5 および C5 インスタンスの問題を修正しました (PV ドライバーを見つけられない) • インスタンスタイプ、最新の PV ドライバ、および NVMe ドライバのコンソールでのログ記録を追加しました。	2018 年 3 月 13 日
4.9.2549	新しいバージョンの SSM Agent (2.2.325.0)	2018 年 3 月 8 日
4.9.2461	新しいバージョンの SSM Agent (2.2.257.0)	2018 年 2 月 15 日
4.9.2439	新しいバージョンの SSM Agent (2.2.191.0)	2018 年 2 月 6 日
4.9.2400	新しいバージョンの SSM Agent (2.2.160.0)	2018 年 1 月 16 日
4.9.2327	• 新しいバージョンの SSM Agent (2.2.120.0) • Amazon EC2 ベアメタルインスタンスに COM ポート検出を追加 • Amazon EC2 ベアメタルインスタンスに Hyper-V のステータスログを追加	2018 年 1 月 2 日
4.9.2294	新しいバージョンの SSM Agent (2.2.103.0)	2017 年 12 月 4 日
4.9.2262	新しいバージョンの SSM Agent (2.2.93.0)	2017 年 11 月 15 日
4.9.2246	新しいバージョンの SSM Agent (2.2.82.0)	2017 年 11 月 11 日

バージョン	詳細	リリース日
4.9.2218	新しいバージョンの SSM Agent (2.2.64.0)	2017 年 10 月 29 日
4.9.2212	新しいバージョンの SSM Agent (2.2.58.0)	2017 年 10 月 23 日
4.9.2203	新しいバージョンの SSM Agent (2.2.45.0)	2017 年 10 月 19 日
4.9.2188	新しいバージョンの SSM Agent (2.2.30.0)	2017 年 10 月 10 日
4.9.2180	• 新しいバージョンの SSM Agent (2.2.24.0) • GPU インスタンスの Elastic GPU が追加されました	2017 年 10 月 5 日
4.9.2143	新しいバージョンの SSM Agent (2.2.16.0)	2017 年 10 月 1 日
4.9.2140	新しいバージョンの SSM Agent (2.1.10.0)	
4.9.2130	新しいバージョンの SSM Agent (2.1.4.0)	
4.9.2106	新しいバージョンの SSM Agent (2.0.952.0)	
4.9.2061	新しいバージョンの SSM Agent (2.0.922.0)	
4.9.2047	新しいバージョンの SSM Agent (2.0.913.0)	
4.9.2031	新しいバージョンの SSM Agent (2.0.902.0)	
4.9.2016	• 新しいバージョンの SSM Agent (2.0.879.0) • Windows Server 2003 の CloudWatch Logs ディレクトリパスを修正しました。	

バージョン	詳細	リリース日
4.9.1981	• 新しいバージョンの SSM Agent (2.0.847.0) • EBS ボリュームで生成される `important.txt` の問題を修正しました。	
4.9.1964	新しいバージョンの SSM Agent (2.0.842.0)	
4.9.1951	• 新しいバージョンの SSM Agent (2.0.834.0) • エフェメラルドライブに対して Z からドライブ文字がマッピングされない問題を修正しました。	
4.9.1925	• 新しいバージョンの SSM Agent (2.0.822.0) • [バグ] このバージョンは、SSM Agent v4.9.1775 の有効な更新ターゲットではありません。	
4.9.1900	新しいバージョンの SSM Agent (2.0.805.0)	
4.9.1876	• 新しいバージョンの SSM Agent (2.0.796.0) • 管理者とユーザーデータ実行の出力/エラーリダイレクトの問題を修正しました。	
4.9.1863	• 新しいバージョンの SSM Agent (2.0.790.0) • Amazon EC2 インスタンスに複数の EBS ボリュームをアタッチする際の問題を修正しました。 • 設定パスを指定し、下位互換性を維持するよう CloudWatch を強化しました。	
4.9.1791	新しいバージョンの SSM Agent (2.0.767.0)	

バージョン	詳細	リリース日
4.9.1775	新しいバージョンの SSM Agent (2.0.761.0)	
4.9.1752	新しいバージョンの SSM Agent (2.0.755.0)	
4.9.1711	新しいバージョンの SSM Agent (2.0.730.0)	
4.8.1676	新しいバージョンの SSM Agent (2.0.716.0)	
4.7.1631	新しいバージョンの SSM Agent (2.0.682.0)	
4.6.1579	• 新しいバージョンの SSM Agent (2.0.672.0) • v4.3、v4.4、v4.5 のエージェント更新問題を修正しました	
4.5.1534	新しいバージョンの SSM Agent (2.0.645.1)	
4.4.1503	新しいバージョンの SSM Agent (2.0.633.0)	
4.3.1472	新しいバージョンの SSM Agent (2.0.617.1)	
4.2.1442	新しいバージョンの SSM Agent (2.0.599.0)	
4.1.1378	新しいバージョンの SSM Agent (2.0.558.0)	

バージョン	詳細	リリース日
4.0.1343	• Run Command、State Manager、CloudWatch エージェント、およびドメイン結合のサポートは、SSM Agent と呼ばれる別のエージェントに移行されました。SSM Agent は、EC2Config アップグレードの一部としてインストールされます。詳細については、EC2Config および AWS Systems Manager を参照してください。 • EC2Config でプロキシをセットアップしてある場合は、アップグレードする前に SSM Agent のプロキシ設定を更新する必要があります。プロキシ設定を更新しない場合、Run Command を使用してインスタンスを管理することはできません。この状態を回避するためには、新しいバージョンに更新する前に、AWS Systems Manager ユーザーガイドのWindows インスタンスでの SSM Agent のインストールと設定を確認してください。 • ローカル設定ファイル `AWS.EC2.Windows.CloudWatch.json` を使用して、既にインスタンスでの CloudWatch 統合を有効にしている場合、SSM Agent と連携して動作するように、そのファイルを設定する必要があります。	
3.19.1153	• 古い AWS KMS 設定を持つインスタンス用に、アクティブ化のプラグインを再び有効にしました。BYOL ユーザーのアクティベーションをスキップしてください。 • ディスクフォーマットオペレーション中はデフォルトの TRIM 動作を無効にするよう変更し、InitializeDisks プラグインをユーザーデータで上書きするため FormatWithTRIM を追加しました。	

バージョン	詳細	リリース日
3.18.1118	• プライマリネットワークアダプタに確実にルートを追加するための修正。 • AWS サービスのサポートを向上させる更新。	
3.17.1032	• フィルタが同じカテゴリに設定された場合に表示される、重複したシステムログが修正されます。 • ディスク初期化時のハングを防ぐ修正。	
3.16.930	起動時に Windows イベントログに 「Windows の使用準備ができている」イベントを記録するサポートを追加しました。	
3.15.880	'.' 文字を持つ S3 バケット名への Systems Manager Run Command 出力のアップロードを許可する修正。	
3.14.786	InitializeDisks プラグインの設定を上書きするサポートが追加されました。例: SSD のディスク初期化を高速化するには、ユーザーデータでこれを指定することで TRIM を一時的に無効にすることができます。 <InitializeDrivesSettings><SettingsGroup>FormatWithoutTRIM</SettingsGroup></InitializeDrivesSettings	
3.13.727	Systems Manager RunCommand - Windows の再起動後にコマンドを確実に処理するための修正。	
3.12.649	• コマンド/スクリプトの実行時に再起動を適切に処理するための修正。 • 実行中のコマンドを確実にキャンセルするための修正。 • Systems Manager Run Command を通じてアプリケーションをインストールする際に、(オプションで) MSI ログを S3 にアップロードするサポートを追加します。	

バージョン	詳細	リリース日
3.11.521	• Windows Server 2003 用の RDP サムプリントの生成を有効にするための修正。 • EC2Config ログの行にタイムゾーンと UTC オフセットを含めるための修正。 • コマンドを並行して実行するための Systems Manager サポート。 • パーティション分割ディスクをオンライン状態にするための、以前の変更のロールバック。	
3.10.442	• MSI アプリケーションをインストールする際の Systems Manager 設定エラーを修正します。 • ストレージディスクを確実にオンライン状態にするための修正。 • AWS サービスのサポートを向上させる更新。	
3.9.359	• Windows アップデートの設定をデフォルト状態のままにする、Sysprep 後のスクリプトの修正。 • GPO パスワードポリシーの設定を取得する際の信頼性を向上させるパスワード生成プラグインの修正。 • ローカル管理者グループへの EC2Config/SSM ログフォルダのアクセス許可を制限します。 • AWS サービスのサポートを向上させる更新。	

バージョン	詳細	リリース日
3.8.294	• プライマリドライブ上にないときに、ログがアップロードされない CloudWatch の問題を修正しました。 • 再試行ロジックを追加して、ディスクの初期化プロセスを改善しました。 • SetPassword プラグインが AMI の作成中に失敗したときのエラー処理を強化し、追加しました。 • AWS サービスのサポートを向上させる更新。	
3.7.308	• インスタンス内の設定のテストとトラブルシューティング用の ec2config-cli ユーティリティの改良。 • OpenVPN アダプターで AWS KMS とメタデータサービスの静的ルートを追加できないようにしました。 • ユーザーデータの実行が 「永続的な」タグに従わない問題を修正しました。 • EC2 コンソールへのログインが利用できないときのエラー処理を強化しました。 • AWS サービスのサポートを向上させる更新。	
3.6.269	• AWS KMS を介した Windows のアクティブ化に最初にリンクローカルアドレス 169.254.0.250/251 を使用する、Windows のアクティブ化の信頼性の修正 • Systems Manager、Windows アクティベーション、およびドメイン結合のシナリオのプロキシ処理の強化 • 重複した行のユーザーアカウントが Sysprep 応答ファイルに追加された問題を修正しました	

バージョン	詳細	リリース日
3.5.228	• CloudWatch プラグインが、Windows イベントログの読み取りで過剰に CPU とメモリを消費するシナリオに対応しました • EC2Config 設定 UI で CloudWatch 設定ドキュメントへのリンクを追加しました	
3.4.212	• VM-Import と組み合わせて使用したときの EC2Config の修正。 • WiX インストーラのサービス名変更の問題を修正しました。	
3.3.174	• Systems Manager とドメイン結合の失敗の例外処理を強化しました。 • Systems Manager SSM スキーマのバージョニングをサポートする変更。 • Win2K3 での エフェメラルディスクのフォーマットを修正しました。 • 2TB 以上のディスクサイズの設定をサポートする変更。 • GC モードをデフォルトに設定して、仮想メモリ使用量を減らしました。 • `aws:psModule` および `aws:application` プラグインの UNC パスからの、アーティファクトのダウンロードをサポート。 • Windows アクティベーションプラグインのログ記録を強化しました。	

バージョン	詳細	リリース日
3.2.97	• Systems Manager SSM アセンブリのロードの遅延によるパフォーマンスの向上。 • 誤った形式の sysprep2008.xml の例外処理を強化しました。 • Systems Manager の 「適用」設定のコマンドラインのサポート。 • 保留中のコンピュータの名前変更がある場合のドメイン結合のサポートの変更。 • `aws:applications` プラグインのオプションパラメータのサポート。 • `aws:psModule` プラグインのコマンド配列のサポート。	
3.0.54	• Systems Manager サポートの有効化。 • Systems Manager を介して、EC2 Windows インスタンスを自動的に AWS ディレクトリに対しドメイン参加させます。 • Systems Manager を介して CloudWatch ログ/メトリクスを設定し、アップロードします。 • Systems Manager を介して PowerShell モジュールをインストールします。 • Systems Manager を介して MSI アプリケーションをインストールします。	

バージョン	詳細	リリース日
2.4.233	• サービス起動の障害から EC2Config を復旧するためのスケジュールされたタスクが追加されました。 • コンソールログのエラーメッセージの改良。 • AWS サービスのサポートを向上させる更新。	
2.3.313	• CloudWatch Logs 機能を有効にしたときに、大量のメモリを消費する問題を修正しました。 • アップグレードのバグを修正し、2.1.19 以前のバージョンの ec2config バージョンを最新バージョンにアップグレードできるようになりました。 • COM ポートを開く例外を更新し、ログでよりわかりやすく便利になるようにしました。 • Ec2configServiceSettings UI でサイズ変更を無効にし、UI での帰属とバージョン表示の配置を修正しました。	
2.2.12	• Windows Sysprep の状態を判断するためにレジストリキーに問い合わせる際に、null を返すことがある NullPointerException を処理しました。 • 最終ブロックでアン管理されたリソースを解放しました。	
2.2.11	空のログの行の処理に関する CloudWatch プラグインの問題を修正しました。	

バージョン	詳細	リリース日
2.2.10	• UI から CloudWatch Logs 設定を行う機能を削除しました。 • 将来の機能強化のために、ユーザーによる `%ProgramFiles%\Amazon\Ec2ConfigService\Settings\AWS.EC2.Windows.CloudWatch.json` ファイル内での CloudWatch Logs 設定の定義を有効化しました。	
2.2.9	処理されない例外を修正し、ログ記録を追加しました。	
2.2.8	• Windows Server 2003 SP1 以降をサポートするよう、EC2Config インストーラの Windows OS バージョンチェックを修正しました。 • Sysprep 設定ファイルの更新に関連するレジストリキーを読み取る際の、null 値の処理を修正しました。	
2.2.7	• Windows 2008 以降用の Sysprep の実行中に、EC2Config を実行するサポートを追加しました。 • より良い診断のため、例外処理とログ記録を強化しました。	
2.2.6	• ログイベントのアップロード時のインスタンスと CloudWatch Logs への負荷を減らしました。 • CloudWatch Logs プラグインが常に有効にならないアップグレードの問題に対応しました	

バージョン	詳細	リリース日
2.2.5	• CloudWatch ログサービスにログをアップロードするサポートを追加しました。 • Ec2OutputRDPCert プラグインの競合状態の問題を修正しました • EC2Config サービス復旧オプションを TakeNoAction から再起動するよう変更しました • EC2Config がクラッシュしたときの例外情報を追加しました	
2.2.4	• PostSysprep.cmd のタイプミスを修正しました • EC2Config が OS2012+ のスタートメニューにピン留めされないバグを修正しました	
2.2.3	• インストール時に即座にサービスを開始することなく EC2Config をインストールするオプションを追加しました。使用するには、コマンドプロンプトから 'Ec2Install.exe start=false' を実行します • 壁紙プラグインに、壁紙の追加/削除を制御するパラメータを追加しました。使用するには、コマンドプロンプトから 'Ec2WallpaperInfo.exe set' または 'Ec2WallpaperInfo.exe revert' を実行します • RealTimeIsUniversal キーのチェックを追加し、RealTimeIsUniveral レジストリキーの正しくない設定をコンソールに出力しました • Windows の temp フォルダの EC2Config 依存関係を削除しました • .Net 3.5 で UserData の実行の依存関係を削除しました	

バージョン	詳細	リリース日
2.2.2	• リソースがリリースされていることを確認するため、サービス停止動作のチェックを追加しました • ドメインに結合されると長い実行時間がかかる問題を修正しました	
2.2.1	• 古いバージョンからのアップグレードを可能にするため、インストーラを更新しました • Net4.5 のみの環境での Ec2WallpaperInfo バグを修正しました • 断続的なドライバー検出バグを修正しました • サイレントインストールオプションを追加しました。'q' オプションを指定して 'Ec2Install.exe -q' を実行します。	
2.2.0	• Net4 および Net4.5 のみの環境のサポートが追加されました • インストーラを更新しました	
2.1.19	• Intel ネットワークドライバー (例: C3 インスタンスタイプ) 詳細については、Amazon EC2 インスタンスでの拡張ネットワーキングを参照してください。 • コンソールの出力に AMI オリジンバージョンと AMI オリジン名のサポートが追加されました • 一貫性のあるフォーマット/解析のためにコンソール出力に変更を加えました。 • ヘルプファイルを更新しました	

バージョン	詳細	リリース日
2.1.18	• 完了通知用の EC2Config WMI オブジェクト (-Namespace root\Amazon -Class EC2_ConfigService) が追加されました • 大きなイベントログがあるスタートアップ WMI クエリのパフォーマンスを改善しました。この問題により、最初の実行時に高い CPU 利用率が長く続く可能性があります。	
2.1.17	• 標準出力および標準エラーバッファがいっぱいになる UserData 実行の問題を修正しました • w2k8 OS 以降で、コンソール出力に正しくない RDP サムプリントが表示される問題を修正しました • Windows 2008 以降のコンソール出力で、'RDPCERTIFICATE-SubjectName:' が含まれ、マシン名の値が含まれるようになりました。 • D:\ をドライブ文字のマッピングドロップダウンに追加しました • [Help] ボタンを右上に移動し、外観とフィーリングを変更しました • [Feedback survey] リンクを右上に追加しました	

バージョン	詳細	リリース日
2.1.16	• [General] タブには、新しいバージョン用の EC2Config ダウンロードページへのリンクが含まれるようになりました • デスクトップの壁紙のオーバーレイは、MyDoc のリダイレクトをサポートするため、My Documents ではなく Users Local Appdata フォルダに保存されるようになりました • MSSQLServer の名前が、Post-Sysprep スクリプト (2008 以降) のシステムと同期されました • アプリケーションフォルダの順序を変更しました (ファイルを Plugin ディレクトリに移動し、重複したファイルを削除しました) • システムログ出力を変更しました (コンソール): • *解析を容易にするため、日付、名前、値の形式に移行しました (依存関係の新しい形式への移行を開始してください) • *'Ec2SetPassword' プラグインステータスが追加されました • *Sysprep の開始時刻と終了時刻が追加されました • 英語以外のオペレーティングシステムで、エフェメラルディスクが一時ストレージとして分類されない問題を修正しました • Sysprep の実行後に EC2Config のアンインストールに失敗する問題を修正しました	

バージョン	詳細	リリース日
2.1.15	• メタデータサービスへのリクエストを最適化しました • メタデータはプロキシ設定をバイパスするようになりました • エフェメラルディスクが見つかった場合、一時ストレージとして分類され、Important.txt がボリュームに配置されます (Citrix PV ドライバーのみ)。詳細については、EC2 Windows インスタンスでの PV ドライバーのアップグレードを参照してください。 • エフェメラルディスクはドライブ文字を Z から A まで割り当てました (Citrix PV ドライバーのみ) が、ボリュームラベル 'Temporary Storage X' を使用してドライブ文字マッピングプラグインで割り当てを上書きできるようになりました (x は 0 ~ 25 の数値)。 • UserData は、'Windows の準備が完了' の直後に実行できるように	
2.1.14	デスクトップの壁紙の修正	
2.1.13	• デスクトップの壁紙はデフォルトでホスト名を表示します • Windows タイムサービスの依存関係を削除しました • 複数の IP が単一のインターフェイスに割り当てられる場合にルートを追加しました	
2.1.11	• Ec2Activation プラグインを変更しました • -30 日ごとにアクティベーションステータスを確認します • -猶予期間の残りが 90 日 (180 日中) の場合、アクティベーションを再試行します	

バージョン	詳細	リリース日
2.1.10	• Sysprep を使用した場合、または Sysprep なしでシャットダウンした場合、デスクトップ壁紙のオーバーレイは保持されなくなりました • <persist>true</persist> を使用したサービスの開始ごとに実行するユーザーデータオプション • /DisableWinUpdate.cmd の場所と名前を /Scripts/PostSysprep.cmd に変更しました • /Scripts/PostSysprep.cmd で、デフォルトでは管理者パスワードの有効期限が切れないように設定されました • アンインストールで EC2Config PostSysprep スクリプトが c:\windows\setup\script\CommandComplete.cmd から削除されます • [Add Route] でカスタムインターフェイスメトリクスがサポートされます	
2.1.9	UserData の実行は 3851 文字に制限されなくなりました	

バージョン	詳細	リリース日
2.1.7	• OS バージョンと言語 ID がコンソールに書き込まれます • EC2Config バージョンがコンソールに書き込まれます • PV ドライバーバージョンがコンソールに書き込まれます • バグチェックが検出され、見つかったときは次回の起動時にコンソールに出力されます • Sysprep 認証情報を保持するためにオプションが config.xml に追加されました • ENI が起動できない場合に備えて、ルート再試行ロジックが追加されました • ユーザーデータの実行 PID がコンソールに書き込まれます • 生成される最小のパスワードの長さが GPO から取得されます • 3 回再試行を開始するようサービスが設定されます • S3_DownloadFile.ps1 および S3_Upload file.ps1 の例を / Scripts フォルダに追加しました	

バージョン	詳細	リリース日
2.1.6	• [General] タブに追加されたバージョン情報 • [Bundle] タブの名前が [Image] に変更されました • パスワードの指定プロセスを簡略化し、パスワード関連の UI を [General] タブから [Image] タブに移動しました • [Disk Settings] タブの名前を [Storage] に変更しました • トラブルシューティング用の一般的なツールを備えた [Support] タブが追加されました • Windows Server 2003 `sysprep.ini` がデフォルトで OS パーティションを拡張するよう設定されました • プライベート IP アドレスを壁紙に追加しました • プライベート IP アドレスが壁紙に表示されます • コンソール出力の再試行ロジックが追加されました • メタデータのアクセスのしやすさのため、Com ポートの例外を修正しました。この問題により、コンソール出力が表示される前に EC2Config が終了していました • 起動ごとにアクティベーションのステータスを確認 -- 必要に応じてアクティベートされます • 相対パスの問題を修正 -- この問題は、スタートアップフォルダから壁紙のショートカットを手動で実行したときに発生し、Administrator/logs を指していました • Windows Server 2003 ユーザーのデフォルトの背景色を修正しました (管理者以外)	

バージョン	詳細	リリース日
2.1.2	• UTC (ズールー) のコンソールタイムスタンプ • [Sysprep] タブのハイパーリンクの外観を削除しました • Windows 2008 以降の初回起動時にルートボリュームを動的に拡張する機能の追加 • パスワードの設定を有効にすると、EC2Config で自動的にパスワードの設定が有効になります • EC2Config は Sysprep を実行する前にアクティベーションのステータスをチェックします (アクティベートされていない場合は警告を表示) • Windows Server 2003 `Sysprep.xml` は、デフォルトで太平洋時間ではなく UTC タイムゾーンを使用するようになりました • ランダム化されたアクティベーションサーバー • [Drive Mapping] タブの名前を [Disk Settings] に変更しました • [Initialize Drives] の UI 項目を [General] から [Disk Settings] タブに移動しました • [Help] ボタンは HTML ヘルプファイルを指すようになりました • HTML ヘルプファイルを変更し、更新しました • ドライブ文字のマッピングに関する注意の内容を更新しました • パッチを自動化し、Sysprep の前にクリーンアップするため、InstallUpdates.ps1 を /Scripts フォルダに追加しました	

バージョン	詳細	リリース日
2.1.0	• デスクトップの壁紙に、(接続/再接続ではなく) 初回のログオン時にデフォルトでインスタンス情報が表示されます • コードを <powershell></powershell> で囲むことにより、PowerShell をユーザーデータから実行できるように	

Windows インスタンスで EC2 Fast Launch を使用する

EC2 Fast Launch の Windows Server AMI を設定すると、Amazon EC2 は次のように、より高速に起動するために使用する事前プロビジョニングされたスナップショットのセットを作成します。

1. Amazon EC2 は、ユーザーの設定に基づいて一時的な t3 インスタンスのセットを起動します。
2. それぞれの一時インスタンスが標準の起動ステップを完了すると、Amazon EC2 はインスタンスの事前プロビジョニングされたスナップショットを作成します。スナップショットは Amazon S3 バケットに保存されます。
3. スナップショットの準備が整うと、Amazon EC2 は関連する t3 インスタンスを終了して、リソースコストを可能な限り低く抑えます。
4. 次に Amazon EC2 が EC2 Fast Launch が有効になっている AMI からインスタンスを起動したときは、スナップショットのうちの 1 つを使って起動時間を大幅に短縮します。

Amazon EC2 は、手元にあるスナップショットを使用して EC2 Fast Launch が有効になっている AMI からインスタンスを起動する際に、自動的にスナップショットを補充します。

EC2 Fast Launch が有効になっている AMI へのアクセス権を持つアカウントは、起動時間を短縮できるという利点があります。インスタンスを起動するためのアクセス権を AMI 所有者がユーザーに対し付与する際は、事前プロビジョニングのスナップショットが AMI 所有者のアカウントから取得されます。

EC2 Fast Launch をサポートしている AMI を共有しているユーザーは、その共有した AMI での高速起動を有効または無効にすることができます。ユーザーが、共有した AMI で EC2 Fast Launch を有効にしている場合、Amazon EC2 は事前プロビジョニングされたスナップショットをユーザーのアカウントに作成します。ユーザーがアカウントのスナップショットを使い果たした場合でも、AMI 所有者のアカウントのスナップショットを引き続き使用することが可能です。

> **Note**
>
> EC2 Fast Launch では、事前にプロビジョニングされたスナップショットは、起動時に消費されるとすぐに削除されるため、ストレージコストが最小限に抑えられ、再利用が防がれます。ただし、削除されたスナップショットが保持ルールに一致する場合、ごみ箱がそれらを自動的に保持します。このようなことが起こらないように、ごみ箱の保持ルールの適用範囲を確認することを推奨します。詳細については、「Amazon EBS ユーザーガイド」の「ごみ箱」を参照してください。
> この機能は、EBS 高速スナップショットの復元と同一ではありません。EBS の高速スナップショット復元は、スナップショットごとに明示的に有効化する必要があり、関連コストは個別にかかります。

次のビデオでは、Windows AMI を迅速に起動できるように設定する方法と、関連する重要な用語とその定義の簡単な概要を説明しています: Launching EC2 Windows instances up to 65% faster on AWS。

リソースのコスト

Windows AMI を EC2 Fast Launch 対応に設定するのにサービス料金はかかりません。ただし、Amazon EC2 が使用する、すべての基盤となる AWS リソースには標準価格が適用されます。関連するリソースコストとその管理方法の詳細については、「EC2 Fast Launch の基盤となるリソースにかかるコストを管理する」を参照してください。

内容

- 重要な用語
- Windows 用に EC2 Fast Launch を設定するための前提条件
- Amazon EC2 Windows Server AMI 向けに EC2 Fast Launch 設定を構成する
- EC2 Fast Launch が有効になっている AMI を表示する
- EC2 Fast Launch の基盤となるリソースにかかるコストを管理する
- EC2 Fast Launch のモニタリング
- EC2 Fast Launch でのサービスにリンクしたロール
- Windows EC2 Fast Launch のトラブルシューティング

重要な用語

EC2 Fast Launch 機能では、次の重要な用語を使用します。

事前プロビジョニングされたスナップショット

EC2 Fast Launch が有効になっている Windows AMI から起動し、その後の Windows 起動手順を完了し、必要に応じて再起動するインスタンスのスナップショット。

- Sysprep 専門
- Windows Out of Box Experience (OOBE)

これらの手順が完了すると、EC2 Fast Launch はインスタンスを停止し、後ほど AMI からの高速起動で使用するスナップショットを設定に基づいて作成します。

起動頻度

指定された期間内に Amazon EC2 が起動できる、事前プロビジョニングされたスナップショットの数を制御します。AMI に対して EC2 Fast Launch を有効にする場合、Amazon EC2 は事前プロビジョニング済みのスナップショットの初期セットをバックグラウンドで作成します。例えば、起動頻度が 1 時間に 5 回と設定されている場合 (デフォルト)、EC2 Fast Launch は、5 つの事前プロビジョニングされたスナップショットから成る初期セットを作成します。

Amazon EC2 が、EC2 Fast Launch が有効になっている AMI からインスタンスを起動した場合、事前プロビジョニング済みのスナップショットのうちの 1 つを使って起動時間が短縮されます。スナップショットが使用されると、起動頻度で指定された数まで、スナップショットが自動的に補充されます。

例えば、特別なイベントの間に AMI から起動されるインスタンスの数が急増することが予想される場合は、事前に起動頻度を増やして、必要な追加のインスタンスをカバーできます。起動レートが通常に戻ると、頻度を元に戻すことができます。

予想よりも多くの起動が発生した場合、利用可能な事前プロビジョニングされたスナップショットを使い切ることがあります。これによって起動が失敗することはありません。ただし、スナップショットが補充されるまで、一部のインスタンスが標準の起動プロセスを経ることがあります。

ターゲットリソース数

EC2 Fast Launch が有効になっている Amazon EC2 Windows Server AMI で手元に保持される、事前プロビジョニングされたスナップショットの数。

最大並列起動

Amazon EC2 が同時に起動できるインスタンスの数を制御して、EC2 Fast Launch 用に事前プロビジョニングされたスナップショットを作成します。ターゲットリソース数が設定した最大並列起動数よりも多い場合、Amazon EC2 は、スナップショットの作成を開始するための [最大並列起動] で指定された数のインスタンスを起動します。これらのインスタンスがプロセスを完了すると、Amazon EC2 はスナップショットを取得してインスタンスを停止します。その後、使用可能なスナップショットの総数がターゲットリソース数に達するまで、さらにインスタンスを起動し続けます。[最大並列起動] の値は 6 以上でなければなりません。

Windows 用に EC2 Fast Launch を設定するための前提条件

EC2 Fast Launch を設定する前に、AWS アカウント 内の AMI のスナップショットを作成するのに必要な次の前提条件を満たしていることを確認してください。

- EC2 Fast Launch を設定するときにカスタム起動テンプレートを指定すると、サービスはその起動テンプレートに定義されている VPC とその他設定を使用します。詳細については、「EC2 Fast Launch を設定するときに起動テンプレートを使用する」を参照してください。
- カスタム起動テンプレートを使用して設定を行わない場合は、EC2 Fast Launch を有効にする前に、EC2FastLaunchFullAccess ポリシーを現在の IAM ロールにアタッチする必要があります。その後、サービスが AWS アカウントにある次のリソースを使用して CloudFormation スタックを自動的に作成します。
  - 仮想プライベートクラウド (VPC)
  - 複数のアベイラビリティーゾーンにまたがるプライベートサブネット
  - Instance Metadata Service Version 2 (IMDSv2) で設定された起動テンプレート
  - インバウンドルールまたはアウトバウンドルールがないセキュリティグループ
- プライベート EC2 Fast Launch AMI は、ユーザーデータスクリプトの実行をサポートしている必要があります。
- AMI に対して EC2 Fast Launch を設定するには、シャットダウンオプションで Sysprep を使用して AMI を作成する必要があります。EC2 Fast Launch 機能は現在、実行中のインスタンスから作成された AMI をサポートしていません。

  Sysprep を使用して AMI を作成するには、「Windows Sysprep を使用して Amazon EC2 AMI を作成する」を参照してください。
- 暗号化にカスタマーマネージドキーを使用する暗号化された AMI に対して EC2 Fast Launch を有効にするには、EC2 Fast Launch のサービスにリンクされたロールに CMK を使用するアクセス許

可を付与する必要があります。詳細については、「the section called “カスタマーマネージドキーへのアクセス”」を参照してください。

- AWS アカウント内のすべての AMI における [最大並列起動] のデフォルトクォータは 1 リージョンあたり 40 です。アカウントでの Service Quotas の増加は、次のようにリクエストできます。

  1. Service Quotas のコンソールを開きます。https://console.aws.amazon.com/servicequotas/
  2. ナビゲーションペインで、AWS のサービス を選択します。
  3. 検索バーに **`EC2 Fast Launch`** と入力し、結果を選択します。
  4. [並列インスタンスの起動] のリンクを選択して、サービスクォータの詳細ページを開きます。
  5. [アカウントレベルでの引き上げをリクエスト] をクリックします。

  詳細については「Service Quotas ユーザーガイド」の「クォータの引き上げのリクエスト」を参照してください。

Amazon EC2 Windows Server AMI 向けに EC2 Fast Launch 設定を構成する

EC2 Fast Launch は、所有している Windows AMI に対して、または、AWS マネジメントコンソール、API、SDK、CloudFormation、または AWS Command Line Interface AWS CLIから共有した AMI に対して設定できます。EC2 Fast Launch を設定する前に、対象の AMI が、事前プロビジョニングのスナップショットを作成するのに必要な、すべての前提条件を満たしていることを確認してください。詳細については、「Windows 用に EC2 Fast Launch を設定するための前提条件」を参照してください。

Windows インスタンスの高速起動を有効にすると、Amazon EC2 は暗号化された AMI のアクセス許可を含め、指定された AMI および起動テンプレート (提供されている場合) からインスタンスを起動するために必要なアクセス許可があることをチェックします。インスタンスの起動プロセス中のエラーを防ぐため、サービスで EC2 Fast Launch を有効にする前にアクセス許可の検証が行われます。必要なアクセス許可がない場合、サービスでエラーが返され、EC2 Fast Launch が有効になりません。

EC2 Fast Launch を EC2 Image Builder と統合すると、EC2 Fast Launch が有効になっているカスタムイメージを作成しやすくなります。詳細については、「EC2 Image Builder ユーザーガイド」の「Create distribution settings for a Windows AMI with EC2 Fast Launch enabled (AWS CLI)」を参照してください。

EC2 Fast Launch を有効にする

これらの設定を変更する前に、AMI と、これを実行するリージョンが、Windows 用に EC2 Fast Launch を設定するための前提条件 のすべてを満たしていることを確認します。

Console

EC2 Fast Launch を有効にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインの [Images] (イメージ) で、[AMI] (AMI) を選択してください。
3. [Name] (名前) の横にあるチェックボックスをオンにして、更新する AMI を選択してください。
4. AMI のリストの上にある [アクション] メニューから、[高速起動の設定] を選択してください。この結果 [高速起動を設定] ページが開くため、ここで EC2 Fast Launch の設定を構成します。
5. 事前プロビジョニングされたスナップショットの使用を開始して Windows AMI からインスタンスをより速く起動するには、[Windows に高速起動を有効化] チェックボックスをオンにします。
6. [Set anticipated launch frequency] (予想される起動頻度の設定) ドロップダウンリストから、予想されるインスタンス起動ボリュームに対応できるように作成および維持されているスナップショット数を、指定する値を選択してください。
7. 変更が完了したら、[Save changes] (変更を保存) を選択してください。

> **Note**
>
> VPC を指定する、または IMDSv2 のメタデータ設定を構成するための起動テンプレートを使用する必要がある場合は、「EC2 Fast Launch を設定するときに起動テンプレートを使用する」を参照してください。

AWS CLI

EC2 Fast Launch を有効にするには

次の enable-fast-launch コマンドを使用して、指定した AMI に対して EC2 Fast Launch を有効にし、事前プロビジョニングのために 6 つの平行インスタンスを起動します。

```
aws ec2 enable-fast-launch \
    --image-id ami-0abcdef1234567890  \
    --max-parallel-launches 6 \
    --resource-type snapshot
```

以下は出力の例です。

```
{
 "ImageId": "ami-0abcdef1234567890",
 "ResourceType": "snapshot",
 "SnapshotConfiguration": {
     "TargetResourceCount": 10
 },
 "LaunchTemplate": {},
 "MaxParallelLaunches": 6,
 "OwnerId": "0123456789123",
 "State": "enabling",
 "StateTransitionReason": "Client.UserInitiated",
 "StateTransitionTime": "2022-01-27T22:16:03.199000+00:00"
}
```

PowerShell

EC2 Fast Launch を有効にするには

Enable-EC2FastLaunch コマンドレットを使用して、指定した AMI で EC2 Fast Launch を有効にし、事前プロビジョニングのために 6 つの平行インスタンスを起動します。

```
Enable-EC2FastLaunch `
 -ImageId ami-0abcdef1234567890 `
 -MaxParallelLaunch 6 `
 -Region us-west-2 `
 -ResourceType snapshot
```

以下は出力の例です。

```
ImageId               : ami-0abcdef1234567890
 LaunchTemplate        :
 MaxParallelLaunches   : 6
 OwnerId               : 0123456789123
 ResourceType          : snapshot
 SnapshotConfiguration : Amazon.EC2.Model.FastLaunchSnapshotConfigurationResponse
```

```
State                 : enabling
StateTransitionReason : Client.UserInitiated
StateTransitionTime   : 2/25/2022 12:24:11 PM
```

EC2 Fast Launch を無効にする

これらの設定を変更する前に、AMI と、これを実行するリージョンが、Windows 用に EC2 Fast Launch を設定するための前提条件 のすべてを満たしていることを確認します。

Console

EC2 Fast Launch を無効にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインの [Images] (イメージ) で、[AMI] (AMI) を選択してください。
3. [Name] (名前) の横にあるチェックボックスをオンにして、更新する AMI を選択してください。
4. AMI のリストの上にある [アクション] メニューから、[高速起動の設定] を選択してください。この結果 [高速起動を設定] ページが開くため、ここで EC2 Fast Launch の設定を構成します。
5. [Windows に高速起動を有効化] チェックボックスをオフにし、EC2 Fast Launch を無効にして事前プロビジョニングされたスナップショットを削除します。その結果、AMI は今後、各インスタンスに対して標準の起動プロセスを使用するようになります。

   > **Note**
   >
   > Windows イメージの最適化を無効にすると、既存の事前プロビジョニングされたスナップショットが自動的に削除されます。この機能を再度使用するには、この手順を完了する必要があります。

6. 変更が完了したら、[Save changes] (変更を保存) を選択してください。

AWS CLI

EC2 Fast Launch を無効にするには

次の disable-fast-launch コマンドを使用して、指定した AMI で EC2 Fast Launch を無効にし、事前プロビジョニングされた既存のスナップショットをクリーンアップします。

```
aws ec2 disable-fast-launch --image-id ami-01234567890abcedf
```

以下は出力の例です。

```
{
    "ImageId": "ami-01234567890abcedf",
    "ResourceType": "snapshot",
    "SnapshotConfiguration": {},
    "LaunchTemplate": {
        "LaunchTemplateId": "lt-01234567890abcedf",
        "LaunchTemplateName": "EC2FastLaunchDefaultResourceCreation-
a8c6215d-94e6-441b-9272-dbd1f87b07e2",
        "Version": "1"
    },
    "MaxParallelLaunches": 6,
    "OwnerId": "0123456789123",
    "State": "disabling",
    "StateTransitionReason": "Client.UserInitiated",
    "StateTransitionTime": "2022-01-27T22:47:29.265000+00:00"
}
```

PowerShell

EC2 Fast Launch を無効にするには

Disable-EC2FastLaunch コマンドレットを使用して、指定した AMI で EC2 Fast Launch を無効にし、事前プロビジョニングされた既存のスナップショットをクリーンアップします。

```
Disable-EC2FastLaunch -ImageId ami-0abcdef1234567890
```

以下は出力の例です。

```
ImageId               : ami-0abcdef1234567890
 LaunchTemplate        :
 Amazon.EC2.Model.FastLaunchLaunchTemplateSpecificationResponse
 MaxParallelLaunches   : 6
 OwnerId               : 0123456789123
 ResourceType          : snapshot
 SnapshotConfiguration :
 State                 : disabling
 StateTransitionReason : Client.UserInitiated
```

```
StateTransitionTime   : 2/25/2022 1:10:08 PM
```

EC2 Fast Launch を設定するときに起動テンプレートを使用する

起動テンプレートを使用すると、そのテンプレートからインスタンスを起動するたびに Amazon EC2 が使用する、一連の起動パラメータを設定することができます。指定できるのは、基本イメージに使用される AMI、インスタンスタイプ、ストレージ、ネットワーク設定などです。

起動テンプレートはオプションです。ただし、高速起動を設定するときに Windows AMI で起動テンプレートを使用する必要がある、次の特定の場合を除きます。

- Windows AMI 用の既存の VPC を指定するには起動テンプレートを使用する必要があります。AWS アカウントのデフォルト VPC を使用する場合は必要ありません。
- アカウントに、Amazon EC2 インスタンスに IMDSv2 を強制するポリシーが含まれている場合、IMDSv2 を強制するメタデータ設定を指定する、起動テンプレートを作成する必要があります。

  AWS CLI で enable-fast-launch コマンドを実行するとき、または EnableFastLaunch API アクションを呼び出すとき、EC2 コンソールのメタデータ設定を含む起動テンプレートを使用します。

起動テンプレートを使用する場合、Amazon EC2 EC2 Fast Launch では以下の設定はサポートされていません。EC2 Fast Launch の起動テンプレートを使用する場合は、以下を指定しないでください。

- ユーザーデータスクリプト
- 終了保護
- 無効なメタデータ
- スポットオプション
- インスタンスを終了させるシャットダウン動作
- ネットワークインターフェイス、Elastic Graphics、スポットインスタンスのリクエスト用のリソースタグ

VPC を指定する

ステップ 1: 起動テンプレートを作成する

Windows インスタンスの次の詳細を指定する起動テンプレートを作成します。

- VPC サブネット。
- `t3.xlarge` のインスタンスタイプ。

詳細については、「Amazon EC2 起動テンプレートを作成する」を参照してください。

ステップ 2: EC2 Fast Launch AMI に起動テンプレートを指定する

Console

EC2 Fast Launch AMI に起動テンプレートを指定するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインの [Images] (イメージ) で、[AMI] (AMI) を選択してください。
3. [Name] (名前) の横にあるチェックボックスをオンにして、更新する AMI を選択してください。
4. AMI のリストの上にある [アクション] メニューから、[高速起動の設定] を選択してください。この結果 [高速起動を設定] ページが開くため、ここで EC2 Fast Launch の設定を構成します。
5. [Launch template] (起動テンプレート) ボックスが、フィルター済みの検索を実行し、入力したテキストにマッチする起動テンプレートを、現在のリージョンのアカウントで特定します。ボックスで、起動テンプレートの名前または ID の、全部または一部を指定し、一致する起動テンプレートの一覧を表示します。例えば、ボックスに `fast` と入力すると、Amazon EC2 は、現在のリージョンのアカウントで、名前に「fast」を含む起動テンプレートをすべて検索します。

   起動テンプレートを新規作成するときは、[Create launch template] (起動テンプレートを作成) を選択してください。
6. 起動テンプレートを選択すると、Amazon EC2 は、[Source template version] (ソーステンプレートのバージョン) ボックスにそのテンプレートのデフォルトバージョンを表示します。別のバージョンを指定するときは、置き換えるデフォルトのバージョンを強調表示し、ボックスに必要なバージョンの番号を入力してください。
7. 変更が完了したら、[Save changes] (変更を保存) を選択してください。

AWS CLI

EC2 Fast Launch AMI に起動テンプレートを指定するには

enable-fast-launch コマンドを --launch-template オプションとともに使用して、起動テンプレートの名前または ID を指定します。

```
--launch-template LaunchTemplateName=my-launch-template
```

PowerShell

EC2 Fast Launch AMI に起動テンプレートを指定するには

Enable-EC2FastLaunch コマンドレットを -LaunchTemplate_LaunchTemplateId または -LaunchTemplate_LaunchTemplateName パラメータと共に使用します。

```
-LaunchTemplate_LaunchTemplateName my-launch-template
```

EC2 起動テンプレートの詳細については、「Amazon EC2 起動テンプレートにインスタンス起動パラメータを保存する」(起動テンプレートからのインスタンスの起動) を参照してください。

EC2 Fast Launch が有効になっている AMI を表示する

AWS CLI の describe-fast-launch-images コマンド、または Get-EC2FastLaunchImage Tools for PowerShell コマンドレットを使用して、EC2 Fast Launch が有効になっている AMI の詳細を取得できます。

Amazon EC2 は、結果で返される各 Windows AMI について、次の詳細を提供します。

- EC2 Fast Launch が有効になっている AMI のイメージ ID。
- 関連付けられた Windows AMI の事前プロビジョニングに使用されるリソースタイプ。サポートされている値は snapshot です。
- スナップショット設定。スナップショットを使用して、関連付けられた Windows AMI の事前プロビジョニングを設定するパラメータのグループです。
- 事前プロビジョニング済みのスナップショットから Windows インスタンスを起動するときに、関連付けられた AMI が使用する起動テンプレートの ID、名前、バージョンなどの、起動テンプレートに関する情報。
- リソースを作成するために同時に起動できるインスタンスの最大数。
- 関連付けられた AMI の所有者 ID。これは、共有された AMI に対しては追加されていません。
- 関連付けられた AMI に対する EC2 Fast Launch の現在の状態。サポートされる値を次に示します。enabling | enabling-failed | enabled | enabled-failed | disabling | disabling-failed。

> **Note**
>
> 現在の状態は、EC2 コンソールの [Manage image optimization] (イメージ最適化を管理) ページにも、[Image optimization state] (イメージ最適化の状態) として表示されます。

- 関連付けられた AMI に対する EC2 Fast Launch が現在の状態に変更された理由。
- 関連付けられた AMI に対する EC2 Fast Launch が現在の状態に変更された時刻。

AWS CLI

EC2 Fast Launch 用に設定された AMI を検索するには

次の describe-fast-launch-images コマンドを使用して、EC2 Fast Launch 用に設定されたアカウント内の、各 AMI の詳細を記述します。この例では、アカウント内の AMI が 1 つだけ EC2 Fast Launch 対応に設定されています。

```
aws ec2 describe-fast-launch-images
```

以下は出力例です。

```
{
    "FastLaunchImages": [
        {
            "ImageId": "ami-01234567890abcedf",
            "ResourceType": "snapshot",
            "SnapshotConfiguration": {},
            "LaunchTemplate": {
                "LaunchTemplateId": "lt-01234567890abcedf",
                "LaunchTemplateName": "EC2FastLaunchDefaultResourceCreation-
a8c6215d-94e6-441b-9272-dbd1f87b07e2",
                "Version": "1"
            },
            "MaxParallelLaunches": 6,
            "OwnerId": "0123456789123",
            "State": "enabled",
            "StateTransitionReason": "Client.UserInitiated",
            "StateTransitionTime": "2022-01-27T22:20:06.552000+00:00"
        }
    ]
}
```

PowerShell

EC2 Fast Launch 用に設定された AMI を検索するには

次の Get-EC2FastLaunchImage コマンドレットを使用して、EC2 Fast Launch 用に設定されたアカウント内の、各 AMI の詳細を記述します。この例では、アカウント内の AMI が 1 つだけ EC2 Fast Launch 対応に設定されています。

```
Get-EC2FastLaunchImage -ImageId ami-0abcdef1234567890
```

以下は出力例です。

```
ImageId               : ami-0abcdef1234567890
LaunchTemplate        :
 Amazon.EC2.Model.FastLaunchLaunchTemplateSpecificationResponse
MaxParallelLaunches   : 6
OwnerId               : 012345678912
ResourceType          : snapshot
SnapshotConfiguration :
State                 : enabled
StateTransitionReason : Client.UserInitiated
StateTransitionTime   : 2/25/2022 12:54:43 PM
```

EC2 Fast Launch の基盤となるリソースにかかるコストを管理する

Windows AMI を EC2 Fast Launch 対応に設定するのにサービス料金はかかりません。ただし、Amazon EC2 Windows AMI で EC2 Fast Launch を有効にすると、Amazon EC2 が事前プロビジョニングされたスナップショットを準備および保存するために使用する基本の AWS リソースには、標準料金が適用されます。コスト配分タグを設定すると、EC2 Fast Launch リソースに関連するコストの追跡と管理がしやすくなります。コスト配分タグを設定する方法の詳細については、「請求の EC2 Fast Launch コストを追跡する」を参照してください。

次の例は、EC2 Fast Launch スナップショットのコストに関連するコストがどのように配分されるかを示しています。

シナリオ例: AtoZ Example 社は、50 GiB の EBS ルートボリュームを備える Windows AMI を使用しています。同社は AMI に対し EC2 Fast Launch を有効化していて、ターゲットリソース数を 5 に設定しています。自社の AMI で EC2 Fast Launch を 1 か月間使用すると、コストは約 5.00 USD になります。その内訳は次のとおりです。

1. AtoZ Example 社が EC2 Fast Launch を有効にすると、Amazon EC2 は 5 つのスモールインスタンスを起動します。各インスタンスは Sysprep および OOBE Windows の起動ステップを通じて実行され、必要に応じて再起動します。これには、インスタンスごとに数分かかります (時間はリージョンまたはアベイラビリティーゾーン (AZ) のビジー状態、および AMI のサイズに応じて異なります) 。

   コスト

   - インスタンスのランタイムコスト (該当する場合は最小ランタイム): 5 つのインスタンス
   - ボリュームコスト: 5 つの EBS ルートボリューム

2. 事前プロビジョニングの処理が完了すると、Amazon EC2 はインスタンスのスナップショットを取得します。これは Amazon S3 に保存されます。スナップショットは通常、起動に使用される前に 4 ~ 8 時間保存されます。その場合、コストはスナップショットあたり約 0.02 ~ 0.05 USD です。

   コスト

   - スナップショットストレージ (Amazon S3): 5 つのスナップショット

3. Amazon EC2 は、スナップショットを取得するとインスタンスを停止します。その時点から、インスタンスのコストは発生しなくなります。ただし、EBS ボリュームコストでは引き続きコストが発生します。

   コスト

   - EBS ボリューム: 関連する EBS ルートボリュームのコストは、引き続き発生します。

> **Note**
>
> こちらに示すコストは一例です。コストは、ご使用の AMI 構成および料金プランに応じて異なります。

請求の EC2 Fast Launch コストを追跡する

コスト配分タグは、EC2 Fast Launch に関連するコストを反映するように AWS 請求を整理するのに役立ちます。EC2 Fast Launch 用に、事前プロビジョニング済みのスナップショットを準備および保存するときに、Amazon EC2 が作成するリソースに追加する次のタグを使用できます。

タグキー: `CreatedBy`、値: `EC2 Fast Launch`

Billing and Cost Management コンソールでタグをアクティブ化し、詳細な請求レポートを設定すると、レポートに [user:CreatedBy] 列が表示されます。この列には、すべてのサービスの値が含まれます。ただし、CSV ファイルをダウンロードした場合は、データをスプレッドシートにインポートして、値で EC2 Fast Launch をフィルタリングできます。この情報は、タグがアクティブ化されたときに AWS Cost and Usage Report にも表示されます。

ステップ 1: ユーザー定義のコスト配分タグを有効にする

コストレポートにリソースタグを含めるには、まず、Billing and Cost Management コンソールでタグをアクティブ化する必要があります。詳細については、AWS Billing and Cost Management ユーザーガイドの「ユーザー定義のコスト配分タグのアクティブ化」を参照してください。

> **Note**
>
> アクティベーションには最長で 24 時間かかることがあります。

ステップ 2: コストレポートを設定する

コストレポートを既に設定している場合は、アクティベーションの完了後、次にレポートを実行したときにタグの列が表示されます。コストレポートを初めて設定する場合は、次のいずれかを選択します。

- 「AWS Billing and Cost Management ユーザーガイド」の「月次コスト配分レポートの設定」を参照してください。
- 「AWS Cost and Usage Report ユーザーガイド」の「Creating Cost and Usage Reports」(コストと使用状況レポートの作成) を参照してください。

> **Note**
>
> AWS から S3 バケットへのレポートの配信を開始するまで、最大 24 時間かかる場合があります。

EC2 Fast Launch は、所有している Windows AMI に対して、または、Amazon EC2 コンソール、API、SDK、CloudFormation、または AWS CLI の ec2 コマンドから共有した AMI に対して設定できます。以下のセクションでは、Amazon EC2 コンソールおよび AWS CLI の設定手順について説明します。

EC2 Image Builder で EC2 Fast Launch 用に構成されたカスタム Windows AMI を作成することもできます。詳細については、「EC2 Fast Launch が有効になっている Windows AMI のディストリビューション設定を作成する (AWS CLI)」を参照してください。

EC2 Fast Launch のモニタリング

このセクションでは、EC2 Fast Launch が有効になっているアカウントで Amazon EC2 Windows Server AMI をモニタリングする方法について説明します。

EventBridge を使用して、EC2 Fast Launch の状態の変更をモニタリングする

EC2 Fast Launch が有効になっている Windows AMI の状態が変わると、Amazon EC2 は `EC2 Fast Launch State-change Notification` イベントを生成します。次に、Amazon EC2 は Amazon EventBridge (旧 Amazon CloudWatch Events) に送信します。

状態変更イベントに応じて、1 つ以上のアクションをトリガーする EventBridge ルールを作成できます。例えば、EC2 Fast Launch が有効になったことを検出し、次のアクションを実行する、EventBridge ルールを作成できます。

- サブスクライバーに通知するメッセージを Amazon SNS トピックに送信します。
- 何らかのアクションを実行する、Lambda 関数を呼び出します。
- 状態変更データを Amazon Data Firehose に送信し、分析します。

EventBridge ルールの作成の詳細については、Amazon EventBridge ユーザーガイドの「イベントに反応する Amazon EventBridge ルールの作成」を参照してください 。

状態変更イベント

EC2 Fast Launch 機能では、ベストエフォートベースで JSON 形式の状態変更イベントを発行します。Amazon EC2 は、ほぼリアルタイムにイベントを EventBridge に送信します。このセクションでは、イベントフィールドについて説明し、イベントのフォーマットの例を示します。

`EC2 Fast Launch State-change Notification`

imageId

EC2 Fast Launch の状態が変更された AMI を特定します。

resourceType

事前プロビジョニングに使用するリソースのタイプ。サポートされている値は `snapshot` です。デフォルト値は `snapshot` です。

state

指定された AMI に対する EC2 Fast Launch 機能の現在の状態。有効な値には次のようなものがあります。

- enabling – 対象 AMI に対する EC2 Fast Launch 機能が有効になり、Amazon EC2 が事前プロビジョニングプロセス用のスナップショットの作成を開始します。
- enabling-failed – AMI に対して EC2 Fast Launch を初めて有効にしたときに、問題が発生して事前プロビジョニングプロセスが失敗しました。これは、事前プロビジョニングのプロセス中いつでも発生する可能性があります。
- 有効 – EC2 Fast Launch 機能が有効になっています。Amazon EC2 が、新しく有効になった EC2 Fast Launch AMI に対して事前プロビジョニングされた最初のスナップショットを作成するとすぐに、この状態は `enabled` に変わります。AMI が既に有効になっており、再度事前プロビジョニングを行うと、状態の変更が直ちに行われます。
- enabled-failed – この状態は、EC2 Fast Launch AMI での事前プロビジョニングプロセスが今回が初めてでない場合にのみ適用されます。これは、EC2 Fast Launch 機能を無効にしてから再度有効にした場合や、初回の事前プロビジョニングが完了した後に設定の変更やその他のエラーがあった場合に発生する可能性があります。
- disabling – AMI の所有者がその AMI に対し EC2 Fast Launch 機能を無効にし、Amazon EC2 がクリーンアッププロセスを開始しました。
- 無効 – EC2 Fast Launch 機能は無効になっています。Amazon EC2 がクリーンアッププロセスを完了するとすぐに状態は `disabled` に変わります。
- disabling-failed - 問題が発生したため、クリーンアッププロセスが失敗しました。これは事前にプロビジョニングされたスナップショットの一部がアカウントに残っている可能性があることを意味します。

StateTransitionReason

EC2 Fast Launch AMI の状態が変更された理由。

> **Note**
>
> このイベントメッセージのすべてのフィールドは必須です。

次の例は、新しく有効になった EC2 Fast Launch AMI が、最初のインスタンスを起動して事前プロビジョニングプロセスを開始したところを示しています。この時点で、状態は `enabling` で

す。Amazon EC2 が事前にプロビジョニングされた最初のスナップショットを作成すると、状態は enabled に変わります。

```
{
 "version": "0",
 "id": "01234567-0123-0123-0123-012345678901",
 "detail-type": "EC2 Fast Launch State-change Notification",
 "source": "aws.ec2",
 "account": "123456789012",
 "time": "2022-08-31T20:30:12Z",
 "region": "us-east-1",
 "resources": [
  "arn:aws:ec2:us-east-1:123456789012:image/ami-123456789012"
 ],
 "detail": {
  "imageId": "ami-123456789012",
  "resourceType": "snapshot",
  "state": "enabling",
  "stateTransitionReason": "Client.UserInitiated"
 }
}
```

CloudWatch で EC2 Fast Launch メトリクスをモニタリングする

EC2 Fast Launch が有効になっている Amazon EC2 AMI は、Amazon CloudWatch にメトリクスを送信します。AWS マネジメントコンソール、AWS CLI、または API を使用して、EC2 Fast Launch が CloudWatch に送信するメトリクスを一覧表示できます。AWS/EC2 名前空間には、次の EC2 Fast Launch メトリクスが含まれます。

メトリクス	説明
NumberOfAvailableFastLaunchSnapshots	EC2 Fast Launch が有効になっている AMI あたりの使用可能な事前プロビジョニングされたスナップショットの数。
NumberOfInstancesFastLaunched	事前プロビジョニングされたスナップショットから起動された EC2 Fast Launch が有効になっている AMI あたりのインスタンスの数。
NumberOfInstancesNotFastLaunched	起動時に使用可能な事前プロビジョニングされたスナップショットがないためにコールドブー

メトリクス	説明
	トになった、EC2 Fast Launch が有効になっている AMI あたりのインスタンスの数。
FastLaunchSnapshotUsedToRefillStartTime	Amazon EC2 が、既存のスナップショットを使用した後に別のスナップショットを作成するために、EC2 Fast Launch が有効になっている AMI から新しいイメージを起動したときのタイムスタンプ。
FastLaunchSnapshotCreationTime	Amazon EC2 がインスタンスを起動し、EC2 Fast Launch が有効になっている AMI のスナップショットを作成するのにかかった時間を測定します。

EC2 Fast Launch でのサービスにリンクしたロール

Amazon EC2 は、ユーザーに代わって他の AWS のサービスを呼び出すために必要なアクセス許可のために、サービスにリンクされたロールを使用します。サービスにリンクされたロールはAWS のサービス に直接リンクされた一意のタイプの IAM ロールです。サービスにリンクされたロールは、AWS のサービスにアクセス許可を委任するためのセキュアな方法を提供します。これは、リンクされたサービスのみが、サービスにリンクされたロールを引き受けることができるためです。Amazon EC2 がサービスリンクロールを含めた IAM ロールを使用する方法の詳細については、「Amazon EC2 の IAM ロール」を参照してください。

Amazon EC2 では、AWSServiceRoleForEC2FastLaunch という名前のサービスリンクロールを使用して、Windows AMI からのインスタンスの起動にかかる時間を短縮する、事前プロビジョニングされたスナップショットのセットを作成および管理します。

AWSServiceRoleForEC2FastLaunch によって付与されるアクセス許可

AWSServiceRoleForEC2FastLaunch サービスにリンクされたロールはその引き受け時に、以下のサービスを信頼します。

- `ec2fastlaunch.amazonaws.com`

Amazon EC2 は、EC2FastLaunchServiceRolePolicy 管理ポリシーを使用して、次のアクションを実行します。

- AWS CloudFormation – 関連する CloudFormation スタックの説明を取得することを EC2 Fast Launch に許可します。
- Amazon CloudWatch – EC2 Fast Launch に関連付けられたメトリクスデータを Amazon EC2 の名前空間にポストします。
- Amazon EC2 – EC2 Fast Launch が以下のアクションを実行するためのアクセス権が付与されます。
  - プロビジョニング手順を実行するために、EC2 Fast Launch が有効になっている Amazon EC2 Windows Server AMI からインスタンスを起動する。さらに、License Manager に関連付けられている AMI の `ec2:RunInstances` を許可するリソースパターンを指定する。
  - EC2 Fast Launch が事前プロビジョニングされたスナップショットを作成した後で、EC2 Fast Launch が起動したインスタンスを停止し、終了する。
  - EC2 Fast Launch が有効化された Amazon EC2 Windows Server AMI からインスタンスを起動するために使用されたイメージとインスタンスタイプのリソースを記述し、それらからスナップショットを作成する。
  - 起動テンプレートリソースを記述し、起動テンプレートからインスタンスを起動する。
  - インスタンス、インスタンス属性、インスタンスステータス、ボリューム、ボリューム属性について説明する。
  - ネットワークインターフェイスを記述する。
  - EC2 Fast Launch が作成したリソース (スナップショット、起動テンプレート、ボリューム、ネットワークインターフェイスなど) を削除する。
  - EC2 Fast Launch が Windows インスタンスを起動および事前プロビジョニングするために作成するリソースにタグを付け、最終的な起動プロセスが使用するスナップショットを作成する。
- Amazon EventBridge – EventBridge イベントルールを作成し、作成したルールの詳細を取得またはそれらを削除するためのアクセスを含めます。EC2 Fast Launch は、イベントルールに基づいて転送される EC2 Fast Launch イベントを受信するターゲットサービスのリストを取得し、作成したイベントルールにターゲットサービスの追加したり、ルールから削除したりすることもできます。
- IAM – `EC2FastLaunchServiceRolePolicy` サービスリンクロールを作成する、名前に `ec2fastlaunch` が含まれているインスタンスプロファイルを取得して使用する、および起動テンプレートからのインスタンスプロファイルを使用してユーザーに代わってインスタンスを起動することを EC2 Fast Launch に許可します。

- AWS KMS – グラントを作成するアクセスと、EC2 Fast Launch が作成したグラントで廃止できるものを一覧表示するためのアクセスを含めます。また、EC2 Fast Launch が作成するインスタンスにアタッチされたボリュームを暗号化または復号化するためのキーを記述または使用、およびプレーンテキストではないデータキーを生成するためのアクセスも含めます。

このポリシーのアクセス許可を確認するには、「AWS マネージドポリシーリファレンス」の「EC2FastLaunchServiceRolePolicy」を参照してください。

Amazon EC2 のマネージドポリシーの使用方法の詳細については、「Amazon EC2 の AWS マネージドポリシー」を参照してください。

サービスにリンクされたロールの作成

このサービスリンクロールを手動で作成する必要はありません。AMI に対し EC2 Fast Launch の使用を開始した際に、サービスにリンクしたロールが存在しない場合、Amazon EC2 はそのロールを作成します。

サービスにリンクしたロールがアカウントから削除された場合、別の Windows AMI に対し EC2 Fast Launch を有効にして、アカウントでロールを再作成することができます。または、現在の AMI に対して EC2 Fast Launch を無効にし、再度有効にすることもできます。ただし、機能を無効にすると、AMI ではすべての新しいインスタンスに対して標準の起動プロセスが使用され、Amazon EC2 では事前プロビジョニングされたスナップショットがすべて削除されます。事前プロビジョニングされたスナップショットがすべて削除されたら、AMI に対し EC2 Fast Launch の使用を再び有効にすることができます。

カスタマーマネージドキーへのアクセス

暗号化にカスタマーマネージドキーを使用する暗号化された AMI に対して EC2 Fast Launch を有効にするには、AWSServiceRoleForEC2FastLaunch ロールに CMK を使用するアクセス許可を付与する必要があります。これを行うには、create-grant コマンドを呼び出します。`--grantee-principal` には、アカウントの AWSServiceRoleForEC2FastLaunch ロールの ARN を指定します。`--operations` の場合、`CreateGrant` を指定します。

```
aws kms create-grant \
    --key-id arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab \
    --grantee-principal arn:aws:iam::111122223333:role/AWSServiceRoleForEC2FastLaunch \
    --operations CreateGrant
```

サービスにリンクされたロールの編集

Amazon EC2 では、AWSServiceRoleForEC2FastLaunch のサービスリンクロールを編集することはできません。サービスにリンクされたロールを作成すると、多くのエンティティによってロールが参照される可能性があるため、ロール名を変更することはできません。ただし、IAM を使用してロールの説明を編集することはできます。詳細については、「IAM ユーザーガイド」の「サービスリンクロールの編集」を参照してください。

サービスにリンクされたロールを削除

サービスリンクロールは、関連リソースをすべて削除した後でしか削除できません。この結果、Amazon EC2 リソースへのアクセス許可が誤って削除できなくなるため、EC2 Fast Launch が有効になっている Amazon EC2 Windows Server AMI に関連付けられたリソースが保護されます。

IAM コンソール、AWS CLI、または AWS API を使用して、AWSServiceRoleForEC2FastLaunch サービスリンクロールを削除します。詳細については、「IAM ユーザーガイド」の「サービスにリンクされたロールの削除」を参照してください

サポート対象のリージョン

Amazon EC2 では、Amazon EC2 サービスを利用できるすべてのリージョンで、EC2 Fast Launch のサービスにリンクしたロールがサポートされています。

Windows EC2 Fast Launch のトラブルシューティング

トラブルシューティングシナリオ

以下のシナリオは、EC2 Fast Launch を有効にしようとするときに発生する可能性がある、一般的な問題の診断と修正に役立ちます。

スナップショット作成のためにインスタンスを停止できない

説明

EC2 Fast Launch を有効にすると、事前にプロビジョニングされたスナップショットの作成に使用される一連のインスタンスをサービスが起動します。各インスタンスには、プロセスを完了するために 30 分が割り振られます。インスタンスのいずれかが正常に完了すると、サービスは AMI の Fast Launch ステータスを `Enabled` に設定します。しかし、インスタンスが割り振られた時間内にプロセスを完了できず、他のどのインスタンスもプロセスを完了していない場合は、サービスがすべてのインスタンスを終了して、AMI の Fast Launch ステータスを `enabling_failed` に設定し、以下のように Fast Launch ステータスの理由を提供します。

```
Unable to stop instance ID=i-1234567890abcdef0 for snapshot creation.
```

原因

ほとんどの場合、この問題は、実行中のインスタンスから作成された Windows AMI、または EC2 Fast Launch の前提条件のすべてを満たさない AMI のために EC2 Fast Launch を有効にしようとすることが原因で発生します。

ソリューション

使用する AMI が EC2 Fast Launch の前提条件のすべてを満たしていることを確認してください。

AMI に対して EC2 Fast Launch を設定するには、シャットダウンオプションで Sysprep を使用して AMI を作成する必要があります。詳細については、「Windows Sysprep を使用して Amazon EC2 AMI を作成する」を参照してください。

VPC 制限に到達した

説明

起動テンプレートを使用して既存の VPC を指定しておらず、アカウントにデフォルトの VPC が定義されていない場合は、EC2 Fast Launch の前提条件で説明されているとおり、VPC とその他のリソースが含まれた CloudFormation スタックをサービスが自動的に作成します。

原因

リージョンの AWS アカウントで許可されている VPC の最大数に到達しており、EC2 Fast Launch が使用する既存の VPC が指定されていません。このため、プロセスが失敗します。

ソリューション

この問題は、次のいずれかのオプションを使用して解決できます。

- クォータの引き上げをリクエストする
- 既存の VPC を指定する起動テンプレートを提供する

アカウントがリージョンごとに定義できる VPC 数の引き上げをリクエストするには、次の手順を実行します。

1. Service Quotas のコンソールを開きます。https://console.aws.amazon.com/servicequotas/

2. [サービスコンソールダッシュボード] で [Amazon Virtual Private Cloud (Amazon VPC)] を選択します。選択すると、VPC のサービスクォータが開きます。
3. `VPCs per Region` でフィルタリングして、クォータに直接移動します。
4. [リージョン当たりの VPC] を選択してから、[アカウントレベルでの引き上げをリクエスト] を選択します。

緊急のクォータリクエストがある、またはクォータ引き上げリクエストが拒否された場合は、サポートにお問い合わせください。詳細については「Service Quotas ユーザーガイド」の「クォータの引き上げのリクエスト」を参照してください。

EC2 Fast Launch を有効にするための許可が不十分

説明

起動テンプレートを指定せずに EC2 Fast Launch を初めて有効にすると、EC2 Fast Launch がサービスのデフォルトリソースを使用してサービス所有の CloudFormation スタックを作成します。しかし、IAM プリンシパル (ロールまたはユーザー) に必要な許可がない場合は、CloudFormation テンプレートのデプロイが失敗します。

ログメッセージは、次のような内容になります。

```
Can't enable EC2 Fast Launch. The IAM credentials that you are using do not have
 sufficient permissions. Attach EC2FastLaunchFullAccess in the IAM console.
```

原因

IAM ユーザーまたはロールに、EC2 Fast Launch を有効化するために必要な許可がありません。

ソリューション

EC2 Fast Launch を有効にする IAM プリンシパル (ユーザーまたはロール) に `EC2FastLaunchFullAccess` ポリシーがアタッチされていることを確認してください。この AWS マネージドポリシーは、すべての EC2 Fast Launch リソースへのフルアクセスを許可します。このポリシーの許可を確認するには、「AWS マネージドポリシーリファレンス」で「EC2FastLaunchFullAccess」ポリシーを参照してください。

Amazon EC2 インスタンスの Windows 管理者パスワードを変更する

AWS Windows AMI からインスタンスを起動する場合、プリインストールされている起動エージェントはデフォルトのパスワードを次のように設定します。

- Windows Server 2022 以降の場合、EC2Launch v2 によりデフォルトのパスワードが生成されます。
- Windows Server 2016 および 2019 の場合、EC2Launch エージェントによりデフォルトのパスワードが生成されます。
- Windows Server 2012 R2 以前の場合、EC2Config サービス によりデフォルトのパスワードが生成されます。

> **Note**
>
> Windows Server 2016 以降の AMI では、ローカル管理者の `Password never expires` は無効になっています。Windows Server 2016 より前の AMI バージョンでは、ローカル管理者の `Password never expires` は有効です。

接続後の管理者パスワードの変更

初めてインスタンスに接続するときは、管理者パスワードをデフォルトの値から変更することをお勧めします。Windows インスタンスの管理者パスワードを変更するには、次の手順を使用します。

> **Important**
>
> 新しいパスワードを安全な場所に保存します。Amazon EC2 コンソールを使用して新しいパスワードを取得することはできません。コンソールでは、デフォルトのパスワードのみ取得できます。パスワードを変更した後、デフォルトのパスワードを使用してインスタンスに接続しようとすると、「お使いの資格情報は機能しませんでした」というエラーになります。

ローカル管理者パスワードを変更するには

1. インスタンスに接続して、コマンドプロンプトを開きます。
2. 次のコマンドを実行します。新しいパスワードに特殊文字が含まれている場合は、そのパスワードを二重引用符で囲みます。

```
net user Administrator "new_password"
```

3. 新しいパスワードを安全な場所に保存します。

紛失または期限切れのパスワードの変更

パスワードを忘れた場合、またはパスワードの有効期限が切れた場合は、新しいパスワードを生成できます。パスワードのリセット手順については、Amazon EC2 Windows インスタンスの Windows 管理者パスワードをリセットするを参照してください。

Amazon EC2 Windows インスタンスにオプションの Windows Server コンポーネントを追加する

オプションコンポーネントにアクセスし、インストールするには、Windows Server のバージョンに合った正しい EBS スナップショットを探して、スナップショットからボリュームを作成し、インスタンスにボリュームをアタッチします。

開始する前に

インスタンスのインスタンス ID とアベイラビリティーゾーンを取得するには、AWS マネジメントコンソール またはコマンドラインツールを使用します。インスタンスと同じアベイラビリティーゾーンに新しい EBS ボリュームを作成する必要があります。

インスタンスに Windows Server コンポーネントを追加するには、次のいずれかの手順を実行します。

Console

インスタンスに Windows コンポーネントを追加するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Snapshots] を選択します。
3. [Filter] (フィルター) バーで、[Public snapshots] (パブリックスナップショット) を選択します。
4. [Owner Alias] (所有者のエイリアス) フィルターを追加して、[amazon] を選択します。
5. [説明] フィルタを追加して、**`Windows`**と入力します。
6. Enter キーを押します。
7. システムアーキテクチャと言語設定に一致するスナップショットを選択します。例えば、インスタンスで Windows Server 2019 を実行している場合は、[Windows 2019 English Installation Media] を選択します。
8. [Actions] (アクション)、[Create volume from snapshot] (スナップショットからボリュームを作成する) の順に選択します。

9. [アベイラビリティーゾーン] で、Windows インスタンスに一致するアベイラビリティーゾーンを選択します。[Add tag] (タグの追加) を選択し、タグキーの **Name** と、タグ値のわかりやすい名前を入力します。[Create volume] (ボリュームの作成) を選択します。
10. [Successfully created volume] (ボリュームが正常に作成されました) というメッセージ (緑色のバナー) で、先ほど作成したボリュームを選択します。
11. [Actions] (アクション)、[Attach volume] (ボリュームのアタッチ) の順に選択します。
12. [Instance] (インスタンス) から、インスタンス ID を選択します。
13. [Device name] (デバイス名) で、アタッチメントのデバイスの名前を入力します。デバイス名に役立つヘルプが必要な場合は、「Amazon EC2 インスタンス上のボリュームのデバイス名」を参照してください。
14. [ボリュームのアタッチ] を選択します。
15. インスタンスに接続してボリュームを使用できるようにします。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームを使用できるようにする」を参照してください。

> **Important**
>
> ボリュームを初期化しないでください。

16. [コントロールパネル] を開き、[プログラムと機能] を選択します。[Turn Windows features on or off] を選択します。インストールメディアの指定を求められたら、インストールメディアを使用して EBS ボリュームを指定します。
17. (オプション) インストールメディアを終了したら、ボリュームをデタッチできます。デタッチした後で、ボリュームを削除できます。

AWS CLI

インスタンスに Windows コンポーネントを追加するには

1. describe-snapshots コマンドを使用して、`owner-ids` パラメータと `description` フィルタを適用し、使用できるインストールメディアのスナップショットのリストを取得します。

```
aws ec2 describe-snapshots \
    --owner-ids amazon \
    --filters Name=description,Values=Windows*
```

2. 返される出力で、システムアーキテクチャと言語設定に一致するスナップショットの ID を書き留めます。次に例を示します。

```
{
    "Snapshots": [
    ...
        {
            "OwnerAlias": "amazon",
            "Description": "Windows 2019 English Installation Media",
            "Encrypted": false,
            "VolumeId": "vol-be5eafcb",
            "State": "completed",
            "VolumeSize": 6,
            "Progress": "100%",
            "StartTime": "2019-10-25T20:00:47.000Z",
            "SnapshotId": "snap-22da283e",
            "OwnerId": "123456789012"
        },
    ...
   ]
}
```

3. create-volume コマンドを使用して、スナップショットからボリュームを作成します。インスタンスと同じアベイラビリティーゾーンを指定します。

```
aws ec2 create-volume \
    --snapshot-id snap-0abcdef1234567890 \
    --volume-type gp2 \
    --availability-zone us-east-1a
```

4. 出力で、ボリューム ID を書き留めます。

```
{
    "AvailabilityZone": "us-east-1a",
    "Encrypted": false,
    "VolumeType": "gp2",
    "VolumeId": "vol-01234567890abcdef",
    "State": "creating",
    "Iops": 100,
    "SnapshotId": "snap-0abcdef1234567890",
    "CreateTime": "2017-04-18T10:33:10.940Z",
    "Size": 6
```

```
}
```

5. attach-volume コマンドを使用して、インスタンスにボリュームをアタッチします。

```
aws ec2 attach-volume \
    --volume-id vol-0c98b37f30bcbc290 \
    --instance-id i-01474ef662b89480 \
    --device xvdg
```

6. インスタンスに接続してボリュームを使用できるようにします。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームを使用できるようにする」を参照してください。

> **Important**
>
> ボリュームを初期化しないでください。

7. [コントロールパネル] を開き、[プログラムと機能] を選択します。[Turn Windows features on or off] を選択します。インストールメディアの指定を求められたら、インストールメディアを使用して EBS ボリュームを指定します。
8. (オプション) インストールメディアが完了したら、detach-volume コマンドを使用してインスタンスからボリュームをデタッチします。ボリュームをデタッチした後、delete-volume コマンドを使用してボリュームを削除できます。

PowerShell

インスタンスに Windows コンポーネントを追加するには

1. Get-EC2Snapshot コマンドを使用して、Owner フィルタおよび description フィルタを適用し、使用できるインストールメディアのスナップショットのリストを取得します。

```
Get-EC2Snapshot `
    -Owner amazon `
    -Filter @{ Name="description"; Values="Windows*" }
```

2. 返される出力で、システムアーキテクチャと言語設定に一致するスナップショットの ID を書き留めます。次に例を示します。

```
...
DataEncryptionKeyId :
```

```
Description         : Windows 2019 English Installation Media
Encrypted           : False
KmsKeyId            :
OwnerAlias          : amazon
OwnerId             : 123456789012
Progress            : 100%
SnapshotId          : snap-0abcdef1234567890
StartTime           : 10/25/2019 8:00:47 PM
State               : completed
StateMessage        :
Tags                : {}
VolumeId            : vol-01234567890abcdef
VolumeSize          : 6
...
```

3. New-EC2Volume コマンドレットを使用して、スナップショットからボリュームを作成します。インスタンスと同じアベイラビリティーゾーンを指定します。

```
New-EC2Volume `
    -AvailabilityZone us-east-1a `
    -VolumeType gp2 `
    -SnapshotId snap-0abcdef1234567890
```

4. 出力で、ボリューム ID を書き留めます。

```
Attachments      : {}
AvailabilityZone : us-east-1a
CreateTime       : 4/18/2017 10:50:25 AM
Encrypted        : False
Iops             : 100
KmsKeyId         :
Size             : 6
SnapshotId       : snap-0abcdef1234567890
State            : creating
Tags             : {}
VolumeId         : vol-01234567890abcdef
VolumeType       : gp2
```

5. Add-EC2Volume コマンドレットを使用して、インスタンスにこのボリュームをアタッチします。

```
Add-EC2Volume `
    -InstanceId i-1234567890abcdef0 `
```

```
    -VolumeId vol-01234567890abcdef `
    -Device xvdh
```

6. インスタンスに接続してボリュームを使用できるようにします。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームを使用できるようにする」を参照してください。

> **Important**
>
> ボリュームを初期化しないでください。

7. [コントロールパネル] を開き、[プログラムと機能] を選択します。[Turn Windows features on or off] を選択します。インストールメディアの指定を求められたら、インストールメディアを使用して EBS ボリュームを指定します。
8. (オプション) インストールメディアの使用が完了したら、Dismount-EC2Volume コマンドレットを使用してボリュームをインスタンスからデタッチします。ボリュームをデタッチした後、Remove-EC2Volume コマンドレットを使用してボリュームを削除できます。

EC2 Windows インスタンスに Windows Subsystem for Linux をインストールする

Windows Subsystem for Linux (WSL) は、Microsoft Windows の機能です。EC2 Windows インスタンスに WSL をインストールすることで、ネイティブ Linux ツールを Windows インスタンスで直接実行できます。

Windows Subsystem for Linux (WSL) は、WSL 1 と WSL 2 の 2 つのバージョンがあります。詳細については、Microsoft ウェブサイトの「Windows Subsystem for Linux のドキュメント」を参照してください。

要件

- オペレーティングシステムは、Windows Server 2019 以降である必要があります。
- 仮想化された Windows インスタンス (インスタンスサイズが `.metal` ではない、またはネストされた仮想化をサポートしないもの) にインストールできるのは、WSL 1 のみです。
- ネストされた仮想化をサポートし、`NestedVirtualization` CPU オプションが有効化されているインスタンスには、WSL 1 または WSL 2 をインストールできます。

- ベアメタルインスタンス (インスタンスサイズが .metal) には WSL 1 または WSL 2 のいずれかをインストールできます。ベアメタルインスタンスは、ネストされた仮想化に必要なサポートをデフォルトで提供します。

EC2 のネストされた仮想化に関する詳細については、「ネストされた仮想化を使用して Amazon EC2 インスタンスでハイパーバイザーを実行する」を参照してください。

Windows インスタンスに WSL をインストールする

WSL 1 をインストールするには

1. WSL をインストールします。使用するプロセスは、インスタンスで実行されている Windows Server のバージョンによって異なります。
   - Windows Server 2022 以降 - EC2 インスタンスで、次の標準インストールコマンドを実行します。

     ```
     wsl --install --enable-wsl1 --no-launch
     ```

   - Windows Server 2019 - Microsoft ウェブサイトの「Install WSL on previous versions of Windows Server」の説明に従って、WSL を有効にしてから WSL をインストールします。
2. EC2 インスタンスを再起動します。

   ```
   shutdown -r -t 20
   ```

3. WSL 1 を使用するように WSL を設定するには、インスタンスで次のコマンドを実行します。このステップは、仮想化されたインスタンス (インスタンスサイズが .metal ではない、またはネストされた仮想化向けに設定されていないもの) に必須のステップです。

   ```
   wsl --set-default-version 1
   ```

4. デフォルトのディストリビューションをインストールします。

   ```
   wsl --install
   ```

WSL 2 をインストールする (.metal、またはネストされた仮想化が有効化されているインスタンス)

EC2 インスタンスで、次の標準インストールコマンドを実行します。デフォルトでは、WSL 2 がインストールされます。

```
wsl --install
```

EC2 Windows トラブルシューティングユーティリティ

EC2WinUtil ドライバーは、Windows インスタンスに対して以下のタイプのトラブルシューティングサポートを提供します。

クラッシュコールスタック

EC2WinUtil は、インスタンスから基本的なクラッシュ情報を収集し、その情報をシリアルコンソールに書き込みます。以下のリストには、ユーティリティがコンソールに書き込む主な詳細情報の一部が含まれています。

- 障害を引き起こしたモジュールの ID。
- イベントに関連付けられた Windows エラーコード。
- 最近行われたコールのスタックトレース。

これらの詳細情報を使用することで、根本原因の初期分析を実行し、それ以上の分析が必要かどうかを判断できます。シリアルコンソールへの出力は、AWS が Amazon EC2 ドライバーのクラッシュ傾向を追跡し、大規模なクラッシュイベントを診断することも可能にします。

> **Note**
>
> EC2WinUtil は、クラッシュコールスタック内のお客様データを収集しません。

ドライバーリリースノートについては、「EC2 Windows ユーティリティドライバーのバージョン履歴」を参照してください。

EC2 Windows ユーティリティドライバーのバージョン履歴

以下は、Amazon EC2 上の各 Windows Server バージョンで実行される EC2WinUtil ドライバーを示す表です。オペレーティングシステムの以前のバージョンは、インスタンスの起動元である AWS Windows Server AMI にプリインストールされたドライバーを使用します。ユーザーと共有されている AMI、またはユーザーが AWS Marketplace 経由でサブスクライブした AMI には、ドライバーがプリインストールされていません。

Windows サーバー バージョン	EC2WinUtil ドライバーバージョン
Windows Server 2025	最新バージョン
Windows Server 2022	最新バージョン
Windows Server 2019	最新バージョン
Windows Server 2016	最新バージョン

> **Note**
>
> 3.0.0 より前のドライバーバージョンでは、手動インストールのために `EC2WinUtil` ドライバーをダウンロードすることができませんでした。以前のバージョンのドライバーは、AWS Windows AMI にプリインストールされたドライバーとしての利用に限定されていました。

以下は、`EC2WinUtil` ドライバーのリリース済みバージョンを説明する表です。

パッケージバージョン	ドライバーのバージョン	詳細	リリース日
3.2.0	3.2.0	EC2Launch v2 エージェントで使用するコンソールのログ記録メカニズムを強化しました。	2026 年 6 月 2 日
3.1.1	3.1.1	コンソール出力へのログの記録時におけるコールスタックを長くしました。	2026 年 3 月 3 日
3.1.0	3.1.0	電源管理イベントの処理方法を改善しました。	2026 年 2 月 4 日
3.0.0	3.0.0	Windows 10 用のドライバーをモダナイズし、プリミティブドライバーとしてのインストールのサポートを追加しました。	2024 年 6 月 13 日
2.0.0	2.0.0	メタルインスタンスタイプの MMIO シリアルポートでの出力に対するサポートが追加されました。また、	2018 年 8 月 23 日

パッケージバージョン	ドライバーのバージョン	詳細	リリース日
		クラッシュ解析の改善と出力形式の更新も行われました。	
1.0.1	1.0.1	Amazon Inspector との名前空間競合のため、ドライバー名が `EC2WinUtil` に変更されました。複数のバグ修正が含まれています。	2018 年 3 月 1 日
1.0.0	1.0.0	初回リリース。ドライバーは、当初 `AwsAgent` と呼ばれていました。	2017 年 11 月 28 日

EC2 Windows インスタンスのより新しいバージョンの Windows Server へのアップグレード

EC2 Windows インスタンスの Windows Server オペレーティングシステムを旧バージョンからアップグレードする場合は次のいずれかの方法を使用できます。

インプレースアップグレード

インプレースアップグレードは既存インスタンスで動作します。このプロセス中にオペレーティングシステムファイルのみが影響を受ける一方、設定、サーバーロール、データはそのまま残ります。

移行 (並行アップグレードとも呼ばれます)

移行では設定、構成、データを取り込み、この情報を新しい EC2 Windows インスタンス上のより新しいバージョンのオペレーティングシステムに移行します。インスタンスはAWS Marketplace からサブスクライブしているパブリックまたはプライベートの Windows AMI、または共有されている AMI から起動できます。EC2 Image Builder を使用してカスタム AMI を作成することもできます。詳細については「Image Builder ユーザーガイド」を参照してください。

> **Note**
>
> AWS はEC2 インスタンスで実行される Windows Server バージョンに対して一連の一般利用可能な Amazon マシンイメージ (AMI) を提供します。これらの AMI は毎月更新され

ています。最新の Windows AMI の詳細については「AWS Windows AMI リファレンス」を参照してください。

Microsoft では従来、インプレースアップグレードする代わりに、より新しいバージョンの Windows Server に移行することを推奨しています。移行はアップグレードのエラーや問題がより少ない反面、新しいインスタンスのプロビジョニング、アプリケーションの計画と実施、新しいインスタンスでの環境設定の調整が必要であるため、インプレースアップグレードより時間がかかる場合があります。インプレースアップグレードはより高速である反面、ソフトウェアの非互換性に伴うエラーが生じる場合があります。

内容

- EC2 Windows インスタンスでのインプレースアップグレードの実行
- Automation ランブックを使用した EC2 Windows インスタンスのアップグレード
- EC2 Windows インスタンスを Nitro ベースのインスタンスタイプに移行する
- EC2 Windows インスタンスのオペレーティングシステムアップグレードに関するトラブルシューティング

EC2 Windows インスタンスでのインプレースアップグレードの実行

インプレースアップグレードを実行する前に、どのネットワークドライバをインスタンスで実行しているかを確認する必要があります。PV ネットワークドライバを使用すると、リモートデスクトップを使用してインスタンスにアクセスできます。インスタンスは AWS PV、Intel Network Adapter、あるいは拡張ネットワーキングドライバーのいずれかを使用します。詳細については「Windows インスタンス用 Paravirtual ドライバー」を参照してください。

インプレースアップグレードを開始する前に

インプレースアップグレードを始める前に、以下のタスクを完了し、以下の重要な詳細情報を確認してください。

- Microsoft のドキュメントを参照し、アップグレードの要件、既知の問題、制限事項を把握します。また、アップグレードに関する公式の手順も確認します。
  - Windows Server 2012 のアップグレードオプション
  - Windows Server 2012 R2 のアップグレードオプション
  - Windows Server 2016 以降のアップグレードおよび変換のオプション

  - Windows Server をアップグレードする
- 少なくとも 2 つの vCPU と 4GB の RAM を持つインスタンスでオペレーティングシステムのアップグレードを実行することをお勧めします。必要に応じて、インスタンスを同じタイプのより大きなサイズ (例えば t2.small から t2.large) に変更し、アップグレードを実行してから元のサイズにサイズ変更することができます。インスタンスサイズを保持する必要がある場合はインスタンスコンソールのスクリーンショットを使用して進行状況を監視できます。詳細についてはAmazon EC2 インスタンスタイプの変更を参照してください。
- Windows インスタンス上のルートボリュームに十分な空きディスク容量があることを確認します。Windows セットアッププロセスによってディスク容量の不足が警告されないことがあります。特定のオペレーティングシステムのアップグレードに必要なディスク容量の詳細については Microsoft のマニュアルを参照してください。ボリュームに十分な空きディスク容量がない場合はその容量を拡張できます。詳細については Amazon EBS ユーザーガイド」の「Amazon EBS Elastic Volumes」を参照してください。
- アップグレードパスを決定します。オペレーティングシステムは同じアーキテクチャにアップグレードする必要があります。例えば、32 ビットシステムは 32 ビットシステムにアップグレードする必要があります。Windows Server 2008 R2 以降は 64 ビットシステムのみです。
- アンチウイルスとアンチスパイウェアのソフトウェアとファイアウォールを無効にします。このようなタイプのソフトウェアはアップグレードプロセスと競合する場合があります。アップグレードが完了したら、アンチウイルスとアンチスパイウェアのソフトウェア、およびファイアウォールを再度有効にします。
- EC2 Windows インスタンスを Nitro ベースのインスタンスタイプに移行するトピックで説明した最新のドライバーに更新します。
- Upgrade Helper Service ではCitrix PV ドライバーを実行しているインスタンスのみがサポートされています。インスタンスが Red Hat ドライバーを実行している場合は最初に手動でこれらのドライバーをアップグレードする必要があります。

AWSPV、Intel Network Adapter、または拡張ネットワーキングドライバーを使用したインスタンスをインプレースアップグレードする

次の手順に従って、AWS PV、Intel Network Adapter、または拡張ネットワーキングドライバーを使用して Windows Server インスタンスをアップグレードします。

インプレースアップグレードを実行するには

1. アップグレード予定のシステムの AMI をバックアップまたはテスト用に作成します。その後、テスト環境として用意したこのコピーでアップグレードを実行できます。アップグレードが完了

した場合はこのインスタンスにトラフィックをほとんどダウンタイムなしで切り替えることができます。アップグレードが失敗した場合はバックアップに戻すことができます。詳細については「Amazon EBS-backed AMI を作成する」を参照してください。

2. Windows Server インスタンスが最新のネットワークドライバを使用していることを確認します。

   a. AWS PV ドライバーを更新するには「EC2 Windows インスタンスでの PV ドライバーのアップグレード」を参照してください。

   b. ENA ドライバーを更新するには「EC2 Windows インスタンスに ENA ドライバーをインストールする」を参照してください。

   c. Intel ドライバーを更新するには「Intel 82599 VF インターフェイスを使用する拡張ネットワーキング」を参照してください。

3. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

4. ナビゲーションペインで、[インスタンス] を選択してください。インスタンスを見つけます。そのインスタンスの [インスタンス ID] および [アベイラビリティーゾーン] をメモしておきます。この情報はこの手順で後ほど使用します。

5. Windows Server 2012 または 2012 R2 から Windows Server 2016 以降にアップグレードする場合、手順を進める前にインスタンスで次の操作を実行します。

   a. EC2Config サービスをアンインストールします。詳細については「EC2Launch v2 および EC2Config エージェントの Windows サービス管理」を参照してください。

   b. EC2Launch v1 または EC2Launch v2 エージェントをインストールします。詳細については「EC2 Windows インスタンスの起動時に EC2Launch v1 エージェントを使用してタスクを実行する」および「EC2 Windows インスタンスの起動時に EC2Launch v2 エージェントを使用してタスクを実行する」を参照してください。

   c. AWS Systems Manager SSM Agent をインストールします。詳細については、「AWS Systems Manager ユーザーガイド」の「Manually install SSM Agent on Amazon EC2 for Windows Server」を参照してください。

6. Windows Server インストールメディアスナップショットから新しいボリュームを作成します。

   a. 左サイドバーのナビゲーションペインで [Elastic Block Store] の [スナップショット] を選択してください。

   b. フィルターバーで、[パブリックスナップショット] を選択してください。

   c. 検索バーで、次のフィルターを指定します。

      - [所有者のエイリアス]、[=]、[amazon] の順に選択してください。

- [説明] を選択し、**Windows** の入力を開始します。アップグレード先のシステムアーキテクチャと言語設定に一致する Windows フィルターを選択してください。例えば、Windows Server 2019 にアップグレードする場合には[Windows 2019 English Installation Media] を選びます。

d. アップグレード先のシステムアーキテクチャおよび言語設定と一致するスナップショットの横にあるチェックボックスをオンにし、[アクション] および [スナップショットからボリュームを作成] を選択してください。

e. [ボリュームの作成] ダイアログボックスで、Windows インスタンスと同一のアベイラビリティーゾーンを選択し、[ボリュームの作成] を選択してください。

7. ページ上部の [ボリューム vol-***1234567890example*** が正常に作成されました] バナーで、作成したボリュームの ID を選択してください。

8. [Actions] (アクション)、[Attach volume] (ボリュームのアタッチ) の順に選択してください。

9. [ボリュームのアタッチ] ページの[インスタンス]で、Windows インスタンスのインスタンス ID を選択し、[ボリュームのアタッチ] を選択してください。

10. 「Amazon EBS ボリュームを使用できるようにする」のステップに従い、新しいボリュームを使用可能にします。

> **Important**
>
> ディスクを初期化すると既存のデータが削除されるため、初期化は実行しないでください。

11. Windows PowerShell で、新しいボリュームドライブに切り替えます。インスタンスにアタッチしたインストールメディアボリュームを開き、アップグレードを開始します。

a. Windows Server 2016 以降にアップグレードする場合は以下を実行します。

```
.\setup.exe /auto upgrade /dynamicupdate disable
```

> **Note**
>
> setup.exe を /dynamicupdate オプションを無効に設定して実行すると、Windows は Windows Server のアップグレードプロセス中に更新プログラムをインストールできなくなります。これはアップグレード中に更新プログラムをイン

ストールするとエラーが発生する可能性があるためです。アップグレードの完了後に、Windows Update で更新プログラムをインストールできます。

Windows Server の旧バージョンにアップグレードする場合は以下を実行します。

```
Sources\setup.exe
```

b. [インストールするオペレーティングシステムを選択] ページで、Windows Server インスタンスの完全インストール版を選択し、[次へ] を選択します。

c. [Which type of installation do you want? (どのタイプのインストールが必要ですか。)] で、[アップグレード] を選択してください。

d. ウィザードを終了します。

Windows Server セットアップはファイルをコピーして処理します。数分後、リモートデスクトップセッションが終了します。アップグレードにかかる時間はアプリケーションの数と Windows Server インスタンスで実行されているサーバーロールによって異なります。短くて 40 分、長くて数時間かかることがあります。インスタンスはアップグレードプロセス中に 1 つ以上のステータスチェックに失敗する可能性があります。アップグレードが完了すると、すべてのステータスチェックが成功します。コンソール出力のシステムログを確認するか、Amazon CloudWatch メトリクスでディスクと CPU の動作を確認して、アップグレードが進行しているかどうかを確認できます。

> **Note**
>
> Windows Server 2019 にアップグレードする場合、アップグレードが完了した後で、必要に応じて、デスクトップの背景を手動で変更して、以前のオペレーティングシステム名を削除できます。

インスタンスが数時間後にすべてのステータスチェックに成功しなかった場合、「EC2 Windows インスタンスのオペレーティングシステムアップグレードに関するトラブルシューティング」を参照してください。

アップグレード後のタスク

1. インスタンスにログインし、.NET Framework のアップグレードを開始します。システムを再起動するように求められたら、その指示に従います。

2. 前のステップで実行しなかった場合はEC2Launch v1 または EC2Launch v2 エージェントをインストールします。詳細についてはEC2 Windows インスタンスの起動時に EC2Launch v1 エージェントを使用してタスクを実行する および EC2 Windows インスタンスの起動時に EC2Launch v2 エージェントを使用してタスクを実行する を参照してください。
3. Windows Server 2012 R2 にアップグレードした場合、PV ドライバーを AWS PV ドライバーにアップグレードすることをお勧めします。Nitro ベースのインスタンスでアップグレードした場合、NVME および ENA ドライバーをインストールまたはアップグレードすることをお勧めします。詳細については「NVMe ドライバー」または「Windows の拡張ネットワーキングの有効化」を参照してください。
4. アンチウイルスとアンチスパイウェアのソフトウェアとファイアウォールを再度有効にします。

Automation ランブックを使用した EC2 Windows インスタンスのアップグレード

AWS Systems Manager オートメーションランブックを使用して、AWS で Windows および SQL Server インスタンスの自動アップグレードを実行することができます。

内容

- 関連サービス
- 実行オプション
- Windows Server をアップグレードする
- SQL Server のアップグレード

関連サービス

自動アップグレードプロセスでは次の AWS サービスを使用します。

- AWS Systems Manager。AWS Systems Manager はAWS リソースを集中管理する強力な統合インターフェイスです。詳細についてはAWS Systems Manager ユーザーガイドを参照してください。
- AWS Systems Manager エージェント (SSM Agent) はAmazon EC2 インスタンス、オンプレミスのサーバー、または仮想マシン (VM) にインストールして設定できる Amazon のソフトウェアです。SSM Agent により、Systems Manager がこれらのリソースを更新、管理、および設定できるようにします。このエージェントは AWS クラウド上の Systems Manager サービスからのリ

クエストを処理し、リクエストに指定されたとおりにそれらを実行します。詳細についてはAWS Systems Manager ユーザーガイドのSSM Agent を使用するを参照してください。

- AWS Systems Manager SSM ランブック。SSM ランブックはマネージドインスタンスで Systems Manager が実行するアクションを定義します。SSM ランブックは JavaScript Object Notation (JSON) や YAML を使用し、これにはユーザーが指定するステップおよびパラメータが含まれます。このトピックでは2 つのオートメーション用 Systems Manager SSM ランブックを使用します。詳細については「AWS Systems Manager ユーザーガイド」の「AWS Systems Manager オートメーションランブックリファレンス」を参照してください。

実行オプション

Systems Manager コンソールで [Automation] を選択する際、[実行] を選択してください。Automation ドキュメントを選択すると、自動化の実行オプションを選択するよう求められます。以下のオプションから選択してください。このトピックで後ほど示すパスのステップでは[シンプルな実行] オプションを使用します。

シンプルな実行

1 つのインスタンスを更新するが、自動化の各ステップを実行して結果を監査しない場合はこのオプションを選択してください。このオプションについては以降のアップグレード手順で詳しく説明します。

レート制御

アップグレードを複数のインスタンスに適用する場合はこのオプションを選択してください。以下の設定を定義します。

- パラメータ

  [マルチアカウント] および [リージョン] でも設定されているこの設定では自動化の分岐方法を定義します。
- [Targets] (ターゲット)

  自動化を適用するターゲットを選択してください。この設定は[マルチアカウント] および [リージョン] でも設定されます。
- パラメータ値

  オートメーションドキュメントのパラメータで定義されている値を使用します。
- Resource Group (リソースグループ)

AWS ではリソースはユーザーが操作できるエンティティです。例えば、Amazon EC2 インスタンス、AWS CloudFormation スタック、または Amazon S3 バケットなどがあります。複数のリソースを使用する場合はタスクごとに 1 つの AWS サービスから別のサービスに移動するのではなく、グループとしてそれらを管理する方が有益な場合があります。場合によってはアプリケーション層を構成する EC2 インスタンスなど、多数の関連リソースを管理する場合があります。この場合はこれらのリソースに対して一括してアクションを実行する必要があります。

- タグ

  タグはAWS リソースを目的、所有者、環境などさまざまな方法で分類するのに役立ちます。この分類は同じ種類のリソースが多い場合に便利です。割り当てたタグを使用して、特定のリソースをすばやく識別することができます。

- レート制御

  レート制御は[マルチアカウント] および [リージョン] でも設定されます。レート制御のパラメータを設定する際、ターゲットカウントまたはターゲットの割合 (%) によって、自動化を適用するフリートの数を定義します。

マルチアカウントおよびマルチリージョン

マルチアカウントとマルチリージョンの設定でも使用されるレート制御で指定されたパラメータに加えて、2 つの設定があります。

- アカウントと組織単位 (OU)

  自動化を実行する複数のアカウントを指定します。

- AWS リージョン

  自動化を実行する複数の AWS リージョン を指定します。

手動による実行

このオプションは[シンプルな実行] に似ていますが、このオプションでは自動化の各ステップを進め、結果を監査することができます。

Windows Server をアップグレードする

AWSEC2-CloneInstanceAndUpgradeWindows ランブックではアカウントの Windows Server インスタンスから Amazon マシンイメージ (AMI) を作成し、この AMI を、サポートされている希望の

バージョンにアップグレードします。このマルチステッププロセスは完了するまで 2 時間かかる場合があります。

自動アップグレードプロセスには 2 つの AMI が含まれています。

- 現在実行中のインスタンス。最初の AMI は現在実行中のインスタンスです。このインスタンスはアップグレードされません。この AMI は別のインスタンスを起動してインプレースアップグレードを実行するために使用されます。プロセスが完了したら、この AMI はアカウントから削除されます。ただし、元のインスタンスを保持するように特別にリクエストした場合を除きます。この設定を行うにはKeepPreUpgradeImageBackUp パラメータを使用します (デフォルト値は `false` です。つまり、AMI はデフォルトで削除されます)。
- 更新された AMI。この AMI は自動化プロセスの結果です。

最終結果は1 つの AMI です。つまり、AMI の更新されたインスタンスです。

アップグレードが完了したら、Amazon VPC で新しい AMI を起動して、アプリケーション機能をテストできます。テストが終了したら、別のアップグレードを実行する前に、アプリケーションのダウンタイムをスケジュールしてから、アップグレードされたインスタンスに完全に切り替えます。

前提条件

AWS Systems Manager オートメーションドキュメントを使用して Windows Server のアップグレードを自動化するには以下のタスクを実行する必要があります。

- 指定された IAM ポリシーを使用して IAM ロールを作成することで、Systems Manager が Amazon EC2 インスタンスに対して自動化タスクを実行できるようにし、Systems Manager を使用するための前提条件を満たしていることを確認します。詳細については「AWS Identity and Access Management ユーザーガイド」の「AWS のサービスにアクセス許可を委任するロールの作成」を参照してください。
- 自動化の実行方法に関するオプションを選択してください。実行のオプションには[シンプルな実行]、[レートの制御]、[複数のアカウントとリージョン]、[手動の実行] があります。これらのパラメータの詳細については「実行オプション」を参照してください。
- インスタンスに SSM Agent がインストールされていることを確認します。詳細については「Installing and configuring SSM Agent on Amazon EC2 instances for Windows Server (Windows Server の Amazon EC2 インスタンスで SSM Agent をインストールして設定する)」を参照してください。
- Windows PowerShell 3.0 以降をインスタンスにインストールする必要があります。

- Microsoft Active Directory ドメインに参加しているインスタンスの場合はホスト名の競合を避けるために、ドメインコントローラーに接続できない `SubnetId` を指定することをお勧めします。
- インスタンスサブネットにはインターネットへのアウトバウンド接続が必要です。これにより、Amazon S3 などの AWS のサービス へのアクセスと、Microsoft からのパッチのダウンロードが可能になります。この要件はサブネットがパブリックサブネットでインスタンスにパブリック IP アドレスがある場合、またはサブネットがインターネットトラフィックをパブリック NAT デバイスに送信するルートを持つプライベートサブネットの場合に満たされます。
- このオートメーションはWindows Server 2008 R2、Windows Server 2012 R2、Windows Server 2016、および Windows Server 2019 を実行しているインスタンスで機能します。
- インスタンスでブートディスクに 20 GB の空きディスク領域があることを確認します。
- インスタンスが AWS の提供する Windows ライセンスを使用しない場合はWindows Server 2012 R2 インストールメディアを含む Amazon EBS スナップショット ID を指定します。これを実行するには:
  1. Amazon EC2 インスタンスで Windows Server 2012 以降が実行されていることを確認します。
  2. インスタンスが実行されているのと同じアベイラビリティーゾーンに 6 GB の Amazon EBS ボリュームを作成します。ボリュームをインスタンスにアタッチします。それをマウントします (例えばドライブ D として)。
  3. ISO を右クリックし、インスタンスにマウントします (例えばドライブ E として)。
  4. ISO の内容をドライブ E:\ からドライブ D:\ にコピーします。
  5. 上記のステップ 2 で作成した 6 GB ボリュームの Amazon EBS スナップショットを作成します。

Windows Server のアップグレードの制限事項

このオートメーションではWindows のドメインコントローラー、クラスター、または Windows デスクトップオペレーティングシステムのアップグレードはサポートされていません。さらに、このオートメーションでは以下のロールがインストールされた Windows Server の Amazon EC2 インスタンスもサポートされていません。

- リモートデスクトップセッションホスト (RDSH)
- リモートデスクトップ接続ブローカー (RDCB)
- リモートデスクトップ仮想化ホスト (RDVH)
- リモートデスクトップウェブアクセス (RDWA)

Windows Server の自動アップグレードの実行のステップ

以下のステップに従い、AWSEC2-CloneInstanceAndUpgradeWindows オートメーションランブックを使用して Windows Server インスタンスをアップグレードします。

1. AWS マネジメントコンソールから Systems Manager を開きます。
2. 左のナビゲーションペインの [Change Management] (変更管理) で、[Automation] (オートメーション) を選択してください。
3. [Execute automation (自動化の実行)] を選択してください。
4. `AWSEC2-CloneInstanceAndUpgradeWindows` と呼ばれるオートメーションドキュメントを検索します。
5. ドキュメント名が表示されたら、選択してください。選択すると、ドキュメントの詳細が表示されます。
6. [Execute automation] (オートメーションの実行) を選択して、このドキュメントのパラメータを入力してください。ページの上部にある [シンプルな実行] は選択したままにします。
7. 次のガイダンスに従って、リクエストされたパラメータを入力してください。

   - `InstanceID`

     タイプ: 文字列

     (必須) SSM エージェントがインストールされている Windows Server 2008 R2、2012 R2、2016、2019 を実行しているインスタンス。

   - `InstanceProfile.`

     タイプ: 文字列

     (必須) IAM インスタンスプロファイル。この IAM ロールはAmazon EC2 インスタンスと AWS AMI に対して Systems Manager のオートメーションを実行するために使用されます。詳細については、AWS Systems Managerユーザーガイドの「EC2 インスタンスの権限の設定」を参照してください。

   - `TargetWindowsVersion`

     タイプ: 文字列

     (必須) ターゲットの Windows バージョンを選択してください。

   - `SubnetId`

タイプ: 文字列

(必須) このサブネットはアップグレードプロセス用であり、ソース EC2 インスタンスの場所を指します。サブネットに Amazon S3 などの AWS サービス や Microsoft (パッチのダウンロード用) へのアウトバウンド接続があることを確認します。

- `KeepPreUpgradedBackUp`

  タイプ: 文字列

  (オプション) このパラメータが `true` に設定されている場合は自動化によって、インスタンスから作成されたイメージが保持されます。デフォルトの設定は`false` です。

- `RebootInstanceBeforeTakingImage`

  タイプ: 文字列

  (オプション) デフォルトは `false` です (再起動なし)。このパラメータが `true` に設定されている場合はSystems Managerによって、アップグレード用の AMI を作成する前にインスタンスが再起動されます。

8. パラメータを入力したら、[実行] を選択してください。自動化が開始したら、実行の進行状況をモニタリングすることができます。
9. 自動化が完了すると、AMI ID が表示されます。Windows OS がアップグレードされたことを確認するにはAMI を起動します。

> **Note**
>
> 自動化のすべてのステップを実行する必要はありません。それぞれのステップには自動化とインスタンスの動作に基づいた条件が付けられています。Systems Manager は必須でない一部のステップをスキップする場合があります。
> さらに、いくつかの手順がタイムアウトすることもあります。Systems Manager ではすべての最新パッチについて、インストールとアップグレードが試みられます。ただし、場合によっては特定のステップの定義可能なタイムアウト設定に基づいてパッチがタイムアウトします。この場合、Systems Manager の自動化は次のステップに進み、内部 OS ターゲットの Windows Server バージョンにアップグレードされるようにします。

10. 自動化が完了したら、AMI ID を使用して Amazon EC2 インスタンスを起動して、アップグレードを確認することができます。AWS AMI から Amazon EC2 インスタンスを作成する方法の詳細については「カスタム AMI から EC2 インスタンスを起動する方法」を参照してください。

SQL Server のアップグレード

AWSEC2-CloneInstanceAndUpgradeSQLServer スクリプトではアカウントで SQL Server を実行している Amazon EC2 インスタンスから AMI を作成し、その AMI を SQL Server の新しいバージョンにアップグレードします。このマルチステッププロセスは完了するまで 2 時間かかる場合があります。

自動化はインスタンスから AMI を作成し、指定したサブネットで新しい AMI を起動します。その後、オートメーションはSQL Server のインプレースアップグレードを実行します。アップグレードが完了したら、自動化によって、アップグレードされたインスタンスを終了する前に新しい AMI が作成されます。

自動アップグレードプロセスには 2 つの AMI が含まれています。

- 現在実行中のインスタンス。最初の AMI は現在実行中のインスタンスです。このインスタンスはアップグレードされません。この AMI は別のインスタンスを起動してインプレースアップグレードを実行するために使用されます。プロセスが完了したら、この AMI はアカウントから削除されます。ただし、元のインスタンスを保持するように特別にリクエストした場合を除きます。この設定を行うにはKeepPreUpgradeImageBackUp パラメータを使用します (デフォルト値は `false` です。つまり、AMI はデフォルトで削除されます)。
- 更新された AMI。この AMI は自動化プロセスの結果です。

最終結果は1 つの AMI です。つまり、AMI の更新されたインスタンスです。

アップグレードが完了したら、Amazon VPC で新しい AMI を起動して、アプリケーション機能をテストできます。テストが終了したら、別のアップグレードを実行する前に、アプリケーションのダウンタイムをスケジュールしてから、アップグレードされたインスタンスに完全に切り替えます。

前提条件

AWS Systems Manager オートメーションドキュメントを使用して SQL Server のアップグレードを自動化するには以下のタスクを実行する必要があります。

- 指定された IAM ポリシーを使用して IAM ロールを作成することで、Systems Manager が Amazon EC2 インスタンスに対して自動化タスクを実行できるようにし、Systems Manager を使用するための前提条件を満たしていることを確認します。詳細については「AWS Identity and Access Management ユーザーガイド」の「AWS のサービス に許可を委任するロールの作成」を参照してください。

- 自動化の実行方法に関するオプションを選択してください。実行のオプションには[シンプルな実行]、[レートの制御]、[複数のアカウントとリージョン]、[手動の実行] があります。これらのパラメータの詳細については「実行オプション」を参照してください。
- Amazon EC2 インスタンスではWindows Server 2008 R2 以降および SQL Server 2008 以降を使用する必要があります。
- インスタンスに SSM Agent がインストールされていることを確認します。詳細については「Windows Server の Amazon EC2 インスタンスで SSM Agent を使用する」を参照してください。
- インスタンスに十分な空きディスク容量があることを確認します。
  - Windows Server 2008 R2 から 2012 R2 にアップグレードする場合、または Windows Server 2012 R2 からそれ以降のオペレーティングシステムにアップグレードする場合はインスタンスのブートディスクに 20 GB の空きディスク容量があることを確認してください。
  - Windows Server 2008 R2 から 2016 以降にアップグレードする場合はインスタンスのブートディスクに 40 GB の空きディスク領域があるか、インスタンスを確認します。
- Bring Your Own License (BYOL) SQL Server バージョンを使用するインスタンスの場合、次の前提条件が追加で適用されます。
  - ターゲットの SQL Server インストールメディアを含む Amazon EBS スナップショット ID を提供します。これを実行するには:
    1. Amazon EC2 インスタンスで Windows Server 2008 R2 以降が実行されていることを確認します。
    2. インスタンスが実行されているのと同じアベイラビリティーゾーンに 6 GB の Amazon EBS ボリュームを作成します。ボリュームをインスタンスにアタッチします。それをマウントします (例えばドライブ D として)。
    3. ISO を右クリックし、インスタンスにマウントします (例えばドライブ E として)。
    4. ISO の内容をドライブ E:\ からドライブ D:\ にコピーします。
    5. ステップ 2 で作成した 6 GB ボリュームの Amazon EBS スナップショットを作成します。

SQL Server 自動アップグレードの制限事項

AWSEC2-CloneInstanceAndUpgradeSQLServer ランブックを使用して自動アップグレードを実行する場合は以下の制限が適用されます。

- アップグレードはWindows 認証を使用して SQL Server 上でのみ実行できます。

- インスタンスに保留中のセキュリティパッチの更新がないことを確認します。[Control Panel (コントロール パネル)] を開き、[Check for updates (更新の確認)] を選択してください。
- HA およびミラーリングモードでの SQL Server のデプロイはサポートされていません。

SQL Server の自動アップグレードの実行のステップ

以下のステップに従い、AWSEC2-CloneInstanceAndUpgradeSQLServer オートメーションランブックを使用して SQL Server をアップグレードします。

1. まだダウンロードしていない場合はSQL Server 2016 の .iso ファイルをダウンロードして、ソースサーバーにマウントします。
2. .iso ファイルがマウントされたら、コンポーネントファイルをすべてコピーし、任意のボリューム上に置きます。
3. そのボリュームの Amazon EBS スナップショットを取得し、後で使用できるようにそのスナップショット ID をクリップボードにコピーします。6+6詳細については「[Amazon EBS ユーザーガイド]」の「Amazon EBS スナップショットの作成」を参照してください。
4. インスタンスプロファイルを Amazon EC2 ソースインスタンスにアタッチします。これにより、Systems Manager はEC2 インスタンスと通信し、AWS Systems Manager サービスに追加されているコマンドを実行できるようになります。例えば、ロールに `SSM-EC2-Profile-Role` という名前を付け、そのロールに `AmazonSSMManagedInstanceCore` ポリシーをアタッチします。
5. AWS Systems Manager コンソールの左のナビゲーションペインで、[マネージドインスタンス] を選択してください。EC2 インスタンスがマネージドインスタンスのリストに含まれていることを確認します。数分後にインスタンスが表示されない場合はAWS Systems Manager ユーザーガイドのインスタンスがある場所を参照してください。
6. 左のナビゲーションペインの [Change Management] (変更管理) で、[Automation] (オートメーション) を選択してください。
7. [Execute automation (自動化の実行)] を選択してください。
8. `AWSEC2-CloneInstanceAndUpgradeSQLServer` と呼ばれるオートメーションドキュメントを検索します。
9. `AWSEC2-CloneInstanceAndUpgradeSQLServer` SSM ドキュメントを選択し、[Next] (次へ) を選択してください。
10. [シンプルな実行] オプションが選択されていることを確認します。
11. 次のガイダンスに従って、リクエストされたパラメータを入力してください。

- `InstanceId`

  タイプ: 文字列

  (必須) SQL Server 2008 R2 (またはそれ以降) を実行しているインスタンス。

- `IamInstanceProfile`

  タイプ: 文字列

  (必須) IAM インスタンスプロファイル。

- `SQLServerSnapshotId`

  タイプ: 文字列

  (必須) ターゲット SQL Server インストールメディア用のスナップショット ID。このパラメータはSQL Server ライセンス込みインスタンスには必要ありません。

- `SubnetId`

  タイプ: 文字列

  (必須) このサブネットはアップグレードプロセス用であり、ソース EC2 インスタンスの場所を指します。サブネットに Amazon S3 などの AWS サービス や Microsoft (パッチのダウンロード用) へのアウトバウンド接続があることを確認します。

- `KeepPreUpgradedBackUp`

  タイプ: 文字列

  (オプション) このパラメータが `true` に設定されている場合は自動化によって、インスタンスから作成されたイメージが保持されます。デフォルトの設定は`false` です。

- `RebootInstanceBeforeTakingImage`

  タイプ: 文字列

  (オプション) デフォルトは `false` です (再起動なし)。このパラメータが `true` に設定されている場合はSystems Managerによって、アップグレード用の AMI を作成する前にインスタンスが再起動されます。

- `TargetSQLVersion`

  タイプ: 文字列

(オプション) ターゲット SQL Server のバージョン。デフォルトは 2016 です。

12. パラメータを入力したら、[実行] を選択してください。自動化が開始したら、実行の進行状況をモニタリングすることができます。
13. [実行ステータス] に [成功] と表示されている場合は[出力] を展開して AMI 情報を確認します。AMI ID を使用して、任意の VPC で SQL Server インスタンスを起動します。
14. Amazon EC2 コンソールを開きます。左のナビゲーションペインで [AMI] を選択してください。新しい AMI が表示されます。
15. SQL Server の新しいバージョンが正常にインストールされていることを確認するには新しい AMI を選択して、[Launch] (起動) を選択してください。
16. AMI で使用するインスタンスのタイプ、デプロイする VPC とサブネット、使用するストレージを選択してください。AMI から新しいインスタンスを起動しているため、起動している新しい EC2 インスタンスに含めるボリュームがオプションとして提示されます。これらのボリュームは必要に応じて削除または追加できます。
17. インスタンスを識別しやすいようにタグを追加します。
18. 1 つ以上のセキュリティグループをインスタンスに追加します。
19. [インスタンスの作成] を選択してください。
20. インスタンスのタグ名を選択し、[アクション] ドロップダウンの [接続] を選択してください。
21. SQL Server の新しいバージョンが新しいインスタンスのデータベースエンジンとして表示されていることを確認します。

EC2 Windows インスタンスを Nitro ベースのインスタンスタイプに移行する

AWS Windows AMI はMicrosoft のインストールメディアで使用されるデフォルト設定で設定されており、いくつかのカスタマイズがあります。M5 や C5 といった Nitro ベースのインスタンスをサポートするドライバーや設定などをカスタマイズできます。

以下のようなケースで Xen ベースのインスタンスから Nitro ベースのインスタンス (ベアメタルインスタンスなど) に移行する場合はこのトピックの手順に従うことをお勧めします。

- カスタム Windows AMI からインスタンスを作成する場合
- 2018 年 8 月より前に作成された Amazon が提供する Windows AMI からインスタンスを作成する場合

また、AWSSupport-UpgradeWindowsAWSDrivers オートメーションドキュメントを使用して、パート 1、パート 2、パート 3 で説明した手順を自動化することもできます。自動化された手順を使用する場合は(代替案) AWS Systems Manager を使用した AWS PV、ENA、NVMe ドライバーのアップグレード を参照して、パート 4 とパート 5 に進みます。

詳細についてはAmazon EC2 の更新 - 追加インスタンスタイプ、Nitro システム、および CPU オプションを参照してください。

> **Note**
>
> Windows Server バージョン 2016 以降では以下の移行手順を実行できます。オペレーティングシステムのサポート期限が切れた古いバージョンではテストされておらず、最新のインスタンスタイプと互換性がない可能性があります。
> Linux インスタンスを移行するには「the section called “インスタンスタイプの変更”」を参照してください。

目次

- パート 1: AWS PV ドライバーのインストールとアップグレード
- パート 2: ENA のインストールとアップグレード
- パート 3: AWS NVMe ドライバーをアップグレード
- パート 4: EC2Config および EC2Launch の更新
- パート 5: ベアメタルインスタンスのシリアルポートドライバーのインストール
- パート 6: 電源管理設定の更新
- パート 7: 新しいインスタンスタイプ用の Intel チップセットドライバーの更新
- (代替案) AWS Systems Manager を使用した AWS PV、ENA、NVMe ドライバーのアップグレード

[開始する前に]

この手順ではXen ベースのインスタンス (M4 や C4 など) を使用中で、それを Nitro ベースのインスタンスに移行しようとしていることが前提となります。

アップグレードを正常に実行するにはPowerShell バージョン 3.0 以降を使用する必要があります。

> **Note**
>
> 移行する場合、インスタンスが新しい Enhanced Networking Adapter デバイスにデフォルトで設定されるため、既存のネットワークインターフェイスカードの静的 IP またはカスタム DNS ネットワーク設定は失われる場合があります。

この手順のステップを実行する前に、インスタンスのバックアップを作成することをお勧めします。EC2 コンソールで、移行が必要なインスタンスを選択し、コンテキストメニュー (右クリック) を開いたら、[インスタンスの状態]、[停止] の順に選択してください。

> **Warning**
>
> インスタンスを停止すると、インスタンスストアボリューム上のデータは消去されます。インスタンスストアボリューム上のデータを保持するには永続的ストレージにデータをバックアップするようにしてください。

EC2 コンソールでインスタンスの右クリックコンテキストメニューを開き、[イメージ]、[イメージの作成] の順に選択してください。

> **Note**
>
> この手順のパート 4 およびパート 5 はインスタンスタイプを移行または変更した後で完了できます。ただし、特にベアメタルインスタンスタイプに移行する場合は移行前に完了することをお勧めします。

パート 1: AWS PV ドライバーのインストールとアップグレード

AWS PV ドライバーが Nitro システムで使用されていなくても、Citrix PV または AWS PV の以前のバージョンを使用している場合はアップグレードする必要があります。最新の AWS PV ドライバーはNitro システムがある場合、または Xen ベースインスタンスに戻す必要がある場合に表面化するかもしれない以前のバージョンのドライバーのバグを解決します。ベストプラクティスとして、AWS の Windows インスタンスの最新ドライバーに常に更新することをお勧めします。

次の手順に従って、Windows Server 2008 R2、Windows Server 2012、Windows Server 2012 R2、Windows Server 2016、または Windows Server 2019 で、 AWS PV ドライバーのインプレースアップグレードを実行するか、Citrix PV ドライバーから AWS PV ドライバーにアップグレードしま

す。詳細については[EC2 Windows インスタンスでの PV ドライバーのアップグレード]を参照してください。

ドメインコントローラーをアップグレードするには[ドメインコントローラーのアップグレード (AWS PV アップグレード)]を参照してください。

AWS PV ドライバーのアップグレードを実行するには

1. リモートデスクトップを使用してインスタンスに接続し、アップグレードのためにインスタンスを準備します。アップグレードを実行する前に、システムディスク以外をすべてオフラインにします。AWS PV ドライバーのインプレースアップグレードを実行する場合はこのステップは必要ありません。また、サービスコンソールで、不可欠でないサービスを [手動] 起動に設定します。
2. インスタンスに最新のドライバーパッケージを[ダウンロード]します。
3. フォルダの内容を抽出し、`AWSPVDriverSetup.msi` を実行します。

MSI の実行後、インスタンスは自動的に再起動され、ドライバーがアップグレードされます。インスタンスは最大 15 分間、使用できなる場合があります。

アップグレードが完了し、インスタンスが Amazon EC2 コンソールの両方のヘルスチェックに合格した後、リモートデスクトップを使用してインスタンスに接続して、新しいドライバーがインストールされたことを確認します。デバイスマネージャーの [Storage Controllers] で、[AWS PV Storage Host Adapter] を見つけます。ドライバーバージョンがドライバーのバージョン履歴の表に掲載されている最新バージョンと同じであることを確認します。詳細については[AWS PV ドライバーパッケージの履歴]を参照してください。

パート 2: ENA のインストールとアップグレード

すべてのネットワーク機能がサポートされるように、最新の Elastic Network Adapter ドライバーにアップグレードします。インスタンスを起動し、すでに拡張ネットワーキングが有効になっていない場合、必要なネットワークアダプタドライバーをダウンロードしてインスタンスにインストールします。次に、拡張ネットワークを有効にするように enaSupport インスタンス属性を設定します。この属性はサポートされるインスタンスタイプにおいて、ENA ドライバーがインストールされている場合のみ有効にできます。詳細については[EC2 インスタンスで ENA による拡張ネットワーキングを有効にする]を参照してください。

1. インスタンスに最新のドライバーを[ダウンロード]します。以前のバージョンのドライバーが必要な場合は「[ENA Windows ドライバーのバージョン履歴]」を参照してください。

2. zip アーカイブを展開します。
3. 展開したフォルダから `install.ps1` PowerShell スクリプトを実行してドライバーをインストールします。

> **Note**
>
> インストールエラーを回避するには`install.ps1` スクリプトを管理者として実行します。

4. ご使用の AMI が enaSupport を有効にしているかどうか確認します。有効でない場合はEC2 インスタンスで ENA による拡張ネットワーキングを有効にする のマニュアルに従ってください。

パート 3: AWS NVMe ドライバーをアップグレード

AWS NVMe ドライバーはNitro システム内では NVMe ブロックデバイスとして表示される Amazon EBS および SSD インスタンスストアボリュームとやり取りを行い、パフォーマンスを向上させるために使用されます。

> **Important**
>
> Xen ベースのインスタンスで AWS NVMe をインストールまたはアップグレードできるように、次の手順が変更されています。これはインスタンスを Nitro ベースのインスタンスに移行することが目的です。

1. インスタンスに最新のドライバーパッケージをダウンロードします。

   以前のバージョンのドライバーが必要な場合は「AWS NVMe Windows ドライバーのリリース」でサポートされるバージョンを確認してください。
2. zip アーカイブを展開します。
3. `Readme.txt` の説明に従ってドライバーをインストールします。
4. PowerShell セッションを開き、次のコマンドを実行します。

```
PS C:\> start rundll32.exe sppnp.dll,Sysprep_Generalize_Pnp -wait
```

> **Note**
>
> コマンドを適用するには管理者として PowerShell セッションを実行している必要があります。PowerShell (x86) のバージョンではエラーが発生します。
> このコマンドはデバイスドライバーでのみ sysprep を実行します。完全な sysprep 準備は実行しません。

5. Windows Server 2008 R2 および Windows Server 2012 の場合、インスタンスをシャットダウンし、インスタンスタイプを変更し起動してからパート 4 に進みます。Nitro ベースのインスタンスタイプに移行する前に、インスタンスを再度 Xen ベースのインスタンスタイプで起動しようとしても動作しません。他のサポートされた Windows AMI の場合はデバイスの sysprep の後いつでもインスタンスタイプを変更できます。

パート 4: EC2Config および EC2Launch の更新

Windows インスタンスの場合、最新の EC2Config および EC2Launch ユーティリティが EC2 ベアメタルの場合を含む Nitro Systemで実行されると追加の機能および情報が提供されます。EC2Config サービスはWindows Server 2016 より前の AMI にデフォルトで含まれています。EC2Launch は Windows Server 2016 以降の AMI の EC2Config を置き換えます。

EC2Config サービスおよび EC2Launch サービスが更新されると、AWS からの新しい Windows AMI には最新バージョンのサービスが含まれます。ただし、EC2Config および EC2Launch の最新バージョンを使用して、独自の Windows AMI とインスタンスを更新する必要があります。

EC2Config のインストールまたは更新

1. EC2Config インストーラをダウンロードして解凍します。
2. `EC2Install.exe` を実行します。オプションの完全なリストについては`EC2Install` オプションを指定して `/?` を実行してください。デフォルトではセットアップによってプロンプトが表示されます。プロンプトを表示せずにコマンドを実行するには`/quiet` オプションを使用します。

詳細については EC2Config の最新バージョンのインストールを参照してください。

EC2Launch のインストールまたは更新

1. インスタンスで EC2Launch をすでにインストールして設定している場合はEC2Launch 設定ファイルのバックアップを作成します。インストールプロセスではこのファイルに変更内容が

保存されません。デフォルトではこのファイルは C:\ProgramData\Amazon\EC2-Windows\Launch\Config ディレクトリにあります。

2. EC2-Windows-Launch.zip をインスタンス上のディレクトリにダウンロードします。
3. install.ps1 を EC2-Windows-Launch.zip のダウンロード先と同じディレクトリにダウンロードします。
4. install.ps1 を実行します。

> **Note**
>
> インストールエラーを回避するにはinstall.ps1 スクリプトを管理者として実行します。

5. EC2Launch 設定ファイルのバックアップを作成する場合は同ファイルを C:\ProgramData\Amazon\EC2-Windows\Launch\Config ディレクトリにコピーします。

詳細についてはEC2 Windows インスタンスの起動時に EC2Launch v1 エージェントを使用してタスクを実行するを参照してください。

パート 5: ベアメタルインスタンスのシリアルポートドライバーのインストール

i3.metal インスタンスタイプではI/O ポートベースのシリアルデバイスではなく、PCI ベースのシリアルデバイスを使用しています。最新の Windows AMI は自動的に PCI ベースのシリアル・デバイスを使用し、シリアル・ポート・ドライバーがインストール済みです。日付が 2018.04.11 以降の Amazon が提供する Windows AMI から作成したインスタンスを使用していない場合、シリアル・ポート・ドライバーをインストールしてパスワード生成やコンソール出力などの EC2 機能用のシリアル・デバイスを有効にする必要があります。最新の EC2Config および EC2Launch はi3.metal もサポートし、追加機能を提供します。まだ実行していない場合はパート 4 のステップに従います。

シリアルポートドライバーをインストールするには

1. インスタンスにシリアルドライバーパッケージをダウンロードします。
2. フォルダの内容を展開し、aws_ser.INF のコンテキストメニューを開き (右クリック)、[インストール] を選択してください。
3. [OK] を選択してください。

パート 6: 電源管理設定の更新

次のように電源管理設定を更新すると、ディスプレイが消灯しないように設定されます。これにより、Nitro Systemで正常な OS のシャットダウンが可能になります。2018 年 11 月 28 日時点で Amazon が提供しているすべての Windows AMI は既にこのデフォルトの設定になっています。

1. コマンドプロンプトまたは PowerShell セッションを開きます。
2. 以下のコマンドを実行します。

```
powercfg /setacvalueindex 381b4222-f694-41f0-9685-ff5bb260df2e 7516b95f-
f776-4464-8c53-06167f40cc99 3c0bc021-c8a8-4e07-a973-6b14cbcb2b7e 0
powercfg /setacvalueindex 8c5e7fda-e8bf-4a96-9a85-a6e23a8c635c 7516b95f-
f776-4464-8c53-06167f40cc99 3c0bc021-c8a8-4e07-a973-6b14cbcb2b7e 0
powercfg /setacvalueindex a1841308-3541-4fab-bc81-f71556f20b4a 7516b95f-
f776-4464-8c53-06167f40cc99 3c0bc021-c8a8-4e07-a973-6b14cbcb2b7e 0
```

パート 7: 新しいインスタンスタイプ用の Intel チップセットドライバーの更新

`u-6tb1.metal`、`u-9tb1.metal`、および `u-12tb1.metal` インスタンスタイプは以前は Windows AMI にインストールされていなかったチップセットドライバーを必要とするハードウェアを使用します。日付が 2018 年 11 月 19 日以降の Amazon が提供する Windows AMI から起動したインスタンスを使用していない場合、Intel チップセット INF ユーティリティを使用して、そのドライバーをインストールする必要があります。

チップセットドライバーをインストールするには

1. インスタンスへのチップセット INF ユーティリティ。
2. ファイルを展開します。
3. `SetupChipset.exe` を実行します。
4. Intel ソフトウェアライセンス契約に同意し、チップセットドライバーをインストールします。
5. インスタンスを再起動します。

(代替案) AWS Systems Manager を使用した AWS PV、ENA、NVMe ドライバーのアップグレード

AWSSupport-UpgradeWindowsAWSDrivers オートメーションドキュメントはパート 1、パート 2、パート 3 で説明した手順を自動化することもできます。この方法はドライバーのアップグレードに失敗したインスタンスを修復することもできます。

AWSSupport-UpgradeWindowsAWSDrivers オートメーションドキュメントは指定された EC2 インスタンスでストレージおよびネットワーク AWS ドライバーをアップグレードまたは修復します。このドキュメントではAWS Systems Manager エージェント (SSM Agent) を呼び出すことによって、最新バージョンの AWS ドライバーをオンラインでインストールしようとしています。SSM Agent が接続可能でない場合、明示的に要求された場合、ドキュメントは AWS ドライバーのオフラインインストールを実行できます。

> **Note**
>
> ドメインコントローラーではこの手順は失敗します。ドメインコントローラーでドライバを更新するにはドメインコントローラーのアップグレード (AWS PV アップグレード)を参照してください。

AWS Systems Manager を使用して AWS PV、ENA および NVMe ドライバーを自動的にアップグレードするには

1. https://console.aws.amazon.com/ Systems Manager で Systems Manager コンソールを開きます。
2. [オートメーション]、[オートメーションの実行] の順に選択してください。
3. AWSSupport-UpgradeWindowsAWSDrivers オートメーションドキュメントを検索して選択し、[オートメーションの実行] を選択してください。
4. [入力パラメータ] セクションで、次のオプションを設定します。

   [インスタンス ID]

   アップグレードするインスタンスの一意の ID を入力してください。

   AllowOffline

   (オプション) 次のオプションのいずれかを選択してください。

- `True` —オフラインインストールを実行するにはこのオプションを選択してください。インスタンスはアップグレード処理中に停止され、再開されます。

  > **Warning**
  >
  > インスタンスを停止すると、インスタンスストアボリューム上のデータは消去されます。インスタンスストアボリューム上のデータを保持するには永続的ストレージにデータをバックアップするようにしてください。

- `False` — (デフォルト) オンラインインストールを実行するにはこのオプションを選択したままにします。インスタンスはアップグレード処理中に再起動されます。

  > **Important**
  >
  > オンラインおよびオフラインのアップグレードではアップグレード操作を試みる前に AMI が作成されます。AMI は自動化が完了した後も維持されます。AMI へのアクセスを保護するか、不要になった場合は削除してください。

SubnetId

(オプション) 以下のいずれかの値にエラーがあります。

- `SelectedInstanceSubnet` — (デフォルト) アップグレードプロセスではアップグレードするインスタンスと同じサブネットにヘルパーインスタンスを起動します。サブネットはSystems Manager エンドポイント (`ssm.*`) との通信を可能にする必要があります。
- `CreateNewVPC` — アップグレードプロセスではヘルパーインスタンスが新しい VPC で起動されます。ターゲットインスタンスのサブネットが `ssm.*` エンドポイントと通信できるかどうかが不明な場合はこのオプションを使用します。ユーザーはVPC を作成するアクセス許可が必要です。
- 特定のサブネット ID — ヘルパーインスタンスを起動する特定のサブネットの ID を指定します。サブネットはアップグレードするインスタンスと同じアベイラビリティーゾーンに存在し、`ssm.*` エンドポイントとの通信を許可する必要があります。

5. [実行] を選択してください。
6. アップグレードの完了を許可します。オンラインアップグレードの完了には最大 10 分、オフラインアップグレードの完了までには最大 25 分かかります。

EC2 Windows インスタンスのオペレーティングシステムアップグレードに関するトラブルシューティング

AWS は Upgrade Helper Service に関する問題についてアップグレードサポートを提供しています。Upgrade Helper Service はCitrix PV ドライバーを伴うインプレースアップグレードの実行を支援する AWS ユーティリティです。

アップグレードの後で、.NET Runtime Optimization サービスが .NET フレームワークを最適化する間、インスタンスの平均 CPU 使用率が一時的に高くなる場合があります。これは想定される動作です。

インスタンスが数時間後にすべてのステータスチェックに成功しなかった場合、次の項目を確認してください。

- Windows Server 2008 にアップグレードして数時間後にすべてのステータスチェックに失敗した場合、アップグレードが失敗した可能性があり、ロールバックを確定する [OK をクリック] のプロンプトが表示されます。この状態ではコンソールにアクセスできないため、[OK] ボタンをクリックすることはできません。この状態を回避するにはAmazon EC2 コンソールまたは API を使用して再起動を行います。再起動が始まるまで 10 分以上かかります。インスタンスが使用可能になるまで 25 分ほどかかることがあります。
- サーバーからアプリケーションまたはサーバーロールを削除した後、もう一度試します。

サーバーからアプリケーションまたはサーバーロールを削除した後、インスタンスがすべてのステータスチェックに失敗した場合、次の操作を行います。

- インスタンスを停止し、ルートボリュームを別のインスタンスにアタッチします。詳細については"メタデータサービスを待っています"でルートボリュームを停止して別のインスタンスにアタッチする方法の説明を参照してください。
- Windows セットアップのログファイルとイベントログで失敗の原因を調べます。

オペレーティングシステムのアップグレードまたは移行に関するその他の問題についてはインプレースアップグレードを開始する前に一覧表示されている記事を確認することをお勧めします。

チュートリアル: Amazon RDS データベースに Amazon EC2 インスタンスを接続する

チュートリアルの目的

このチュートリアルでは、AWS マネジメントコンソール を使用して、Amazon EC2 インスタンスと Amazon RDS データベース間のセキュア接続を設定する方法について説明します。

接続を設定する方法は複数あります。このチュートリアルでは、以下の 3 つのオプションについて説明します。

- オプション 1: EC2 コンソールを使用してインスタンスを RDS データベースに自動的に接続する

  EC2 コンソールの自動接続機能を使用して、EC2 インスタンスと RDS データベース間のトラフィックを許可するように EC2 インスタンスと RDS データベース間の接続を自動的に設定します。
- オプション 2: RDS コンソールを使用して、インスタンスを RDS データベースに自動的に接続する

  RDS コンソールの自動接続機能を使用して、EC2 インスタンスと RDS データベース間のトラフィックを許可するように EC2 インスタンスと RDS データベース間の接続を自動的に設定します。
- オプション 3: セキュリティグループを作成して、インスタンスを RDS データベースに手動で接続する

  EC2 インスタンスと RDS データベース間の接続を設定するには、セキュリティグループを手動で設定してから、セキュリティグループを割り当てて、オプション 1 とオプション 2 の自動接続機能が自動作成する設定を再現します。

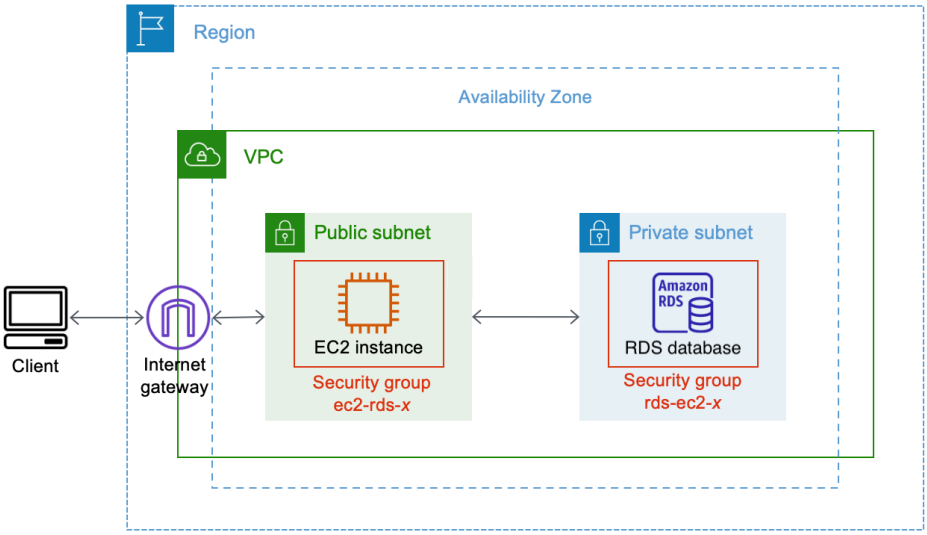
Context

EC2 インスタンスと RDS データベース間の接続を設定する理由の背景として、次のシナリオを考えてみましょう。ウェブサイトでユーザーに入力してもらうフォームがある場合、フォームデータをデータベースに取り込む必要があります。ウェブサーバーとして設定された EC2 インスタンスでウェブサイトをホストし、フォームデータを RDS データベースに取得できます。フォームデータを EC2 インスタンスから RDS データベースに送信するには、EC2 インスタンスと RDS データベースを相互接続する必要があります。このチュートリアルでは、その接続を構成する方法について説明し

ます。なお、これは EC2 インスタンスと RDS データベースを接続するユースケースの一例にすぎません。

アーキテクチャ

次の図は、作成されるリソースと、このチュートリアルのすべてのステップを完了した結果のアーキテクチャ構成を示しています。



図は、作成する以下のリソースを示しています。

- 同じ AWS リージョン、VPC、およびアベイラビリティーゾーンに EC2 インスタンスと RDS データベースを作成します。
- パブリックサブネットに EC2 インスタンスを作成します。
- プライベートサブネットに RDS データベースを作成します。

  RDS コンソールを使用して RDS データベースを作成し、EC2 インスタンスに自動接続すると、VPC、DB サブネットグループ、およびデータベースのパブリックアクセス設定が自動的に選択されます。RDS データベースは、EC2 インスタンスと同じ VPC 内のプライベートサブネットに自動的に作成されます。
- インターネットユーザーは、SSH または HTTP/HTTPS を使用して、インターネットゲートウェイ経由で EC2 インスタンスに接続できます。
- インターネットユーザーは RDS データベースに直接接続することはできません。EC2 インスタンスのみが RDS データベースに接続します。
- 自動接続機能を使用して EC2 インスタンスと RDS データベース間のトラフィックを許可すると、次のセキュリティグループが自動的に作成および追加されます。

- セキュリティグループ ec2-rds-*x* が作成され、EC2 インスタンスに追加されます。ここには、rds-ec2-*x* セキュリティグループを送信先として参照するアウトバウンドルールが 1 つ存在します。このルールにより、EC2 インスタンスからのトラフィックが rds-ec2-*x* セキュリティグループのある RDS データベースに到達できるようになります。
- セキュリティグループ rds-ec2-*x* が作成され、RDS データベースに追加されます。ec2-rds-*x* セキュリティグループを送信元として参照するインバウンドルールが 1 つ存在します。このルールにより、ec2-rds-*x* セキュリティグループを持つ EC2 インスタンスからのトラフィックが RDS データベースに到達できるようになります。

個別のセキュリティグループ (EC2 インスタンス用と RDS データベース用にそれぞれ 1 つずつ) を使用することで、インスタンスとデータベースのセキュリティをより適切に管理できます。インスタンスとデータベースの両方で同じセキュリティグループを使用し、例えばデータベースのみと合うようにセキュリティグループを変更した場合、その変更はインスタンスとデータベースの両方に影響します。言い換えると、1 つのセキュリティグループを使用した場合、セキュリティグループがアタッチされていることを忘れてしまい、リソース (インスタンスまたはデータベース) のセキュリティを意図せず変更してしまう可能性があることになります。

また、自動的に作成されるセキュリティグループは最小特権に従っており、ワークロード固有のセキュリティグループペアを作成することで、データベースポート上のこのワークロードに対する相互接続のみを許可します。

考慮事項

このチュートリアルのステップを実行する際には、次の点を考慮してください。

- 2 つのコンソール — このチュートリアルでは、次の 2 つのコンソールを使用します。
  - Amazon EC2 コンソール — EC2 コンソールを使用してインスタンスを起動したり、EC2 インスタンスを RDS データベースに自動接続したり、手動オプションのときにはセキュリティグループを作成して接続を設定します。
  - Amazon RDS コンソール — RDS コンソールを使用して RDS データベースを作成したり、EC2 インスタンスを RDS データベースに自動接続します。
- 1 つの VPC — 自動接続機能を使用するには、EC2 インスタンスと RDS データベースが同じ VPC 内にある必要があります。

  EC2 インスタンスと RDS データベース間の接続を手動で設定する場合、VPC で EC2 インスタンスを起動して、別の VPC で RDS データベースを起動することができますが、追加のルーティン

グと VPC 設定を設定する必要があります。このシナリオについては、このチュートリアルでは説明しません。

- 1 つの AWS リージョン — EC2 インスタンスと RDS データベースは同じリージョンにある必要があります。
- 2 つのセキュリティグループ — EC2 インスタンスと RDS データベース間の接続は、EC2 インスタンスのセキュリティグループと RDS データベースのセキュリティグループという 2 つのセキュリティグループが設定します。

  EC2 コンソールまたは RDS コンソールの自動接続機能を使用して接続を設定する場合 (このチュートリアルのオプション 1 とオプション 2)、セキュリティグループは自動的に作成され、EC2 インスタンスと RDS データベースに割り当てられます。

  自動接続機能を使用しない場合、セキュリティグループを手動で作成して割り当てる必要があります。これは、このチュートリアルのオプション 3 で行います。

チュートリアル完了までの時間

30 分

チュートリアルの全過程を 1 回で完了することも、1 つのタスクずつに分けて完了することもできます。

コスト

このチュートリアルを完了すると、作成する AWS リソースのコストが発生する可能性があります。

AWS アカウントが無料利用枠の対象で、無料利用枠の要件に従ってリソースを設定している場合は、Amazon EC2 の無料利用枠を利用できます。詳細については、2025 年 7 月 15 日より前とそれ以降の無料利用枠の特典 を参照してください。

EC2 インスタンスと RDS データベースが異なるアベイラビリティーゾーンにある場合は、データ転送料金が発生します。これらの料金の発生を回避するには、EC2 インスタンスと RDS データベースが同じアベイラビリティーゾーンにある必要があります。データ転送料金の詳細については、「Amazon EC2 オンデマンド料金」ページの「データ転送」を参照してください。

チュートリアル完了後にコストが発生するのを避けるため、不要になったリソースは必ず削除してください。リソースを削除するステップについては、「タスク 4 (オプション): クリーンアップする」を参照してください。

オプション 1: EC2 コンソールを使用してインスタンスを RDS データベースに自動的に接続する

オプション 1 では、EC2 コンソールの自動接続機能を使用して、EC2 インスタンスと RDS データベース間のトラフィックを許可するように EC2 インスタンスと RDS データベース間の接続を自動的に設定します。オプション 3 では、この接続を手動で設定する方法について説明します。

タスク

- [開始する前に]
- タスク 1 (オプション): RDS データベースを作成する
- タスク 2 (オプション): EC2 インスタンスを起動する
- タスク 3: EC2 インスタンスを RDS データベースに自動接続する
- タスク 4: 接続設定を検証する
- タスク 5 (オプション): クリーンアップする

[開始する前に]

このチュートリアルを完了するには、以下が必要です。

- EC2 インスタンスと同じ VPC にある RDS データベース。既存の RDS データベースを使用するか、タスク 1 のステップに従って新しい RDS データベースを作成することができます。
- RDS データベースと同じ VPC にある EC2 インスタンス。既存の EC2 インスタンスを使用するか、タスク 2 のステップに従って新しい EC2 インスタンスを作成することができます。
- 次の操作を呼び出すアクセス許可:
  - `ec2:AssociateRouteTable`
  - `ec2:AuthorizeSecurityGroupEgress`
  - `ec2:CreateRouteTable`
  - `ec2:CreateSecurityGroup`
  - `ec2:CreateSubnet`
  - `ec2:DescribeInstances`
  - `ec2:DescribeNetworkInterfaces`
  - `ec2:DescribeRouteTables`
  - `ec2:DescribeSecurityGroups`

- `ec2:DescribeSubnets`
- `ec2:ModifyNetworkInterfaceAttribute`
- `ec2:RevokeSecurityGroupEgress`

タスク 1 (オプション): RDS データベースを作成する

> **Note**
>
> Amazon RDS データベースの作成については、このチュートリアルの対象外です。RDS データベースがすでにあり、このチュートリアルで使用する場合は、このタスクをスキップできます。
> 既存の RDS データベースを使用する場合は、自動接続機能を使用できるように、その RDS データベースが EC2 インスタンスと同じ VPC 内にあることを確認してください。

このタスクでは、RDS データベースを作成して、EC2 インスタンスと使用する RDS データベース間の接続を設定するタスク 3 を完了できるようにします。このタスクのステップでは、RDS データベースを次のように設定します。

- エンジンタイプ: MySQL
- テンプレート: 無料利用枠
- DB インスタンス識別子: **`tutorial-database-1`**
- DB インスタンスクラス: `db.t3.micro`

> **Important**
>
> 本番環境では、具体的なニーズに合わせて、データベースを設定する必要があります。

MySQL RDS データベースを作成するには

1. Amazon RDS コンソール (https://console.aws.amazon.com/rds/) を開きます。
2. リージョンセレクター (右上) から AWS リージョン を選択します。EC2 コンソールの自動接続機能を使用するには、データベースと EC2 インスタンスが同じリージョンにある必要があります。

3. ダッシュボード で、[Create database] (データベースの作成) を選択します。
4. [Choose a database creation method] (データベース作成方法を選択) で [Standard Create] (スタンダード作成) が選択されていることを確認します。[Easy create] (簡易作成) を選択した場合、VPC セレクターは利用できません。EC2 コンソールの自動接続機能を使用するには、データベースが EC2 インスタンスと同じ VPC 内にあることを必ず確認してください。
5. [Engine options] (エンジンオプション) にある [Engine type] (エンジンタイプ) で MySQL を選択します。
6. [Templates] (テンプレート) では、ニーズに合うサンプルテンプレートを選択します。このチュートリアルでは、[Free tier] (無料利用枠) を選択して RDS データベースを無料で作成できます。ただし、無料利用枠は、無料利用枠の対象となるアカウントでのみ利用できることに注意してください。詳細については、[Free tier] (無料利用枠) ボックスにある [Info] (情報) リンクを選択してください。
7. [設定] で、次のいずれかを実行します。

   a. [DB instance identifier] (DB インスタンス識別子) に、データベースの名前を入力します。このチュートリアルでは、**`tutorial-database-1`** と入力します。

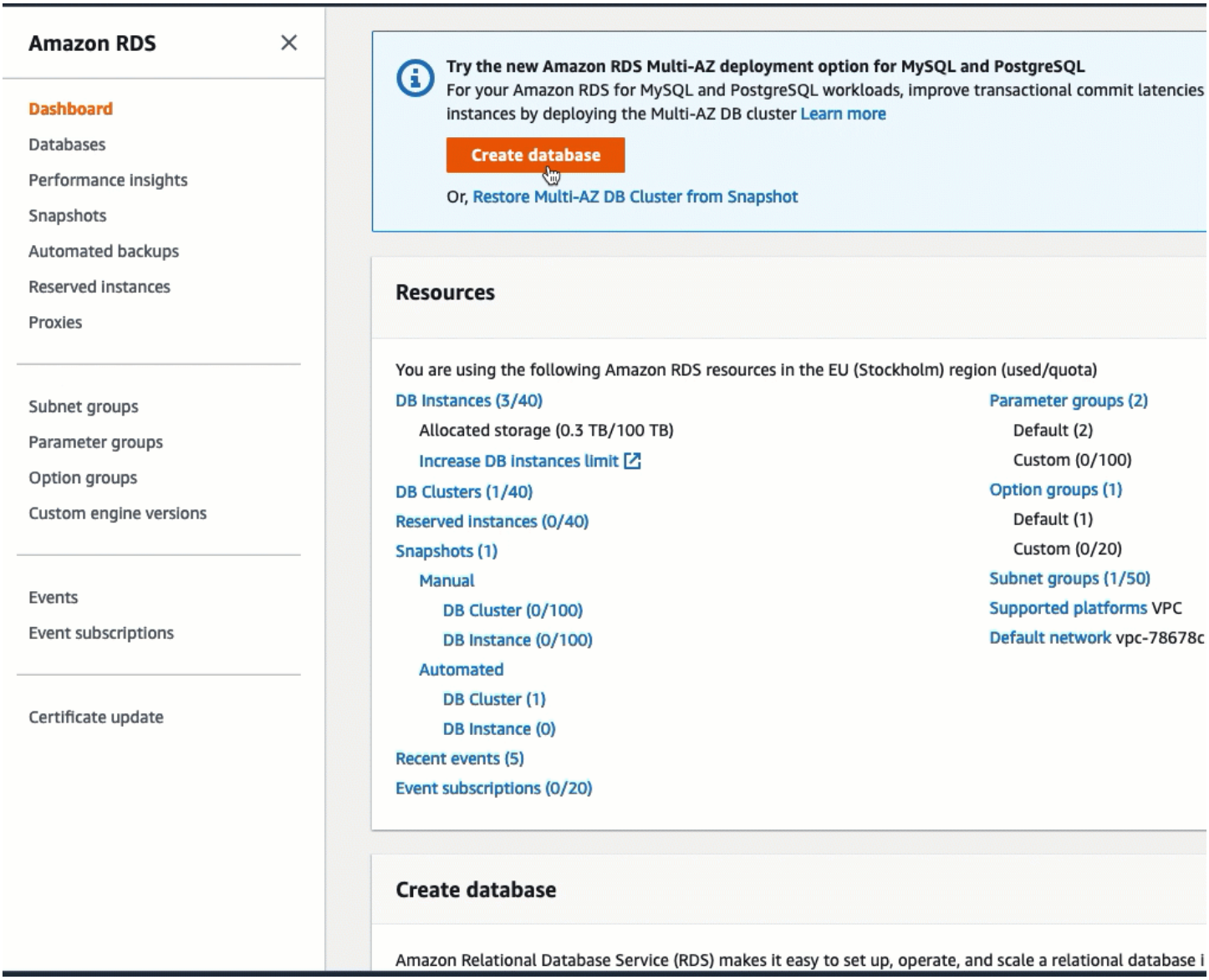
   b. [Master username] (マスターユーザー名) は、デフォルトの名前である **`admin`** のままにします。

   c. [Master password] (マスターパスワード) に、このチュートリアルに使用するパスワードを入力し、[Confirm password] (パスワードの確認) にパスワードをもう一度入力します。
8. [インスタンス設定] の [DB インスタンスクラス] は、デフォルトの [db.t3.micro] のままにします。アカウントが無料利用枠の対象である場合は、このデータベースクラスを無料で使用できます。詳細については、「AWS 無料利用枠」を参照してください。
9. [Connectivity] (接続) では、[Compute resource] (コンピュートリソース) に [Don't connect to an EC2 compute resource] (EC2 コンピュートリソースに接続しない) を選択します。EC2 インスタンスと RDS データベースは、タスク 3 の後半で接続するからです。

   (後ほど、このチュートリアルのオプション 2 で、[Connect to an EC2 compute resource] (EC2 コンピュートリソースに接続) を選択して RDS コンソールの自動接続機能を試します。)
10. [Virtual private cloud (VPC)] (仮想プライベートクラウド (VPC)) には VPC を選択します。VPC には DB サブネットグループが必要です。自動接続機能を使用するには、EC2 インスタンスと RDS データベースが同じ VPC 内にある必要があります。
11. このページの他のフィールドについては、すべてのデフォルト値をそのまま使用します。
12. [データベースの作成] を選択します。

[Databases] (データベース) 画面では、データベースが使用可能になるまで、新しいデータベースの [Status] (ステータス) は [Creating] (作成中) になります。ステータスが [Available] (使用可能) に変わったら、データベースに接続できます。データベースクラスとストレージ量によっては、新しいデータベースが使用可能になるまで最長 20 分かかることがあります。

アニメーションを表示: RDS データベースの作成



タスク 2 (オプション): EC2 インスタンスを起動する

> **Note**
>
> インスタンスの起動は、このチュートリアルの対象外です。Amazon EC2 インスタンスがすでにあり、このチュートリアルで使用する場合は、このタスクをスキップできます。
> 既存の EC2 インスタンスを使用する場合は、自動接続機能を使用できるように、その EC2 インスタンスが RDS データベースと同じ VPC 内にあることを確認してください。

このタスクでは、EC2 インスタンスを起動して、EC2 インスタンスと使用する Amazon RDS データベース間の接続を設定するタスク 3 を完了できるようにします。このタスクのステップでは、EC2 インスタンスを次のように設定します。

- インスタンス名: **`tutorial-instance-1`**
- AMI: Amazon Linux 2
- インスタンスタイプ: `t2.micro`
- パブリック IP の自動割り当て: 有効
- 次の 3 つのルールを持つセキュリティグループ:
  - IP アドレスからの SSH を許可
  - 任意の場所からの HTTPS トラフィックを許可
  - 任意の場所からの HTTP トラフィックを許可

> **Important**
>
> 本番環境では、具体的なニーズに合わせて、インスタンスを設定する必要があります。

EC2 インスタンスを起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. リージョンセレクター (右上) から AWS リージョン を選択します。EC2 コンソールの自動接続機能を使用するには、インスタンスと RDS データベースが同じリージョンにある必要があります。
3. [EC2 Dashboard] (EC2 ダッシュボード) で、[Launch instance] (インスタンスの作成) を選択します。

4. [Names and tags] (名前とタグ) にある [Name] (名前) には、インスタンスを識別するための名前を入力します。このチュートリアルでは、インスタンス名は「**`tutorial-instance-1`**」にします。インスタンス名は必須ではありませんが、EC2 コンソールでインスタンスを選択するときに、名前があると識別しやすくなります。
5. [Application and OS Images] (アプリケーションと OS イメージ) で、ウェブサーバーに必要な AMI を選択します。このチュートリアルでは Amazon Linux 2 を使用します。
6. [Instance type] (インスタンスタイプ) にある [Instance type] (インスタンスタイプ) には使用しているウェブサーバーに必要なインスタンスタイプを選択します。このチュートリアルでは、`t2.micro` を使用します。

   > **Note**
   >
   > アカウントの作成日に応じて、Amazon EC2 を無料利用枠で使用できる場合があります。
   > 2025 年 7 月 15 日より前に AWS アカウントを作成していて、12 か月が経過していない場合は、[t2.micro] インスタンスタイプ ([t2.micro] が利用できないリージョンでは [t3.micro] インスタンスタイプ) を選択することで、Amazon EC2 の無料利用枠を使用できます。[t3.micro] インスタンスを起動すると、デフォルトで [Unlimited] モードになり、CPU 使用率に基づいて追加料金が発生する可能性があることに注意してください。インスタンスタイプを無料利用枠で使用できる場合、[無料利用枠の対象] とラベル付けされます。
   > 2025 年 7 月 15 日以降に AWS アカウント を作成した場合、t3.micro、t3.small、t4g.micro、t4g.small、c7i-flex.large、m7i-flex.large インスタンスタイプを 6 か月間、またはクレジットを使い切るまで使用できます。
   > 詳細については「2025 年 7 月 15 日より前とそれ以降の無料利用枠の特典」を参照してください。

7. [Key pair (login)] (キーペア (ログイン)) にある [Key pair name] (キーペア名) には、使用するキーペアを選択します。
8. [Network settings] (ネットワーク設定) で、次の操作を行います：

   a. デフォルトの VPC またはサブネットに変更を加えていない場合は、[Network] (ネットワーク) と [Subnet] (サブネット) でデフォルトの設定をそのまま使用できます。

      デフォルトの VPC またはサブネットに変更を加えた場合は、以下を確認してください。

i. 自動接続機能を使用するには、インスタンスと RDS データベースが同じ VPC 内にある必要があります。デフォルトでは、VPC は 1 つのみです。

ii. インスタンスを起動する VPC には、インターネットからウェブサーバーにアクセスできるように、インターネットゲートウェイがアタッチされている必要があります。デフォルト VPC はインターネットゲートウェイで自動的に設定されます。

iii. インスタンスがパブリック IP アドレスを受け取れるように、[Auto-assign Public IP] (自動割り当てパブリック IP) で [Enable] (有効) が選択されていることを確認します。[Disable] (無効) が選択されている場合は、[Edit] (編集) ([Network Settings] (ネットワーク設定) の右側) を選択し、その後 [Auto-assign public IP]] (パブリック IP の自動割り当て) で [Enable] (有効) を選択します。

b. SSH を使用してインスタンスに接続するには、コンピュータのパブリック IPv4 アドレスからの SSH (Linux) または RDP (Windows) トラフィックを承認するセキュリティグループのルールが必要です。デフォルトでは、インスタンスを起動すると、任意の場所からのインバウンド SSH トラフィックを許可するルールで新しいセキュリティグループが作成されます。

お使いの IP アドレスのみがインスタンスに接続できるようにするには、[ファイアウォール (セキュリティグループ)] にある [SSH トラフィックを許可する送信元] チェックボックスの横にあるドロップダウンリストから [マイ IP] を選択します。

c. インターネットからインスタンスへのトラフィックを許可するには、次のチェックボックスをオンにします。

- [Allow HTTPs traffic from the internet] (インターネットからの HTTPs トラフィックを許可する)
- [Allow HTTP traffic from the internet] (インターネットからの HTTP トラフィックを許可する)

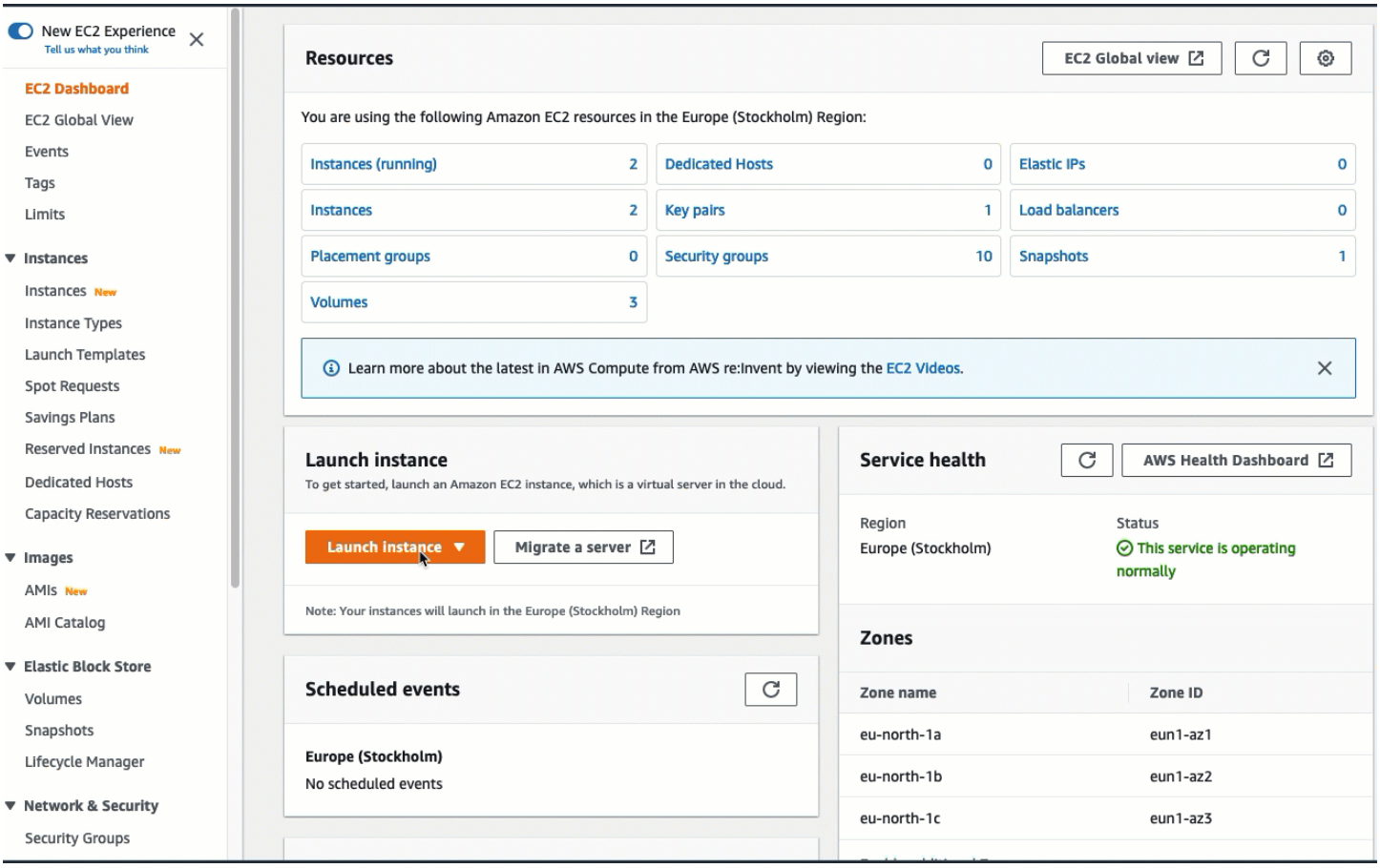
9. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動する) を選択します。

10. 確認ページは開いたままにします。次のタスクで、インスタンスをデータベースに自動接続するときに必要になります。

インスタンスが起動しないか、状態が `terminated` ではなくすぐに `running` になる場合は[Amazon EC2 インスタンスの起動に関する問題のトラブルシューティング]を参照してください。

インスタンスの起動方法の詳細については、「コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する」を参照してください。

アニメーションを表示: EC2 インスタンスを起動する



タスク 3: EC2 インスタンスを RDS データベースに自動接続する

このタスクでは、EC2 コンソールの自動接続機能を使用して、EC2 インスタンスと RDS データベース間の接続を自動的に設定します。

EC2 コンソールを使用して、EC2 インスタンスを RDS データベースに自動接続する場合

1. インスタンスの起動確認ページ (前述のタスク実行時に開いたままにしておく) で、[Connect an RDS database] (RDS データベースに接続) を選択します。

   確認ページを閉じてしまった場合は、次の手順に従ってください。

   a. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

   b. ナビゲーションペインで、[インスタンス] を選択してください。

c. 先ほど作成した EC2 インスタンスを選択し、[Actions] (アクション)、[Networking] (ネットワーク)、[Connect RDS database] (RDS データベースを接続) の順に選択します。

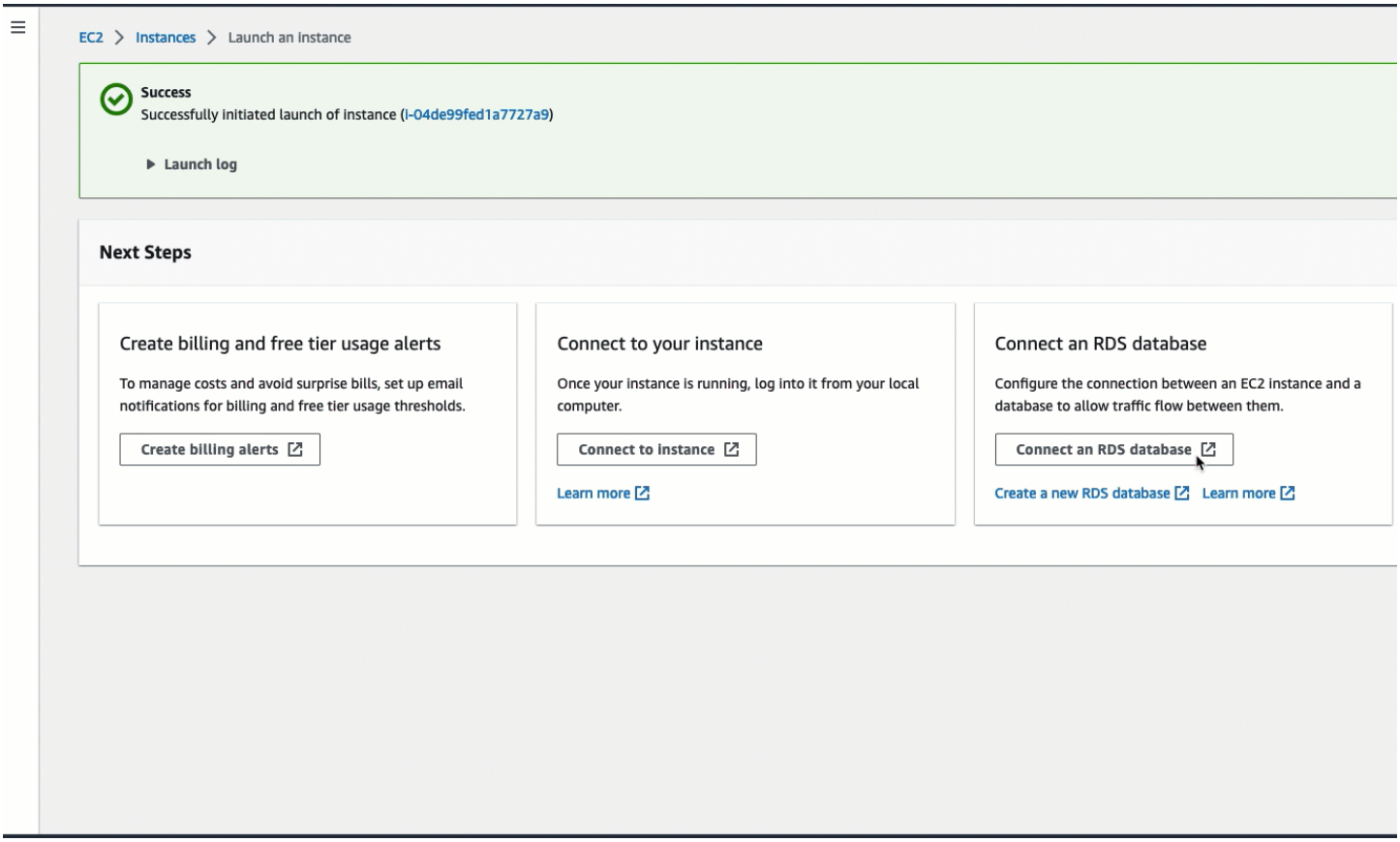
   [Connect RDS database] (RDS データベースに接続) を選択できない場合は、EC2 インスタンスが [Running] (実行中) の状態であることを確認します。

2. [Database role] (データベースロール) で [Instance] (インスタンス) を選択します。この場合のインスタンスとは、データベースインスタンスを指します。
3. [RDS database] (RDS データベース) の場合は、タスク 1 で作成した RDS データベースを選択します。

> **Note**
>
> EC2 インスタンスと RDS データベースが相互接続するためは、同じ VPC 内にある必要があります。

4. 接続 を選択します。

アニメーションを表示: 新しく起動した EC2 インスタンスを RDS データベースに自動接続する



タスク 4: 接続設定を検証する

このタスクでは、2 つのセキュリティグループが作成され、インスタンスとデータベースに割り当てられていることを確認します。

コンソールの自動接続機能を使用して接続を設定する場合、セキュリティグループは自動的に作成され、次のようにインスタンスとデータベースに割り当てられます。

- セキュリティグループ rds-ec2-*x* が作成され、RDS データベースに追加されます。ec2-rds-*x* セキュリティグループを送信元として参照するインバウンドルールが 1 つ存在します。このルールにより、ec2-rds-*x* セキュリティグループを持つ EC2 インスタンスからのトラフィックが RDS データベースに到達できるようになります。
- セキュリティグループ ec2-rds-*x* が作成され、EC2 インスタンスに追加されます。ここには、rds-ec2-*x* セキュリティグループを送信先として参照するアウトバウンドルールが 1 つ存在します。このルールにより、EC2 インスタンスからのトラフィックが rds-ec2-*x* セキュリティグループのある RDS データベースに到達できるようになります。

コンソールを使用して接続構成を確認する場合

1. Amazon RDS コンソール (https://console.aws.amazon.com/rds/) を開きます。
2. ナビゲーションページで、[Databases] (データベース) を選択します。
3. このチュートリアル用に作成した RDS データベースを選択します。
4. [接続とセキュリティ] タブの [セキュリティ] にある [VPC セキュリティグループ] に、[rds-ec2-*x*] というのセキュリティグループが表示されていることを確認します。
5. rds-ec2-*x* セキュリティグループを選択します。EC2 コンソールにある [Security Groups] (セキュリティグループ) 画面が開きます。
6. rds-ec2-*x* セキュリティグループを選択して開きます。
7. [Inbound rules] (インバウンドルール) タブを開きます。
8. 次のセキュリティグループルールが存在することを確認します。

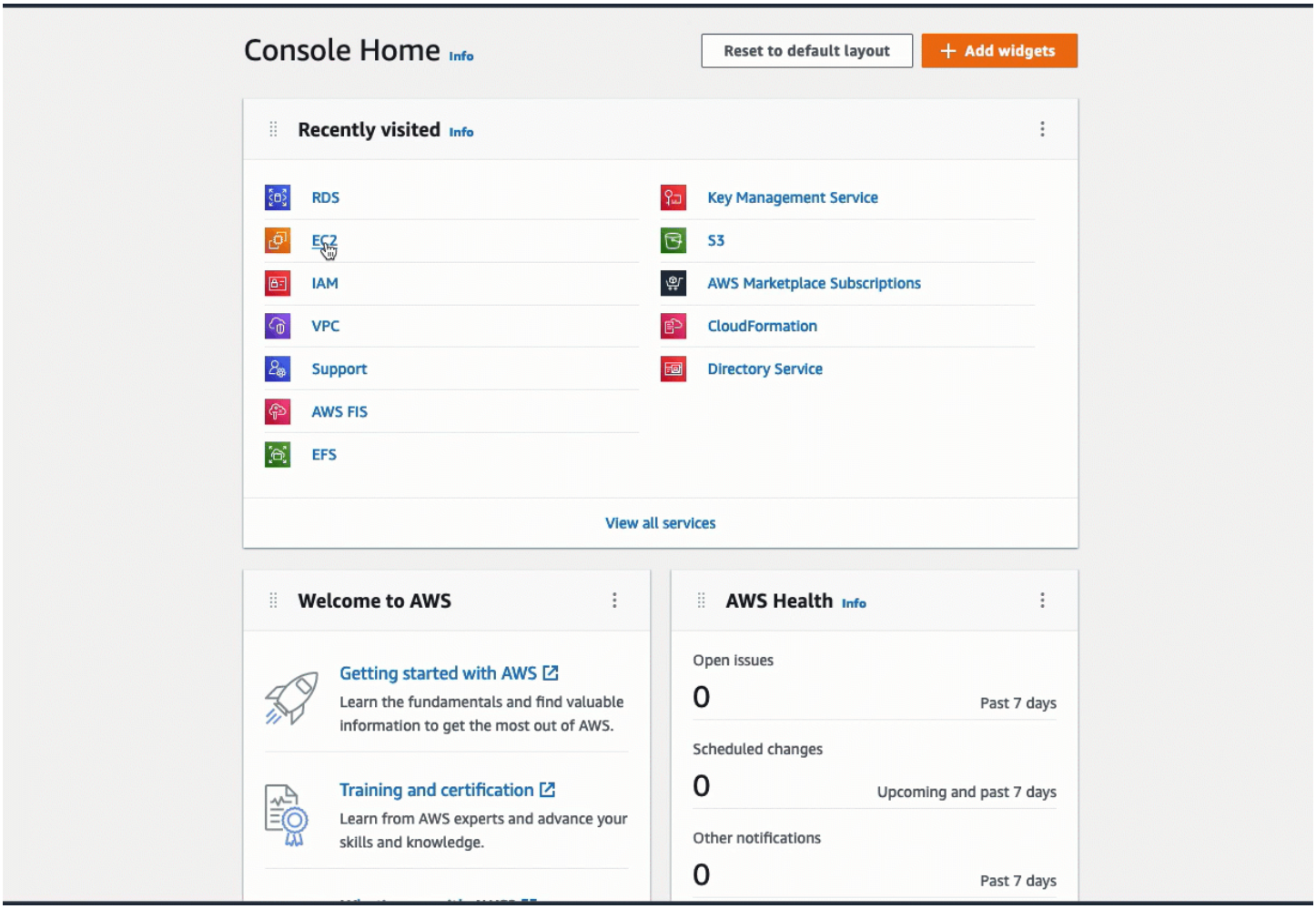
    - タイプ: MYSQL/Aurora
    - ポート範囲: 3306
    - ソース: ***sg-0987654321example*** / ec2-rds-*x* – これは、前述の手順で検証した EC2 インスタンスに割り当てられているセキュリティグループです。
    - 説明: ***sg-1234567890example*** が添付された EC2 インスタンスからの接続を許可するルール

9. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
10. ナビゲーションペインで、[インスタンス] を選択してください。
11. 前述のタスクで RDS データベースに接続するために選択した EC2 インスタンスを選択し、[Security] (セキュリティ) タブを選択します。
12. [セキュリティの詳細] の [セキュリティグループ] にあるリストの中に、[ec2-rds-*x*] というセキュリティグループがあることを確認します。*x* は数字です。
13. ec2-rds-*x* セキュリティグループを選択して開きます。
14. [Outbound rules] (アウトバウンドルール) タブを選択します。
15. 次のセキュリティグループルールが存在することを確認します。

    - タイプ: MYSQL/Aurora
    - ポート範囲: 3306
    - 送信先: ***sg-1234567890example*** / rds-ec2-*x*
    - 説明: このセキュリティグループがアタッチされている任意のインスタンスからの **database-tutorial** への接続を許可するルール

これらのセキュリティグループとセキュリティグループルールが存在し、この手順の記述にあるように RDS データベースと EC2 インスタンスに割り当てられていることを確認することで、自動接続機能を使用して接続が自動的に設定されたことを確認できます。

アニメーションを表示: 接続設定を確認する



これで、このチュートリアルのオプション 1 が完了しました。これで、RDS コンソールを使用して EC2 インスタンスを RDS データベースに自動接続する方法について説明するオプション 2 を完了するか、オプション 1 で自動的に作成されたセキュリティグループを手動で設定する方法について説明するオプション 3 を完了することができます。

タスク 5 (オプション): クリーンアップする

> **Warning**
>
> インスタンスの終了は恒久的で、元に戻すことができません。
> インスタンスを終了すると、そのインスタンスには接続できなくなり、復元することもできません。アタッチされている Amazon EBS ボリュームで、終了時に削除されるように設定されているものも、すべて恒久的に削除され、復元できません。インスタンスストアボリュームに保存されているデータは、すべて恒久的に失われます。詳細については「インスタンスの終了方法」を参照してください。

インスタンスを終了する前に、終了後も保持する必要があるすべてのデータを永続的ストレージにバックアップしたことを確認してください。

チュートリアルを完了したので、使用しなくなったリソースをすべてクリーンアップ (削除) することをおすすめします。AWS リソースをクリーンアップすることで、アカウントに追加料金が発生するのを防ぐことができます。

このチュートリアル専用に EC2 インスタンスを起動した場合、そのインスタンスを終了することで、それに関連した料金が発生するのを防ぐことができます。

コンソールを使用してインスタンスを終了するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. このチュートリアルで作成したインスタンスを選択し、[Instance state] (インスタンスの状態)、[Terminate instance] (インスタンスの終了) の順に選択します。
4. 確認を求めるメッセージが表示されたら、[Terminate (終了)] を選択してください。

このチュートリアル専用に RDS データベースを作成した場合、削除することで、それに関連した料金が発生するのを防ぐことができます。

コンソールを使用して RDS データベースを削除するには

1. Amazon RDS コンソール (https://console.aws.amazon.com/rds/) を開きます。
2. ナビゲーションペインで、[Databases] (データベース) を選択します。
3. このチュートリアル用に作成した RDS データベースを選択し、[Actions] (アクション)、[Delete] (削除) を選択します。
4. ボックスに **`delete me`** を入力し、[Delete] (削除) をクリックします。

オプション 2: RDS コンソールを使用して、インスタンスを RDS データベースに自動的に接続する

オプション 2 では、RDS コンソールの自動接続機能を使用して、EC2 インスタンスと RDS データベース間のトラフィックを許可するように EC2 インスタンスと RDS データベース間の接続を自動的に設定します。オプション 3 では、この接続を手動で設定する方法について説明します。

タスク

- [開始する前に]
- タスク 1 (オプション): EC2 インスタンスを起動する
- タスク 2: RDS データベースを作成し、それを EC2 インスタンスに自動接続する
- タスク 3: 接続設定を検証する
- タスク 4 (オプション): クリーンアップする

[開始する前に]

このチュートリアルを完了するには、以下が必要です。

- RDS データベースと同じ VPC にある EC2 インスタンス。既存の EC2 インスタンスを使用することも、タスク 1 のステップに従って新しいインスタンスを作成することもできます。
- 次の操作を呼び出すアクセス許可:
  - `ec2:AssociateRouteTable`
  - `ec2:AuthorizeSecurityGroupEgress`
  - `ec2:CreateRouteTable`
  - `ec2:CreateSecurityGroup`
  - `ec2:CreateSubnet`
  - `ec2:DescribeInstances`
  - `ec2:DescribeNetworkInterfaces`
  - `ec2:DescribeRouteTables`
  - `ec2:DescribeSecurityGroups`
  - `ec2:DescribeSubnets`
  - `ec2:ModifyNetworkInterfaceAttribute`
  - `ec2:RevokeSecurityGroupEgress`

タスク 1 (オプション): EC2 インスタンスを起動する

> **Note**
>
> インスタンスの起動は、このチュートリアルの対象外です。Amazon EC2 インスタンスがすでにあり、このチュートリアルで使用する場合は、このタスクをスキップできます。

このタスクでは、EC2 インスタンスを起動して、EC2 インスタンスと使用する Amazon RDS データベース間の接続を設定するタスク 2 を完了できるようにします。このタスクのステップでは、EC2 インスタンスを次のように設定します。

- インスタンス名: **`tutorial-instance-2`**
- AMI: Amazon Linux 2
- インスタンスタイプ: `t2.micro`
- パブリック IP の自動割り当て: 有効
- 次の 3 つのルールを持つセキュリティグループ:
  - IP アドレスからの SSH を許可
  - 任意の場所からの HTTPS トラフィックを許可
  - 任意の場所からの HTTP トラフィックを許可

> **Important**
>
> 本番環境では、具体的なニーズに合わせて、インスタンスを設定する必要があります。

EC2 インスタンスを起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. [EC2 Dashboard] (EC2 ダッシュボード) で、[Launch instance] (インスタンスの作成) を選択します。
3. [Names and tags] (名前とタグ) にある [Name] (名前) には、インスタンスを識別するための名前を入力します。このチュートリアルでは、インスタンス名は「**`tutorial-instance-2`**」にします。インスタンス名は必須ではありませんが、RDS コンソールでインスタンスを選択するときに、名前があると識別しやすくなります。

4. [Application and OS Images] (アプリケーションと OS イメージ) で、ウェブサーバーに必要な AMI を選択します。このチュートリアルでは Amazon Linux を使用します。
5. [Instance type] (インスタンスタイプ) にある [Instance type] (インスタンスタイプ) には使用しているウェブサーバーに必要なインスタンスタイプを選択します。このチュートリアルでは、`t2.micro` を使用します。

   > **Note**
   >
   > アカウントの作成日に応じて、Amazon EC2 を無料利用枠で使用できる場合があります。
   > 2025 年 7 月 15 日より前に AWS アカウントを作成していて、12 か月が経過していない場合は、[t2.micro] インスタンスタイプ ([t2.micro] が利用できないリージョンでは [t3.micro] インスタンスタイプ) を選択することで、Amazon EC2 の無料利用枠を使用できます。[t3.micro] インスタンスを起動すると、デフォルトで [Unlimited] モードになり、CPU 使用率に基づいて追加料金が発生する可能性があることに注意してください。インスタンスタイプを無料利用枠で使用できる場合、[無料利用枠の対象] とラベル付けされます。
   > 2025 年 7 月 15 日以降に AWS アカウント を作成した場合、t3.micro、t3.small、t4g.micro、t4g.small、c7i-flex.large、m7i-flex.large インスタンスタイプを 6 か月間、またはクレジットを使い切るまで使用できます。
   > 詳細については「2025 年 7 月 15 日より前とそれ以降の無料利用枠の特典」を参照してください。

6. [Key pair (login)] (キーペア (ログイン)) にある [Key pair name] (キーペア名) には、使用するキーペアを選択します。
7. [Network settings] (ネットワーク設定) で、次の操作を行います：

   a. デフォルトの VPC またはサブネットに変更を加えていない場合は、[Network] (ネットワーク) と [Subnet] (サブネット) でデフォルトの設定をそのまま使用できます。

      デフォルトの VPC またはサブネットに変更を加えた場合は、以下を確認してください。

      i. 自動接続設定を使用するには、インスタンスが RDS データベースと同じ VPC 内に存在している必要があります。デフォルトでは、VPC は 1 つのみです。

      ii. インスタンスを起動する VPC には、インターネットからウェブサーバーにアクセスできるように、インターネットゲートウェイがアタッチされている必要があります。デフォルト VPC はインターネットゲートウェイで自動的に設定されます。

iii. インスタンスがパブリック IP アドレスを受け取れるように、[Auto-assign Public IP] (自動割り当てパブリック IP) で [Enable] (有効) が選択されていることを確認します。[Disable] (無効) が選択されている場合は、[Edit] (編集) ([Network Settings] (ネットワーク設定) の右側) を選択し、その後 [Auto-assign public IP]] (パブリック IP の自動割り当て) で [Enable] (有効) を選択します。

b. SSH を使用してインスタンスに接続するには、コンピュータのパブリック IPv4 アドレスからの SSH (Linux) または RDP (Windows) トラフィックを承認するセキュリティグループのルールが必要です。デフォルトでは、インスタンスを起動すると、任意の場所からのインバウンド SSH トラフィックを許可するルールで新しいセキュリティグループが作成されます。

お使いの IP アドレスのみがインスタンスに接続できるようにするには、[ファイアウォール (セキュリティグループ)] にある [SSH トラフィックを許可する送信元] チェックボックスの横にあるドロップダウンリストから [マイ IP] を選択します。

c. インターネットからインスタンスへのトラフィックを許可するには、次のチェックボックスをオンにします。

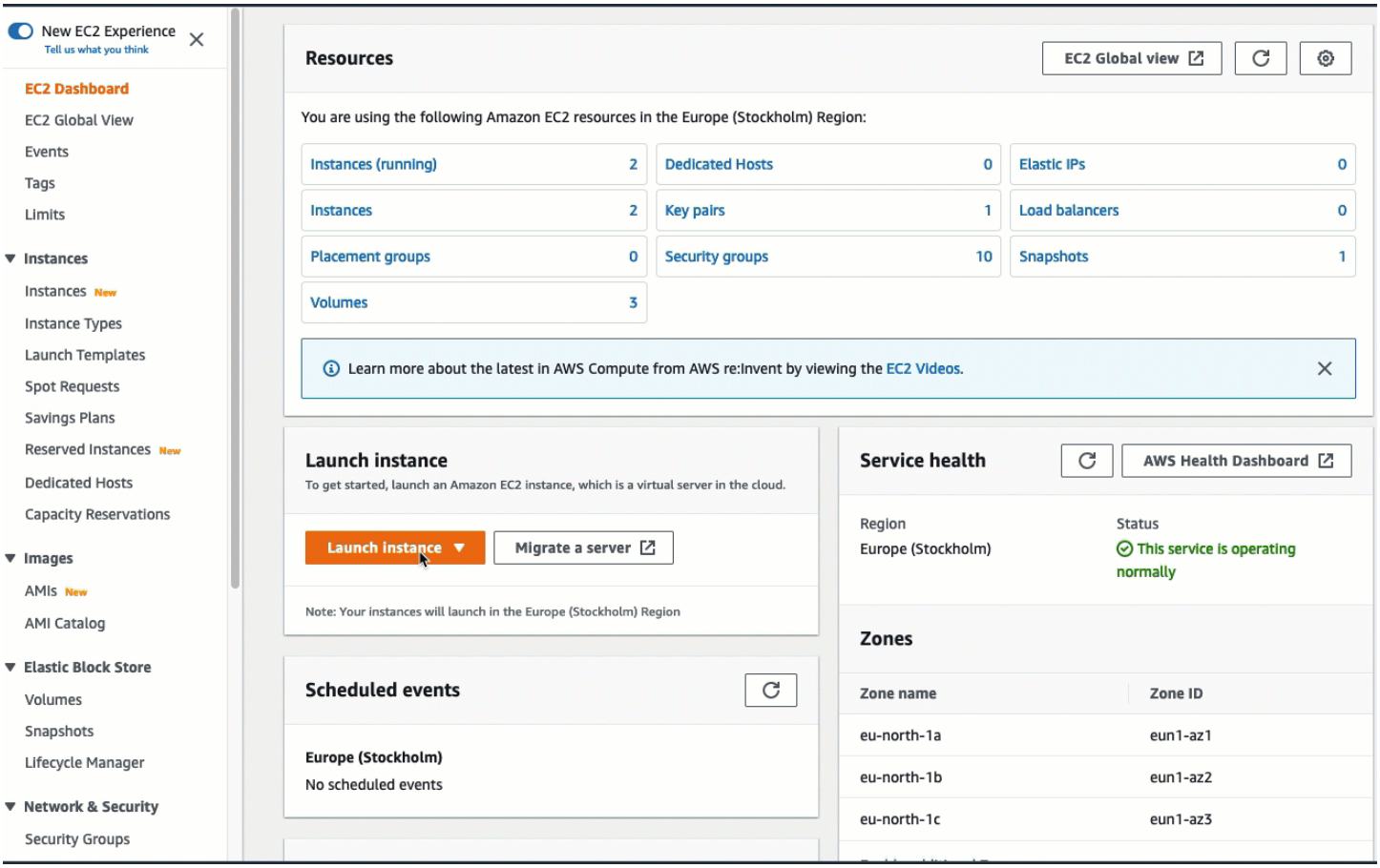
- [Allow HTTPs traffic from the internet] (インターネットからの HTTPs トラフィックを許可する)
- [Allow HTTP traffic from the internet] (インターネットからの HTTP トラフィックを許可する)

8. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動する) を選択します。
9. [View all instances] (すべてのインスタンスの表示) を選択して確認ページを閉じ、コンソールに戻ります。インスタンスは最初 `pending` 状態になり、その後 `running` 状態になります。

インスタンスが起動しないか、状態が `terminated` ではなくすぐに `running` になる場合は[Amazon EC2 インスタンスの起動に関する問題のトラブルシューティング]を参照してください。

インスタンスの起動方法の詳細については、「[コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する]」を参照してください。

アニメーションを表示: EC2 インスタンスを起動する



タスク 2: RDS データベースを作成し、それを EC2 インスタンスに自動接続する

このタスクでは、RDS データベースを作成して RDS コンソールの自動接続機能を使用し、EC2 インスタンスと RDS データベース間の接続を自動的に設定します。このタスクのステップでは、DB インスタンスを次のように設定します。

- エンジンタイプ: MySQL
- テンプレート: 無料利用枠
- DB インスタンス識別子: **`tutorial-database`**
- DB インスタンスクラス: `db.t3.micro`

> **Important**
>
> 本番環境では、具体的なニーズに合わせて、インスタンスを設定する必要があります。

RDS データベースを作成して EC2 インスタンスに自動接続する

1. Amazon RDS コンソール (https://console.aws.amazon.com/rds/) を開きます。
2. リージョンセレクター (右上) から、EC2 インスタンスを作成した AWS リージョン を選択します。EC2 インスタンスと RDS データベースは同じリージョンにある必要があります。
3. ダッシュボード で、[Create database] (データベースの作成) を選択します。
4. [Choose a database creation method] (データベース作成方法を選択) で [Standard Create] (スタンダード作成) が選択されていることを確認します。[Easy create] (簡易作成) を選択した場合、自動接続機能は使用できません。
5. [Engine options] (エンジンオプション) にある [Engine type] (エンジンタイプ) で MySQL を選択します。
6. [Templates] (テンプレート) では、ニーズに合うサンプルテンプレートを選択します。このチュートリアルでは、[Free tier] (無料利用枠) を選択して RDS データベースを無料で作成できます。ただし、無料利用枠は、アカウントが無料利用枠の対象である場合にのみ使用できることに注意してください。詳細については、[Free tier] (無料利用枠) ボックスにある [Info] (情報) リンクを選択してください。
7. [設定] で、次のいずれかを実行します。

   a. [DB instance identifier] (DB インスタンス識別子) に、データベースの名前を入力します。このチュートリアルでは、**`tutorial-database`** と入力します。

   b. [Master username] (マスターユーザー名) は、デフォルトの名前である **`admin`** のままにします。

   c. [Master password] (マスターパスワード) に、このチュートリアルに使用するパスワードを入力し、[Confirm password] (パスワードの確認) にパスワードをもう一度入力します。
8. [インスタンス設定] の [DB インスタンスクラス] は、デフォルトの [db.t3.micro] のままにします。アカウントが無料利用枠の対象である場合は、このインスタンスを無料で使用できます。詳細については、「AWS 無料利用枠」を参照してください。
9. [Connectivity] (接続) にある [Compute resource] (コンピューティングリソース) で、[Connect to an EC2 compute resource] (EC2 コンピューティングリソースに接続する) を選択します。これは RDS コンソールの自動接続機能です。
10. [EC2 instance] (EC2 インスタンス) で、接続先のインスタンスを選択します。このチュートリアルでは、前のタスクで作成して **`tutorial-instance`** と名付けたインスタンスを選択することも、別の既存のインスタンスを選択することもできます。リストにインスタンスが表示されない場合は、[Connectivity] (接続) の右にある更新アイコンを選択します。

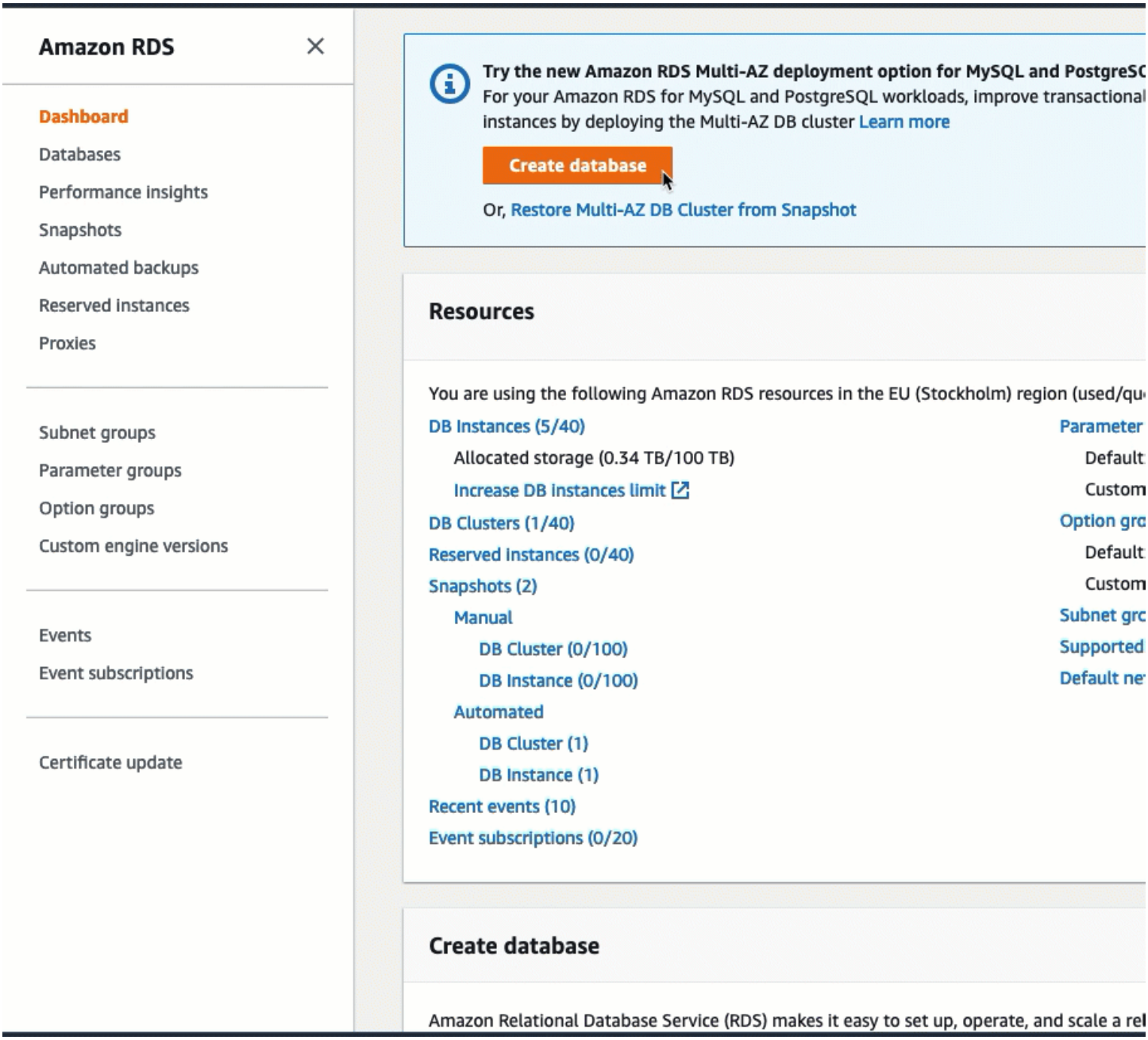
自動接続機能を使用すると、この EC2 インスタンスにセキュリティグループが追加され、RDS データベースに別のセキュリティグループが追加されます。セキュリティグループは、EC2 インスタンスと RDS データベース間のトラフィックを許可するように自動的に設定されます。次のタスクでは、セキュリティグループが作成され、EC2 インスタンスと RDS データベースに割り当てられていることを確認します。

11. [データベースの作成] を選択します。

    [Databases] (データベース) 画面では、データベースが使用可能になるまで、新しいデータベースの [Status] (ステータス) は [Creating] (作成中) になります。ステータスが [Available] (使用可能) に変わったら、データベースに接続できます。データベースクラスとストレージ量によっては、新しいデータベースが使用可能になるまで最長 20 分かかることがあります。

詳細については、「Amazon RDS ユーザーガイド」の「EC2 インスタンスとの自動ネットワーク接続を設定する」を参照してください。

アニメーションを表示: RDS データベースを作成して、EC2 インスタンスに自動接続する



タスク 3: 接続設定を検証する

このタスクでは、2 つのセキュリティグループが作成され、インスタンスとデータベースに割り当てられていることを確認します。

コンソールの自動接続機能を使用して接続を設定する場合、セキュリティグループは自動的に作成され、次のようにインスタンスとデータベースに割り当てられます。

- セキュリティグループ rds-ec2-*x* が作成され、RDS データベースに追加されます。ec2-rds-*x* セキュリティグループを送信元として参照するインバウンドルールが 1 つ存在します。このルールにより、ec2-rds-*x* セキュリティグループを持つ EC2 インスタンスからのトラフィックが RDS データベースに到達できるようになります。
- セキュリティグループ ec2-rds-*x* が作成され、EC2 インスタンスに追加されます。ここには、rds-ec2-*x* セキュリティグループを送信先として参照するアウトバウンドルールが 1 つ存在します。このルールにより、EC2 インスタンスからのトラフィックが rds-ec2-*x* セキュリティグループのある RDS データベースに到達できるようになります。

コンソールを使用して接続構成を確認する場合

1. Amazon RDS コンソール (https://console.aws.amazon.com/rds/) を開きます。
2. ナビゲーションページで、[Databases] (データベース) を選択します。
3. このチュートリアル用に作成した RDS データベースを選択します。
4. [接続とセキュリティ] タブの [セキュリティ] にある [VPC セキュリティグループ] に、[rds-ec2-*x*] というのセキュリティグループが表示されていることを確認します。
5. rds-ec2-*x* セキュリティグループを選択します。EC2 コンソールにある [Security Groups] (セキュリティグループ) 画面が開きます。
6. rds-ec2-*x* セキュリティグループを選択して開きます。
7. [Inbound rules] (インバウンドルール) タブを開きます。
8. 次のセキュリティグループルールが存在することを確認します。

   - タイプ: MYSQL/Aurora
   - ポート範囲: 3306
   - ソース: ***sg-0987654321example*** / ec2-rds-*x* – これは、前述の手順で検証した EC2 インスタンスに割り当てられているセキュリティグループです。
   - 説明: ***sg-1234567890example*** が添付された EC2 インスタンスからの接続を許可するルール

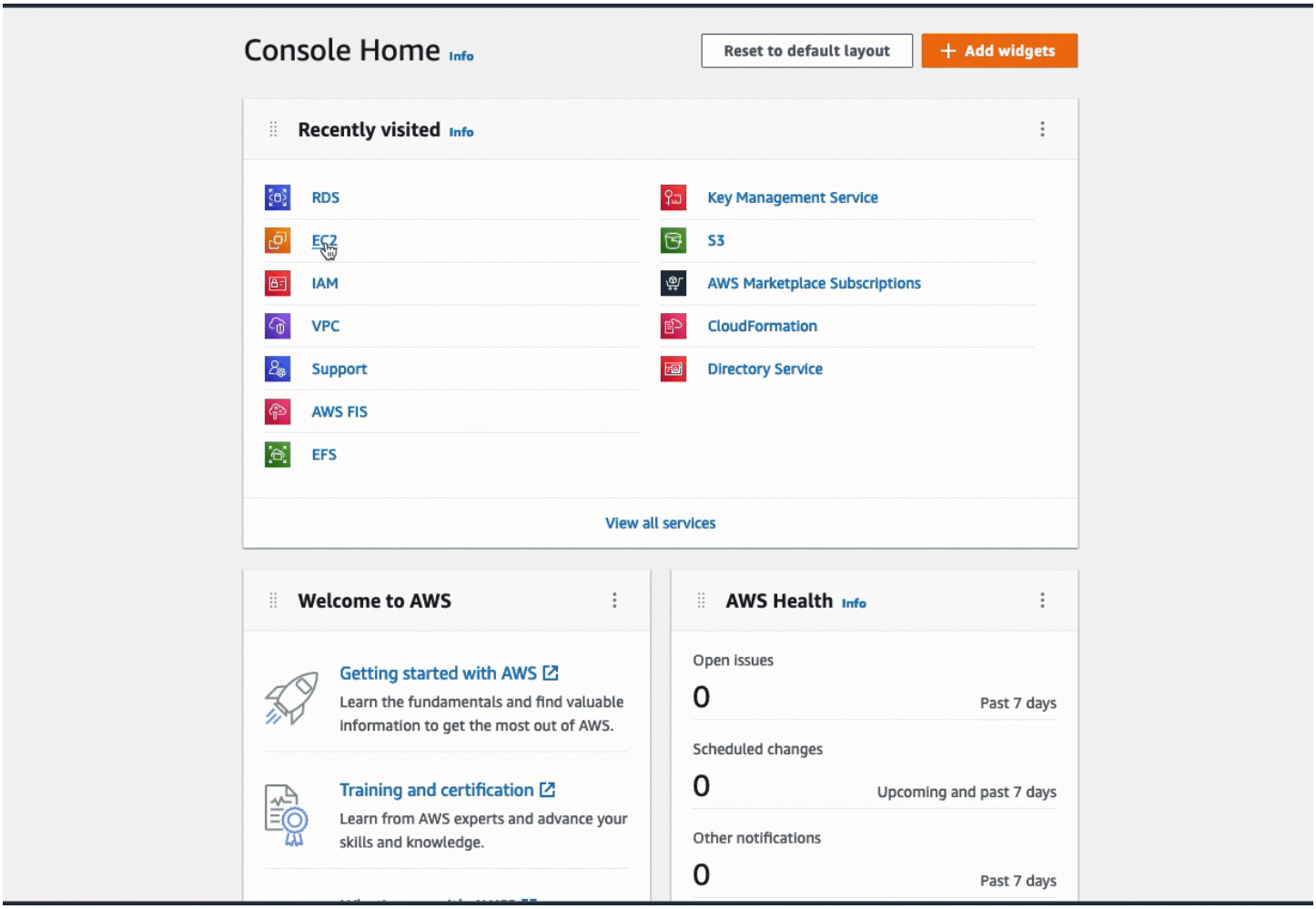
9. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
10. ナビゲーションペインで、[インスタンス] を選択してください。
11. 前述のタスクで RDS データベースに接続するために選択した EC2 インスタンスを選択し、[Security] (セキュリティ) タブを選択します。
12. [セキュリティの詳細] の [セキュリティグループ] にあるリストの中に、[ec2-rds-*x*] というセキュリティグループがあることを確認します。*x* は数字です。

13. ec2-rds-*x* セキュリティグループを選択して開きます。
14. [Outbound rules] (アウトバウンドルール) タブを選択します。
15. 次のセキュリティグループルールが存在することを確認します。

    - タイプ: MYSQL/Aurora
    - ポート範囲: 3306
    - 送信先: ***sg-1234567890example*** / rds-ec2-*x*
    - 説明: このセキュリティグループがアタッチされている任意のインスタンスからの **database-tutorial** への接続を許可するルール

これらのセキュリティグループとセキュリティグループルールが存在し、この手順の記述にあるように RDS データベースと EC2 インスタンスに割り当てられていることを確認することで、自動接続機能を使用して接続が自動的に設定されたことを確認できます。

アニメーションを表示: 接続設定を確認する



これで、このチュートリアルのオプション 2 が完了しました。これで、オプション 2 で自動的に作成されたセキュリティグループを手動で設定する方法を説明するオプション 3 を完了することもできます。

タスク 4 (オプション): クリーンアップする

> **Warning**
>
> インスタンスの終了は恒久的で、元に戻すことができません。
> インスタンスを終了すると、そのインスタンスには接続できなくなり、復元することもできません。アタッチされている Amazon EBS ボリュームで、終了時に削除されるように設定されているものも、すべて恒久的に削除され、復元できません。インスタンスストアボリュームに保存されているデータは、すべて恒久的に失われます。詳細については「インスタンスの終了方法」を参照してください。
> インスタンスを終了する前に、終了後も保持する必要があるすべてのデータを永続的ストレージにバックアップしたことを確認してください。

チュートリアルを完了したので、使用しなくなったリソースをすべてクリーンアップ (削除) することをおすすめします。AWS リソースをクリーンアップすることで、アカウントに追加料金が発生するのを防ぐことができます。

このチュートリアル専用に EC2 インスタンスを起動した場合、そのインスタンスを終了することで、それに関連した料金が発生するのを防ぐことができます。

コンソールを使用してインスタンスを終了するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. このチュートリアルで作成したインスタンスを選択し、[Instance state] (インスタンスの状態)、[Terminate instance] (インスタンスの終了) の順に選択します。
4. 確認を求めるメッセージが表示されたら、[Terminate (終了)] を選択してください。

このチュートリアル専用に RDS データベースを作成した場合、削除することで、それに関連した料金が発生するのを防ぐことができます。

コンソールを使用して RDS データベースを削除するには

1. Amazon RDS コンソール (https://console.aws.amazon.com/rds/) を開きます。
2. ナビゲーションペインで、[Databases] (データベース) を選択します。
3. このチュートリアル用に作成した RDS データベースを選択し、[Actions] (アクション)、[Delete] (削除) を選択します。
4. ボックスに **`delete me`** を入力し、[Delete] (削除) をクリックします。

オプション 3: セキュリティグループを作成して、インスタンスを RDS データベースに手動で接続する

オプション 3 では、自動接続機能の設定を手動で再現することで、EC2 インスタンスと RDS データベース間の接続を手動で設定する方法を説明します。

タスク

- [開始する前に]
- タスク 1 (オプション): EC2 インスタンスを起動する
- タスク 2 (オプション): RDS データベースを作成する
- タスク 3: セキュリティグループを作成してインスタンスに割り当てることで、EC2 インスタンスを RDS データベースに手動で接続する
- タスク 4 (オプション): クリーンアップする

[開始する前に]

このチュートリアルを完了するには、以下が必要です。

- RDS データベースと同じ VPC にある EC2 インスタンス。既存の EC2 インスタンスを使用することも、タスク 1 のステップに従って新しいインスタンスを作成することもできます。
- EC2 インスタンスと同じ VPC にある RDS データベース。既存の RDS データベースを使用することも、タスク 2 のステップに従って新しいデータベースを作成することもできます。
- 次の操作を呼び出すアクセス許可:
  - `ec2:AssociateRouteTable`
  - `ec2:AuthorizeSecurityGroupEgress`
  - `ec2:CreateRouteTable`

- ec2:CreateSecurityGroup
- ec2:CreateSubnet
- ec2:DescribeInstances
- ec2:DescribeNetworkInterfaces
- ec2:DescribeRouteTables
- ec2:DescribeSecurityGroups
- ec2:DescribeSubnets
- ec2:ModifyNetworkInterfaceAttribute
- ec2:RevokeSecurityGroupEgress

タスク 1 (オプション): EC2 インスタンスを起動する

> **Note**
>
> インスタンスの起動は、このチュートリアルの対象外です。Amazon EC2 インスタンスがすでにあり、このチュートリアルで使用する場合は、このタスクをスキップできます。

このタスクでは、EC2 インスタンスを起動して、EC2 インスタンスと使用する Amazon RDS データベース間の接続を設定するタスク 3 を完了できるようにします。このタスクのステップでは、EC2 インスタンスを次のように設定します。

- インスタンス名: **tutorial-instance**
- AMI: Amazon Linux 2
- インスタンスタイプ: t2.micro
- パブリック IP の自動割り当て: 有効
- 次の 3 つのルールを持つセキュリティグループ:
  - IP アドレスからの SSH を許可
  - 任意の場所からの HTTPS トラフィックを許可
  - 任意の場所からの HTTP トラフィックを許可

> **Important**
>
> 本番環境では、具体的なニーズに合わせて、インスタンスを設定する必要があります。

EC2 インスタンスを起動するには

1. AWS マネジメントコンソール にサインインし、Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [EC2 Dashboard] (EC2 ダッシュボード) で、[Launch instance] (インスタンスの作成) を選択します。
3. [Names and tags] (名前とタグ) にある [Name] (名前) には、インスタンスを識別するための名前を入力します。このチュートリアルでは、インスタンス名は「**`tutorial-instance-manual-1`**」にします。インスタンス名は必須ではありませんが、名前があると識別しやすくなります。
4. [Application and OS Images] (アプリケーションと OS イメージ) で、ウェブサーバーに必要な AMI を選択します。このチュートリアルでは Amazon Linux を使用します。
5. [Instance type] (インスタンスタイプ) にある [Instance type] (インスタンスタイプ) には使用しているウェブサーバーに必要なインスタンスタイプを選択します。このチュートリアルでは、`t2.micro` を使用します。

   > **Note**
   >
   > アカウントの作成日に応じて、Amazon EC2 を無料利用枠で使用できる場合があります。
   > 2025 年 7 月 15 日より前に AWS アカウントを作成していて、12 か月が経過していない場合は、[t2.micro] インスタンスタイプ ([t2.micro] が利用できないリージョンでは [t3.micro] インスタンスタイプ) を選択することで、Amazon EC2 の無料利用枠を使用できます。[t3.micro] インスタンスを起動すると、デフォルトで [Unlimited] モードになり、CPU 使用率に基づいて追加料金が発生する可能性があることに注意してください。インスタンスタイプを無料利用枠で使用できる場合、[無料利用枠の対象] とラベル付けされます。
   > 2025 年 7 月 15 日以降に AWS アカウント を作成した場合、t3.micro、t3.small、t4g.micro、t4g.small、c7i-flex.large、m7i-flex.large インスタンスタイプを 6 か月間、またはクレジットを使い切るまで使用できます。

詳細については「2025 年 7 月 15 日より前とそれ以降の無料利用枠の特典」を参照してください。

6. [Key pair (login)] (キーペア (ログイン)) にある [Key pair name] (キーペア名) には、使用するキーペアを選択します。
7. [Network settings] (ネットワーク設定) で、次の操作を行います：

   a. デフォルトの VPC またはサブネットに変更を加えていない場合は、[Network] (ネットワーク) と [Subnet] (サブネット) でデフォルトの設定をそのまま使用できます。

      デフォルトの VPC またはサブネットに変更を加えた場合は、以下を確認してください。

      i. インスタンスは、RDS データベースと同じ VPC 内に存在する必要があります。デフォルトでは、VPC は 1 つのみです。

      ii. インスタンスを起動する VPC には、インターネットからウェブサーバーにアクセスできるように、インターネットゲートウェイがアタッチされている必要があります。デフォルト VPC はインターネットゲートウェイで自動的に設定されます。

      iii. インスタンスがパブリック IP アドレスを受け取れるように、[Auto-assign Public IP] (自動割り当てパブリック IP) で [Enable] (有効) が選択されていることを確認します。[Disable] (無効) が選択されている場合は、[Edit] (編集) ([Network Settings] (ネットワーク設定) の右側) を選択し、その後 [Auto-assign public IP]] (パブリック IP の自動割り当て) で [Enable] (有効) を選択します。

   b. SSH を使用してインスタンスに接続するには、コンピュータのパブリック IPv4 アドレスからの SSH (Linux) または RDP (Windows) トラフィックを承認するセキュリティグループのルールが必要です。デフォルトでは、インスタンスを起動すると、任意の場所からのインバウンド SSH トラフィックを許可するルールで新しいセキュリティグループが作成されます。

      お使いの IP アドレスのみがインスタンスに接続できるようにするには、[ファイアウォール (セキュリティグループ)] にある [SSH トラフィックを許可する送信元] チェックボックスの横にあるドロップダウンリストから [マイ IP] を選択します。

   c. インターネットからインスタンスへのトラフィックを許可するには、次のチェックボックスをオンにします。

      - [Allow HTTPs traffic from the internet] (インターネットからの HTTPs トラフィックを許可する)

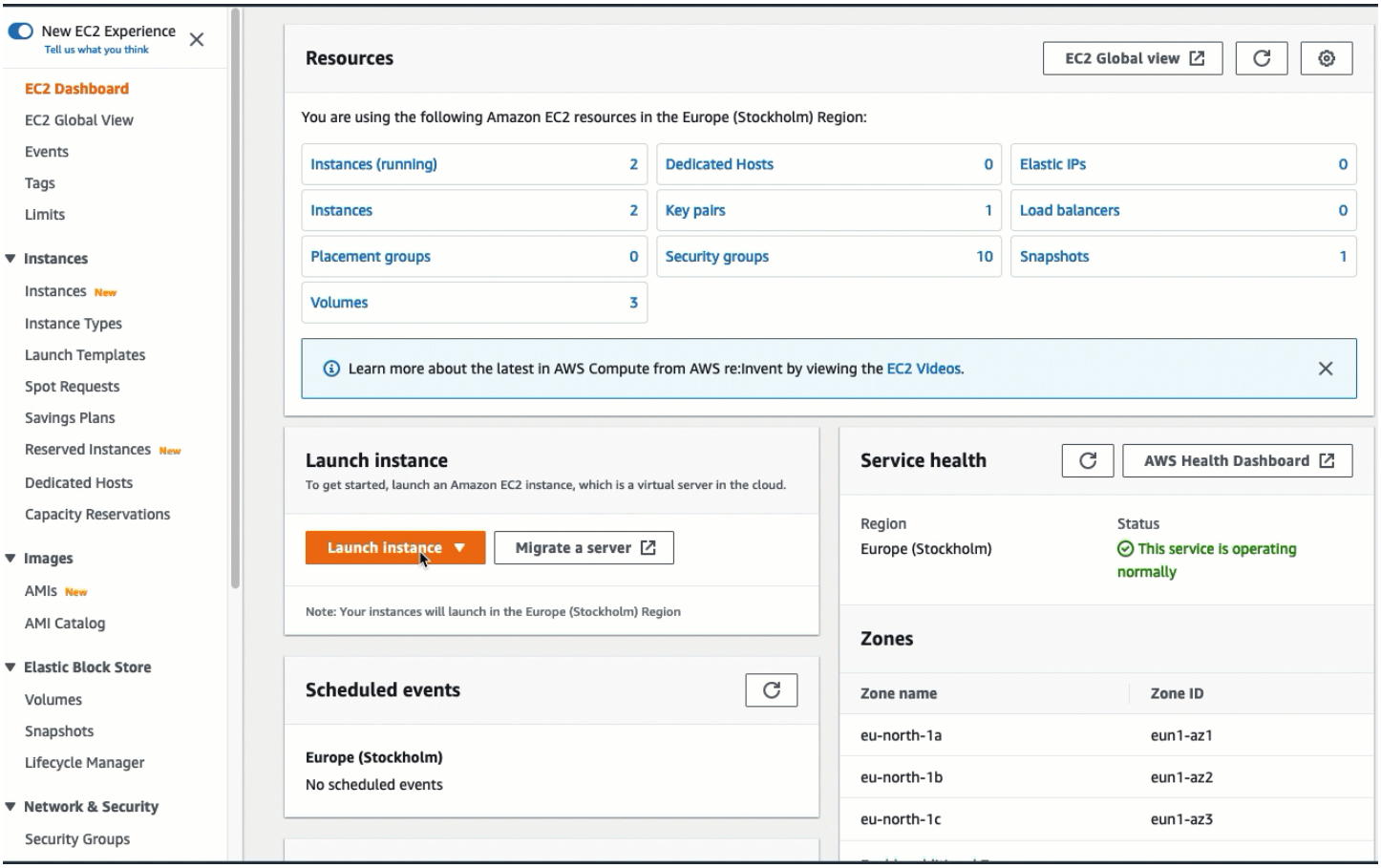
- [Allow HTTP traffic from the internet] (インターネットからの HTTP トラフィックを許可する)

8. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動する) を選択します。
9. [View all instances] (すべてのインスタンスの表示) を選択して確認ページを閉じ、コンソールに戻ります。インスタンスは最初 `pending` 状態になり、その後 `running` 状態になります。

   インスタンスが起動しないか、状態が `terminated` ではなくすぐに `running` になる場合はAmazon EC2 インスタンスの起動に関する問題のトラブルシューティングを参照してください。

インスタンスの起動方法の詳細については、「コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する」を参照してください。

アニメーションを表示: EC2 インスタンスを起動する



タスク 2 (オプション): RDS データベースを作成する

> **Note**
>
> RDS データベースの作成については、このチュートリアルでカバーしません。RDS データベースがすでにあり、このチュートリアルで使用する場合は、このタスクをスキップできます。

このタスクでは、RDS データベースを作成します。このインスタンスは、EC2 インスタンスに接続するタスク 3 で使用します。このタスクのステップでは、RDS データベースを次のように設定します。

- エンジンタイプ: MySQL
- テンプレート: 無料利用枠
- DB インスタンス識別子: **`tutorial-database-manual`**
- DB インスタンスクラス: `db.t3.micro`

> **Important**
>
> 本番環境では、具体的なニーズに合わせて、インスタンスを設定する必要があります。

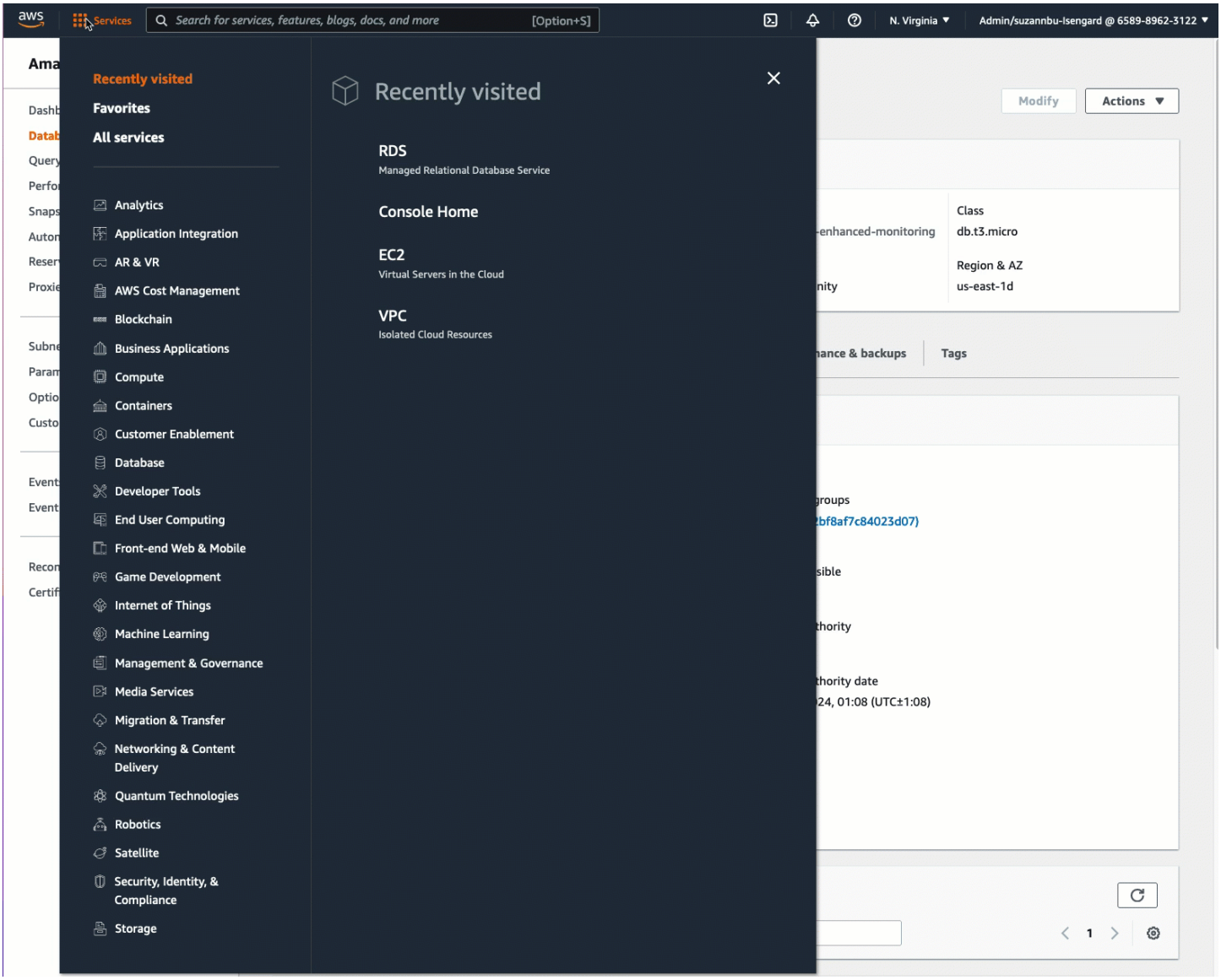
MySQL DB インスタンスを作成するには

1. Amazon RDS コンソール (https://console.aws.amazon.com/rds/) を開きます。
2. リージョンセレクター (右上) から、EC2 インスタンスを作成した AWS リージョン を選択します。EC2 インスタンスと DB インスタンスは同じリージョンに存在する必要があります。
3. ダッシュボード で、[Create database] (データベースの作成) を選択します。
4. [Choose a database creation method] (データベース作成方法を選択) で [Easy Create] (簡易作成) を選択します。このオプションを選択すると、接続を自動的に構成する自動接続機能は使用できません。
5. [Engine options] (エンジンオプション) にある [Engine type] (エンジンタイプ) で MySQL を選択します。
6. [DB インスタンスサイズ] で、[無料利用枠] を選択します。

7. [DB instance identifier] (DB インスタンス識別子) に、RDS データベースの名前を入力します。このチュートリアルでは、**tutorial-database-manual** と入力します。
8. [Master username] (マスターユーザー名) は、デフォルトの名前である **admin** のままにします。
9. [Master password] (マスターパスワード) に、このチュートリアルに使用するパスワードを入力し、[Confirm password] (パスワードの確認) にパスワードをもう一度入力します。
10. [データベースの作成] を選択します。

    [Databases] (データベース) 画面では、DB インスタンスが使用可能な状態になるまで、新しい DB インスタンスの [Status] (ステータス) は [Creating] (作成中) になります。ステータスが [Available] (利用可能) に変わったら、DB インスタンスに接続できます。DB インスタンスクラスとストレージの合計によっては、新しいインスタンスを使用できるようになるまで最長 20 分かかることがあります。

アニメーションを表示: DB インスタンスの作成



タスク 3: セキュリティグループを作成してインスタンスに割り当てることで、EC2 インスタンスを RDS データベースに手動で接続する

このタスクでは、次の手順を手動で実行して、自動接続機能の接続設定を再現します: 新しいセキュリティグループを 2 つ作成し、それぞれのセキュリティグループを EC2 インスタンスと RDS データベースに追加します。

2 つの新しいセキュリティグループを作成し、それぞれを EC2 インスタンスと RDS データベースに割り当てるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

2. まずは、EC2 インスタンスに追加するセキュリティグループを次のように作成します:

   a. ナビゲーションペインで、[Security Groups] を選択してください。

   b. [セキュリティグループの作成] を選択してください。

   c. [Security group name] (セキュリティグループ名) には、分かりやすいセキュリティグループ名を入力します。このチュートリアルでは、**ec2-rds-manual-configuration** と入力します。

   d. [Description] (説明) に、簡単な説明を入力します。このチュートリアルでは、**EC2 instance security group to allow EC2 instance to securely connect to RDS database** と入力します。

   e. [セキュリティグループの作成] を選択してください。RDS データベースのセキュリティグループを作成したら、このセキュリティグループに戻ってアウトバウンドルールを追加します。

3. 次に、以下の操作を行い、RDS データベースに追加するセキュリティグループを作成します。

   a. ナビゲーションペインで、[Security Groups] を選択してください。

   b. [セキュリティグループの作成] を選択してください。

   c. [Security group name] (セキュリティグループ名) には、分かりやすいセキュリティグループ名を入力します。このチュートリアルでは、**rds-ec2-manual-configuration** と入力します。

   d. [Description] (説明) に、簡単な説明を入力します。このチュートリアルでは、**RDS database security group to allow EC2 instance to securely connect to RDS database** と入力します。

   e. [Inbound rules] (インバウンドルール) タブで [Add rule] (ルールの追加) を選択し、以下の操作を行います。

      i. [Type] (タイプ) では MySQL/Aurora を選択します。

      ii. [Source] (ソース) には、この手順のステップ 2 で作成した EC2 インスタンスのセキュリティグループ ec2-rds-manual-configuration を選択します。

   f. [セキュリティグループの作成] を選択してください。

4. 次のように、EC2 インスタンスのセキュリティグループを編集して、アウトバウンドルールを追加します。

   a. ナビゲーションペインで、[Security Groups] を選択してください。

b. EC2 インスタンスのセキュリティグループ (**`ec2-rds-manual-configuration`** と名前を付けたもの) を選択し、[Outbound rules] (アウトバウンドルール) タブを選択します。

c. [Edit outbound rules] (アウトバウンドルールの編集) を選択します。

d. [Add rule] (ルールの追加) を選択し、次の操作を行います。

i. [Type] (タイプ) では MySQL/Aurora を選択します。

ii. [送信先] には、この手順のステップ 3 で作成した RDS データベースのセキュリティグループ rds-ec2-manual-configuration を選択します。

iii. [ルールの保存] を選択します。

5. EC2 インスタンスのセキュリティグループを、次のように EC2 インスタンスに追加します:

a. ナビゲーションペインで、[インスタンス] を選択してください。

b. EC2 インスタンスを選択し、[Actions] (アクション)、[Security] (セキュリティ)、[Change security groups] (セキュリティグループの変更) の順に選択します。

c. [Associated security groups] (関連するセキュリティグループ) で [Select security groups] (セキュリティグループの選択) フィールドを選択し、先程作成した ec2-rds-manual-configuration を選択して、[Add security group] (セキュリティグループの追加) を選択します。

d. [保存] を選択します。

6. 以下の操作を行い、RDS データベースのセキュリティグループを RDS データベースに追加します。

a. Amazon RDS コンソール (https://console.aws.amazon.com/rds/) を開きます。

b. ナビゲーションペインで [Databases] (データベース) を選択してから、使用するデータベースを選択します。

c. [Modify] (変更) を選択します。

d. [Connectivity] (接続) にある [Security group] (セキュリティグループ) では、先程作成した rds-ec2-manual-configuration を選択し、[Continue] (続行) を選択します。

e. [Scheduling of modifications] (変更のスケジュール) で [Apply immediately] (すぐに適用) を選択します。

f. [DB インスタンスを変更] を選択します。

これで、自動接続機能を使用した場合に発生する自動のステップを模倣した手動によるステップが完了しました。

これで、このチュートリアルのオプション 3 が完了しました。オプション 1、2、3 を完了し、このチュートリアルで作成したリソースが不要になった場合は、不要なコストが発生しないように、それらを削除する必要があります。詳細については「タスク 4 (オプション): クリーンアップする」を参照してください。

タスク 4 (オプション): クリーンアップする

> **Warning**
>
> インスタンスの終了は恒久的で、元に戻すことができません。
> インスタンスを終了すると、そのインスタンスには接続できなくなり、復元することもできません。アタッチされている Amazon EBS ボリュームで、終了時に削除されるように設定されているものも、すべて恒久的に削除され、復元できません。インスタンスストアボリュームに保存されているデータは、すべて恒久的に失われます。詳細については「インスタンスの終了方法」を参照してください。
> インスタンスを終了する前に、終了後も保持する必要があるすべてのデータを永続的ストレージにバックアップしたことを確認してください。

チュートリアルを完了したので、使用しなくなったリソースをすべてクリーンアップ (削除) することをおすすめします。AWS リソースをクリーンアップすることで、アカウントに追加料金が発生するのを防ぐことができます。

このチュートリアル専用に EC2 インスタンスを起動した場合、そのインスタンスを終了することで、それに関連した料金が発生するのを防ぐことができます。

コンソールを使用してインスタンスを終了するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. このチュートリアルで作成したインスタンスを選択し、[Instance state] (インスタンスの状態)、[Terminate instance] (インスタンスの終了) の順に選択します。
4. 確認を求めるメッセージが表示されたら、[Terminate (終了)] を選択してください。

このチュートリアル専用に RDS データベースを作成した場合、削除することで、それに関連した料金が発生するのを防ぐことができます。

コンソールを使用して RDS データベースを削除するには

1. Amazon RDS コンソール (https://console.aws.amazon.com/rds/) を開きます。
2. ナビゲーションペインで、[Databases] (データベース) を選択します。
3. このチュートリアル用に作成した RDS データベースを選択し、[Actions] (アクション)、[Delete] (削除) を選択します。
4. ボックスに **`delete me`** を入力し、[Delete] (削除) をクリックします。

EC2 フリートとスポットフリート

EC2 フリートとスポットフリートは、数十、数百、数千もの Amazon EC2 インスタンスからなるフリートを 1 回の操作で起動できる便利な方法です。フリート内の各インスタンスは、起動テンプレートで設定することも、起動時に手動で設定する起動パラメータセットで設定することもできます。

トピック

- 機能とメリット
- フリートにどの方法を使用するのが最適か
- EC2 フリートまたはスポットフリートの設定オプション
- EC2 フリートの操作
- スポットフリートを操作する
- EC2 フリートまたはスポットフリートをモニタリングする
- EC2 フリートのチュートリアル
- EC2 フリートの CLI 設定の例
- スポットフリートの CLI 設定の例
- EC2 フリートとスポットフリートのクォータ

機能とメリット

フリートには以下の機能とメリットがあり、複数の EC2 インスタンスでアプリケーションを実行する際に、コストを最大限に削減すると共に、可用性とパフォーマンスを最適化できます。

複数のインスタンスタイプ

1 つのフリートで複数のインスタンスタイプを起動できるため、単一のインスタンスタイプの可用性に依存しません。そのため、フリート内のインスタンスの可用性全体が向上します。

複数のアベイラビリティーゾーンにインスタンスを分散する

フリートは複数のアベイラビリティーゾーンに起動できるため、コストを削減し、可用性を向上させることができます。フリートにスポットインスタンスが含まれている場合、料金と中断に関する設定に基づいてアベイラビリティーゾーンが自動的に選択されます。

複数の購入オプション

フリートでは、複数の購入オプション (スポットインスタンスとオンデマンドインスタンス) を起動でき、スポットインスタンスを使用することでコストを最適化できます。また、リザーブドインスタンスと Savings Plans の割引は、フリート内のオンデマンドインスタンスと併用することで活用できます。

スポットインスタンスの自動交換

フリートにスポットインスタンスが含まれている場合、スポットインスタンスが中断したら、フリートは代わりのスポットキャパシティを自動的にリクエストできます。キャパシティリバランスにより、フリートは中断されるリスクが高いスポットインスタンスをモニタリングし、プロアクティブに交換できます。

オンデマンドキャパシティを予約する

フリートは、オンデマンドキャパシティ予約を使用して、オンデマンドキャパシティを予約できます。フリートには ML 用キャパシティブロックも含めることができ、短期間の機械学習 (ML) ワークロードをサポートするために、GPU インスタンスを将来の日付で予約することができます。

フリートにどの方法を使用するのが最適か

一般的なベストプラクティスとして、Amazon EC2 Auto Scaling でスポットインスタンスとオンデマンドインスタンスのフリートを起動することをお勧めします。フリートの管理に使用できる追加機能が提供されるからです。追加機能には、スポットインスタンスとオンデマンドインスタンスの両方の自動ヘルスチェック交換、アプリケーションベースのヘルスチェック、アプリケーショントラフィックを正常なインスタンスに均等に分散するための Elastic Load Balancing との統合が含まれます。Amazon ECS、Amazon EKS (セルフ管理されたノードグループ)、Amazon VPC Lattice などの AWS サービスを使用するときも、Auto Scaling グループを使用できます。詳細については、Amazon EC2 Auto Scaling ユーザーガイドを参照してください。

Amazon EC2 Auto Scaling を使用できない場合は、EC2 フリートまたはスポットフリートの使用を検討してください。EC2 フリートとスポットフリートは、同じコア機能を備えています。ただし、EC2 フリートはコマンドラインからのみ使用でき、コンソールからは使用できません。スポットフリートは、コンソールから使用できますが、投資予定がないレガシー API を使用しています。

以下の表に従って、フリートに使用する方法を決定してください。

フリートに使用する方法	どのようなときに使うか?	ユースケース
Amazon EC2 Auto Scaling	• 単一の構成または混合の構成を持つ、複数のインスタンスが必要です。 • インスタンスのライフサイクル管理を自動化したい。	必要な数のインスタンスを維持しながら、インスタンスのライフサイクルを管理する Auto Scaling グループを作成します。指定した最小値と最大限度の間の水平スケーリング (インスタンスの追加) をサポートします。
EC2 Fleet	• 単一の構成または混合の構成を持つ、複数のインスタンスが必要です。 • インスタンスのライフサイクルを自己管理するのがよいでしょう。 • 自動スケーリングが必要ない場合は、`instant` タイプの EC2 フリートの使用をお勧めします。	インスタンスタイプ別、AMI 別、アベイラビリティーゾーン別、またはサブネット別で異なる複数の起動仕様を指定し、オンデマンドインスタンスとスポットインスタンスの両方の `instant` フリートを 1 回の操作で作成します。スポットインスタンス配分戦略は、デフォルトではユニット単位で `lowest-price` に設定されていますが、`price-capacity-optimized` に変更することをお勧めします。
スポットフリート	• スポットフリートは投資予定がないレガシー API であるため、使用はお勧めしません。 • インスタンスのライフサイクルを管理する場合は、EC2 フリートを使用してください。	スポットフリートは、通常なら EC2 フリートを使用するユースケースで、コンソールのサポートが必要な場合にのみ使用します。

フリートに使用する方法	どのようなときに使うか?	ユースケース
	• インスタンスのライフサイクルを管理しない場合は、Auto Scaling グループを使用してください。	

EC2 フリートまたはスポットフリートの設定オプション

EC2 フリートまたはスポットフリートを計画する場合は、フリートの設定方法を決定するときに、以下のオプションを検討することをお勧めします。

設定オプション	質問	ドキュメント
フリートリクエストタイプ	目的のターゲットキャパシティのワンタイムリクエストを送信するフリートが必要ですか、あるいはこの先も継続してターゲットキャパシティを維持するフリートが必要ですか。	EC2 フリートとスポットフリートのリクエストタイプ
スポットインスタンス	フリートにスポットインスタンスを含めるつもりですか。フリートを計画するときにスポットのベストプラクティスを確認のうえ使用して、できる限り安い料金でインスタンスをプロビジョニングできるようにします。	Amazon EC2 スポットのベストプラクティス
フリートの支出制限	フリートに支払う 1 時間あたりの金額を制限しますか。	EC2 フリートまたはスポットフリートの支出制限を設定する
インスタンスタイプおよび属性ベースのインスタンスタイプの選択	フリートのインスタンスタイプを自分で指定しますか、それともアプリケーション要件を満たすインスタンスタイプを Amazon EC2 が選択できるようにしますか。	EC2 フリートまたはスポットフリートのインスタンスタイプを選択するための属性を指定する

設定オプション	質問	ドキュメント
インスタンスの重み付け	インスタンスタイプごとにコンピューティング能力とパフォーマンスを表す重みを割り当てて、Amazon EC2 が使用可能なインスタンスタイプを任意に組み合わせて目的のターゲットキャパシティを満たすタイプを選択できるようにしますか。	インスタンスの重み付けを使用して、EC2 フリートまたはスポットフリートのコストとパフォーマンスを管理する
配分戦略	フリート内のスポットインスタンスとオンデマンドインスタンスに使用可能なキャパシティ、料金、またはインスタンスタイプを最適化するかどうかを自分で決定しますか。	配分戦略を使用して、EC2 フリートまたはスポットフリートがどのようにスポットキャパシティとオンデマンドキャパシティを満たすかを決定します。
キャパシティーの再調整	フリートがリスクのあるスポットインスタンスを自動的に置き換えるようにしますか。	EC2 フリートとスポットフリートでキャパシティの再調整を使用して、リスクのあるスポットインスタンスを置き換える
オンデマンドキャパシティ予約	フリート内のオンデマンドインスタンス用にキャパシティを予約しますか。	キャパシティ予約を使用して、EC2 フリートのオンデマンドキャパシティを予約する

EC2 フリートとスポットフリートのリクエストタイプ

> **Note**
>
> スポットフリートは、計画された投資なしでレガシー API を使用します。代わりに EC2 フリートまたは Auto Scaling グループを使用することをお勧めします。詳細については、「フリートにどの方法を使用するのが最適か」を参照してください。

EC2 フリートまたはスポットフリートのリクエストタイプによって、リクエストが同期的か非同期的かが決まり、さらに目的のターゲットキャパシティを 1 回限りでリクエストするのか、今後とも継続的にターゲットキャパシティを維持していくのかが決まります。フリートを設定するときに、リクエストタイプを指定する必要があります。

EC2 フリートとスポットフリートのどちらにも、`request` と `maintain` の 2 つのリクエストタイプがあります。また、EC2 フリートには `instant` という 3 つ目のリクエストタイプがあります。

フリートのリクエストタイプ

`instant` (EC2 フリートのみ)

リクエストタイプを `instant` として設定すると、EC2 フリートは必要な容量に対して同期ワンタイムリクエストを送信します。API レスポンスで、起動したインスタンスが返され、さらに起動できなかったインスタンスに関するエラーも返されます。詳細については、「タイプが instant の EC2 フリートを設定する」を参照してください。

`request`

リクエストタイプを `request` として設定すると、フリートは必要なキャパシティに対して 1 回限りのリクエストを非同期に送信します。スポットの中断のためにキャパシティが減少しても、スポットインスタンスの補充を試みません。また、キャパシティが利用不可になっても、代替スポットキャパシティプールにリクエストを送信しません。コンソールを使用してタイプ `request` のスポットフリートを作成するとき、[ターゲットキャパシティを維持] チェックボックスをオフにします。

`maintain` (デフォルト)

(デフォルト) リクエストタイプを `maintain` に設定した場合、フリートは目的のキャパシティに非同期リクエストを送り、中断したスポットインスタンスを自動的に補充して、キャパシティを維持します。コンソールを使用してタイプ `maintain` のスポットフリートを作成するとき、[ターゲットキャパシティを維持] チェックボックスをオンにします。

タイプが instant の EC2 フリートを設定する

EC2 フリート インスタント タイプは、希望する容量を起動するために 1 回だけ試行する、同期ワンタイムリクエストです。API レスポンスは、起動したインスタンスとともに、起動できなかったインスタンスのエラーを一覧表で表示します。このガイドで述べている、EC2 フリートの インスタント タイプを使用することにはいくつかの利点があります。構成例については、ガイドの最後に記載しています。

EC2 インスタンスを起動するために起動専用 API が必要なワークロードの場合は、RunInstances API を使用できます。ただし、RunInstances では、オンデマンドインスタンスまたはスポットインスタンスのみを起動できますが、同じリクエストで両方を起動することはできません。さらに、RunInstances を使用してスポットインスタンスを起動する場合、スポットインスタンスリクエストは 1 つのインスタンスタイプと 1 つのアベイラビリティーゾーンに制限されます。これは、単一のスポットキャパシティープール (同じインスタンスタイプとアベイラビリティーゾーンを有する、未使用のインスタンスセット) をターゲットにしています。スポットキャパシティープールに、リクエストに対して十分なスポットインスタンス容量がない場合、RunInstances 呼び出しは失敗します。

RunInstances を使用してスポットインスタンスを起動する代わりに、CreateFleet API を `instant` に設定した `type` パラメータと使用すると、以下の利点があります。

- オンデマンドインスタンスとスポットインスタンスを 1 回のリクエストで起動します。EC2 フリートは、オンデマンドインスタンス、スポットインスタンス、またはその両方を起動できます。スポットインスタンスへのリクエストは、利用可能な容量があり、リクエストで指定した 1 時間あたりの上限料金がスポット料金を超えている場合に達成されます。
- スポットインスタンスの可用性を向上させます。EC2 フリートタイプ `instant` を使用してスポットインスタンスを起動でき、以下のような スポットベストプラクティス という利点があります:
  - ベストプラクティス: インスタンスタイプとアベイラビリティーゾーンについて柔軟に対応する。

    利点:複数のインスタンスタイプとアベイラビリティーゾーンを指定すると、スポットキャパシティープールの数が増加します。これにより、スポットサービスは、希望するスポットコンピューティング容量を見つけて割り当てる可能性が高くなります。経験則としては、ワークロードごとに少なくとも 10 種類のインスタンスタイプで柔軟に対応し、すべてのアベイラビリティーゾーンが VPC で使用するように設定されていることを確認します。
  - スポットベストプラクティス: price-capacity-optimized配分戦略を使用する。

    利点: `price-capacity-optimized`配分戦略により、最も可用性の高いスポットキャパシティープールのインスタンスが特定され、そしてこのプールで最も低価格のもののインスタンスを自動的にプロビジョニングされます。最適な容量を持つプールからスポットインスタンス容量が供給されるため、Amazon EC2 が容量を元に戻す必要があるときにスポットインスタンスが中断されます。
- 幅広い機能にアクセスする。起動専用 API が必要なワークロードで、EC2 フリートにインスタンスのライフサイクルを管理させるのではなく、インスタンスのライフサイクルを管理したい場合は、RunInstances API の代わりに EC2 フリートタイプ `instant` を使用します。EC2 フリート

は、次の例で示すように、RunInstances よりも幅広い機能を提供します。その他のすべてのワークロードについては、Amazon EC2 Auto Scaling を使用する必要があります。これは、ELB ベースのアプリケーション、コンテナ化されたワークロード、キュー処理ジョブなど、さまざまなワークロードに対してより包括的な機能セットを提供するからです。

EC2 フリートインスタントタイプを使用して、キャパシティブロックにインスタンスを起動できます。詳細については、「チュートリアル: キャパシティブロックにインスタンスを起動するように EC2 フリートを設定する」を参照してください。

Amazon EC2 Auto Scaling や Amazon EMR などの AWS サービスでは、EC2 フリートの インスタント タイプを使用し、EC2 インスタンスを起動します。

EC2 フリートインスタントタイプ の前提条件

EC2 フリートを作成するための前提条件については、「EC2 フリートの前提条件」を参照してください。

瞬時に実行される EC2 フリート機能

EC2 フリートタイプ `instant` で作業する場合、イベントのシーケンスは以下のようになります。

1. 設定 `instant` のような CreateFleet リクエストタイプを設定してください。詳細については、「EC2 フリートの作成」を参照してください。API コールを行った後は、それを変更することはできません。
2. リクエスト: API コールを行うと、Amazon EC2 は希望する容量に同期ワンタイムリクエストを配置します。
3. レスポンス: API レスポンスには、起動したインスタンスとともに、起動できなかったインスタンスのエラーがリストされます。
4. 説明: EC2 フリートの説明、EC2 フリートに関連付けられたインスタンスの一覧表示、EC2 フリートの履歴の表示を行うことができます。
5. インスタンスの終了: インスタンスはいつでも終了できます。
6. フリートリクエストの削除: フリートリクエストは、手動または自動で削除できます。
   - 手動: インスタンスの起動後にフリートリクエストを削除できます。

     実行中のインスタンスを持つ削除した `instant` フリートはサポートされていないことに注意してください。`instant` フリートを削除すると、Amazon EC2 はすべてのインスタンスを自動的に終了させます。1000 個を超えるインスタンスで構成された フリートの場合、削除リクエスト

は失敗する可能性があります。1000 個を超えるインスタンスで構成されたフリートの場合は、まずほとんどのインスタンスを手動で終了させ、残りのインスタンスが 1000 個以下になるようにします。次に、フリートを削除すると、残りのインスタンスは自動的に終了します。

- 自動: Amazon EC2 は、次のいずれかの後にフリートリクエストを削除します。
  - すべてのインスタンスが終了します。
  - フリートはインスタンスの起動に失敗します。

例

以下の例では、EC2 フリートタイプ `instant` の使用方法を示します。さまざまなユースケースで使用できます。EC2 CreateFleet API パラメータの使用方法の詳細については、Amazon EC2 API リファレンス 内の CreateFleet を参照してください。

例

- 例 1: 容量最適化配分戦略を使用してスポットインスタンスを起動する
- 例 2: 容量最適化割り当て戦略を使用して 1 つのスポットインスタンスを起動する
- 例 3: インスタンスの重み付けを使用して、スポットインスタンスを起動する
- 例 4: 1 つのアベイラビリティーゾーン内でスポットインスタンスを起動する
- 例 5: 単一アベイラビリティーゾーン内で単一インスタンスタイプのスポットインスタンスを起動する
- 例 6: 最小ターゲット容量を起動できる場合にのみスポットインスタンスを起動する
- 例 7: 単一のアベイラビリティーゾーンで同じインスタンスタイプで最小ターゲット容量を起動できる場合にのみスポットインスタンスを起動する
- 例 8: 複数の起動テンプレートを使用したインスタンスの起動
- 例 9: オンデマンドインスタンスのベースを使用してスポットインスタンスを起動する
- 例 10: キャパシティーの予約および優先順位配分戦略を使用したオンデマンドインスタンスをベースにして、キャパシティー最適化配分戦略を使用しスポットインスタンスを起動する
- 例 11: 容量最適化優先順位配分戦略を使用してスポットインスタンスを起動する
- 例 12: AMI ID ではなく Systems Manager パラメータを指定する

例 1: 容量最適化配分戦略を使用してスポットインスタンスを起動する

次の例では、EC2 フリートのタイプで必要なパラメータ(起動テンプレート、ターゲット容量、デフォルト購入オプション、および起動テンプレートオーバーライド) を指定します。`instant`

- 起動テンプレートは、起動テンプレート名とバージョン番号によって識別されます。
- 12 の起動テンプレートオーバーライドでは、4 つの異なるインスタンスタイプと 3 つの異なるサブネットが指定され、それぞれ別のアベイラビリティーゾーンに配置されます。各インスタンスタイプとサブネットの組み合わせによってスポットキャパシティープールが定義され、12 個のスポットキャパシティープールが作成されます。
- フリートのターゲット容量は 20 インスタンスです。
- デフォルト購入オプションの spot では、フリートは、起動中のインスタンス数の最適な容量のスポットキャパシティプールに 20 個のスポットインスタンスを起動しようとします。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized"
    },
    "LaunchTemplateConfigs": [
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt1",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-e7188bab"
            },
```

```
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5d.large",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"m5d.large",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"m5d.large",
               "SubnetId":"subnet-49e41922"
            }
         ]
      }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 20,
        "DefaultTargetCapacityType": "spot"
    },
    "Type": "instant"
}
```

例 2: 容量最適化割り当て戦略を使用して 1 つのスポットインスタンスを起動する

複数の EC2 フリート API コールタイプ `instant` を行い、TotalTargetCapacity を 1 に設定することで、一度に 1 つのスポットインスタンスを最適に起動できます。

次の例では、EC2 フリートインスタントタイプで必要なパラメータ (起動テンプレート、ターゲット容量、デフォルト購入オプション、および起動テンプレートオーバーライド) を指定します。起動テンプレートは、起動テンプレート名とバージョン番号によって識別されます。12 の起動テンプレートオーバーライドには、4 つの異なるインスタンスタイプと 3 つの異なるサブネットがあり、それぞれ別のアベイラビリティーゾーンにあります。フリートのターゲット容量は 1 インスタンスで、デフォルトの購入オプションはスポットです。これにより、フリートは、容量最適化の割り当て戦略に基づいて 12 のスポットキャパシティープールのいずれかからスポットインスタンスを起動し、最も利用可能な容量プールからスポットインスタンスを起動しようとします。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized"
    },
    "LaunchTemplateConfigs": [
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt1",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5d.large",
```

```
                "SubnetId":"subnet-49e41922"
            },
            {
                "InstanceType":"m5.large",
                "SubnetId":"subnet-fae8c380"
            },
            {
                "InstanceType":"m5.large",
                "SubnetId":"subnet-e7188bab"
            },
            {
                "InstanceType":"m5.large",
                "SubnetId":"subnet-49e41922"
            },
            {
                "InstanceType":"m5d.large",
                "SubnetId":"subnet-fae8c380"
            },
            {
                "InstanceType":"m5d.large",
                "SubnetId":"subnet-e7188bab"
            },
            {
                "InstanceType":"m5d.large",
                "SubnetId":"subnet-49e41922"
            }
        ]
      }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 1,
        "DefaultTargetCapacityType": "spot"
    },
    "Type": "instant"
}
```

例 3: インスタンスの重み付けを使用して、スポットインスタンスを起動する

次の例では、インスタンス分量指定を使っています。これは、料金が 1 インスタンス時間当たりではなく、1 ユニット時間当たりであることを意味します。各起動設定では、ワークロードのユニットに 15 GB のメモリと 4 vCPU が必要であると仮定して、インスタンスで実行できるワークロードのユニット数に基づいて、異なるインスタンスタイプと異なる重みが表示されます。たとえば、m5.xlarge (4 vCPU と 16 GB のメモリ) は 1 つのユニットを実行でき、重み付けは

1、m5.2xlarge (8 vCPU と 32 GB のメモリ) は 2 ユニットを実行でき、重み付けは 2 というように続きます。総目標容量は 40 ユニットに設定されています。デフォルトの購入オプションはスポットで、割り当て戦略は容量最適化です。これにより、40 m5.xlarge (40 を 1 で割ったもの) 、20 m5.2xlarge (40 を 2 で割ったもの) 、10 m5.4xlarge (40 を 4 で割ったもの) 、5 m5.8xlarge (40 を 8 で割ったもの) 、またはインスタンスタイプの組み合わせのいずれかになります。容量を最適化した割り当て戦略に基づきます。

詳細については、「インスタンスの重み付けを使用して、EC2 フリートまたはスポットフリートのコストとパフォーマンスを管理する」を参照してください。

```
{
   "SpotOptions":{
      "AllocationStrategy":"capacity-optimized"
   },
   "LaunchTemplateConfigs":[
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt1",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"m5.xlarge",
               "SubnetId":"subnet-fae8c380",
               "WeightedCapacity":1
            },
            {
               "InstanceType":"m5.xlarge",
               "SubnetId":"subnet-e7188bab",
               "WeightedCapacity":1
            },
            {
               "InstanceType":"m5.xlarge",
               "SubnetId":"subnet-49e41922",
               "WeightedCapacity":1
            },
            {
               "InstanceType":"m5.2xlarge",
               "SubnetId":"subnet-fae8c380",
               "WeightedCapacity":2
            },
            {
               "InstanceType":"m5.2xlarge",
```

```
            "SubnetId":"subnet-e7188bab",
            "WeightedCapacity":2
         },
         {
            "InstanceType":"m5.2xlarge",
            "SubnetId":"subnet-49e41922",
            "WeightedCapacity":2
         },
         {
            "InstanceType":"m5.4xlarge",
            "SubnetId":"subnet-fae8c380",
            "WeightedCapacity":4
         },
         {
            "InstanceType":"m5.4xlarge",
            "SubnetId":"subnet-e7188bab",
            "WeightedCapacity":4
         },
         {
            "InstanceType":"m5.4xlarge",
            "SubnetId":"subnet-49e41922",
            "WeightedCapacity":4
         },
         {
            "InstanceType":"m5.8xlarge",
            "SubnetId":"subnet-fae8c380",
            "WeightedCapacity":8
         },
         {
            "InstanceType":"m5.8xlarge",
            "SubnetId":"subnet-e7188bab",
            "WeightedCapacity":8
         },
         {
            "InstanceType":"m5.8xlarge",
            "SubnetId":"subnet-49e41922",
            "WeightedCapacity":8
         }
      ]
   }
],
"TargetCapacitySpecification":{
   "TotalTargetCapacity":40,
   "DefaultTargetCapacityType":"spot"
```

```
   },
   "Type":"instant"
}
```

例 4: 1 つのアベイラビリティーゾーン内でスポットインスタンスを起動する

スポットオプション singleAvailabilityZone を true に設定することで、1 つのアベイラビリティーゾーンですべてのインスタンスを起動するようにフリートを設定できます。

12 の起動テンプレートオーバーライドでは、インスタンスタイプとサブネット (それぞれ別々のアベイラビリティーゾーン内) が異なりますが、重み容量は同じです。合計ターゲット容量は 20 インスタンスで、デフォルトの購入オプションは スポットで、スポット配分戦略は容量最適化です。EC2 フリートは、起動仕様を使用して最適な容量を持つスポットキャパシティープールから、1 つの AZ で 20 個のスポットインスタンスをすべて起動します。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized",
        "SingleAvailabilityZone": true
    },
    "LaunchTemplateConfigs": [
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt1",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
```

```
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5d.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"m5d.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"m5d.4xlarge",
               "SubnetId":"subnet-49e41922"
            }
         ]
      }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 20,
        "DefaultTargetCapacityType": "spot"
    },
    "Type": "instant"
}
```

例 5: 単一アベイラビリティーゾーン内で単一インスタンスタイプのスポットインスタンスを起動する

SpotOptions singleInstanceType を true、SingleAvailabilityZone を true に設定することで、同じインスタンスタイプのすべてのインスタンスを単一のアベイラビリティーゾーンで起動するようにフリートを設定できます。

12 の起動テンプレートオーバーライドでは、インスタンスタイプとサブネット (それぞれ別々のアベイラビリティーゾーン内) が異なりますが、重み容量は同じです。合計ターゲット容量は 20 インスタンスで、デフォルトの購入オプションは spot で、スポット配分戦略は容量最適化です。EC2 フリートは、起動仕様を使用して最適な容量でスポットインスタンスプールから、同じインスタンスタイプの 20 個のスポットインスタンスをすべて単一の AZ で起動します。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized",
        "SingleInstanceType": true,
        "SingleAvailabilityZone": true
    },
    "LaunchTemplateConfigs": [
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt1",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
```

```
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5d.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"m5d.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"m5d.4xlarge",
               "SubnetId":"subnet-49e41922"
            }
         ]
      }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 20,
        "DefaultTargetCapacityType": "spot"
    },
    "Type": "instant"
}
```

例 6: 最小ターゲット容量を起動できる場合にのみスポットインスタンスを起動する

スポットオプション MinTargetCapacity を一緒に起動する最小ターゲットキャパシティに設定することで、最小ターゲットキャパシティを起動できる場合にのみインスタンスを起動するようにフリートを設定できます。

MinTargetCapacity を指定する場合は、SingleInstanceType または SingleAvailabilityZone のパラメータを少なくとも 1 つ指定する必要があります。この例では、SingleInstanceType が指定されているため、20 個のインスタンスすべてが同じインスタンスタイプを使用する必要があります。

12 の起動テンプレートオーバーライドでは、インスタンスタイプとサブネット (それぞれ別々のアベイラビリティーゾーン内) が異なりますが、重み容量は同じです。合計ターゲット容量と最小ターゲット容量は両方とも 20 インスタンスに設定され、デフォルトの購入オプションはスポット、スポット割り当て戦略は容量最適化です。EC2 フリートは、起動テンプレートのオーバーライドを使用して、最適な容量でスポットキャパシティープールから 20 個のスポットインスタンスを起動します。これは、20 個のインスタンスをすべて同時に起動できる場合のみです。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized",
        "SingleInstanceType": true,
        "MinTargetCapacity": 20
    },
    "LaunchTemplateConfigs": [
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt1",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-49e41922"
            },
```

```
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5d.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"m5d.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"m5d.4xlarge",
               "SubnetId":"subnet-49e41922"
            }
         ]
      }
   ],
   "TargetCapacitySpecification": {
       "TotalTargetCapacity": 20,
       "DefaultTargetCapacityType": "spot"
   },
   "Type": "instant"
```

```
}
```

例 7: 単一のアベイラビリティーゾーンで同じインスタンスタイプで最小ターゲット容量を起動できる場合にのみスポットインスタンスを起動する

スポットオプション MinTargetCapacity を SingleInstanceType および SingleAvailabilityZone オプションとともに起動する最小ターゲットキャパシティに設定することで、単一のアベイラビリティーゾーン内の単一のインスタンスタイプで最小ターゲットキャパシティを起動できる場合にのみ、インスタンスを起動するようにフリートを設定できます。

起動テンプレートをオーバーライドする 12 の起動条件は、インスタンスタイプとサブネットが異なりますが(それぞれ異なるアベイラビリティーゾーン内で)、加重容量は同じです。合計ターゲット容量と最小ターゲット容量は両方とも 20 インスタンスに設定され、デフォルトの購入オプションはスポット、スポット割り当て戦略は容量最適化、SingleInstanceType は true、SingleAvailabilityZone は true です。EC2 フリートは、同じインスタンスタイプの 20 個のスポットインスタンスをすべて 1 つの AZ で起動し、起動条件を使用して最適な容量を持つスポットキャパシティープールから起動します。これは、20 個のインスタンスをすべて同時に起動できる場合に限ります。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized",
        "SingleInstanceType": true,
        "SingleAvailabilityZone": true,
        "MinTargetCapacity": 20
    },
    "LaunchTemplateConfigs": [
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt1",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5.4xlarge",
```

```
            "SubnetId":"subnet-49e41922"
         },
         {
            "InstanceType":"c5d.4xlarge",
            "SubnetId":"subnet-fae8c380"
         },
         {
            "InstanceType":"c5d.4xlarge",
            "SubnetId":"subnet-e7188bab"
         },
         {
            "InstanceType":"c5d.4xlarge",
            "SubnetId":"subnet-49e41922"
         },
         {
            "InstanceType":"m5.4xlarge",
            "SubnetId":"subnet-fae8c380"
         },
         {
            "InstanceType":"m5.4xlarge",
            "SubnetId":"subnet-e7188bab"
         },
         {
            "InstanceType":"m5.4xlarge",
            "SubnetId":"subnet-49e41922"
         },
         {
            "InstanceType":"m5d.4xlarge",
            "SubnetId":"subnet-fae8c380"
         },
         {
            "InstanceType":"m5d.4xlarge",
            "SubnetId":"subnet-e7188bab"
         },
         {
            "InstanceType":"m5d.4xlarge",
            "SubnetId":"subnet-49e41922"
         }
      ]
   }
],
"TargetCapacitySpecification": {
    "TotalTargetCapacity": 20,
    "DefaultTargetCapacityType": "spot"
```

```
    },
    "Type": "instant"
}
```

例 8: 複数の起動テンプレートを使用したインスタンスの起動

複数の起動テンプレートを指定することで、異なるインスタンスタイプまたはインスタンスタイプのグループに対して異なる起動条件でインスタンスを起動するようにフリートを設定できます。この例では、インスタンスタイプごとに異なる EBS ボリュームサイズが必要で、起動テンプレート ec2-fleet-lt-4xl、ec2-fleet-lt-9xl、ec2-fleet-lt-18xl で設定されています。

この例では、サイズに基づいて 3 種類のインスタンスタイプに対して 3 種類の起動テンプレートを使用します。すべての起動テンプレートの起動条件オーバーライドでは、インスタンスタイプの vCPUs に基づくインスタンスの重みが使用されます。合計ターゲット容量は 144 ユニットで、デフォルトの購入オプションは spot で、スポット配分戦略は容量最適化です。EC2 フリートは、起動テンプレート ec2-fleet-4xl を使用して 9 c5n.4xlarge (144 を 16 で割った) を起動するか、起動テンプレート ec2-fleet-9xl を使用して 4 c5n.9xlarge (144 を 36 で割った) を起動するか、 または起動テンプレート ec2-fleet-18xl を使用して 2 c5n.18xlarge (144 を 72 で割った) を起動するか、もしくはインスタンスタイプ容量最適化の割り当て戦略に基づいて重みを追加したインスタンスタイプの混合を使って起動することができます。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized"
    },
    "LaunchTemplateConfigs": [
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt-18xl",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5n.18xlarge",
               "SubnetId":"subnet-fae8c380",
               "WeightedCapacity":72
            },
            {
               "InstanceType":"c5n.18xlarge",
               "SubnetId":"subnet-e7188bab",
               "WeightedCapacity":72
            },
```

```
            {
               "InstanceType":"c5n.18xlarge",
               "SubnetId":"subnet-49e41922",
               "WeightedCapacity":72
            }
         ]
      },
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt-9xl",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5n.9xlarge",
               "SubnetId":"subnet-fae8c380",
               "WeightedCapacity":36
            },
            {
               "InstanceType":"c5n.9xlarge",
               "SubnetId":"subnet-e7188bab",
               "WeightedCapacity":36
            },
            {
               "InstanceType":"c5n.9xlarge",
               "SubnetId":"subnet-49e41922",
               "WeightedCapacity":36
            }
         ]
      },
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt-4xl",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5n.4xlarge",
               "SubnetId":"subnet-fae8c380",
               "WeightedCapacity":16
            },
            {
               "InstanceType":"c5n.4xlarge",
               "SubnetId":"subnet-e7188bab",
```

```
               "WeightedCapacity":16
            },
            {
               "InstanceType":"c5n.4xlarge",
               "SubnetId":"subnet-49e41922",
               "WeightedCapacity":16
            }
         ]
      }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 144,
        "DefaultTargetCapacityType": "spot"
    },
    "Type": "instant"
}
```

例 9: オンデマンドインスタンスのベースを使用してスポットインスタンスを起動する

次の例では、フリートの合計ターゲット容量を 20 インスタンス、ターゲット容量を 5 オンデマンドインスタンス として指定します。デフォルト購入オプションはスポットです。フリートは指定されたとおり 5 オンデマンドインスタンス を起動しますが、合計ターゲット容量を満たすために、さらに 15 以上のインスタンスを起動する必要があります。差額の購入オプションは、TotalTargetCapacity - OnDemandTargetCapacity = DefaultTargetCapacityType で計算されます。この結果、15 のスポットインスタンスを起動するフリートは容量最適化配分戦略に基づいて 12 スポットキャパシティープールのうちの 1 つを形成します。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized"
    },
    "LaunchTemplateConfigs": [
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt1",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-fae8c380"
            },
```

```
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5d.large",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"m5d.large",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"m5d.large",
               "SubnetId":"subnet-49e41922"
            }
```

```
          ]
       }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 20,
        "OnDemandTargetCapacity": 5,
        "DefaultTargetCapacityType": "spot"
    },
    "Type": "instant"
}
```

例 10: キャパシティーの予約および優先順位配分戦略を使用したオンデマンドインスタンスをベースにして、キャパシティー最適化配分戦略を使用しスポットインスタンスを起動する

キャパシティーの予約の使用戦略を use-capacity-reservations-first に設定することで、デフォルトのターゲット容量タイプをスポットにしたオンデマンドインスタンス の起動時に、最初にオンデマンドキャパシティー予約 を使用するようにフリートを設定できます。また、複数のインスタンスプールに未使用のキャパシティーの予約がある場合、選択したオンデマンド配分戦略が適用されます。この例では、オンデマンド配分戦略は優先されています。

この例では、利用可能な未使用の予約予約 が 6 個あります。これは、フリートの目標オンデマンド容量である 10 オンデマンドインスタンス を下回っています。

アカウントには、2 つのプールに 6 個の未使用キャパシティーの予約 があります。各プールのキャパシティーの予約の数は AvailableInstanceCount で示されます。

```
{
    "CapacityReservationId": "cr-111",
    "InstanceType": "m5.large",
    "InstancePlatform": "Linux/UNIX",
    "AvailabilityZone": "us-east-1a",
    "AvailableInstanceCount": 3,
    "InstanceMatchCriteria": "open",
    "State": "active"
}

{
    "CapacityReservationId": "cr-222",
    "InstanceType": "c5.large",
    "InstancePlatform": "Linux/UNIX",
    "AvailabilityZone": "us-east-1a",
    "AvailableInstanceCount": 3,
```

```
    "InstanceMatchCriteria": "open",
    "State": "active"
}
```

以下のフリート設定は、この例に関連する設定のみを示しています。オンデマンド配分戦略は優先的であり、キャパシティーの予約の使用戦略予約は use-capacity-reservations-first です。スポット配分戦略を容量最適化する 合計ターゲット容量は 20 で、オンデマンドターゲット容量は 10 で、デフォルトのターゲット容量タイプはスポットです。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized"
    },
    "OnDemandOptions":{
       "CapacityReservationOptions": {
         "UsageStrategy": "use-capacity-reservations-first"
       },
       "AllocationStrategy":"prioritized"
    },
    "LaunchTemplateConfigs": [
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt1",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-fae8c380",
               "Priority": 1.0
            },
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-e7188bab",
               "Priority": 2.0
            },
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-49e41922",
               "Priority": 3.0
            },
            {
               "InstanceType":"c5d.large",
```

```
            "SubnetId":"subnet-fae8c380",
            "Priority": 4.0
         },
         {
            "InstanceType":"c5d.large",
            "SubnetId":"subnet-e7188bab",
            "Priority": 5.0
         },
         {
            "InstanceType":"c5d.large",
            "SubnetId":"subnet-49e41922",
            "Priority": 6.0
         },
         {
            "InstanceType":"m5.large",
            "SubnetId":"subnet-fae8c380",
            "Priority": 7.0
         },
         {
            "InstanceType":"m5.large",
            "SubnetId":"subnet-e7188bab",
            "Priority": 8.0
         },
         {
            "InstanceType":"m5.large",
            "SubnetId":"subnet-49e41922",
            "Priority": 9.0
         },
         {
            "InstanceType":"m5d.large",
            "SubnetId":"subnet-fae8c380",
            "Priority": 10.0
         },
         {
            "InstanceType":"m5d.large",
            "SubnetId":"subnet-e7188bab",
            "Priority": 11.0
         },
         {
            "InstanceType":"m5d.large",
            "SubnetId":"subnet-49e41922",
            "Priority": 12.0
         }
      ]
```

```
      }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 20,
        "OnDemandTargetCapacity": 10,
        "DefaultTargetCapacityType": "spot"
    },
    "Type": "instant"
}
```

上記の設定を使用してインスタントフリートを作成すると、目標容量を満たすために以下の 20 個のインスタンスが起動されます。

- 7 c5.large オンデマンドインスタンス (us-east-1a) – c5.large (us-east-1a) が最初に優先され、利用可能な未使用 c5.large キャパシティーの予約が 3 つあります。キャパシティーの予約は、3 つの オンデマンドインスタンス を起動するために最初に使用され、さらにオンデマンド配分戦略に従って、この例で優先されている追加の 4 つのオンデマンドインスタンス が起動されます。
- 3 m5.large オンデマンドインスタンス (us-east-1a) – m5.large (us-east-1a) が 2 番目に優先され、利用可能な未使用の m5.large キャパシティ予約が 3 つあります。
- 容量最適化割り当て戦略に従って最適な容量を持つ 12 個のスポットキャパシティープールのうちの 1 つからの 10 個のスポットインスタンス。

フリートの起動後、describe-capacity-reservations を実行して、未使用のキャパシティ予約の数を確認できます。この例では、以下のレスポンスが表示されます。これは、c5.large および m5.large のすべてのキャパシティーの予約が使用されていることを示しています。

```
{
    "CapacityReservationId": "cr-111",
    "InstanceType": "m5.large",
    "AvailableInstanceCount": 0
}

{
    "CapacityReservationId": "cr-222",
    "InstanceType": "c5.large",
    "AvailableInstanceCount": 0
}
```

例 11: 容量最適化優先順位配分戦略を使用してスポットインスタンスを起動する

次の例では、EC2 フリートインスタントタイプで必要なパラメータ (起動テンプレート、ターゲット容量、デフォルト購入オプション、および起動テンプレートオーバーライド) を指定します。起動テンプレートは、起動テンプレート名とバージョン番号によって識別されます。起動テンプレートを上書きする 12 の起動仕様には、優先順位が割り当てられた 4 つの異なるインスタンスタイプと、それぞれ別のアベイラビリティーゾーンに 3 つの異なるサブネットがあります。フリートのターゲット容量は 20 インスタンスで、デフォルトの購入オプションはスポットです。このため、フリートは、容量最適化優先順位付き割り当て戦略に基づいて 12 のスポットキャパシティープールのいずれかから 20 のスポットインスタンスを起動しようとします。これは、ベスト工フォート方式で優先順位を実装します。ですが、最初に容量を最適化します。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized-prioritized"
    },
    "LaunchTemplateConfigs": [
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt1",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-fae8c380",
               "Priority": 1.0
            },
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-e7188bab",
               "Priority": 1.0
            },
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-49e41922",
               "Priority": 1.0
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-fae8c380",
               "Priority": 2.0
```

```
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-e7188bab",
               "Priority": 2.0
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-49e41922",
               "Priority": 2.0
            },
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-fae8c380",
               "Priority": 3.0
            },
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-e7188bab",
               "Priority": 3.0
            },
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-49e41922",
               "Priority": 3.0
            },
            {
               "InstanceType":"m5d.large",
               "SubnetId":"subnet-fae8c380",
               "Priority": 4.0
            },
            {
               "InstanceType":"m5d.large",
               "SubnetId":"subnet-e7188bab",
               "Priority": 4.0
            },
            {
               "InstanceType":"m5d.large",
               "SubnetId":"subnet-49e41922",
               "Priority": 4.0
            }
         ]
      }
   ],
```

```
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 20,
        "DefaultTargetCapacityType": "spot"
    },
    "Type": "instant"
}
```

例 12: AMI ID ではなく Systems Manager パラメータを指定する

以下の例では、起動テンプレートを使用して、フリート内のインスタンスの設定を指定しています。この例では、AMI ID を指定する代わりに `ImageId` を使用しており、AMI は Systems Manager パラメータで参照されます。Systems Manager パラメータは、インスタンスの起動時に AMI ID に解決されます。

この例では、Systems Manager パラメータを `resolve:ssm:golden-ami` という有効な形式で指定しています。Systems Manager パラメータには、この他にも有効な形式があります。詳細については、「AMI ID のかわりに Systems Manager パラメータを使用する」を参照してください。

> **Note**
>
> フリートタイプはタイプ`instant`である必要があります。他のフリートタイプでは、AMI ID の代わりに Systems Manager パラメータを指定することはできません。

```
{
    "LaunchTemplateData": {
        "ImageId": "resolve:ssm:golden-ami",
        "InstanceType": "m5.4xlarge",
        "TagSpecifications": [{
            "ResourceType": "instance",
            "Tags": [{
                "Key": "Name",
                "Value": "webserver"
            }]
        }]
    }
}
```

EC2 フリートまたはスポットフリートの支出制限を設定する

EC2 フリートまたはスポットフリートに対する 1 時間あたりの支出金額に制限を設定できます。支出制限に達すると、フリートはターゲットキャパシティに達していない場合でもインスタンスの起動を停止します。

オンデマンドインスタンスとスポットインスタンスに個別に支出制限を設定できます。

EC2 フリート内のオンデマンドインスタンスとスポットインスタンスに支出制限を設定するには

create-fleet コマンドと以下のパラメータを使用します。

- オンデマンドインスタンスの場合: `OnDemandOptions` 構造で、`MaxTotalPrice` フィールドに支出制限を指定します。
- スポットインスタンスの場合: `SpotOptions` 構造で、`MaxTotalPrice` フィールドに支出制限を指定します。

スポットフリート内のオンデマンドインスタンスとスポットインスタンスに支出制限を設定するには

Amazon EC2 コンソールまたは AWS CLI を使用して、支出制限を設定できます。

(コンソール) スポットフリートを作成するとき、[スポットインスタンスの最大コストを設定] チェックボックスをオンにし、[最大コスト (1 時間あたり) を設定] の値を入力してください。詳細については、「定義済みパラメータを使用してスポットフリートリクエストを作成する」のステップ 6.e を参照してください。

(AWS CLI) request-spot-fleet コマンドと以下のパラメータを使用します。

- オンデマンドインスタンスの場合: `OnDemandMaxTotalPrice` フィールドに支出制限を指定します。
- スポットインスタンスの場合: `SpotMaxTotalPrice` フィールドに支出制限を指定します。

例

以下の例は、2 つの異なるシナリオを示しています。最初の例では、オンデマンドインスタンスに設定されたターゲットキャパシティに達したことを確認すると (`OnDemandTargetCapacity`)、フリートはオンデマンドインスタンスの起動を停止します。2 番目の例では、オンデマンドインスタンスに対する 1 時間あたりの最大支出金額に達すると (`MaxTotalPrice`)、フリートはオンデマンドインスタンスの起動を停止します。

例: ターゲットキャパシティに達したらオンデマンドインスタンスの起動を停止する

m4.large オンデマンドインスタンス に対するリクエストの内容が以下のとおりとします。

- オンデマンド料金: 1 時間あたり 0.10 USD
- OnDemandTargetCapacity: 10
- MaxTotalPrice: 1.50 USD

フリートは、10 個のオンデマンドインスタンスを起動します。合計料金 1.00 USD (10 インスタンス x 0.10 USD) がオンデマンドインスタンスの MaxTotalPrice (1.50 USD) を超えないようにするためです。

例: 最大合計料金に達したらオンデマンドインスタンスの起動を停止する

m4.large オンデマンドインスタンス に対するリクエストの内容が以下のとおりとします。

- オンデマンド料金: 1 時間あたり 0.10 USD
- OnDemandTargetCapacity: 10
- MaxTotalPrice: 0.80 USD

フリートがオンデマンドターゲットキャパシティ (10 個のオンデマンドインスタンス) を起動した場合、1 時間あたりの合計コストは 1.00 USD になります。これは オンデマンドインスタンス の MaxTotalPrice に指定した料金 (0.80 USD) を超えます。支払い可能な額を超えないように、フリートは 8 個のオンデマンドインスタンス (オンデマンドターゲットキャパシティ未満) を起動します。これを超えて起動すると、オンデマンドインスタンスの MaxTotalPrice を超えてしまいます。

バーストパフォーマンスインスタンス

バーストパフォーマンスインスタンスタイプ を使用して スポットインスタンス を起動し、CPU クレジットを蓄積するアイドル時間なしでバーストパフォーマンス スポットインスタンス をすぐに短時間使用する場合は、支払いコストが高くなるのを避けるために、インスタンスを 標準モード で起動することをお勧めします。バーストパフォーマンス スポットインスタンス を Unlimited モード で起動し、すぐに CPU をバーストさせると、余分なクレジットがバーストに消費されます。インスタンスを短時間使用する場合、インスタンスは余分なクレジットに見合うだけの CPU クレジットを蓄積する時間がないため、インスタンスの終了時に余分なクレジットに対して課金されます。

Unlimited モードがバーストパフォーマンス スポットインスタンス に適しているのはバースト用の CPU クレジットが蓄積されるまで、そのインスタンスが十分に長く実行される場合のみです。それ以外の場合は余分なクレジットを支払う必要があるため、バーストパフォーマンス スポットインスタンス は他のインスタンスよりも、使用コストが高くなります。詳細については、Unlimited モードと固定 CPU を使用する場合を参照してください。

起動クレジットは、インスタンスを構成するために十分なコンピューティングリソースを提供し、T2 インスタンスの初期起動を効率的に実現することを意図しています。T2 インスタンスの起動を繰り返して新しい起動クレジットにアクセスすることは許可されていません。CPU が持続的に必要な場合、(一定期間のアイドリングにより) クレジットを獲得して T2 スポットインスタンス の Unlimited モード を使用するか、専用 CPU を搭載したインスタンスタイプを使用します。

EC2 フリートまたはスポットフリートのインスタンスタイプを選択するための属性を指定する

EC2 フリートまたはスポットフリートを作成するときは、フリートのオンデマンドインスタンスとスポットインスタンスを設定するために、1 つ以上のインスタンスタイプを指定する必要があります。インスタンスタイプを手動で指定する代わりに、インスタンスが持つ必要がある属性を指定でき、Amazon EC2 は、それらの属性を持つすべてのインスタンスタイプを識別します。これは 属性ベースのインスタンスタイプの選択 と呼ばれます。例えば、インスタンスに必要な vCPU の最小数および最大数を指定できます。フリートは、これらの vCPU 要件を満たす使用可能なインスタンスタイプを使用してインスタンスを起動します。

属性ベースのインスタンスタイプの選択は、コンテナやウェブフリートの実行、ビッグデータの処理、継続的インテグレーションおよびデプロイ (CI/CD) ツールの実装など、使用するインスタンスタイプについて柔軟に使用できるワークロードとフレームワークに最適です。

利点

属性ベースのインスタンスタイプを選択すると、次の利点があります。

- 適切なインスタンスタイプを簡単に使用 - 利用可能なインスタンスタイプの数が多いため、ワークロードに適したインスタンスタイプを見つけるには時間がかかることがあります。インスタンス属性を指定すると、インスタンスタイプにはワークロードに必要な属性が自動的に設定されます。
- 設定の簡素化 - フリートに複数のインスタンスタイプを手動で指定するには、インスタンスタイプごとに個別の起動テンプレートのオーバーライドを作成する必要があります。ただし、属性ベースのインスタンスタイプを選択すると、複数のインスタンスタイプを提供するには、起動テンプレートまたは起動テンプレートの上書きでインスタンス属性を指定するだけで済みます。

- 新しいインスタンスタイプを自動的に使用 - インスタンスタイプではなくインスタンス属性を指定すると、フリートではリリース時に新しい世代のインスタンスタイプを使用できます。これにより、フリートの設定の将来の対応性も確保されます。
- インスタンスタイプの柔軟性 - インスタンスタイプではなくインスタンス属性を指定すると、フリートはスポットインスタンスを起動するために幅広いインスタンスタイプから選択することができ、インスタンスタイプの柔軟性というスポットのベストプラクティスに準拠することができます。

トピック

- 属性ベースのインスタンスタイプ選択の仕組み
- 料金保護
- パフォーマンス保護
- 考慮事項
- 属性ベースのインスタンスタイプを選択した EC2 フリートを作成する
- 属性ベースのインスタンスタイプを選択してスポットフリートを作成する
- EC2 フリートの有効な設定と無効な設定の例
- スポットフリートの有効な設定と無効な設定の例
- 指定された属性でインスタンスタイプをプレビューする

属性ベースのインスタンスタイプ選択の仕組み

フリート設定で属性ベースのインスタンスタイプの選択を使用するには、インスタンスタイプのリストをインスタンスが必要とするインスタンス属性のリストに置き換えます。EC2 フリートまたはスポットフリートは、指定されたインスタンス属性を持つ使用可能なインスタンスタイプでインスタンスを起動します。

トピック

- インスタンス属性のタイプ
- 属性ベースのインスタンスタイプの選択を設定する場所
- フリートをプロビジョニングするときに、EC2 フリートまたはスポットフリートが属性ベースのインスタンスタイプの選択をどのように使用するか

インスタンス属性のタイプ

コンピューティング要件を表現するために指定できるインスタンス属性はいくつかあります。

- vCPU 数 – インスタンスあたりの vCPU の最小数と最大数。
- メモリ – インスタンスあたりのメモリの最小および最大 GiB。
- ローカルストレージ – EBS ボリュームとインスタンスストアボリュームのどちらをローカルストレージに使用するか。
- バースト可能なパフォーマンス – T4g、T3a、T3、および T2 タイプを含む T インスタンスファミリーを使用するかどうか。

各属性の説明およびデフォルト値については、「Amazon EC2 API リファレンス」の「InstanceRequirements」を参照してください。

属性ベースのインスタンスタイプの選択を設定する場所

コンソールと AWS CLI のどちらを使用するかによって、属性ベースのインスタンスタイプ選択のインスタンス属性を次のように指定できます。

コンソールでは、次のフリート設定コンポーネントでインスタンス属性を指定できます。

- 起動テンプレートでフリートリクエストの起動テンプレートを参照する
- (スポットフリートのみ) フリートリクエスト内

AWS CLI で、以下のフリート設定コンポーネントのいずれかまたはすべてでインスタンスの属性を指定することができます。

- 起動テンプレートでフリートリクエストの起動テンプレートを参照する
- 起動テンプレートの上書きで

  異なる AMI を使用するインスタンスを混在させたい場合は、複数の起動テンプレートの上書きでインスタンス属性を指定できます。例えば、異なるインスタンスタイプで x86 および ARM ベースのプロセッサを使用できます。
- (スポットフリートのみ) 起動仕様内

フリートをプロビジョニングするときに、EC2 フリートまたはスポットフリートが属性ベースのインスタンスタイプの選択をどのように使用するか

EC2 フリートまたはスポットフリートは、以下の方法でフリートをプロビジョニングします。

- これらのフリートは、指定された属性を持つインスタンスタイプを識別します。
- また、料金保護を使用して、除外するインスタンスタイプを決定します。
- インスタンスタイプが一致する AWS リージョンまたはアベイラビリティーゾーンに基づいて、インスタンスの起動を検討するキャパシティプールを決定します。
- 指定された配分戦略を適用して、インスタンスを起動するキャパシティプールを決定します。

  属性ベースのインスタンスタイプの選択では、フリートをプロビジョニングするキャパシティプールは選択されません。これは、配分戦略のジョブです。

  配分戦略を指定すると、フリートは指定された配分戦略に従ってインスタンスを起動します。
  - スポットインスタンスの場合、属性ベースのインスタンスタイプの選択では料金キャパシティ最適化、キャパシティ最適化、最低料金の各配分戦略がサポートされます。ただし、最低料金スポット配分戦略は、スポットインスタンスの中断リスクが最も高いため、推奨されないことに注意してください。
  - オンデマンドインスタンスの場合、属性ベースのインスタンスタイプの選択では最低料金配分戦略がサポートされます。
- 指定されたインスタンス属性を持つインスタンスタイプの容量がない場合、インスタンスは起動できず、フリートはエラーを返します。

料金保護

料金保護とは、EC2 フリートまたはスポットフリートが指定した属性に適合した場合でも、コストが高すぎると見なされたインスタンスタイプは利用できないようにする機能です。料金保護を使用するには、料金のしきい値を設定します。Amazon EC2 が属性を持つインスタンスタイプを選択すると、しきい値を超える料金が設定されたインスタンスタイプは除外されます。

Amazon EC2 が料金のしきい値を計算する方法は、次のとおりです。

- Amazon EC2 はまず、属性に一致するものから最低料金のインスタンスタイプを識別します。
- Amazon EC2 は、料金保護パラメータに指定した値 (パーセンテージで表される) を受け取り、識別されたインスタンスタイプの料金でそれを乗算します。その結果、料金しきい値として使用される料金になります。

オンデマンドインスタンスとスポットインスタンスには個別の料金しきい値があります。

属性ベースのインスタンスタイプを選択してフリートを作成すると、料金保護がデフォルトで有効になります。デフォルト値のままにすることも、独自の値を指定することもできます。

料金保護をオフにすることもできます。料金保護のしきい値を指定しない場合は、999999 などの高いパーセンテージ値を指定します。

トピック

- 最低料金のインスタンスタイプを特定する方法
- オンデマンドインスタンスの料金保護
- スポットインスタンスの料金保護
- 料金保護のしきい値を指定する

最低料金のインスタンスタイプを特定する方法

Amazon EC2 は、指定した属性に一致するものから最低料金のインスタンスタイプを特定することで、料金のしきい値に基づく料金を決定します。これは、次の方法で行います。

- まず、属性に一致する現行世代の C、M、または R インスタンスタイプを調べます。一致するものがある場合は、最低料金のインスタンスタイプを特定します。
- 一致するものがない場合は、属性に一致する現行世代のインスタンスタイプを調べます。一致するものがある場合は、最低料金のインスタンスタイプを特定します。
- 一致するものがない場合は、属性に一致する以前の世代のインスタンスタイプを調べ、最低料金のインスタンスタイプを特定します。

オンデマンドインスタンスの料金保護

オンデマンドインスタンスタイプの料金保護のしきい値は、特定された最低料金のオンデマンドインスタンスタイプ `OnDemandMaxPricePercentageOverLowestPrice`よりも高いパーセンテージで計算されます。支払い可能なパーセンテージを高く指定します。このパラメータを指定しない場合は、デフォルト値の `20` を使用して、識別された料金よりも 20% 高い料金保護しきい値が計算されます。

例えば、特定されたオンデマンドインスタンスの料金が `0.4271` で、`25` を指定した場合、料金のしきい値は `0.4271` より 25% 高くなります。これは、次のように計算されます: `0.4271 * 1.25 = 0.533875`。計算された料金は、オンデマンドインスタンスに対して支払うことができる最大額であ

り、この例では、Amazon EC2 は 0.533875 を超えるコストがかかるオンデマンドインスタンスタイプを除外します。

スポットインスタンスの料金保護

デフォルトでは、Amazon EC2 は最適なスポットインスタンス料金保護を自動的に適用し、幅広いインスタンスタイプから一貫して選択してください。料金保護を手動で設定することもできます。ただし、Amazon EC2 に任せることで、スポット容量が満たされる可能性を高めることができます。

料金保護は、次のいずれかのオプションを使用して手動で指定できます。料金保護を手動で設定する場合は、最初のオプションを使用することをお勧めします。

- 特定された最低料金のオンデマンドインスタンスタイプ **MaxSpotPriceAsPercentageOfOptimalOnDemandPrice**のパーセンテージ

  例えば、特定されたオンデマンドインスタンスタイプの料金が 0.4271 で、60 を指定した場合、料金のしきい値は 0.4271 の 60% になります。これは、次のように計算されます: 0.4271 * 0.60 = 0.25626。計算された料金は、スポットインスタンスに対して支払うことができる最大額であり、この例では、Amazon EC2 は 0.25626 を超えるコストがかかるスポットインスタンスタイプを除外します。
- 特定された最低料金のスポットインスタンスタイプ **SpotMaxPricePercentageOverLowestPrice**よりも高いパーセンテージ

  例えば、特定されたスポットインスタンスタイプの料金が 0.1808 で、25 を指定した場合、料金のしきい値は 0.1808 より 25% 高くなります。これは、次のように計算されます: 0.1808 * 1.25 = 0.226。計算された料金は、スポットインスタンスに対して支払うことができる最大額であり、この例では、Amazon EC2 は 0.266 を超えるコストがかかるスポットインスタンスタイプを除外します。スポット料金が変動する可能性があり、料金保護のしきい値も変動する可能性があるため、このパラメータの使用はお勧めしません。

料金保護のしきい値を指定する

AWS CLI を使用して料金保護のしきい値を指定するには

AWS CLI を使用して EC2 フリートまたはスポットフリートを作成するときに、属性ベースのインスタンスタイプの選択が使用されるようにフリートを設定してから、次の手順を実行します。

- オンデマンドインスタンスの料金保護のしきい値を指定するには、JSON 設定ファイルの InstanceRequirements 構造の OnDemandMaxPricePercentageOverLowestPrice で、料金保護のしきい値をパーセンテージ (%) で入力してください。

- スポットインスタンスの料金保護のしきい値を指定するには、JSON 設定ファイルの `InstanceRequirements` 構造で、次のいずれかのパラメータを指定します。
  - `MaxSpotPriceAsPercentageOfOptimalOnDemandPrice` で、料金保護のしきい値をパーセンテージ (%) で入力してください。
  - `SpotMaxPricePercentageOverLowestPrice` で、料金保護のしきい値をパーセンテージ (%) で入力してください。

詳細については「属性ベースのインスタンスタイプを選択した EC2 フリートを作成する」または「属性ベースのインスタンスタイプを選択してスポットフリートを作成する」を参照してください。

(スポットフリートのみ) コンソールを使用して料金保護のしきい値を指定するには

コンソールでスポットフリートを作成するときに、属性ベースのインスタンスタイプの選択が使用されるようにフリートを設定してから、次の手順を実行します。

- オンデマンドインスタンスの料金保護のしきい値を指定するには、[追加のインスタンス属性] で、[オンデマンド料金の保護]、[属性を追加] を選択してから、料金保護のしきい値をパーセンテージで入力してください。
- スポットインスタンスの料金保護のしきい値を指定するには、[追加のインスタンス属性] で、[スポット料金の保護]、[属性を追加] を選択し、料金のベースとなる基準値を選択してから、料金保護のしきい値をパーセンテージで入力してください。

> **Note**
>
> フリートを作成するときに、`TargetCapacityUnitType` を `vcpu` または `memory-mib` に設定すると、インスタンスごとの料金ではなく、vCPU ごとまたはメモリごとの料金に基づいて料金保護のしきい値が適用されます。

パフォーマンス保護

パフォーマンス保護は、EC2 フリートまたはスポットフリートが指定されたパフォーマンスベースラインと同等以上のインスタンスタイプを使用できるようにする機能です。パフォーマンス保護を使用するには、ベースラインリファレンスとしてインスタンスファミリーを指定します。指定されたインスタンスファミリーの機能は、許容される最低レベルのパフォーマンスを確立します。Amazon EC2 がフリートのインスタンスタイプを選択すると、指定した属性およびパフォーマンスベースラインが考慮されます。パフォーマンスベースラインを下回るインスタンスタイプは、他の指定された

属性と一致しても、自動的に選択から除外されます。選択したすべてのインスタンスタイプが、指定されたインスタンスファミリーによって確立されたベースラインと同等以上のパフォーマンスを提供できるようにします。Amazon EC2 はこのベースラインを使用してインスタンスタイプの選択をガイドしますが、選択したインスタンスタイプがすべてのアプリケーションのベースラインを常に上回る保証はありません。

現在、この機能はベースラインパフォーマンス係数として CPU パフォーマンスのみをサポートしています。指定されたインスタンスファミリーの CPU プロセッサの CPU パフォーマンスがパフォーマンスベースラインとして使用され、選択したインスタンスタイプがこのベースラインと同等以上であるようにします。同じ CPU プロセッサを持つインスタンスファミリーは、ネットワークまたはディスクのパフォーマンスが異なっても、同じフィルタリング結果になります。例えば、ベースラインリファレンスとして `c6in` または `c6i` を指定すると、両方のインスタンスファミリーが同じ CPU プロセッサを使用するため、パフォーマンスベースのフィルタリング結果が同じになります。

サポートされていないインスタンスファミリー

次のインスタンスファミリーは、パフォーマンス保護ではサポートされていません。

- 汎用: Mac1 | Mac2 | Mac2-m1ultra | Mac2-m2 | Mac2-m2pro | M1 | M2 | T1
- コンピューティングの最適化: C1
- メモリ最適化: U-3tb1 | U-6tb1 | U-9tb1 | U-12tb1 | U-18tb1 | u-24tb1 | U7i-12tb | U7in-16tb | U7in-24tb | U7in-32tb
- 高速コンピューティング: G3 | G3s | P3dn | P4d | P5
- ハイパフォーマンスコンピューティング: Hpc7g

サポートされているインスタンスファミリーを指定してパフォーマンス保護を有効にした場合、返されるインスタンスタイプは上記のサポートされていないインスタンスファミリーを除外します。

サポートされていないインスタンスファミリーをベースラインパフォーマンスの値として指定した場合、API は GetInstanceTypesFromInstanceRequirements に空のレスポンスを返し、CreateFleet、RequestSpotFleet、ModifyFleet、ModifySpotFleetRequest に例外を返します。

例: CPU パフォーマンスベースラインの設定

次の例では、インスタンス要件は、`c6i` インスタンスファミリーと同等のパフォーマンスの CPU コアを持つインスタンスタイプで起動することです。性能が下回る CPU プロセッサーが指定した他のインスタンス要件 (vCPU 数など) を満たしても、その CPU プロセッサーを持つインスタンスタイプを除外します。例えば、指定したインスタンス属性に 4 つの vCPUs および 16 GB のメモリが含ま

れている場合、これらの属性を持っても c6i より CPU パフォーマンスが低いインスタンスタイプは選択から除外されます。

```
"BaselinePerformanceFactors": {
        "Cpu": {
            "References": [
                {
                    "InstanceFamily": "c6i"
                }
            ]
        }
```

考慮事項

- EC2 フリートまたはスポットフリートでは、インスタンスタイプまたはインスタンス属性のいずれかを指定できますが、両方を同時に指定することはできません。

  CLI を使用する場合、起動テンプレートの上書きによって起動テンプレートが上書きされます。例えば、起動テンプレートにインスタンスタイプが含まれ、起動テンプレートの上書きにインスタンス属性が含まれている場合、インスタンス属性によって識別されるインスタンスは、起動テンプレートのインスタンスタイプを上書きします。
- CLI を使用していて、インスタンス属性の上書きを指定する場合、重みまたは優先順位も指定できません。
- リクエスト設定では、最大 4 つの InstanceRequirements 構造を指定できます。

属性ベースのインスタンスタイプを選択した EC2 フリートを作成する

属性ベースのインスタンスタイプの選択が使用されるように EC2 フリートを設定できます。Amazon EC2 コンソールを使用して EC2 フリートを作成することはできません。

属性に基づくインスタンスタイプ選択のための属性は、InstanceRequirements 構造で指定されます。InstanceRequirements がフリート設定に含まれる場合、InstanceType と WeightedCapacity は除外しなければならず、インスタンス属性と同時にフリート設定を決定することはできません。

AWS CLI および PowerShell の例では、次の属性が指定されています。

- VCpuCount – 最小 2 つの vCPU と最大 4 つの vCPU。最大制限が必要ない場合は、最大値を省略できます。

- MemoryMiB – 最小 8 GiB のメモリと最大 16 GiB のメモリ。最大制限が必要ない場合は、最大値を省略できます。

この設定では、2 ~ 4 個の vCPU と 8 ~ 16 GiB のメモリを持つインスタンスタイプが識別されます。ただし、EC2 フリートがフリートをプロビジョニングする場合、価格保護と配分戦略によって一部のインスタンスタイプが除外される場合があります。

AWS CLI

属性ベースのインスタンスタイプ選択を使用する EC2 フリートを作成するには

create-fleet コマンドを使用して、EC2 フリートを作成します。JSON ファイルでフリート設定を指定します。

```
aws ec2 create-fleet \
    --region us-east-1 \
    --cli-input-json file://file_name.json
```

次の例の *file_name*.json ファイルには、属性ベースのインスタンスタイプ選択を使用するように EC2 フリートを設定するパラメータが含まれています。

```
{
    "SpotOptions": {
        "AllocationStrategy": "price-capacity-optimized"
    },
    "LaunchTemplateConfigs": [{
        "LaunchTemplateSpecification": {
            "LaunchTemplateName": "my-launch-template",
            "Version": "1"
        },
        "Overrides": [{
            "InstanceRequirements": {
                "VCpuCount": {
                    "Min": 2,
                    "Max": 4
                },
                "MemoryMiB": {
                    "Min": 8192,
                    "Max": 16384
                }
            }
```

```
        }]
    }],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 20,
        "DefaultTargetCapacityType": "spot"
    },
    "Type": "instant"
}
```

PowerShell

属性ベースのインスタンスタイプ選択を使用する EC2 フリートを作成するには

New-EC2Fleet コマンドレットを使用します。

```
$vcpuCount = New-Object Amazon.EC2.Model.VCpuCountRangeRequest
$vcpuCount.Min = 2
$vcpuCount.Max = 4
$memoryMiB = New-Object Amazon.EC2.Model.MemoryMiBRequest
$memoryMiB.Min = 8192
$memoryMiB.Max = 16384
$instanceRequirements = New-Object Amazon.EC2.Model.InstanceRequirementsRequest
$instanceRequirements.VCpuCount = $vcpuCount
$instanceRequirements.MemoryMiB = $memoryMiB

$launchTemplateSpec = New-Object
 Amazon.EC2.Model.FleetLaunchTemplateSpecificationRequest
$launchTemplateSpec.LaunchTemplateName = "my-launch-template"
$launchTemplateSpec.Version = "1"
$override = New-Object Amazon.EC2.Model.FleetLaunchTemplateOverridesRequest
$override.InstanceRequirements = $instanceRequirements
$launchTemplateConfig = New-Object Amazon.EC2.Model.FleetLaunchTemplateConfigRequest
$launchTemplateConfig.LaunchTemplateSpecification = $launchTemplateSpec
$launchTemplateConfig.Overrides = @($override)

New-EC2Fleet `
    -SpotOptions_AllocationStrategy "price-capacity-optimized" `
    -LaunchTemplateConfig @($launchTemplateConfig) `
    -TargetCapacitySpecification_DefaultTargetCapacityType "spot" `
    -TargetCapacitySpecification_TotalTargetCapacity 20 `
    -Type "instant"
```

属性ベースのインスタンスタイプを選択してスポットフリートを作成する

> **Important**
>
> スポットフリートは、計画された投資なしでレガシー API を使用します。代わりに EC2 フリートまたは Auto Scaling グループを使用することをお勧めします。詳細については、「フリートにどの方法を使用するのが最適か」を参照してください。

属性ベースのインスタンスタイプの選択が使用されるようにフリートを設定できます。

属性に基づくインスタンスタイプ選択のための属性は、`InstanceRequirements` 構造で指定されます。`InstanceRequirements` がフリート設定に含まれる場合、`InstanceType` と `WeightedCapacity` は除外しなければならず、インスタンス属性と同時にフリート設定を決定することはできません。

AWS CLI および PowerShell の例では、次の属性が指定されています。

- `VCpuCount` – 最小 2 つの vCPU と最大 4 つの vCPU。最大制限が必要ない場合は、最大値を省略できます。
- `MemoryMiB` – 最小 8 GiB のメモリと最大 16 GiB のメモリ。最大制限が必要ない場合は、最大値を省略できます。

この設定では、2 ~ 4 個の vCPU と 8 ~ 16 GiB のメモリを持つインスタンスタイプが識別されます。ただし、スポットフリートがフリートをプロビジョニングする場合、価格保護と配分戦略によって一部のインスタンスタイプが除外される場合があります。

Console

属性ベースのインスタンスタイプの選択のためにスポットフリートを設定するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Spot Requests] (スポットリクエスト) を選択した後、[Request Spot Instances] (スポットインスタンスのリクエスト) を選択してください。
3. 手順に従ってスポットフリートを作成します。詳細については、「定義済みパラメータを使用してスポットフリートリクエストを作成する」を参照してください。

   スポットフリートを作成するときに、属性ベースのインスタンスタイプ選択にフリートを次のように設定します。

a. [Instance type requirements] (インスタンスタイプの要件) では、[Specify instance attributes that match your compute requirements] (コンピューティング要件に一致するインスタンス属性を指定する) を選択してください。

b. [vCPUs] に、希望する vCPU の最小数と最大数を入力してください。制限なしを指定するには、[No minimum] (最小値なし)、[No maximum] (最大値なし)、または両方を選択してください。

c. [Memory (GiB)] (メモリ (GiB)) に、希望する最小値と最大値を入力してください。制限なしを指定するには、[No minimum] (最小値なし)、[No maximum] (最大値なし)、または両方を選択してください。

d. (オプション) [Additional instance attributes] (その他のインスタンス属性) では、オプションで 1 つ以上の属性を指定して、コンピューティング要件をより詳細に表現できます。追加の属性は、リクエストにさらに制約を追加します。

e. (オプション) [Preview matching instance types] (一致するインスタンスタイプをプレビューする) を展開して、指定した属性を持つインスタンスタイプを表示します。

AWS CLI

属性ベースのインスタンスタイプの選択のためにスポットフリートを設定するには

スポットフリートリクエストを作成するには、 request-spot-fleet マンドを使用します。JSON ファイルでフリート設定を指定します。

```
aws ec2 request-spot-fleet \
    --region us-east-1 \
    --spot-fleet-request-config file://file_name.json
```

次の例の *file_name*.json ファイルには、属性ベースのインスタンスタイプ選択を使用するようにスポットフリートを設定するパラメータが含まれています。

```
{
    "AllocationStrategy": "priceCapacityOptimized",
    "TargetCapacity": 20,
    "Type": "request",
    "LaunchTemplateConfigs": [{
        "LaunchTemplateSpecification": {
            "LaunchTemplateName": "my-launch-template",
            "Version": "1"
        },
```

```
        "Overrides": [{
            "InstanceRequirements": {
                "VCpuCount": {
                    "Min": 2,
                    "Max": 4
                },
                "MemoryMiB": {
                    "Min": 8192,
                    "Max": 16384
                }
            }
        }]
    }]
}
```

PowerShell

属性ベースのインスタンスタイプの選択のためにスポットフリートを設定するには

Request-EC2SpotFleet コマンドレットを使用します。

```
$vcpuCount = New-Object Amazon.EC2.Model.VCpuCountRange
$vcpuCount.Min = 2
$vcpuCount.Max = 4
$memoryMiB = New-Object Amazon.EC2.Model.MemoryMiB
$memoryMiB.Min = 8192
$memoryMiB.Max = 16384
$instanceRequirements = New-Object Amazon.EC2.Model.InstanceRequirements
$instanceRequirements.VCpuCount = $vcpuCount
$instanceRequirements.MemoryMiB = $memoryMiB

$launchTemplateSpec = New-Object Amazon.EC2.Model.FleetLaunchTemplateSpecification
$launchTemplateSpec.LaunchTemplateName = "my-launch-template"
$launchTemplateSpec.Version = "1"
$override = New-Object Amazon.EC2.Model.LaunchTemplateOverrides
$override.InstanceRequirements = $instanceRequirements
$launchTemplateConfig = New-Object Amazon.EC2.Model.LaunchTemplateConfig
$launchTemplateConfig.LaunchTemplateSpecification = $launchTemplateSpec
$launchTemplateConfig.Overrides = @($override)

Request-EC2SpotFleet `
    -SpotFleetRequestConfig_AllocationStrategy "PriceCapacityOptimized" `
    -SpotFleetRequestConfig_TargetCapacity 20 `
    -SpotFleetRequestConfig_Type "Request" `
```

```
    -SpotFleetRequestConfig_LaunchTemplateConfig $launchTemplateConfig
```

EC2 フリートの有効な設定と無効な設定の例

AWS CLI を使用して EC2 フリートを作成する場合は、フリートの設定が有効であることを確認する必要があります。次の例は、有効な設定と無効な設定を示しています。

次のものが含まれている場合、設定は無効と見なされます。

- InstanceRequirements および InstanceType を持つ 1 つの Overrides 構造
- 一つは InstanceRequirements、もう一つは InstanceType を持つ 2 つの Overrides 構造
- 同じ LaunchTemplateSpecification 内で属性値が重複している 2 つの InstanceRequirements 構造

設定例

- 有効な設定: 上書きを含む単一の起動テンプレート
- 有効な設定: 複数のインスタンス要件を持つ単一の起動テンプレート
- 有効な設定: 2 つの起動テンプレート、それぞれに上書きがある
- 有効な設定: InstanceRequirements のみ指定され、重複する属性値なし
- 設定が無効です: Overrides が InstanceRequirements および InstanceType を含んでいる
- 設定が無効です: 2 つの Overrides に InstanceRequirements および InstanceType が含まれている
- 設定が無効です: 重複する属性値

有効な設定: 上書きを含む単一の起動テンプレート

以下の設定は有効です。これには、1 つの起動テンプレートと、InstanceRequirements 構造を含む 1 つの Overrides が含まれています。以下に、構成例をテキストで説明します。

```
{
        "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "My-launch-template",
                "Version": "1"
                },
                "Overrides": [
```

```
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 2,
                            "Max": 8
                        },
                        "MemoryMib": {
                            "Min": 0,
                            "Max": 10240
                        },
                        "MemoryGiBPerVCpu": {
                            "Max": 10000
                        },
                        "RequireHibernateSupport": true
                    }
                }
            ]
        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 5000,
        "DefaultTargetCapacityType": "spot",
        "TargetCapacityUnitType": "vcpu"
        }
    }
}
```

InstanceRequirements

属性ベースのインスタンス選択を使用するには、フリート設定に InstanceRequirements 構造を含め、フリート内のインスタンスに必要な属性を指定する必要があります。

前の例に、以下のインスタンス属性が指定されています。

- VCpuCount - インスタンスタイプには、2 個以上、最大 8 個の vCPU が必要です。
- MemoryMiB - インスタンスタイプには最大 10,240 MiB のメモリが必要です。最小数が 0 の場合、最小制限がないことを示します。
- MemoryGiBPerVCpu - インスタンスタイプには、vCPU あたり最大 10,000 GiB のメモリが必要です。Min パラメータはオプションです。省略すると、最小制限がないことを示します。

TargetCapacityUnitType

TargetCapacityUnitType パラメータは、ターゲット容量の単位を指定します。この例では、ターゲット容量は 5000 であり、ターゲット容量ユニットタイプは vcpu で、これを組み合わせて 5,000 vCPU の希望するターゲット容量を指定します。EC2 フリートは、フリート内の vCPU の総数が 5,000 vCPU になるように、十分なインスタンスを起動します。

有効な設定: 複数のインスタンス要件を持つ単一の起動テンプレート

以下の設定は有効です。これには、1 つの起動テンプレートと、InstanceRequirements 構造を含む 2 つの Overrides が含まれています。InstanceRequirements で指定された属性は、値が重複していないため有効です。最初の InstanceRequirements 構造は VCpuCount の 0～2 vCPU を指定し、2 つ目の InstanceRequirements 構造は 4～8 vCPU を指定しています。

```
{
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                },
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 4,
                            "Max": 8
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                }
              ]
```

```
            }
        ],
        "TargetCapacitySpecification": {
            "TotalTargetCapacity": 1,
            "DefaultTargetCapacityType": "spot"
        }
    }
}
```

有効な設定: 2 つの起動テンプレート、それぞれに上書きがある

以下の設定は有効です。これには 2 つの起動テンプレートが含まれ、各起動テンプレートには 1 つの InstanceRequirements 構造を含む Overrides 構造が 1 つ含まれています。この設定は、同じフリートで arm と x86 のアーキテクチャをサポートする場合に有効です。

```
{
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "armLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                },
                {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "x86LaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
```

```
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                }
              ]
            }
        ],
         "TargetCapacitySpecification": {
            "TotalTargetCapacity": 1,
            "DefaultTargetCapacityType": "spot"
        }
    }
}
```

有効な設定: **InstanceRequirements** のみ指定され、重複する属性値なし

以下の設定は有効です。2 つの LaunchTemplateSpecification 構造が含まれ、各構造にそれぞれ起動テンプレートと、Overrides 構造を含む InstanceRequirements 構造が含まれています。InstanceRequirements で指定された属性は、値が重複していないため有効です。最初の InstanceRequirements 構造は VCpuCount の 0～2 vCPU を指定し、2 つ目の InstanceRequirements 構造は 4～8 vCPU を指定しています。

```
{
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
```

```
                    }
                }
              ]
            },
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyOtherLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 4,
                            "Max": 8
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                }
              ]
            }
        ],
        "TargetCapacitySpecification": {
            "TotalTargetCapacity": 1,
            "DefaultTargetCapacityType": "spot"
        }
    }
}
```

設定が無効です: **Overrides** が **InstanceRequirements** および **InstanceType** を含んでいる

以下は設定が有効ではありません。Overrides 構造体には InstanceRequirements および InstanceType が両方含まれています。Overrides では、InstanceRequirements または InstanceType のどちらかを指定できますが、両方は指定できません。

```
{
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyLaunchTemplate",
                    "Version": "1"
```

```
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                },
                {
                    "InstanceType": "m5.large"
                }
              ]
            }
        ],
        "TargetCapacitySpecification": {
            "TotalTargetCapacity": 1,
            "DefaultTargetCapacityType": "spot"
        }
    }
}
```

設定が無効です: 2 つの **Overrides** に **InstanceRequirements** および **InstanceType** が含まれている

以下は設定が有効ではありません。Overrides 構造に InstanceRequirements および InstanceType が両方含まれています。異なる Overrides 構造にある場合、InstanceRequirements または InstanceType を指定できますが、両方を指定することはできません。

```
{
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
```

```
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                }
              ]
            },
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyOtherLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceType": "m5.large"
                }
              ]
            }
        ],
         "TargetCapacitySpecification": {
            "TotalTargetCapacity": 1,
            "DefaultTargetCapacityType": "spot"
        }
    }
}
```

設定が無効です: 重複する属性値

以下は設定が有効ではありません。2 つの InstanceRequirements 構造がそれぞれ "VCpuCount": {"Min": 0, "Max": 2} を含んでいます。これらの属性の値が重複するため、容量プールが重複します。

```
{
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyLaunchTemplate",
                    "Version": "1"
```

```
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    },
                    {
                      "InstanceRequirements": {
                          "VCpuCount": {
                              "Min": 0,
                              "Max": 2
                          },
                          "MemoryMiB": {
                              "Min": 0
                          }
                      }
                  }
                }
              ]
            }
        ],
         "TargetCapacitySpecification": {
            "TotalTargetCapacity": 1,
            "DefaultTargetCapacityType": "spot"
        }
    }
}
```

スポットフリートの有効な設定と無効な設定の例

AWS CLI を使用してスポットフリートを作成する場合は、フリート設定が有効であることを確認する必要があります。次の例は、有効な設定と無効な設定を示しています。

次のものが含まれている場合、設定は無効と見なされます。

- InstanceRequirements および InstanceType を持つ 1 つの Overrides 構造
- 一つは InstanceRequirements、もう一つは InstanceType を持つ 2 つの Overrides 構造

- 同じ LaunchTemplateSpecification 内で属性値が重複している 2 つの InstanceRequirements 構造

設定例

- 有効な設定: 上書きを含む単一の起動テンプレート
- 有効な設定: 複数のインスタンス要件を持つ単一の起動テンプレート
- 有効な設定: 2 つの起動テンプレート、それぞれに上書きがある
- 有効な設定: InstanceRequirements のみ指定され、重複する属性値なし
- 設定が無効です: Overrides が InstanceRequirements および InstanceType を含んでいる
- 設定が無効です: 2 つの Overrides に InstanceRequirements および InstanceType が含まれている
- 設定が無効です: 重複する属性値

有効な設定: 上書きを含む単一の起動テンプレート

以下の設定は有効です。これには、1 つの起動テンプレートと、InstanceRequirements 構造を含む 1 つの Overrides が含まれています。以下に、構成例をテキストで説明します。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "priceCapacityOptimized",
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "My-launch-template",
                "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 2,
                            "Max": 8
                        },
                        "MemoryMib": {
                            "Min": 0,
```

```
                            "Max": 10240
                        },
                        "MemoryGiBPerVCpu": {
                            "Max": 10000
                        },
                        "RequireHibernateSupport": true
                    }
                }
            ]
        }
    ],
        "TargetCapacity": 5000,
            "OnDemandTargetCapacity": 0,
            "TargetCapacityUnitType": "vcpu"
    }
}
```

InstanceRequirements

属性ベースのインスタンス選択を使用するには、フリート設定に InstanceRequirements 構造を含め、フリート内のインスタンスに必要な属性を指定する必要があります。

前の例に、以下のインスタンス属性が指定されています。

- VCpuCount - インスタンスタイプには、2 個以上、最大 8 個の vCPU が必要です。
- MemoryMiB - インスタンスタイプには最大 10,240 MiB のメモリが必要です。最小数が 0 の場合、最小制限がないことを示します。
- MemoryGiBPerVCpu - インスタンスタイプには、vCPU あたり最大 10,000 GiB のメモリが必要です。Min パラメータはオプションです。省略すると、最小制限がないことを示します。

TargetCapacityUnitType

TargetCapacityUnitType パラメータは、ターゲット容量の単位を指定します。この例では、ターゲット容量は 5000 であり、ターゲット容量ユニットタイプは vcpu で、これを組み合わせて 5,000 vCPU の希望するターゲット容量を指定します。スポットフリートは、フリート内の vCPU の総数が 5,000 vCPU になるように、十分なインスタンスを起動します。

有効な設定: 複数のインスタンス要件を持つ単一の起動テンプレート

以下の設定は有効です。これには、1 つの起動テンプレートと、InstanceRequirements 構造を含む 2 つの Overrides が含まれています。InstanceRequirements で指定された属性は、値が

重複していないため有効です。最初の InstanceRequirements 構造は VCpuCount の 0～2 vCPU を指定し、2 つ目の InstanceRequirements 構造は 4～8 vCPU を指定しています。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "priceCapacityOptimized",
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                },
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 4,
                            "Max": 8
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                }
              ]
            }
        ],
        "TargetCapacity": 1,
        "OnDemandTargetCapacity": 0,
```

```
        "Type": "maintain"
    }
}
```

有効な設定: 2 つの起動テンプレート、それぞれに上書きがある

以下の設定は有効です。これには 2 つの起動テンプレートが含まれ、各起動テンプレートには 1 つの InstanceRequirements 構造を含む Overrides 構造が 1 つ含まれています。この設定は、同じフリートで arm と x86 のアーキテクチャをサポートする場合に有効です。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "priceCapacityOptimized",
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "armLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                },
                {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "x86LaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
```

```
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                }
              ]
            }
        ],
        "TargetCapacity": 1,
        "OnDemandTargetCapacity": 0,
        "Type": "maintain"
    }
}
```

有効な設定: **InstanceRequirements** のみ指定され、重複する属性値なし

以下の設定は有効です。2 つの LaunchTemplateSpecification 構造が含まれ、各構造にぞれぞれ起動テンプレートと、Overrides 構造を含む InstanceRequirements 構造が含まれています。InstanceRequirements で指定された属性は、値が重複していないため有効です。最初の InstanceRequirements 構造は VCpuCount の 0～2 vCPU を指定し、2 つ目の InstanceRequirements 構造は 4～8 vCPU を指定しています。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "priceCapacityOptimized",
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
```

```
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                }
              ]
            },
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyOtherLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 4,
                            "Max": 8
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                }
              ]
            }
        ],
        "TargetCapacity": 1,
        "OnDemandTargetCapacity": 0,
        "Type": "maintain"
    }
}
```

設定が無効です: **Overrides** が **InstanceRequirements** および **InstanceType** を含んでいる

以下は設定が有効ではありません。Overrides 構造体には InstanceRequirements および InstanceType が両方含まれています。Overrides では、InstanceRequirements または InstanceType のどちらかを指定できますが、両方は指定できません。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "priceCapacityOptimized",
```

```
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                },
                {
                    "InstanceType": "m5.large"
                }
              ]
            }
        ],
        "TargetCapacity": 1,
        "OnDemandTargetCapacity": 0,
        "Type": "maintain"
    }
}
```

設定が無効です: 2 つの **Overrides** に **InstanceRequirements** および **InstanceType** が含まれている

以下は設定が有効ではありません。Overrides 構造に InstanceRequirements および InstanceType が両方含まれています。異なる Overrides 構造にある場合、InstanceRequirements または InstanceType を指定できますが、両方を指定することはできません。

```
{
    "SpotFleetRequestConfig": {
```

```
        "AllocationStrategy": "priceCapacityOptimized",
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                }
              ]
            },
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyOtherLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceType": "m5.large"
                }
              ]
            }
        ],
        "TargetCapacity": 1,
        "OnDemandTargetCapacity": 0,
        "Type": "maintain"
    }
}
```

設定が無効です: 重複する属性値

以下は設定が有効ではありません。2 つの InstanceRequirements 構造がそれぞれ "VCpuCount": {"Min": 0, "Max": 2} を含んでいます。これらの属性の値が重複するため、容量プールが重複します。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "priceCapacityOptimized",
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    },
                    {
                      "InstanceRequirements": {
                          "VCpuCount": {
                              "Min": 0,
                              "Max": 2
                          },
                          "MemoryMiB": {
                              "Min": 0
                          }
                      }
                  }
                }
              ]
            }
```

```
        ],
        "TargetCapacity": 1,
        "OnDemandTargetCapacity": 0,
        "Type": "maintain"
    }
}
```

指定された属性でインスタンスタイプをプレビューする

get-instance-types-from-instance-requirements マンドを使用して、指定した属性に一致するインスタンスタイプをプレビューします。これは、インスタンスを起動せずにリクエスト設定で指定する属性を調べる場合に特に便利です。このコマンドでは、使用可能な容量は考慮されません。

AWS CLI を使用して属性を指定してインスタンスタイプのリストをプレビューするには

1. (オプション) 指定可能なすべての属性を生成するには、get-instance-types-from-instance-requirements コマンドと --generate-cli-skeleton パラメータを使用します。オプションで、input > *attributes.json* を使用して出力を保存用ファイルに送ることができます。

   ```
   aws ec2 get-instance-types-from-instance-requirements \
       --region us-east-1 \
       --generate-cli-skeleton input > attributes.json
   ```

   正常な出力

   ```
   {
       "DryRun": true,
       "ArchitectureTypes": [
           "i386"
       ],
       "VirtualizationTypes": [
           "hvm"
       ],
       "InstanceRequirements": {
           "VCpuCount": {
               "Min": 0,
               "Max": 0
           },
           "MemoryMiB": {
               "Min": 0,
               "Max": 0
           },
   ```

```
        "CpuManufacturers": [
            "intel"
        ],
        "MemoryGiBPerVCpu": {
            "Min": 0.0,
            "Max": 0.0
        },
        "ExcludedInstanceTypes": [
            ""
        ],
        "InstanceGenerations": [
            "current"
        ],
        "SpotMaxPricePercentageOverLowestPrice": 0,
        "OnDemandMaxPricePercentageOverLowestPrice": 0,
        "BareMetal": "included",
        "BurstablePerformance": "included",
        "RequireHibernateSupport": true,
        "NetworkInterfaceCount": {
            "Min": 0,
            "Max": 0
        },
        "LocalStorage": "included",
        "LocalStorageTypes": [
            "hdd"
        ],
        "TotalLocalStorageGB": {
            "Min": 0.0,
            "Max": 0.0
        },
        "BaselineEbsBandwidthMbps": {
            "Min": 0,
            "Max": 0
        },
        "AcceleratorTypes": [
            "gpu"
        ],
        "AcceleratorCount": {
            "Min": 0,
            "Max": 0
        },
        "AcceleratorManufacturers": [
            "nvidia"
        ],
```

```
        "AcceleratorNames": [
            "a100"
        ],
        "AcceleratorTotalMemoryMiB": {
            "Min": 0,
            "Max": 0
        },
        "NetworkBandwidthGbps": {
            "Min": 0.0,
            "Max": 0.0
        },
        "AllowedInstanceTypes": [
            ""
        ]
    },
    "MaxResults": 0,
    "NextToken": ""
}
```

2. 前のステップの出力を使用して JSON 設定ファイルを作成し、次のように設定します。

> **Note**
>
> ArchitectureTypes、VirtualizationTypes、VCpuCount、および MemoryMiB の値を指定する必要があります。その他の属性は省略できます。省略すると、デフォルト値が使用されます。
> 各属性およびそのデフォルト値の説明については、「get-instance-types-from-instance-requirements」を参照してください。

a. ArchitectureTypes に、1 つ以上のタイプのプロセッサアーキテクチャを指定します。

b. VirtualizationTypes に、1 つまたは複数のタイプの仮想化を指定します。

c. VCpuCount に、vCPU の最小数と最大数を指定します。最小制限を指定しない場合は、Min で、0 を指定します。最大制限を指定しない場合は、Max パラメータを省略します。

d. MemoryMiB に、最小値と最大値を MiB 単位で指定します。最小制限を指定しない場合は、Min で、0 を指定します。最大制限を指定しない場合は、Max パラメータを省略します。

e. オプションで、他の属性を 1 つ以上指定して、返されるインスタンスタイプのリストをさらに制約できます。

3. JSON ファイルで指定した属性を持つインスタンスタイプをプレビューするには、get-instance-types-from-instance-requirements コマンドを入力し、`--cli-input-json` パラメータを使用して、JSON ファイルの名前とパスを指定します。オプションで、出力が表形式で表示されるようにフォーマットできます。

```
aws ec2 get-instance-types-from-instance-requirements \
    --cli-input-json file://attributes.json \
    --output table
```

例: *attributes.json* ファイル

この例では、JSON ファイルに必須属性が含まれています。それらは、ArchitectureTypes、VirtualizationTypes、VCpuCount、および MemoryMiB です。さらに、オプションで InstanceGenerations 属性も含まれます。MemoryMiB では、Max の値を省略し、制限がないことを示すことができることを注意してください。

```
{

    "ArchitectureTypes": [
        "x86_64"
    ],
    "VirtualizationTypes": [
        "hvm"
    ],
    "InstanceRequirements": {
        "VCpuCount": {
            "Min": 4,
            "Max": 6
        },
        "MemoryMiB": {
            "Min": 2048
        },
        "InstanceGenerations": [
            "current"
        ]
    }
}
```

出力の例

```
-----------------------------------------
|GetInstanceTypesFromInstanceRequirements|
+---------------------------------------+
||             InstanceTypes             ||
|+-------------------------------------+|
||             InstanceType              ||
|+-------------------------------------+|
||  c4.xlarge                          ||
||  c5.xlarge                          ||
||  c5a.xlarge                         ||
||  c5ad.xlarge                        ||
||  c5d.xlarge                         ||
||  c5n.xlarge                         ||
||  d2.xlarge                          ||
...
```

4. ニーズに合ったインスタンスタイプを特定したら、フリートリクエストを設定するときにそれらを使用できるように、使用したインスタンスの属性をメモしておきます。

インスタンスの重み付けを使用して、EC2 フリートまたはスポットフリートのコストとパフォーマンスを管理する

インスタンスの重み付けでは、EC2 フリートまたはスポットフリート内の各インスタンスタイプに重みを割り当てて、それぞれのコンピューティング能力とパフォーマンスを相対的に表します。この重みに基づいて、目的のターゲットキャパシティを満たしている限り、指定されたインスタンスタイプを任意に組み合わせて使用できます。そのため、フリートのコストとパフォーマンスを容易に管理できます。

重みは、インスタンスタイプが合計ターゲットキャパシティに寄与するキャパシティユニットを表します。

例: パフォーマンス管理にインスタンスの重み付けを使用する

例えば、フリートにインスタンスタイプが 2 つあり、以下のようにそれぞれのインスタンスタイプに異なる重みを割り当てているとします。数値は、各タイプが同じパフォーマンスを発揮するためにはそれぞれいくつ必要になるかを表しています。

- `m5.large` – 重み: 1

- m5.2xlarge – 重み: 4

こうした重みを割り当てることで、1 個の m5.large と同じパフォーマンスを発揮するためには 4 個の m5.2xlarge インスタンスが必要であることを示すことができます。

特定のターゲットキャパシティに必要なインスタンスの数をインスタンスタイプごとに計算するには、以下の式を使用します。

```
target capacity / weight = number of instances
```

ターゲットキャパシティが 8 ユニットである場合、フリートは以下のように m5.large または m5.2xlarge、あるいはその両方を混在させて、ターゲットキャパシティを達成できます。

- 8 個の m5.large インスタンス (キャパシティ 8 / 重み 1 = 8 インスタンス)
- 2 個の m5.2xlarge インスタンス (キャパシティ 8 / 重み 4 = 2 インスタンス)
- 4 個の m5.large および 1 個の m5.2xlarge

例: コスト管理にインスタンスの重み付けを使用する

デフォルトでは、指定する料金は 1 インスタンス時間あたり の料金となります。インスタンスの分量指定機能を使用すると、指定した料金は ユニット時間ごと の料金となります。ユニット時間あたりの使用料金はインスタンスタイプの料金を対応するユニット数で割って計算できます。フリートは、ターゲットキャパシティをインスタンスの重みで割ることで、起動するインスタンスの数を求めます。その結果が整数でなければ、フリートはその数を次の整数に切り上げ、これによりフリートのサイズがターゲット容量以上になります。起動されたインスタンスの容量がリクエストされたターゲット容量を超えた場合でも、フリートは起動仕様で指定したどのプールでも選択できます。

以下の表に、ターゲットキャパシティが 10 のフリートについてユニットあたりの料金を求める計算の例を示しています。

インスタンスタイプ	インスタンスの分量	ターゲット容量	起動されたインスタンスの数	インスタンス時間あたりのスポット料金	ユニット時間あたりの価格
r3.xlarge	2	10	5	0.05 USD	0.025 USD

インスタンスタイプ	インスタンスの分量	ターゲット容量	起動されたインスタンスの数	インスタンス時間あたりのスポット料金	ユニット時間あたりの価格
			(10 ÷ 2)		(.05 ÷ 2)
r3.8xlarge	8	10	2 (10 ÷ 8、結果切り上げ)	0.10 USD	0.0125 USD (.10 ÷ 8)

以下のようにフリートインスタンスの重み付けを使用して、達成時にユニットあたりの料金が最も安くなるプールに目的のターゲットキャパシティをプロビジョニングします。

1. インスタンス (デフォルト) または任意のユニット (vCPU、メモリ、ストレージ、スループットなど) でフリートのターゲットキャパシティを設定します。
2. ユニットあたりの料金を設定します。
3. 各起動条件で、インスタンスタイプがターゲット容量に対して必要なユニット数である分量を指定します。

インスタンスの分量指定例

フリートリクエストが以下のように設定されている例を検討します。

- ターゲット容量 24
- r3.2xlarge のインスタンスタイプの起動条件と分量 6
- c3.xlarge のインスタンスタイプの起動条件と分量 5

分量とは、インスタンスタイプがターゲット容量に対して必要なユニット数を表します。最初の起動仕様がユニットあたりの料金を最安値で提供する場合 (インスタンス時間あたりの r3.2xlarge の料金を 6 で割ったもの)、フリートはこれらのインスタンスのうち 4 つ (24 割る 6) を起動します。

2 番目の起動仕様がユニットあたりの料金を最安値で提供する場合 (インスタンス時間あたりの c3.xlarge の料金を 5 で割ったもの)、フリートはこれらのインスタンスのうち 5 つ (24 割る 5、結果を切り上げ) を起動します。

インスタンスの分量指定と配分戦略

フリートリクエストが以下のように設定されている例を検討します。

- ターゲット容量 30 スポットインスタンス
- `c3.2xlarge` のインスタンスタイプの起動条件と分量 8
- `m3.xlarge` のインスタンスタイプの起動条件と分量 8
- `r3.xlarge` のインスタンスタイプの起動条件と分量 8

フリートは、4 つのインスタンス (30 割る 8、結果を切り上げ) を起動します。`diversified` 戦略では、フリートは 3 プールごとに 1 つのインスタンスを起動し、そしてこの 3 つのプールのいずれかから取得された 4 つ目のインスタンスがユニットあたりの最低価格を提供することになります。

配分戦略を使用して、EC2 フリートまたはスポットフリートがどのようにスポットキャパシティとオンデマンドキャパシティを満たすかを決定します。

EC2 フリートまたはスポットフリート内で複数のキャパシティプール (それぞれインスタンスタイプとアベイラビリティーゾーンで構成) を使用する場合は、配分戦略を使用して、そうしたキャパシティプールから Amazon EC2 がどのようにスポットキャパシティとオンデマンドキャパシティを満たすかを管理できます。配分戦略を使用すると、使用可能なキャパシティ、料金、インスタンスタイプを最適化できます。スポットインスタンスとオンデマンドインスタンスには、さまざまな配分戦略があります。

トピック

- スポットインスタンスの配分戦略
- オンデマンドインスタンスの配分戦略
- 適切なスポット配分戦略を選択する
- スポットインスタンスのターゲットキャパシティを維持する
- オンデマンド容量に基づくインスタンスタイプの優先順位付け

スポットインスタンスの配分戦略

起動設定によって、EC2 フリートまたはスポットフリートがスポットインスタンスを起動できるすべてのスポットキャパシティプール (インスタンスタイプとアベイラビリティーゾーン) が決まりま

す。ただし、インスタンスを起動する際、フリートは指定された配分戦略を使用して、使用可能なすべてのプールから特定のプールを選択してください。

> **Note**
>
> (Linux インスタンスのみ) AMD SEV-SNP を有効にして起動するようにスポットインスタンスを設定すると、選択したインスタンスタイプのオンデマンド時間料金の 10% に相当する追加の時間単位使用料が請求されます。配分戦略で料金を入力として使用する場合、フリートにはこの追加料金は含まれず、スポット料金のみが使用されます。

スポットインスタンスには次のいずれかの配分戦略を指定できます。

料金キャパシティ最適化 (推奨)

フリートは、起動中のインスタンスの数に最適なキャパシティを使用できるプールを特定します。つまり、短期的に中断の可能性が最も低いと思われるプールからスポットインスタンスをリクエストすることになります。次に、そうしたプールのうち料金が最も安いスポットインスタンスをリクエストします。

料金キャパシティ最適化配分戦略は、ステートレスコンテナ化アプリケーション、マイクロサービス、ウェブアプリケーション、データおよび分析ジョブ、バッチ処理など、ほとんどのスポットワークロードに最適です。

AWS CLI を使用している場合、パラメータ名は EC2 フリートでは `price-capacity-optimized`、スポットフリートでは `priceCapacityOptimized` となります。

キャパシティ最適化

フリートは、起動中のインスタンスの数に最適なキャパシティを使用できるプールを特定します。つまり、短期的に中断の可能性が最も低いと思われるプールからスポットインスタンスをリクエストすることになります。

スポットインスタンス では、価格は需要と供給の長期的な傾向に基づいて時間の経過とともに緩やかに変動しますが、容量はリアルタイムで変動します。キャパシティ最適化戦略では、リアルタイムのキャパシティデータを調べ、可用性の最も高いプールを予測することで、そのプールからスポットインスタンスを自動的に起動します。この戦略は、作業の再開に関連する中断のコストが高くなる可能性のあるワークロード (長時間の継続的インテグレーション (CI)、画像とメディアのレンダリング、深層学習およびハイパフォーマンスコンピューティング (HPC) など) に対応します。中断の可能性を低くすることにより、キャパシティ最適化戦略ではワークロードの全体的なコストを削減できます。

また、優先順位付きキャパシティ最適化と共に優先パラメータを使用して、インスタンスタイプを優先順位の高い順から低い順に並べることができます。異なるインスタンスタイプに対し同じ優先順位を設定できます。フリートはまずキャパシティを最適化しますが、ベストエフォートに基づいてインスタンスタイプの優先順位を考慮します (例えば、優先順位を考慮しても、フリートの最適なキャパシティをプロビジョニングする能力に大きな影響を与えない場合など)。これは、中断の可能性を最小限に抑える必要があり、特定のインスタンスタイプを優先することが重要なワークロードに適したオプションです。なお、スポットキャパシティのインスタンスタイプの優先順位を設定する場合、オンデマンド配分戦略が優先順位付きに設定されていると、同じ優先順位がオンデマンドインスタンスにも適用されます。スポットフリートの場合、優先順位を使用できるのはフリートが起動テンプレートを使用している場合に限られます。

AWS CLI を使用している場合、パラメータ名は EC2 フリートでは `capacity-optimized` と `capacity-optimized-prioritized`、スポットフリートでは `capacityOptimized` と `capacityOptimizedPrioritized` となります。

分散

スポットインスタンス は、すべてのスポットキャパシティープールに分散されます。AWS CLI を使用している場合、パラメータ名は EC2 フリートでもスポットフリートでも `diversified` となります。

最低料金 (非推奨)

> **Warning**
>
> スポットインスタンスの中断リスクが非常に高いため、最低料金配分戦略はお勧めしません。

スポットインスタンスは、使用可能な容量を持つ最低価格のプールから取得されます。AWS CLI を使用する場合は、これがデフォルトの戦略です。ただし、料金キャパシティ最適化配分戦略を指定して、このデフォルトをオーバーライドすることをお勧めします。

最低料金戦略では、料金が最も安いプールに使用可能なキャパシティがない場合、その次に料金が安く、かつ使用可能なキャパシティがあるプールからスポットインスタンスを取得します。目的のキャパシティを満たす前にプールのキャパシティが不足した場合、フリートはその次に料金が安いプールからキャパシティを引き出して、継続してリクエストを満たします。希望する容量を確実に満たすために、複数のプールからスポットインスタンスを受け取る場合があります。

この戦略では、インスタンスの価格のみが考慮され、容量の可用性は考慮されないため、中断率が高くなる可能性があります。

最低料金配分戦略は、AWS CLI を使用しているときにのみ利用できます。パラメータ名は EC2 フリートでは `lowest-price`、スポットフリートでは `lowestPrice` となります。

使用するプールの数

ターゲットスポット容量を割り当てる先のスポットプールの数。配分戦略が最低料金に設定されている場合にのみ有効です。フリートは、最低料金のスポットプールを選択し、指定した数のスポットプールにターゲットスポットキャパシティを均等に割り当てます。

その際、ベストエフォートに基づいて、指定したプール数からスポットインスタンスを引き出そうとします。ターゲットキャパシティを満たす前にプールのスポットキャパシティが不足した場合は、その次に料金が安いプールのキャパシティを利用して継続的にリクエストを満たします。ターゲット容量を確実に満たすために、スポットインスタンスが、指定した数を超えるプールから割り当てられることがあります。また、ほとんどのプールにスポット容量がない場合には、指定した数より少ないプールからターゲット容量のすべてが割り当てられることがあります。

このパラメータは、最低料金配分戦略を指定し、AWS CLI を使用しているときにのみ利用できます。パラメータ名は、EC2 フリートでもスポットフリートでも `InstancePoolsToUseCount` となります。

オンデマンドインスタンスの配分戦略

起動設定によって、EC2 フリートまたはスポットフリートがオンデマンドインスタンスを起動できるすべてのスポットキャパシティプール (インスタンスタイプとアベイラビリティーゾーン) が決まります。ただし、インスタンスを起動する際、フリートは指定された配分戦略を使用して、使用可能なすべてのプールから特定のプールを選択してください。

オンデマンドインスタンスには、以下のいずれかの配分戦略を指定できます。

最低料金

オンデマンドインスタンスは、使用可能なキャパシティがある最低料金のプールから取得されます。これはデフォルトの戦略です。

最低料金のプールに使用可能なキャパシティがない場合、その次に料金が安く、かつ使用可能なキャパシティがあるプールからオンデマンドインスタンスが取得されます。

目的のキャパシティを満たす前にプールのキャパシティが不足した場合、フリートはその次に料金が安いプールからキャパシティを引き出して、継続してリクエストを満たします。目的のキャパシティを確実に満たすために、複数のプールからスポットインスタンスが取得される場合があります。

優先順位付き

フリートは、各起動テンプレートのオーバーライドに割り当てられた優先順位に従って、まず優先順位が最も高いインスタンスタイプを起動します。この戦略は、属性ベースのインスタンスタイプの選択では使用できません。この配分戦略の使用方法の例については、「オンデマンド容量に基づくインスタンスタイプの優先順位付け」を参照してください。

適切なスポット配分戦略を選択する

適切なスポット割り当て戦略を選択することで、ユースケースに合わせてフリートを最適化できます。

最低価格と容量可用性のバランスをとる

料金が最も安いスポットキャパシティプールと、キャパシティの可用性が最も高いスポットキャパシティプールとのトレードオフのバランスをとるには、料金キャパシティ最適化配分戦略を使用することをお勧めします。この戦略では、プールの価格とプール内のスポットインスタンスの空き容量の両方に基づいて、どのプールからスポットインスタンスをリクエストするかを決定します。つまり、価格を考慮しながらも短期的に中断の可能性が最も低いと思われるプールからスポットインスタンスをリクエストすることになります。

コンテナ化されたアプリケーション、マイクロサービス、ウェブアプリケーション、データおよび分析ジョブ、バッチ処理など、レジリエントでステートレスなワークロードをフリートが実行している場合は、最適なコスト削減とキャパシティの可用性を実現する料金キャパシティ最適化配分戦略を使用してください。

作業の再開に関連する中断に伴うコストが高くなる可能性があるワークロードをフリートで実行している場合は、中断があった場合にアプリケーションがそのポイントから再起動できるようにチェックポイントの設定を実装する必要があります。チェックポイントを使用すると、スポットインスタンスの中断率も低い最低料金のプールからキャパシティが割り当てられるため、こうしたワークロードには料金キャパシティ最適化配分戦略が適しています。

料金キャパシティ最適化配分戦略を使用する JSON 設定の例については、以下を参照してください。

- EC2 フリート - 例 10: price-capaity-optimized フリートでスポットインスタンスを起動する
- スポットフリート - 例 11: priceCapacityOptimized フリートでスポットインスタンスを起動する

ワークロードの中断コストが高い場合

同様の料金のインスタンスタイプを使用するワークロードを実行する場合や、中断のコストが非常に高いために中断のわずかな増加に比べてコスト削減が不十分である場合は、必要に応じてこのキャパシティ最適化戦略を使用できます。この戦略では、中断の可能性がより低く、最も可用性の高いスポットキャパシティプールから容量を割り当てることで、ワークロードの総コストを削減することができます。

中断の可能性を最小限に抑える必要があるものの、特定のインスタンスタイプの優先順位が重要である場合は、優先順位付きキャパシティ最適化配分戦略を使用し、インスタンスタイプを優先順位の高い順に並べることでプールの優先順位を設定できます。

なお、優先順位付きキャパシティ最適化の優先順位を設定するとき、オンデマンド配分戦略が優先順位付きに設定されていると、同じ優先順位がオンデマンドインスタンスにも適用されます。また、スポットフリートの場合、優先順位を使用できるのはフリートが起動テンプレートを使用している場合に限られます。

キャパシティ最適化配分戦略を使用する JSON 設定の例については、以下を参照してください。

- EC2 フリート - 例 8: 容量最適化フリートでスポットインスタンスを起動する
- スポットフリート - 例 9: 容量最適化フリートでスポットインスタンスを起動する

優先順位付きキャパシティ最適化配分戦略を使用する JSON 設定の例については、以下を参照してください。

- EC2 フリート - 例 9: 優先順位のある容量最適化フリートでスポットインスタンスを起動する
- スポットフリート - 例 10: 優先順位のある容量最適化フリートでスポットインスタンスを起動する

ワークロードに時間的な柔軟性があり、キャパシティの可用性が問題にならない場合

フリートが小さい場合や実行時間が短い場合は、キャパシティの可用性を考慮しつつ、料金キャパシティ最適化を使用してコスト削減を最大化できます。

フリートが大きい場合や長時間稼働している場合

フリートが大きい場合や実行時間が長い場合は、分散戦略を使用して複数のプールにスポットインスタンスを分散させることで、フリートの可用性を改善できます。例えば、プール数を 10、ターゲットキャパシティを 100 インスタンスに指定した場合、フリートはプールごとに 10 個のスポットイン

スタンスを起動します。1 つのプールのスポット料金がこのプールの上限料金を超える場合、フリートの 10% のみに影響がおよびます。この戦略を使用すると、いずれのプールにおいても経時的にフリートが受けるスポット料金の上昇の影響を減少させます。分散戦略では、スポットフリートはスポット料金がオンデマンド料金以上のプールにあるスポットインスタンスを起動しません。

スポットインスタンスのターゲットキャパシティを維持する

スポット料金やスポットキャパシティプールで使用可能なキャパシティが変動したためにスポットインスタンスが終了すると、タイプ `maintain` のフリートが代替スポットインスタンスを起動します。配分戦略によって、次のように置換先インスタンスを起動するプールが決まります。

- 配分戦略が料金キャパシティ最適化の場合、フリートはスポットインスタンスキャパシティの可用性が最も高いプールで代替インスタンスを起動します。同時に料金も考慮し、料金が安くかつキャパシティの可用性が高いプールを特定します。
- 配分戦略がキャパシティ最適化の場合、フリートはスポットインスタンスキャパシティの可用性が最も高いプールで代替インスタンスを起動します。
- 配分戦略が分散の場合、フリートは残っているプール全体に代替スポットインスタンスを分散させます。

オンデマンド容量に基づくインスタンスタイプの優先順位付け

EC2 フリートまたはスポットフリートがオンデマンドキャパシティを満たそうとする場合、デフォルトではまず料金が最も安いインスタンスタイプを起動します。オンデマンド配分戦略が優先順位付きの場合、フリートは優先順位に従って、オンデマンドキャパシティを満たす際に最初にどのインスタンスタイプを使用するかを決定します。優先度は起動テンプレートの上書きに割り当てられ、最も高い優先度が最初に起動されます。

例: インスタンスタイプの優先付け

例えば、3 つの起動テンプレートの上書きに、それぞれ異なるインスタンスタイプを設定したとします。

インスタンスタイプのオンデマンド料金は、幅があります。以下は、この例で使用しているインスタンスタイプで、料金の安いものから順に並んでいます。

- `m4.large` — 最も安価
- `m5.large`
- `m5a.large`

優先度を使って順番を決めない場合、フリートは、最も価格が低いインスタンスタイプから始めてオンデマンドの容量を満たします。

ただし、最初に使用する m5.large リザーブドインスタンスが未使用である場合、次のように、インスタンスタイプが優先度順に使われるように、起動テンプレートの、上書きの優先度を設定できます。

- m5.large – 優先度 1
- m4.large – 優先度 2
- m5a.large – 優先度 3

EC2 フリートとスポットフリートでキャパシティの再調整を使用して、リスクのあるスポットインスタンスを置き換える

キャパシティの再調整を使用すると、EC2 フリートまたはスポットフリートは中断のリスクがあるスポットインスタンスをプロアクティブに置き換えることで、目的のスポットキャパシティを維持できます。スポットインスタンスで中断のリスクが高まった場合、Amazon EC2 は再調整推奨通知を送信します。キャパシティの再調整が有効になっている場合、リスクのあるインスタンスが中断する前に、再調整推奨通知が新しいスポットインスタンスの起動をトリガーします。

キャパシティの再調整は、実行中のインスタンスが Amazon EC2 によって中断される前に、新しいスポットインスタンスでフリートをプロアクティブに強化することで、ワークロードの可用性を維持するのに効果的です。

キャパシティの再調整を使用して代替スポットインスタンスを起動するように EC2 フリートを設定するには

create-fleet コマンドMaintenanceStrategies と構造内の関連するパラメータを使用します。JSON 設定の例については、「例 7: 容量の再調整を設定して代替スポットインスタンスを起動する」を参照してください。

キャパシティの再調整を使用して代替スポットインスタンスを起動するようにスポットフリートを設定するには

Amazon EC2 コンソールまたは AWS CLI を使用して、キャパシティの再調整を設定できます。

(コンソール) スポットフリートを作成するとき、[キャパシティの再調整] チェックボックスをオンにします。詳細については、「定義済みパラメータを使用してスポットフリートリクエストを作成する」のステップ 6。d を参照してください。

(AWS CLI) request-spot-fleet コマンドと SpotMaintenanceStrategies 構造内の関連するパラメータを使用します。JSON 設定の例については、「例 8: 容量の再調整を設定して代替スポットインスタンスを起動する」を参照してください。

トピック

- 制限事項
- 設定オプション
- 考慮事項

制限事項

- 容量の再調整は、タイプ maintain のフリートでのみ使用可能です。
- フリートが実行されているときは、容量の再調整設定を変更できません。容量の再調整設定を変更するには、フリートを削除し、新しいフリートを作成する必要があります。

設定オプション

EC2 フリートとスポットフリートの ReplacementStrategy では、次の 2 つの値がサポートされています。

`launch-before-terminate`

Amazon EC2 フリートは、新しい置換先スポットインスタンスが起動された後に、再調整通知を受信するスポットインスタンスを終了します。launch-before-terminate を指定する場合は、termination-delay の値も指定する必要があります。新しい置換先インスタンスが起動された後、Amazon EC2 フリートは termination-delay の時間だけ待って、古いインスタンスを終了させます。termination-delay では、最小値は 120 秒 (2 分)、最大値は 7200 秒 (2 時間) です。

launch-before-terminate は、インスタンスのシャットダウン手順が完了するまでの時間が予測できる場合にのみ使用することをお勧めします。これにより、シャットダウン手順が完了した後にのみ、古いインスタンスが確実に終了されます。Amazon EC2 は、termination-delay の前に 2 分間の警告を行い、その後古いインスタンスを中断する可能性があることに注意してください。

また、lowest-price (EC2 フリート) または lowestPrice (スポットフリート) 配分戦略を launch-before-terminate と組み合わせて使用することは、代替スポットインスタンスが中断するリスクが高まるため、お勧めしません。

launch

Amazon EC2 フリートは、既存のスポットインスタンスに対して再調整通知が送信されると、置換先スポットインスタンスを起動します。Amazon EC2 フリートは、再調整通知を受け取るインスタンスを終了しません。古いインスタンスを終了することも、実行したままにすることもできます。実行中は、すべてのインスタンスに対して課金されます。

考慮事項

キャパシティの再調整用に EC2 フリートまたはスポットフリートを設定する場合は、次の点を考慮してください。

リクエストでは可能な限り多くのスポットキャパシティープールを指定する

複数のインスタンスタイプとアベイラビリティーゾーンを使用するようにフリートを設定します。これにより、さまざまなスポットキャパシティープールでスポットインスタンスを起動するための柔軟性が得られます。詳細については、「インスタンスタイプとアベイラビリティーゾーンについて柔軟に対応する」を参照してください。

代替スポットインスタンスが中断されるリスクの増大を回避する

中断のリスクが高まるのを避けるため、capacity-optimized または capacity-optimized-prioritized の割り当て戦略をお勧めします。これらの戦略により、代替スポットインスタンスが最適なスポットキャパシティプールで起動されるため、近い将来中断される可能性が低くなります。詳細については、「価格と容量を最適化する配分戦略を使用する」を参照してください。

lowest-price 配分戦略を使用すると、代替スポットインスタンスが中断されるリスクが高まることがあります。これは、置換先スポットインスタンスが起動後すぐに中断される可能性があっても、Amazon EC2 は、その時点で利用可能な容量を持つ最低価格のプールでインスタンスを常に起動するためです。

Amazon EC2 は、可用性が同じかそれ以上の場合にのみ、新しいインスタンスを起動します

容量の再調整の目的の 1 つは、スポットインスタンスの可用性を改善することです。既存のスポットインスタンスが再調整のレコメンデーションを受け取った場合、Amazon EC2 は、新しいインスタンスが既存のインスタンスと同等かそれ以上の可用性を提供する場合にのみ新しいイン

スタンスを起動します。新しいインスタンスの中断のリスクが既存のインスタンスよりもひどい場合、Amazon EC2 は新しいインスタンスを起動しません。ただし、Amazon EC2 は引き続きスポットキャパシティプールを評価し、可用性が向上したら新しいインスタンスを起動します。

Amazon EC2 が新しいインスタンスをプロアクティブに起動しないと、既存のインスタンスが中断する可能性があります。これが発生する場合、Amazon EC2 は、新しいインスタンスの中断リスクが高いかどうかに関らず、新しいインスタンスの起動を試みます。

キャパシティーの再調整は、スポットインスタンスの中断率を増加させるものではありません

キャパシティの再調整を有効にしても、スポットインスタンスの中断率 (Amazon EC2 がキャパシティを取り戻す必要があるときに再利用されるスポットインスタンスの数) は増加しません。ただし、インスタンスに中断のリスクがあることを容量の再調整が検出した場合、Amazon EC2 Auto Scaling は直ちに新しいインスタンスの起動を試みます。その結果、リスクのあるインスタンスが中断された後に Amazon EC2 が新しいインスタンスを起動するのを待つ場合よりも多くのインスタンスが置き換えられる可能性があります。

キャパシティーの再調整が有効になっているインスタンスをさらに置き換える可能性がありますが、インスタンスが中断される前にアクションを実行するための時間をより長く確保できるため、事後対応ではなくプロアクティブに対応できるというメリットがあります。スポットインスタンスの中断通知では、通常、インスタンスを正常にシャットダウンするための猶予期間が最大 2 分しかありません。キャパシティーの再調整で新しいインスタンスを事前に起動することで、既存のプロセスがリスクのあるインスタンスで完了する可能性が高くなり、インスタンスのシャットダウン手順を開始して、リスクのあるインスタンスで新しい作業がスケジュールされないようにできます。新しく起動したインスタンスの準備を開始して、アプリケーションを引き継ぐこともできます。キャパシティーの再調整のプロアクティブな置き換えにより、正常な継続性の恩恵を受けることができます。

キャパシティーの再調整を使用するリスクとメリットを示す理論的な例として、次のシナリオを検討してください。

- 午後 2 時 – インスタンス A の再調整の推奨が受信され、Amazon EC2 は直ちに置換先インスタンス B の起動の試行を開始するため、シャットダウン手順を開始する時間を確保できます。*
- 午後 2 時 30 分 – インスタンス B の再調整の推奨が受信され、インスタンス C に置き換えられるため、シャットダウン手順を開始する時間を確保できます。*
- 午後 2 時 32 分 – キャパシティーの再調整が有効になっておらず、インスタンス A のスポットインスタンスの中断通知が午後 2 時 32 分に受信されていたとすれば、アクションを実行するための猶予期間は最大でも 2 分だけでしたが、インスタンス A はこの時間まで稼働していたことでしょう。

* `launch-before-terminate` が指定されている場合、Amazon EC2 は、置換先インスタンスがオンラインになった後、リスクのあるインスタンスを終了します。

Amazon EC2 フリート は、満たされた容量がターゲット容量の 2 倍になるまで、新しい置換先スポットインスタンスを起動できます

キャパシティの再調整に対応するように設定した場合、フリートはスポットインスタンスが再調整推奨通知を受け取るたびに、新しい代替スポットインスタンスを起動しようとします。スポットインスタンスが再調整勧告を受け取った後は、満たされた容量の一部としてカウントされなくなります。交換戦略に応じて、Amazon EC2 は事前設定された終了遅延の後にインスタンスを終了するか、インスタンスを実行のままにします。これにより、インスタンスで 再調整アクション を実行できるようになります。

フリートがターゲットキャパシティの 2 倍に達すると、代替インスタンス自体が再調整に関する推奨事項を受け取った場合でも、新しい代替インスタンスの起動を停止します。

例えば、100 個のスポットインスタンスをターゲットキャパシティとするスポットフリートを作成するとします。すべてのスポットインスタンスは、再調整に関するレコメンデーションを受け取ります。これにより、Amazon EC2 は 100 個の置換先スポットインスタンスを起動します。これにより、満たされたスポットインスタンスの数が 200 になり、ターゲットキャパシティの 2 倍になります。一部の代替インスタンスが再調整推奨通知を受け取るものの、フリートがターゲットキャパシティの 2 倍を超えることができないため、代替インスタンスはそれ以上起動されません。

インスタンスの実行中は、すべてのインスタンスに対して課金されることに注意してください。

再調整推奨通知を受け取ったスポットインスタンスを終了するようにフリートを設定することを推奨する

キャパシティの再調整に対応するようにフリートを設定する場合は、インスタンスのシャットダウン手順を完了するまでの時間が予測できる場合に限り、`launch-before-terminate` を選択して適切な終了遅延を設定することをお勧めします。これにより、シャットダウン手順が完了した後にのみ、古いインスタンスが確実に終了されます。

再調整のために推奨されるインスタンスを終了する場合は、フリートのスポットインスタンスが受信する再調整レコメンデーションシグナルをモニタリングすることをお勧めします。シグナルをモニタリングすることで、Amazon EC2 が中断する前に、影響を受けるインスタンスで 再調整のアクション をすばやく実行し、手動で終了できます。インスタンスを終了しない場合、インスタンスの実行中、課金が継続します。Amazon EC2 は、再調整に関する推奨を受け取るインスタンスを自動的に終了しません。

Amazon EventBridge またはインスタンスメタデータを使用して通知を設定できます。詳細については、「再調整に関する推奨事項シグナルのモニタリング」を参照してください。

スケールインまたはスケールアウト中に満たされたキャパシティを計算する際、フリートが再調整推奨通知を受け取るインスタンスをカウントしない

キャパシティの再調整に対応するようにフリートが設定されている場合に、ターゲットキャパシティをスケールインまたはスケールアウトするように変更すると、次のように、フリートは再調整の対象としてマークされたインスタンスを、満たされたキャパシティの一部としてカウントしません。

- スケールイン – 希望するターゲット容量を減らすと、Amazon EC2 は目的の容量に達するまで、再調整の対象としてマークされていないインスタンスを終了します。再調整の対象としてマークされたインスタンスは、満たされた容量にはカウントされません。

  例えば、100 個のスポットインスタンスをターゲットキャパシティとするフリートを作成し、そのうち 10 個のインスタンスが再調整推奨通知を受け取ったとすると、Amazon EC2 が新たに 10 個の代替インスタンスを起動するため、110 個のインスタンスのキャパシティが満たされます。その後、ターゲット容量を 50 個に減らしますが (スケールイン)、再調整の対象としてマークされた 10 個のインスタンスは Amazon EC2 によって終了されないため、満たされた容量は実際には 60 インスタンスになります。このようなインスタンスは手動で終了する必要があります。または、実行したままにしておくことができます。

- スケールアウト – 目的のターゲット容量を増やすと、目的の容量に達するまで Amazon EC2 は新しいインスタンスを起動します。再調整の対象としてマークされたインスタンスは、満たされた容量にはカウントされません。

  例えば、100 個のスポットインスタンスをターゲットキャパシティとするフリートを作成し、そのうち 10 個のインスタンスが再調整推奨通知を受け取ったとすると、フリートが新たに 10 個の代替インスタンスを起動するため、110 個のインスタンスのキャパシティが満たされます。その後、ターゲット容量を 200 個に増やし (スケールアウトし) ますが、再調整の対象としてマークされた 10 個のインスタンスは、フリートによってターゲット容量の一部としてカウントされないため、実際には 210 個のインスタンスになります。このようなインスタンスは手動で終了する必要があります。または、実行したままにしておくことができます。

キャパシティ予約を使用して、EC2 フリートのオンデマンドキャパシティを予約する

オンデマンド容量予約を使用すると、特定のアベイラビリティーゾーンで任意の所要時間だけ、オンデマンドインスタンスのコンピューティング性能を予約できます。オンデマンドインスタンスを起動するときに、容量予約を最初に使用するように EC2 フリートを設定できます。

オンデマンドキャパシティ予約は、リクエストタイプが `instant` に設定された EC2 フリートでのみ使用できます。

容量予約は、`open` または `targeted` のいずれかで設定されます。EC2 フリートは、オンデマンドインスタンスを `open` または `targeted` キャパシティ予約で設定できます (次のとおり) :

- キャパシティ予約が `open` の場合、一致する属性を持つオンデマンドインスタンスは、リザーブドキャパシティで自動的に実行されます。
- キャパシティ予約 が `targeted` の場合、オンデマンドインスタンスはそれがリザーブドキャパシティで実行されるように具体的に設定する必要があります。これは、特定のキャパシティ予約を使い切ったり、特定のキャパシティ予約をいつ使用するかを制御する場合に便利です。

また、`targeted` を使用した場合、EC2 フリートのキャパシティ予約では、ターゲットのオンデマンドキャパシティを満たすのに十分なキャパシティ予約が必要です。そうしないと、起動に失敗します。起動が失敗するのを避けるには、`targeted` キャパシティ予約をリソースグループに加え、リソースグループをターゲットにします。リソースグループは十分なキャパシティ予約を持っている必要はありません。ターゲットのオンデマンドキャパシティが満たされる前にキャパシティ予約がなくなった場合、フリートは残りのターゲットキャパシティを通常のオンデマンドキャパシティに起動できます。

EC2 フリートでキャパシティ予約を使用するには

1. フリートをタイプ `instant` として設定する: その他のタイプのフリートには、キャパシティーの予約を使用することはできません。
2. キャパシティ予約の使用戦略として、`use-capacity-reservations-first` を設定する。
3. 起動テンプレートで、キャパシティ予約 には、オープン または グループ別のターゲット のいずれかを選択してください。グループ別のターゲット を選択した場合、キャパシティ予約リソースグループ ID を指定します。

フリートがオンデマンド容量を満たそうとしたときに、複数のインスタンスプールで一致するキャパシティ予約が未使用であることがわかった場合、オンデマンド割り当て戦略に基づいてオンデマンドインスタンスを起動するプールを決定します (`lowest-price` または `prioritized`)。

関連リソース

- オンデマンドキャパシティを満たすために、キャパシティ予約を使用するようにフリートを設定する CLI の例については、「EC2 フリートの CLI 設定の例」(特に例 5 から 7) を参照してください。
- キャパシティ予約の作成、フリートでの使用、および残っているキャパシティ予約の数を確認する手順を説明するチュートリアルについては、「チュートリアル: ターゲットのキャパシティ予約を使用してオンデマンドインスタンスを起動するように EC2 フリートを設定する」を参照してください。
- キャパシティ予約の設定の詳細については、EC2 オンデマンドキャパシティ予約を使用してコンピューティングキャパシティを予約 と オンデマンドキャパシティ予約のよくある質問 を参照してください。

EC2 フリートの操作

EC2 フリートの使用を開始するには、リクエストを作成します。リクエストには、合計ターゲットキャパシティ、オンデマンドキャパシティ、スポットキャパシティのほか、フリート内のインスタンスの設定を記載した起動テンプレートを含めます。必要に応じて、さらに別のパラメータを指定したり、デフォルト値を使用するようにフリートを設定したりできます。また、フリートを作成するときに、フリートリクエスト、およびフリート内のインスタンスとボリュームにタグ付けすることもできます。

フリートは、使用可能なキャパシティがあるときはオンデマンドインスタンスを起動し、上限価格がスポット料金より大きく、キャパシティが利用可能なときはスポットインスタンスを起動します。

フリートを起動したら、フリートリクエスト、フリート内のインスタンス、およびフリートイベントの説明を記述できます。また、必要に応じてさらに別のタグを割り当てることもできます。

合計ターゲットキャパシティなどのフリートパラメータを変更する必要がある場合は、キャパシティを維持するように設定されているのであれば、フリートを変更できます。送信後にワンタイムリクエストのキャパシティを変更することはできません。

フリートリクエストは、期限切れになるか、お客様によって削除されるまで、アクティブのままになります。フリートリクエストを削除する場合、インスタンスを終了するか、実行したままにします。

実行したままにする場合、オンデマンドインスタンスは、終了されるまで実行され、スポットインスタンスは中断されるか終了されるまで実行されます。

トピック

- EC2 フリートリクエストの状態
- EC2 フリートの前提条件
- EC2 フリートの作成
- 新規または既存の EC2 フリートリクエスト、およびそのフリートが起動するインスタンスとボリュームにタグ付けします。
- EC2 フリート、そのインスタンス、イベントを記述する
- EC2 フリート の変更
- EC2 フリートリクエストおよびフリート内のインスタンスを削除する

EC2 フリートリクエストの状態

EC2 フリートリクエストは、いろいろある状態のいずれかになります。その状態ごとにリクエストライフサイクルのステージが異なり、状態を見ればインスタンスの管理状況がわかります。

EC2 フリート リクエストは、次に示す状態のいずれかになります。

`submitted`

EC2 フリート リクエストは評価中です。Amazon EC2 は目標数のインスタンスを起動する準備をしています。フリートの上限を超えたリクエストは、即時削除されます。

`active`

EC2 フリート リクエストは検証済みです。Amazon EC2 は実行中のインスタンスをターゲット数分、確保しようとしています。リクエストは、変更または削除されるまで、この状態のままになります。

`modifying`

EC2 フリート リクエストは変更中です。リクエストは、変更が完全に処理されるか、 リクエストが削除されるまで、この状態のままになります。`maintain` フリートタイプのみを変更できます。この状態はその他のリクエストタイプには適用されません。

`deleted_running`

EC2 フリートリクエストが削除され、追加のスポットインスタンスは起動されません。その既存のインスタンスは、手動で中断または終了されるまで実行され続けます。リクエストは、すべてのインスタンスが中断されるか終了されるまで、この状態のままになります。EC2 フリートリクエストが削除された後、タイプ `maintain` または `request` の EC2 フリート のみがインスタンスを実行できます。実行中のインスタンスを持つ削除した `instant` フリートはサポートされていません。この状態は `instant` フリートには適用されません。

`deleted_terminating`

EC2 フリートリクエストが削除され、そのインスタンスが終了します。リクエストは、すべてのインスタンスが終了されるまで、この状態のままになります。

`deleted`

EC2 フリート リクエストが削除され、実行中のインスタンスがありません。リクエストは、そのインスタンスが終了されてから 2 日後に削除されます。

EC2 フリートの前提条件

EC2 フリートを作成するには、以下の前提条件を設定する必要があります。

- 起動テンプレート
- EC2 フリート用のサービスにリンクされたロール
- 暗号化された AMI および EBS スナップショット用のカスタマー管理された型キーへのアクセス権限の付与
- EC2 フリートユーザーのアクセス許可

起動テンプレート

起動テンプレートには、インスタンスタイプやアベイラビリティーゾーンなど、起動するインスタンスに関する設定情報を指定します。起動テンプレートの詳細については、「Amazon EC2 起動テンプレートにインスタンス起動パラメータを保存する」を参照してください。

EC2 フリート用のサービスにリンクされたロール

`AWSServiceRoleForEC2Fleet` ロールは、インスタンスのリクエスト、起動、終了、タグ付けを行う許可を EC2 フリートに付与します。Amazon EC2 は、このサービスにリンクされたロールを使用して、以下のアクションを完了します。

- `ec2:RunInstances` – インスタンスを起動します。
- `ec2:RequestSpotInstances` – スポットインスタンスをリクエストします。
- `ec2:TerminateInstances` – インスタンスを終了します。
- `ec2:DescribeImages` - インスタンスの Amazon マシンイメージ (AMI) を表示します。
- `ec2:DescribeInstanceStatus` - インスタンスのステータスを表示します。
- `ec2:DescribeSubnets` - インスタンスのサブネットを表示します。
- `ec2:CreateTags` – EC2 フリート、インスタンス、ボリュームにタグを追加します。

AWS CLI または API を使用して EC2 フリート を作成する前に、このロールが存在していることを確認します。

> **Note**
>
> `instant` EC2 フリート に、このロールは必要ありません。

ロールを作成するには、IAM コンソールを次のように使用します。

EC2 フリート の AWSServiceRoleForEC2Fleet ロールを作成するには

1. IAM コンソールの https://console.aws.amazon.com/iam/ を開いてください。
2. ナビゲーションペインで Roles (ロール) を選択してください。
3. [Create role] を選択してください。
4. [信頼されたエンティティを選択] ページで、以下の操作を実行します。
   a. [信頼できるエンティティタイプ] で、[AWS サービス] を選択してください。
   b. [ユースケース] の [サービスまたはユースケース] で、[EC2 - フリート] を選択してください。

      > **Tip**
      >
      > 必ず [EC2 - フリート] を選択してください。[EC2] を選択した場合、[EC2 - フリート] ユースケースは [ユースケース] リストに表示されません。[EC2 - フリート] ユースケースでは、必要な IAM アクセス許可を持つポリシーが自動的に作成され、ロール名として AWSServiceRoleForEC2Fleet が提案されます。

c. [Next] を選択してください。

5. [アクセス許可を追加] ページで [次へ] を選択してください。

6. [名前、確認、および作成] ページで、[ロールの作成] をクリックしてください。

EC2 フリート を使用する必要がなくなった場合は、AWSServiceRoleForEC2Fleet ロールを削除することをお勧めします。このロールがアカウントから削除された後で、別のフリートを作成した場合はロールを再度作成できます。

詳細については「IAM ユーザーガイド」の「Service-linked roles」を参照してください。

暗号化された AMI および EBS スナップショット用のカスタマー管理された型キーへのアクセス権限の付与

暗号化された AMI または暗号化された Amazon EBS スナップショットを EC2 フリートで指定し、暗号化の AWS KMS キーを使用する場合は、カスタマー管理された型キーを使用して、Amazon EC2 がユーザーの代わりにインスタンスを起動する許可を、AWSServiceRoleForEC2Fleet ロールに付与する必要があります。これを行うには、次の手順で示すように、カスタマー管理された型キーに許可を追加する必要があります。

アクセス権限を設定するときは付与がキーポリシーの代わりになります。詳細については、「AWS Key Management Service デベロッパーガイド」で「許可の使用」と「AWS KMS でのキーポリシーの使用」を参照してください。

AWSServiceRoleForEC2Fleet ロールにカスタマー管理された型キーを使用する許可を付与するには

- 許可の作成 コマンドを使用して、カスタマー管理された型キーに許可を付与し、オペレーションを実行する許可を追加するプリンシパル (AWSServiceRoleForEC2Fleet サービスにリンクされたロール) を指定します。カスタマー管理された型キーは、`key-id` パラメーターとカスタマー管理された型キーの ARN を指定されます。プリンシパルを指定するには、`grantee-principal` パラメータと AWSServiceRoleForEC2Fleet サービスにリンクされたロールの ARN を使用します。

```
aws kms create-grant \
    --region us-east-1 \
    --key-id arn:aws:kms:us-
east-1:444455556666:key/1234abcd-12ab-34cd-56ef-1234567890ab \
    --grantee-principal arn:aws:iam::111122223333:role/AWSServiceRoleForEC2Fleet \
```

```
    --operations "Decrypt" "Encrypt" "GenerateDataKey"
"GenerateDataKeyWithoutPlaintext" "CreateGrant" "DescribeKey" "ReEncryptFrom"
"ReEncryptTo"
```

EC2 フリートユーザーのアクセス許可

ユーザーが EC2 フリートを作成または管理する場合、必ず必要な許可を付与してください。

EC2 フリートのポリシーを作成するには

1. IAM コンソールの https://console.aws.amazon.com/iam/ を開いてください。
2. ナビゲーションペインで、ポリシー を選択してください。
3. [Create policy] (ポリシーの作成) を選択してください。
4. [Create policy] (ポリシーの作成) ページで、JSON タブを選択し、テキストを以下に置き換えて [Review policy] (ポリシーの確認) を選択してください。

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
              "iam:ListRoles",
              "iam:PassRole",
              "iam:ListInstanceProfiles"
            ],
            "Resource":"arn:aws:iam::123456789012:role/DevTeam*"
        }
    ]
}
```

`ec2:*` は、ユーザーにすべての Amazon EC2 API アクションを呼び出す許可を付与します。特定の Amazon EC2 API アクションに制限するには、代わりにこれらのアクションを指定します。

IAM ユーザーは、既存の IAM ロールを列挙する `iam:ListInstanceProfiles` アクション、EC2 フリートロールを指定する `iam:PassRole` アクション、および既存のインスタンスプロファイルを列挙する `iam:ListRoles` アクションを呼び出すには、許可が必要です。

(オプション) ユーザーが IAM コンソールを使用してロールまたはインスタンスプロファイルを作成できるようにするには、次のアクションをポリシーに追加する必要もあります。

- `iam:AddRoleToInstanceProfile`
- `iam:AttachRolePolicy`
- `iam:CreateInstanceProfile`
- `iam:CreateRole`
- `iam:GetRole`
- `iam:ListPolicies`

5. [Review policy] (ポリシーの確認) ページでポリシー名と説明を入力し、[Create policy] (ポリシーの作成) を選択してください。
6. アクセス許可を付与するにはユーザー、グループ、またはロールにアクセス許可を追加します。

   - AWS IAM アイデンティティセンター のユーザーとグループ:

     アクセス許可セットを作成します。「AWS IAM アイデンティティセンター ユーザーガイド」の「アクセス許可セットを作成する」の手順に従ってください。

   - IAM 内で、ID プロバイダーによって管理されているユーザー:

     ID フェデレーションのロールを作成します。詳細については IAM ユーザーガイド の サードパーティー ID プロバイダー (フェデレーション) 用のロールを作成する を参照してください。

   - IAM ユーザー:

     - ユーザーが担当できるロールを作成します。手順については IAM ユーザーガイド の IAM ユーザーのロールの作成 を参照してください。

- (お奨めできない方法) ポリシーをユーザーに直接アタッチするか、ユーザーをユーザーグループに追加します。「IAM ユーザーガイド」の「ユーザー (コンソール) へのアクセス許可の追加」の指示に従います。

EC2 フリートの作成

EC2 フリートを作成するには、JSON ファイルにフリート設定を定義し、 create-fleet マンドでそのファイルを参照してください。JSON ファイルでは、フリートの合計ターゲットキャパシティ、スポットインスタンスとオンデマンドインスタンスの個別のターゲットキャパシティ、および起動テンプレートを指定する必要があります。起動テンプレートにはフリート内のインスタンスの設定、例えば AMI、インスタンスタイプ、サブネットやアベイラビリティーゾーン、1 つ以上のセキュリティグループなどを定義します。必要に応じて、さらに別の設定を指定できます。例えば、起動テンプレート設定をオーバーライドするためのパラメータ、EC2 キャパシティプールからスポットインスタンスとオンデマンドインスタンスを選択するための配分戦略、フリートに対する最大支出金額などです。詳細については「EC2 フリートまたはスポットフリートの設定オプション」を参照してください。

EC2 フリートは、キャパシティが使用可能であるときはオンデマンドインスタンスを起動し、最大料金がスポット料金を超えていてキャパシティが使用可能であるときはスポットインスタンスを起動します。

フリートにスポットインスタンスが含まれ、タイプが `maintain` である場合、Amazon EC2 はスポットインスタンスが中断したときにフリートのターゲットキャパシティを維持しようとします。

EC2 フリート の制限事項

以下の制限が EC2 フリート に適用されます。

- EC2 フリートは、Amazon EC2 API、AWS CLI、AWS SDK、および CloudFormation からのみ作成できます。
- EC2 フリート リクエストは、AWS リージョンにまたがることはできません。リージョンごとに別個の EC2 フリート を作成する必要があります。
- EC2 フリート リクエストは、同じアベイラビリティーゾーンから複数の異なるサブネットにまたがることはできません。

EC2 フリートの作成

EC2 フリートを使用して複数のインスタンスからなるフリートを起動する場合、必要な手順はフリートリクエストに以下のパラメータを指定することだけです。その他のパラメータには、デフォルト値が使用されます。

- `LaunchTemplateId` または `LaunchTemplateName` — 使用する起動テンプレートを指定します (インスタンスタイプやアベイラビリティーゾーンなど、起動するインスタンスのパラメータを指定します)。
- `TotalTargetCapacity` — フリートの合計ターゲット容量を指定します。
- `DefaultTargetCapacityType` — デフォルトの購入オプションをオンデマンドにするかスポットにするかを指定します。

起動テンプレートに指定されているパラメータをオーバーライドするには、1 つ以上のオーバーライドを指定します。各オーバーライドはインスタンスタイプ、アベイラビリティーゾーン、サブネット、最大料金によって異なり、オーバーライドごとに加重キャパシティを変えることができます。インスタンスタイプを指定する代わりに、インスタンスに必要な属性を指定でき、Amazon EC2 はそうした属性ですべてのインスタンスタイプを識別します。詳細については、「EC2 フリートまたはスポットフリートのインスタンスタイプを選択するための属性を指定する」を参照してください。

タイプ `instant` の EC2 フリートでは、次の起動テンプレートパラメータを上書きすることもできます。

- `IamInstanceProfile` – インスタンスに関連付ける IAM インスタンスプロファイル。詳細については、「Amazon EC2 の IAM ロール」を参照してください。
- `KeyName` – インスタンスに使用するキーペアの名前。詳細については、「Amazon EC2 のキーペアと Amazon EC2 インスタンス」を参照してください。
- `MetadataOptions` – インスタンスのインスタンスメタデータサービス設定。詳細については、「インスタンスメタデータサービスを使用してインスタンスメタデータにアクセスする」を参照してください。

EC2 フリートのタイプが `instant` の場合、AMI ID ではなく Systems Manager パラメータを指定できます。Systems Manager パラメータは、オーバーライドまたは起動テンプレートに指定できます。詳細については、「AMI ID のかわりに Systems Manager パラメータを使用する」を参照してください。

タイプ instant の EC2 フリートの場合、LaunchTemplateSpecificationUserData パラメータを使用して起動テンプレート仕様レベルでユーザーデータを指定することもできます。このパラメータを使用すると、base64 でエンコードされたユーザーデータを、起動テンプレート自体に含めることなく提供することができます。ただし、base64 エンコード前のユーザーデータが 16 KB を超えることはできません。

JSON ファイルにフリートパラメータを指定できます。指定可能なすべてのパラメータについては、「EC2 フリートの設定オプションをすべて表示する」を参照してください。

フリート設定の例については、「EC2 フリートの CLI 設定の例」を参照してください。

EC2 フリートを作成するためのコンソールのサポートは現在ありません。

EC2 フリートを作成するには

create-fleet コマンドを使用して、フリートを作成し、フリート設定パラメータを記載した JSON ファイルを指定します。

```
aws ec2 create-fleet --cli-input-json file://file_name.json
```

タイプ request またはタイプ maintain のフリートの出力例を次に示します。

```
{
    "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE"
}
```

ターゲット容量を起動したタイプ instant のフリートの出力例を次に示します。

レスポンスには、起動されたインスタンスの各セットのアベイラビリティーゾーン ID、アベイラビリティーゾーン名、サブネット ID が含まれます。

```
{
  "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE",
  "Errors": [],
  "Instances": [
    {
      "LaunchTemplateAndOverrides": {
        "LaunchTemplateSpecification": {
          "LaunchTemplateId": "lt-01234a567b8910abcEXAMPLE",
          "Version": "1"
        },
```

```
        "Overrides": {
          "InstanceType": "c5.large",
          "AvailabilityZone": "us-east-1a"
        }
      },
      "Lifecycle": "on-demand",
      "InstanceIds": [
        "i-1234567890abcdef0",
        "i-9876543210abcdef9"
      ],
      "InstanceType": "c5.large",
      "Platform": null,
      "AvailabilityZoneId": "use1-az1",
      "AvailabilityZone": "us-east-1a",
      "SubnetId": "subnet-0123456789abcdefEXAMPLE"
    },
    {
      "LaunchTemplateAndOverrides": {
        "LaunchTemplateSpecification": {
          "LaunchTemplateId": "lt-01234a567b8910abcEXAMPLE",
          "Version": "1"
        },
        "Overrides": {
          "InstanceType": "c4.large",
          "AvailabilityZone": "us-east-1a"
        }
      },
      "Lifecycle": "on-demand",
      "InstanceIds": [
        "i-5678901234abcdef0",
        "i-5432109876abcdef9"
      ],
      "AvailabilityZoneId": "use1-az1",
      "AvailabilityZone": "us-east-1a",
      "SubnetId": "subnet-0123456789abcdefEXAMPLE"
    }
  ]
}
```

ターゲット容量の一部を起動し、起動されなかったインスタンスをエラーとするタイプ instant のフリートの出力例を次に示します。

```
{
```

```
  "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE",
  "Errors": [
    {
      "LaunchTemplateAndOverrides": {
        "LaunchTemplateSpecification": {
          "LaunchTemplateId": "lt-01234a567b8910abcEXAMPLE",
          "Version": "1"
        },
        "Overrides": {
          "InstanceType": "c4.xlarge",
          "AvailabilityZone": "us-east-1a",
        }
      },
      "Lifecycle": "on-demand",
      "ErrorCode": "InsufficientInstanceCapacity",
      "ErrorMessage": ""
    },
  ],
  "Instances": [
    {
      "LaunchTemplateAndOverrides": {
        "LaunchTemplateSpecification": {
          "LaunchTemplateId": "lt-01234a567b8910abcEXAMPLE",
          "Version": "1"
        },
        "Overrides": {
          "InstanceType": "c5.large",
          "AvailabilityZone": "us-east-1a"
        }
      },
      "Lifecycle": "on-demand",
      "InstanceIds": [
        "i-1234567890abcdef0",
        "i-9876543210abcdef9"
      ]
  ]
}
```

インスタンスを起動しなかったタイプ instant のフリートの出力例を次に示します。

```
{
  "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE",
  "Errors": [
```

```
    {
      "LaunchTemplateAndOverrides": {
        "LaunchTemplateSpecification": {
          "LaunchTemplateId": "lt-01234a567b8910abcEXAMPLE",
          "Version": "1"
        },
        "Overrides": {
          "InstanceType": "c4.xlarge",
          "AvailabilityZone": "us-east-1a",
        }
      },
      "Lifecycle": "on-demand",
      "ErrorCode": "InsufficientCapacity",
      "ErrorMessage": ""
    },
    {
      "LaunchTemplateAndOverrides": {
        "LaunchTemplateSpecification": {
          "LaunchTemplateId": "lt-01234a567b8910abcEXAMPLE",
          "Version": "1"
        },
        "Overrides": {
          "InstanceType": "c5.large",
          "AvailabilityZone": "us-east-1a",
        }
      },
      "Lifecycle": "on-demand",
      "ErrorCode": "InsufficientCapacity",
      "ErrorMessage": ""
    },
  ],
  "Instances": []
}
```

異常なスポットインスタンスを置き換える EC2 フリートを作成する

EC2 フリート は、2 分ごとにフリートのインスタンスのヘルスステータスをチェックします。インスタンスのヘルスステータスは healthy または unhealthy です。

EC2 フリート は Amazon EC2 によって提供されるステータスチェックを使用して、インスタンスのヘルスステータスを判断します。インスタンスステータスチェックまたはシステムステータスチェックのいずれかのステータスが 3 回の連続したヘルスステータスチェックで impaired の場合、イン

スタンスは unhealthy と判断されます。詳細については「Amazon EC2 インスタンスのステータスチェック」を参照してください。

フリートを設定して、異常のある スポットインスタンス を置き換えることができます。ReplaceUnhealthyInstances を true に設定した後、unhealthy として報告されたときにスポットインスタンスが置き換えられます。異常のあるスポットインスタンスを置き換えている間、最大数分間、フリートがターゲット容量を下回る場合があります。

要件

- ヘルスチェックによる置き換えは、タイプ request または instant のフリートではなく、ターゲットキャパシティを維持している EC2 フリート (タイプ maintain のフリート) でのみサポートされます。
- ヘルスチェックによる置き換えは、スポットインスタンス でのみサポートされます。この機能はオンデマンドインスタンス ではサポートされていません。
- 作成時のみ異常なインスタンスを置き換えるよう EC2 フリート を設定できます。
- ユーザーは、ec2:DescribeInstanceStatus アクションを呼び出す許可を持っている場合のみ、ヘルスチェックの置き換えを使用できます。

異常のある スポットインスタンス を置き換えるように EC2 フリート を設定するには

1. 「EC2 フリートの作成」で EC2 フリートを作成するための情報を使用します。
2. 異常のあるスポットインスタンスを置き換えるようにフリートを設定するには、JSON ファイルの ReplaceUnhealthyInstances に true を指定します。

EC2 フリートの設定オプションをすべて表示する

フリート設定パラメータの詳細なリストを見るには、JSON ファイルを次のように作成できます。各パラメータの説明については create-fleet を参照してください。

使用可能なすべての EC2 フリートパラメータを含めて JSON ファイルを生成するには

create-fleet AWS CLIコマンドと --generate-cli-skeleton パラメータを使用して、EC2 フリート JSON ファイルを生成し、出力のファイルへの保存を指示します。

```
aws ec2 create-fleet \
    --generate-cli-skeleton input > ec2createfleet.json
```

以下は出力の例です。

```
{
    "DryRun": true,
    "ClientToken": "",
    "SpotOptions": {
        "AllocationStrategy": "price-capacity-optimized",
        "MaintenanceStrategies": {
            "CapacityRebalance": {
                "ReplacementStrategy": "launch"
            }
        },
        "InstanceInterruptionBehavior": "hibernate",
        "InstancePoolsToUseCount": 0,
        "SingleInstanceType": true,
        "SingleAvailabilityZone": true,
        "MinTargetCapacity": 0,
        "MaxTotalPrice": ""
    },
    "OnDemandOptions": {
        "AllocationStrategy": "prioritized",
        "CapacityReservationOptions": {
            "UsageStrategy": "use-capacity-reservations-first"
        },
        "SingleInstanceType": true,
        "SingleAvailabilityZone": true,
        "MinTargetCapacity": 0,
        "MaxTotalPrice": ""
    },
    "ExcessCapacityTerminationPolicy": "termination",
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateId": "",
                "LaunchTemplateName": "",
                "Version": ""
            },
            "Overrides": [
                {
                    "InstanceType": "r5.metal",
                    "MaxPrice": "",
                    "SubnetId": "",
                    "AvailabilityZone": "",
                    "WeightedCapacity": 0.0,
```

```
                    "Priority": 0.0,
                    "Placement": {
                        "AvailabilityZone": "",
                        "Affinity": "",
                        "GroupName": "",
                        "PartitionNumber": 0,
                        "HostId": "",
                        "Tenancy": "dedicated",
                        "SpreadDomain": "",
                        "HostResourceGroupArn": ""
                    },
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 0
                        },
                        "MemoryMiB": {
                            "Min": 0,
                            "Max": 0
                        },
                        "CpuManufacturers": [
                            "amd"
                        ],
                        "MemoryGiBPerVCpu": {
                            "Min": 0.0,
                            "Max": 0.0
                        },
                        "ExcludedInstanceTypes": [
                            ""
                        ],
                        "InstanceGenerations": [
                            "previous"
                        ],
                        "SpotMaxPricePercentageOverLowestPrice": 0,
                        "OnDemandMaxPricePercentageOverLowestPrice": 0,
                        "BareMetal": "included",
                        "BurstablePerformance": "required",
                        "RequireHibernateSupport": true,
                        "NetworkInterfaceCount": {
                            "Min": 0,
                            "Max": 0
                        },
                        "LocalStorage": "excluded",
                        "LocalStorageTypes": [
```

```
                        "ssd"
                    ],
                    "TotalLocalStorageGB": {
                        "Min": 0.0,
                        "Max": 0.0
                    },
                    "BaselineEbsBandwidthMbps": {
                        "Min": 0,
                        "Max": 0
                    },
                    "AcceleratorTypes": [
                        "inference"
                    ],
                    "AcceleratorCount": {
                        "Min": 0,
                        "Max": 0
                    },
                    "AcceleratorManufacturers": [
                        "amd"
                    ],
                    "AcceleratorNames": [
                        "a100"
                    ],
                    "AcceleratorTotalMemoryMiB": {
                        "Min": 0,
                        "Max": 0
                    }
                }
            }
        ]
    }
],
"TargetCapacitySpecification": {
    "TotalTargetCapacity": 0,
    "OnDemandTargetCapacity": 0,
    "SpotTargetCapacity": 0,
    "DefaultTargetCapacityType": "on-demand",
    "TargetCapacityUnitType": "memory-mib"
},
"TerminateInstancesWithExpiration": true,
"Type": "instant",
"ValidFrom": "1970-01-01T00:00:00",
"ValidUntil": "1970-01-01T00:00:00",
"ReplaceUnhealthyInstances": true,
```

```
    "TagSpecifications": [
        {
            "ResourceType": "fleet",
            "Tags": [
                {
                    "Key": "",
                    "Value": ""
                }
            ]
        }
    ],
    "Context": ""
}
```

新規または既存の EC2 フリートリクエスト、およびそのフリートが起動するインスタンスとボリュームにタグ付けします。

EC2 フリートリクエスト、およびそのフリートが起動するインスタンスとボリュームを分類および管理しやすくするために、カスタムメタデータでタグ付けできます。EC2 フリート タグは、作成時または作成後にリクエストに割り当てることができます。同様に、インスタンスとボリュームにも、フリートによって起動されるときまたは起動後に、タグを割り当てることができます。

フリートリクエストにタグを付けると、フリートによって起動されるインスタンスとボリュームには自動的にタグ付けされません。フリートによって起動されるインスタンスとボリュームには、明示的にタグを付ける必要があります。タグは、フリートリクエストのみに割り当てるか、フリートによって起動されたインスタンスのみに割り当てるか、フリートによって起動されたインスタンスにアタッチされたボリュームのみに割り当てるか、フリートによって作成されたネットワークインターフェイスのみに割り当てるか、またはこれらすべてに割り当てるかを選択できます。

> **Note**
>
> `instant` フリートタイプでは、 オンデマンドインスタンス および スポットインスタンス にアタッチされているボリュームにタグ付けできます。`instant` フリートタイプのネットワークインターフェイスにタグを付けることもできます。`request` または `maintain` フリートタイプでは、オンデマンドインスタンス にアタッチされているボリュームにのみタグ付けできます。

タグの仕組みの詳細については、「Amazon EC2 リソースのタグ付け」を参照してください。

前提条件

リソースにタグ付けする許可をユーザーに付与します。詳細については「例: リソースのタグ付け」を参照してください。

リソースにタグ付けする許可をユーザーに付与するには

以下を含む IAM ポリシーを作成します。

- `ec2:CreateTags` アクション。これにより、タグを作成する許可がユーザーに付与されます。
- `ec2:CreateFleet` アクション。これにより、EC2 フリートリクエストを作成する許可がユーザーに付与されます。
- `Resource` に対しては、`"*"` を指定することをお勧めします。これにより、ユーザーはすべてのリソースタイプにタグ付けできます。

アクセス許可を付与するにはユーザー、グループ、またはロールにアクセス許可を追加します。

- AWS IAM アイデンティティセンター のユーザーとグループ:

  アクセス許可セットを作成します。「AWS IAM アイデンティティセンター ユーザーガイド」の「アクセス許可セットを作成する」の手順に従ってください。
- IAM 内で、ID プロバイダーによって管理されているユーザー:

  ID フェデレーションのロールを作成します。詳細については IAM ユーザーガイド の サードパーティー ID プロバイダー (フェデレーション) 用のロールを作成する を参照してください。
- IAM ユーザー:
  - ユーザーが担当できるロールを作成します。手順については IAM ユーザーガイド の IAM ユーザーのロールの作成 を参照してください。
  - (お奨めできない方法) ポリシーをユーザーに直接アタッチするか、ユーザーをユーザーグループに追加します。「IAM ユーザーガイド」の「ユーザー (コンソール) へのアクセス許可の追加」の指示に従います。

新しい EC2 フリート リクエストにタグ付けするには

作成時に EC2 フリート リクエストをタグ付けするには、フリートを作成するために使用した JSON ファイル でキーと値のペアを指定します。`ResourceType` の値は `fleet` にする必要があります。別の値で指定すると、フリートリクエストは失敗します。

EC2 フリート によって起動されたインスタンスおよびボリュームにタグ付けするには

フリートが起動したインスタンスおよびボリュームにタグ付けするには、EC2 フリート リクエストで参照される 起動テンプレート でタグを指定します。

> **Note**
>
> `request` または `maintain` フリートタイプによって起動される スポットインスタンス にアタッチされたボリュームにタグを付けることはできません。

既存の EC2 フリートリクエスト、インスタンス、ボリュームにタグ付けするには

create-tags コマンドを使用して、既存のリソースにタグを付けます。

```
aws ec2 create-tags \
    --resources fleet-12a34b55-67cd-8ef9-
ba9b-9208dEXAMPLE i-1234567890abcdef0 vol-1234567890EXAMPLE \
    --tags Key=purpose,Value=test
```

EC2 フリート、そのインスタンス、イベントを記述する

EC2 フリートの設定、EC2 フリートのインスタンス、EC2 フリートのイベント履歴を記述できます。

タスク

- EC2 フリートを記述する
- EC2 フリート内のすべてのインスタンスを記述する
- EC2 フリートのイベント履歴を表示する

EC2 フリートを記述する

AWS CLI

EC2 フリートを記述するには

describe-fleets コマンドを使用します。

```
aws ec2 describe-fleets \
```

```
    --fleet-ids fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

以下は出力の例です。

```
{
    "Fleets": [
        {
            "ActivityStatus": "fulfilled",
            "CreateTime": "2022-02-09T03:35:52+00:00",
            "FleetId": "fleet-364457cd-3a7a-4ed9-83d0-7b63e51bb1b7",
            "FleetState": "active",
            "ExcessCapacityTerminationPolicy": "termination",
            "FulfilledCapacity": 2.0,
            "FulfilledOnDemandCapacity": 0.0,
            "LaunchTemplateConfigs": [
                {
                    "LaunchTemplateSpecification": {
                        "LaunchTemplateName": "my-launch-template",
                        "Version": "$Latest"
                    }
                }
            ],
            "TargetCapacitySpecification": {
                "TotalTargetCapacity": 2,
                "OnDemandTargetCapacity": 0,
                "SpotTargetCapacity": 2,
                "DefaultTargetCapacityType": "spot"
            },
            "TerminateInstancesWithExpiration": false,
            "Type": "maintain",
            "ReplaceUnhealthyInstances": false,
            "SpotOptions": {
                "AllocationStrategy": "capacity-optimized",
                "InstanceInterruptionBehavior": "terminate"
            },
            "OnDemandOptions": {
                "AllocationStrategy": "lowestPrice"
            }
        }
    ]
}
```

PowerShell

EC2 フリートを記述するには

Get-EC2FleetList コマンドレットを使用します。

```
Get-EC2FleetList `
    -FleetId fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

EC2 フリート内のすべてのインスタンスを記述する

実行中のインスタンスの返されるリストは定期的に更新されますが、古い可能性もあります。

AWS CLI

指定した EC2 フリートのインスタンスを記述するには

describe-fleet-instances コマンドを使用します。

```
aws ec2 describe-fleet-instances \
    --fleet-id fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

以下は出力の例です。

```
{
    "ActiveInstances": [
        {
            "InstanceId": "i-09cd595998cb3765e",
            "InstanceHealth": "healthy",
            "InstanceType": "m4.large",
            "SpotInstanceRequestId": "sir-86k84j6p"
        },
        {
            "InstanceId": "i-09cf95167ca219f17",
            "InstanceHealth": "healthy",
            "InstanceType": "m4.large",
            "SpotInstanceRequestId": "sir-dvxi7fsm"
        }
    ],
    "FleetId": "fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE"
}
```

PowerShell

指定した EC2 フリートのインスタンスを記述するには

Get-EC2FleetInstanceList コマンドレットを使用します。

```
Get-EC2FleetInstanceList `
    -FleetId fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

EC2 フリートのイベント履歴を表示する

イベント履歴のイベントの詳細については、「EC2 フリート イベントタイプ」を参照してください。

AWS CLI

指定した EC2 フリートのイベントを記述するには

describe-fleet-history コマンドを使用します。

```
aws ec2 describe-fleet-history \
    --fleet-id fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --start-time 2020-06-01T00:00:00Z
```

以下は出力の例です。

```
{
    "HistoryRecords": [
        {
            "EventInformation": {
                "EventSubType": "submitted"
            },
            "EventType": "fleetRequestChange",
            "Timestamp": "2020-09-01T18:26:05.000Z"
        },
        {
            "EventInformation": {
                "EventSubType": "active"
            },
            "EventType": "fleetRequestChange",
```

```
            "Timestamp": "2020-09-01T18:26:15.000Z"
        },
        {
            "EventInformation": {
                "EventDescription": "t2.small, ami-07c8bc5c1ce9598c3, ...",
                "EventSubType": "progress"
            },
            "EventType": "fleetRequestChange",
            "Timestamp": "2020-09-01T18:26:17.000Z"
        },
        {
            "EventInformation": {
                "EventDescription": "{\"instanceType\":\"t2.small\", ...}",
                "EventSubType": "launched",
                "InstanceId": "i-083a1c446e66085d2"
            },
            "EventType": "instanceChange",
            "Timestamp": "2020-09-01T18:26:17.000Z"
        },
        {
            "EventInformation": {
                "EventDescription": "{\"instanceType\":\"t2.small\", ...}",
                "EventSubType": "launched",
                "InstanceId": "i-090db02406cc3c2d6"
            },
            "EventType": "instanceChange",
            "Timestamp": "2020-09-01T18:26:17.000Z"
        }
    ],
    "FleetId": "fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
    "LastEvaluatedTime": "1970-01-01T00:00:00.000Z",
    "StartTime": "2020-06-01T00:00:00.000Z"
}
```

PowerShell

指定した EC2 フリートのイベントを記述するには

Get-EC2FleetHistory コマンドレットを使用します。

```
Get-EC2FleetHistory `
    -FleetId fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE `
    -UtcStartTime 2020-06-01T00:00:00Z
```

EC2 フリート の変更

EC2 フリートの合計ターゲットキャパシティ、スポットキャパシティ、オンデマンドキャパシティを変更できます。また、新しい合計ターゲットキャパシティがフリートの現在のサイズを下回った場合に、実行中のインスタンスを終了すべきかどうかも変更できます。

考慮事項

EC2 フリートを変更するときは、以下の点を考慮してください。

- フリートタイプ - 変更できるのは、`maintain` タイプの EC2 フリートだけです。タイプが `request` や `instant` の EC2 フリートは変更できません。
- フリートパラメータ - EC2 フリートの以下のパラメータを変更できます。
  - `target-capacity-specification` – 以下のターゲットキャパシティを増やすか減らします。
    - `TotalTargetCapacity`
    - `OnDemandTargetCapacity`
    - `SpotTargetCapacity`
  - `excess-capacity-termination-policy` – EC2 フリート の合計ターゲット容量がフリートの現在のサイズより小さくなった場合、実行中のインスタンスが終了されるかどうか。次の値を指定できます。
    - `no-termination`
    - `termination`
- 合計ターゲットキャパシティを増やしたときのフリートの動作 – 合計ターゲットキャパシティを増やすと、EC2 フリートは `DefaultTargetCapacityType` に指定されているインスタンス購入オプションに従ってさらに別のインスタンスを起動します。これはオンデマンドインスタンスかスポットインスタンスのいずれかであり、指定された配分戦略に従います。
- スポットターゲットキャパシティを減らしたときのフリートの動作 – ターゲットキャパシティを減らすと、EC2 フリートは新しいターゲットキャパシティを超えるすべてのオープンリクエストをキャンセルします。フリートのサイズが新しいターゲットキャパシティに達するまで、スポットインスタンスを終了させるようにリクエストできます。ターゲット容量が減ったために EC2 フリートがスポットインスタンスを終了する場合、インスタンスはスポットインスタンスの中断通知を受け取ります。

  インスタンスは、割り当て戦略に応じて、終了対象として選択されます。

- `capacity-optimized` – 使用可能なキャパシティが最も少ないプールからのインスタンスを終了します。
- `price-capacity-optimized` – 価格と使用可能なキャパシティの組み合わせを使用します。使用可能なキャパシティが最も少ないプールの中で、最高価格のプールからのインスタンスを終了します。
- `diversified` – すべてのプールのインスタンスを終了します。
- `lowest-price` – 最高価格のプールからのインスタンスを終了します。

あるいは、EC2 フリートの現在のサイズを維持するようにリクエストすることもできますが、中断されたスポットインスタンスや手動で終了したインスタンスの置き換えはできません。

- フリート状態 - 状態が `submitted` または `active` の EC2 フリートを変更できます。フリートを変更すると、そのフリートは `modifying` 状態に移行します。

EC2 フリートを変更するためのコマンド

AWS CLI

EC2 フリートの合計ターゲットキャパシティを変更するには

modify-fleet コマンドを使用します。

```
aws ec2 modify-fleet \
    --fleet-id fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --target-capacity-specification TotalTargetCapacity=20
```

ターゲットキャパシティを小さくしてもフリートの現在のサイズを保持する場合は、前の例を以下のように変更できます。

```
aws ec2 modify-fleet \
    --fleet-id fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --target-capacity-specification TotalTargetCapacity=10 \
    --excess-capacity-termination-policy no-termination
```

PowerShell

EC2 フリートの合計ターゲットキャパシティを変更するには

Edit-EC2Fleet コマンドレットを使用します。

```
Edit-EC2Fleet `
    -FleetId "fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE" `
    -TargetCapacitySpecification_TotalTargetCapacity 20
```

ターゲットキャパシティを小さくしてもフリートの現在のサイズを保持する場合は、前の例を以下のように変更できます。

```
Edit-EC2Fleet `
    -FleetId "fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE" `
    -TargetCapacitySpecification_TotalTargetCapacity 20 `
    -ExcessCapacityTerminationPolicy "NoTermination"
```

EC2 フリートリクエストおよびフリート内のインスタンスを削除する

EC2 フリートリクエストが不要になった場合には削除できます。フリートリクエストを削除すると、そのフリートに関連付けられているすべてのスポットリクエストがキャンセルされるため、新しいスポットインスタンスは起動されません。

EC2 フリートリクエストを削除するときは、そのインスタンスをすべて終了させるかどうかを指定する必要があります。これには、オンデマンドインスタンスとスポットインスタンスの両方が含まれます。instant フリートの場合、EC2 フリートはフリートの削除時にインスタンスを終了する必要があります。実行中のインスタンスを持つ削除した instant フリートはサポートされていません。

> **Warning**
>
> インスタンスの終了は恒久的で、元に戻すことができません。
> インスタンスを終了すると、そのインスタンスには接続できなくなり、復元することもできません。アタッチされている Amazon EBS ボリュームで、終了時に削除されるように設定されているものも、すべて恒久的に削除され、復元できません。インスタンスストアボリュームに保存されているデータは、すべて恒久的に失われます。詳細については「インスタンスの終了方法」を参照してください。
> インスタンスを終了する前に、終了後も保持する必要があるすべてのデータを永続的ストレージにバックアップしたことを確認してください。

フリートリクエストを削除するときに、インスタンスを終了するように指定した場合、フリートリクエストは deleted_terminating 状態に移行します。それ以外の場合は deleted_running 状態になり、インスタンスは中断または手動終了されるまで、引き続き実行されます。

制限事項

- 1 回の操作で最大 25 個の `instant` タイプのフリートを削除できます。
- 1 回の操作で最大 100 個の `maintain` または `request` タイプのフリートを削除できます。
- 上記のように、各フリートタイプのクォータを超えない場合は、1 回の操作で最大 125 個のフリートを削除できます。
- 削除するフリートの指定された数を超えると、フリートは削除されません。
- 実行中のインスタンスを持つ削除した `instant` フリートはサポートされていません。`instant` フリートを削除すると、Amazon EC2 はすべてのインスタンスを自動的に終了させます。1000 個を超えるインスタンスで構成された `instant` フリートの場合、削除リクエストは失敗する可能性があります。1000 個を超えるインスタンスで構成されたフリートの場合は、まずほとんどのインスタンスを手動で終了させ、残りのインスタンスが 1000 個以下になるようにします。次に、フリートを削除すると、残りのインスタンスは自動的に終了します。

AWS CLI

EC2 フリートリクエストを削除してインスタンスを終了するには

delete-fleets コマンドを `--terminate-instances` オプションと共に使用します。

```
aws ec2 delete-fleets \
    --fleet-ids fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --terminate-instances
```

以下は出力の例です。

```
{
    "UnsuccessfulFleetDeletions": [],
    "SuccessfulFleetDeletions": [
        {
            "CurrentFleetState": "deleted_terminating",
            "PreviousFleetState": "active",
            "FleetId": "fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE"
        }
    ]
}
```

インスタンスを終了せずに EC2 フリートリクエストを削除するには

代わりに --no-terminate-instances オプションを使用して、前の例を変更します。--no-terminate-instances は instant フリートではサポートされていないことに注意してください。

```
aws ec2 delete-fleets \
    --fleet-ids fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --no-terminate-instances
```

以下は出力の例です。

```
{
    "UnsuccessfulFleetDeletions": [],
    "SuccessfulFleetDeletions": [
        {
            "CurrentFleetState": "deleted_running",
            "PreviousFleetState": "active",
            "FleetId": "fleet-4b8aaae8-dfb5-436d-a4c6-3dafa4c6b7dcEXAMPLE"
        }
    ]
}
```

PowerShell

EC2 フリートリクエストを削除してインスタンスを終了するには

Remove-EC2Fleet コマンドレットを -TerminateInstance パラメータと共に使用します。

```
Remove-EC2Fleet `
    -FleetId "fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE" `
    -TerminateInstance $true
```

インスタンスを終了せずに EC2 フリートリクエストを削除するには

-TerminateInstance パラメータの値を変更して、前の例を変更します。

```
Remove-EC2Fleet `
    -FleetId "fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE" `
    -TerminateInstance $false
```

フリートの削除に失敗した場合のトラブルシューティング

EC2 フリートリクエスト の削除に失敗すると、出力中の UnsuccessfulFleetDeletions は EC2 フリートリクエスト の ID、エラーコード、エラーメッセージを返します。

エラーコードは次のとおりです。

- ExceededInstantFleetNumForDeletion
- fleetIdDoesNotExist
- fleetIdMalformed
- fleetNotInDeletableState
- NoTerminateInstancesNotSupported
- UnauthorizedOperation
- unexpectedError

ExceededInstantFleetNumForDeletion のトラブルシューティング

1 回のリクエストで 25 個 を超える instant フリートを削除しようとすると、ExceededInstantFleetNumForDeletion エラーが返されます。このエラーの出力例を次に示します。

```
{
    "UnsuccessfulFleetDeletions": [
     {
          "FleetId": " fleet-5d130460-0c26-bfd9-2c32-0100a098f625",
          "Error": {
                  "Message": "Can’t delete more than 25 instant fleets in a single
 request.",
                  "Code": "ExceededInstantFleetNumForDeletion"
           }
     },
     {
           "FleetId": "fleet-9a941b23-0286-5bf4-2430-03a029a07e31",
           "Error": {
                  "Message": "Can’t delete more than 25 instant fleets in a single
 request.",
                  "Code": "ExceededInstantFleetNumForDeletion"
            }
     }
```

```
      .
      .
      .
      ],
      "SuccessfulFleetDeletions": []
}
```

NoTerminateInstancesNotSupported のトラブルシューティング

フリートを削除するときに instant フリート内のインスタンスを終了しないように指定した場合、NoTerminateInstancesNotSupported エラーが返されます。--no-terminate-instances は instant フリートではサポートされていません。このエラーの出力例を次に示します。

```
{
      "UnsuccessfulFleetDeletions": [
            {
                  "FleetId": "fleet-5d130460-0c26-bfd9-2c32-0100a098f625",
                  "Error": {
                          "Message": "NoTerminateInstances option is not supported for
 instant fleet",
                          "Code": "NoTerminateInstancesNotSupported"
                   }
            }
       ],
       "SuccessfulFleetDeletions": []
}
```

UnauthorizedOperation のトラブルシューティング

インスタンスを終了するアクセス許可がない場合、インスタンスを終了する必要があるフリートを削除するときに UnauthorizedOperation エラーが発生します。以下はエラーレスポンスです。

```
<Response><Errors><Error><Code>UnauthorizedOperation</Code><Message>You are not
 authorized to perform this
operation. Encoded authorization failure message: VvuncIxj7Z_CPGNYXWqnuFV-
YjByeAU66Q9752NtQ-I3-qnDLWs6JLFd
KnSMMiq5s6cGqjjPtEDpsnGHzzyHasFHOaRYJpaDVravoW25azn6KNkUQQlFwhJyujt2dtNCdduJfrqcFYAjlEiRMkfDHt7
BHturzDK6A560Y2nDSUiMmAB1y9UNtqaZJ9SNe5sNxKMqZaqKtjRbk02RZu5V2vn9VMk6fm2aMVHbY9JhLvGypLcMUjtJ76
VPiU5v2s-
UgZ7h0p2yth6ysUdhlONg6dBYu8_y_HtEI54invCj4CoK0qawqzMNe6rcmCQHvtCxtXsbkgyaEbcwmrm2m01-
EMhekLFZeJLr
```

```
DtYOpYcEl4_nWFX1wtQDCnNNCmxnJZAoJvb3VMDYpDTsxjQv1PxODZuqWHs23YXWVywzgnLtHeRf2o4lUhGBw17mXsS07k7
PT9vrHtQiILor5VVTsjSPWg7edj__1rsnXhwPSu8gI48ZLRGrPQqFq0RmKO_QIE8N8s6NWzCK4yoX-9gDcheurOGpkprPIC
</Message></Error></Errors><RequestID>89b1215c-7814-40ae-a8db-41761f43f2b0</
RequestID></Response>
```

エラーを解決するには、次の例に示すように、ec2:TerminateInstances アクションを IAM ポリシーに追加する必要があります。

スポットフリートを操作する

> **Important**
>
> スポットフリートは投資予定がないレガシー API であるため、使用はお勧めしません。インスタンスのライフサイクルを管理する場合は、代わりに EC2 フリートを使用してください。インスタンスのライフサイクルを管理しない場合は、代わりに Auto Scaling グループを使用してください。スポットフリートは、通常なら EC2 フリートを使用するユースケースで、コンソールのサポートが必要な場合にのみ使用します。詳細については「フリートにどの方法を使用するのが最適か」を参照してください。

スポットフリートの使用を開始するには、スポットインスタンスの合計ターゲットキャパシティ、オプションのオンデマンド部分を含むリクエストを作成し、AMI とキーペアを手動で指定するか、フリート内のインスタンスの設定を含む起動テンプレートを指定します。必要に応じて、さらに別のパラメータを指定したり、デフォルト値を使用するようにフリートを設定したりできます。また、フリートを作成するときに、フリートリクエスト、およびフリート内のインスタンスとボリュームにタグ付けすることもできます。

フリートは、使用可能なキャパシティがあるときはオンデマンドインスタンスを起動し、上限価格がスポット料金より大きく、キャパシティが利用可能なときはスポットインスタンスを起動します。

フリートを起動したら、フリートリクエスト、フリート内のインスタンス、およびフリートイベントの説明を記述できます。また、必要に応じてさらに別のタグを割り当てることもできます。

合計ターゲットキャパシティなどのフリートパラメータを変更する必要がある場合は、キャパシティを維持するように設定されているのであれば、フリートを変更できます。送信後にワンタイムリクエストのキャパシティを変更することはできません。

フリートリクエストは、期限切れになるかお客様によってキャンセル (削除) されるまで、アクティブのままになります。フリートリクエストをキャンセルするときは、インスタンスを終了すること

も、実行したままにすることもできます。実行したままにする場合、オンデマンドインスタンスは、終了されるまで実行され、スポットインスタンスは中断されるか終了されるまで実行されます。

トピック

- スポットフリートリクエストの状態
- スポットフリートアクセス許可
- スポットフリートを作成する
- 新しいまたは既存のスポットフリートリクエスト、およびそのフリートが起動するインスタンスとボリュームにタグ付けする
- スポットフリートリクエスト、そのインスタンス、イベント履歴を記述する
- スポットフリートリクエストを変更します。
- スポットフリートリクエストをキャンセル (削除) します。
- スポットフリートの自動スケーリングについて

スポットフリートリクエストの状態

> **Important**
>
> スポットフリートは、計画された投資なしでレガシー API を使用します。代わりに EC2 フリートまたは Auto Scaling グループを使用することをお勧めします。詳細については「フリートにどの方法を使用するのが最適か」を参照してください。

スポットフリートリクエストはさまざまな状態の 1 つであり、各状態はリクエストライフサイクルとインスタンスの管理の異なる段階を示します。

スポットフリートリクエストは、次の状態のいずれかになります。

`submitted`

スポットフリートリクエストは評価中です。Amazon EC2 は目標数のインスタンスを起動する準備をしています。スポットフリートのクォータを超えたリクエストは、即時キャンセルされます。

`active`

スポットフリートは検証済みです。Amazon EC2 はターゲット数の実行中のスポットインスタンスを維持しようとしています。リクエストは、変更またはキャンセルされるまで、この状態のままになります。

`modifying`

スポットフリートリクエストは変更中です。リクエストは、変更が完全に処理されるか、リクエストがキャンセルされるまで、この状態のままになります。`maintain` フリートタイプのみを変更できます。この状態は 1 回限りの `request` フリートタイプには適用されません。

`cancelled_running`

スポットフリートはキャンセル (削除) され、追加のスポットインスタンスを起動しません。その既存のインスタンスは、手動で中断または終了されるまで実行され続けます。リクエストは、すべてのインスタンスが中断されるか終了されるまで、この状態のままになります。

`cancelled_terminating`

スポットフリートがキャンセル (削除) され、そのインスタンスが終了しています。リクエストは、すべてのインスタンスが終了されるまで、この状態のままになります。

`cancelled`

このスポットフリートはキャンセル (削除) されており、実行中のインスタンスはありません。リクエストは、そのインスタンスが終了されてから 2 日後に削除されます。

スポットフリートアクセス許可

> **Important**
>
> スポットフリートは、計画された投資なしでレガシー API を使用します。代わりに EC2 フリートまたは Auto Scaling グループを使用することをお勧めします。詳細については「フリートにどの方法を使用するのが最適か」を参照してください。

ユーザーがスポットフリートを作成または管理する場合、必要な許可を付与する必要があります。

Amazon EC2 コンソールを使用してスポットフリートを作成した場合、`AWSServiceRoleForEC2SpotFleet` および `AWSServiceRoleForEC2Spot` というサービスにリンクされた 2 つのロールと、`aws-ec2-spot-fleet-tagging-role` というロールが作成され

ます。ユーザーの代わりに、リソースのリクエスト、起動、終了、タグ付けを行うアクセス許可をスポットフリートに与えます。AWS CLI または API を使用する場合は、これらのロールが存在することを確認する必要があります。

次の手順に従って、必要なアクセス許可を付与し、ロールを作成します。

アクセス許可とロール

- ユーザーにスポットフリートの許可を付与する
- スポットフリート用のサービスにリンクされたロール
- スポットインスタンス用のサービスにリンクされたロール
- スポットフリートにタグ付けするための IAM ロール

ユーザーにスポットフリートの許可を付与する

ユーザーがスポットフリートを作成または管理する場合、必ず必要な許可を付与してください。

スポットフリートのポリシーを作成するには

1. IAM コンソールの https://console.aws.amazon.com/iam/ を開いてください。
2. ナビゲーションペインで、[Policies]、[Create policy] の順に選択してください。
3. [ポリシーの作成] ページで、 [JSON] を選択し、テキストを以下に置き換えます。

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances",
                "ec2:CreateTags",
                "ec2:RequestSpotFleet",
                "ec2:ModifySpotFleetRequest",
                "ec2:CancelSpotFleetRequests",
                "ec2:DescribeSpotFleetRequests",
                "ec2:DescribeSpotFleetInstances",
                "ec2:DescribeSpotFleetRequestHistory"
            ],
```

```
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::*:role/aws-ec2-spot-fleet-tagging-role"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole",
                "iam:ListRoles",
                "iam:ListInstanceProfiles"
            ],
            "Resource": "*"
        }
    ]
}
```

前述したポリシーの例では、ほとんどのスポットフリートのユースケースで必要な許可をユーザーに付与します。特定の API アクションに制限するには、代わりにこれらの API アクションのみを指定します。

必要な EC2 および IAM の API

ポリシーには、次の API を含める必要があります。

- `ec2:RunInstances` - スポットフリートでインスタンスを起動するために必要
- `ec2:CreateTags` - スポットフリートのリクエスト、インスタンス、またはボリュームのタグ付けに必要
- `iam:PassRole` - スポットフリートロールを指定するために必要
- `iam:CreateServiceLinkedRole` - サービスにリンクされたロールの作成に必要
- `iam:ListRoles` - 既存の IAM ロールを列挙するために必要
- `iam:ListInstanceProfiles` - 既存のインスタンスプロファイルを列挙するために必要

> **Important**
>
> 起動仕様または起動テンプレートで IAM インスタンスプロファイルのロールを指定する場合は、そのロールをサービスに渡す許可をユーザーに付与する必要があります。これを行うには、IAM ポリシーで `iam:PassRole` アクションのリソースとして "`arn:aws:iam::*:role/IamInstanceProfile-role`" を含めます。詳細については、「IAM ユーザーガイド」の「AWS サービスにロールを渡すアクセス権限をユーザーに付与する」を参照してください。

スポットフリートの API

必要に応じて、次のスポットフリート API アクションをポリシーに追加します。

- `ec2:RequestSpotFleet`
- `ec2:ModifySpotFleetRequest`
- `ec2:CancelSpotFleetRequests`
- `ec2:DescribeSpotFleetRequests`
- `ec2:DescribeSpotFleetInstances`
- `ec2:DescribeSpotFleetRequestHistory`

オプションの IAM API

(オプション) ユーザーが IAM コンソールを使用してロールまたはインスタンスプロファイルを作成できるようにするには、次のアクションをポリシーに追加する必要があります。

- `iam:AddRoleToInstanceProfile`
- `iam:AttachRolePolicy`
- `iam:CreateInstanceProfile`
- `iam:CreateRole`
- `iam:GetRole`
- `iam:ListPolicies`

4. [ポリシーの確認] を選択してください。
5. [ポリシーの確認] ページでポリシー名と説明を入力し、[ポリシーの作成] を選択してください。

6. アクセス許可を付与するにはユーザー、グループ、またはロールにアクセス許可を追加します。

   - AWS IAM アイデンティティセンター のユーザーとグループ:

     アクセス許可セットを作成します。「AWS IAM アイデンティティセンター ユーザーガイド」の「アクセス許可セットを作成する」の手順に従ってください。
   - IAM 内で、ID プロバイダーによって管理されているユーザー:

     ID フェデレーションのロールを作成します。詳細については IAM ユーザーガイド の サードパーティー ID プロバイダー (フェデレーション) 用のロールを作成する を参照してください。
   - IAM ユーザー:
     - ユーザーが担当できるロールを作成します。手順については IAM ユーザーガイド の IAM ユーザーのロールの作成 を参照してください。
     - (お奨めできない方法) ポリシーをユーザーに直接アタッチするか、ユーザーをユーザーグループに追加します。「IAM ユーザーガイド」の「ユーザー (コンソール) へのアクセス許可の追加」の指示に従います。

スポットフリート用のサービスにリンクされたロール

Amazon EC2 は、ユーザーに代わって AWS の他のサービスを呼び出すために必要なアクセス許可のために、サービスにリンクされたロールを使用します。サービスにリンクされたロールは、AWS のサービスに直接リンクされた一意のタイプの IAM ロールです。サービスにリンクされたロールは、AWS のサービスにアクセス許可を委任するためのセキュアな方法を提供します。これは、リンクされたサービスのみが、サービスにリンクされたロールを引き受けることができるためです。詳細については「IAM ユーザーガイド」の「Service-linked roles」を参照してください。

Amazon EC2 は、AWSServiceRoleForEC2SpotFleet という、サービスにリンクされたロールを使用して、ユーザーの代わりにインスタンスを起動して管理します。

> **Important**
>
> 暗号化された AMI または暗号化された Amazon EBS スナップショットをスポットフリートで指定した場合は、CMK を使用して Amazon EC2 がユーザーの代わりにインスタンスを起動する許可を AWSServiceRoleForEC2SpotFleet ロールに付与する必要があります。詳細については「暗号化された AMI および EBS スナップショット用の CMK へのアクセス権の付与」を参照してください。

AWSServiceRoleForEC2SpotFleet によって付与されるアクセス許可

[AWSServiceRoleForEC2SpotFleet] ロールは、ユーザーに代わってインスタンスのリクエスト、起動、終了、タグ付けを行う許可をスポットフリートに付与します。Amazon EC2 は、このサービスにリンクされたロールを使用して、以下のアクションを完了します。

- `ec2:RequestSpotInstances` - スポットインスタンスをリクエスト
- `ec2:RunInstances` - インスタンスを起動
- `ec2:TerminateInstances` - インスタンスを終了
- `ec2:DescribeImages` - インスタンスの Amazon マシンイメージ (AMI) を表示
- `ec2:DescribeInstanceStatus` - インスタンスのステータスを表示
- `ec2:DescribeSubnets` - インスタンスのサブネットを記述
- `ec2:CreateTags` - スポットフリートリクエスト、インスタンス、ボリュームにタグを追加
- `elasticloadbalancing:RegisterInstancesWithLoadBalancer` - 指定されたインスタンスを指定されたロードバランサーに追加
- `elasticloadbalancing:RegisterTargets` - 指定されたターゲットを指定されたターゲットグループに登録

サービスにリンクされたロールの作成

ほとんどの状況では、サービスにリンクされたロールを手動で作成する必要はありません。Amazon EC2 は、コンソールを使用してスポットフリートを初めて作成するときに、AWSServiceRoleForEC2SpotFleet サービスにリンクされたロールを作成します。

Amazon EC2 がこのサービスにリンクされたロールのサポートを開始した 2017 年 10 月よりも前にアクティブなスポットフリートリクエストを行った場合、Amazon EC2は AWS アカウントで AWSServiceRoleForEC2SpotFleet ロールを作成します。詳細については、IAM ユーザーガイドの「 アカウントに新しいロールが表示される」を参照してください。AWS

AWS CLI または API を使用してスポットフリートを作成する場合、最初にこのロールが存在しているか確認する必要があります。

コンソールを使用してスポットフリートの [AWSServiceRoleForEC2SpotFleet] ロールを作成するには

1. IAM コンソールの https://console.aws.amazon.com/iam/ を開いてください。
2. ナビゲーションペインで Roles (ロール) を選択してください。

3. [Create role] を選択してください。
4. [信頼されたエンティティを選択] ページで、以下の操作を実行します。

   a. [信頼できるエンティティタイプ] で、[AWS サービス] を選択してください。
   b. [ユースケース] の [サービスまたはユースケース] で、[EC2] を選択してください。
   c. [ユースケース] で、[EC2 - スポットフリート] を選択してください。

   > **Note**
   >
   > [EC2 - スポットフリート] ユースケースでは、必要な IAM アクセス許可を持つポリシーが自動的に作成され、ロール名として [AWSEC2SpotFleetServiceRolePolicy] が提案されます。

   d. [Next] を選択してください。
5. [アクセス許可を追加] ページで [次へ] を選択してください。
6. [名前、確認、および作成] ページで、[ロールの作成] をクリックしてください。

AWS CLI を使用してスポットフリートの [AWSServiceRoleForEC2SpotFleet] ロールを作成するには

次のように、create-service-linked-role コマンドを使用します。

```
aws iam create-service-linked-role --aws-service-name spotfleet.amazonaws.com
```

スポットフリートを使用する必要がなくなった場合は、AWSServiceRoleForEC2SpotFleet ロールを削除することをお勧めします。このロールがアカウントから削除された後、コンソールを使用してスポットフリートをリクエストすると、Amazon EC2 はロールを再作成します。詳細については、「IAM ユーザーガイド」の「サービスにリンクされたロールの削除」を参照してください。

暗号化された AMI および EBS スナップショット用の CMK へのアクセス権の付与

暗号化された AMI または暗号化された Amazon EBS スナップショットをスポットフリートリクエストで指定し、カスタマー管理された型キーを暗号化に使用する場合は、CMK を使用して、Amazon EC2 がユーザーの代わりにインスタンスを起動する許可を、AWSServiceRoleForEC2SpotFleet ロールに付与する必要があります。これを行うには、次の手順で示すように、CMK に付与を追加する必要があります。

アクセス権限を設定するときは、付与がキーポリシーの代わりになります。詳細については、 デベロッパーガイドの「許可の使用」と「 でのキーポリシーの使用」を参照してください。https://

docs.aws.amazon.com/kms/latest/developerguide/grants.htmlAWS KMSAWS Key Management Service

CMK を使用するアクセス許可を AWSServiceRoleForEC2SpotFleet ロールに付与するには

- create-grant コマンドを使用して CMK に付与を追加し、プリンシパル (サービスにリンクされたロール AWSServiceRoleForEC2SpotFleet) を指定します。このプリンシパルには、付与が許可するオペレーションを実行するためのアクセス許可が提供されます。https://docs.aws.amazon.com/cli/latest/reference/kms/create-grant.htmlCMK を指定するには、 パラメータと CMK の ARN を使用します。`key-id`プリンシパルを指定するには、 パラメータとサービスにリンクされたロール AWSServiceRoleForEC2SpotFleet の ARN を使用します。`grantee-principal`

```
aws kms create-grant \
    --region us-east-1 \
    --key-id arn:aws:kms:us-
east-1:444455556666:key/1234abcd-12ab-34cd-56ef-1234567890ab \
    --grantee-principal arn:aws:iam::111122223333:role/aws-service-role/
spotfleet.amazonaws.com/AWSServiceRoleForEC2SpotFleet \
    --operations "Decrypt" "Encrypt" "GenerateDataKey"
 "GenerateDataKeyWithoutPlaintext" "CreateGrant" "DescribeKey" "ReEncryptFrom"
 "ReEncryptTo"
```

スポットインスタンス用のサービスにリンクされたロール

Amazon EC2 は、AWSServiceRoleForEC2Spot という、サービスにリンクされたロールを使用して、ユーザーの代わりに スポットインスタンス を起動して管理します。詳細については「スポットインスタンスリクエスト向けのサービスにリンクされたロール」を参照してください。

スポットフリートにタグ付けするための IAM ロール

`aws-ec2-spot-fleet-tagging-role` IAM ロールは、スポットフリートリクエスト、インスタンス、ボリュームにタグ付けするアクセス権限をスポットフリートに付与します。詳細については「新しいまたは既存のスポットフリートリクエスト、およびそのフリートが起動するインスタンスとボリュームにタグ付けする」を参照してください。

> **Important**
>
> フリートのインスタンスにタグ付けすることを選択し、ターゲット容量を維持することを選択した場合 (スポットフリートリクエストのタイプは `maintain`)、ユーザーと `IamFleetRole` の許可の違いにより、フリートのインスタンスのタグ付け動作に整合性がなくなる可能性があります。`IamFleetRole` に `CreateTags` アクセス許可が含まれていない場合、フリートによって起動されたインスタンスの一部がタグ付けされていない可能性があります。当社はこの不整合の修正に取り組んでいますが、フリートによって起動されたすべてのインスタンスがタグ付けされるようにするために、`IamFleetRole`には`aws-ec2-spot-fleet-tagging-role`ロールを使用することをお勧めします。または、既存のロールを使用するには、`AmazonEC2SpotFleetTaggingRole` の AWS 管理ポリシーを既存のロールにアタッチします。それ以外の場合は、既存のポリシーに `CreateTags` アクセス許可を手動で追加する必要があります。

スポットフリートにタグ付けする IAM ロールを作成するには

1. IAM コンソールの https://console.aws.amazon.com/iam/ を開いてください。
2. ナビゲーションペインで Roles (ロール) を選択してください。
3. [Create role] を選択してください。
4. [Select trusted entity] (信頼できるエンティティの選択) ページの [Trusted entity type] (信頼できるエンティティタイプ) で、[AWS service] (のサービス) を選択してください。
5. [Use case] (ユースケース) で、[Use cases for other AWS services] (他の サービスでのユースケース) から [EC2] を選択し、[EC2 - Spot Fleet Tagging] (EC2 - スポットフリートのタグ付け) を選択してください。
6. [Next] を選択してください。
7. [アクセス許可を追加] ページで [次へ] を選択してください。
8. [Name, review, and create] (名前、レビュー、および作成) ページで、[Role name] (ロール名) にロールの名前 (例えば、**`aws-ec2-spot-fleet-tagging-role`**) を入力してください。
9. ページ内の情報を確認し、[Create role] (ロールを作成) をクリックしてください。

サービス間の混乱した代理の防止

「混乱した代理」問題は、アクションを実行するためのアクセス許可を持たないエンティティが、より特権のあるエンティティにアクションの実行を強制できてしまう場合に生じる、セキュリティ

上の問題です。aws-ec2-spot-fleet-tagging-role 信頼ポリシー内で aws:SourceArn と aws:SourceAccount のグローバル条件コンテキストキーを使用して、リソースについてスポットフリートが別のサービスに付与するアクセス許可を制限することをお勧めします。

aws:SourceArn および aws:SourceAccount 条件キーを **aws-ec2-spot-fleet-tagging-role** 信頼ポリシーに追加するには

1. IAM コンソールの https://console.aws.amazon.com/iam/ を開いてください。
2. ナビゲーションペインで [Roles] (ロール) を選択してください。
3. 前に作成した aws-ec2-spot-fleet-tagging-role を見つけ、リンク (チェックボックスではありません) を選択してください。
4. [Summary] (概要) にある [Trust relationships] (信頼関係) タブを開き、[Edit trust policy] (信頼ポリシーの編集) をクリックしてください。
5. 「混乱した代理」問題を防止するために、JSON ステートメント内で、以下のようにグローバル条件コンテキストキー aws:SourceAccount および aws:SourceArn を含む Condition 要素を追加します。

```
"Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:ec2:us-east-1:111122223333:spot-fleet-request/
sfr-*"
      },
      "StringEquals": {
        "aws:SourceAccount": "111122223333"
      }
```

> **Note**
>
> aws:SourceArn の値にアカウント ID が含まれており、上記のグローバル条件コンテキストキーの両方を同じポリシーステートメント内で使用する場合、aws:SourceAccount 値と aws:SourceArn 値の中のアカウントには、同じアカウント ID を使用する必要があります。

最終的な信頼ポリシーは次のようになります。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
      "Service": "spotfleet.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:ec2:us-east-1:111122223333:spot-fleet-
request/sfr-*"
      },
      "StringEquals": {
        "aws:SourceAccount": "111122223333"
      }
    }
  }
}
```

6. [ポリシーの更新] を選択してください。

次の表に、aws-ec2-spot-fleet-tagging-role の範囲を制限するために想定される aws:SourceArn の値を、その特異性の様々なレベルについてまとめました。

API オペレーション:	呼び出されたサービス	スコープ	**aws:SourceArn**
RequestSpotFleet	AWS STS (AssumeRole)	aws-ec2-spot-fleet-tagging-role が持つ AssumeRole の機能を、指定されたアカウントの spot-flee	arn:aws:ec2:*: *123456789012* :spot-fleet-request/sfr-*

API オペレーション:	呼び出されたサービス	スコープ	**aws:SourceArn**
		t-requests に制限します。	
RequestSpotFleet	AWS STS (AssumeRole)	aws-ec2-spot-fleet-tagging-role が持つ AssumeRole の機能を、指定されたアカウントおよび指定されたリージョンの spot-fleet-requests に制限します。このロールは他のリージョンでは使用できないことに注意してください。	arn:aws:ec2: *us-east-1* :*123456789012* :spot-fleet-request/sfr-*

API オペレーション:	呼び出されたサービス	スコープ	**aws:SourceArn**
RequestSpotFleet	AWS STS (AssumeRole)	aws-ec2-spot-fleet-tagging-role が持つ AssumeRole の機能を、フリート sfr-11111111-1111-1111-1111-111111111111 に影響を与えるアクションのみに制限します。このロールは、他のスポットフリートでは使用できない場合があることに注意してください。また、このロールを使用して request-spot-fleet により新しいスポットフリートを起動することはできません。	arn:aws:ec2: *us-east-1* :*123456789012* :spot-fleet-request/sfr-*11111111-1111-1111-1111-111111111111*

スポットフリートを作成する

> **Important**
>
> スポットフリートは、計画された投資なしでレガシー API を使用します。代わりに EC2 フリートまたは Auto Scaling グループを使用することをお勧めします。詳細については「フリートにどの方法を使用するのが最適か」を参照してください。

AWS マネジメントコンソールを使用すると、AMI と、必要な合計のターゲットキャパシティのみを選択して、スポットフリートリクエストをすばやく作成することができます。Amazon EC2 は、

ユーザーのニーズに最も適したフリートを設定し、スポットのベストプラクティスに従います。あるいは、デフォルト設定のいずれかを変更することができます。

フリートにオンデマンドインスタンスを含める場合は、リクエストで起動テンプレートを指定し、希望するオンデマンドキャパシティを指定する必要があります。

フリートは、キャパシティが使用可能であるときはオンデマンドインスタンスを起動し、最大料金がスポット料金を超えていてキャパシティが使用可能であるときはスポットインスタンスを起動します。

フリートにスポットインスタンスが含まれ、タイプが `maintain` である場合、Amazon EC2 はスポットインスタンスが中断したときにフリートのターゲットキャパシティを維持しようとします。

必要なアクセス許可

詳細については「the section called “スポットフリートアクセス許可”」を参照してください。

タスク

- スポットフリートリクエストをすばやく作成する
- 定義済みパラメータを使用してスポットフリートリクエストを作成する
- 異常なスポットインスタンスを置き換えるスポットフリートを作成する

スポットフリートリクエストをすばやく作成する

Amazon EC2 でスポットフリートリクエストをすばやく作成するには、次の手順に従います。

推奨設定を使用してスポットフリートリクエストを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. スポットを初めて使用する場合は、ウェルカムページが表示されるので、そこで [Get started] を選択してください。それ以外の場合は、[スポットフリートリクエストの作成] を選択します。
4. [Launch parameters] (起動パラメータ) で、[Manually configure launch parameters] (起動パラメータを手動で構成する) を選択してください。
5. AMI で、AMI を選択してください。
6. [Target capacity] (ターゲット容量) の下の [Total target capacity] (総ターゲット容量) で、リクエストする単位数を指定します。ユニットのタイプには、[インスタンス]、[vCPU]、または [メモリ (MiB)] を選択できます。

7. [フリートリクエストの概要] で、フリートの設定を確認し、[起動] を選択してください。

定義済みパラメータを使用してスポットフリートリクエストを作成する

定義済みパラメータを使用して、スポットフリートを作成できます。

Console

定義済みパラメータを使用してスポットフリートリクエストを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. スポットを初めて使用する場合は、ウェルカムページが表示されるので、そこで [Get started] を選択してください。それ以外の場合は、[スポットフリートリクエストの作成] を選択します。
4. [起動パラメータ] では、以下のように、起動パラメータを手動で設定することも、起動テンプレートを使用することもできます。

   a. [手動設定] Amazon EC2 コンソールで起動パラメータを定義するには、[起動パラメータを手動で構成する] を選択し、以下の手順を実行します。

      i. [AMI] で、AWS が提供する基本 AMI のいずれかを選択してください。あるいは、[Search for AMI] (AMI を検索) をクリックして、ユーザーコミュニティの AMI、AWS Marketplace、または独自の AMI を選択することも可能です。

      > **Note**
      >
      > 起動パラメータで指定された AMI が登録解除または無効になっている場合、AMI から新しいインスタンスを起動することはできません。ターゲット容量を維持するように設定されたフリートの場合、ターゲット容量は維持されません。

      ii. (オプション) [Key pair name] で、既存のキーペアを使用するか、新しいキーペアを作成するかを選択してください。

      [既存のキーペア] キーペアを選択してください。

[新しいキーペア] [新しいキーペアの作成] を選択して [キーペア] ページに進みます。完了したら、[Spot Requests] (スポットリクエスト) ページに戻ってリストを更新します。

iii. (オプション) [Additional launch parameters] (追加の起動パラメータ) を展開し、次の操作を実行します。

A. (オプション) Amazon EBS 最適化を有効にするには、[EBS-optimized] (EBS に最適化された) で [Launch EBS-optimized instances] (EBS に最適化されたインスタンスの起動) を選択してください。

B. (オプション) インスタンス用の一時ブロックレベルストレージを追加するには、[Instance store] (インスタンスストア) で [Attach at launch] (起動時にアタッチ) を選択してください。

C. (オプション) ストレージを追加するには、[Add new volume] (新しいボリュームの追加) を選択し、インスタンスタイプに応じて追加のインスタンスストアボリュームまたは Amazon EBS ボリュームを指定します。

D. (オプション) デフォルトでは、インスタンスに対して基本モニタリングが有効になります。詳細モニタリングを有効にするには、[Monitoring] (モニタリング) で [Enable CloudWatch detailed monitoring] (CloudWatch 詳細モニタリングの有効化) を選択してください。

E. (オプション) 専有スポットインスタンスを実行するには、[Tenancy] (テナンシー) で [Dedicated - run a dedicated instance] (専有 - 専有インスタンスの実行) を選択してください。

F. (オプション) [Security groups] で、1 つ以上のセキュリティグループを選択するか、新しいセキュリティグループを作成します。

[既存のセキュリティグループ] 1 つ以上のセキュリティグループを選択してください。

[新しいセキュリティグループ] [Create new security group] (新しいセキュリティグループの作成) を選択し、[Security Groups] (セキュリティグループ) ページに移動します。完了したら、[Spot Requests] (スポットリクエスト) に戻ってリストを更新します。

G. (オプション) インスタンスにインターネットからアクセスできるようにするには、[Auto-assign IPv4 Public IP] (IPv4 パブリック IP の自動割り当て) で [Enable] (有効化) を選択してください。

H. (オプション) IAM ロールを指定して スポットインスタンス を起動するには、[IAM instance profile] でロールを選択してください。

I. (オプション) 起動スクリプトを実行するには、スクリプトを [User data] (ユーザーデータ) にコピーします。

J. (オプション) タグを追加するには、[Create tag] (タグの作成) を選択し、タグのキーと値を入力してから [Create] (作成) を選択してください。各タグについて、これを繰り返します。

タグごとに、インスタンスとスポットフリートリクエストに同じタグを付けるには、[Instances] と [Fleet] (フリート) の両方が選択されていることを確認します。フリートによって起動されたインスタンスのみにタグ付けするには、[Fleet] (フリート) をクリアします。スポットフリートリクエストのみにタグ付けするには、[Istances] をクリアします。

b. [起動テンプレート] 起動テンプレートで作成した設定を使用するには、[起動テンプレートの使用] を選択し、[起動テンプレート] で起動テンプレートを選択してください。

> **Note**
>
> スポットフリートでオンデマンドキャパシティが必要な場合は、起動テンプレートを指定する必要があります。

5. [Additional request details] で、以下を実行します。

a. 追加リクエストの詳細を確認します。変更するには、[Apply defaults] をオフにします。

b. (オプション) [IAM fleet role] で、デフォルトのロールを使用するか、または別のロールを選択できます。ロールの変更後にデフォルトのロールを使用するには、[Use default role] を選択してください。

c. (オプション) 特定の期間中のみ有効なリクエストを作成するには、[Request valid from] および [Request valid until] を編集します。

d. (オプション) デフォルトでは、スポットフリートリクエストの有効期限が切れると Amazon EC2 がスポットインスタンスを終了します。リクエストの有効期限が切れた後も実行し続ける場合、[Terminate the instances when the request expires] をオフにします。

e. (オプション) ロードバランサーを使用する スポットインスタンス を登録するには、[Receive traffic from one or more load balancers] を選択して、1 つ以上の クラシックロードバランサー またはターゲットグループを選択してください。

6. [Target capacity] (ターゲット容量) で、以下の操作を実行します。

   a. [Total target capacity] (総ターゲット容量) で、ターゲット容量にリクエストする単位数を指定します。ユニットのタイプには、[Instances] (ユニット)、[vCPU]、または [Memory (MiB)] (メモリ (MiB)) を選択できます。ターゲットキャパシティを 0 に指定して後でキャパシティを追加できるようにするには、[ターゲット容量を維持する] を選択する必要があります。

   b. (オプション) [Include On-Demand base capacity] (オンデマンドベースの容量を含める) で、リクエストするオンデマンド単位数を指定します。数値は [Total target capacity] (ターゲットキャパシティの合計) 未満にする必要があります。Amazon EC2 は差分を計算し、この差をリクエストするスポット単位に割り当てます。

   > **Important**
   >
   > オプションのオンデマンド容量を指定する場合、最初に起動テンプレートを選択する必要があります。

   c. (オプション) デフォルトでは、Amazon EC2 は中断されるとスポットインスタンスを削除します。ターゲット容量を維持するには、[ターゲット容量を維持する] を選択してください。これで、中断時に Amazon EC2 がスポットインスタンスを終了、停止、または休止するように指定できます。これを行うには、[Interruption behavior] から対応するオプションを選択してください。

   > **Note**
   >
   > 起動パラメータで指定された AMI が登録解除または無効になっている場合、AMI から新しいインスタンスを起動することはできません。この場合、ターゲットキャパシティを維持するようにフリートが設定されていても、ターゲットキャパシティは維持されません。

   d. (オプション) フリートの既存スポットインスタンスにインスタンスの再調整の通知が発行されたときに、スポットフリートが代替スポットインスタンスを起動できるようにするには、[Capacity rebalance] (容量の再調整) を選択し、インスタンス置換戦略を選択してください。[終了前に起動] を選択した場合、Amazon EC2 が古いインスタンスを終了させるまでの遅延時間 (秒単位) を指定します。詳細については「EC2 フリートとスポットフリートでキャパシティの再調整を使用して、リスクのあるスポットインスタンスを置き換える」を参照してください。

e. (オプション) フリートのすべての スポットインスタンスに対して 1 時間あたりに支払う金額を制御するには、[Set maximum cost for Spot Instances] (スポットインスタンスの上限価格を設定する) を選択し、1 時間あたりに支払うことができる上限の合計金額を入力してください。上限の合計金額に達すると、ターゲット容量に満たない場合でも、スポットフリートはスポットインスタンスの起動を停止します。詳細については「EC2 フリートまたはスポットフリートの支出制限を設定する」を参照してください。

7. [Network] (ネットワーク) で、以下の操作を実行します。

   a. [Network] (ネットワーク) で既存の VPC を選択するか、新しい VPC を作成します。

      [既存の VPC] VPC を選択してください。

      [新しい VPC] [新しい VPC の作成] を選択して Amazon VPC コンソールにアクセスします。完了したら、この画面に戻ってリストを更新します。

   b. (オプション) [アベイラビリティーゾーン] では、自分のアベイラビリティーゾーンに適したスポットインスタンスを Amazon EC2 に選択させるか、自分で 1 つ以上のアベイラビリティーゾーンを指定します。

      アベイラビリティーゾーンに複数のサブネットがある場合、[Subnet] から適切なサブネットを選択してください。サブネットを追加するには、[Create new subnet] を選択して Amazon VPC にアクセスします。完了したら、この画面に戻ってリストを更新します。

8. [Instance type requirements] (インスタンスタイプの要件) では、インスタンス属性を指定して、Amazon EC2 にこれらの属性を持つ最適なインスタンスタイプを識別させるか、またはインスタンスのリストを指定することができます。詳細については「EC2 フリートまたはスポットフリートのインスタンスタイプを選択するための属性を指定する」を参照してください。

   a. [Specify instance attributes that match your compute requirements] (コンピューティング要件に一致するインスタンス属性を指定する) を選択した場合、インスタンス属性を次のように指定します。

      i. [vCPUs] に、希望する vCPU の最小数と最大数を入力してください。制限なしを指定するには、[最小値なし]、[最大値なし]、またはその両方を選択してください。

      ii. [Memory (GiB)] (メモリ (GiB)) に、希望する最小値と最大値を入力してください。制限なしを指定するには、[最小値なし]、[最大値なし]、またはその両方を選択してください。

iii. (オプション) [追加のインスタンス属性] では、必要に応じて 1 つ以上の属性を指定して、コンピューティング要件を詳細に表現できます。追加の属性はリクエストにさらに制約を追加します。追加の属性は省略できます。省略すると、デフォルト値が使用されます。各属性およびそのデフォルト値の説明については、Amazon EC2 コマンドラインリファレンスの get-spot-placement-scores.を参照してください。

iv. (オプション) 指定した属性を持つインスタンスタイプを表示するには[Preview matching instance types] (一致するインスタンスタイプをプレビューする) を展開します。インスタンスタイプがリクエストで使用されないようにするには、インスタンスを選択し、[Exclude selected instance types] (選択したインスタンスタイプを除外する) を選択してください。

b. [Manually select instance types] (インスタンスタイプを手動で選択する) を選択すると、スポットフリートはインスタンスタイプのデフォルトのリストを提供します。さらにインスタンスタイプを選択するには、[Add instance types] (インスタンスタイプの追加) を選択し、リクエストで使用するインスタンスタイプを選択してから [Select] (選択) を選択してください。インスタンスタイプを削除するには、インスタンスタイプを選択し、[Delete] (削除) を選択してください。

9. [配分戦略] では、ニーズを満たすスポット配分戦略とオンデマンド配分戦略を選択してください。詳細については「配分戦略を使用して、EC2 フリートまたはスポットフリートがどのようにスポットキャパシティとオンデマンドキャパシティを満たすかを決定します。」を参照してください。

10. [Your fleet request at a glance] (フリートリクエストの概要) で、フリートの設定を確認し、必要な調整を行います。

11. (オプション) AWS CLI で使用される起動設定のコピーをダウンロードするには、[JSON config] (JSON 設定) を選択してください。

12. スポットフリートを起動する準備が整ったら、[起動] を選択してください。

    スポットフリートリクエストタイプは `fleet` です。リクエストが実行されると、タイプ `instance` のリクエストが追加されます。このとき、状態は `active` になり、ステータスは `fulfilled` になります。

AWS CLI

スポットフリートリクエストを作成するには

request-spot-fleet コマンドを使用します。

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://config.json
```

設定ファイルの例については、「スポットフリートの CLI 設定の例」を参照してください。

PowerShell

スポットフリートリクエストを作成するには

Request-EC2SpotFleet コマンドレットを使用します。次の例では、キャパシティ最適化フリートでスポットインスタンスを起動します。

```
Request-EC2SpotFleet `
    -SpotFleetRequestConfig_TargetCapacity 50 `
    -SpotFleetRequestConfig_AllocationStrategy "CapacityOptimized" `
    -SpotFleetRequestConfig_IamFleetRole "arn:aws:iam::123456789012:role/my-spot-fleet-role" `
    -SpotFleetRequestConfig_LaunchTemplateConfig @($launchConfig)
```

起動設定を次のように定義し、起動テンプレートを設定し、必要なプロパティを上書きします。設定例についてはスポットフリートの CLI 設定の例を参照してください。

```
$lcSpec = Amazon.EC2.Model.FleetLaunchTemplateSpecification
# To do - Set FleetLaunchTemplateSpecification properties
$lcOverrides = New-Object Amazon.EC2.Model.LaunchTemplateOverrides
# To do - Set LaunchTemplateOverrides properties
$launchConfig = New-Object Amazon.EC2.Model.LaunchTemplateConfig
$launchConfig.LaunchTemplateSpecification $lcSpec
$launchConfig.Overrides @($lcOverrides)
```

異常なスポットインスタンスを置き換えるスポットフリートを作成する

スポットフリートは、2 分ごとにフリートのスポットインスタンスのヘルスステータスをチェックします。インスタンスのヘルスステータスは healthy または unhealthy です。

スポットフリートは、Amazon EC2 が提供するステータスチェックを使用して、インスタンスのヘルスステータスを判断します。インスタンスステータスとシステムステータスのいずれかのチェック結果において、ステータスが 3 回連続して impaired を示した場合、そのインスタンスは unhealthy と判断されます。詳細については「Amazon EC2 インスタンスのステータスチェック」を参照してください。

フリートを設定して、異常のある スポットインスタンス を置き換えることができます。ヘルスチェックによる置き換えを有効化すると、 と報告されたスポットインスタンスが置き換えられます。unhealthy異常なスポットインスタンスの置き換え中、最大数分間フリートがターゲット容量を下回る場合があります。

要件

- ヘルスチェックによる置き換えは、1 回限りの スポットフリート (maintain のフリート) ではなく、ターゲットキャパシティを維持しているスポットフリート (タイプ request のフリート) でのみサポートされます。
- ヘルスチェックによる置き換えは、スポットインスタンス でのみサポートされます。この機能はオンデマンドインスタンス ではサポートされていません。
- 作成時のみ、異常なインスタンスを置き換えるようスポットフリートを設定できます。
- ユーザーは、ec2:DescribeInstanceStatus アクションを呼び出す許可を持っている場合のみ、ヘルスチェックの置き換えを使用できます。

Console

異常なスポットインスタンスは置き換えるようにスポットフリートを設定するには

1. 定義済みパラメータを使用してスポットフリートリクエストを作成する の手順に従ってスポットフリートを作成します。
2. 異常なスポットインスタンスを置き換えるようにフリートを設定するには、[追加の起動パラメータ] を展開し、[ヘルスチェック] で、[異常なインスタンスを置き換える] を選択してください。このオプションを有効にするには、まず [Maintain target capacity](ターゲット容量の維持) を選択する必要があります。

AWS CLI

異常なスポットインスタンスは置き換えるようにスポットフリートを設定するには

request-spot-fleet コマンドを SpotFleetRequestConfig の ReplaceUnhealthyInstances プロパティと共に使用します。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "lowestPrice",
```

```
        "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-
role",
        "TargetCapacity": 10,
        "ReplaceUnhealthyInstances": true
    }
}
```

PowerShell

異常なスポットインスタンスは置き換えるようにスポットフリートリクエストを設定するには

Request-EC2SpotFleet コマンドレットを -SpotFleetRequestConfig_ReplaceUnhealthyInstance パラメータと共に使用します。

```
-SpotFleetRequestConfig_ReplaceUnhealthyInstance $true
```

新しいまたは既存のスポットフリートリクエスト、およびそのフリートが起動するインスタンスとボリュームにタグ付けする

> **Important**
>
> スポットフリートは、計画された投資なしでレガシー API を使用します。代わりに EC2 フリートまたは Auto Scaling グループを使用することをお勧めします。詳細については「フリートにどの方法を使用するのが最適か」を参照してください。

スポットフリートリクエスト、およびそのフリートが起動するインスタンスとボリュームを分類および管理しやすくするために、カスタムメタデータでタグ付けできます。スポットフリートリクエストへのタグの割り当ては、リクエストの作成時または作成後に行うことができます。同様に、インスタンスとボリュームにも、フリートによって起動されるときまたは起動後に、タグを割り当てることができます。

フリートリクエストにタグを付けると、フリートによって起動されるインスタンスとボリュームには自動的にタグ付けされません。フリートによって起動されるインスタンスとボリュームには、明示的にタグを付ける必要があります。タグは、フリートリクエストのみに割り当てるか、フリートによって起動されたインスタンスのみに割り当てるか、フリートによって起動されたインスタンスにアタッチされたボリュームのみに割り当てるか、またはそのすべてに割り当てるかを選択できます。

> **Note**
>
> オンデマンドインスタンスにアタッチされているボリュームにのみタグ付けできます。スポットインスタンス にアタッチされているボリュームにタグを付けることはできません。

Amazon EC2 コンソールまたはコマンドラインツールを使用してタグを割り当てることができます。

タグの仕組みの詳細については、「Amazon EC2 リソースのタグ付け」を参照してください。

内容

- 前提条件
- 新しいスポットフリート、およびそれが起動するインスタンスおよびボリュームにタグ付けします。
- 既存のスポットフリートにタグを付けます。
- スポットフリートリクエストタグを表示する

前提条件

リソースにタグ付けする許可をユーザーに付与します。詳細については「例: リソースのタグ付け」を参照してください。

リソースにタグ付けする許可をユーザーに付与するには

以下を含む IAM ポリシーを作成します。

- `ec2:CreateTags` アクション。これにより、タグを作成する許可がユーザーに付与されます。
- `ec2:RequestSpotFleet` アクション。これにより、スポットフリートリクエストを作成する許可がユーザーに付与されます。
- `Resource` で、`"*"` を指定する必要があります。これにより、ユーザーはすべてのリソースタイプにタグ付けできます。

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "TagSpotFleetRequest",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags",
                "ec2:RequestSpotFleet"
            ],
            "Resource": "*"
        }
    ]
}
```

Important

現在、spot-fleet-request リソースに対するリソースレベルのアクセス許可はサポートされていません。リソースとして spot-fleet-request を指定した場合、フリートにタグ付けしようとすると、不正な例外が発生します。以下の例は、ポリシーを設定 しない 方法を示しています。

```
{
    "Effect": "Allow",
    "Action": [
        "ec2:CreateTags",
        "ec2:RequestSpotFleet"
    ],
    "Resource": "arn:aws:ec2:us-east-1:111122223333:spot-fleet-request/*"
}
```

アクセス許可を付与するにはユーザー、グループ、またはロールにアクセス許可を追加します。

- AWS IAM アイデンティティセンター のユーザーとグループ:

  アクセス許可セットを作成します。「AWS IAM アイデンティティセンター ユーザーガイド」の「アクセス許可セットを作成する」の手順に従ってください。

- IAM 内で、ID プロバイダーによって管理されているユーザー:

ID フェデレーションのロールを作成します。詳細については IAM ユーザーガイド の サードパーティー ID プロバイダー (フェデレーション) 用のロールを作成する を参照してください。

- IAM ユーザー:
  - ユーザーが担当できるロールを作成します。手順については IAM ユーザーガイド の IAM ユーザーのロールの作成 を参照してください。
  - (お奨めできない方法) ポリシーをユーザーに直接アタッチするか、ユーザーをユーザーグループに追加します。「IAM ユーザーガイド」の「ユーザー (コンソール) へのアクセス許可の追加」の指示に従います。

新しいスポットフリート、およびそれが起動するインスタンスおよびボリュームにタグ付けします。

新しいスポットフリートリクエスト、およびそのフリートがコンソールを使用して起動するインスタンスとボリュームにタグ付けするには

1. 定義済みパラメータを使用してスポットフリートリクエストを作成するの手順に従います。
2. タグを追加する方法は、フリートを手動で設定したか、起動テンプレートを使用したかによって異なります。

   - フリートを手動で設定した場合は、次の操作を行います。

     タグを追加するには、[追加の起動パラメータ] を展開し、[タグの作成] を選択して、タグのキーと値を入力してください。各タグについて、これを繰り返します。

     タグごとに、スポットフリートリクエストとインスタンスに同じタグを付けることができます。両方にタグを付けるには、[インスタンス] と [フリート] の両方が選択されていることを確認します。スポットフリートリクエストのみにタグ付けするには、[Istances] をクリアします。フリートによって起動されたインスタンスのみにタグ付けするには、[フリートタグ]をクリアします。

     > **Note**
     >
     > フリートを手動で設定する場合、ボリュームにタグを付けるオプションはありません。ボリュームタグは、オンデマンドインスタンス にアタッチされたボリュームでのみサポートされます。フリートを手動で設定する場合、オンデマンドインスタンスを指定することはできません。

- 起動テンプレートを使用した場合は、次の操作を行います。

  フリートリクエストにタグを追加するには、[タグ] で [タグの作成] を選択し、タグのキーと値を入力してください。各タグについて、これを繰り返します。

  フリート内のリソースにタグを付けるには、起動テンプレートでタグを指定する必要があります。

新しいスポットフリートリクエストと、AWS CLI を使用して起動するインスタンスおよびボリュームにタグ付けするには

作成時にスポットフリートリクエストにタグ付けし、フリートがインスタンスを起動するときにインスタンスおよびボリュームにタグ付けするには、スポットフリートリクエスト設定を次のようにします。

スポットフリートリクエストのタグ

- スポットフリートリクエストのタグを SpotFleetRequestConfig で指定します。
- ResourceType の場合、spot-fleet-request を指定します。別の値を指定すると、フリートリクエストは失敗します。
- Tags で、キーと値のペアを指定します。キーと値のペアは複数指定できます。

インスタンスタグ:

- LaunchSpecifications で、インスタンスのタグを指定します。
- ResourceType の場合、instance を指定します。別の値を指定すると、フリートリクエストは失敗します。
- Tags で、キーと値のペアを指定します。キーと値のペアは複数指定できます。

  または、スポットフリートリクエストで参照される起動テンプレートで、インスタンスのタグを指定できます。???

ボリュームタグ:

- スポットフリートリクエストで参照される起動テンプレートのボリュームのタグを指定します。???LaunchSpecifications でのボリュームのタグ付けはサポートされていません。

以下の例では、スポットフリートリクエストに 2 つのタグ (Key=Environment、Value=Production、および Key=Cost-Center、Value=123) が付けられています。フリートが起動するインスタンスには、1 つのタグ (スポットフリートリクエストのタグの 1 つと同じ) Key=Cost-Center and Value=123 が付けられます。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "priceCapacityOptimized",
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::111122223333:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchSpecifications": [
            {
                "ImageId": "ami-0123456789EXAMPLE",
                "InstanceType": "c4.large",
                "TagSpecifications": [
                    {
                        "ResourceType": "instance",
                        "Tags": [
                            {
                                "Key": "Cost-Center",
                                "Value": "123"
                            }
                        ]
                    }
                ]
            }
        ],
        "SpotPrice": "5",
        "TargetCapacity": 2,
        "TerminateInstancesWithExpiration": true,
        "Type": "maintain",
        "ReplaceUnhealthyInstances": true,
        "InstanceInterruptionBehavior": "terminate",
        "InstancePoolsToUseCount": 1,
        "TagSpecifications": [
            {
                "ResourceType": "spot-fleet-request",
                "Tags": [
                    {
                        "Key": "Environment",
                        "Value":"Production"
                    },
```

```
                    {
                        "Key": "Cost-Center",
                        "Value":"123"
                    }
                ]
            }
        ]
    }
}
```

AWS CLI を使用して、スポットフリートが起動したインスタンスにタグ付けするには

フリートがインスタンスを起動するときにインスタンスにタグ付けするには、スポットフリートリクエストで参照される起動テンプレートでタグを指定するか、以下のようにスポットフリートリクエスト設定でタグを指定できます。???

- LaunchSpecifications で、インスタンスのタグを指定します。
- ResourceType の場合、instance を指定します。別の値を指定すると、フリートリクエストは失敗します。
- Tags で、キーと値のペアを指定します。キーと値のペアは複数指定できます。

以下の例では、フリートによって起動されるインスタンスに 1 つのタグ (Key=Cost-Center and Value=123) が付けられています。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "priceCapacityOptimized",
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::111122223333:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchSpecifications": [
            {
                "ImageId": "ami-0123456789EXAMPLE",
                "InstanceType": "c4.large",
                "TagSpecifications": [
                    {
                        "ResourceType": "instance",
                        "Tags": [
                            {
                                "Key": "Cost-Center",
                                "Value": "123"
```

```
                            }
                        ]
                    }
                ]
            }
        ],
        "SpotPrice": "5",
        "TargetCapacity": 2,
        "TerminateInstancesWithExpiration": true,
        "Type": "maintain",
        "ReplaceUnhealthyInstances": true,
        "InstanceInterruptionBehavior": "terminate",
        "InstancePoolsToUseCount": 1
    }
}
```

AWS CLI を使用して、スポットフリートが起動するオンデマンドインスタンスにアタッチされたボリュームにタグ付けするには

フリートが作成したときにボリュームにタグ付けするには、スポットフリートリクエストで参照される起動テンプレートでタグを指定する必要があります。???

> **Note**
>
> ボリュームタグは、オンデマンドインスタンス にアタッチされたボリュームでのみサポートされます。スポットインスタンス にアタッチされているボリュームにタグを付けることはできません。
> `LaunchSpecifications` でのボリュームのタグ付けはサポートされていません。

既存のスポットフリートにタグを付けます。

コンソールを使用して、既存のスポットフリートリクエストにタグ付けするには

スポットフリートリクエストを作成した後、コンソールを使用してフリートリクエストにタグを追加できます。

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. スポットフリートリクエストを選択してください。

4. [Tags (タグ)] タブを選択してから、[タグの作成] を選択してください。

AWS CLI を使用して、既存のスポットフリートリクエストにタグ付けするには

create-tags コマンドを使用して、既存のリソースにタグ付けできます。https://docs.aws.amazon.com/cli/latest/reference/ec2/create-tags.html以下の例では、既存のスポットフリートリクエストにタグ Key=purpose and Value=test が付けられています。

```
aws ec2 create-tags \
    --resources sfr-11112222-3333-4444-5555-66666EXAMPLE \
    --tags Key=purpose,Value=test
```

スポットフリートリクエストタグを表示する

コンソールを使用して、スポットフリートリクエストタグを表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. スポットフリートリクエストを選択し、[タグ] タブを選択してください。

スポットフリートリクエストタグを記述するには

describe-tags コマンドを使用して、指定したリソースのタグを表示します。https://docs.aws.amazon.com/cli/latest/reference/ec2/describe-tags.html以下の例では、指定したスポットフリートリクエストのタグを記述します。

```
aws ec2 describe-tags \
    --filters "Name=resource-id,Values=sfr-11112222-3333-4444-5555-66666EXAMPLE"
```

```
{
    "Tags": [
        {
            "Key": "Environment",
            "ResourceId": "sfr-11112222-3333-4444-5555-66666EXAMPLE",
            "ResourceType": "spot-fleet-request",
            "Value": "Production"
        },
        {
```

```
            "Key": "Another key",
            "ResourceId": "sfr-11112222-3333-4444-5555-66666EXAMPLE",
            "ResourceType": "spot-fleet-request",
            "Value": "Another value"
        }
    ]
}
```

スポットフリートリクエストを記述することで、スポットフリートリクエストのタグを表示することもできます。

describe-spot-fleet-requests コマンドを使用して、指定したスポットフリートリクエストの設定を表示します。これには、フリートリクエストに指定されたタグが含まれます。

```
aws ec2 describe-spot-fleet-requests \
    --spot-fleet-request-ids sfr-11112222-3333-4444-5555-66666EXAMPLE
```

```
{
    "SpotFleetRequestConfigs": [
        {
            "ActivityStatus": "fulfilled",
            "CreateTime": "2020-02-13T02:49:19.709Z",
            "SpotFleetRequestConfig": {
                "AllocationStrategy": "capacityOptimized",
                "OnDemandAllocationStrategy": "lowestPrice",
                "ExcessCapacityTerminationPolicy": "Default",
                "FulfilledCapacity": 2.0,
                "OnDemandFulfilledCapacity": 0.0,
                "IamFleetRole": "arn:aws:iam::111122223333:role/aws-ec2-spot-fleet-
tagging-role",
                "LaunchSpecifications": [
                    {
                        "ImageId": "ami-0123456789EXAMPLE",
                        "InstanceType": "c4.large"
                    }
                ],
                "TargetCapacity": 2,
                "OnDemandTargetCapacity": 0,
                "Type": "maintain",
                "ReplaceUnhealthyInstances": false,
                "InstanceInterruptionBehavior": "terminate"
            },
```

```
            "SpotFleetRequestId": "sfr-11112222-3333-4444-5555-66666EXAMPLE",
            "SpotFleetRequestState": "active",
            "Tags": [
                {
                    "Key": "Environment",
                    "Value": "Production"
                },
                {
                    "Key": "Another key",
                    "Value": "Another value"
                }
            ]
        }
    ]
}
```

スポットフリートリクエスト、そのインスタンス、イベント履歴を記述する

> **Important**
>
> スポットフリートは、計画された投資なしでレガシー API を使用します。代わりに EC2 フリートまたは Auto Scaling グループを使用することをお勧めします。詳細については「フリートにどの方法を使用するのが最適か」を参照してください。

スポットフリートの設定、スポットフリートのインスタンス、スポットフリートのイベント履歴を記述できます。

Console

スポットフリートを記述するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. スポットフリートリクエストを選択してください。ID は、sfr- で始まります。設定の詳細を表示するには、[Description] を選択してください。
4. スポットフリートのスポットインスタンスを一覧表示するには、インスタンスを選択してください。

5. スポットフリートの履歴を表示するには、[履歴] を選択してください。

AWS CLI

スポットフリートリクエストを記述するには

describe-spot-fleet-requests コマンドを使用します。

```
aws ec2 describe-spot-fleet-requests \
    --spot-fleet-request-ids sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

指定したスポットフリートリクエストの実行中のインスタンスを記述するには

describe-spot-fleet-instances コマンドを使用します。

```
aws ec2 describe-spot-fleet-instances \
    --spot-fleet-request-id sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

指定したスポットフリートリクエストのイベント履歴を記述するには

describe-spot-fleet-request-history コマンドを使用します。

```
aws ec2 describe-spot-fleet-request-history \
    --spot-fleet-request-id sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --start-time 2024-05-18T00:00:00Z
```

PowerShell

スポットフリートリクエストを記述するには

Get-EC2SpotFleetRequest コマンドレットを使用します。

```
Get-EC2SpotFleetRequest
```

指定したスポットフリートリクエストの実行中のインスタンスを記述するには

Get-EC2SpotFleetInstance コマンドレットを使用します。

```
Get-EC2SpotFleetInstance `
    -SpotFleetRequestId "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE"
```

指定したスポットフリートリクエストのイベント履歴を記述するには

Get-EC2SpotFleetRequestHistory コマンドレットを使用します。

```
Get-EC2SpotFleetRequestHistory `
    -SpotFleetRequestId "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE" `
    -UtcStartTime 2024-05-18T00:00:00Z
```

スポットフリートリクエストを変更します。

> **Important**
>
> スポットフリートは、計画された投資なしでレガシー API を使用します。代わりに EC2 フリートまたは Auto Scaling グループを使用することをお勧めします。詳細については「フリートにどの方法を使用するのが最適か」を参照してください。

以下のタスクを完了するように、アクティブスポットフリートリクエストを変更できます。

- 合計ターゲットキャパシティとオンデマンド部分を増やす
- 合計ターゲットキャパシティとオンデマンド部分を減らす

合計ターゲットキャパシティを増やすと、スポットフリートはスポットフリートリクエストの配分戦略に従って、追加のスポットインスタンスを起動します。オンデマンド部分を増やすと、スポットフリートは追加のオンデマンドインスタンスを起動します。

合計ターゲットキャパシティを減らすと、スポットフリートは新しいターゲットキャパシティを超えるすべてのオープンリクエストをキャンセルします。フリートのサイズが新しいターゲット容量に達するまで、スポットフリートがスポットインスタンスを終了させるようにリクエストできます。配分戦略が diversified の場合、スポットフリートはプール全体でインスタンスを終了させます。または、スポットフリートが現在のサイズを維持するようにリクエストすることもできますが、中断されたり手動で終了されたスポットインスタンスを置き換えることはできません。

考慮事項

- ワンタイムスポットフリートリクエストは変更できません。スポットフリートリクエストの作成時に [ターゲット容量の維持] を選択した場合にのみ、スポットフリートリクエストを変更することができます。

- ターゲット容量が減ったためにスポットフリートがインスタンスを終了する場合、インスタンスはスポットインスタンスの中断通知を受け取ります。

Console

スポットフリートリクエストを変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. スポットフリートリクエストを選択してください。
4. [Actions]、[Modify target capacity] の順に選択してください。
5. [Modify target capacity] で、以下の操作を実行します。
   a. 新しいターゲット容量とオンデマンド部分を入力してください。
   b. (オプション) ターゲット容量を小さくしてもスポット群の現在のサイズを保持する場合は、[Terminate instances] をオフにします。
   c. [Submit] を選択してください。

AWS CLI

スポットフリートリクエストを変更するには

modify-spot-fleet-requestコマンドを使用して、指定するスポットフリートリクエストのターゲット容量を更新します。https://docs.aws.amazon.com/cli/latest/reference/ec2/modify-spot-fleet-request.html

```
aws ec2 modify-spot-fleet-request \
    --spot-fleet-request-id sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --target-capacity 20
```

前のコマンドを以下のように変更して、結果的にいずれのスポットインスタンスも終了せずに、指定したスポットフリートのターゲット容量を減らすことができます。

```
aws ec2 modify-spot-fleet-request \
    --spot-fleet-request-id sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --target-capacity 10 \
    --excess-capacity-termination-policy NoTermination
```

PowerShell

スポットフリートリクエストを変更するには

Edit-EC2SpotFleetRequest コマンドレットを使用して、指定したスポットフリートリクエストのターゲットキャパシティを更新します。

```
Edit-EC2SpotFleetRequest `
    -SpotFleetRequestId "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE" `
    -TargetCapacity 20
```

前のコマンドを以下のように変更して、結果的にいずれのスポットインスタンスも終了せずに、指定したスポットフリートのターゲット容量を減らすことができます。

```
Edit-EC2SpotFleetRequest `
    -SpotFleetRequestId "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE" `
    -TargetCapacity 20 `
    -ExcessCapacityTerminationPolicy "NoTermination"
```

スポットフリートリクエストをキャンセル (削除) します。

> **Important**
>
> スポットフリートは、計画された投資なしでレガシー API を使用します。代わりに EC2 フリートまたは Auto Scaling グループを使用することをお勧めします。詳細については「フリートにどの方法を使用するのが最適か」を参照してください。

スポットフリートが不要になった場合は、スポットフリートリクエストをキャンセルできます。キャンセルすると、リクエストが削除されます。フリートリクエストをキャンセルすると、フリートに関連付けられているすべてのスポットリクエストがキャンセルされるため、新しいスポットインスタンスは起動されません。

スポットフリートをキャンセルするときは、そのインスタンスをすべて終了させるかどうかも指定する必要があります。これには、オンデマンドインスタンスとスポットインスタンスの両方が含まれます。

> **Warning**
>
> インスタンスの終了は恒久的で、元に戻すことができません。
> インスタンスを終了すると、そのインスタンスには接続できなくなり、復元することもできません。アタッチされている Amazon EBS ボリュームで、終了時に削除されるように設定されているものも、すべて恒久的に削除され、復元できません。インスタンスストアボリュームに保存されているデータは、すべて恒久的に失われます。詳細については「インスタンスの終了方法」を参照してください。
> インスタンスを終了する前に、終了後も保持する必要があるすべてのデータを永続的ストレージにバックアップしたことを確認してください。

フリートリクエストをキャンセルするときにインスタンスを終了する必要があることを指定した場合、フリートリクエストは `cancelled_terminating` 状態へ移行します。それ以外の場合は `cancelled_running` 状態になり、インスタンスは中断または手動終了されるまで、引き続き実行されます。

制限事項

- 1 回のリクエストで最大 100 個のフリートをキャンセルできます。指定した数を超えると、フリートはキャンセルされません。

Console

スポットフリートリクエストをキャンセル (削除) するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. スポットフリートリクエストを選択してください。
4. [アクション]、[リクエストのキャンセル] の順にクリックしてください。
5. [スポットリクエストのキャンセル] ダイアログボックスで、次の操作を行います。
   a. スポットフリートリクエストのキャンセルと同時に関連するインスタンスを終了するには、[インスタンスの終了] チェックボックスをオンのままにします。関連するインスタンスを終了せずにスポットフリートリクエストをキャンセルするには、[インスタンスの終了] チェックボックスをオフにします。
   b. [確認] を選択してください。

AWS CLI

スポットフリートリクエストをキャンセル (削除) し、そのインスタンスを終了するには

cancel-spot-fleet-requests コマンドを --terminate-instances オプションと共に使用します。

```
aws ec2 cancel-spot-fleet-requests \
    --spot-fleet-request-ids sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --terminate-instances
```

以下は出力の例です。

```
{
    "SuccessfulFleetRequests": [
        {
            "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
            "CurrentSpotFleetRequestState": "cancelled_terminating",
            "PreviousSpotFleetRequestState": "active"
        }
    ],
    "UnsuccessfulFleetRequests": []
}
```

そのインスタンスを終了せずにスポットフリートリクエストをキャンセル (削除) するには

代わりに --no-terminate-instances オプションを使用して、前の例を変更します。

```
aws ec2 cancel-spot-fleet-requests \
    --spot-fleet-request-ids sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --no-terminate-instances
```

以下は出力の例です。

```
{
    "SuccessfulFleetRequests": [
        {
            "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
            "CurrentSpotFleetRequestState": "cancelled_running",
            "PreviousSpotFleetRequestState": "active"
        }
```

```
    ],
    "UnsuccessfulFleetRequests": []
}
```

PowerShell

スポットフリートリクエストをキャンセル (削除) し、そのインスタンスを終了するには

Stop-EC2SpotFleetRequest コマンドレットを -TerminateInstance パラメータと共に使用します。

```
Stop-EC2SpotFleetRequest `
    -SpotFleetRequestId "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE" `
    -TerminateInstance $true
```

そのインスタンスを終了せずにスポットフリートリクエストをキャンセル (削除) するには

-TerminateInstance パラメータの値を変更して、前の例を変更します。

```
Stop-EC2SpotFleetRequest `
    -SpotFleetRequestId "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE" `
    -TerminateInstance $false
```

スポットフリートの自動スケーリングについて

> **Important**
>
> スポットフリートは、計画された投資なしでレガシー API を使用します。代わりに EC2 フリートまたは Auto Scaling グループを使用することをお勧めします。詳細については「フリートにどの方法を使用するのが最適か」を参照してください。

自動スケーリングにより、需要に応じてスポットフリートのターゲットキャパシティを増減できます。自動スケーリングを使用すると、スポットフリートは、1 つ以上のスケーリングポリシーに応答して、指定された範囲内でインスタンスを起動 (スケールアウト) するか、インスタンスを終了 (スケールイン) できます。

スポットフリートの自動スケーリングは、Amazon EC2、Amazon CloudWatch、および Application Auto Scaling API の組み合わせによって可能になります。スポットフリートは Amazon EC2 で作成

され、アラームは CloudWatch で作成され、スケーリングポリシーは Application Auto Scaling で作成されます。

自動スケーリングのタイプ

スポットフリートは、以下のタイプの自動スケーリングをサポートします。

- ターゲット追跡スケーリング – 特定のメトリクスの値をターゲットにすることで、フリートの現在のキャパシティを増減させます。これはサーモスタットで家の温度を管理する方法と似ています (希望の温度を選択すれば、後はサーモスタットがすべてを実行します)。
- ステップスケーリング - アラーム違反のサイズに応じて変動する一連のスケーリング調整値 (ステップ調整値) に基づいて、フリートの現在の容量を増減させます。???
- スケジュールに基づくスケーリング - 日付と時刻に基づいて、フリートの現在の容量を増減させます。

考慮事項

スポットフリートに自動スケーリングを使用する場合は、次の点を考慮してください。

- インスタンスの重み付け – インスタンスの重み付けを使用している場合は、スポットフリートが必要に応じてターゲットキャパシティを超える可能性があることに注意してください。取得済みの容量が浮動小数点数となってもターゲット容量は整数でなければならないため、スポットフリートはその数を次の整数に切り上げます。アラームがトリガーされたときにスケーリングポリシーの結果を確認する際は、このような動作を考慮に入れる必要があります。例えば、ターゲット容量が 30、取得済み容量が 30.1 で、スケーリングポリシーが 1 を減算するとします。アラームがトリガーされると、自動スケーリングプロセスは 30.1 から 1 を減算して 29.1 を得るため、この数は 30 に切り上げられることになり、スケーリングアクションは実行されません。別の例として、選択したインスタンスの分量が 2、4、8 であり、ターゲット容量が 10 であるとします。分量 2 のインスタンスが利用できなかったために、スポットフリートは分量 4 と 8 のインスタンスをプロビジョニングして取得済みの容量が 12 になったとします。スケーリングポリシーがターゲット容量を 20% 減らしてアラームがトリガーされた場合、自動スケーリングプロセスは 12 から 12×0.2 を減算して 9.6 を得るため、この数は 10 に切り上げられることになり、スケーリングアクションは実行されません。
- クールダウン期間 - スポットフリート用に作成したスケーリングポリシーは、クールダウン期間をサポートしています。クールダウン期間は、以前のトリガー関連のスケーリングアクティビティが以後のスケーリングイベントに影響を及ぼすことができる期限であり、スケーリングアクティビティが終了した時点からの秒数として指定します。スケールアウトポリシーにクールダウン期間を

設定すると、その期間中にクールダウンを開始したスケールアウトイベントによって追加された容量は、次のスケールアウトに予定される容量の一部として繰り入れられます。これにより、スケールアウトが継続的に (ただし過剰になることなく) 行われます。スケールインポリシーにクールダウン期間を設定すると、その期間が過ぎるまでは以後のスケールインリクエストがブロックされます。これにより、スケールインが抑制されてアプリケーションの可用性が確保されます。ただし、スケールイン後のクールダウン期間中に別のアラームによってスケールアウトポリシーがトリガーされると、自動スケーリングによってスケーラブルなターゲットが即座にスケールアウトされます。

- 詳細モニタリングの使用 – 使用率の変化により迅速に対応できるように、1 分間隔でインスタンスメトリクスに基づいてスケールすることをお勧めします。5 分間隔でメトリクスをスケールすると、応答時間が低速になり、古いメトリクスデータに基づいてスケールすることになる可能性があります。1 分ごとにインスタンスのメトリクスデータを CloudWatch に送信するには、インスタンスで詳細モニタリングを有効にできます。詳細については、「EC2 インスタンスの詳細モニタリングを管理する」および「定義済みパラメータを使用してスポットフリートリクエストを作成する」を参照してください。
- AWS CLI – AWS CLI を使用してスポットフリートのスケーリングを設定する場合は、application-autoscaling CLI を使用します。

スポットフリートの自動スケーリングに必要な IAM のアクセス許可

スポットフリートの自動スケーリングは、Amazon EC2、Amazon CloudWatch、および Application Auto Scaling API の組み合わせによって可能になります。スポットフリートは Amazon EC2 で作成され、アラームは CloudWatch で作成され、スケーリングポリシーは Application Auto Scaling で作成されます。Amazon EC2 とスポットフリートの使用に必要な IAM アクセス許可に加えて、フリートスケーリング設定にアクセスするユーザーは、自動スケーリングをサポートするサービスに対する適切なアクセス許可が必要です。

スポットフリートの自動スケーリングを使用するには、ユーザーは次のポリシー例に示されているアクションを使用するアクセス許可を持っている必要があります。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "application-autoscaling:*",
                "ec2:DescribeSpotFleetRequests",
                "ec2:ModifySpotFleetRequest",
                "cloudwatch:DeleteAlarms",
                "cloudwatch:DescribeAlarmHistory",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:DescribeAlarmsForMetric",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics",
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DisableAlarmActions",
                "cloudwatch:EnableAlarmActions",
                "iam:CreateServiceLinkedRole",
                "sns:CreateTopic",
                "sns:Subscribe",
                "sns:Get*",
                "sns:List*"
            ],
            "Resource": "*"
        }
    ]
}
```

独自の IAM ポリシーを作成し、アプリケーションの Auto Scaling API を呼び出すためのよりきめ細かなアクセス許可を付与することもできます。詳細については、「Application Auto Scaling ユーザーガイド」の「Identity and Access Management for Application Auto Scaling」を参照してください。

Application Auto Scaling サービスには、スポットフリートおよび CloudWatch アラームを記述するアクセス許可、およびユーザーの代わりにスポットフリートターゲット容量を変更するアクセス許可も必要です。スポットフリートの自動スケーリングを有効にすると、サービスにリンクされた `AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest` というロールが作成されます。このサービスにリンクされたロールは、アプリケーションの Auto Scaling に対して、ポリシーのアラームの記述、フリートの現容量のモニタリング、およびフリートの容量の変更を行うためのアクセス許可を付与します。Application Auto Scaling の元の管理された型のスポットフリートロールは `aws-ec2-spot-fleet-autoscale-role` ですが、これは不要になりました。サービスにリンクされたロールは、Application Auto Scaling のデフォルトロールです。詳細については、「Application Auto Scaling ユーザーガイド」の「Application Auto Scaling 用のサービスリンクロール」を参照してください。

ターゲット追跡スケーリング: 特定のメトリクスの値を対象にしてスポットフリートをスケールする

ターゲット追跡スケーリングで、メトリクスを選択し、ターゲット値を設定することで、ターゲット追跡スケーリングポリシーを作成します。次に、スポットフリートは、スケーリングポリシーをトリガーする CloudWatch アラームを作成および管理し、選択したメトリクスとターゲット値に基づいてスケーリング調整を計算します。スケーリングポリシーは、必要に応じてインスタンスを追加または削除することでキャパシティを調整し、メトリクスを指定されたターゲット値またはそれに近い値に維持します。ターゲット追跡ポリシーは、メトリクスをターゲット値近くに維持することに加えて、負荷パターンの変動によるメトリクスの変動に合わせて調整し、急激なキャパシティの変動を最小化します。

各ポリシーが異なるメトリクスを使用していれば、スポットフリートに複数のターゲットの追跡スケーリングポリシーを作成できます。フリートは、最大のフリートキャパシティを指定するポリシーに基づいてスケールされます。これにより、複数のシナリオに対応して、アプリケーションワークロードに十分なキャパシティを確保できます。

アプリケーションの可用性を高めるために、フリートのスケールアウトはメトリクスに比例して可能な限り高速に行われますが、スケールインはより緩やかです。

ターゲットキャパシティが減ったためにスポットフリートがスポットインスタンスを終了する場合、インスタンスはスポットインスタンスの中断通知を受け取ります。

> **Note**
>
> ターゲットの追跡スケーリングポリシーのためにスポットフリートが管理する CloudWatch アラームを編集または削除しないでください。ターゲット追跡スケーリングポリシーを削除すると、スポットフリートが自動的にアラームを削除します。

前提条件

- スポットフリートリクエストには、タイプが `maintain` のリクエストが必要です。自動スケーリングはタイプ `request` のリクエストではサポートされていません。
- スポットフリートの自動スケーリングに必要な IAM のアクセス許可 を設定します。
- 「考慮事項」を確認します。

ターゲット追跡ポリシーを設定するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. スポットフリートリクエストを選択してください。
4. 画面の下部にある [自動スケーリング] タブを選択してください。スポットフリートのリンクを選択した場合、タブはありません。代わりに、[自動スケーリング] セクションまでスクロールします。
5. 自動スケーリングが設定されていない場合は、[Configure] を選択してください。
6. スポットフリートの最小容量および最大容量を設定するには、[Scale capacity between] を使用します。自動スケーリングにより、最小容量以下または最大容量以上にスポットフリートがスケールされることはありません。
7. [Policy Name] にこのポリシーの名前を入力してください。
8. [Target metric] を選択します。
9. メトリクスの [Target value] を入力してください。
10. [クールダウン期間] には、新しい値 (秒単位) を指定するか、デフォルトのままにします。
11. (オプション) 現在の構成に基づいてスケールインポリシーの作成を省略するには、[スケールインの無効化] を選択してください。別の構成を使用してスケールインポリシーを作成できます。
12. [Save] を選択してください。

AWS CLI を使用して、ターゲットの追跡スケーリングポリシーを設定します。

1. register-scalable-target コマンドを使用して、スケーラブルなターゲットとしてスポットフリートリクエストを登録します。
2. put-scaling-policy コマンドを使用して、スケーリングポリシーを作成します。

ステップスケーリング: ステップスケーリングポリシーを使用して、スポットフリートをスケールする

ステップスケーリングポリシーでは、CloudWatch アラームを指定してスケーリングプロセスをトリガーします。例えば、CPU 利用率が一定のレベルに達したときにスケールアウトする場合、Amazon EC2 によって提供される `CPUUtilization` メトリクスを使用してアラームを作成します。

ステップスケーリングポリシーを作成したら、次のいずれかのスケーリング調整タイプを指定する必要があります。

- [追加] - 指定した数の容量ユニットまたは指定した割合の現在の容量で、スポットフリートのターゲット容量を増やします。
- [削除] - 指定した数の容量ユニットまたは指定した割合の現在の容量で、スポットフリートのターゲット容量を減らします。
- [設定] - 指定した数の容量ユニットに、スポットフリートのターゲット容量を設定します。

アラームがトリガーされると、自動スケーリングプロセスは、取得済み容量およびスケーリングポリシーを使用して新しいターゲット容量を計算し、必要に応じてターゲット容量を更新します。例えば、ターゲット容量と取得済み容量がそれぞれ 10 で、スケーリングポリシーが 1 を加算するとします。アラームがトリガーされると、自動スケーリングプロセスは 10 に 1 を加えて 11 を得るため、スポットフリートは 1 インスタンスを起動します。

ターゲットキャパシティが減ったためにスポットフリートがスポットインスタンスを終了する場合、インスタンスはスポットインスタンスの中断通知を受け取ります。

前提条件

- スポットフリートリクエストには、タイプが `maintain` のリクエストが必要です。自動スケーリングはタイプ `request` のリクエストではサポートされていません。
- スポットフリートの自動スケーリングに必要な IAM のアクセス許可 を設定します。
- アプリケーションにとってどの CloudWatch メトリクスが重要化を検討します。AWS または独自のカスタムメトリクスが提供するメトリクスに基づいて、CloudWatch アラームを作成できます。
- スケーリングポリシーで使用する AWS メトリクスについて、メトリクスを提供するサービスがデフォルトで有効にならない場合、CloudWatch メトリクスの収集を有効にします。
- 「考慮事項」を確認します。

CloudWatch アラームを作成するには

1. CloudWatch コンソールの https://console.aws.amazon.com/cloudwatch/ を開いてください。
2. ナビゲーションペインで、[アラーム] を展開し、[すべてのアラーム] を選択してください。
3. [アラームの作成] を選択してください。
4. [Specify metric and conditions] (メトリクスと条件の指定) ページで、[Select metric] (メトリクスの選択) を選択してください。

5. [EC2 スポット]、[フリートリクエストメトリクス] の順に選択し、メトリクス (TargetCapacity など) を選択して、[メトリクスの選択] を選択してください。

   [Specify metric and conditions (メトリクスと条件の指定)] ページに、選択したメトリクスに関するグラフや他の情報が表示されます。

6. [期間] でアラームの評価期間 ([1 分] など) を選択してください。アラームを評価する場合、各期間は 1 つのデータポイントに集約されます。

   > **Note**
   >
   > 期間が短いほど、作成されるアラームの感度が高くなります。

7. [条件] で、しきい値条件を定義してアラームを定義します。例えば、メトリクスの値が 80% 以上になるたびにアラームをトリガーするように、しきい値を定義できます。

8. [Additional configuration (追加設定)] の [Datapoints to alarm (アラームするデータポイント)] で、アラームをトリガーするために ALARM 状態になる必要があるデータポイント (評価期間) の数を指定します (3 個の評価期間のうち 1 個または 2 個の評価期間など)。これでアラームが作成されます。このアラームは、指定した数の期間で連続してしきい値を超過すると、ALARM 状態に移行します。詳細については、Amazon CloudWatch ユーザーガイド のアラームを評価するを参照してください。

9. [Missing data treatment (不足しているデータの扱い)] で、いずれかのオプションを選択します (または、デフォルトの [Treat missing data as missing (不足しているデータを不足として扱う)] のままにします)。詳細については、Amazon CloudWatch ユーザーガイドの「CloudWatch アラームが欠落データを処理する方法の設定」を参照してください。

10. [Next] を選択してください。

11. (オプション) スケーリングイベントの通知を受け取る場合は、[通知] で、通知を受け取るために使用する Amazon SNS トピックを選択または作成できます。それ以外の場合は、通知を削除し、必要に応じて後で追加できます。

12. [Next] を選択してください。

13. [名前と説明の追加] にアラームの名前と説明を入力し、[次へ] を選択してください。

14. [アラームの作成] を選択します。

スポットフリートのステップスケーリングポリシーを設定するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. スポットフリートリクエストを選択してください。
4. 画面の下部にある [自動スケーリング] タブを選択してください。スポットフリートのリンクを選択した場合、タブはありません。代わりに、[自動スケーリング] セクションまでスクロールします。
5. 自動スケーリングが設定されていない場合は、[Configure] を選択してください。
6. スポットフリートの最小容量および最大容量を設定するには、[Scale capacity between] を使用します。スケーリングポリシーにより、最小容量未満に、または最大容量を超えてフリートがスケールされることはありません。
7. [スケーリングポリシー] の [ポリシータイプ] で [ステップスケーリングポリシー] を選択してください。
8. 初期状態では、[スケーリングポリシー] には ScaleUp と ScaleDown という名前のステップスケーリングポリシーが含まれています。これらのポリシーは、完了するか、[Remove policy] を選択して削除できます。[Add policy] を選択することもできます。
9. ポリシーを定義するには、以下の作業を行います。

   a. [Policy Name] にこのポリシーの名前を入力してください。
   b. [ポリシートリガー] で、既存のアラームを選択するか、[アラームを作成] を選択して Amazon CloudWatch コンソールを開き、アラームを作成します。
   c. [容量の変更] では、スケーリングする量と、ステップ調整の下限と上限を指定します。特定の数のインスタンスまたは既存のグループサイズに対する割合を追加または削除したり、フリートを正確なサイズに設定したりできます。

      例えば、フリートのキャパシティを 30 パーセント増やすステップスケーリングポリシーを作成するには、[追加] を選択し、次のフィールドに [30] と入力して、[パーセント] を選択してください。デフォルトでは、[ポリシーを追加] の下限はアラームしきい値であり、上限は正 (+) の無限大です。デフォルトでは、[ポリシーを削除] の上限はアラームしきい値であり、下限は負 (-) の無限大です。
   d. (オプション) 別のステップを追加するには、[ステップを追加] を選択してください。
   e. [クールダウン期間] には、新しい値 (秒単位) を指定するか、デフォルトのままにします。

10. [Save] を選択してください。

AWS CLI を使用して、スポットフリートのステップスケーリングポリシーを設定するには

1. register-scalable-target コマンドを使用して、スケーラブルなターゲットとしてスポットフリートリクエストを登録します。
2. put-scaling-policy コマンドを使用して、スケーリングポリシーを作成します。
3. put-metric-alarm コマンドを使用してスケーリングポリシーをトリガーするアラームを作成します。

スケジュールされたスケーリング: スケジュールに従ってスポットフリートをスケールする

スケジュールに従ってフリートをスケールすると、予想可能な需要の変化に応じてアプリケーションをスケールすることができます。スケジュールされたアクションを作成することで、スポットフリートに特定の時刻にスケーリングアクティビティを実行するように指示できます。スケジュールされたアクションを作成するには、既存のスポットフリート、スケーリングアクティビティを実行する必要がある時刻、および必要な最小キャパシティと最大キャパシティを指定する必要があります。スケジュールされたアクションは、1 回または定期的なスケジュールでスケールされるように設定できます。変更が必要な場合は、スケジュールされたアクションを編集または削除できます。

前提条件

- スケジュールされたアクションは、既存のスポットフリートに対してのみ作成できます。スケジュールされたアクションは、スポットフリートの作成時に作成することはできません。
- スポットフリートリクエストには、タイプが `maintain` のリクエストが必要です。自動スケーリングはタイプ `request` のリクエストではサポートされていません。
- スポットフリートの自動スケーリングに必要な IAM のアクセス許可 を設定します。
- 「考慮事項」を確認します。

1 回のアクションを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. スポットフリートリクエストを選択してください。
4. 画面の下部にある [スケジュールされたスケーリング] タブを選択してください。スポットフリートのリンクを選択した場合、タブはありません。代わりに、[スケジュールされたスケーリング] セクションまでスクロールします。

5. [スケジュールされたアクションの作成] を選択してください。
6. [名前] に、予定アクションの名前を指定します。
7. [最小容量]、[最大容量]、または両方の値を入力してください。
8. [繰り返し] で、[1 回] を選択してください。
9. (オプション) [開始時刻]、[終了時刻]、またはその両方の日付と時刻を選択してください。
10. [Create] (作成) を選択してください。

定期的なスケジュールされたアクションを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. スポットフリートリクエストを選択してください。
4. 画面の下部にある [スケジュールされたスケーリング] タブを選択してください。スポットフリートのリンクを選択した場合、タブはありません。代わりに、[スケジュールされたスケーリング] セクションまでスクロールします。
5. [名前] に、予定アクションの名前を指定します。
6. [最小容量]、[最大容量]、または両方の値を入力してください。
7. [繰り返し] で、事前定義済みのスケジュール (例えば、[毎日]) のいずれかを選択するか、[カスタム] を選択して cron 式を入力してください。スケジュールされたスケーリングでサポートされる cron 式の詳細については、「Amazon EventBridge ユーザーガイド」の「Cron expressions」を参照してください。
8. (オプション) [開始時刻]、[終了時刻]、またはその両方の日付と時刻を選択してください。
9. [Submit] を選択してください。

スケジュールされたアクションを編集するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. スポットフリートリクエストを選択してください。
4. 画面の下部にある [スケジュールされたスケーリング] タブを選択してください。スポットフリートのリンクを選択した場合、タブはありません。代わりに、[スケジュールされたスケーリング] セクションまでスクロールします。
5. スケジュールされたアクション を選択して、[Actions]、[Edit] の順に選択してください。

6. 必要な変更を加えて、[Submit] を選択してください。

スケジュールされたアクションを削除するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Spot Requests] を選択してください。
3. スポットフリートリクエストを選択してください。
4. 画面の下部にある [スケジュールされたスケーリング] タブを選択してください。スポットフリートのリンクを選択した場合、タブはありません。代わりに、[スケジュールされたスケーリング] セクションまでスクロールします。
5. スケジュールされたアクションを選択して、[アクション]、[削除] の順に選択してください。
6. 確認を求めるメッセージが表示されたら、[削除] を選択してください。

AWS CLI を使用してスケジュールされたスケーリングを管理するには

次のコマンドを使用します。

- put-scheduled-actionhttps://docs.aws.amazon.com/cli/latest/reference/application-autoscaling/put-scheduled-action.html
- describe-scheduled-actionshttps://docs.aws.amazon.com/cli/latest/reference/application-autoscaling/describe-scheduled-actions.html
- delete-scheduled-actionhttps://docs.aws.amazon.com/cli/latest/reference/application-autoscaling/delete-scheduled-action.html

EC2 フリートまたはスポットフリートをモニタリングする

EC2 フリートまたはスポットフリートの効果的なモニタリングは、最適なパフォーマンスを維持し、信頼性を確保するために不可欠です。これを実現するには、このトピックで説明する Amazon CloudWatch や Amazon EventBridge など、さまざまなツールがあります。

CloudWatch を使用して、メトリクスの収集と追跡、アラーム設定、フリートの状態の変更に対する自動的な対応ができます。

EventBridge を使用すると、フリートによって出力されたイベントをプログラムでモニタリングして対応できます。EventBridge でルールを定義することで、インスタンスの終了やフリート状態の変更などの特定のフリートイベントへの応答を自動化し、運用効率を向上させることができます。

トピック

- CloudWatch を使用して EC2 フリートまたはスポットフリートをモニタリングする
- Amazon EventBridge を使用して EC2 フリートまたはスポットフリートによって出力されたイベントをモニタリングし、プログラムで応答します。

CloudWatch を使用して EC2 フリートまたはスポットフリートをモニタリングする

ここで説明する Amazon CloudWatch メトリクスを使用すると、EC2 フリートまたはスポットフリートをモニタリングできます。

> **Important**
>
> 正確性を確実にするため、これらのメトリクスを使用する際は詳細モニタリングを有効にすることをお勧めします。詳細については、「EC2 インスタンスの詳細モニタリングを管理する」を参照してください。

CloudWatch の使用の詳細については、「CloudWatch を使用したインスタンスのモニタリング」を参照してください。

EC2 フリートとスポットフリートのメトリクス

AWS/EC2Spot 名前空間には、フリートに使用できる次のメトリクスに加えて、フリート内のスポットインスタンスに使用できる CloudWatch メトリクスが含まれています。詳細については、「インスタンスメトリクス」を参照してください。

メトリクス	説明
AvailableInstancePoolsCount	フリートリクエストに指定されているスポットキャパシティプール。 単位: カウント
BidsSubmittedForCapacity	Amazon EC2 がフリートリクエストを送信したキャパシティ。 単位: カウント

メトリクス	説明
EligibleInstancePoolCount	Amazon EC2 がリクエストを受理できるフリートリクエストに指定されたスポットキャパシティプール スポットインスタンスに支払う上限価格がスポット料金よりも低いか、スポット料金がオンデマンドインスタンス料金よりも高いプールでは、Amazon EC2 はリクエストを受理しません。 単位: カウント
FulfilledCapacity	Amazon EC2 が落札した容量。 単位: カウント
MaxPercentCapacityAllocation	フリートリクエストに指定されたすべてのフリートプールでの PercentCapacityAllocation の最大値。 単位: パーセント
PendingCapacity	TargetCapacity と FulfilledCapacity の違い。 単位: カウント
PercentCapacityAllocation	指定されたディメンションのスポットキャパシティープールに配分された容量。すべてのスポットキャパシティープールにおける最大値を取得するには、 を使用します。MaxPercentCapacityAllocation 単位: パーセント
TargetCapacity	フリートリクエストのターゲットキャパシティ。 単位: カウント
TerminatingCapacity	プロビジョニングされた容量が目標の容量より大きいために終了した容量。 単位: カウント

メトリクスの測定単位が Count である場合、最も有用な統計は Average です。

EC2 フリートとスポットフリートのディメンション

フリートのデータをフィルタリングするには、次のディメンションを使用します。

ディメンション	説明
AvailabilityZone	アベイラビリティーゾーン別にデータをフィルタリングします。
FleetRequestId	フリートのリクエスト別にデータをフィルタリングします。
InstanceType	インスタンスタイプ別にデータをフィルタリングします。

EC2 フリートまたはスポットフリートの CloudWatch メトリクスを表示します。

Amazon CloudWatch コンソールを使用して、フリートの CloudWatch メトリクスを表示できます。これらのメトリクスは、モニタリング用のグラフのように表示されます。これらのグラフは、フリートがアクティブである場合にデータポイントを表示します。

メトリクスはまず名前空間ごとにグループ化され、次に各名前空間内の種々のディメンションの組み合わせごとにグループ化されます。例えば、すべてのフリートメトリクスまたはフリートメトリクスグループは、フリートリクエスト ID、インスタンスタイプ、またはアベイラビリティーゾーン別に表示できます。

フリートメトリクスを表示するには

1. CloudWatch コンソールの https://console.aws.amazon.com/cloudwatch/ を開いてください。
2. ナビゲーションペインで、[メトリクス] を展開し、[すべてのメトリクス] を選択してください。
3. EC2 スポットの名前空間を選択してください。

> **Note**
>
> EC2 スポットの名前空間が表示されない場合、これには 2 つの原因があります。リージョンで EC2 フリートもスポットフリートもまだ使用したことがありません。使用中の AWS サービスのみがメトリクスを Amazon CloudWatch に送信します。あるいは、リージョンで EC2 フリートまたはスポットフリートを使用したことはあるものの、過去 2 週間にはない場合には、名前空間が表示されません。

4. ディメンション別にメトリクスをフィルタリングするには、次のいずれかを選択してください。

   - [フリートリクエストメトリクス] - フリートリクエスト別にグループ化
   - [アベイラビリティーゾーン別] - フリートリクエストおよびアベイラビリティーゾーン別にグループ化
   - [インスタンスタイプ別] - フリートリクエストおよびインスタンスタイプ別にグループ化
   - [アベイラビリティーゾーン/インスタンスタイプ別] - フリートリクエスト、アベイラビリティーゾーン、インスタンスタイプ別にグループ化

5. メトリクスのデータを表示するには、メトリクスの横にあるチェックボックスをオンにします。

Amazon EventBridge を使用して EC2 フリートまたはスポットフリートによって出力されたイベントをモニタリングし、プログラムで応答します。

EC2 フリートかスポットフリートの状態に変化があった場合、そのフリートから通知が発せられます。通知は、Amazon EventBridge (旧称 Amazon CloudWatch Events) に送信されるイベントとして利用できます。イベントは、ベストエフォートベースで出力されます。

Amazon EventBridge を使用して、イベントに応答してプログラムによるアクションをトリガーするルールを作成できます。例えば、フリートの状態が変更されたときにトリガーされるルールと、フリートのインスタンスが終了したときにトリガーされるルールという 2 つの EventBridge ルールを作成できます。この例の最初のルールでは、フリートの状態が変更された際に、ルールが SNS トピックを呼び出して E メール通知を送信するように設定できます。2 番目のルールでは、フリートのインスタンスが終了した場合に Lambda 関数を呼び出して、新しいインスタンスを起動するように設定できます。

> **Note**
>
> タイプ `maintain` と `request` のフリートのみがイベントを発行します。タイプ `instant` のフリートは、同期された 1 回限りのリクエストを送信するため、イベントを発行しません。フリートの状態は応答ですぐに認識されます。Amazon EventBridge を使用してフリートイベントをモニタリングするには、リクエストタイプが `maintain` または `request` である必要があります。

フリートのイベント履歴を表示する方法については、「EC2 フリートのイベント履歴を表示する」を参照してください。

トピック

- EC2 フリートイベントまたはスポットフリートイベントをモニタリングする Amazon EventBridge ルールを作成する
- EC2 フリート イベントタイプ
- スポットフリートイベントタイプ

EC2 フリートイベントまたはスポットフリートイベントをモニタリングする Amazon EventBridge ルールを作成する

EC2 フリートまたはスポットフリートの状態変更に関する通知が送信される場合、その通知のためのイベントが Amazon EventBridge に対し JSON ファイルとして送信されます。EventBridge がルールで定義されているパターンに一致するイベントパターンを検出すると、EventBridge はルールで指定されているターゲットを呼び出します。

EventBridge ルールを記述して、一致するイベントパターンに基づいてアクションを自動化できます。

イベント内の次のフィールドはイベントパターンを形成します。このパターンはルールで定義されます。

`"source": "aws.ec2fleet"`

イベントが EC2 フリート からのものであることを特定します。

`"detail-type": "EC2 Fleet State Change"`

イベントタイプを特定します。

`"detail": { "sub-type": "submitted" }`

イベントのサブタイプを特定します。

EC2 フリートおよびスポットフリートのイベントの一覧とイベントデータの例については、「EC2 フリート イベントタイプ」と「スポットフリートイベントタイプ」を参照してください。

例

- 通知を送信する EventBridge ルールを作成する
- Lambda 関数をトリガーする EventBridge ルールの作成

通知を送信する EventBridge ルールを作成する

次の例では、Amazon EC2 が EC2 フリート状態変更通知を発するたびに、E メール、テキストメッセージ、またはモバイルプッシュ通知を送信する EventBridge ルールを作成します。この例のシグナルは `EC2 Fleet State Change` イベントとして送信され、ルールによって定義されたアクションがトリガーされます。

前提条件

EventBridge ルールを作成する前に、E メール、テキストメッセージ、またはモバイルプッシュ通知用の Amazon SNS トピックを作成する必要があります。

EventBridge ルールを作成して EC2 フリート状態が変更されたときに通知を送信するには

1. Amazon EventBridge コンソールの https://console.aws.amazon.com/events/ を開いてください。
2. [Create rule] を選択してください。
3. [Define rule detail] (詳細の定義) で、次の操作を行います。
   a. ルールの [Name (名前)] を入力し、必要に応じて説明を入力してください。

      ルールには同じリージョン内および同じイベントバス上の別のルールと同じ名前を付けることはできません。
   b. [イベントバス] として、[デフォルト] を選択してください。アカウント内の AWS のサービスで生成されたイベントは常に、そのアカウントのデフォルトのイベントバスに送られます。
   c. [ルールタイプ] で、[イベントパターンを持つルール] を選択してください。
   d. [Next] を選択してください。
4. [Build event pattern] (イベントパターンの作成) で、次の操作を行います。
   a. [Event source] (イベントソース) で、[AWS events or EventBridge partner events] (イベントまたは EventBridge パートナーイベント) を選択してください。
   b. この例では [Event pattern] (イベントパターン) で、`EC2 Fleet Instance Change` イベントと一致するように以下のイベントパターンを指定します。

```
{
 "source": ["aws.ec2fleet"],
 "detail-type": ["EC2 Fleet Instance Change"]
```

```
}
```

イベントパターンを追加するには以下のように [Event pattern form] (イベントパターンフォーム) を選択してテンプレートを使用するか、[Custom pattern (JSON editor)] (カスタムパターン (JSON エディター)) を選択して独自のパターンを指定します。

i. テンプレートを使用してイベントパターンを作成するには以下の操作を行います。

A. [Event pattern form] (イベントパターンフォーム) を選択してください。

B. [イベントソース] で、[AWS のサービス] を選択してください。

C. [AWS Service] (サービス) で、[EC2 Fleet] (EC2 フリート) を選択してください。

D. [Event type] (イベントタイプ) で、[EC2 Fleet Instance Change] (EC2 フリートインスタンスの変更) を選択してください。

E. テンプレートをカスタマイズするには[Edit pattern] (パターンを編集) を選択した上で、この例のイベントパターンに合わせた変更を行います。

ii. (代替案) 以下の操作を行って、カスタムイベントパターンを指定します。

A. [Custom pattern (JSON editor)] (カスタムパターン (JSON エディター)) を選択してください。

B. [Event pattern] (イベントパターン) ボックスに、この例のイベントパターンを追加します。

c. [Next] を選択してください。

5. [Select target(s)] (ターゲットを選択) で、以下の操作を行います。

a. [ターゲットタイプ] で、[AWS のサービス] を選択してください。

b. イベントの発生時に E メール、テキストメッセージ、またはモバイルプッシュ通知を送信するために、[Select a target] (ターゲットを選択) で、[SNS topic] (SNS トピック) を選択してください。

c. [Topic (トピック)] で、既存のトピックを選択してください。先ず、Amazon SNS コンソールを使用して Amazon SNS トピックを作成する必要があります。詳細については、Amazon Simple Notification Service デベロッパーガイド のAmazon SNS を使用した Application-to-Person (A2P) メッセージングを参照してください。

d. (オプション) [Additional settings] (追加設定) で、その他の設定を行うこともできます。詳細については、「Amazon EventBridge ユーザーガイド」の「イベントに反応する Amazon EventBridge ルールの作成」(ステップ 16) を参照してください。

e. [Next] を選択してください。

6. (オプション) 必要な場合は、[Tags] (タグ) で 1 つ以上のタグを作成したルールに割り当て、[Next] (次へ) を選択してください。
7. [Review and create] (確認して作成) で、以下の操作を行います。

   a. ルールの詳細を確認し、必要な場合は変更を行います。

   b. ルールの作成を選択してください。

詳細については、「Amazon EventBridge ユーザーガイド」の「Amazon EventBridge ルール」と「Amazon EventBridge イベントパターン」を参照してください。

Lambda 関数をトリガーする EventBridge ルールの作成

次の例では、Amazon EC2 が EC2 フリートインスタンスが起動したときのインスタンス変更通知を発するたびに Lambda 関数をトリガーする EventBridge ルールを作成します。この例のシグナルは `EC2 Fleet Instance Change` イベント、サブタイプ `launched` として発され、ルールによって定義されたアクションがトリガーされます。

EventBridge ルールを作成する前に、Lambda 関数を作成する必要があります。

EventBridge ルールで使用する Lambda 関数を作成するには

1. AWS Lambda コンソールの https://console.aws.amazon.com/lambda/ を開いてください。
2. [関数の作成] を選択してください。
3. 関数の名前を入力し、コードを設定し、[Create function (関数の作成)] を選択してください。

   詳細については、「AWS Lambda デベロッパーガイド」の「初めての Lambda 関数の作成」を参照してください。

EC2 フリート のインスタンスの状態が変更されたときに Lambda 関数をトリガーする EventBridge ルールを作成するには

1. Amazon EventBridge コンソールの https://console.aws.amazon.com/events/ を開いてください。
2. [Create rule] を選択してください。
3. [Define rule detail] (詳細の定義) で、次の操作を行います。

   a. ルールの [Name (名前)] を入力し、必要に応じて説明を入力してください。

ルールには同じリージョン内および同じイベントバス上の別のルールと同じ名前を付けることはできません。

b. [イベントバス] として、[デフォルト] を選択してください。アカウント内の AWS のサービスで生成されたイベントは常に、そのアカウントのデフォルトのイベントバスに送られます。

c. [ルールタイプ] で、[イベントパターンを持つルール] を選択してください。

d. [Next] を選択してください。

4. [Build event pattern] (イベントパターンの作成) で、次の操作を行います。

a. [Event source] (イベントソース) で、[AWS events or EventBridge partner events] (イベントまたは EventBridge パートナーイベント) を選択してください。

b. この例では [Event pattern] (イベントパターン) で `EC2 Fleet Instance Change` イベントと `launched` サブタイプに一致するよう、以下のイベントパターンを指定します。

```
{
 "source": ["aws.ec2fleet"],
 "detail-type": ["EC2 Fleet Instance Change"],
 "detail": {
   "sub-type": ["launched"]
}
```

イベントパターンを追加するには以下のように [Event pattern form] (イベントパターンフォーム) を選択してテンプレートを使用するか、[Custom pattern (JSON editor)] (カスタムパターン (JSON エディター)) を選択して独自のパターンを指定します。

i. テンプレートを使用してイベントパターンを作成するには以下の操作を行います。

A. [Event pattern form] (イベントパターンフォーム) を選択してください。

B. [イベントソース] で、[AWS のサービス] を選択してください。

C. [AWS Service] (サービス) で、[EC2 Fleet] (EC2 フリート) を選択してください。

D. [Event type] (イベントタイプ) で、[EC2 Fleet Instance Change] (EC2 フリートインスタンスの変更) を選択してください。

E. [Edit pattern] (パターンの編集) を選択し、サンプルのイベントパターンに一致するように `"detail": {"sub-type": ["launched"]` を追加します。適切な JSON 形式を得るには、前にある角括弧]の後にコンマ ,を入力してください。

ii. (代替案) 以下の操作を行って、カスタムイベントパターンを指定します。

A. [Custom pattern (JSON editor)] (カスタムパターン (JSON エディター)) を選択してください。

B. [Event pattern] (イベントパターン) ボックスに、この例のイベントパターンを追加します。

c. [Next] を選択してください。

5. [Select target(s)] (ターゲットを選択) で、以下の操作を行います。

a. [ターゲットタイプ] で、[AWS のサービス] を選択してください。

b. イベントの発生時に E メール、テキストメッセージ、またはモバイルプッシュ通知を送信するために、[Select a target] (ターゲットを選択) で、[SNS topic] (SNS トピック) を選択してください。

c. [Topic] (トピック) で [Lambda function] (Lambda 関数) を選択し、[Function] (関数) では、イベント発生時に応答するように作成した関数を選択してください。

d. (オプション) [Additional settings] (追加設定) で、その他の設定を行うこともできます。詳細については、「Amazon EventBridge ユーザーガイド」の「イベントに反応する Amazon EventBridge ルールの作成」(ステップ 16) を参照してください。

e. [Next] を選択してください。

6. (オプション) 必要な場合は、[Tags] (タグ) で 1 つ以上のタグを作成したルールに割り当て、[Next] (次へ) を選択してください。

7. [Review and create] (確認して作成) で、以下の操作を行いす。

a. ルールの詳細を確認し、必要な場合は変更を行います。

b. ルールの作成を選択してください。

Lambda 関数を作成する方法のチュートリアルと Lambda 関数を実行する EventBridge ルールについては、AWS Lambda デベロッパーガイドの「チュートリアル: EventBridge を使用して Amazon EC2 インスタンスの状態をログに記録する」を参照してください。

EC2 フリート イベントタイプ

5 つの EC2 フリート イベントタイプがあります。イベントタイプごとに、いくつかのサブタイプがあります。

イベントタイプ

- EC2 フリートの状態の変更

- EC2 フリートのスポットインスタンスリクエストの変更
- EC2 フリートインスタンスの変更
- EC2 フリート情報
- EC2 フリートエラー

EC2 フリートの状態の変更

EC2 フリート は、EC2 フリート の状態が変更されたときに `EC2 Fleet State Change` イベントを Amazon EventBridge に送信します。

以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
    "id": "715ed6b3-b8fc-27fe-fad6-528c7b8bf8a2",
    "detail-type": "EC2 Fleet State Change",
    "source": "aws.ec2fleet",
    "account": "123456789012",
    "time": "2020-11-09T09:00:20Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:fleet/fleet-598fb973-87b7-422d-
be4d-6b0809bfff0a"
    ],
    "detail": {
        "sub-type": "active"
    }
}
```

`sub-type` に指定できる値は次のとおりです。

`active`

EC2 フリート リクエストは検証済みです。Amazon EC2 は実行中のインスタンスをターゲット数分、確保しようとしています。

`deleted`

EC2 フリート リクエストが削除され、実行中のインスタンスがありません。EC2 フリート は、そのインスタンスが終了してから 2 日後に削除されます。

`deleted_running`

EC2 フリート リクエストが削除され、追加のインスタンスは起動されません。その既存のインスタンスは、中断または終了されるまで実行され続けます。リクエストは、すべてのインスタンスが中断されるか終了されるまで、この状態のままになります。

`deleted_terminating`

EC2 フリートリクエストが削除され、そのインスタンスが終了します。リクエストは、すべてのインスタンスが終了されるまで、この状態のままになります。

`expired`

EC2 フリートリクエストが期限切れです。このリクエストが `TerminateInstancesWithExpiration` セットを使用して作成されている場合、後続の `terminated` イベントは、インスタンスが終了済みなことを示します。

`modify_in_progress`

EC2 フリート リクエストは変更中です。リクエストは、この変更が完全に処理されるまで、同じ状態を維持します。

`modify_succeeded`

EC2 フリートリクエストが変更されました。

`submitted`

EC2 フリート リクエストは評価中です。Amazon EC2 は目標数のインスタンスを起動する準備をしています。

`progress`

EC2 フリートリクエストは受理中です。

EC2 フリートのスポットインスタンスリクエストの変更

EC2 フリート は、フリート内のスポットインスタンスリクエストの状態が変更されたときに `EC2 Fleet Spot Instance Request Change` イベントを Amazon EventBridge に送信します。

以下はこのイベントのサンプルデータです。

```
{
```

```
    "version": "0",
    "id": "19331f74-bf4b-a3dd-0f1b-ddb1422032b9",
    "detail-type": "EC2 Fleet Spot Instance Request Change",
    "source": "aws.ec2fleet",
    "account": "123456789012",
    "time": "2020-11-09T09:00:05Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:fleet/
fleet-83fd4e48-552a-40ef-9532-82a3acca5f10"
    ],
    "detail": {
        "spot-instance-request-id": "sir-rmqske6h",
        "description": "SpotInstanceRequestId sir-rmqske6h, PreviousState:
 cancelled_running",
        "sub-type": "cancelled"
    }
}
```

sub-type に指定できる値は次のとおりです。

active

スポットインスタンスリクエストは受理された状態であり、スポットインスタンスの関連付けが完了しています。

cancelled

スポットインスタンスリクエストがキャンセルされている、あるいは、そのリクエストの有効期限が切れています。

disabled

スポットインスタンスが停止されています。

submitted

スポットインスタンスリクエストは送信済みです。

EC2 フリートインスタンスの変更

EC2 フリート は、フリート内のインスタンスの状態が変更されたときに EC2 Fleet Instance Change イベントを Amazon EventBridge に送信します。

以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
    "id": "542ce428-c8f1-0608-c015-e8ed6522c5bc",
    "detail-type": "EC2 Fleet Instance Change",
    "source": "aws.ec2fleet",
    "account": "123456789012",
    "time": "2020-11-09T09:00:23Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:fleet/fleet-598fb973-87b7-422d-
be4d-6b0809bfff0a"
    ],
    "detail": {
        "instance-id": "i-0c594155dd5ff1829",
        "description": "{\"instanceType\":\"c5.large\",\"image\":\"ami-6057e21a\",
\"productDescription\":\"Linux/UNIX\",\"availabilityZone\":\"us-east-1d\"}",
        "sub-type": "launched"
    }
}
```

sub-type に指定できる値は次のとおりです。

launched

新しいインスタンスが起動されました。

terminated

このインスタンスは終了しています。

termination_notified

フリートのターゲット容量のスケールダウン中 (ターゲット容量 が 4 から 3 に変更される場合など) に、Amazon EC2 によってスポットインスタンスが終了されたので、インスタンス終了通知が送信されました。

EC2 フリート情報

EC2 フリート は、受理中にエラーが発生したときに EC2 Fleet Information イベントを Amazon EventBridge に送信します。情報イベントは、フリートがターゲット容量を満たすことをブロックしません。

以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
    "id": "76529817-d605-4571-7224-d36cc1b2c0c4",
    "detail-type": "EC2 Fleet Information",
    "source": "aws.ec2fleet",
    "account": "123456789012",
    "time": "2020-11-09T08:17:07Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:fleet/fleet-8becf5fe-
bb9e-415d-8f54-3fa5a8628b91"
    ],
    "detail": {
        "description": "c4.xlarge, ami-0947d2ba12ee1ff75, Linux/UNIX, us-east-1a,
 Spot price in either SpotFleetRequestConfigData or SpotFleetLaunchSpecification or
 LaunchTemplate or LaunchTemplateOverrides is less than Spot market price $0.0619",
        "sub-type": "launchSpecUnusable"
    }
}
```

sub-type に指定できる値は次のとおりです。

`fleetProgressHalted`

すべての起動仕様の料金は、スポット料金を下回っているため無効です (すべての起動仕様が launchSpecUnusable イベントを生成しました)。スポット料金が変更されると、起動仕様が有効になる場合があります。

`launchSpecTemporarilyBlacklisted`

設定が有効ではなく、インスタンスを起動しようとして何回か失敗しました。詳細については、イベントの説明をご覧ください。

`launchSpecUnusable`

この起動仕様の料金は、スポット料金を下回っているため無効です。

`registerWithLoadBalancersFailed`

ロードバランサーにインスタンスを登録しようとして失敗しました。詳細については、イベントの説明をご覧ください。

EC2 フリートエラー

EC2 フリート は、受理中にエラーが発生したときに EC2 Fleet Error イベントを Amazon EventBridge に送信します。エラーイベントは、フリートがターゲット容量を満たすことをブロックします。

以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
    "id": "69849a22-6d0f-d4ce-602b-b47c1c98240e",
    "detail-type": "EC2 Fleet Error",
    "source": "aws.ec2fleet",
    "account": "123456789012",
    "time": "2020-10-07T01:44:24Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:fleet/fleet-9bb19bc6-60d3-4fd2-ae47-
d33e68eafa08"
    ],
    "detail": {
        "description": "m3.large, ami-00068cd7555f543d5, Linux/UNIX: IPv6 is not
 supported for the instance type 'm3.large'. ",
        "sub-type": "spotFleetRequestConfigurationInvalid"
    }
}
```

sub-type に指定できる値は次のとおりです。

`iamFleetRoleInvalid`

この EC2 フリートには、インスタンスの起動または終了に必要なアクセス許可がありません。

`allLaunchSpecsTemporarilyBlacklisted`

有効な設定はありません。インスタンスを起動しようとして何回か失敗しました。詳細については、イベントの説明をご覧ください。

`spotInstanceCountLimitExceeded`

起動できるスポットインスタンスの数の上限に達しました。

`spotFleetRequestConfigurationInvalid`

設定が有効ではありません。詳細については、イベントの説明をご覧ください。

スポットフリートイベントタイプ

> **Note**
>
> スポットフリートは、計画された投資なしでレガシー API を使用します。代わりに EC2 フリートまたは Auto Scaling グループを使用することをお勧めします。詳細については、「フリートにどの方法を使用するのが最適か」を参照してください。

5 つのスポットフリートイベントタイプがあります。イベントタイプごとに、いくつかのサブタイプがあります。

イベントタイプ

- EC2 スポットフリートの状態の変更
- EC2 スポットフリートのスポットインスタンスリクエストの変更
- EC2 スポットフリートインスタンスの変更
- EC2 スポットフリート情報
- EC2 スポットフリートのエラー

EC2 スポットフリートの状態の変更

スポットフリートは、スポットフリートの状態が変更されたときに EC2 Spot Fleet State Change イベントを Amazon EventBridge に送信します。

以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
    "id": "d1af1091-6cc3-2e24-203a-3b870e455d5b",
    "detail-type": "EC2 Spot Fleet State Change",
    "source": "aws.ec2spotfleet",
    "account": "123456789012",
    "time": "2020-11-09T08:57:06Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:spot-fleet-request/sfr-4b6d274d-0cea-4b2c-
b3be-9dc627ad1f55"
    ],
    "detail": {
```

```
        "sub-type": "submitted"
    }
}
```

sub-type に指定できる値は次のとおりです。

active

このスポットフリートリクエストは検証済みです。Amazon EC2 は実行中のインスタンスを目標数分、確保しようとしています。

cancelled

このスポットフリートリクエストはキャンセルされており、実行中のインスタンスはありません。スポットフリートは、そのインスタンスが終了されてから 2 日後に削除されます。

cancelled_running

このスポットフリートリクエストはキャンセルされており、追加のインスタンスは起動されません。その既存のインスタンスは、中断または終了されるまで実行され続けます。リクエストは、すべてのインスタンスが中断されるか終了されるまで、この状態のままになります。

cancelled_terminating

このスポットフリートリクエストはキャンセルされており、対象のインスタンスを終了中です。リクエストは、すべてのインスタンスが終了されるまで、この状態のままになります。

expired

スポットフリートリクエストの有効期限が切れました。このリクエストが TerminateInstancesWithExpiration セットを使用して作成されている場合、後続の terminated イベントは、インスタンスが終了済みなことを示します。

modify_in_progress

スポットフリートリクエストは変更中です。リクエストは、この変更が完全に処理されるまで、同じ状態を維持します。

modify_succeeded

スポットフリートリクエストが変更されました。

submitted

スポットフリートリクエストは評価中です。Amazon EC2 は目標数のインスタンスを起動する準備をしています。

progress

スポットフリートリクエストは受理中です。

EC2 スポットフリートのスポットインスタンスリクエストの変更

スポットフリートは、フリート内のスポットインスタンスリクエストの状態が変更されたときに EC2 Spot Fleet Spot Instance Request Change イベントを Amazon EventBridge に送信します。

以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
    "id": "cd141ef0-14af-d670-a71d-fe46e9971bd2",
    "detail-type": "EC2 Spot Fleet Spot Instance Request Change",
    "source": "aws.ec2spotfleet",
    "account": "123456789012",
    "time": "2020-11-09T08:53:21Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:spot-fleet-request/sfr-
a98d2133-941a-47dc-8b03-0f94c6852ad1"
    ],
    "detail": {
        "spot-instance-request-id": "sir-a2w9gc5h",
        "description": "SpotInstanceRequestId sir-a2w9gc5h, PreviousState:
 cancelled_running",
        "sub-type": "cancelled"
    }
}
```

sub-type に指定できる値は次のとおりです。

active

スポットインスタンスリクエストは受理された状態であり、スポットインスタンスの関連付けが完了しています。

cancelled

スポットインスタンスリクエストがキャンセルされている、あるいは、そのリクエストの有効期限が切れています。

`disabled`

スポットインスタンスが停止されています。

`submitted`

スポットインスタンスリクエストは送信済みです。

EC2 スポットフリートインスタンスの変更

スポットフリートは、フリート内のインスタンスの状態が変更されたときに `EC2 Spot Fleet Instance Change` イベントを Amazon EventBridge に送信します。

以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
    "id": "11591686-5bd7-bbaa-eb40-d46529c2710f",
    "detail-type": "EC2 Spot Fleet Instance Change",
    "source": "aws.ec2spotfleet",
    "account": "123456789012",
    "time": "2020-11-09T07:25:02Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:spot-fleet-request/sfr-c8a764a4-bedc-4b62-
af9c-0095e6e3ba61"
    ],
    "detail": {
        "instance-id": "i-08b90df1e09c30c9b",
        "description": "{\"instanceType\":\"r4.2xlarge\",\"image\":
\"ami-032930428bf1abbff\",\"productDescription\":\"Linux/UNIX\",\"availabilityZone\":
\"us-east-1a\"}",
        "sub-type": "launched"
    }
}
```

`sub-type` に指定できる値は次のとおりです。

`launched`

新しいインスタンスが起動されました。

`terminated`

このインスタンスは終了しています。

`termination_notified`

フリートのターゲット容量のスケールダウン中 (ターゲット容量 が 4 から 3 に変更される場合など) に、Amazon EC2 によってスポットインスタンスが終了されたので、インスタンス終了通知が送信されました。

EC2 スポットフリート情報

スポットフリートは、受理中にエラーが発生したときに `EC2 Spot Fleet Information` イベントを Amazon EventBridge に送信します。情報イベントは、フリートがターゲット容量を満たすことをブロックしません。

以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
    "id": "73a60f70-3409-a66c-635c-7f66c5f5b669",
    "detail-type": "EC2 Spot Fleet Information",
    "source": "aws.ec2spotfleet",
    "account": "123456789012",
    "time": "2020-11-08T20:56:12Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:spot-fleet-request/sfr-2531ea06-
af18-4647-8757-7d69c94971b1"
    ],
    "detail": {
        "description": "r3.8xlarge, ami-032930428bf1abbff, Linux/UNIX, us-east-1a, Spot
 bid price is less than Spot market price $0.5291",
        "sub-type": "launchSpecUnusable"
    }
}
```

`sub-type` に指定できる値は次のとおりです。

`fleetProgressHalted`

すべての起動仕様の料金は、スポット料金を下回っているため無効です (すべての起動仕様が `launchSpecUnusable` イベントを生成しました)。スポット料金が変更されると、起動仕様が有効になる場合があります。

`launchSpecTemporarilyBlacklisted`

設定が有効ではなく、インスタンスを起動しようとして何回か失敗しました。詳細については、イベントの説明をご覧ください。

`launchSpecUnusable`

この起動仕様の料金は、スポット料金を下回っているため無効です。

`registerWithLoadBalancersFailed`

ロードバランサーにインスタンスを登録しようとして失敗しました。詳細については、イベントの説明をご覧ください。

EC2 スポットフリートのエラー

スポットフリートは、受理中にエラーが発生したときに `EC2 Spot Fleet Error` イベントを Amazon EventBridge に送信します。エラーイベントは、フリートがターゲット容量を満たすことをブロックします。

以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
    "id": "10adc4e7-675c-643e-125c-5bfa1b1ba5d2",
    "detail-type": "EC2 Spot Fleet Error",
    "source": "aws.ec2spotfleet",
    "account": "123456789012",
    "time": "2020-11-09T06:56:07Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:spot-fleet-request/
sfr-38725d30-25f1-4f30-83ce-2907c56dba17"
    ],
    "detail": {
        "description": "r4.2xlarge, ami-032930428bf1abbff, Linux/UNIX: The
 associatePublicIPAddress parameter can only be specified for the network interface
 with DeviceIndex 0. ",
```

```
        "sub-type": "spotFleetRequestConfigurationInvalid"
    }
}
```

sub-type に指定できる値は次のとおりです。

iamFleetRoleInvalid

このスポットフリートには、インスタンスの起動または終了に必要なアクセス許可がありません。

allLaunchSpecsTemporarilyBlacklisted

有効な設定はありません。インスタンスを起動しようとして何回か失敗しました。詳細については、イベントの説明をご覧ください。

spotInstanceCountLimitExceeded

起動できるスポットインスタンスの数の上限に達しました。

spotFleetRequestConfigurationInvalid

設定が有効ではありません。詳細については、イベントの説明をご覧ください。

EC2 フリートのチュートリアル

EC2 フリートを設定するには、さまざまな方法があります。選択する設定は、特定のユースケースによって異なります。

以下のチュートリアルでは、考えられるユースケースの一部について説明し、その実装に必要なタスクについて説明します。

ユースケース	チュートリアルへのリンク
インスタンスの重み付けを使用して、EC2 フリートの可用性とパフォーマンスを管理します。 インスタンスの重み付けでは、EC2 フリート内の各インスタンスタイプに重みを割り当てて、それぞれのコンピューティングキャパシティとパフォーマンスを相対的に表します。	チュートリアル: インスタンスの重み付けを使用するように EC2 フリートを設定する

ユースケース	チュートリアルへのリンク
この重みに基づいて、目的のターゲットキャパシティを満たしている限り、指定されたインスタンスタイプを任意に組み合わせて使用できます。	
オンデマンドキャパシティを使用して、ピーク時の可用性を確保します。また、追加のスポットキャパシティを低コストで活用します。 ピーク時に使用可能なキャパシティを確保するために、プライマリキャパシティとしてオンデマンドインスタンスを使用するように EC2 フリートを設定します。また、Amazon EC2 がキャパシティを必要とする場合、スポットインスタンスが中断される可能性があることに留意しながら、スポットインスタンスにある程度のキャパシティを割り当てて、割引料金の恩恵を受けることができます。	チュートリアル: オンデマンドインスタンスをプライマリキャパシティとして使用するように EC2 フリートを設定する
キャパシティ予約を使用して、オンデマンドインスタンスのコンピューティングキャパシティを予約します。 オンデマンドインスタンスを起動するときに、`targeted` キャパシティ予約を最初に使用するように EC2 フリートを設定します。キャパシティ要件が厳しく、一定レベルの長期または短期のキャパシティ保証を必要とするビジネスクリティカルなワークロードを実行している場合は、キャパシティ予約を作成して、必要なときに、必要な限り常に Amazon EC2 キャパシティにアクセスできるようにすることをお勧めします。	チュートリアル: ターゲットのキャパシティ予約を使用してオンデマンドインスタンスを起動するように EC2 フリートを設定する

ユースケース	チュートリアルへのリンク
キャパシティブロックを使用して、ML ワークロードの非常に需要の高い GPU インスタンスを予約します。 キャパシティブロックにインスタンスを起動するように EC2 フリートを設定します。	チュートリアル: キャパシティブロックにインスタンスを起動するように EC2 フリートを設定する
中断可能なキャパシティ予約を使用して、アイドル状態のキャパシティを AWS 組織全体で一時的に再利用してコストを削減します。 AWS 組織内の予備容量を表す中断可能なキャパシティ予約に対してインスタンスを起動するように EC2 フリートを設定します。キャパシティ所有者はいつでもキャパシティを再利用できます。EC2 は再利用されると、2 分間の通知後にインスタンスを終了します。	チュートリアル: キャパシティ予約に対してインスタンスを起動するように EC2 フリートを設定する

チュートリアル: インスタンスの重み付けを使用するように EC2 フリートを設定する

このチュートリアルでは、サンプル株式会社という名前の架空の会社を使用して、インスタンスの分量指定を使った EC2 フリートリクエストのプロセスを説明します。

目的

製薬会社であるサンプル株式会社は、癌と闘うために使用できる可能性のある化合物をスクリーニングするために Amazon EC2 の計算処理能力を活用したいと考えています。

計画

サンプル株式会社はまず、「Spot Best Practices」を確認します。次に、サンプル株式会社は EC2 フリート に関する要件を確認します。

インスタンスタイプ

サンプル株式会社には、60 GB 以上のメモリと 8 つの仮想 CPU (vCPU) で最適に実行される、計算能力とメモリに負担がかかるアプリケーションがあります。同社は、できるだけ低価格でアプリケーション用のこれらのリソースを最大化したいと考えています。サンプル株式会社は、以下のいずれかの EC2 インスタンスタイプがそのニーズを満たすと判断します。

インスタンスタイプ	メモリ (GiB)	vCPU
r3.2xlarge	61	8
r3.4xlarge	122	16
r3.8xlarge	244	32

ユニット単位のターゲット容量

インスタンスの分量指定を使用すると、ターゲット容量はインスタンスの数 (デフォルト)、またはコア (vCPU)、メモリ (GiB) とストレージ (GB) との要素の組み合わせで表すことができます。アプリケーションのベース (60 GB の RAM と 8 個の vCPU) を 1 ユニットとして考えることで、サンプル株式会社はこの量の 20 倍で十分ニーズに合うと決定します。これにより、会社は EC2 フリートリクエストのターゲットキャパシティを 20 ユニットに設定します。

インスタンスの分量

ターゲット容量の決定後、サンプル株式会社はインスタンスの分量を計算します。各インスタンスタイプのインスタンスの分量を計算することは、以下のように、ターゲット容量に達するために必要な各インスタンスタイプのユニットの数を決定することです。

- r3.2xlarge (61.0 GB、8 個の vCPU) = 1/20 ユニット
- r3.4xlarge (122.0 GB、16 個の vCPU) = 2/20 ユニット
- r3.8xlarge (244.0 GB、32 個の vCPU) = 4/20 ユニット

これよりサンプル株式会社は、1、2 と 4 のインスタンス分量を EC2 フリート リクエストのそれぞれの起動設定に割り当てます。

ユニット時間あたりの価格

サンプル株式会社は、料金の出発点としてインスタンス時間あたりの「オンデマンド料金」を使用します。最近のスポット料金または 2 つの組み合わせを使用することもできます。ユニット時間あたりの料金を計算するために、インスタンス時間あたりの出発点の料金を分量で割ります。次に例を示します。

インスタンスタイプ	オンデマンド価格	インスタンスの分量	ユニット時間あたりの価格
r3.2xLarge	$0.7	1	$0.7
r3.4xLarge	$1.4	2	$0.7
r3.8xLarge	$2.8	4	$0.7

サンプル株式会社は、ユニット時間あたりのグローバルな料金として 0.7 USD を使用し、3 つのインスタンスタイプすべてで競争力を高めることもできます。また、`r3.8xlarge` の起動条件のなかで、1 ユニット時間あたりの全体料金を 0.7 USD、そして 1 ユニット時間あたりの指定入力料金を 0.9 USD とすることもできます。

アクセス許可の確認

EC2 フリート を作成する前に、サンプル株式会社は必要なアクセス許可の IAM ロールがあることを確認します。詳細については、「EC2 フリートの前提条件」を参照してください。

起動テンプレートの作成

次に、Example Corp は起動テンプレートを作成します。起動テンプレート ID は、次のステップで使用されます。詳細については、「Amazon EC2 起動テンプレートを作成する」を参照してください。

EC2 フリート の作成

サンプル株式会社は、その EC2 フリート用に次の設定を使用して `config.json` ファイルを作成します。次の例では、リソース識別子を独自のリソース識別子に置き換えます。

```
{
```

```
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateId": "lt-07b3bc7625cdab851",
                "Version": "1"
            },
            "Overrides": [
                {
                    "InstanceType": "r3.2xlarge",
                    "SubnetId": "subnet-482e4972",
                    "WeightedCapacity": 1
                },
                {
                    "InstanceType": "r3.4xlarge",
                    "SubnetId": "subnet-482e4972",
                    "WeightedCapacity": 2
                },
                {
                    "InstanceType": "r3.8xlarge",
                    "MaxPrice": "0.90",
                    "SubnetId": "subnet-482e4972",
                    "WeightedCapacity": 4
                }
            ]
        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 20,
        "DefaultTargetCapacityType": "spot"
    }
}
```

サンプル株式会社は、次の create-fleet コマンドを使用して EC2 フリート を作成します。https://docs.aws.amazon.com/cli/latest/reference/ec2/create-fleet.html

```
aws ec2 create-fleet --cli-input-json file://config.json
```

詳細については、「EC2 フリートの作成」を参照してください。

フルフィルメント

配分戦略では、スポットインスタンスの提供元となるスポットキャパシティプールが決定されます。

lowest-price 戦略 (デフォルトの戦略) では、受理時にユニットあたりの料金が最低値であるプールから スポットインスタンス が取得されます。20 ユニットの容量を提供するためには、20 個の r3.2xlarge インスタンス (20 ÷ 1)、10 個の r3.4xlarge インスタンス (20 ÷ 2)、あるいは 5 個の r3.8xlarge インスタンス (20 ÷ 4) が EC2 フリート から起動されることになります。

サンプル株式会社が diversified 戦略を採用する場合、スポットインスタンス は 3 つのすべてのプールから取得されます。EC2 フリート は、6 個の r3.2xlarge インスタンス (6 ユニットを提供)、3 個の r3.4xlarge インスタンス (6 ユニットを提供)、そして 2 個の r3.8xlarge インスタンス (8 ユニットを提供) の全部で 20 ユニットを起動します。

チュートリアル: オンデマンドインスタンスをプライマリキャパシティとして使用するように EC2 フリートを設定する

このチュートリアルでは、ABC Online という架空の会社を使用して、プライマリ容量および使用可能な場合はスポット容量としてオンデマンドの EC2 フリートをリクエストするプロセスを説明します。

目的

レストラン向け配達会社である ABC Online は、EC2 インスタンスタイプおよび購入オプション間で Amazon EC2 キャパシティをプロビジョンし、必要なスケール、パフォーマンス、コストの実現を目指しています。

プラン

ABC Online はピーク時に対応するために固定キャパシティを必要としていますが、より低コストで追加キャパシティの恩恵を受けたいと考えています。ABC Online は、EC2 フリートについて以下の要件を設定しました。

- オンデマンドインスタンス容量 - ABC Online には、ピーク期間のトラフィックに対応できるように、15 個のオンデマンドインスタンスが必要です。
- スポットインスタンスキャパシティ – 低価格でパフォーマンスを向上させるために、ABC Online は 5 つのスポットインスタンスをプロビジョニングする予定です。

アクセス許可の確認

EC2 フリート を作成する前に、ABC Online は必要なアクセス許可の IAM ロールがあることを確認します。詳細については、「EC2 フリートの前提条件」を参照してください。

起動テンプレートの作成

次に、ABC Online によって起動テンプレートが作成されます。起動テンプレート ID は、次のステップで使用されます。詳細については、「Amazon EC2 起動テンプレートを作成する」を参照してください。

EC2 フリート の作成

ABC Online は、その EC2 フリート用に次の設定を使用して config.json ファイルを作成します。次の例では、リソース識別子を独自のリソース識別子に置き換えます。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateId": "lt-07b3bc7625cdab851",
                "Version": "2"
            }

        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 20,
        "OnDemandTargetCapacity":15,
        "DefaultTargetCapacityType": "spot"
    }
}
```

ABC Online は、次の create-fleet コマンドを使用して EC2 フリート を作成します。https://docs.aws.amazon.com/cli/latest/reference/ec2/create-fleet.html

```
aws ec2 create-fleet --cli-input-json file://config.json
```

詳細については、「EC2 フリートの作成」を参照してください。

フルフィルメント

配分戦略により、オンデマンドキャパシティが常に受理され、使用可能なキャパシティがある場合はターゲットキャパシティがスポットとして受理されることが決定されます。

チュートリアル: ターゲットのキャパシティ予約を使用してオンデマンドインスタンスを起動するように EC2 フリートを設定する

このチュートリアルでは実行するべきステップを段階的に解説しています。それにより、EC2 フリートがオンデマンドインスタンスをtargetedキャパシティー予約で起動できるようにします。

オンデマンドインスタンスの起動時に、最初にtargetedオンデマンドキャパシティー予約を使用するようにフリートを設定する方法を学習します。また、オンデマンドターゲット容量の合計が使用可能な未使用のキャパシティ予約数を超えた場合、フリートは指定された割り当て戦略を使用して、残りのターゲット容量を起動するインスタンスプールを選択するようにフリートを設定する方法についても学習します。

EC2 フリートの設定

このチュートリアルでは、フリートは次のように設定されます。

- ターゲット容量:10 オンデマンドインスタンス
- 未使用の合計targetedキャパシティー予約:6 (フリートの目標オンデマンド容量である 10 オンデマンドインスタンスを下回っています)
- キャパシティ予約のプールの数:2 (us-east-1aおよびus-east-1b)
- プールあたりのキャパシティ予約数:3
- オンデマンド割り当て戦略：lowest-price(未使用キャパシティーの予約の数が目標オンデマンド容量より少ない場合、フリートは、オンデマンド配分戦略に基づいて、残りのオンデマンド容量を起動するプールを決定します)。

  また、lowest-price割り当て戦略の代わりにprioritized割り当て戦略を使用することもできます。

targetedキャパシティ予約にオンデマンドインスタンスを起動するには、次のように、いくつかの手順を実行する必要があります：

- ステップ 1: キャパシティー予約を作成する
- ステップ 2: キャパシティー予約のリソースグループを作成する
- ステップ 3: キャパシティ予約リソースグループにキャパシティ予約を追加する
- (オプション) ステップ 4: リソースグループのキャパシティーの予約を表示する
- ステップ 5: キャパシティ予約が特定のリソースグループをターゲットに指定する起動テンプレートを作成する

- (オプション) ステップ 6: 起動テンプレートを説明する
- ステップ 7: EC2 フリートを作成する
- (オプション) ステップ 8: 未使用のキャパシティ予約の残りの数を表示する

ステップ 1: キャパシティー予約を作成する

キャパシティー予約の作成コマンドを使用してキャパシティ予約を作成します。3 つはus-east-1aの目的に、別の3つはus-east-1bの目的にします。アベイラビリティーゾーンを除き、キャパシティー予約の他の属性は同じです。

us-east-1aでの 3 つのキャパシティー予約

```
aws ec2 create-capacity-reservation \
    --availability-zone us-east-1a \
    --instance-type c5.xlarge \
    --instance-platform Linux/UNIX \
    --instance-count 3 \
    --instance-match-criteria targeted
```

キャパシティー予約 ID の結果の例

```
cr-1234567890abcdef1
```

us-east-1bでの 3 つのキャパシティー予約

```
aws ec2 create-capacity-reservation \
    --availability-zone us-east-1b \
    --instance-type c5.xlarge \
    --instance-platform Linux/UNIX \
    --instance-count 3 \
    --instance-match-criteria targeted
```

キャパシティー予約 ID の結果の例

```
cr-54321abcdef567890
```

ステップ 2: キャパシティー予約のリソースグループを作成する

resource-groupsサービスを使用する、およびグループを作成するコマンドを使用して、キャパシティー予約のリソースグループを作成します。この例では、プレイスメントグループ名はmy-cr-

groupです。リソースグループを作成する必要がある理由の詳細については、キャパシティ予約を使用して、EC2 フリートのオンデマンドキャパシティを予約するを参照してください。

```
aws resource-groups create-group \
    --name my-cr-group \
    --configuration '{"Type":"AWS::EC2::CapacityReservationPool"}'
 '{"Type":"AWS::ResourceGroups::Generic", "Parameters": [{"Name": "allowed-resource-
types", "Values": ["AWS::EC2::CapacityReservation"]}]}'
```

ステップ 3: キャパシティ予約リソースグループにキャパシティ予約を追加する

resource-groupsサービス、およびグループリソースコマンドを使用して、手順 1 で作成したキャパシティ予約をキャパシティ予約リソースグループに追加します。オンデマンドキャパシティ予約は、ARN ごとに参照する必要があります。

```
aws resource-groups group-resources \
    --group my-cr-group \
    --resource-arns \
      arn:aws:ec2:us-east-1:123456789012:capacity-reservation/cr-1234567890abcdef1 \
      arn:aws:ec2:us-east-1:123456789012:capacity-reservation/cr-54321abcdef567890
```

出力の例

```
{
   "Failed": [],
   "Succeeded": [
   "arn:aws:ec2:us-east-1:123456789012:capacity-reservation/cr-1234567890abcdef1",
   "arn:aws:ec2:us-east-1:123456789012:capacity-reservation/cr-54321abcdef567890"
   ]
}
```

(オプション) ステップ 4: リソースグループのキャパシティーの予約を表示する

resource-groupsサービス、およびグループリソースを表示するコマンドを使用して、キャパシティ予約を表示するリソースグループをオプションで記述します。

```
aws resource-groups list-group-resources --group my-cr-group
```

出力の例

```
{
```

```
    "ResourceIdentifiers": [
        {
            "ResourceType": "AWS::EC2::CapacityReservation",
            "ResourceArn": "arn:aws:ec2:us-east-1:123456789012:capacity-reservation/
cr-1234567890abcdef1"
        },
        {
            "ResourceType": "AWS::EC2::CapacityReservation",
            "ResourceArn": "arn:aws:ec2:us-east-1:123456789012:capacity-reservation/
cr-54321abcdef567890"
        }
    ]
}
```

ステップ 5: キャパシティ予約が特定のリソースグループをターゲットに指定する起動テンプレートを作成する

起動テンプレートを作成するコマンドを使用して、使用するキャパシティー予約を指定する起動テンプレートを作成します。この例では、フリートはtargetedキャパシティー予約を使用して、ソースグループに追加されます。したがって、起動テンプレートデータでは、キャパシティ予約が特定のリソースグループをターゲットに指定します。次の例で、プレイスメントグループ名はmy-launch-templateです。

```
aws ec2 create-launch-template \
    --launch-template-name my-launch-template \
    --launch-template-data \
        '{"ImageId": "ami-0123456789example",
          "CapacityReservationSpecification":
            {"CapacityReservationTarget":
                { "CapacityReservationResourceGroupArn": "arn:aws:resource-groups:us-
east-1:123456789012:group/my-cr-group" }
            }
        }'
```

(オプション) ステップ 6: 起動テンプレートを説明する

describe-launch-template-versions コマンドを使用して、オプションで、その設定を表示するための起動テンプレートを説明します。

```
aws ec2 describe-launch-template-versions --launch-template-name my-launch-template
```

出力の例

```
{
    "LaunchTemplateVersions": [
        {
            "LaunchTemplateId": "lt-01234567890example",
            "LaunchTemplateName": "my-launch-template",
            "VersionNumber": 1,
            "CreateTime": "2021-01-19T20:50:19.000Z",
            "CreatedBy": "arn:aws:iam::123456789012:user/Admin",
            "DefaultVersion": true,
            "LaunchTemplateData": {
                "ImageId": "ami-0947d2ba12ee1ff75",
                "CapacityReservationSpecification": {
                    "CapacityReservationTarget": {
                        "CapacityReservationResourceGroupArn": "arn:aws:resource-
groups:us-east-1:123456789012:group/my-cr-group"
                    }
                }
            }
        }
    ]
}
```

ステップ 7: EC2 フリートを作成する

起動するインスタンスの設定情報を指定する EC2 フリートを作成します。以下の EC2 フリート設定は、この例に関連する設定のみを示しています。起動テンプレート`my-launch-template`は、ステップ 5 で作成した起動テンプレートです。2 つのインスタンスプールがあり、それぞれ同じインスタンスタイプ (`c5.xlarge`)ですが、異なるアベイラビリティーゾーン (`us-east-1a`および`us-east-1b`)にあります。料金は、アベイラビリティーゾーンごとではなく、リージョンに対して定義されるため、インスタンスプールの料金は同じです。合計ターゲット容量は 10 で、デフォルトのターゲット容量タイプは`on-demand`です。オンデマンド配分戦略は`lowest-price`です。キャパシティ予約の使用戦略は`use-capacity-reservations-first`です。

> **Note**
>
> フリートタイプは`instant`である必要があります。他のフリートタイプは`use-capacity-reservations-first`をサポートしていません。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1"
            },
            "Overrides": [
               {
                   "InstanceType": "c5.xlarge",
                   "AvailabilityZone": "us-east-1a"
               },
               {
                    "InstanceType": "c5.xlarge",
                    "AvailabilityZone": "us-east-1b"
               }
            ]
        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 10,
        "DefaultTargetCapacityType": "on-demand"
    },
    "OnDemandOptions": {
        "AllocationStrategy": "lowest-price",
        "CapacityReservationOptions": {
             "UsageStrategy": "use-capacity-reservations-first"
         }
    },
    "Type": "instant"
}
```

上記の設定を使用してinstantフリートを作成すると、目標容量を満たすために以下の 10 個のインスタンスが起動されます。

- 次のように、6 つのオンデマンドインスタンスを起動するために、キャパシティ予約が最初に使用されます。
  - 3 つのオンデマンドインスタンスが、us-east-1aでの 3 つのc5.xlargetargetedキャパシティー予約で起動します。
  - 3 つのオンデマンドインスタンスが、us-east-1bでの 3 つのc5.xlargetargetedキャパシティー予約で起動します。

- ターゲット容量を満たすために、4 つの追加のオンデマンドインスタンスは、オンデマンド配分戦略 (この例では`lowest-price`) に従って通常のオンデマンド容量で起動します。ただし、プールの価格は同じであるため (価格はアベイラビリティーゾーンごとではなく、リージョンごとであるため)、フリートは残りの 4 つのオンデマンドインスタンスをいずれかのプールで起動します。

(オプション) ステップ 8: 未使用のキャパシティ予約の残りの数を表示する

フリートの起動後、describe-capacity-reservations を実行して、未使用のキャパシティー予約の残りの数を確認できます。この例では、以下のレスポンスが表示されます。これは、すべてのプール内のすべてのキャパシティ予約が使用されたことを示しています。

```
{ "CapacityReservationId":  "cr-111",
     "InstanceType":  "c5.xlarge",
     "AvailableInstanceCount":  0
}

 { "CapacityReservationId":  "cr-222",
     "InstanceType":  "c5.xlarge",
     "AvailableInstanceCount":  0
}
```

チュートリアル: キャパシティブロックにインスタンスを起動するように EC2 フリートを設定する

このチュートリアルでは実行すべきステップを段階的に解説しています。これらのステップを実行すると、EC2 フリートがキャパシティブロックでインスタンスを起動します。

ほとんどの場合、EC2 フリートリクエストのターゲットキャパシティは、ターゲットとするキャパシティブロック予約の使用可能なキャパシティ以下でなければなりません。キャパシティブロック予約の制限を超えるターゲットキャパシティリクエストは受理されません。ターゲットキャパシティリクエストがキャパシティブロック予約の制限を超えると、キャパシティブロック予約の制限を超えるキャパシティに対して、`Insufficient Capacity Exception` が発生します。

> **Note**
>
> キャパシティブロックの場合、EC2 フリートは希望するターゲットキャパシティの残りをオンデマンドインスタンスの起動にフォールバックしません。

EC2 フリートが利用可能なキャパシティブロック予約で要求されたターゲットキャパシティを満たすことができない場合、EC2 フリートは可能な限り多くの容量を満たし、起動できたインスタンスを返します。すべてのインスタンスがプロビジョニングされるまで EC2 フリート の呼び出しを繰り返すことができます。

EC2 フリートリクエストを設定したら、キャパシティブロック予約の開始日まで待つ必要があります。まだ開始されていないキャパシティブロックで EC2 フリートに起動するようリクエストすると、`Insufficient Capacity Error` が発生します。

キャパシティブロック予約が有効になったら、EC2 フリートの API コールを行い、選択したパラメータに基づいてキャパシティブロックにインスタンスをプロビジョニングできます。キャパシティブロックで実行されているインスタンスは、手動で停止あるいは終了するまで、またはキャパシティブロック予約が完了して Amazon EC2 がインスタンスを終了するまで実行され続けます。

キャパシティーブロックの詳細については、機械学習用のキャパシティブロック を参照してください。

考慮事項

- キャパシティブロックへのインスタンスの起動では、`instant` タイプの EC2 フリートリクエストのみがサポートされます。詳細については、「タイプが instant の EC2 フリートを設定する」を参照してください。
- 同じ EC2 フリートリクエストでは、複数のキャパシティブロックはサポートされません。
- `OnDemandTargetCapacity` または `SpotTargetCapacity` を使用しながら `DefaultTargetCapacity` として `capacity-block` を設定することはサポートされていません。
- `DefaultTargetCapacityType` が `capacity-block` に設定されている場合、`OnDemandOptions::CapacityReservationOptions` は提供できません。例外が発生します。

キャパシティブロックにインスタンスを起動するように EC2 フリートを設定するには

1. 起動テンプレートを作成します。

   起動テンプレートで、次の操作を行います。

   - `InstanceMarketOptionsRequest` の場合、`MarketType` を `capacity-block` に設定します。

- キャパシティブロックの予約をターゲットにするには、CapacityReservationID にキャパシティブロックの予約 ID を指定します。

起動テンプレートの名前とバージョンを書き留めます。この情報は、次のステップで使用します。

起動テンプレートの作成の詳細については、「Amazon EC2 起動テンプレートを作成する」を参照してください。

2. EC2 フリートを設定します。

EC2 フリート用に次の設定を使用して config.json ファイルを作成します。次の例では、リソース識別子を独自のリソース識別子に置き換えます。

EC2 フリートの設定の詳細については、「EC2 フリートの作成」を参照してください。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "CBR-launch-template",
                "Version": "1"
            },
            "Overrides": [
                {
                    "InstanceType": "p5.48xlarge",
                    "AvailabilityZone": "us-east-1a"
                },
            ]
        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 10,
        "DefaultTargetCapacityType": "capacity-block"
    },
    "Type": "instant"
}
```

3. フリートを起動します。

次の create-fleet コマンドを使用します。

```
aws ec2 create-fleet --cli-input-json file://config.json
```

詳細については、「EC2 フリートの作成」を参照してください。

チュートリアル: キャパシティ予約に対してインスタンスを起動するように EC2 フリートを設定する

このチュートリアルでは実行すべきステップを段階的に解説しています。これらのステップを実行すると、EC2 フリートが中断可能なキャパシティ予約に対してインスタンスを起動します。

中断可能なキャパシティ予約は、AWS 組織内のオンデマンドキャパシティ予約の所有者によって貸与された予備のキャパシティを表します。これらの予約では、キャパシティ所有者が 2 分間の中断通知でいつでもキャパシティを再利用できるため、中断可能なワークロードに適しています。その後、EC2 はインスタンスを終了します。中断可能なキャパシティ予約の詳細については、「中断可能なキャパシティ予約」を参照してください。

EC2 フリートリクエストのターゲットキャパシティは、ターゲットとする中断可能なキャパシティ予約の使用可能なキャパシティ以下でなければなりません。ターゲットキャパシティリクエストが中断可能なキャパシティ予約の使用可能なキャパシティを超えると、EC2 フリートはできるだけ多くのインスタンスを起動し、起動されたインスタンスを API レスポンスで報告します。

中断可能なキャパシティ予約で実行されているインスタンスは、手動で停止または終了するか、キャパシティ所有者がキャパシティを再利用するまで実行され続けます。キャパシティ所有者がキャパシティを再利用すると、Amazon EC2 はインスタンス終了の 2 分前に Amazon EventBridge 通知を送信します。

考慮事項

- キャパシティ予約へのインスタンスの起動については、`instant` タイプの EC2 フリートリクエストのみがサポートされます。
- `OnDemandTargetCapacity` または `SpotTargetCapacity` を使用する際にさらに `reserved-capacity` を `DefaultTargetCapacityType` として 設定することはサポートされていません。
- それぞれが異なる中断可能なキャパシティ予約をターゲットとする複数の起動テンプレートを指定すると、EC2 フリートは一致するすべての予約にインスタンスをプロビジョニングします。
- 中断可能なキャパシティ予約の場合、EC2 フリートは希望するターゲットキャパシティの残りをオンデマンドインスタンスまたはスポットインスタンスの起動にフォールバックしません。

アクセス許可の確認

EC2 フリート を作成する前に、必要なアクセス許可が含まれている IAM ロールがあることを確認します。詳細については、「EC2 フリートの前提条件」を参照してください。

中断可能なキャパシティ予約に対してインスタンスを起動するには、次の手順を実行する必要があります。

ステップ 1: 起動テンプレートを作成する

create-launch-template コマンドを使用して、ターゲットにする中断可能なキャパシティ予約が指定される起動テンプレートを作成します。起動テンプレートで、MarketType を interruptible-capacity-reservation に設定し、中断可能なキャパシティ予約の CapacityReservationId を指定します。

起動テンプレート設定の例:

```
{
    "LaunchTemplateName": "interruptible-cr-launch-template",
    "LaunchTemplateData": {
        "InstanceType": "m5.large",
        "ImageId": "ami-0abcdef1234567890",
        "CapacityReservationSpecification": {
            "CapacityReservationTarget": {
                "CapacityReservationId": "cr-0123456789abcdef0"
            }
        },
        "InstanceMarketOptions": {
            "MarketType": "interruptible-capacity-reservation"
        }
    }
}
```

出力の例

```
{
    "LaunchTemplate": {
        "LaunchTemplateId": "lt-0123456789example",
        "LaunchTemplateName": "interruptible-cr-launch-template",
        "CreateTime": "2026-03-12T10:00:00.000Z",
        "CreatedBy": "arn:aws:iam::123456789012:user/Admin",
        "DefaultVersionNumber": 1,
```

```
        "LatestVersionNumber": 1
    }
}
```

詳細については、「Amazon EC2 起動テンプレートを作成する」を参照してください。

ステップ 2: EC2 フリートを設定する

起動テンプレートとターゲットキャパシティが指定されるように EC2 フリートの設定ファイルを作成します。次の設定では、ステップ 1 で作成した起動テンプレート `interruptible-cr-launch-template` を使用します。

`reserved-capacity` を `DefaultTargetCapacityType` として使用する場合は、`ReservationType` が `interruptible-capacity-reservation` に設定されている `ReservedCapacityOptions` を指定する必要があります。

以下の内容で `config.json` という名前のファイルを作成します。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "interruptible-cr-launch-template",
                "Version": "1"
            },
            "Overrides": [
                {
                    "InstanceType": "m5.large",
                    "AvailabilityZone": "us-east-1a"
                }
            ]
        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 10,
        "DefaultTargetCapacityType": "reserved-capacity"
    },
    "ReservedCapacityOptions": {
        "ReservationType": ["interruptible-capacity-reservation"]
    },
    "Type": "instant"
}
```

主な設定パラメータ:

パラメータ	説明
DefaultTargetCapacityType	リザーブドキャパシティに対してインスタンスを起動する必要があることを示すには、reserved-capacity に設定します。
ReservedCapacityOptions	リザーブドキャパシティのタイプを指定します。中断可能なキャパシティ予約の場合は、ReservationType を ["interruptible-capacity-reservation"] に設定します。
Type	instant に設定する必要があります。中断可能なキャパシティ予約では、インスタントフリートのみがサポートされています。

ステップ 3: フリートを起動して結果を表示する

create-fleet コマンドを使用してフリートを作成します。

```
aws ec2 create-fleet \
    --cli-input-json file://config.json
```

上記の設定を使用して instant フリートを作成すると、EC2 フリートはターゲットキャパシティを満たすために、中断可能なキャパシティ予約に対して 10 個のインスタンスを起動します。

> **Note**
>
> フリートがターゲットキャパシティを完全には満たせない場合、レスポンスには、起動されたインスタンスと、満たされていないキャパシティに関するエラーが含まれます。

出力の例

```
{
    "FleetId": "fleet-12345678-1234-1234-1234-123456789012",
```

```
    "Instances": [
        {
            "LaunchTemplateAndOverrides": {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateId": "lt-0123456789example",
                    "Version": "1"
                },
                "Overrides": {
                    "InstanceType": "m5.large",
                    "AvailabilityZone": "us-east-1a"
                }
            },
            "Lifecycle": "interruptible-capacity-reservation",
            "InstanceIds": [
                "i-0123456789example1",
                "i-0123456789example2",
                "i-0123456789example3",
                "i-0123456789example4",
                "i-0123456789example5",
                "i-0123456789example6",
                "i-0123456789example7",
                "i-0123456789example8",
                "i-0123456789example9",
                "i-0123456789example0"
            ],
            "InstanceType": "m5.large",
            "Platform": "Linux/UNIX"
        }
    ],
    "Errors": []
}
```


詳細については、「EC2 フリートの作成」を参照してください。

クリーンアップ

料金の発生を防ぐため、不要になったインスタンスは終了させてください。また、キャパシティ所有者がキャパシティを再利用すると、EC2 は中断可能なキャパシティ予約に対して起動されたインスタンスを自動的に終了させるので注意してください。

関連リソース

- 中断可能なキャパシティ予約

- EC2 オンデマンドキャパシティ予約を使用してコンピューティングキャパシティを予約
- EC2 フリートの操作
- EC2 フリートの作成
- Amazon EC2 起動テンプレートにインスタンス起動パラメータを保存する

EC2 フリートの CLI 設定の例

以下のように、JSON ファイルに EC2 フリート設定を定義し、create-fleet コマンドでそのファイルを参照して、フリートを作成できます。

```
aws ec2 create-fleet --cli-input-json file://file_name.json
```

以下の例は、EC2 フリートのさまざまなユースケースの起動設定を示しています。設定パラメータの詳細については、「 コマンドリファレンス」の create-fleet を参照してください。

例

- 例 1: スポットインスタンスをデフォルト購入オプションとして起動する
- 例 2: オンデマンドインスタンスをデフォルト購入オプションとして起動する
- 例 3: オンデマンドインスタンスをプライマリ容量として起動する
- 例 4: 複数のキャパシティー予約を使用して オンデマンドインスタンス を起動する
- 例 5: 合計ターゲット容量が未使用キャパシティーの予約の数を超えたときに、キャパシティーの予約を使用してオンデマンドインスタンスを起動する
- 例 6: ターゲットのキャパシティー予約を使用してオンデマンドインスタンスを起動する
- 例 7: 容量の再調整を設定して代替スポットインスタンスを起動する
- 例 8: 容量最適化フリートでスポットインスタンスを起動する
- 例 9: 優先順位のある容量最適化フリートでスポットインスタンスを起動する
- 例 10: price-capaity-optimized フリートでスポットインスタンスを起動する
- 例 11: 属性ベースのインスタンスタイプの選択を設定する

タイプ `instant` のフリートに対する他の CLI の例については、「タイプが instant の EC2 フリートを設定する」を参照してください。

例 1: スポットインスタンスをデフォルト購入オプションとして起動する

次の例では、EC2 フリートで必要な最小限のパラメータ (起動テンプレート、ターゲットキャパシティ、デフォルト購入オプション) を指定します。起動テンプレートは、起動テンプレート ID とバージョン番号によって識別されます。フリートのターゲット容量は 2 インスタンスであり、デフォルト購入オプションは spot です。この結果、フリートは 2 スポットインスタンス を起動します。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateId": "lt-0e8c754449b27161c",
                "Version": "1"
            }

        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 2,
        "DefaultTargetCapacityType": "spot"
    }
}
```

例 2: オンデマンドインスタンスをデフォルト購入オプションとして起動する

次の例では、EC2 フリート で必要な最小限のパラメータ (起動テンプレート、ターゲット容量、デフォルト購入オプション) を指定します。起動テンプレートは、起動テンプレート ID とバージョン番号によって識別されます。フリートのターゲット容量は 2 インスタンスであり、デフォルト購入オプションは on-demand です。この結果、フリートは 2 オンデマンドインスタンス を起動します。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateId": "lt-0e8c754449b27161c",
                "Version": "1"
            }
```

```
        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 2,
        "DefaultTargetCapacityType": "on-demand"
    }
}
```

例 3: オンデマンドインスタンスをプライマリ容量として起動する

次の例では、フリートの合計ターゲット容量 2 インスタンス、ターゲット容量を 1 オンデマンドインスタンスとして指定します。デフォルト購入オプションは spot です。フリートは指定されたとおり 1 オンデマンドインスタンス を起動しますが、合計ターゲット容量を満たすために、さらに 1 つ以上のインスタンスを起動する必要があります。差額の購入オプションは、TotalTargetCapacity - OnDemandTargetCapacity = DefaultTargetCapacityType で計算されます。この結果、フリートはスポットインスタンスを起動します。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateId": "lt-0e8c754449b27161c",
                "Version": "1"
            }

        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 2,
        "OnDemandTargetCapacity": 1,
        "DefaultTargetCapacityType": "spot"
    }
}
```

例 4: 複数のキャパシティー予約を使用して オンデマンドインスタンス を起動する

キャパシティ予約の使用戦略を use-capacity-reservations-first に設定することで、オンデマンドインスタンスの起動時に最初にオンデマンドキャパシティ予約を使用するようにフリートを

設定できます。この例では、目標容量を満たすために必要以上のキャパシティ予約がある場合に、フリートが使用するキャパシティ予約を選択する方法を示します。

この例では、フリート設定は次のようになります。

- ターゲット容量:12 オンデマンドインスタンス
- 未使用のキャパシティー予約の合計:15 (フリートの目標容量である 12 オンデマンドインスタンスを超えています)
- キャパシティ予約プールの数:3 (m5.large、m4.xlarge、およびm4.2xlarge)
- プールあたりのキャパシティ予約数:5
- オンデマンド割り当て戦略：lowest-price(複数のインスタンスプールに未使用のキャパシティ予約が複数ある場合、フリートはオンデマンド割り当て戦略に基づいてオンデマンドインスタンスを起動するプールを決定します)。

  また、lowest-price割り当て戦略の代わりにprioritized割り当て戦略を使用することもできます。

キャパシティ予約

アカウントには、3 つの異なるプールに以下の 15 個の未使用のキャパシティ予約があります。各プールのキャパシティ予約の数は AvailableInstanceCount で示されます。

```
{
    "CapacityReservationId": "cr-111",
    "InstanceType": "m5.large",
    "InstancePlatform": "Linux/UNIX",
    "AvailabilityZone": "us-east-1a",
    "AvailableInstanceCount": 5,
    "InstanceMatchCriteria": "open",
    "State": "active"
}

{
    "CapacityReservationId": "cr-222",
    "InstanceType": "m4.xlarge",
    "InstancePlatform": "Linux/UNIX",
    "AvailabilityZone": "us-east-1a",
    "AvailableInstanceCount": 5,
    "InstanceMatchCriteria": "open",
    "State": "active"
```

```
}

{
    "CapacityReservationId": "cr-333",
    "InstanceType": "m4.2xlarge",
    "InstancePlatform": "Linux/UNIX",
    "AvailabilityZone": "us-east-1a",
    "AvailableInstanceCount":5,
    "InstanceMatchCriteria": "open",
    "State": "active"
}
```

フリート設定

以下のフリート設定は、この例に関連する設定のみを示しています。合計ターゲット容量は 12 で、デフォルトのターゲット容量タイプは on-demand です。オンデマンド配分戦略はlowest-priceです。キャパシティ予約の使用戦略はuse-capacity-reservations-firstです。

この例では、オンデマンドインスタンスの料金は以下のようになります。

- m5.large – 1 時間あたり 0.096 USD
- m4.xlarge – 1 時間あたり 0.20 USD
- m4.2xlarge – 1 時間あたり 0.40 USD

> **Note**
>
> フリートタイプはタイプinstantである必要があります。他のフリートタイプはuse-capacity-reservations-firstをサポートしていません。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateId": "lt-abc1234567example",
                "Version": "1"
            },
            "Overrides": [
                {
                  "InstanceType": "m5.large",
```

```
                    "AvailabilityZone": "us-east-1a",
                    "WeightedCapacity": 1
                },
                {
                    "InstanceType": "m4.xlarge",
                    "AvailabilityZone": "us-east-1a",
                    "WeightedCapacity": 1
                },
                {
                    "InstanceType": "m4.2xlarge",
                    "AvailabilityZone": "us-east-1a",
                    "WeightedCapacity": 1
                }
            ]

        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 12,
        "DefaultTargetCapacityType": "on-demand"
    },
    "OnDemandOptions": {
        "AllocationStrategy": "lowest-price",
        "CapacityReservationOptions": {
            "UsageStrategy": "use-capacity-reservations-first"
        }
    },
    "Type": "instant"
}
```

上記の設定を使用して instant フリートを作成すると、目標容量を満たすために以下の 12 個のインスタンスが起動されます。

- us-east-1aの中のus-east-1a – m5.largeにある 5 つのm5.largeオンデマンドインスタンスが最低価格です。そしてそこに 5 つの利用可能な未使用のm5.largeキャパシティー予約があります。
- 5 つのm4.xlargeオンデマンドインスタンス (us-east-1a) – m4.xlarge us-east-1aは次に低い料金であり、利用可能な未使用m4.xlargeキャパシティーの予約が 5 つあります。
- 2 つのm4.2xlargeオンデマンドインスタンス (us-east-1a) – m4.2xlarge us-east-1aは 3 番目に低い料金であり、利用可能な未使用m4.2xlargeキャパシティーの予約は 5 つあります。そのうちの 2 つのみが目標容量を満たすために必要です。

フリートの起動後、describe-capacity-reservations を実行して、未使用のキャパシティ予約の数を確認できます。この例では、以下のレスポンスが表示されます。これは、m5.largeおよびm4.xlargeのすべてのキャパシティーの予約が使用され、m4.2xlargeの 3 つのキャパシティーの予約が未使用のままであることを示しています。

```
{
    "CapacityReservationId": "cr-111",
    "InstanceType": "m5.large",
    "AvailableInstanceCount": 0
}

{
    "CapacityReservationId": "cr-222",
    "InstanceType": "m4.xlarge",
    "AvailableInstanceCount": 0
}

{
    "CapacityReservationId": "cr-333",
    "InstanceType": "m4.2xlarge",
    "AvailableInstanceCount": 3
}
```

例 5: 合計ターゲット容量が未使用キャパシティーの予約の数を超えたときに、キャパシティーの予約を使用してオンデマンドインスタンスを起動する

キャパシティ予約の使用戦略を use-capacity-reservations-first に設定することで、オンデマンドインスタンスの起動時に最初にオンデマンドキャパシティ予約を使用するようにフリートを設定できます。この例では、総ターゲット容量が使用可能な未使用のキャパシティ予約数を超えた場合に、オンデマンドインスタンスを起動するインスタンスプールをフリートがどのように選択するかを示します。

この例では、フリート設定は次のようになります。

- ターゲット容量:16 オンデマンドインスタンス
- 未使用キャパシティー予約の合計:15 (フリートのターゲット容量である 16 オンデマンドインスタンスを下回っています)
- キャパシティ予約プールの数:3 (m5.large、m4.xlarge、およびm4.2xlarge)

- プールあたりのキャパシティ予約数:5
- オンデマンド割り当て戦略：lowest-price(未使用キャパシティーの予約の数が目標オンデマンド容量より少ない場合、フリートは、オンデマンド配分戦略に基づいて、残りのオンデマンド容量を起動するプールを決定します)。

  また、lowest-price割り当て戦略の代わりにprioritized割り当て戦略を使用することもできます。

キャパシティ予約

アカウントには、3 つの異なるプールに以下の 15 個の未使用のキャパシティ予約があります。各プールのキャパシティ予約の数は AvailableInstanceCount で示されます。

```
{
    "CapacityReservationId": "cr-111",
    "InstanceType": "m5.large",
    "InstancePlatform": "Linux/UNIX",
    "AvailabilityZone": "us-east-1a",
    "AvailableInstanceCount": 5,
    "InstanceMatchCriteria": "open",
    "State": "active"
}

{
    "CapacityReservationId": "cr-222",
    "InstanceType": "m4.xlarge",
    "InstancePlatform": "Linux/UNIX",
    "AvailabilityZone": "us-east-1a",
    "AvailableInstanceCount": 5,
    "InstanceMatchCriteria": "open",
    "State": "active"
}

{
    "CapacityReservationId": "cr-333",
    "InstanceType": "m4.2xlarge",
    "InstancePlatform": "Linux/UNIX",
    "AvailabilityZone": "us-east-1a",
    "AvailableInstanceCount":5,
    "InstanceMatchCriteria": "open",
    "State": "active"
```

```
}
```

フリート設定

以下のフリート設定は、この例に関連する設定のみを示しています。合計ターゲット容量は 16 で、デフォルトのターゲット容量タイプは on-demand です。オンデマンド配分戦略はlowest-priceです。キャパシティ予約の使用戦略はuse-capacity-reservations-firstです。

この例では、オンデマンドインスタンスの料金は以下のようになります。

- m5.large – 0.096 USD/時間
- m4.xlarge – 0.20 USD/時間
- m4.2xlarge – 0.40 USD/時間

> **Note**
>
> フリートタイプはinstantである必要があります。他のフリートタイプはuse-capacity-reservations-firstをサポートしていません。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateId": "lt-0e8c754449b27161c",
                "Version": "1"
            },
            "Overrides": [
                {
                  "InstanceType": "m5.large",
                  "AvailabilityZone": "us-east-1a",
                  "WeightedCapacity": 1
                },
                {
                  "InstanceType": "m4.xlarge",
                  "AvailabilityZone": "us-east-1a",
                  "WeightedCapacity": 1
                },
                {
```

```
                    "InstanceType": "m4.2xlarge",
                    "AvailabilityZone": "us-east-1a",
                    "WeightedCapacity": 1
                }
            ]

        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 16,
        "DefaultTargetCapacityType": "on-demand"
    },
    "OnDemandOptions": {
        "AllocationStrategy": "lowest-price"
        "CapacityReservationOptions": {
            "UsageStrategy": "use-capacity-reservations-first"
        }
    },
    "Type": "instant",
}
```

上記の設定を使用して instant フリートを作成すると、目標容量を満たすために以下の 16 個のインスタンスが起動されます。

- 6 つのm5.largeオンデマンドインスタンス (us-east-1aのus-east-1a – m5.large) が最低価格です。そして 5 つの利用可能な未使用m5.largeキャパシティーの予約があります。5 つのオンデマンドインスタンスを起動するために、キャパシティ予約が最初に使用されます。残りのm4.xlargeおよびm4.2xlargeキャパシティーの予約を使用してターゲット容量を満たすために、追加のオンデマンドインスタンスは、オンデマンド配分戦略 (この例ではlowest-price) に従って起動します。
- 5 つのm4.xlargeオンデマンドインスタンス (us-east-1a の us-east-1a – m4.xlarge) が次に低い料金であり、5 つの利用可能な未使用m4.xlargeキャパシティーの予約があります。
- 5 つのm4.2xlargeオンデマンドインスタンス (us-east-1a の us-east-1a – m4.2xlarge) が 3 番目に低い料金であり、5 つの利用可能な未使用m4.2xlargeキャパシティーの予約があります。

フリートの起動後、describe-capacity-reservations を実行して、未使用のキャパシティ予約の数を確認できます。この例では、以下のレスポンスが表示されます。これは、すべてのプール内のすべてのキャパシティーの予約 が使用されたことを示しています。

```
{
    "CapacityReservationId": "cr-111",
    "InstanceType": "m5.large",
    "AvailableInstanceCount": 0
}

{
    "CapacityReservationId": "cr-222",
    "InstanceType": "m4.xlarge",
    "AvailableInstanceCount": 0
}

{
    "CapacityReservationId": "cr-333",
    "InstanceType": "m4.2xlarge",
    "AvailableInstanceCount": 0
}
```

例 6: ターゲットのキャパシティー予約を使用してオンデマンドインスタンスを起動する

キャパシティーの予約の使用戦略をuse-capacity-reservations-firstに設定することで、オンデマンドインスタンスの起動時に最初にtargetedオンデマンドキャパシティー予約を使用するようにフリートを設定できます。この例では、オンデマンドインスタンスをtargetedキャパシティ予約で起動する方法を示します。キャパシティ予約の属性は、アベイラビリティーゾーン (us-east-1aおよびus-east-1b) を除いて同じになります。また、総ターゲット容量が使用可能な未使用のキャパシティ予約数を超えた場合に、オンデマンドインスタンスを起動するインスタンスプールをフリートがどのように選択するかについても説明します。

この例では、フリート設定は次のようになります。

- ターゲット容量:10 オンデマンドインスタンス
- 未使用の合計targetedキャパシティー予約:6 (フリートの目標オンデマンド容量である 10 オンデマンドインスタンスを下回っています)
- キャパシティ予約のプールの数:2 (us-east-1aおよびus-east-1b)
- プールあたりのキャパシティ予約数:3
- オンデマンド割り当て戦略：lowest-price(未使用キャパシティーの予約の数が目標オンデマンド容量より少ない場合、フリートは、オンデマンド配分戦略に基づいて、残りのオンデマンド容量を起動するプールを決定します)。

また、lowest-price割り当て戦略の代わりにprioritized割り当て戦略を使用することもできます。

この例を実行するために必要な手順のチュートリアルについては、チュートリアル: ターゲットのキャパシティ予約を使用してオンデマンドインスタンスを起動するように EC2 フリートを設定するを参照してください。

キャパシティ予約

アカウントには、2 つの異なるプールに以下の 6 個の未使用キャパシティーの予約があります。この例では、プールはアベイラビリティーゾーンによって異なります。各プールのキャパシティ予約の数は AvailableInstanceCount で示されます。

```
{
    "CapacityReservationId": "cr-111",
    "InstanceType": "c5.xlarge",
    "InstancePlatform": "Linux/UNIX",
    "AvailabilityZone": "us-east-1a",
    "AvailableInstanceCount": 3,
    "InstanceMatchCriteria": "open",
    "State": "active"
}

{
    "CapacityReservationId": "cr-222",
    "InstanceType": "c5.xlarge",
    "InstancePlatform": "Linux/UNIX",
    "AvailabilityZone": "us-east-1b",
    "AvailableInstanceCount": 3,
    "InstanceMatchCriteria": "open",
    "State": "active"
}
```

フリート設定

以下のフリート設定は、この例に関連する設定のみを示しています。合計ターゲット容量は 10 で、デフォルトのターゲット容量タイプはon-demandです。オンデマンド配分戦略はlowest-priceです。キャパシティ予約の使用戦略はuse-capacity-reservations-firstです。

この例では、us-east-1でのc5.xlargeのオンデマンドインスタンスの料金は時間当たり 0.17 USD になります。

> **Note**
>
> フリートタイプはinstantである必要があります。他のフリートタイプはuse-capacity-reservations-firstをサポートしていません。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1"
            },
            "Overrides": [
               {
                   "InstanceType": "c5.xlarge",
                   "AvailabilityZone": "us-east-1a"
               },
               {
                    "InstanceType": "c5.xlarge",
                    "AvailabilityZone": "us-east-1b"
               }
            ]
        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 10,
        "DefaultTargetCapacityType": "on-demand"
    },
    "OnDemandOptions": {
        "AllocationStrategy": "lowest-price",
        "CapacityReservationOptions": {
             "UsageStrategy": "use-capacity-reservations-first"
         }
    },
    "Type": "instant"
}
```

上記の設定を使用してinstantフリートを作成すると、目標容量を満たすために以下の 10 個のインスタンスが起動されます。

- 次のように、6 つのオンデマンドインスタンスを起動するために、キャパシティ予約が最初に使用されます。
  - 3 つのオンデマンドインスタンスが、us-east-1aでの 3 つのc5.xlargetargetedキャパシティー予約で起動します。
  - 3 つのオンデマンドインスタンスが、us-east-1bでの 3 つのc5.xlargetargetedキャパシティー予約で起動します。
- ターゲット容量を満たすために、4 つの追加のオンデマンドインスタンスは、オンデマンド配分戦略 (この例ではlowest-price) に従って通常のオンデマンド容量で起動します。ただし、プールの価格は同じであるため (価格はアベイラビリティーゾーンごとではなく、リージョンごとであるため)、フリートは残りの 4 つのオンデマンドインスタンスをいずれかのプールで起動します。

フリートの起動後、describe-capacity-reservations を実行して、未使用のキャパシティ予約の数を確認できます。この例では、以下のレスポンスが表示されます。これは、すべてのプール内のすべてのキャパシティーの予約 が使用されたことを示しています。

```
{
    "CapacityReservationId": "cr-111",
    "InstanceType": "c5.xlarge",
    "AvailableInstanceCount": 0
}

{
    "CapacityReservationId": "cr-222",
    "InstanceType": "c5.xlarge",
    "AvailableInstanceCount": 0
}
```

例 7: 容量の再調整を設定して代替スポットインスタンスを起動する

次の例では、Amazon EC2 がフリートのスポットインスタンスに再調整に関する推奨を送信したときに、代替スポットインスタンスを起動するように EC2 フリートを設定します。スポットインスタンスの自動代替を設定するには、ReplacementStrategy で、launch-before-terminate を指定します。置換用の新しいスポットインスタンスが起動してから、古いスポットインスタンスが自動的に削除されるまでの時間を設定するには、termination-delay に値を秒単位で指定します。詳細については、「設定オプション」を参照してください。

> **Note**
>
> launch-before-terminate を使用するのは、インスタンスのシャットダウン処理にかかる時間を予測できる場合に限り、これらの処理が完了した後に古いインスタンスが終了するようにすることをお勧めします。実行中は、すべてのインスタンスに対して課金されます。

容量の再調整戦略の有効性は、EC2 フリートリクエストで指定されたスポットキャパシティプールの数に左右されます。インスタンスタイプとアベイラビリティーゾーンの多様なセットを使ってフリートを設定し、AllocationStrategy では capacity-optimized を指定することをお勧めします。EC2 フリート の容量の再調整を行う際に考慮すべき事項の詳細については、「」を参照してください。EC2 フリートとスポットフリートでキャパシティの再調整を使用して、リスクのあるスポットインスタンスを置き換える

```
{
    "ExcessCapacityTerminationPolicy": "termination",
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "LaunchTemplate",
                "Version": "1"
            },
                 "Overrides": [
                       {
                           "InstanceType": "c3.large",
                           "WeightedCapacity": 1,
                            "Placement": {
                               "AvailabilityZone": "us-east-1a"
                           }
                       },
                       {
                           "InstanceType": "c4.large",
                           "WeightedCapacity": 1,
                            "Placement": {
                               "AvailabilityZone": "us-east-1a"
                           }
                       },
                       {
                           "InstanceType": "c5.large",
                           "WeightedCapacity": 1,
                            "Placement": {
```

```
                                "AvailabilityZone": "us-east-1a"
                            }
                        }
                ]
            }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 5,
        "DefaultTargetCapacityType": "spot"
    },
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized",
        "MaintenanceStrategies": {
            "CapacityRebalance": {
                "ReplacementStrategy": "launch-before-terminate",
                "TerminationDelay": "720"
            }
        }
    }
}
```

例 8: 容量最適化フリートでスポットインスタンスを起動する

次の例は、容量を最適化するスポット配分戦略で、EC2 フリートを設定する方法を示しています。容量を最適化するには、AllocationStrategy を capacity-optimized に設定する必要があります。

次の例では、3 つの起動仕様で 3 つのスポットキャパシティプールが指定されています。ターゲット容量はスポットインスタンス 50 個です。EC2 フリートは、起動中のインスタンス数の最適な容量のスポットキャパシティープールに 50 個のスポットインスタンスを起動しようとします。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized",
        },
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1"
            },
                 "Overrides": [
```

```
                        {
                            "InstanceType": "r4.2xlarge",
                            "Placement": {
                                "AvailabilityZone": "us-west-2a"
                            },
                        },
                        {
                            "InstanceType": "m4.2xlarge",
                            "Placement": {
                                "AvailabilityZone": "us-west-2b"
                            },
                        },
                        {
                            "InstanceType": "c5.2xlarge",
                            "Placement": {
                                "AvailabilityZone": "us-west-2b"
                            }
                        }
                ]
        }
    ],
    "TargetCapacitySpecification": {
            "TotalTargetCapacity": 50,
            "DefaultTargetCapacityType": "spot"

    }
}
```

例 9: 優先順位のある容量最適化フリートでスポットインスタンスを起動する

次の例は、ベスト工フォートベースで優先順位を使用しながら、容量を最適化するスポット配分戦略を使用して、EC2 フリートを設定する方法を示しています。

capacity-optimized-prioritized 配分戦略を使用する場合は、Priority パラメータを使用して、スポットキャパシティプールの優先順位を指定します。数値が小さいほど優先順位が高くなります。また、優先度が同じならば、複数のスポットキャパシティープールに同じ優先順位を設定することもできます。プールに優先順位を設定しない場合、そのプールは優先順位が最も低いとみなされます。

スポットキャパシティプールに優先順位を付けるには、AllocationStrategy を capacity-optimized-prioritized に設定する必要があります。EC2 フリートは最初に容量を最適化しま

すが、インスタンスタイプの優先順位をベストエフォートベースで決定します (例えば、優先順位を尊重しても、EC2 フリートの最適な容量をプロビジョニングする能力に大きな影響を与えない場合など) 。これは、中断の可能性を最小限に抑える必要があり、特定のインスタンスタイプを優先することが重要なワークロードに適したオプションです。

次の例では、3 つの起動仕様で 3 つのスポットキャパシティープールが指定されています。各プールには優先順位が設定されています。数値が小さいほど優先順位が高くなります。ターゲット容量は 50 個のスポットインスタンスです。EC2 フリートは、ベストエフォートベースで優先順位が最も高いスポットキャパシティープールに 50 個のスポットインスタンスを起動しようとしますが、最初に容量を最適化します。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized-prioritized"
        },
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1"
            },
                 "Overrides": [
                        {
                           "InstanceType": "r4.2xlarge",
                           "Priority": 1,
                           "Placement": {
                               "AvailabilityZone": "us-west-2a"
                           },
                      },
                       {
                           "InstanceType": "m4.2xlarge",
                           "Priority": 2,
                           "Placement": {
                               "AvailabilityZone": "us-west-2b"
                           },
                       },
                       {
                           "InstanceType": "c5.2xlarge",
                           "Priority": 3,
                           "Placement": {
                               "AvailabilityZone": "us-west-2b"
                           }
```

```
                    }
                ]
            }
    ],
    "TargetCapacitySpecification": {
            "TotalTargetCapacity": 50,
            "DefaultTargetCapacityType": "spot"
}
```

例 10: price-capaity-optimized フリートでスポットインスタンスを起動する

次の例は、容量と価格の両方を最適化するスポット配分戦略で、EC2 フリートを設定する方法を示しています。価格を考慮しながら容量を最適化するには、スポット AllocationStrategy を price-capacity-optimized に設定する必要があります。

次の例では、3 つの起動仕様で 3 つのスポットキャパシティプールが指定されています。ターゲット容量は 50 個のスポットインスタンスです。EC2 フリートは、起動するインスタンス数に最適な容量を持つスポットキャパシティプールに 50 個のスポットインスタンスを起動し、同時に価格が最も低いプールを選択することを試みます。

```
{
    "SpotOptions": {
        "AllocationStrategy": "price-capacity-optimized",
        "MinTargetCapacity": 2,
        "SingleInstanceType": true
    },
    "OnDemandOptions": {
        "AllocationStrategy": "lowest-price"
    },
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1"
            },
                 "Overrides": [
                       {
                           "InstanceType": "r4.2xlarge",
                           "Placement": {
                               "AvailabilityZone": "us-west-2a"
                           },
                      },
```

```
                        {
                            "InstanceType": "m4.2xlarge",
                            "Placement": {
                                "AvailabilityZone": "us-west-2b"
                            },
                        },
                        {
                            "InstanceType": "c5.2xlarge",
                            "Placement": {
                                "AvailabilityZone": "us-west-2b"
                            }
                        }
                    ]
            }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 50,
        "OnDemandTargetCapacity":0,
        "SpotTargetCapacity":50,
        "DefaultTargetCapacityType": "spot"
    },
    "Type": "instant"
}
```

例 11: 属性ベースのインスタンスタイプの選択を設定する

次の例は、インスタンスタイプの識別に属性ベースのインスタンスタイプ選択を使用するように EC2 フリート を設定する方法を示しています。必要なインスタンス属性を指定するには、InstanceRequirements 構造に属性を指定します。

次の例では、2 つのインスタンス属性が指定されています。

- VCpuCount — 最低 2 つの vCPUs が指定されています。最大値は指定されていないため、上限はありません。
- MemoryMiB — 4 MiB 以上のメモリが指定されています。最大値は指定されていないため、上限はありません。

2 つ以上の vCPUs と 4 MiB 以上のメモリを持つすべてのインスタンスタイプが識別されます。ただし、EC2 フリートがフリートをプロビジョニングする場合、価格保護と配分戦略によって一部のインスタンスタイプが除外される場合があります。

指定できるすべての属性のリストと説明については、「Amazon EC2 API リファレンス」の「InstanceRequirements」を参照してください。

```
{
 "SpotOptions": {
  "AllocationStrategy": "price-capacity-optimized"
 },
 "LaunchTemplateConfigs": [{
  "LaunchTemplateSpecification": {
   "LaunchTemplateName": "my-launch-template",
   "Version": "1"
  },
  "Overrides": [{
   "InstanceRequirements": {
    "VCpuCount": {
     "Min": 2
    },
    "MemoryMiB": {
     "Min": 4
    }
   }
  }]
 }],
 "TargetCapacitySpecification": {
  "TotalTargetCapacity": 20,
  "DefaultTargetCapacityType": "spot"
 },
 "Type": "instant"
}
```

スポットフリートの CLI 設定の例

以下のように、JSON ファイルにスポットフリート設定を定義し、request-spot-fleet AWS CLI コマンドでそのファイルを参照して、フリートを作成できます。

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://file_name.json
```

以下の例は、スポットフリートのさまざまなユースケースの起動設定を示しています。設定パラメータの詳細については、「 コマンドリファレンス」の request-spot-fleet を参照してください。スポットフリートの作成の詳細については、「スポットフリートを作成する」を参照してください。

> **Note**
>
> スポットフリートでは、ネットワークインターフェイス ID を起動テンプレートか起動仕様に指定できません。起動テンプレートまたは起動仕様から `NetworkInterfaceID` パラメータを必ず省略してください。

例

- 例 1: リージョンの最低価格のアベイラビリティーゾーンを使用してスポットインスタンスを起動する
- 例 2: リストからの最低価格のアベイラビリティーゾーンを使用してスポットインスタンスを起動する
- 例 3: リストからの最低価格のインスタンスタイプを使用してスポットインスタンスを起動する
- 例 4。リクエストの料金を上書きする
- 例 5: 分散配分戦略を使用して、スポットフリートを起動する
- 例 6: インスタンスの分量指定を使用して、スポットフリートを起動する
- 例 7: オンデマンドキャパシティを使用してスポットフリートを起動する
- 例 8: 容量の再調整を設定して代替スポットインスタンスを起動する
- 例 9: 容量最適化フリートでスポットインスタンスを起動する
- 例 10: 優先順位のある容量最適化フリートでスポットインスタンスを起動する
- 例 11: priceCapacityOptimized フリートでスポットインスタンスを起動する
- 例 12: 属性ベースのインスタンスタイプの選択を設定する

例 1: リージョンの最低価格のアベイラビリティーゾーンを使用してスポットインスタンスを起動する

以下の例では、アベイラビリティーゾーンまたはサブネットを使用しない 1 つの起動仕様を指定しています。スポットフリートはデフォルトのサブネットを持つ最低価格のアベイラビリティーゾーンでインスタンスを起動します。お支払いいただく料金はオンデマンド価格を上回りません。

```
{
  "TargetCapacity": 20,
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
```

```
        {
            "ImageId": "ami-1a2b3c4d",
            "KeyName": "my-key-pair",
            "SecurityGroups": [
                {
                    "GroupId": "sg-1a2b3c4d"
                }
            ],
            "InstanceType": "m3.medium",
            "IamInstanceProfile": {
                "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
            }
        }
    ]
}
```

例 2: リストからの最低価格のアベイラビリティーゾーンを使用してスポットインスタンスを起動する

以下の例では、アベイラビリティーゾーン/サブネットは異なるがインスタンスタイプおよび AMI が同じである、2 つの起動仕様を指定しています。

アベイラビリティーゾーン

スポットフリートは、指定した最低価格のアベイラビリティーゾーンのデフォルトサブネットでインスタンスを起動します。

```
{
  "TargetCapacity": 20,
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "KeyName": "my-key-pair",
          "SecurityGroups": [
              {
                  "GroupId": "sg-1a2b3c4d"
              }
          ],
          "InstanceType": "m3.medium",
          "Placement": {
              "AvailabilityZone": "us-west-2a, us-west-2b"
```

```
            },
            "IamInstanceProfile": {
                "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
            }
        }
    ]
}
```

Subnets

デフォルトのサブネットまたはデフォルト以外のサブネットを指定できますが、デフォルト以外のサブネットは、デフォルトの VPC またはデフォルト以外の VPC 内から選択できます。スポットサービスは、最低価格のアベイラビリティーゾーンにあるいずれかのサブネットでインスタンスを起動します。

スポットフリートリクエストで、同じアベイラビリティーゾーンから異なるサブネットを指定することはできません。

```
{
  "TargetCapacity": 20,
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "KeyName": "my-key-pair",
          "SecurityGroups": [
              {
                  "GroupId": "sg-1a2b3c4d"
              }
          ],
          "InstanceType": "m3.medium",
          "SubnetId": "subnet-a61dafcf, subnet-65ea5f08",
          "IamInstanceProfile": {
              "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
          }
      }
  ]
}
```

インスタンスがデフォルトの VPC で起動される場合は、デフォルトでパブリック IPv4 アドレスが割り当てられます。インスタンスがデフォルト以外の VPC で起動される場合は、デフォルトでパブリック IPv4 アドレスは割り当てられません。起動仕様でネットワークインターフェイスを使用

して、デフォルト以外の VPC で起動されるインスタンスにパブリック IPv4 アドレスを割り当てます。ネットワークインターフェイスの指定時、ネットワークインターフェイスを使用してサブネット ID とセキュリティグループ ID を含める必要があります。

```
  ...
      {
          "ImageId": "ami-1a2b3c4d",
          "KeyName": "my-key-pair",
          "InstanceType": "m3.medium",
          "NetworkInterfaces": [
              {
                  "DeviceIndex": 0,
                  "SubnetId": "subnet-1a2b3c4d",
                  "Groups": [ "sg-1a2b3c4d" ],
                  "AssociatePublicIpAddress": true
              }
          ],
          "IamInstanceProfile": {
              "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
          }
      }
  ...
```

例 3: リストからの最低価格のインスタンスタイプを使用してスポットインスタンスを起動する

次の例では、同じ AMI と アベイラビリティーゾーンまたはサブネットで、複数の異なるインスタンスタイプを使用する 2 つの起動設定を指定します。スポットフリートは、指定された最低価格のインスタンスタイプを使用してインスタンスを起動します。

アベイラビリティーゾーン

```
{
  "TargetCapacity": 20,
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "SecurityGroups": [
              {
                  "GroupId": "sg-1a2b3c4d"
```

```
                }
            ],
            "InstanceType": "c5.4xlarge",
            "Placement": {
              "AvailabilityZone": "us-west-2b"
            }
        },
        {
            "ImageId": "ami-1a2b3c4d",
            "SecurityGroups": [
                {
                    "GroupId": "sg-1a2b3c4d"
                }
            ],
            "InstanceType": "r3.8xlarge",
            "Placement": {
                "AvailabilityZone": "us-west-2b"
            }
        }
    ]
}
```

サブネット

```
{
  "TargetCapacity": 20,
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "SecurityGroups": [
              {
                  "GroupId": "sg-1a2b3c4d"
              }
          ],
          "InstanceType": "c5.4xlarge",
          "SubnetId": "subnet-1a2b3c4d"
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "SecurityGroups": [
              {
                  "GroupId": "sg-1a2b3c4d"
```

```
            }
        ],
        "InstanceType": "r3.8xlarge",
        "SubnetId": "subnet-1a2b3c4d"
    }
  ]
}
```

例 4。リクエストの料金を上書きする

オンデマンド価格であるデフォルトの上限料金を使用することをお勧めします。必要に応じて、フリートリクエストの上限料金と個々の起動条件の上限料金を指定することができます。

以下の例は、フリートリクエストの上限料金と、3 つの起動条件のうちの 2 つの上限料金を指定しています。フリートリクエストの上限料金は、上限料金を指定しないすべての起動条件に適用されます。スポットフリートは、最低価格のインスタンスタイプを使用してインスタンスを起動します。

アベイラビリティーゾーン

```
{
  "SpotPrice": "1.00",
  "TargetCapacity": 30,
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c3.2xlarge",
          "Placement": {
              "AvailabilityZone": "us-west-2b"
          },
          "SpotPrice": "0.10"
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c3.4xlarge",
          "Placement": {
              "AvailabilityZone": "us-west-2b"
          },
          "SpotPrice": "0.20"
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c3.8xlarge",
```

```
            "Placement": {
                "AvailabilityZone": "us-west-2b"
            }
        }
    ]
}
```

サブネット

```
{
  "SpotPrice": "1.00",
  "TargetCapacity": 30,
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c3.2xlarge",
          "SubnetId": "subnet-1a2b3c4d",
          "SpotPrice": "0.10"
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c3.4xlarge",
          "SubnetId": "subnet-1a2b3c4d",
          "SpotPrice": "0.20"
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c3.8xlarge",
          "SubnetId": "subnet-1a2b3c4d"
      }
  ]
}
```

例 5: 分散配分戦略を使用して、スポットフリートを起動する

次の例では、diversified の配分戦略を使用します。これらの起動仕様では、インスタンスタイプは異なりますが、AMI およびアベイラビリティーゾーン/サブネットは同じです。スポットフリートは、各タイプのインスタンスが 10 個になるように、3 個の起動仕様全体に 30 個のインスタンスを分散します。詳細については、「配分戦略を使用して、EC2 フリートまたはスポットフリートがどのようにスポットキャパシティとオンデマンドキャパシティを満たすかを決定します。」を参照してください。

アベイラビリティーゾーン

```
{
  "SpotPrice": "0.70",
  "TargetCapacity": 30,
  "AllocationStrategy": "diversified",
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c4.2xlarge",
          "Placement": {
              "AvailabilityZone": "us-west-2b"
          }
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "m3.2xlarge",
          "Placement": {
              "AvailabilityZone": "us-west-2b"
          }
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "r3.2xlarge",
          "Placement": {
              "AvailabilityZone": "us-west-2b"
          }
      }
  ]
}
```

サブネット

```
{
    "SpotPrice": "0.70",
    "TargetCapacity": 30,
    "AllocationStrategy": "diversified",
    "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
    "LaunchSpecifications": [
        {
            "ImageId": "ami-1a2b3c4d",
            "InstanceType": "c4.2xlarge",
```

```
            "SubnetId": "subnet-1a2b3c4d"
        },
        {
            "ImageId": "ami-1a2b3c4d",
            "InstanceType": "m3.2xlarge",
            "SubnetId": "subnet-1a2b3c4d"
        },
        {
            "ImageId": "ami-1a2b3c4d",
            "InstanceType": "r3.2xlarge",
            "SubnetId": "subnet-1a2b3c4d"
        }
    ]
}
```

アベイラビリティーゾーンの 1 つで機能停止が発生した場合にスポットリクエストが EC2 のキャパシティーによって満たされる可能性を高めるためのベストプラクティスは、ゾーン間で多様化することです。このシナリオでは、使用可能な各アベイラビリティーゾーンを起動仕様に含めます。また、毎回同じサブネットを使用するのではなく、3 つの固有のサブネット (それぞれ異なるゾーンへのマッピング) を使用してください。

アベイラビリティーゾーン

```
{
  "SpotPrice": "0.70",
  "TargetCapacity": 30,
  "AllocationStrategy": "diversified",
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c4.2xlarge",
          "Placement": {
              "AvailabilityZone": "us-west-2a"
          }
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "m3.2xlarge",
          "Placement": {
              "AvailabilityZone": "us-west-2b"
          }
      },
```

```
        {
            "ImageId": "ami-1a2b3c4d",
            "InstanceType": "r3.2xlarge",
            "Placement": {
                "AvailabilityZone": "us-west-2c"
            }
        }
    ]
}
```

サブネット

```
{
    "SpotPrice": "0.70",
    "TargetCapacity": 30,
    "AllocationStrategy": "diversified",
    "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
    "LaunchSpecifications": [
        {
            "ImageId": "ami-1a2b3c4d",
            "InstanceType": "c4.2xlarge",
            "SubnetId": "subnet-1a2b3c4d"
        },
        {
            "ImageId": "ami-1a2b3c4d",
            "InstanceType": "m3.2xlarge",
            "SubnetId": "subnet-2a2b3c4d"
        },
        {
            "ImageId": "ami-1a2b3c4d",
            "InstanceType": "r3.2xlarge",
            "SubnetId": "subnet-3a2b3c4d"
        }
    ]
}
```

例 6: インスタンスの分量指定を使用して、スポットフリートを起動する

次の例では、インスタンス分量指定を使っています。これは、料金が 1 インスタンス時間当たりではなく、1 ユニット時間当たりであることを意味します。それぞれの起動設定には、異なるインスタンスタイプおよび異なる分量がリストされます。スポットフリートはユニット時間の最低価格のインスタンスタイプを選択してください。スポットフリートは、ターゲット容量をインスタンス分量で割

ることで起動するスポットインスタンス数を計算します。結果が整数でない場合、スポットフリートはその数を次の整数に切り上げ、そのためフリートのサイズがターゲット容量以上になります。

r3.2xlarge のリクエストが成功すると、スポットはこれらのインスタンスのうち、4 つをプロビジョニングします。3.33 インスタンスまで 20 を 6 で割り、そして残りの 4 つのインスタンスを切り上げます。

c3.xlarge のリクエストが成功すると、スポットはこれらのインスタンスのうち、7 つをプロビジョニングします。6.66 インスタンスまで 20 を 3 で割り、そして残りの 7 つのインスタンスを切り上げます。

詳細については、「インスタンスの重み付けを使用して、EC2 フリートまたはスポットフリートのコストとパフォーマンスを管理する」を参照してください。

アベイラビリティーゾーン

```
{
  "SpotPrice": "0.70",
  "TargetCapacity": 20,
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "r3.2xlarge",
          "Placement": {
              "AvailabilityZone": "us-west-2b"
          },
          "WeightedCapacity": 6
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c3.xlarge",
          "Placement": {
              "AvailabilityZone": "us-west-2b"
          },
          "WeightedCapacity": 3
      }
    ]
}
```

サブネット

```
{
```

```
  "SpotPrice": "0.70",
  "TargetCapacity": 20,
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "r3.2xlarge",
          "SubnetId": "subnet-1a2b3c4d",
          "WeightedCapacity": 6
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c3.xlarge",
          "SubnetId": "subnet-1a2b3c4d",
          "WeightedCapacity": 3
      }
  ]
}
```

例 7: オンデマンドキャパシティを使用してスポットフリートを起動する

インスタンス容量を常に確保するには、オンデマンド容量のリクエストをスポットフリートリクエストに含めることができます。オンデマンドリクエストは、容量がある限り、常に実行されます。ターゲット容量は、キャパシティーと可用性がある場合にスポットとして実行されます。

次の例では、希望するターゲット容量を 10 とし、そのうち 5 をオンデマンドキャパシティーとして指定する必要があります。スポットキャパシティーは指定しません。これは、ターゲット容量からオンデマンド容量を引いたバランスを意味します。Amazon EC2 は、利用可能な Amazon EC2 容量および可用性がある場合、オンデマンドとして 5 容量単位を、スポットとして 5 容量単位 (10-5=5) をスポットとして起動します。

```
{
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "AllocationStrategy": "lowestPrice",
  "TargetCapacity": 10,
  "SpotPrice": null,
  "ValidFrom": "2018-04-04T15:58:13Z",
  "ValidUntil": "2019-04-04T15:58:13Z",
  "TerminateInstancesWithExpiration": true,
  "LaunchSpecifications": [],
  "Type": "maintain",
  "OnDemandTargetCapacity": 5,
```

```
  "LaunchTemplateConfigs": [
    {
      "LaunchTemplateSpecification": {
        "LaunchTemplateId": "lt-0dbb04d4a6cca5ad1",
        "Version": "2"
      },
      "Overrides": [
        {
          "InstanceType": "t2.medium",
          "WeightedCapacity": 1,
          "SubnetId": "subnet-d0dc51fb"
        }
      ]
    }
  ]
}
```

例 8: 容量の再調整を設定して代替スポットインスタンスを起動する

次の例では、Amazon EC2 がフリートのスポットインスタンスに再調整の推奨を発したときに、スポットフリートが代替スポットインスタンスを起動するように設定します。スポットインスタンスの自動代替を設定するには、ReplacementStrategy で、launch-before-terminate を指定します。新しい交換用スポットインスタンスが起動してから古いスポットインスタンスが自動削除されるまでの時間を設定するには、termination-delay に秒単位で値を指定します。詳細については、「設定オプション」を参照してください。

> **Note**
>
> launch-before-terminate は、インスタンスのシャットダウン手順が完了するまでの時間が予測できる場合にのみ使用することをお勧めします。これにより、古いインスタンスは、シャットダウン手順が完了した後にのみ終了されます。実行中は、すべてのインスタンスに対して課金されます。

容量の再調整戦略の有効性は、スポットフリートリクエストで指定されたスポットキャパシティプールの数に左右されます。インスタンスタイプとアベイラビリティーゾーンの多様なセットを使ってフリートを設定し、AllocationStrategy では capacityOptimized を指定することをお勧めします。スポットフリートの容量の再調整を行うときに考慮すべき事項の詳細については、「EC2 フリートとスポットフリートでキャパシティの再調整を使用して、リスクのあるスポットインスタンスを置き換える」を参照してください。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "capacityOptimized",
        "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "LaunchTemplate",
                    "Version": "1"
                },
                 "Overrides": [
                       {
                           "InstanceType": "c3.large",
                           "WeightedCapacity": 1,
                           "Placement": {
                               "AvailabilityZone": "us-east-1a"
                           }
                       },
                       {
                           "InstanceType": "c4.large",
                           "WeightedCapacity": 1,
                           "Placement": {
                               "AvailabilityZone": "us-east-1a"
                           }
                       },
                       {
                           "InstanceType": "c5.large",
                           "WeightedCapacity": 1,
                           "Placement": {
                               "AvailabilityZone": "us-east-1a"
                           }
                       }
                ]
          }
    ],
        "TargetCapacity": 5,
        "SpotMaintenanceStrategies": {
            "CapacityRebalance": {
                "ReplacementStrategy": "launch-before-terminate",
                "TerminationDelay": "720"
            }
        }
```

```
    }
}
```

例 9: 容量最適化フリートでスポットインスタンスを起動する

以下の例は、容量を最適化するスポット配分戦略で、スポットフリートを設定する方法を示しています。容量を最適化するには、AllocationStrategy を capacityOptimized に設定する必要があります。

次の例では、3 つの起動仕様で 3 つのスポットキャパシティプールが指定されています。ターゲット容量は 50 個のスポットインスタンスです。スポットインスタンスは、起動中のインスタンス数に最適な容量のスポットキャパシティプールに、50 個のスポットインスタンスを起動しようとします。

```
{
    "TargetCapacity": "50",
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "capacityOptimized",
    },
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1"
            },
            "Overrides": [
                {
                    "InstanceType": "r4.2xlarge",
                    "AvailabilityZone": "us-west-2a"
                },
                {
                    "InstanceType": "m4.2xlarge",
                    "AvailabilityZone": "us-west-2b"
                },
                {
                    "InstanceType": "c5.2xlarge",
                    "AvailabilityZone": "us-west-2b"
                }
            ]
        }
    ]
}
```

例 10: 優先順位のある容量最適化フリートでスポットインスタンスを起動する

次の例は、ベストエフォートベースで優先順位を使用しながら、容量を最適化するスポット配分戦略を使用して、スポットフリートを設定する方法を示しています。

capacityOptimizedPrioritized 配分戦略を使用する場合は、Priority パラメータを使用して、スポットキャパシティプールの優先順位を指定します。数値が小さいほど優先順位が高くなります。また、優先度が同じならば、複数のスポットキャパシティープールに同じ優先順位を設定することもできます。プールに優先順位を設定しない場合、そのプールは優先順位が最も低いとみなされます。

スポットキャパシティプールに優先順位を付けるには、AllocationStrategy を capacityOptimizedPrioritized に設定する必要があります。スポットフリートは最初に容量を最適化しますが、優先順位をベストエフォートベースで決定します (例えば、優先順位を尊重しても、スポットフリートの最適な容量をプロビジョニングする能力に大きな影響を与えない場合など)。これは、中断の可能性を最小限に抑える必要があり、特定のインスタンスタイプを優先することが重要なワークロードに適したオプションです。

次の例では、3 つの起動仕様で 3 つのスポットキャパシティープールが指定されています。各プールには優先順位が設定されています。数値が小さいほど優先順位が高くなります。ターゲット容量は 50 個のスポットインスタンスです。スポットフリートは、ベストエフォートベースで優先順位が最も高いスポットキャパシティープールに 50 個のスポットインスタンスを起動しようとしますが、最初に容量を最適化します。

```
{
    "TargetCapacity": "50",
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "capacityOptimizedPrioritized"
    },
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1"
            },
            "Overrides": [
                {
                    "InstanceType": "r4.2xlarge",
                   "Priority": 1,
```

```
                    "AvailabilityZone": "us-west-2a"
                },
                {
                            "InstanceType": "m4.2xlarge",
                            "Priority": 2,
                            "AvailabilityZone": "us-west-2b"
                },
                {
                            "InstanceType": "c5.2xlarge",
                            "Priority": 3,
                            "AvailabilityZone": "us-west-2b"
                }
            ]
        }
    ]
}
```

例 11: priceCapacityOptimized フリートでスポットインスタンスを起動する

次の例は、容量と最低価格の両方を最適化するスポット配分戦略を使用するスポットフリートを設定する方法を示しています。価格を考慮しながら容量を最適化するには、スポット AllocationStrategy を priceCapacityOptimized に設定する必要があります。

次の例では、3 つの起動仕様で 3 つのスポットキャパシティプールが指定されています。ターゲット容量は 50 個のスポットインスタンスです。スポットフリートは、起動するインスタンス数に最適な容量を持つスポットキャパシティプールに 50 個のスポットインスタンスを起動し、同時に価格が最も低いプールを選択することを試みます。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "priceCapacityOptimized",
        "OnDemandAllocationStrategy": "lowestPrice",
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::111111111111:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateId": "lt-0123456789example",
                    "Version": "1"
                },
```

```
                "Overrides": [
                     {
                          "InstanceType": "r4.2xlarge",
                          "AvailabilityZone": "us-west-2a"
                     },
                     {
                          "InstanceType": "m4.2xlarge",
                          "AvailabilityZone": "us-west-2b"
                     },
                     {
                          "InstanceType": "c5.2xlarge",
                          "AvailabilityZone": "us-west-2b"
                     }
                ]
            }
        ],
        "TargetCapacity": 50,
        "Type": "request"
    }
}
```

例 12: 属性ベースのインスタンスタイプの選択を設定する

次の例は、インスタンスタイプの識別に属性ベースのインスタンスタイプ選択を使用するようにスポットフリートを設定する方法を示しています。必要なインスタンス属性を指定するには、InstanceRequirements 構造に属性を指定します。

次の例では、2 つのインスタンス属性が指定されています。

- VCpuCount — 最低 2 つの vCPUs が指定されています。最大値は指定されていないため、上限はありません。
- MemoryMiB — 4 MiB 以上のメモリが指定されています。最大値は指定されていないため、上限はありません。

2 つ以上の vCPUs と 4 MiB 以上のメモリを持つすべてのインスタンスタイプが識別されます。ただし、スポットフリートがフリートをプロビジョニングする場合、価格保護と配分戦略によって一部のインスタンスタイプが除外される場合があります。

指定できるすべての属性のリストと説明については、「Amazon EC2 API リファレンス」の「InstanceRequirements」を参照してください。

```
{
 "AllocationStrategy": "priceCapacityOptimized",
 "TargetCapacity": 20,
 "Type": "request",
 "LaunchTemplateConfigs": [{
  "LaunchTemplateSpecification": {
   "LaunchTemplateName": "my-launch-template",
   "Version": "1"
  },
  "Overrides": [{
   "InstanceRequirements": {
    "VCpuCount": {
     "Min": 2
    },
    "MemoryMiB": {
     "Min": 4
    }
   }
  }]
 }]
}
```

EC2 フリートとスポットフリートのクォータ

AWS アカウント には、AWS のサービスごとにデフォルトのクォータ (以前は制限と呼ばれていました) があります。特に明記していない限り、クォータはリージョン固有です。

通常の Amazon EC2 のクォータは、EC2 フリートまたはスポットフリートが起動したインスタンスの、スポットインスタンスの上限やボリュームの制限などに適用されます。??????

また、AWS アカウントには EC2 フリートとスポットフリートに関連するクォータとして以下のものがあります。

クォータの説明	クォータ
active、deleted_running 、および cancelled_running 状態の、タイプ maintain および request であるリージョンあたりの EC2 フリートおよびスポットフリートの数	1,000 [1] [2] [3]

クォータの説明	クォータ
タイプ instant である EC2 フリートの数。	無制限
EC2 フリートと、タイプ maintain および request のスポットフリートのスポットキャパシティプール (インスタンスタイプとサブネットの一意の組み合わせ) の数	300 [1]
タイプ instant の EC2 フリートのスポットキャパシティプール (インスタンスタイプとサブネットの一意の組み合わせ) の数	無制限
起動仕様内のユーザーデータのサイズ	16 KB [2]
EC2 フリートまたはスポットフリートあたりのターゲットキャパシティ	10,000
リージョン内のすべての EC2 フリートおよびスポットフリートにおけるターゲットキャパシティ	100,000 [1]
EC2 フリートリクエストまたはスポットフリートリクエストは、リージョンにまたがることはできません。	
EC2 フリートリクエストまたはスポットフリートリクエストは、同じアベイラビリティーゾーンからの複数の異なるサブネットにまたがることはできません。	

[1] これらのクォータは、EC2 フリートとスポットフリートの両方に適用されます。

[2] これらはハードクォータです。これらのクォータについては、引き上げをリクエストできません。

[3] EC2 フリートを削除した後、またはスポットフリートリクエストをキャンセルした後、リクエストを削除またはキャンセルしたときにフリートがスポットインスタンスを終了しないように指定していた場合、フリートリクエストは deleted_running (EC2 フリート) または cancelled_running (スポットフリート) 状態になり、インスタンスは中断されるか手動で終了されるまで引き続き実行

されます。インスタンスを終了する場合、フリートリクエストは deleted_terminating (EC2 フリート) または cancelled_terminating (スポットフリート) 状態になるため、このクォータにはカウントされません。詳細については、「EC2 フリートリクエストおよびフリート内のインスタンスを削除する」および「スポットフリートリクエストをキャンセル (削除) します。」を参照してください。

ターゲットキャパシティのクォータ引き上げをリクエストします

ターゲット容量のデフォルトクォータを超える容量が必要な場合は、クォータの引き上げをリクエストできます。

ターゲットキャパシティのクォータ引き上げをリクエストするには

1. サポート中央の [Create case] (ケースの作成) フォームを開いてください。
2. [Service Limit increase] (サービス制限の緩和) を選択してください。
3. [Limit type] (制限タイプ) には、[EC2 Fleet] (EC2 フリート) を選択してください。
4. [Region] (リージョン) には、クォータの増加をリクエストする AWS リージョンを選択してください。
5. [Limit] (制限) には、どちらのクォータを増やしたいかに応じて、[Target Fleet Capacity per Fleet (in units)] (フリートごとのターゲットフリート容量 (ユニット))、または [Target Fleet Capacity per Region (in units)] (リージョンごとのターゲットフリート容量 (ユニット)) のいずれかを選択してください。
6. [新しい制限値] (New limit value) の場合、任意の値を入力してください。
7. 別のクォータの引き上げを要求するには、[Add another request] (別のリクエストを追加) を選択し、ステップ 4 ~ 6 を繰り返します。
8. [Use case description] (ユースケースの説明) には、クォータの引き上げをリクエストする理由を入力してください。
9. [Contact options] (連絡先オプション)で、希望する連絡言語と連絡方法を指定します。
10. [Submit] を選択してください。

Amazon EC2におけるネットワーク

Amazon VPC を使用すると、Virtual Private Cloud (VPC) と呼ばれる AWS アカウント専用の仮想ネットワークに対して Amazon EC2 インスタンスなどの AWS リソースを起動できます。インスタンスを起動するときに、VPC からサブネットを選択できます。インスタンスには、論理的な仮想ネットワークカードであるプライマリネットワークインターフェイスが設定されています。インスタンスは、サブネットの IPv4 アドレスからプライマリプライベート IP アドレスを受け取ります。そのアドレスは、プライマリネットワークインターフェイスに割り当てられます。

インスタンスが Amazon のパブリック IP アドレスのプールからパブリック IP アドレスを受け取るかどうかをコントロールできます。インスタンスのパブリック IP アドレスは、インスタンスが停止または終了するまでに限り、インスタンスに関連付けられます。永続的なパブリック IP アドレスが必要な場合は、AWS アカウントに Elastic IP アドレスを割り当て、インスタンスまたはネットワークインターフェイスに関連付けることができます。Elastic IP アドレスは、ユーザーが AWS アカウントをリリースするまでアカウントに関連付けられたままであり、必要に応じてインスタンス間でそのアドレスを移動できます。独自の IP アドレスの範囲を AWS アカウントに持ち込み、アドレスプールとして表示して、そこから Elastic IP アドレスを割り当てることができます。

ネットワークのパフォーマンスを向上させ、レイテンシーを低減するために、プレイスメントグループ内でインスタンスを起動できます。拡張ネットワーキングを使用すると、1 秒あたりのパケット (PPS) のパフォーマンスが大幅に向上します。Elastic Fabric Adapter (EFA) を使用すると、ハイパフォーマンスコンピューティングおよび機械学習アプリケーションを高速化できます。EFA は、サポートされているインスタンスタイプにアタッチできるネットワークデバイスです。

機能

- リージョンとゾーン
- Amazon EC2 インスタンスの IP アドレス指定
- EC2 インスタンスのホスト名とドメイン
- Amazon EC2 に自分の IP アドレスを持ち込む (BYOIP)
- Elastic IP アドレス
- Elastic Network Interface
- Amazon EC2 インスタンスのネットワーク帯域幅
- Amazon EC2 インスタンスでの拡張ネットワーキング
- Amazon EC2 の AI/ML および HPC ワークロード用の Elastic Fabric Adapter

- Amazon EC2 トポロジー
- Amazon EC2 インスタンスのプレイスメントグループ
- EC2 インスタンスのネットワークの最大送信単位 (MTU)
- EC2 インスタンスの仮想プライベートクラウド
- セカンダリネットワーク

リージョンとゾーン

Amazon EC2 は世界各地の場所でホスティングされています。これらの場所はAWS リージョン、アベイラビリティーゾーン、Local Zones、AWS Outposts、および Wavelength Zones で構成されます。

- リージョンは別々の地理的エリアです。
- アベイラビリティーゾーンは、各リージョン内の複数の独立した場所です。
- Local Zones を使用すると、コンピューティングやストレージなどのリソースをエンドユーザーに近い複数の場所に配置できます。
- Wavelength Zone を使用すると、5G デバイスやエンドユーザーに非常に低いレイテンシーを提供するアプリケーションを構築できます。Wavelength は標準の AWS コンピューティングおよびストレージサービスを通信事業者の 5G ネットワークのエッジにデプロイします。
- AWS Outposts ではネイティブの AWS サービス、インフラストラクチャ、運用モデルを、ほぼすべてのデータセンター、コロケーションスペース、オンプレミスの施設で利用できます。

AWS は最新の高可用性のデータセンターを運用しています。しかし、非常にまれですが、同じ場所にあるインスタンスすべての可用性に影響する障害が発生することもあります。すべてのインスタンスを 1 か所でホストしている場合、そのような障害が起きると、すべてのインスタンスが利用できなくなります。

詳細についてはAWS グローバルインフラストラクチャを参照してください。

内容

- リージョン
- アベイラビリティーゾーン
- Local Zones
- Wavelength Zone

- AWS Outposts

リージョン

各リージョンは他のリージョンと完全に分離されるように設計されています。これにより、最大限の耐障害性と安定性が達成されます。

インスタンスを起動するときは、インスタンスと特定のお客様の近くに配置したり、法的要件やその他の要件を満たしたりするリージョンを選択してください。複数のリージョンでインスタンスを起動できます。

リソースを表示すると、指定した リージョンに結び付けられているリソースのみが表示されます。これはリージョンが相互に分離されており、リージョン間ではリソースが自動的にレプリケートされないためです。

利用できるリージョン

利用可能なリージョンのリストについては、「AWSリージョン」を参照してください。

Amazon EC2 のリージョンエンドポイント

コマンドラインインターフェイスまたは API アクションを使用してインスタンスを操作するときはそのリージョンエンドポイントを指定する必要があります。Amazon EC2 のリージョンとエンドポイントの詳細については「Amazon EC2 デベロッパーガイド」の「Amazon EC2 service endpoints」を参照してください。

詳細については、「AWS リージョン とアベイラビリティーゾーンのユーザーガイド」の「AWS リージョン」を参照してください。

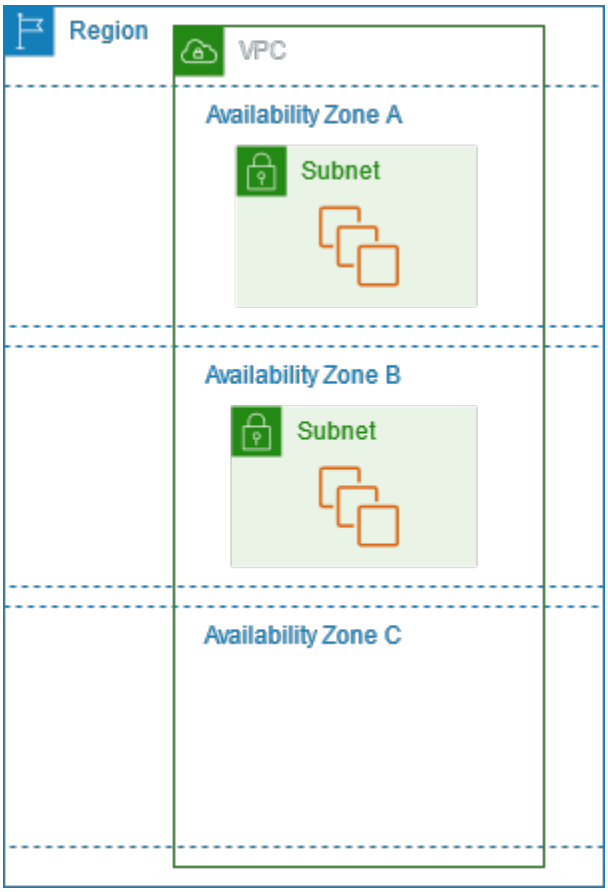
アベイラビリティーゾーン

リージョンごとにアベイラビリティーゾーンと呼ばれる複数の独立した場所があります。アベイラビリティーゾーンのコードはリージョンコードとそれに続く文字識別子です。例えば、`us-east-1a`。

複数のアベイラビリティーゾーンで EC2 インスタンスを起動することにより、リージョン内の 1 つの場所で障害が発生しても、アプリケーションを保護することができます。

次の図はAWS リージョン内の複数のアベイラビリティーゾーンを示しています。アベイラビリティーゾーン A とアベイラビリティーゾーン B にはそれぞれ 1 つのサブネットがあり、各サブネッ

トには EC2 インスタンスがあります。アベイラビリティーゾーン C にはサブネットがないため、このアベイラビリティーゾーンにインスタンスを起動することはできません。



詳細については「EC2 インスタンスの仮想プライベートクラウド」を参照してください。

リージョン別のアベイラビリティゾーン

リージョン別のアベイラビリティーゾーンのリストについては、「AWS アベイラビリティーゾーン」を参照してください。

アベイラビリティーゾーンのインスタンス

インスタンスを起動するときは、リージョンと Virtual Private Cloud (VPC) を選択します。次に、いずれかのアベイラビリティーゾーンからサブネットを選択するか、サブネットを自動的に選択されるようにできます。最初のインスタンスを起動するときは、システムの正常性と利用可能なキャパシティーに基づいて、アベイラビリティーゾーンを自動選択させることをお勧めします。追加のインスタンスを起動する場合、新しいインスタンスを既存のインスタンスに近づける必要がある場合、または既存のインスタンスから分離する必要がある場合にのみ、アベイラビリティーゾーンを指定します。

インスタンスを複数のアベイラビリティーゾーンに分散し、インスタンスに障害が発生した場合は、代わりに別のアベイラビリティーゾーンのインスタンスがリクエストを処理するようにアプリケーションを設計できます。

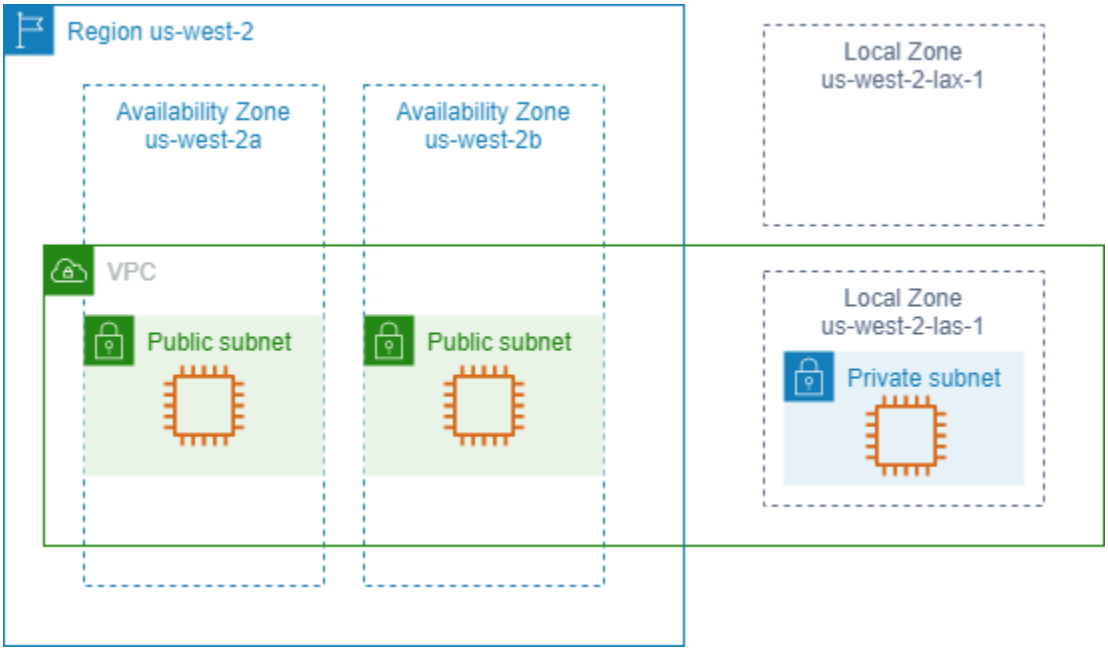
詳細については、「AWS リージョン とアベイラビリティーゾーンのユーザーガイド」の「AWS アベイラビリティーゾーン」を参照してください。

Local Zones

ローカルゾーンは地理的にユーザーに近い場所に位置する AWS リージョンを拡張したものです。Local Zones はインターネットへの独自の接続を持ち、Direct Connect をサポートしているため、Local Zonesで作成されたリソースは低レイテンシーの通信でローカルユーザーにサービスを提供できます。詳細についてはAWS ローカルゾーンユーザーガイドの「AWS ローカルゾーンとは」を参照してください。

ローカルゾーンのコードはそのリージョンコードの後に、物理的な場所を示す識別子が続きます。例えば、ロサンゼルスの `us-west-2-lax-1` です。

次の図はAWS リージョン `us-west-2`、そのアベイラビリティーゾーンのうちの 2 つ、およびそのローカルゾーンのうちの 2 つを示しています。VPC はアベイラビリティーゾーンといずれかのローカルゾーンにまたがっています。VPC 内の各ゾーンには 1 つのサブネットがあり、各サブネットにはインスタンスがあります。



利用可能な Local Zones

利用可能なローカルゾーンのリストについては「AWS ローカルゾーンユーザーガイド」の「Available Local Zones」を参照してください。発表されたローカルゾーンのリストについては「AWS Local Zones ロケーション」を参照してください。

ローカルゾーンのインスタンス

ローカルゾーンを使用するには最初にそれを有効にする必要があります。次に、ローカルゾーン内にサブネットを作成します。インスタンスの起動時にローカルゾーンのサブネットを指定できます。これにより、インスタンスはローカルゾーンのサブネットに配置されます。

ローカルゾーンでインスタンスを起動する際にはネットワーク境界グループからの IP アドレスの割り当ても行います。ネットワークボーダーグループはAWS が IP アドレスをアドバタイズするアベイラビリティーゾーン、Local Zones、または Wavelength Zones の一意のセットです (例: `us-west-2-lax-1a`)。ネットワークボーダーグループから次の IP アドレスを割り当てることができます。

- Amazon が提供する Elastic IPv4 アドレス
- Amazon が提供する IPv6 VPC アドレス (ロサンゼルスゾーンのみで利用可能)

Local Zones でインスタンスを起動する方法の詳細については「AWS Local Zones ユーザーガイド」の「AWS Local Zones 入門」を参照してください。

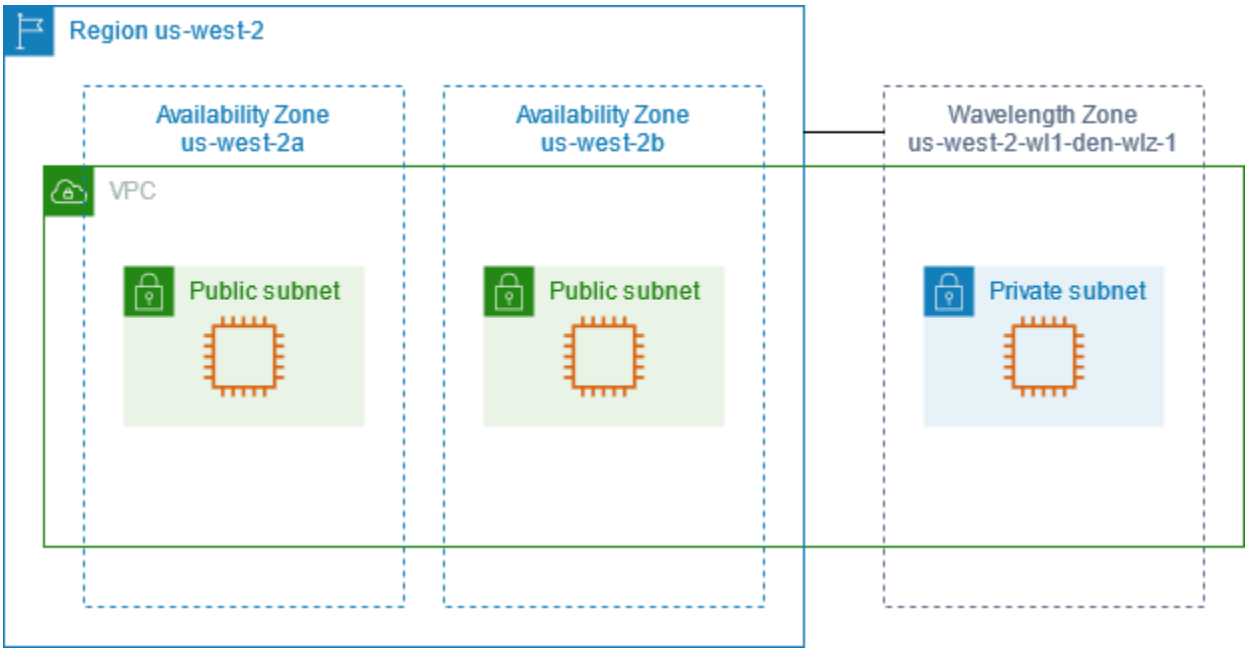
Wavelength Zone

AWS Wavelength を使用することで、デベロッパーはモバイルデバイスおよびエンドユーザー向けに、非常にレイテンシーが低いアプリケーションを構築できます。Wavelength は標準の AWS コンピューティングおよびストレージサービスを通信事業者の 5G ネットワークのエッジにデプロイします。デベロッパーはVirtual Private Cloud (VPC) を 1 つ以上の Wavelength Zones に拡張し、Amazon EC2 インスタンスなどの AWS リソースを使用して、超低レイテンシーやリージョンの AWS サービスへの接続を必要とするアプリケーションを実行できます。

Wavelength Zone はWavelength インフラストラクチャをデプロイする先のキャリアロケーション内の独立したゾーンです。Wavelength Zone はリージョンに関連付けられています。Wavelength Zone はリージョンの論理的な拡張であり、リージョンの制御プレーンによって管理されます。

Wavelength Zone のコードはそのリージョンコードの後に、物理的な場所を示す識別子が続きます。例えば、ボストンの `us-east-1-wl1-bos-wlz-1` です。

次の図はAWS リージョン us-west-2、そのアベイラビリティーゾーンのうちの 2 つ、および Wavelength Zone を示しています。VPC はアベイラビリティーゾーンと Wavelength Zone にまたがっています。VPC 内の各ゾーンには 1 つのサブネットがあり、各サブネットにはインスタンスがあります。



Wavelength Zoneはすべてのリージョンで利用できるわけではありません。Wavelength Zone をサポートするリージョンについてはAWS Wavelength デベロッパーガイドの利用可能な Wavelength Zone をご参照ください。

利用可能な Wavelength Zone

利用可能な Wavelength Zone のリストについては「AWS Wavelength ガイド」の「Available Wavelength Zones」を参照してください。

Wavelength Zone のインスタンス

Wavelength Zone を使用するにはまずゾーンにオプトインする必要があります。次に、Wavelength Zone にサブネットを作成します。インスタンスの起動時に Wavelength のサブネットを指定できます。また、ネットワークボーダーグループからキャリア IP アドレスを割り当てます。これはAWS が IP アドレスをアドバタイズするアベイラビリティーゾーン、Local Zones、または Wavelength Zones の一意のセットです (例: `us-east-1-wl1-bos-wlz-1` など)。

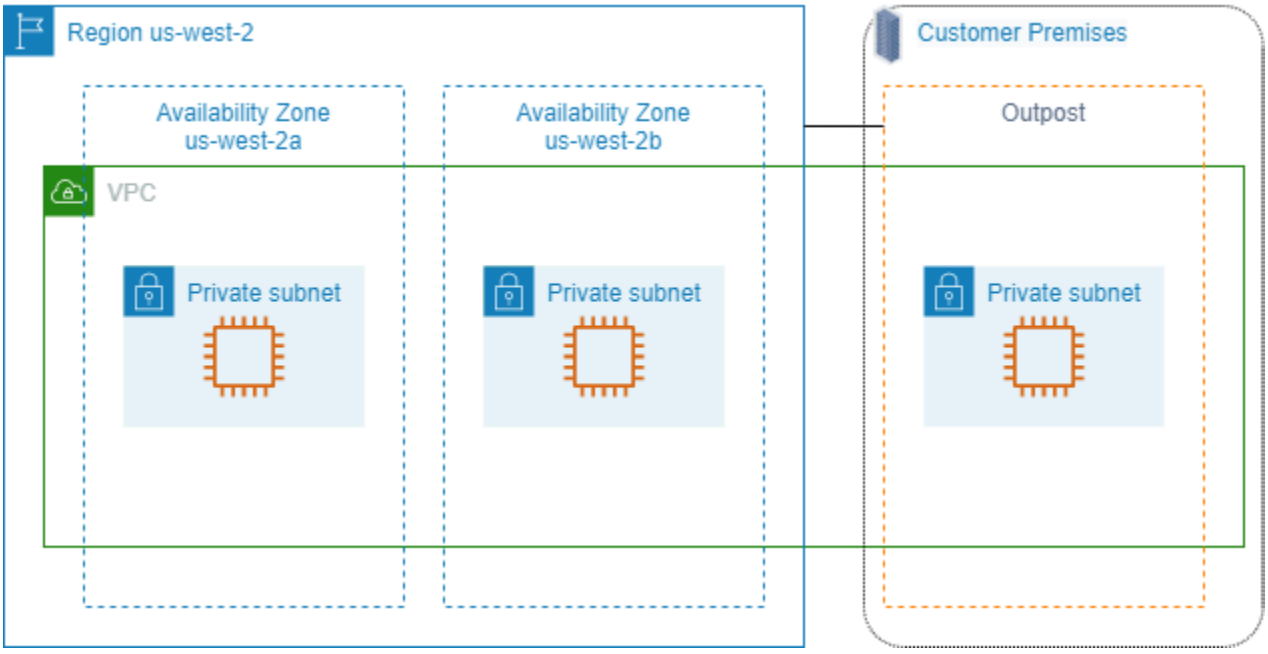
Wavelength Zone でインスタンスを起動する際のステップバイステップの指示については「AWS Wavelength デベロッパーガイド」の「Get started with AWS Wavelength」を参照してください。

AWS Outposts

AWS Outposts はAWS のインフラストラクチャ、サービス、API、ツールをお客様のオンプレミスまで拡張するフルマネージドサービスです。AWS は、AWS Outposts 管理されたインフラストラクチャへのローカルアクセスを提供することで、AWS リージョンと同じプログラミングインターフェイスを使用してオンプレミスでアプリケーションを構築して実行できるようにします。同時に、コンピューティングとストレージのローカルリソースを使用して、レイテンシーを短縮し、ローカルのデータ処理ニーズに対応します。

Outpost とはお客様のサイトにデプロイされる AWS のコンピューティングおよびストレージキャパシティーのプールです。AWS はAWS リージョンの一部としてこのキャパシティーを運営、監視、管理します。Outpost にサブネットを作成し、AWS リソースを作成したときにこれらのサブネットを指定します。Outpost サブネット内のインスタンスはプライベート IP アドレスを使用して、AWS リージョン内の他のインスタンスと通信します。これらはすべて同じ VPC 内にあります。

次の図はAWS リージョン `us-west-2`、そのアベイラビリティーゾーンのうちの 2 つ、および Outpost を示しています。VPC はアベイラビリティーゾーンと Outpost にまたがっています。Outpost はオンプレミスの顧客データセンターにあります。VPC 内の各ゾーンには 1 つのサブネットがあり、各サブネットにはインスタンスがあります。



Outpost のインスタンス

AWS Outposts の使用を開始するにはOutpost を作成し、Outpost 容量を注文する必要があります。AWS Outposts にはOutposts ラックと Outposts サーバーの 2 つのフォームファクタがありま

す。Outposts の設定の詳細については「AWS Outposts ファミリー」を参照してください。Outpost 機器をインストールすると、Outpost で EC2 インスタンスを起動するときに、コンピューティング容量とストレージ容量を使用できます。

EC2 インスタンスを起動するには、Outpost サブネットを作成する必要があります。セキュリティグループはアベイラビリティーゾーンサブネットのインスタンスと同様に、Outpost サブネットのインスタンスのインバウンドトラフィックとアウトバウンドトラフィックを制御します。SSH を使用して Outpost サブネットのインスタンスに接続するには、アベイラビリティーゾーンのサブネット内のインスタンスの場合と同様に、インスタンスの起動時にキー ペアを指定します。

詳細については「Get started with Outposts racks」または「Get started with Outposts servers」を参照してください。

Outposts ラックのボリューム

Outposts コンピューティング容量が Outpost ラックにある場合、作成した Outpost サブネットに EBS ボリュームを作成できます。ボリュームの作成時に、Outpost の Amazon リソースネーム (ARN) を指定します。

次の create-volume コマンドは指定した Outpost に空の 50 GB ボリュームを作成します。

```
aws ec2 create-volume --availability-zone us-east-2a --outpost-arn arn:aws:outposts:us-east-2:123456789012:outpost/op-03e6fecad652a6138 --size 50
```

Amazon EBS gp2 ボリュームのサイズはボリュームをデタッチする必要なく動的に変更することができます。ボリュームをデタッチせずに変更する方法の詳細については「Amazon EBS ユーザーガイド」の「Request modifications to your EBS volumes」を参照してください。

Outpost ラック上のインスタンスのルートボリュームを 30 GiB 以下に制限することをお勧めします。AMI またはインスタンスのブロックデバイスマッピングでデータボリュームを指定し、追加のストレージを提供できます。ブートボリュームから未使用のブロックを削除するにはAWS パートナーネットワークブログの「How to Build Sparse EBS Volumes」を参照してください。

ルートボリュームの NVMe タイムアウトを増やすことをお勧めします。詳細については「Amazon EBS ユーザーガイド」の「I/O オペレーションタイムアウト」を参照してください。

Outposts サーバー上のボリューム

Outposts サーバー上のインスタンスはインスタンスストアボリュームを提供しますが、EBS ボリュームはサポートしません。単一の EBS スナップショットのみを持つ Amazon EBS-backed AMI

を選択してください。アプリケーションの要件を満たすのに十分なインスタンスストレージを備えたインスタンスサイズを選択してください。詳細については「インスタンスストアボリュームの制限」を参照してください。

Amazon EC2 インスタンスの IP アドレス指定

Amazon EC2 と Amazon VPC はIPv4 と IPv6 の両方のアドレス設定プロトコルをサポートします。デフォルトではAmazon VPC は IPv4 アドレス設定プロトコルを使用します。この動作を無効にすることはできません。VPC の作成時には IPv4 CIDR ブロック (プライベート IPv4 アドレスの範囲) を指定する必要があります。必要に応じて、IPv6 CIDR ブロックを VPC に割り当て、そのブロックからサブネットのインスタンスに IPv6 アドレスを割り当てることができます。

EC2 インスタンスを起動するときに、VPC とサブネットを指定できます。インスタンスは、サブネットの CIDR 範囲からプライベート IPv4 アドレスを受け取ります。必要に応じて、インスタンスにパブリック IPv4 アドレスと IPv6 アドレスを設定できます。異なる VPC の EC2 インスタンスがパブリック IP アドレスを使用して通信する場合、トラフィックは AWS プライベートグローバルネットワークにとどまり、パブリックインターネットを通過しません。

内容

- プライベート IPv4 アドレス
- パブリック IPv4 アドレス
- パブリック IPv4 アドレスの最適化
- IPv6 アドレス
- 複数の IP アドレス
- EC2 インスタンスのホスト名
- リンクローカルアドレス
- EC2 インスタンスの IPv4 アドレスを管理する
- EC2 インスタンスの IPv6 アドレスを管理する
- EC2 インスタンスのセカンダリ IP アドレス
- Windows インスタンスのセカンダリプライベート IPv4 アドレスを設定する

プライベート IPv4 アドレス

プライベート IPv4 アドレスはインターネットから到達できない IP アドレスです。プライベート IPv4 アドレスは同じ VPC 内のインスタンス間の通信に使用できます。プライベート IPv4 アドレス

の標準および仕様についてはRFC 1918 を参照してください。DHCP を使用してインスタンスにプライベート IPv4 アドレスが割り当てられます。

> **Note**
>
> RFC 1918 に指定されているプライベート IPv4 アドレスの範囲に含まれない、パブリックにルーティングできる CIDR ブロックを持つ VPC を作成できます。ただし、このドキュメントでプライベート IPv4 アドレス (またはプライベート IP アドレス) と言う場合はVPC の IPv4 CIDR 範囲に含まれる IP アドレスを指します。

VPC サブネットは次のいずれかのタイプです。

- IPv4 専用サブネット - これらのサブネットにはIPv4 アドレスが割り当てられたリソースのみを作成できます。
- IPv6 専用サブネット - これらのサブネットにはIPv6 アドレスが割り当てられたリソースのみを作成できます。
- IPv4 および IPv6 サブネット - これらのサブネットにはIPv4 または IPv6 アドレスのいずれかが割り当てられたリソースを作成できます。

EC2 インスタンスをIPv4 専用サブネットまたはデュアルスタック (IPv4 および IPv6) サブネットで起動すると、インスタンスはサブネットの IPv4 アドレス範囲からプライマリプライベート IP アドレスを受け取ります。詳細については[Amazon VPC User Guide] (Amazon VPC ユーザーガイド)の[IP addressing] (IP アドレス指定) を参照してください。プライマリプライベート IP アドレスを指定しないでインスタンスを起動すると、サブネットの IPv4 範囲内で使用可能な IP アドレスが自動的に選択されます。各インスタンスにはプライマリプライベート IPv4 アドレスが割り当てられたデフォルトのネットワークインターフェイス (eth0) があります。追加のプライベート IPv4 アドレス (セカンダリプライベート IPv4 アドレス) も指定できます。プライマリプライベート IP アドレスとは異なり、セカンダリプライベート IP アドレスは別のインスタンスに割り当て直すことができます。詳細については、「複数の IP アドレス」を参照してください。

プライベート IPv4 アドレスはプライマリアドレスまたはセカンダリアドレスを問わず、インスタンスが停止して起動、または休止して起動した際に、ネットワークインターフェイスに関連付けられたままになり、インスタンスを終了するとリリースされます。

パブリック IPv4 アドレス

パブリック IP アドレスはインターネットから到達可能な IPv4 アドレスです。インスタンスとインターネット間で通信するにはパブリック アドレスを使用できます。

デフォルトの VPC でインスタンスを起動すると、デフォルトでパブリック IP アドレスが割り当てられます。デフォルト以外の VPC でインスタンスを起動するとき、サブネットにはそのサブネットで起動するインスタンスがパブリック IPv4 アドレスプールからパブリック IP アドレスを受け取るかどうかを決定する属性があります。デフォルトではデフォルト以外のサブネットで起動されたインスタンスにパブリック IP アドレスを割り当てません。

インスタンスがパブリック IP アドレスを割り当てられるかどうかを制御するには以下の方法を使用します。

- サブネットのパブリック IP アドレス指定属性を変更します。詳細については「Amazon VPC ユーザーガイド」の「サブネットのパブリック IPv4 アドレス指定属性の変更」を参照してください。
- 起動時にパブリック IP アドレス指定機能を有効または無効にします。これにより、サブネットのパブリック IP アドレス指定属性が上書きされます。詳細については、「起動時にパブリック IPv4 アドレスを割り当てる」を参照してください。
- 起動後にインスタンスからパブリック IP アドレスの割り当てを解除します。詳細については、「the section called “IP アドレスの管理”」を参照してください。

パブリック IP アドレスはAmazon のパブリック IPv4 アドレスプールからインスタンスに割り当てられ、お客様の AWS アカウントには関連付けられません。パブリック IP アドレスをインスタンスから割り当て解除すると、そのパブリック IPv4 アドレスはパブリック IP アドレスプールに戻され、再利用することはできません。

次の場合、パブリック IP アドレスをインスタンスか解放し、新しい IP アドレスを割り当てます。

- インスタンスが停止、休止状態、または終了すると、インスタンスのパブリック IP アドレスが開放されます。停止または休止状態のインスタンスは起動時に、新しいパブリック IP アドレスが割り当てられます。
- Elastic IP アドレスをインスタンスに関連付けると、パブリック IP アドレスが解放されます。インスタンスから Elastic IP アドレスの関連付けを解除すると、そのインスタンスには新しいパブリック IP アドレスが割り当てられます。
- インスタンスのパブリック IP アドレスを開放し、セカンダリネットワークインターフェイスがある場合、新しいパブリック IP アドレスは割り当てられません。

- インスタンスのパブリック IP アドレスが開放され、且つそのインスタンスに Elastic IP アドレスに関連付けられたセカンダリプライベート IP アドレスがある場合、新しいパブリック IP アドレスは割り当てられません。

必要に応じて、インスタンスに関連付けおよびインスタンスから関連付けできる永続的なパブリック IP アドレスが必要な場合はElastic IP アドレスを使用します。

動的 DNS を使用して既存の DNS 名を新しいインスタンスのパブリック IP アドレスにマッピングした場合、その IP アドレスがインターネット内に伝達されるまでに最大 24 時間かかることがあります。その結果、新しいインスタンスはトラフィックを受信せず、終了したインスタンスがリクエストの受信を継続することがあります。この問題を解決するにはElastic IP アドレスを使用します。独自の Elastic IP アドレスを割り当てて、それをインスタンスに関連付けることができます。詳細については「Elastic IP アドレス」を参照してください。

Amazon VPC IP Address Manager (IPAM) を使用している場合はAWS からパブリック IPv4 アドレスの連続したブロックを取得し、それを使用して Elastic IP アドレスを AWS リソースに割り当てることができます。連続した IPv4 アドレスブロックを使用すると、セキュリティアクセスコントロールリストの管理オーバーヘッドを大幅に削減し、AWS でスケーリングするエンタープライズの IP アドレスの割り当てと追跡を簡素化できます。詳細については「Amazon VPC IPAM ユーザーガイド」の「Allocate sequential Elastic IP addresses from an IPAM pool」を参照してください。

考慮事項

- AWS は、実行中のインスタンスに関連付けられているパブリック IPv4 アドレスや Elastic IP アドレスを含めたすべてのパブリック IPv4 アドレスの料金を請求します。詳細については「Amazon VPC の料金」ページの「パブリック IPv4 アドレス」タブを参照してください。
- インスタンスがパブリック NAT IP アドレスを使用して他のインスタンスにアクセスする場合、アクセス先のインスタンスが同じリージョンにあるかどうかによって、リージョンデータ転送またはインターネットデータ転送に対して課金されます。

パブリック IPv4 アドレスの最適化

AWS では実行中のインスタンスに関連付けられているパブリック IPv4 アドレスと Elastic IP アドレスを含む、すべてのパブリック IPv4 アドレスに対して料金が課されます。詳細については「Amazon VPC の料金」ページの「パブリック IPv4 アドレス」タブを参照してください。

次のリストには使用するパブリック IPv4 アドレスの数を最適化するために実行できるアクションが含まれています。

- Elastic Load Balancer を使用して EC2 インスタンスへのトラフィックを分散させ、インスタンスに割り当てられたプライマリ ENI でパブリック IP の自動割り当てを無効にします。ロードバランサーは 1 つのパブリック IPv4 アドレスを使用するため、パブリック IPv4 アドレスの数を削減できます。パブリック IPv4 アドレスの数をさらに削減すために、既存のロードバランサーを統合することもできます。
- NAT ゲートウェイを使用する唯一の理由が、メンテナンスまたは緊急のためにプライベートサブネットの EC2 インスタンスに SSH 接続することである場合は代わりに EC2 Instance Connect Endpoint を使用することを検討してください。EC2 Instance Connect Endpoint を使用すると、パブリック IPv4 アドレスを使わないでインスタンスに接続できます。
- EC2 インスタンスがパブリックサブネットにあり、パブリック IP アドレスが割り当てられている場合はインスタンスをプライベートサブネットに移動し、パブリック IP アドレスを削除して、EC2 インスタンスとの間のアクセスを許可するためにパブリック NAT ゲートウェイを使用することを検討してください。NAT ゲートウェイを使用する場合は費用に注意する必要があります。この計算方法を使用して、NAT ゲートウェイを使用する場合の費用対効果を判断します。この計算に必要な `Number of public IPv4 addresses` はAWS請求コストと使用状況レポートを作成することで取得できます。

  ```
  NAT gateway per hour + NAT gateway public IPs + NAT gateway transfer / Existing
   public IP cost
  ```

  コードの説明は以下のとおりです。

  - `NAT gateway per hour = $0.045 * 730 hours in a month * Number of Availability Zones the NAT gateways are in`
  - `NAT gateway public IPs = $0.005 * 730 hours in a month * Number of IPs associated with your NAT gateways`
  - `NAT gateway transfer = $0.045 * Number of GBs that will go through the NAT gateway in a month`
  - `Existing public IP cost = $0.005 * 730 hours in a month * Number of public IPv4 addresses`

  合計が 1 未満の場合、NAT ゲートウェイの方がパブリック IPv4 アドレスよりも安価に済みます。
- AWS PrivateLink を使用して、パブリック IPv4 アドレスやインターネットゲートウェイを使用するのではなく、AWS サービスまたは他の AWS アカウントによってホストされているサービスにプライベートに接続します。

- Amazon が所有するパブリック IPv4 アドレスを使用するのではなく、独自の IP アドレス範囲 (BYOIP) を AWS 環境に持ち込み、その範囲をパブリック IPv4 アドレスに使用します。
- サブネット内に起動されたインスタンスに対するパブリック IPv4 アドレスの自動割り当てを無効ににします。このオプションは通常、サブネットの作成時に VPC に対してデフォルトで無効になっていますが、既存のサブネットをチェックして無効になっていることを確認する必要があります。
- パブリック IPv4 アドレスを必要としない EC2 インスタンスがある場合はインスタンスにアタッチされたネットワークインターフェイスでパブリック IP の自動割り当てが無効になっていることを確認してください。
- プライベートサブネットの EC2 インスタンス用に AWS Global Accelerator でアクセラレータエンドポイントを設定して、パブリック IP アドレスを使用せずにインターネットトラフィックが VPCs内のエンドポイントに直接流れるようにします。また、独自のアドレスを AWS Global Accelerator に持ち込んで、アクセラレーターの静的 IP アドレスに独自の IPv4 アドレスを使用することもできます。

IPv6 アドレス

IPv6 アドレスはグローバルに一意であり、プライベートのまま、またはインターネット経由で到達可能になるように設定できます。AWS ではパブリック IPv6 とプライベート IPv6 のどちらのアドレス指定も利用可能です。

- プライベート IPv6: AWS ではAWS からインターネット上でアドバタイズされておらず、アドバタイズできない IPv6 アドレスをプライベート IPv6 アドレスと見なします。
- パブリック IPv6: AWS ではAWS からインターネットでアドバタイズされている IPv6 アドレスをパブリック IPv6 アドレスと見なします。

パブリックおよびプライベート IPv6 アドレスの詳細については「Amazon VPC ユーザーガイド」の「IPv6 アドレス」を参照してください。

C1, M1, M2, M3 を除くすべてのインスタンスタイプが IPv6 アドレスをサポートします。 T1

IPv6 CIDR ブロックが VPC およびサブネットと関連付けられており、以下のいずれかに該当する場合、EC2 インスタンスには IPv6 アドレスが割り当てられます。

- 起動時にサブネットからインスタンスに IPv6 アドレスが自動的に割り当てられるように設定されている。詳細については「サブネットの IP アドレス指定属性を変更する」を参照してください。

- 起動時に IPv6 アドレスをインスタンスに割り当てる。
- 起動後に IPv6 アドレスをインスタンスのプライマリネットワークインターフェイスに割り当てる。
- 起動後に IPv6 アドレスを同じサブネットのネットワークインターフェイスに割り当て、そのネットワークインターフェイスをインスタンスにアタッチします。

起動時にインスタンスに IPv6 アドレスが割り当てられると、そのアドレスはインスタンスのプライマリネットワークインターフェイス (eth0) と関連付けられます。インスタンスのプライマリネットワークインターフェイス (eth0) の IPv6 アドレスは次の方法で管理できます。

- ネットワークインターフェイスへ IPv6 アドレスを割り当て/割り当て解除します。ネットワークインターフェイスに割り当てることができる IPv6 アドレスの数と、インスタンスにアタッチできるネットワークインターフェイスの数はインスタンスタイプごとに異なります。詳細については「ネットワークインターフェイスあたりの IP アドレスの最大数」を参照してください。
- プライマリ IPv6 アドレスを有効にします。プライマリ IPv6 アドレスにより、インスタンスまたは ENI へのトラフィックの中断を回避できます。詳細については「EC2 インスタンス用のネットワークインターフェイスを作成する」または「ネットワークインターフェイスの IP アドレスを管理する」を参照してください。

IPv6 アドレスはインスタンスの停止して起動、または休止して起動する際には保持され、インスタンスの終了時にリリースされます。IPv6 アドレスは別のネットワークインターフェイスに割り当てられている間は再割り当てできません。最初に割り当てを解除する必要があります。

インスタンスのサブネットのルーティングを制御するか、セキュリティグループとネットワーク ACL ルールを使用することで、IPv6 アドレスを介してインスタンスに接続できるかどうかを制御できます。詳細については「Amazon VPC User Guide」のインターネットワークトラフィックのプライバシーを参照してください。

予約済み IPv6 アドレスの範囲については「IANA IPv6 Special-Purpose Address Registry」と「RFC4291」を参照してください。

複数の IP アドレス

インスタンスに複数のプライベート IPv4 および IPv6 アドレスを指定できます。インスタンスに指定できるネットワークインターフェイスとプライベート IPv4 および IPv6 アドレスの数はインスタンスタイプによって異なります。詳細については、「ネットワークインターフェイスあたりの IP アドレスの最大数」を参照してください。

ユースケース

- 1 つのサーバーで複数の SSL 証明書を使用し、各インターフェイスに各 IP アドレスに割り当てることで、1 つのサーバーで複数のウェブサイトをホストします。
- 各ネットワークインターフェイス用に複数の IP アドレスを持つネットワークアプライアンス (ファイアウォールやロードバランサーなど) を運用します。
- インスタンスでエラーが発生した場合に、セカンダリ IP アドレスをスタンバイインスタンスに再割り当てすることによって、内部トラフィックをスタンバイインスタンスにリダイレクトします。

複数の IP アドレスを使用する方法

- セカンダリプライベート IPv4 アドレスをネットワークインターフェイスに割り当てることができます。
- IPv6 CIDR ブロックが関連付けられているサブネット内のネットワークインターフェイスに複数の IPv6 アドレスを割り当てることができます。
- ネットワークインターフェイスのサブネットの IPv4 CIDR ブロック範囲からセカンダリ IPv4 アドレスを選択する必要があります。
- ネットワークインターフェイスのサブネットの IPv6 CIDR ブロック範囲から IPv6 アドレスを選択する必要があります。
- セキュリティグループを関連付けるのは個々の IP アドレスではなく、ネットワークインターフェイスです。そのため、ネットワークインターフェイスで指定した各 IP アドレスはそのネットワークインターフェイスのセキュリティグループの対象です。
- 複数の IP アドレスは実行中または停止したインスタンスにアタッチされたネットワークインターフェイスに割り当てたり、割り当て解除したりできます。
- ネットワークインターフェイスに割り当てられているセカンダリプライベート IPv4 アドレスは明示的に許可された場合、別のネットワークインターフェイスに割り当て直すことができます。
- IPv6 アドレスは最初に既存のネットワークインターフェイスから割り当て解除しない限り、別のネットワークインターフェイスに再割り当てすることはできません。
- コマンドラインツールまたは API を使用して複数の IP アドレスをネットワークインターフェイスに割り当てるときに、いずれかの IP アドレスを割り当てることができない場合、オペレーション全体が失敗します。
- プライマリプライベート IPv4 アドレス、セカンダリプライベート IPv4 アドレス、Elastic IP アドレス、および IPv6 アドレスはセカンダリネットワークインターフェイスをインスタンスからデタッチしたり、インスタンスにアタッチしても、セカンダリネットワークインターフェイスへの割り当ては維持します。

- プライマリネットワークインターフェイスをインスタンスからデタッチすることはできませんが、プライマリネットワークインターフェイスのセカンダリプライベート IPv4 アドレスを別のネットワークインターフェイスに再割り当てすることはできます。

詳細については、「the section called “セカンダリ IP アドレス”」を参照してください。

EC2 インスタンスのホスト名

EC2 インスタンスを作成すると、AWS はそのインスタンスのホスト名を作成します。ホスト名のタイプと、AWS によるプロビジョニング方法についての詳細EC2 インスタンスのホスト名とドメインを参照してください。Amazon は、Amazon が提供するホスト名を IPv4 および IPv6 アドレスに解決する DNS サーバーを提供します。Amazon DNS サーバーは VPC ネットワークの範囲に 2 をプラスしたアドレスにあります。詳細については、Amazon VPC ユーザーガイドの「DNS attributes for your VPC」(VPC の DNS 属性) を参照してください。

リンクローカルアドレス

リンクローカルアドレスはよく知られた、ルーティング不可の IP アドレスです。Amazon EC2 はリンクローカルアドレス空間のアドレスを使用して、EC2 インスタンスからのみアクセスできるサービスを提供します。これらのサービスはインスタンス上では実行されず、基盤となるホスト上で実行されます。これらのサービスのリンクローカルアドレスにアクセスすると、Xen ハイパーバイザーまたは Nitro コントローラーのどちらかと通信することになります。

リンクローカルアドレスの範囲

- IPv4 — 169.254.0.0/16 (169.254.0.0 ~ 169.254.255.255)
- IPv6 – fe80::/10

リンクローカルアドレスを使用してアクセスするサービス

- インスタンスメタデータサービス
- Amazon Route 53 Resolver (Amazon DNS サーバーとも呼ばれます)
- Amazon Time Sync Service
- AWS KMS サーバー

EC2 インスタンスの IPv4 アドレスを管理する

インスタンスを起動するときに、パブリック IPv4 アドレスをインスタンスに割り当てることができます。インスタンスの IPv4 アドレスをコンソールに表示するにはInstances (インスタンス) ページまたは [Network Interfaces] (ネットワークインターフェイス) ページを使用します。

タスク

- 起動時にパブリック IPv4 アドレスを割り当てる
- 起動時にプライベート IPv4 アドレスを割り当てる
- プライマリ IPv4 アドレスを表示する
- インスタンスメタデータを使用して IPv4 アドレスを表示するには

起動時にパブリック IPv4 アドレスを割り当てる

各サブネットに、そのサブネット内で起動されるインスタンスにパブリック IP アドレスが割り当てられるかどうかを決定する属性があります。デフォルトではデフォルト以外のサブネットではこの属性が false に設定されており、デフォルトのサブネットではこの属性が true に設定されています。インスタンスを起動する場合、パブリック IPv4 アドレス指定機能を使用してインスタンスにパブリック IPv4 アドレスを割り当てるかどうかを制御することもできます。サブネットの IP アドレス指定属性のデフォルトの動作をオーバーライドできます。パブリック IPv4 アドレスはAmazon のパブリック IPv4 アドレスプールから割り当てられ、デバイスインデックス eth0 を持つネットワークインターフェイスに割り当てられます。この機能はインスタンス起動時の特定の条件により異なります。

考慮事項

- ネットワークインターフェースに関連付けられた IP アドレスを管理することで、起動後にインスタンスからパブリック IP アドレスの割り当てを解除できます。パブリック IPv4 アドレスの詳細についてはパブリック IPv4 アドレスを参照してください。
- 複数のネットワークインターフェイスを指定した場合、パブリック IP アドレスを自動で割り当てることはできません。さらに、eth0 のように既存のネットワークインターフェイスを指定すると、パブリック IP の自動割り当て機能を使用してサブネット設定をオーバーライドすることはできません。
- 起動時にパブリック IP アドレスをインスタンスに割り当てるかどうかにかかわらず、起動後に Elastic IP アドレスをインスタンスに関連付けることができます。詳細については「Elastic IP アドレス」を参照してください。サブネットのパブリック IPv4 アドレス指定動作を変更することもできます。詳細については「サブネットの IPv4 アドレス指定属性の変更」を参照してください。

Console

起動時にパブリック IPv4 アドレスを割り当てるには

手順に従ってインスタンスを起動し、[Network Settings] (ネットワーク設定) を設定するときに、[Auto-assign Public IP] (パブリック IP を自動的に割り当てる) オプションを選択してください。

AWS CLI

起動時にパブリック IPv4 アドレスを割り当てるには

`--associate-public-ip-address` オプションで run-instances コマンドを使用します。

```
--associate-public-ip-address
```

PowerShell

起動時にパブリック IPv4 アドレスを割り当てるには

`-AssociatePublicIp` パラメータで New-EC2Instance コマンドレットを使用します。

```
-AssociatePublicIp $true
```

起動時にプライベート IPv4 アドレスを割り当てる

プライベート IPv4 アドレスは、サブネットの IPv4 アドレス範囲から指定できます。また、Amazon EC2 に自動選択させることもできます。これはプライマリネットワークインターフェイスの IP アドレスです。

起動後に IPv4 アドレスを割り当てる方法については、the section called “インスタンスにセカンダリ IP アドレスを割り当てる” を参照してください。

Console

起動時にプライベート IPv4 アドレスを割り当てるには

インスタンスを起動するための手順に従います。[ネットワーク設定] を設定するときは、[高度なネットワーク設定] を展開してプライマリ IP の値を入力します。

AWS CLI

起動時にプライベート IPv4 アドレスを割り当てるには

`--private-ip-address` オプションで run-instances コマンドを使用します。

```
--private-ip-addresses 10.251.50.12
```

Amazon EC2 に IP アドレスを選択させるときは、このオプションを省略します。

PowerShell

起動時にプライベート IPv4 アドレスを割り当てるには

`-PrivateIpAddress` パラメータで New-EC2Instance コマンドレットを使用します。

```
-PrivateIpAddress 10.251.50.12
```

Amazon EC2 に IP アドレスを選択させるときは、このパラメータを省略します。

プライマリ IPv4 アドレスを表示する

パブリック IPv4 アドレスはコンソールのネットワークインターフェイスのプロパティとして表示されますが、NAT によってプライマリプライベート IPv4 アドレスにマッピングされます。したがって、インスタンスのネットワークインターフェイスのプロパティを、例えば `ifconfig` (Linux) または `ipconfig` (Windows) を通して調べてみると、パブリック IPv4 アドレスは表示されていません。

Console

インスタンスの IPv4 アドレスを表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択します。
4. [ネットワーク] タブで、パブリック IPv4 アドレスとプライベート IPv4 アドレスを見つけます。
5. (オプション) [ネットワーク] タブには、インスタンスのネットワークインターフェイスと Elastic IP アドレスも記載されています。

AWS CLI

インスタンスのプライマリ IPv4 アドレスを表示するには

describe-instances コマンドを使用します。

```
aws ec2 describe-instances \
    --instance-ids i-1234567890abcdef0 \
    --query "Reservations[].Instances[].PrivateIpAddress" \
    --output text
```

以下は出力の例です。

```
10.251.50.12
```

PowerShell

インスタンスのプライマリ IPv4 アドレスを表示するには

Get-EC2Instance コマンドレットを使用します。

```
(Get-EC2Instance `
    -InstanceId i-1234567890abcdef0).Instances.PrivateIpAddress
```

以下は出力の例です。

```
10.251.50.12
```

インスタンスメタデータを使用して IPv4 アドレスを表示するには

インスタンスの IPv4 アドレスは、インスタンスのメタデータを取得することで入手できます。詳細については、「インスタンスメタデータを使用して EC2 インスタンスを管理する」を参照してください。

インスタンスメタデータを使用して IPv4 アドレスを表示するには

1. インスタンスに接続します。詳細については、「EC2 インスタンスに接続する」を参照してください。
2. 以下のいずれかのコマンドを実行します。

   IMDSv2

   Linux

次のコマンドを Linux インスタンスから実行します。

```
TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-
metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/
meta-data/local-ipv4
```

Server

次のコマンドを Windows インスタンスから実行します。

```
[string]$token = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token-ttl-
seconds" = "21600"} `
    -Method PUT -Uri http://169.254.169.254/latest/api/token
```

```
Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} `
    -Method GET -Uri http://169.254.169.254/latest/meta-data/local-ipv4
```

IMDSv1

Linux

次のコマンドを Linux インスタンスから実行します。

```
curl http://169.254.169.254/latest/meta-data/local-ipv4
```

Server

次のコマンドを Windows インスタンスから実行します。

```
Invoke-RestMethod http://169.254.169.254/latest/meta-data/local-ipv4
```

3. パブリック IP アドレスにアクセスするには次のコマンドのうちいずれか 1 つを使用します。Elastic IP アドレスがインスタンスに関連付けられている場合、コマンドは Elastic IP アドレスを返します。

IMDSv2

Linux

次のコマンドを Linux インスタンスから実行します。

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H
 "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/
meta-data/public-ipv4
```

Server

次のコマンドを Windows インスタンスから実行します。

```
[string]$token = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token-ttl-
seconds" = "21600"} `
    -Method PUT -Uri http://169.254.169.254/latest/api/token
```

```
Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} `
    -Method GET -Uri http://169.254.169.254/latest/meta-data/public-ipv4
```

IMDSv1

Linux

次のコマンドを Linux インスタンスから実行します。

```
curl http://169.254.169.254/latest/meta-data/public-ipv4
```

Server

次のコマンドを Windows インスタンスから実行します。

```
Invoke-RestMethod http://169.254.169.254/latest/meta-data/public-ipv4
```

EC2 インスタンスの IPv6 アドレスを管理する

VPC とサブネットに IPv6 CIDR ブロックが関連付けられている場合は起動時または起動後に IPv6 アドレスをインスタンスに割り当てることができます。インスタンスの IPv6 アドレスにコンソールでアクセスするには、[インスタンス] ページまたは [ネットワークインターフェイス] ページを使用します。以下のタスクでは、インスタンスの IP アドレスを設定します。代わりにネットワークイン

ターフェイスの IP アドレスを設定するには、「the section called “IP アドレスの管理”」を参照してください。

タスク

- インスタンスへの IPv6 アドレスの割り当て
- インスタンスの IPv6 アドレスを表示する
- インスタンスメタデータを使用して IPv6 アドレスを表示する
- インスタンスからの IPv6 アドレスの割り当て解除

インスタンスへの IPv6 アドレスの割り当て

IPv6 アドレスは、サブネットの IPv6 アドレス範囲から指定できます。また、Amazon EC2 に自動選択させることもできます。これはプライマリネットワークインターフェイスの IP アドレスです。C1、M1、M2、M3 および T1 のインスタンスタイプは IPv6 アドレスをサポートしていないことに注意してください。

Console

起動時に IPv6 アドレスを割り当てるには

インスタンスを起動するための手順に従います。ネットワーク設定を構成するときはIPv6 IP を自動割り当てするオプションを選択してください。このオプションが表示されない場合、選択したサブネットには IPv6 CIDR ブロックが関連付けられていません。

起動後に IPv6 アドレスをインスタンスに割り当てるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択後、[アクション]、[ネットワーク]、[IP アドレスの管理] の順に選択してください。
4. ネットワークインターフェイスを展開します。[IPv6 addresses] (IPv6 アドレス) で、[Assign new IP address] (新しい IP アドレスの割り当て) を選択してください。
5. サブネットの範囲から IPv6 アドレスを入力します。フィールドを空欄のままにすると、Amazon EC2 が自動的に IPv6 アドレスを選択します。このオプションが表示されない場合、インスタンスのサブネットには IPv6 CIDR ブロックが関連付けられていません。
6. [保存] を選択します。

AWS CLI

起動時に IPv6 アドレスを割り当てるには

`--ipv6-addresses` オプションで run-instances コマンドを使用します。次の例では、2 つの IPv6 アドレスを割り当てます。

```
--ipv6-addresses Ipv6Address=2001:db8::1234:5678:1.2.3.4
 Ipv6Address=2001:db8::1234:5678:5.6.7.8
```

Amazon EC2 に IPv6 アドレスを選択させる場合は、代わりに `--ipv6-address-count` のオプションを使用します。次の例では、2 つの IPv6 アドレスを割り当てます。

```
--ipv6-address-count 2
```

起動後に IPv6 アドレスをインスタンスに割り当てるには

assign-ipv6-addresses コマンドを使用します。次の例では、2 つの IPv6 アドレスを割り当てます。

```
aws ec2 assign-ipv6-addresses \
    --network-interface-id eni-1234567890abcdef0 \
    --ipv6-addresses 2001:db8::1234:5678:1.2.3.4 2001:db8::1234:5678:5.6.7.8
```

Amazon EC2 に IPv6 アドレスを選択させる場合は、代わりに `--ipv6-address-count` のオプションを使用します。次の例では、2 つの IPv6 アドレスを割り当てます。

```
aws ec2 assign-ipv6-addresses \
    --network-interface-id eni-1234567890abcdef0 \
    --ipv6-address-count 2
```

PowerShell

起動時に IPv6 アドレスを割り当てるには

`-Ipv6Address` パラメータで New-EC2Instance コマンドレットを使用します。次の例では、2 つの IPv6 アドレスを割り当てます。

```
-Ipv6Address $ipv6addr1,$ipv6addr2
```

IPv6 アドレスを次のように定義します。

```
$ipv6addr1 = New-Object Amazon.EC2.Model.InstanceIpv6Address
$ipv6addr1.Ipv6Address = "2001:db8::1234:5678:1.2.3.4"
$ipv6addr2 = New-Object Amazon.EC2.Model.InstanceIpv6Address
$ipv6addr2.Ipv6Address = "2001:db8::1234:5678:5.6.7.8"
```

Amazon EC2 に IPv6 アドレスを選択させる場合は、代わりに `-Ipv6AddressCount` のパラメータを使用します。次の例では、2 つの IPv6 アドレスを割り当てます。

```
-Ipv6AddressCount 2
```

起動後に IPv6 アドレスをインスタンスに割り当てるには

Register-EC2Ipv6AddressList コマンドレットを使用します。次の例では、2 つの IPv6 アドレスを割り当てます。

```
Register-EC2Ipv6AddressList `
    -NetworkInterfaceId eni-1234567890abcdef0 `
    -Ipv6Address "2001:db8::1234:5678:1.2.3.4","2001:db8::1234:5678:5.6.7.8"
```

Amazon EC2 に IPv6 アドレスを選択させる場合は、代わりに `-Ipv6AddressCount` のパラメータを使用します。次の例では、2 つの IPv6 アドレスを割り当てます。

```
Register-EC2Ipv6AddressList `
    -NetworkInterfaceId eni-1234567890abcdef0 `
    -Ipv6AddressCount 2
```

インスタンスの IPv6 アドレスを表示する

インスタンスの IPv6 アドレスを表示することができます。

Console

インスタンスの IPv6 アドレスを表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択してください。

4. [Networking] (ネットワーキング) タブで、[IPv6 addresses] (IPv6 アドレス) を見つけます。

AWS CLI

インスタンスの IPv6 アドレスを表示するには

describe-instances コマンドを使用します。

```
aws ec2 describe-instances \
    --instance-ids i-1234567890abcdef0 \
    --query "Reservations[*].Instances[].Ipv6Address" \
    --output text
```

以下は出力の例です。

```
2001:db8::1234:5678:1.2.3.4
```

PowerShell

インスタンスの IPv6 アドレスを表示するには

Get-EC2Instance コマンドレットを使用します。

```
(Get-EC2Instance `
    -InstanceId i-1234567890abcdef0).Instances.Ipv6Address
```

以下は出力の例です。

```
2001:db8::1234:5678:1.2.3.4
```

インスタンスメタデータを使用して IPv6 アドレスを表示する

インスタンスに接続すると、インスタンスメタデータを使用して IPv6 アドレスを取得できるようになります。最初に、`http://169.254.169.254/latest/meta-data/network/interfaces/macs/` からインスタンスの MAC アドレスを取得します。

IMDSv2

Linux

次のコマンドを Linux インスタンスから実行します。

```
TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/network/interfaces/macs/mac-address/ipv6s
```

Server

次のコマンドレットを Windows インスタンスから実行します。

```
[string]$token = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token-ttl-seconds" = "21600"} `
    -Method PUT -Uri http://169.254.169.254/latest/api/token
```

```
Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} `
    -Method GET -Uri http://169.254.169.254/latest/meta-data/network/interfaces/macs/mac-address/ipv6s
```

IMDSv1

Linux

次のコマンドを Linux インスタンスから実行します。

```
curl http://169.254.169.254/latest/meta-data/network/interfaces/macs/mac-address/ipv6s
```

Server

次のコマンドレットを Windows インスタンスから実行します。

```
Invoke-RestMethod -Uri http://169.254.169.254/latest/meta-data/network/interfaces/macs/mac-address/ipv6s
```

インスタンスからの IPv6 アドレスの割り当て解除

IPv6 アドレスはインスタンスからいつでも割り当て解除できます。

Console

インスタンスから IPv6 アドレスを割り当て解除するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択後、[アクション]、[ネットワーク]、[IP アドレスの管理] の順に選択してください。
4. ネットワークインターフェイスを展開します。[IPv6 addresses] (IPv6 アドレス) で、IPv6 アドレスの横にある [Unassign] (割り当て解除) を選択してください。
5. [保存] を選択します。

AWS CLI

インスタンスから IPv6 アドレスを割り当て解除するには

unassign-ipv6-addresses コマンドを使用します。

```
aws ec2 unassign-ipv6-addresses \
    --network-interface-id eni-1234567890abcdef0 \
    --ipv6-addresses 2001:db8::1234:5678:1.2.3.4
```

PowerShell

インスタンスから IPv6 アドレスを割り当て解除するには

Unregister-EC2Ipv6AddressList コマンドレットを使用します。

```
Unregister-EC2Ipv6AddressList `
    -NetworkInterfaceId eni-1234567890abcdef0 `
    -Ipv6Address 2001:db8::1234:5678:1.2.3.4
```

EC2 インスタンスのセカンダリ IP アドレス

ネットワークインターフェイスに最初に割り当てられた IPv4 アドレスを、プライマリ IP アドレスといいます。セカンダリ IP アドレスは、ネットワークインターフェイスに追加で割り当てられた IPv4 アドレスです。詳細については、「the section called “複数の IP アドレス”」を参照してください。

また、1 つのインスタンスに複数の IPv6 アドレスを割り当てることができます。詳細については、「the section called “IPv6 アドレス”」を参照してください。

タスク

- インスタンスにセカンダリ IP アドレスを割り当てる
- セカンダリ IP アドレスを使用するようにオペレーティングシステムを設定する
- インスタンスからセカンダリ IP アドレスの割り当てを解除する

インスタンスにセカンダリ IP アドレスを割り当てる

セカンダリ IP アドレスは、インスタンスの起動時に、または起動後に、インスタンスのネットワークインターフェイスに割り当てることができます。

Console

起動時にセカンダリ IP アドレスを割り当てるには

1. インスタンスを起動するための手順に従います。[ネットワーク設定] を設定するときに、[高度なネットワーク設定] を展開します。
2. [セカンダリ IP] で [自動で割り当て] を選択し、Amazon EC2 に割り当てさせる IP アドレスの数を入力します。または、[手動で割り当て] を選択し、IPv4 アドレスを入力します。
3. インスタンスを起動する残りのステップを完了します。

起動後にセカンダリ IP アドレスを割り当てるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択後、[アクション]、[ネットワーク]、[IP アドレスの管理] の順に選択してください。
4. ネットワークインターフェイスを展開します。
5. IPv4 アドレスを追加するには、[IPv4 アドレス] で [新しい IP アドレスの割り当て] を選択します。サブネットの範囲から IPv4 アドレスを入力します。フィールドを空欄のままにすると Amazon EC2 が自動的にアドレスを選択します。
6. [保存] を選択します。

AWS CLI

起動時にセカンダリ IP アドレスを割り当てるには

`--secondary-private-ip-addresses` オプションで run-instances コマンドを使用します。

```
--secondary-private-ip-addresses 10.251.50.12
```

Amazon EC2 に IP アドレスを選択させる場合は、代わりに `--secondary-private-ip-address-count` のオプションを使用します。次の例では、1 つのセカンダリ IP アドレスを割り当てます。

```
--secondary-private-ip-address-count 1
```

または、ネットワークインターフェイスを作成することができます。詳細については、「the section called “ネットワークインターフェイスの作成”」を参照してください。

起動後にセカンダリ IP アドレスを割り当てるには

assign-private-ip-addresses コマンドを `--private-ip-addresses` オプションと共に使用します。

```
aws ec2 assign-private-ip-addresses \
    --network-interface-ids eni-1234567890abcdef0 \
    --private-ip-addresses 10.251.50.12
```

Amazon EC2 に IPv4 アドレスを選択させる場合は、代わりに `--secondary-private-ip-address-count` のパラメータを使用します。次の例では、1 つの IPv4 アドレスを割り当てます。

```
aws ec2 assign-private-ip-addresses \
    --network-interface-ids eni-1234567890abcdef0 \
    --secondary-private-ip-address-count 1
```

PowerShell

起動時にセカンダリ IP アドレスを割り当てるには

ネットワークインターフェイスを作成する必要があります。詳細については、「the section called “ネットワークインターフェイスの作成”」を参照してください。

起動後にセカンダリ IP アドレスを割り当てるには

Register-EC2PrivateIpAddress コマンドレットを -PrivateIpAddress パラメータと共に使用します。

```
Register-EC2PrivateIpAddress `
    -NetworkInterfaceId eni-1234567890abcdef0 `
    -PrivateIpAddress 10.251.50.12
```

Amazon EC2 に IPv4 アドレスを選択させる場合は、代わりに -SecondaryPrivateIpAddressCount のパラメータを使用します。次の例では、1 つの IPv4 アドレスを割り当てます。

```
Register-EC2PrivateIpAddress `
    -NetworkInterfaceId eni-1234567890abcdef0 `
    -SecondaryPrivateIpAddressCount 1
```

セカンダリ IP アドレスを使用するようにオペレーティングシステムを設定する

セカンダリ IP アドレスをインスタンスに割り当てたら、追加のプライベート IPv4 アドレスを認識するように、インスタンスのオペレーティングシステムを設定する必要があります。

Linux インスタンス

- Amazon Linux を使用している場合、ec2-net-utils パッケージがこの処理を自動実行します。このパッケージはインスタンスの実行中にアタッチされる追加のネットワークインターフェイスを設定し、DHCP リースの更新中にセカンダリ IPv4 アドレスを更新して、関連するルーティングルールを更新します。システムに応じて、sudo systemctl restart systemd-networkd (AL2023) または sudo service network restart (Amazon Linux 2) のいずれか 1 つのコマンドを使用して、インターフェイスのリストをすぐに更新できます。次のコマンドを使用して最新のリストを表示できます: ip addr li。ネットワーク構成を手動で構成する必要がある場合、ec2-net-utils パッケージを削除できます。詳細については「ec2-net-utils を使用したネットワークインターフェイスの設定」を参照してください。
- 別の Linux ディストリビューションを使用している場合、Linux ディストリビューションのドキュメントを参照してください。追加のネットワークインターフェイスとセカンダリ IPv4 アドレスの設定に関する情報が記載されています。同じサブネットのインスタンスに複数のインターフェイスがある場合、非対称のルーティングに対処する方法についてはルーティングルールの使用に関する情報を検索してください。

Windows インスタンス

詳細については、「Windows インスタンスのセカンダリプライベート IPv4 アドレスを設定する」を参照してください。

インスタンスからセカンダリ IP アドレスの割り当てを解除する

セカンダリ IP アドレスが不要になったら、インスタンスやネットワークインターフェイスからその割り当てを解除できます。セカンダリプライベート IPv4 アドレスをネットワークインターフェイスから割り当て解除した場合、Elastic IP アドレス (存在する場合) の関連付けも解除されます。

Console

インスタンスからセカンダリプライベート IPv4 アドレスを割り当て解除するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択してください。
3. インスタンスを選択し、[アクション]、[ネットワーク]、[IP アドレスの管理] の順に選択してください。
4. ネットワークインターフェイスを展開します。[IPv4 アドレス] で、割り当て解除する IPv4 アドレスに対して [割り当て解除] を選択してください。
5. [保存] を選択します。

AWS CLI

セカンダリプライベート IP アドレスの割り当てを解除するには

unassign-private-ip-addresses コマンドを使用します。

```
aws ec2 unassign-private-ip-addresses \
    --network-interface eni-1234567890abcdef0\
    --private-ip-addresses 10.251.50.12
```

PowerShell

セカンダリプライベート IP アドレスの割り当てを解除するには

Unregister-EC2PrivateIpAddress コマンドレットを使用します。

```
Unregister-EC2PrivateIpAddress `
    -NetworkInterface eni-1234567890abcdef0 `
    -PrivateIpAddress 10.251.50.12
```

Windows インスタンスのセカンダリプライベート IPv4 アドレスを設定する

インスタンスに複数のプライベート IPv4 IP アドレスを指定できます。インスタンスにセカンダリプライベート IPv4 アドレスを割り当てたら、セカンダリプライベート IPv4 アドレスを認識するようにインスタンスのオペレーティングシステムを設定する必要があります。

> **Note**
>
> この手順は Windows Server 2022 を対象にしています。Windows インスタンスのオペレーティングシステムによって、これらの手順の実施方法が異なる場合があります。

タスク

- 前提条件
- ステップ 1: インスタンスで静的 IP アドレス指定を設定する
- ステップ 2: インスタンスにセカンダリプライベート IP アドレスを設定する
- ステップ3: セカンダリプライベート IP アドレスを使用するようにアプリケーションを設定する

前提条件

- セカンダリプライベート IPv4 アドレスを、インスタンスのネットワークインターフェイスに割り当てます。セカンダリプライベート IPv4 アドレスは、インスタンスの起動時または実行の開始後に割り当てることができます。詳細については「インスタンスにセカンダリ IP アドレスを割り当てる」を参照してください。

ステップ 1: インスタンスで静的 IP アドレス指定を設定する

Windows インスタンスが複数の IP アドレスを使用するには、インスタンスが DHCP サーバーではなく、静的 IP アドレス指定を使用するように設定する必要があります。

> **Important**
>
> インスタンスで静的 IP アドレス指定を設定する場合、IP アドレスは、コンソール、CLI、または API で表示される IP アドレスと正確に一致している必要があります。これらの IP アドレスを誤って入力すると、インスタンスは到達不能になる可能性があります。

Windows インスタンスで静的 IP アドレス指定を設定するには

1. インスタンスに接続します。
2. 次のステップを実行してインスタンスの IP アドレス、サブネットマスク、デフォルトゲートウェイアドレスを見つけます:

   - PowerShell で次のコマンドを実行します。

     ```
     ipconfig /all
     ```

     出力を確認し、ネットワークインターフェイスの [IPv4 アドレス]、[サブネットマスク]、[デフォルトゲートウェイ]、[DNS サーバー] の値を書き留めます。出力は次の例のようになります。

     ```
     ...

     Ethernet adapter Ethernet 4:

        Connection-specific DNS Suffix  . : us-west-2.compute.internal
        Description . . . . . . . . . . . : Amazon Elastic Network Adapter #2
        Physical Address. . . . . . . . . : 02-9C-3B-FC-8E-67
        DHCP Enabled. . . . . . . . . . . : Yes
        Autoconfiguration Enabled . . . . : Yes
        Link-local IPv6 Address . . . . . : fe80::f4d1:a773:5afa:cd1%7(Preferred)
        IPv4 Address. . . . . . . . . . . : 10.200.0.128(Preferred)
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Lease Obtained. . . . . . . . . . : Monday, April 8, 2024 12:19:29 PM
        Lease Expires . . . . . . . . . . : Monday, April 8, 2024 4:49:30 PM
        Default Gateway . . . . . . . . . : 10.200.0.1
        DHCP Server . . . . . . . . . . . : 10.200.0.1
        DHCPv6 IAID . . . . . . . . . . . : 151166011
        DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-2D-67-AC-FC-12-34-9A-BE-A5-
     E7
     ```

```
   DNS Servers . . . . . . . . . . . : 10.200.0.2
   NetBIOS over Tcpip. . . . . . . . : Enabled
```

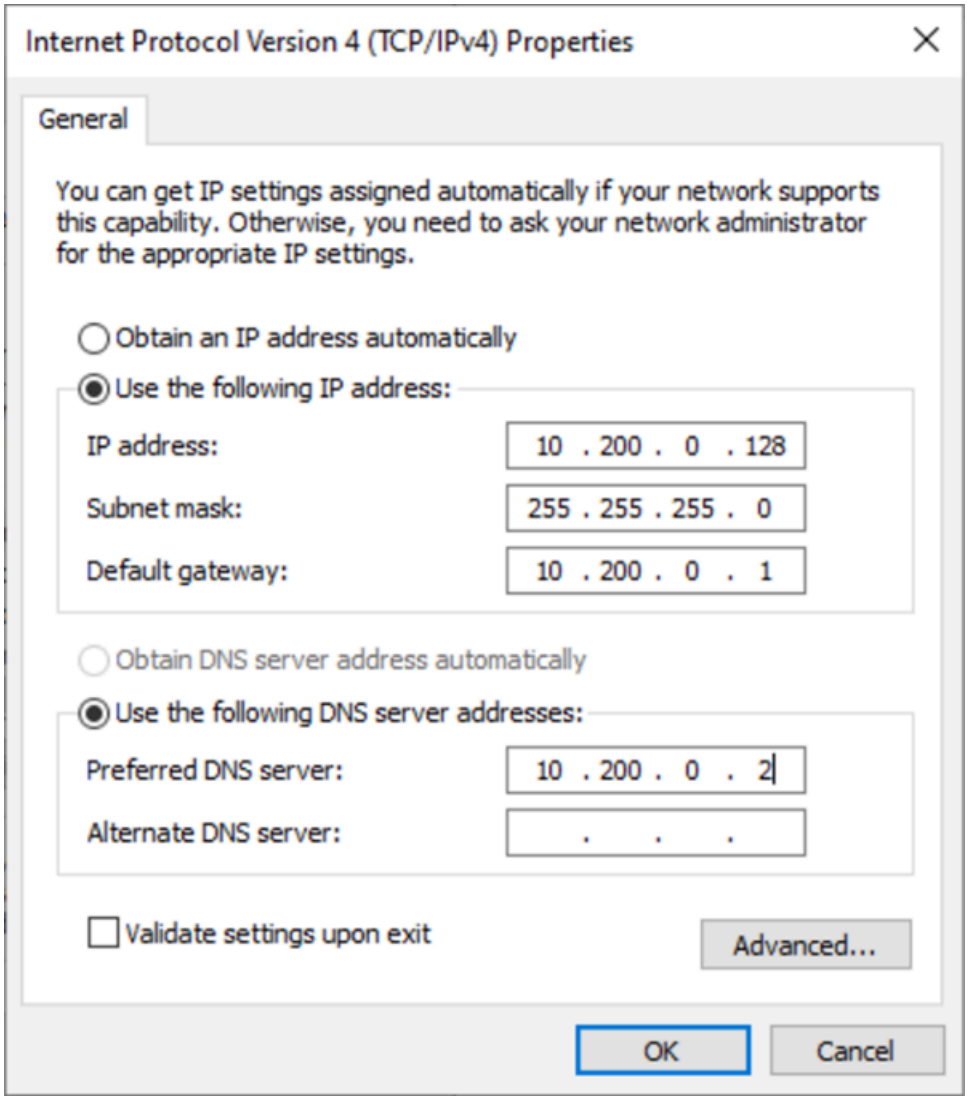
3. PowerShell で次のコマンドを実行して [ネットワークと共有センター] を開きます。

```
& $env:SystemRoot\system32\control.exe ncpa.cpl
```

4. ネットワークインターフェイス (ローカルエリア接続またはイーサネット) のコンテキスト (右クリック) メニューを開き、[プロパティ] を選択します。
5. [Internet Protocol Version 4 (TCP/IPv4)]、[Properties] の順に選択します。
6. [Internet Protocol Version 4 (TCP/IPv4) Properties] ダイアログボックスで [Use the following IP address] を選択して以下の値を入力し、[OK] を選択します。

フィールド	値
IP アドレス	上記のステップ 2 で入手した IPv4 アドレス。
サブネットマスク	上記のステップ 2 で入手したサブネットマスク。
デフォルトゲートウェイ	上記のステップ 2 で入手したデフォルトゲートウェイアドレス。
任意 DNS サーバー	上記のステップ 2 で入手した DNS サーバー。
代替の DNS サーバー	上記のステップ 2 で入手した代替 DNS サーバー。代替 DNS サーバーが表示されなかった場合は、このフィールドを空白のままにしてください。

> **Important**
>
> IP アドレスを現在の IP アドレス以外の値に設定すると、インスタンスへの接続が失われます。



インスタンスで DHCP の使用から静的アドレス指定に変換する間、Windows インスタンスへの RDP 接続が数秒間失われます。インスタンスは以前と同じ IP アドレス情報を保持していますが、この情報は静的であり、DHCP によって管理されていません。

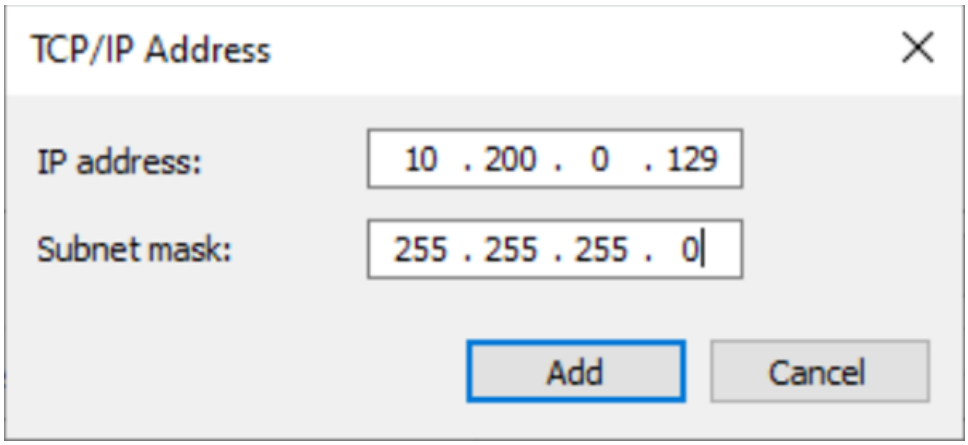
ステップ 2: インスタンスにセカンダリプライベート IP アドレスを設定する

Windows インスタンスで静的 IP アドレスをセットアップしたら、セカンダリ プライベート IP アドレスを設定できます。

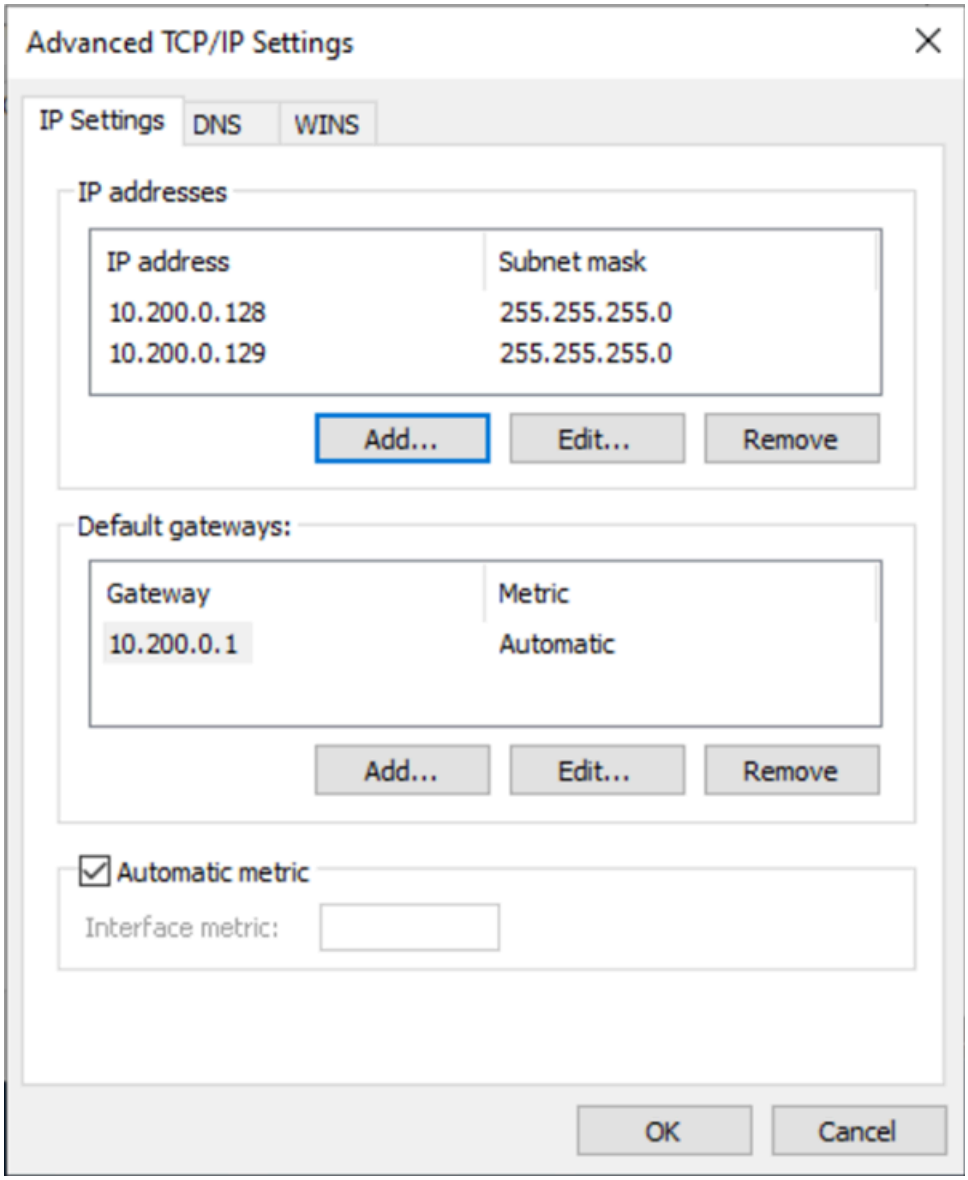
セカンダリ IP アドレスを構成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインで [Instances] を選択し、インスタンスを選択します。
3. [ネットワーキング] で、セカンダリ IP アドレスをメモします。
4. インスタンスに接続します。
5. Windows インスタンスで、[スタート]、[コントロールパネル] の順に選択します。
6. [ネットワークとインターネット]、[ネットワークと共有センター] の順に選択します。
7. ネットワークインターフェイス (ローカルエリア接続またはイーサネット) を選択し、[プロパティ] を選択します。
8. [ローカルエリア接続のプロパティ] ページで、[インターネットプロトコルバージョン 4 (TCP/IPv4)]、[プロパティ]、[詳細設定] の順に選択します。
9. [Add] を選択します。
10. [TCP/IP アドレス] ダイアログボックスで、[IP アドレス] にセカンダリプライベート IP アドレスを入力します。[サブネットマスク] に、ステップ 1: インスタンスで静的 IP アドレス指定を設定する でプライマリプライベート IP アドレス用に入力したものと同じサブネットマスクを入力し、[追加] を選択します。



11. IP アドレス設定を確認して、[OK] を選択します。



12. [OK]、[閉じる] の順に選択します。
13. オペレーティングシステムにセカンダリプライベート IP アドレスが追加されたことを確認するには、PowerShell で、`ipconfig /all` コマンドを実行します。出力は次のようになります。

```
Ethernet adapter Ethernet 4:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Amazon Elastic Network Adapter #2
   Physical Address. . . . . . . . . : 02-9C-3B-FC-8E-67
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::f4d1:a773:5afa:cd1%7(Preferred)
   IPv4 Address. . . . . . . . . . . : 10.200.0.128(Preferred)
```

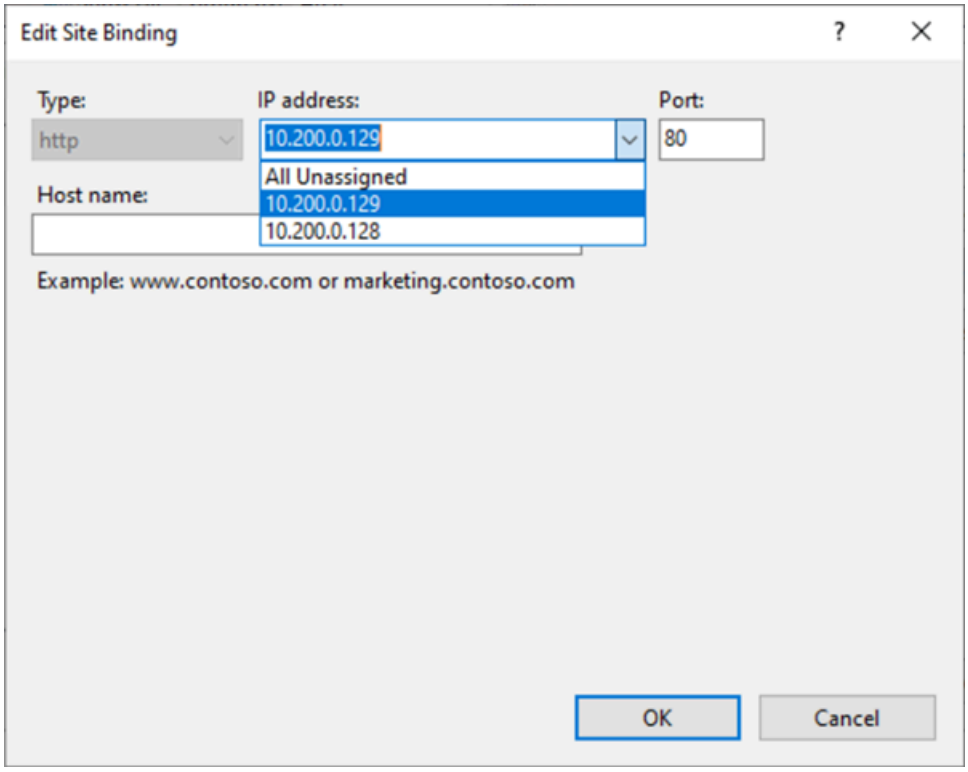
```
Subnet Mask . . . . . . . . . . . : 255.255.255.0
IPv4 Address. . . . . . . . . . . : 10.200.0.129(Preferred)
Subnet Mask . . . . . . . . . . . : 255.255.255.0
Default Gateway . . . . . . . . . : 10.200.0.1
DHCPv6 IAID . . . . . . . . . . . : 151166011
DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-2D-67-AC-FC-12-34-9A-BE-A5-E7
DNS Servers . . . . . . . . . . . : 10.200.0.2
NetBIOS over Tcpip. . . . . . . . : Enabled
```

ステップ3: セカンダリプライベート IP アドレスを使用するようにアプリケーションを設定する

セカンダリプライベート IP アドレスを使用するように任意のアプリケーションを設定できます。例えば、インスタンスが IIS でウェブサイトを実行している場合、セカンダリプライベート IP アドレスを使用するように IIS を設定できます。

セカンダリプライベート IP アドレスを使用するように IIS を設定するには

1. インスタンスに接続します。
2. インターネットインフォメーションサービス (IIS) マネージャーを開きます。
3. [Connections] ペインで、[Sites] を展開します。
4. ウェブサイトのコンテキスト (右クリック) メニューを開き、[Edit Bindings] を選択します。
5. [Site Bindings] ダイアログボックスの [Type] で、[http]、[Edit] の順に選択します。
6. [Edit Site Binding] ダイアログボックスの [IP address] で、セカンダリプライベート IP アドレスを選択します。(デフォルトでは、各ウェブサイトはすべての IP アドレスからの HTTP 要求を受け付けます)。



7. [OK]、[閉じる] の順に選択します。

EC2 インスタンスのホスト名とドメイン

EC2 インスタンスのホスト名とドメインを理解することは、Amazon EC2 インスタンスの効果的な管理とアクセスのために重要です。各 EC2 インスタンスは、プライベートとパブリックの異なる 2 タイプのホスト名を持つことができます。各ホスト名はそれぞれ異なる目的に対応しており、それぞれ固有の命名規則に従います。

このトピックでは、EC2 インスタンスホスト名の構造、構築方法、および使用可能なさまざまなホスト名タイプについて説明します。ホスト名の設定を表示および変更する方法、各タイプを使用するタイミング、および AWS 環境でのホスト名管理のベストプラクティスについて説明します。

内容

- EC2 インスタンスのホスト名とドメインについて
- ホスト名タイプ

EC2 インスタンスのホスト名とドメインについて

EC2 インスタンスアドレスは、複数のコンポーネントで構成されています。インスタンスのプライベート IPv4 アドレスを使用する EC2 インスタンスアドレスの例を次に示します。

```
   IP address         Domain name
   #--------# #-----------------------#
ip-10-24-34-0.us-west-2.compute.internal
#-----------#
  Hostname
#--------------------------------------#
    Fully qualified domain name (FQDN)
```

コードの説明は以下のとおりです。

- IP アドレス: インスタンスに関連付けられたプライマリネットワークインターフェイスのプライマリ IPv4 アドレス。
- ホスト名: 特定の EC2 インスタンスのローカル名 (オペレーティングシステムとローカルネットワークの特定に使用)
- ドメイン名: FQDN のうち AWS が提供する部分
- 完全修飾ドメイン名 (FQDN): ホスト名とドメイン名の両方を含む完全なアドレス。これは、ネットワーク間でインスタンスにアクセスするために使用する、グローバルに一意の完全な識別子となります。

ホスト名とドメイン名の形式は上記の例とは異なる場合があり、インスタンスまたはインスタンスにアタッチされるプライマリネットワークインターフェイス向けに選択したホスト名タイプに応じた形式となります。このセクションでは、ホスト名タイプのオプションについて説明します。

ホスト名タイプ

AWS は、プライベートとパブリックの 2 種類のホスト名を提供します。次の表は、解決方法、設定方法、およびそれぞれのタイプを使用すべき状況など、プライベートホスト名とパブリックホスト名の主な違いを比較したものです。

	プライベートホスト名	パブリックホスト名
DNS 解決	プライベートホスト名は、パブリックインターネットからアクセスで	パブリックホスト名は、パブリックインターネットからアクセスでき

	プライベートホスト名	パブリックホスト名
	きないプライベート FQDN を有効にします。プライベートホスト名では、VPC 内のプライベート IPv4 および IPv6 GUA アドレスに解決するリクエストのみが許可されます。	るパブリック FQDN を有効にします。パブリックホスト名を設定すると、VPC 内のプライベート IPv4 と IPv6 GUA、およびインターネット上のパブリック IP (スプリットホライズン DNS) へのリクエストを解決できます。
設定	プライベートホスト名はインスタンスレベルで設定されます。	パブリックホスト名は、ネットワークインターフェイスレベルで設定されます。
どのようなときに使うか	• 内部通信用の VPC 内でのみ解決されるホスト名向け • 内部命名規則を非公開にしたい場合 • 外部 DNS の考慮事項なしで内部リソース管理を簡素化したい場合	• インターネットから IPv4 と IPv6 の両方を介してインスタンスにアクセスしたい場合 • IPv4 環境から IPv6 環境へのスムーズな移行を有効にしたい場合

内容

- パブリックホスト名
- プライベートホスト名

パブリックホスト名

パブリックホスト名を使用すると、インスタンスのパブリック IPv4 アドレスまたは IPv6 アドレスに解決されるホスト名を使用して EC2 インスタンスにアクセスでき、IPv6 への移行がスムーズになります。

パブリックホスト名を使用すると、以下のメリットがあります。

- IPv4 と IPv6 の両方を介して EC2 インスタンスにアクセスでき、リソースへの接続方法の柔軟性が高まる。

- 自分の好きなタイミングで IPv4 環境から IPv6 環境に移行できる。たとえば、データベースの移行とアプリケーションの移行を切り離して処理することで、複雑さとリスクを軽減できます。
- 複数のホスト名オプション (IPv4 のみ、IPv6 のみ、デュアルスタック) を使用することで、自動で適切な IP アドレスに解決できる。
- VPC 内からクエリを実行するとホスト名がプライベート IP アドレスに解決される、つまりスプリットホライズン DNS によるセキュリティ向上のメリットもあります。

内容

- パブリックホスト名のタイプと、それぞれのタイプを使用すべき状況
- パブリックホスト名を表示する
- パブリックホスト名タイプを変更する

パブリックホスト名のタイプと、それぞれのタイプを使用すべき状況

パブリックホスト名を使用するには、既存のネットワークインターフェイスを変更する必要があります。このセクションでは、3 つのパブリックホスト名タイプのオプションと、それぞれのタイプを使用すべき状況について説明します。

- デュアルスタック – IP ベース名
  - これは、IPv4 から IPv6 に移行する場合、あるいは移行を計画している場合に最適なオプションです。このオプションを使用すると、IPv4 と IPv6 の両方を介した接続が可能になり、どちらか一つのプロトコルのみを使用しているクライアントも柔軟に接続できます。このオプションの場合、クライアントは IPv6 への移行中も同じホスト名を維持できます。
  - VPC 内からのリクエストは、ネットワークインターフェイスのプライベート IPv4 アドレスと IPv6 グローバルユニキャストアドレス (GUA) の両方に解決されます。インターネットからのリクエストは、ネットワークインターフェイスのパブリック IPv4 と IPv6 GUA アドレスの両方に解決されます。
  - 例
    - このオプションを選択すると、このネットワークインターフェイス用にデュアルスタック FQDN が生成されます。こちらが、生成される FQDN の例です。
      - f5lnz-0khrm-nt2u3-gyqqt-nbdl5-q3cdpO.ap-southeast-2.ip.aws
    - コードの説明は以下のとおりです。
      - f5lnz-0khrm-nt2u3-gyqqt-nbdl5-q3cdpO は、ネットワークインターフェイス上のプライマリパブリック IPv6 アドレス (f5lnz-0khrm-nt2u3-gyqt-nbdl5) の base36 表現と、ネットワーク

インターフェイス上のプライマリパブリック IPv4 アドレス (q3cdpO) の base36 表現を組み合わせたホスト名です。

- f5lnz-0khrm-nt2u3-gyqqt-nbdl5 は、Amazon DNS リゾルバーによって IPv6 GUA アドレス FFFF:1407:4:f000:81d:2689:1066:4489 に解決されます。これは、ネットワークインターフェイスに割り当てられた最初の IPv6 GUA です。
- q3cdpO は IPv4 アドレス 52.54.55.56 に解決されます。これは、プライマリネットワークインターフェイスにアタッチされたパブリック IPv4 アドレスです。
- ap-southeast-2 は、ネットワークインターフェイスが存在するサブネットのリージョンです。
- ip.aws は、AWS が提供するドメインです。

- IPv6 – IP ベースの名前
  - これは、すでに IPv6 に移行していて、IPv6 経由の接続のみを必要とする場合に適しています。
  - VPC 内またはインターネットからのリクエストは、ネットワークインターフェイスの IPv6 GUA に解決されます。
  - 例
    - このオプションを選択すると、このネットワークインターフェイス用に FQDN が生成されます。こちらが、生成される FQDN の例です。
      - f5lnz-0khrm-nt2u3-gyqqt-nbdl5.ap-southeast-2.ip.aws
    - コードの説明は以下のとおりです。
      - f5lnz-0khrm-nt2u3-gyqqt-nbdl5 は、ネットワークインターフェイス上のプライマリパブリック IPv6 アドレスの base36 表現であるホスト名です。
      - f5lnz-0khrm-nt2u3-gyqqt-nbdl5 は、Amazon DNS リゾルバーによって IPv6 GUA アドレス FFFF:1407:4:f000:81d:2689:1066:4489 に解決されます。これは、ネットワークインターフェイスに割り当てられた最初の IPv6 GUA です。
      - ap-southeast-2 は、ネットワークインターフェイスが存在するサブネットのリージョンです。
      - ip.aws は、AWS が提供するドメインです。
- IPv4 – IP ベースの名前
  - これは、このネットワークインターフェイスを使用するインスタンスが IPv6 への移行中に IPv4 アクセスを維持する必要がある場合、またはインスタンスで実行されているアプリケーションまたはシステムが IPv4 のみをサポートしている場合に適しています。これは、IPv4 接続のみの維持を必要としており、ワークロードには IPv6 サポートが必要ない場合に最適なオプションで

す。例えば、IPv6 への移行に際して、一部のアプリケーションを IPv4 に残してその他のアプリケーションを IPv6 に移行させたい場合があります。

- VPC 内からのリクエストは、ネットワークインターフェイスのプライベートプライマリ IPv4 アドレスに解決されます。インターネットからのリクエストは、ネットワークインターフェイスのパブリック IPv4 アドレスに解決されます。
- 例
  - このオプションを選択すると、このネットワークインターフェイスに IPv4 対応のパブリックホスト名が生成されます。こちらは、生成される DNS 名の例です。
    - ec2-52-54-55-66.ap-southeast-2.compute.amazonaws.com
  - コードの説明は以下のとおりです。
    - ec2-52-54-55-66 は、ネットワークインターフェイスのプライマリパブリック IPv4 アドレスの base36 表現であるホスト名です。
    - ec2-52-54-55-66 は IPv4 アドレス 52.54.55.56 に解決されます。これは、プライマリネットワークインターフェイスにアタッチされたパブリック IPv4 アドレスです。
    - ap-southeast-2 は、ネットワークインターフェイスが存在するサブネットのリージョンです。
    - ip.aws は、AWS が提供するドメインです。

> **Important**
>
> 上記の例では、ホスト名の生成に IP アドレスが使用されていることがわかります。プライマリプライベート IPv4 アドレスまたはネットワークインターフェイスに割り当てられた最初の IPv6 GUA を変更すると、ホスト名のうち IP アドレスに反映される部分が変更され、以前に生成されたパブリックホスト名は無効になります。さらに、プライマリ IPv4 パブリックアドレスを変更すると、「Amazon EC2 ユーザーガイド」で説明しているインスタンスメタデータサービス (IMDS) のダウンストリーム更新が強制されるため、EC2 インスタンス IMDS メタデータは自動的に更新されます。

パブリックホスト名を表示する

ネットワークインターフェイスがある VPC に EnableDnsHostnames および EnableDnsSupport の両方が有効になっていない場合、ホスト名タイプは定義または生成されません。

Console

インスタンスまたはプライマリネットワークインターフェイスのパブリックホスト名を表示できます。

インスタンスのホスト名タイプと DNS 名を表示する方法

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスのチェックボックスをオンにします。
4. [ネットワーク] タブの [ホスト名と DNS] で、次のものを探します。
   - [パブリックホスト名タイプ]
   - [パブリック DNS]
   - [IPv4 のみの IP ベースの名前]
   - [IPv6 のみ – IP ベースの名前]
   - [デュアルスタック – IP ベースの名前]

ネットワークインターフェイスのホスト名タイプと DNS 名を表示する方法

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Network Interfaces] を選択してください。
3. 検索フィールドに、インスタンスの ID を入力します。ネットワークインターフェイスの ID を選択して、その詳細ページを開きます。
4. [ホスト名と DNS] で次のものを探します。
   - [パブリックホスト名タイプ]
   - Public DNS name
   - [パブリック IPv4 DNS 名]
   - [パブリック IPv6 DNS 名]
   - [パブリックデュアルスタック DNS 名]

AWS CLI

ネットワークインターフェイスのホスト名タイプと DNS 名を表示する方法

describe-network-interfaces コマンドを使用します。

```
aws ec2 describe-network-interfaces \
    --network-interface-id eni-1234567890abcdef0 \
    --query NetworkInterfaces[].PublicIpDnsNameOptions
```

以下は出力の例です。ホスト名タイプが public-dual-stack-dns-name であるため、DNS ホスト名は PublicDualStackDnsName と同じです。

```
[
     {
          "DnsHostnameType": "public-dual-stack-dns-name",
          "PublicIpv4DnsName": "ec2-52-54-55-66.ap-
southeast-2.compute.amazonaws.com",
          "PublicIpv6DnsName": "f5lnz-0khrm-nt2u3-gyqqt-nbdl5.ap-
southeast-2.ip.aws",
          "PublicDualStackDnsName": "f5lnz-0khrm-nt2u3-gyqqt-nbdl5-q3cdp0.ap-
southeast-2.ip.aws"
     }
]
```

PowerShell

ネットワークインターフェイスのホスト名タイプと DNS 名を表示する方法

Get-EC2NetworkInterface コマンドレットを使用します。

```
(Get-EC2NetworkInterface `
    -NetworkInterfaceId eni-1234567890abcdef0).PublicIpDnsNameOptions
```

以下は出力の例です。ホスト名タイプが public-dual-stack-dns-name であるため、DNS ホスト名は PublicDualStackDnsName と同じです。

```
DnsHostnameType        : public-dual-stack-dns-name
PublicDualStackDnsName : f5lnz-0khrm-nt2u3-gyqqt-nbdl5-q3cdp0.ap-southeast-2.ip.aws
PublicIpv4DnsName      : ec2-52-54-55-66.ap-southeast-2.compute.amazonaws.com
PublicIpv6DnsName      : f5lnz-0khrm-nt2u3-gyqqt-nbdl5.ap-southeast-2.ip.aws
```

パブリックホスト名タイプを変更する

パブリックホスト名タイプのオプションは、ネットワークインターフェイスに関連付けられた IP アドレスによって異なります。

- ネットワークインターフェイスにパブリック IPv4 アドレスのみがある場合、ホスト名タイプは IPv4 – IP ベースの名前である必要があります。
- ネットワークインターフェイスに IPv6 アドレスのみがある場合、ホスト名タイプは IPv6 – IP ベースの名前である必要があります。
- ネットワークインターフェイスにパブリック IPv4 アドレスおよび IPv6 アドレスの両方がある場合、ホスト名タイプはデュアルスタック – IP ベースの名前である必要があります。

前提条件

- ネットワークインターフェイスは、関連付けられたパブリック IPv4 アドレスまたは IPv6 アドレスが必要です。
- ネットワークインターフェイスが存在する VPC には、EnableDnsHostnames および EnableDnsSupport が有効になっている必要があります。「Amazon VPC ユーザーガイド」の「VPC の DNS 属性の表示と更新」を参照してください。

Console

パブリックホスト名タイプを変更する方法

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Network Interfaces] を選択してください。
3. 検索フィールドに、インスタンスの ID を入力します。ネットワークインターフェイスのチェックボックスをオンにします。

   または、インスタンスの詳細ページで [ネットワーク] タブを選択し、デバイスインデックス 0 のネットワークインターフェイスの ID を選択します。
4. [アクション] および [パブリックホスト名タイプの変更] を選択します。
5. 次のいずれかのオプションを選択します。

   - デュアルスタック – IP ベースの名前: ネットワークインターフェイスのデュアルスタックパブリックホスト名。VPC 内からのリクエストは、ネットワークインターフェイスのプライベート IPv4 アドレスと IPv6 グローバルユニキャストアドレスの両方に解決されます。

インターネットからのリクエストは、ネットワークインターフェイスのパブリック IPv4 と IPv6 GUA アドレスの両方に解決されます。

- IPv4 – IP ベース名: ネットワークインターフェイスの IPv4 対応パブリックホスト名。VPC 内からのリクエストは、ネットワークインターフェイスのプライベートプライマリ IPv4 アドレスに解決されます。インターネットからのリクエストは、ネットワークインターフェイスのパブリック IPv4 アドレスに解決されます。
- IPv6 – IP ベース名: ネットワークインターフェイスの IPv6 対応パブリックホスト名。VPC 内またはインターネットからのリクエストは、ネットワークインターフェイスの IPv6 GUA に解決されます。

6. [Modify] (変更) を選択します。

AWS CLI

パブリックホスト名タイプを変更する方法

modify-public-ip-dns-name-options コマンドを使用します。

```
aws ec2 modify-public-ip-dns-name-options \
    --network-interface-id eni-1234567890abcdef0 \
    --hostname-type public-dual-stack-dns-name
```

以下は出力の例です。

```
{
    "Successful": true
}
```

PowerShell

パブリックホスト名タイプを変更する方法

Edit-EC2PublicIpDnsNameOption コマンドレットを使用します。

```
Edit-EC2PublicIpDnsNameOption `
    -NetworkInterfaceId eni-1234567890abcdef0 `
    -HostNameType public-dual-stack-dns-name
```

プライベートホスト名

このセクションでは、VPC サブネットでインスタンスを起動する際に使用できる Amazon EC2 インスタンスのプライベートホスト名について説明します。

ネットワーク上の EC2 インスタンスはプライベートホスト名によって区別されます。例えば、ネットワーク上の一部またはすべてのインスタンスと通信するスクリプトを実行する場合、インスタンスのプライベートホスト名を使用できます。

内容

- プライベートホスト名タイプ
- リソース名と IP 名を検索する場所
- リソース名と IP 名の選択
- Amazon EC2 のリソースベースの命名オプションを変更する

プライベートホスト名タイプ

EC2 インスタンスが VPC で起動されるときのゲスト OS ホスト名には、次の 2 つのプライベートホスト名タイプがあります。

- [IP 名]: 従来の命名スキームでは、インスタンスの起動時に、インスタンスのプライベート IPv4 アドレスがインスタンスのホスト名に含まれます。IP 名は EC2 インスタンスの存続中に存在します。プライベート DNS ホスト名として使用すると、プライベート IPv4 アドレス (A レコード) のみが返されます。
- [リソース名]: インスタンスを起動すると、EC2 インスタンス ID がインスタンスのホスト名に含まれます。リソース名はEC2 インスタンスの存続中に存在します。プライベート DNS ホスト名として使用すると、プライベート IPv4 アドレス (A レコード) と IPv6 グローバルユニキャストアドレス (AAAA レコード) の両方を返すことができます。

EC2 インスタンスのゲスト OS ホスト名 のタイプはサブネット設定によって異なります。

- インスタンスが IPv4 専用サブネットで起動された場合、IP 名またはリソース名を選択できます。
- インスタンスがデュアルスタック (IPv4+IPv6) サブネットで起動されている場合、IP 名またはリソース名を選択できます。
- インスタンスが IPv6 専用サブネットで起動された場合、リソース名が自動的に使用されます。

内容

- IP 名
- リソース名
- IP 名とリソース名の違い

IP 名

[IP name] (IP 名) の [Hostname type] (ホスト名タイプ) を使用して EC2 インスタンスを起動すると、ゲスト OS ホスト名がプライベート IPv4 アドレスを使用するように設定されます。

- us-east-1 でのインスタンスのフォーマット: *private-ipv4-address*.ec2.internal
- 例:*ip-10-24-34-0*.ec2.internal
- その他の AWS リージョンのインスタンスのフォーマット: *private-ipv4-address.region*.compute.internal
- 例:*ip-10-24-34-0.us-west-2*.compute.internal

リソース名

EC2 インスタンスを IPv6 専用サブネットで起動すると、[Resource name] (リソース名) の[Hostname type] (ホスト名タイプ) がデフォルトで選択されます。IPv4 専用またはデュアルスタック (IPv4+IPv6) サブネットでインスタンスを起動すると、[Resource name] (リソース名) は選択できるオプションです。インスタンスを起動してから、ホスト名設定を管理できます。詳細については、Amazon EC2 のリソースベースの命名オプションを変更するを参照してください。

[Resource name] (リソース名) の [Hostname type] (ホスト名タイプ) を使用して EC2 インスタンスを起動すると、ゲスト OS ホスト名が EC2 インスタンス ID を使用するように設定されます。

- us-east-1 でのインスタンスのフォーマット: *ec2-instance-id*.ec2.internal
- 例:*i-0123456789abcdef*.ec2.internal
- その他の AWS リージョンのインスタンスのフォーマット: *ec2-instance-id.region*.compute.internal
- 例:*i-0123456789abcdef.us-west-2*.compute.internal

IP 名とリソース名の違い

IP 名とリソース名の両方の DNS クエリが共存して、下位互換性を確保し、ホスト名に対する IP ベースの命名 からリソースベースの命名に移行できます。IP 名に基づくプライベート DNS ホスト名の場合、インスタンスの DNS A レコードクエリに応答するかどうかを設定することはできません。DNS A レコードクエリは、ゲスト OS ホスト名の設定に関係なく、常に応答されます。対照的に、リソース名に基づくプライベート DNS ホスト名の場合、インスタンスの DNS A または DNS AAAA クエリに応答するかどうかを設定できます。インスタンスの起動時またはサブネットの変更時に、レスポンスの動作を設定します。詳細については、「Amazon EC2 のリソースベースの命名オプションを変更する」を参照してください。

リソース名と IP 名を検索する場所

Amazon EC2 コンソールでホスト名のタイプ、リソース名、および IP 名を確認できます。

内容

- EC2 インスタンスを作成する場合
- 既存の EC2 インスタンスの詳細を表示する場合

EC2 インスタンスを作成する場合

EC2 インスタンスを作成する際、選択したサブネットのタイプに応じて、[Resource name] (リソース名) の [Hostname type] (ホスト名タイプ) が使用可能になるか、または選択されて変更できない場合があります。このセクションでは、ホスト名タイプのリソース名と IP 名が表示されるシナリオについて説明します。

シナリオ 1

ウィザード (コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動するを参照) で EC2 インスタンスを作成し、詳細を設定するときに IPv6 専用に設定したサブネットを選択します。

この場合、[Resource name] (リソース名) の [Hostname type] (ホスト名タイプ) は自動的に選択され、変更できません。[Enable IP name IPv4 (A record) DNS requests] (IP 名 IPv4 (A レコード) DNS リクエストを有効にする) の [DNS Hostname] (DNS ホスト名) オプションと [Enable resource-based IPv4 (A record) DNS requests] (リソースベースの IPv4 (A レコード) DNS リクエストを有効にする) は自動的に選択解除され、変更できません。[Enable resource-based IPv6 (AAAA record) DNS requests] (リソースベースの IPv6 (AAAA レコード) DNS リクエストを有効にする) がデフォルトで

選択されていますが、変更可能です。選択した場合、リソース名への DNS リクエストはこの EC2 インスタンスの IPv6 アドレス (AAAA レコード) に解決されます。

シナリオ 2

ウィザード (コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動するを参照) で EC2 インスタンスを作成し、詳細を設定するときに、IPv4 CIDR ブロックまたは IPv4 と IPv6 の両方の CIDR ブロック (「デュアルスタック」) で構成されたサブネットを選択します。

この場合、[Enable IP name IPV4 (A record) DNS requests] (IP 名 IPV4 (A レコード) DNS リクエストを有効にする) は自動的に選択され、変更できません。つまり、IP 名へのリクエストは、この EC2 インスタンスの IPv4 アドレス (A レコード) に解決されます。

オプションはサブネットの設定にデフォルト設定されますが、サブネットの設定に応じてこのインスタンスのオプションを変更できます。

- [Hostname type] (ホスト名タイプ): EC2 インスタンスのゲスト OS ホスト名をリソース名 をリソース名または IP 名にするかを決定します。デフォルト値は [IP name] (IP 名) です。
- [Enable resource-based IPv4 (A record) DNS requests] (リソースベースの IPv4 (A レコード) DNS リクエストを有効にする): リソース名へのリクエストが、この EC2 インスタンスのプライベート IPv4 アドレス (A レコード) に解決されるかどうかを決定します。このオプションはデフォルトで選択されていません。
- [Enable resource-based IPv6 (AAAA record) DNS requests] (リソースベースの IPv6 (AAAA レコード) DNS リクエストを有効にする): リソース名へのリクエストが、この EC2 インスタンスの IPv6 GUA アドレス (AAAA レコード) に解決するかどうかを決定します。このオプションはデフォルトで選択されていません。

既存の EC2 インスタンスの詳細を表示する場合

既存の EC2 インスタンスのホスト名の値は、EC2 インスタンスの [Details] (詳細) タブで確認できます。

- [Hostname type] (ホスト名タイプ): IP 名またはリソース名形式のホスト名
- [Private IP DNS name (IPv4 only)] (プライベート IP DNS 名 (IPv4 専用)): インスタンスのプライベート IPv4 アドレスに常に解決される IP 名
- [Private resource DNS name] (プライベートリソース DNS 名): このインスタンス用に選択された DNS レコードに解決されるリソース名

- [Answer private resource DNS name] (プライベートリソース DNS 名に応答する): リソース名は IPv4 (A)、IPv6 (AAAA)、または IPv4 と IPv6 (A と AAAA) の DNS レコードに解決されます。

さらに、SSH 経由で直接 EC2 インスタンスに接続して、`hostname`コマンドを入力すると、ホスト名が IP 名またはリソース名形式で表示されます。

リソース名と IP 名の選択

EC2 インスタンス (コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動するを参照) を起動する際に [Resource name] (リソース名) の [Hostname type] (ホスト名) を選んだ場合、EC2 インスタンスはリソース名形式のホスト名を使用して起動します。このような場合、この EC2 インスタンスの DNS レコードはリソース名を指すこともあります。これにより、そのホスト名はインスタンスの IPv4 アドレス、IPv6 アドレス、または IPv4 アドレスと IPv6 アドレスの両方に解決されるように選択する柔軟さを実現します。今後 IPv6 を使用する予定がある場合、あるいは現在デュアルスタックのサブネットを使用している場合、[Resource name] (リソース名) の [Hostname type] (ホスト名タイプ) の使用により、DNS レコード自体に変更を加えることなく、インスタンスのホスト名の DNS 解決を変更できます。リソース名を使用すると、EC2 インスタンスで IPv4 および IPv6 DNS リゾリューションを追加して削除できます。

代わりに [IP name] (IP 名) の [Hostname type] (ホスト名タイプ) を選択し、DNS ホスト名として使用する場合、インスタンスの IPv4 アドレスにのみ解決されます。インスタンスに IPv4 アドレスと IPv6 アドレスの両方が関連付けられている場合でも、インスタンスの IPv6 アドレスには解決されません。

Amazon EC2 のリソースベースの命名オプションを変更する

サブネットのホスト名タイプと DNS ホスト名の設定を変更する (そのサブジェクトの後続のすべてのインスタンス起動に影響を与えます) か、EC2 インスタンスの起動後にそのインスタンスの設定を変更することができます。

リソースベースの命名オプション

- [ホスト名のタイプ]: サブネットで起動される EC2 インスタンスのゲスト OS ホスト名のデフォルト設定を決定します。これは、リソース名または IP 名です。
- [DNS ホスト名 IPv4 (A レコード) リクエストを有効にする]: リソース名への DNS リクエスト/クエリがこの EC2 インスタンスのプライベート IPv4 アドレス (A レコード) に解決されるかどうかを決定します。

- [DNS ホスト名 IPv6 (AAAA レコード) リクエストを有効にする]: リソース名への DNS リクエスト/クエリがこの EC2 インスタンスの IPv6 アドレス (AAAA レコード) に解決されるかどうかを決定します。

サブネット

サブネット設定を変更しても、サブネットですでに起動されている EC2 インスタンスの設定は変更されません。

Console

サブネットのオプションを変更するには

Amazon VPC コンソールを開き、サブネットを選択します。[アクション]、[サブネットの設定を編集] の順に選択します。必要に応じて設定を変更し、変更を保存します。

AWS CLI

サブネットのオプションを変更するには

modify-subnet-attribute コマンドを使用します。

```
aws ec2 modify-subnet-attribute \
    --subnet-id subnet-0abcdef1234567890 \
    --private-dns-hostname-type-on-launch resource-name \
    --enable-resource-name-dns-a-record-on-launch \
    --enable-resource-name-dns-aaaa-record-on-launch
```

PowerShell

サブネットのオプションを変更するには

Edit-EC2SubnetAttribute コマンドレットを使用します。

```
Edit-EC2SubnetAttribute `
    -SubnetId subnet-0abcdef1234567890 `
    -PrivateDnsHostnameTypeOnLaunch ResourceName `
    -EnableResourceNameDnsAAAARecordOnLaunch $true `
    -EnableResourceNameDnsARecordOnLaunch $true
```

EC2 インスタンス

考慮事項

- ホスト名タイプを変更するには、まずインスタンスを停止する必要があります。他の 2 つのオプションを変更するためにインスタンスを停止する必要はありません。
- インスタンスストアのルートボリュームを持つインスタンスを停止することはできないため、インスタンスの起動時にのみホスト名タイプと DNS ホスト名のオプションを設定できます。インスタンスストアのルートボリュームをサポートするインスタンスタイプは、C1、C3、D2、I2、M1、M2、M3、R3、X1 のみです。

Console

インスタンスのホスト名タイプと DNS ホスト名のオプションを変更するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. [Use resource based naming as guest OS hostname] (リソースベース命名をゲスト OS ホスト名として使用) の設定を変更する場合、まず EC2 インスタンスを停止してください。それ以外の場合は、この手順をスキップしてください。

   インスタンスを停止し、インスタンスを選択後、[Instance state] (インスタンスの状態)、[Stop instance] (インスタンスの停止) の順にクリックします。
3. インスタンスを選択し、そして [Actions] (アクション)、[Instance settings] (インスタンス設定)、[Change resource based naming options] (リソースベースの命名オプションの変更) を選択します。

   - [Use resource based naming as guest OS hostname] (リソースベース命名をゲスト OS ホスト名として使用) : EC2 インスタンスのゲスト OS ホスト名をリソース名または IP 名にするかを決定します。
   - [Answer DNS hostname IPv4 (A record) requests] (DNS ホスト名 IPv4 (A レコード) リクエストに応答する): リソース名への DNS リクエスト/クエリがこの EC2 インスタンスのプライベート IPv4 アドレスに解決されるかどうかを決定します。
   - [Answer DNS hostname IPv6 (AAAA record) requests] (DNS ホスト名 IPv6 (AAAA レコード) 要求に応答する): リソース名への DNS リクエスト/クエリがこの EC2 インスタンスの IPv6 アドレス (AAAA レコード) に解決されるかどうかを決定します。
4. [保存] を選択します。
5. インスタンスを停止した後は、インスタンスを再起動します。

AWS CLI

インスタンスのホスト名タイプと DNS ホスト名のオプションを変更するには

modify-private-dns-name-options コマンドを使用します。

```
aws ec2 modify-private-dns-name-options \
    --instance-id i-1234567890abcdef0 \
    --private-dns-hostname-type resource-name \
    --enable-resource-name-dns-a-record \
    --enable-resource-name-dns-aaaa-record
```

PowerShell

インスタンスのホスト名タイプと DNS ホスト名のオプションを変更するには

Edit-EC2PrivateDnsNameOption コマンドレットを使用します。

```
Edit-EC2PrivateDnsNameOption `
    -InstanceId i-1234567890abcdef0 `
    -PrivateDnsHostnameType ResourceName `
    -EnableResourceNameDnsAAAARecord $true`
    -EnableResourceNameDnsARecord $true
```

Amazon EC2 に自分の IP アドレスを持ち込む (BYOIP)

オンプレミスのネットワークから、パブリックにルーティング可能な IPv4 または IPv6 アドレス範囲の一部または全部を AWS アカウントに持ち込むことができます。アドレス範囲は引き続き管理でき、AWS を通じてインターネット上でアドレス範囲をアドバタイズできます。アドレス範囲を Amazon EC2 に持ち込むと、そのアドレス範囲がアドレスプールとして AWS アカウントに表示されます。

> **Note**
>
> このドキュメントでは自分の IP アドレス範囲を Amazon EC2 でのみ使用するために持ち込む方法について説明します。AWS Global Acceleratorで使用するために独自の IP アドレス範囲を使用するには「AWS Global Accelerator デベロッパーガイド」の「独自 IP アドレス (BYOIP) の使用」を参照してください。Amazon VPC IP Address Manager で使用する独自

の IP アドレス範囲を導入するには「Amazon VPC IPAM ユーザーガイド」の「チュートリアル: BYOIP アドレス CIDR を IPAM へ」を参照してください。

IP アドレス範囲を AWS に設定すると、AWS は IP アドレス範囲がユーザーによって制御されていることを検証します。範囲を制御していることを示すためには次の 2 つの方法を使用できます。

- IP アドレス範囲が RDAP をサポートするインターネットレジストリ (ARIN、RIPE、APNIC など) に登録されている場合、このページのプロセスを使用して X.509 証明書でドメインの制御を確認できます。証明書はプロビジョニングプロセスの間のみ有効である必要があります。プロビジョニングが完了したら、RIR の記録から証明書を削除できます。
- インターネットレジストリが RDAP をサポートしているかどうかにかかわらず、Amazon VPC IPAM を使用すると DNS TXT レコードでドメインの制御を確認できます。このプロセスについては「Amazon VPC IPAM ユーザーガイド」の「チュートリアル: IP アドレスを IPAM に移行する」を参照してください。

詳細についてはAWS オンラインテックトークの自分の IP アドレス使用の詳細を参照してください。

内容

- BYOIP の定義
- 要件とクォータ
- リージョン別の可用性
- Local Zone の可用性
- Amazon EC2 で BYOIP を使用するための前提条件
- Amazon EC2 で使用するアドレス範囲をオンボードする
- Amazon EC2 で BYOIP アドレス範囲を使用する

BYOIP の定義

- X.509 自己署名証明書 — ネットワーク内のデータを暗号化および認証するために最も一般的に使用される証明書標準。これはRDAP レコードからの IP スペースの制御を検証するために AWS によって使用される証明書です。X.509 証明書の詳細については「RFC 3280」を参照してください。

- AS 番号 (ASN) — 明確に定義された単一のルーティングポリシーを維持する 1 つ以上のネットワークオペレーターによって実行される IP プレフィックスのグループを定義するグローバル一意識別子。
- 地域インターネットレジストリ (RIR) — 世界のある地域内の IP アドレスと ASN の割り当てと登録を管理する組織。
- レジストリデータアクセスプロトコル (RDAP) — RIR 内の現在の登録データを照会する、読み取り専用プロトコルです。クエリされた RIR データベース内のエントリは「RDAP レコード」と呼ばれます。特定のレコードタイプはRIR が提供するメカニズムを使用して顧客により更新される必要があります。これらのレコードはRIR 内のアドレス空間の制御を確認するために AWS によりクエリされます。
- Route Origin Authorization (ROA) — お客様が特定の自律システムで IP アドバタイズメントを認証するために RIR によって作成されたオブジェクト。概要についてはARIN ウェブサイトの「Route Origin Authorizations (ROAs)」(ルートオリジン認証 (ROA)) を参照してください。
- ローカルインターネットレジストリ (LIR) — RIR からの IP アドレスのブロックをお客様に割り当てるインターネットサービスプロバイダーなどの組織。

要件とクォータ

- アドレス範囲は地域インターネットレジストリ (RIR) に登録する必要があります。地理的リージョンに関するポリシーについてはRIR を参照してください。BYOIP は現在、American Registry for Internet Numbers (ARIN)、Réseaux IP Européens Network Coordination Centre (RIPE) または Asia-Pacific Network Information Centre (APNIC) への登録をサポートしています。アドレス範囲は事業体または機関エンティティについて登録を受ける必要があり、個人については登録を受けられない場合があります。
- 取得できる最も具体的な IPv4 アドレス範囲は /24 です。
- 提供できる最も具体的な IPv6 アドレス範囲はパブリックにアドバタイズ可能な CIDR の場合は /48、パブリックにアドバタイズ可能でない CIDR の場合は /60 です。
- ROA はパブリックにアドバタイズ可能でない CIDR 範囲には必要ありませんが、RDAP レコードは更新する必要があります。
- 各アドレス範囲は一度に 1 つの AWS リージョンで使用できます。
- AWS リージョンごとに合計 5 つの BYOIP IPv4 および IPv6 アドレス範囲を AWS アカウントに取り込むことができます。Service Quotas コンソールを使用して BYOIP CIDR のクォータを調整することはできませんが、「AWS 全般のリファレンス」の「AWS サービスクォータ」で説明され

ているように、AWS サポートセンターに連絡してクォータの引き上げをリクエストすることはできます。

- Amazon VPC IP Address Manager (IPAM) を使用し、IPAM と AWS RAM Organizationsを統合しない限り、AWS を使用する他のアカウントとIPアドレス範囲を共有することはできません。詳細についてはAmazon VPC IP アドレス管理ユーザーガイドのAWS Organizations と IPAM を統合するを参照してください。
- IP アドレス範囲内のアドレスには消去履歴が含まれている必要があります。弊社はIP アドレス範囲に評価が低いまたは悪意のある挙動に関連付けられている IP アドレスが含まれている場合、当該範囲の評価を調査したり、当該範囲を拒否する権利を留保したりすることがあります。
- レガシーアドレススペース、つまり Regional Internet Registry (RIR) システムの形成前に Internet Assigned Numbers Authority's (IANA) の中央レジストリによって配布された IPv4 アドレススペースには引き続き対応する ROA オブジェクトが必要です。
- LIR では手動プロセスを使用してレコードを更新するのが一般的です。LIR によってはデプロイに数日かかることがあります。
- 大規模な CIDR ブロックには単一の ROA オブジェクトと RDP レコードが必要です。単一のオブジェクトとレコードを使用して、その範囲から AWS まで (複数の AWS リージョンにまたがることもできます) 複数の小さな CIDR ブロックを使用できます。
- BYOIP はWavelength Zones または AWS Outposts ではサポートされていません。
- RADb やその他の IRR の BYOIP を手動で変更しないでください。BYOIP は RADb を自動的に更新します。BYOIP ASN を含む手動変更を行うと、BYOIP プロビジョニング操作が失敗します。
- IPv4 アドレス範囲を AWS に設定すると、最初のアドレス (ネットワークアドレス) と最後のアドレス (ブロードキャストアドレス) を含む、範囲内のすべての IP アドレスを使用できます。

リージョン別の可用性

BYOIP 機能は現在、AWS中国リージョンを除くすべての商用リージョンで利用できます。

Local Zone の可用性

Local Zone は地理的にユーザーに近い場所に位置する AWS リージョンを拡張したものです。Local Zones は「ネットワークボーダーグループ」にグループ化されます。AWS で、ネットワークボーダーグループはAWS がパブリック IP アドレスをアドバタイズするアベイラビリティーゾーン (AZ)、Local Zones、Wavelength Zones のコレクションです。Local Zones はAWS ネットワークとこれらのゾーンのリソースにアクセスするお客様との間のレイテンシーや物理的距離を最小限に抑え

るために、AWS リージョン内の AZ とは異なるネットワークボーダーグループを持つ場合があります。

`--network-border-group` オプションを使用すると、以下の Local Zone ネットワークボーダーグループに BYOIPv4 アドレス範囲をプロビジョニングしてアドバタイズできます。

- af-south-1-los-1
- ap-northeast-1-tpe-1
- ap-south-1-ccu-1
- ap-south-1-del-1
- ap-southeast-1-bkk-1
- ap-southeast-1-mnl-1
- ap-southeast-2-akl-1
- ap-southeast-2-per-1
- eu-central-1-ham-1
- eu-central-1-ist-1
- eu-central-1-waw-1
- eu-north-1-cph-1
- eu-north-1-hel-1
- me-south-1-mct-1
- us-east-1-atl-2
- us-east-1-bos-1
- us-east-1-bue-1
- us-east-1-chi-2
- us-east-1-dfw-2
- us-east-1-iah-2
- us-east-1-lim-1
- us-east-1-mci-1
- us-east-1-mia-2
- us-east-1-msp-1
- us-east-1-nyc-1

- us-east-1-nyc-2
- us-east-1-phl-1
- us-east-1-qro-1
- us-east-1-scl-1
- us-west-2-den-1
- us-west-2-hnl-1
- us-west-2-las-1
- us-west-2-lax-1
- us-west-2-pdx-1
- us-west-2-phx-2
- us-west-2-sea-1

Local Zones を有効にしている場合 (「Enable a Local Zone」を参照)、BYOIPv4 CIDR のプロビジョニングとアドバタイズをするときにネットワークボーダーグループを選択できます。EIP とそれが関連付けられている AWS リソースは同じネットワークボーダーグループに属している必要があるため、ネットワークボーダーグループは慎重に選択してください。

> **Note**
>
> 現時点では、BYOIPv4 と BYOIPv6 の両方をサポートする `eu-central-1-ist-1` を除き、ローカルゾーンで BYOIPv6 アドレス範囲をプロビジョニングまたはアドバタイズすることはできません。

`eu-central-1-ist-1` ローカルゾーンでは、アドバタイズと取り消しは非同期オペレーションであり、完了までに最大 5 営業日かかる場合があります。アドレス範囲をアドバタイズすると、そのステータスはオペレーションの完了後に `provisioned` から `pending-advertising` に変わり、その後 `advertised` に変わります。取り消すと、ステータスが `advertised` から `pending-withdrawal` に変わり、`provisioned` に戻ります。AWS コンソールまたは CLI を使用して、オペレーションの進行状況をモニタリングできます。

独自の ASN (BYOASN) を使用する場合は、アドレス範囲をアドバタイズするときに指定できます。アドバタイズ後に CIDR に関連付けられた ASN を変更することはできません。ASN を変更するには、CIDR を取り消して再度アドバタイズします。

> **Important**
>
> 保留中のアドバタイズまたは取り消しをキャンセルすることはできません。次のオペレーションを開始する前に、オペレーションが完了するまで待つ必要があります。

Amazon EC2 で BYOIP を使用するための前提条件

BYOIP のオンボーディングプロセスには 2 つのフェーズがあり、そのためには 3 つのステップを実行する必要があります。これらの手順は次の図に示す手順に対応しています。このドキュメントには手動のステップが含まれていますが、RIR はこれらのステップをサポートするマネージドサービスを提供している場合があります。

> **Tip**
>
> このセクションのタスクには Linux ターミナルが必要で、Linux、AWS CloudShell、または Windows Subsystem for Linux を使用して実行できます。

内容

- 概要
- プライベートキーを作成し、X.509 証明書を生成します。
- X.509 証明書を RIR の RDAP レコードにアップロードする
- RIR に ROA オブジェクトを作成する

概要

準備フェーズ

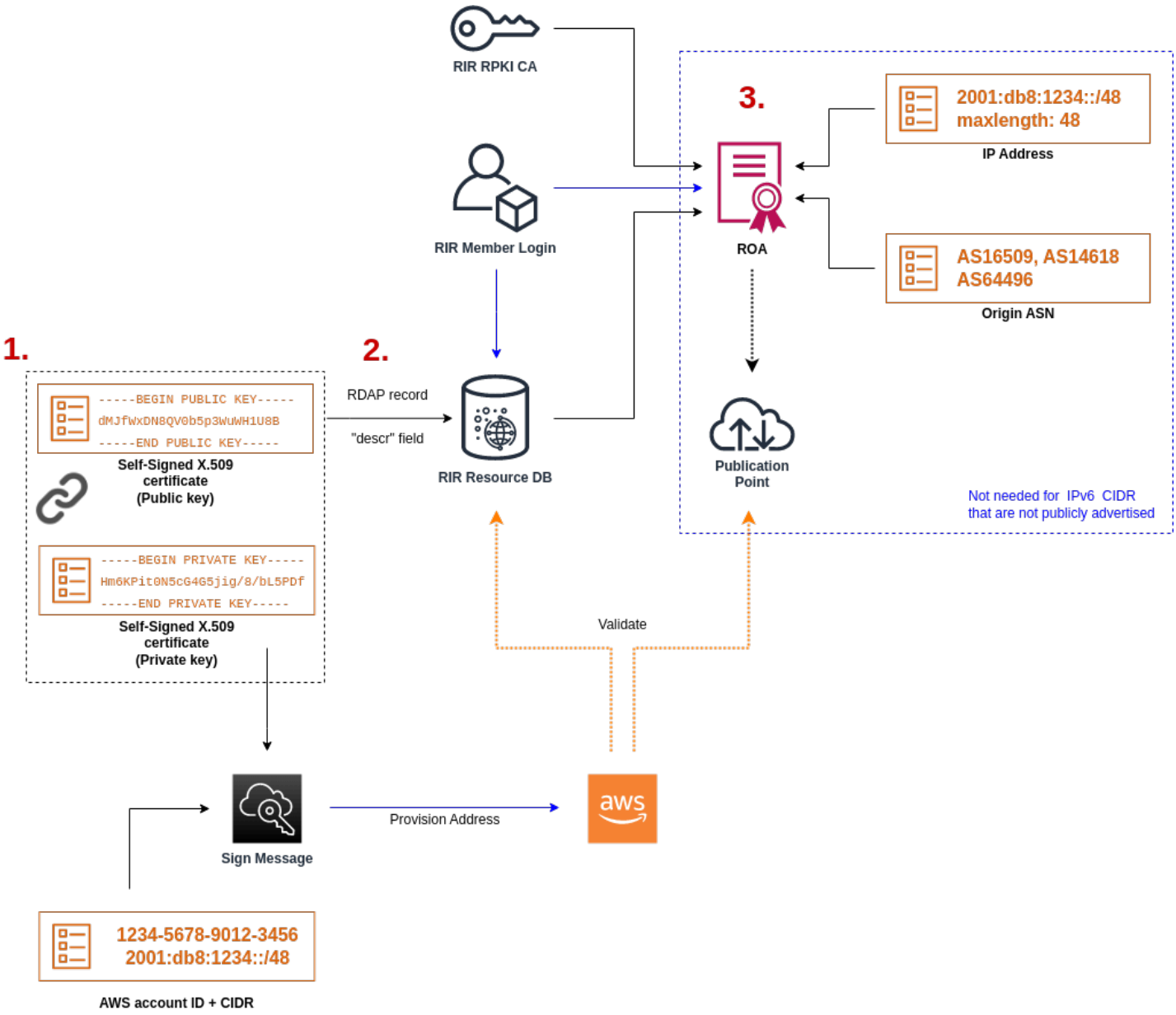
[1] プライベートキーを作成し、それを使用して認証のための自己署名 X.509 証明書を生成します。この証明書はプロビジョニング段階でのみ使用されます。プロビジョニングが完了したら、RIR の記録から証明書を削除できます。

RIR 設定フェーズ

[2] RDAP レコードのコメントに自己署名証明書をアップロードします。

[3] RIR に ROA オブジェクトを作成します。ROA は目的のアドレス範囲、アドレス範囲のアドバタイズを許可する自律システム番号 (ASN)、および RIR の Resource Public Key Infrastructure (RPKI) に登録する有効期限を定義します。

> **Note**
>
> パブリックにアドバタイズ可能でない IPv6 アドレス空間にはROA は必要ありません。



複数のアドレス範囲を使用する場合はそれぞれの不連続のアドレス範囲に対し、このプロセスを繰り返します。ただし、連続したブロックを複数の異なる AWS リージョンに分割する場合は準備の手順と RIR 設定手順を繰り返す必要はありません。

新しくアドレス範囲を追加しても、以前に追加済みのアドレス範囲には影響を与えません。

プライベートキーを作成し、X.509 証明書を生成します。

次の手順を使用して、自己署名 X509 証明書を作成し、RIR の RDAP レコードに追加します。RIR を使用してアドレス範囲を認証するにはこのキーペアを使用します。openssl コマンドには OpenSSL バージョン 1.0.2 以降が必要です。

次のコマンドをコピーし、プレースホルダー値 (色付きの斜体テキスト) のみを置換します。

この手順ではプライベート RSA キーを暗号化し、アクセスするためにパスフレーズを要求するベストプラクティスに従います。

1. 以下に示すように RSA 2048 ビットのプライベートキーを生成します。

```
$ openssl genpkey -aes256 -algorithm RSA -pkeyopt rsa_keygen_bits:2048 -out
 private-key.pem
```

-aes256 パラメータはプライベートキーの暗号化に使用されるアルゴリズムを指定します。コマンドの出力は次の通りです。これにはパスフレーズを設定するためのプロンプトが含まれます。

```
......+++
.+++
Enter PEM pass phrase: xxxxxxx
Verifying - Enter PEM pass phrase: xxxxxxx
```

次のコマンドを使用して、キーを検査します。

```
$ openssl pkey -in private-key.pem -text
```

これは次のようなパスフレーズプロンプトとキーの内容を返します。

```
Enter pass phrase for private-key.pem: xxxxxxx
-----BEGIN PRIVATE KEY-----
MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBKgwggSkAgEAAoIBAQDFBXHRI4HVKAhh
```

```
3seiciooizCRTbJe1+YsxNTja4XyKypVGIFWDGhZs44FCHlPOOSVJ+NqP74w96oM
7DPS3xo9kaQyZBFn2YEp2EBq5vf307KHNRmZZUmkn0zHOSEpNmY2fMxISBxewlxR
FAniwmSd/8TDvHJMY9FvAIvWuTsv5l0tJKk+a91K4+tO3UdDR7Sno5WXExfsBrW3
g1ydo3TBsx8i5/YiVOcNApy7ge2/FiwY3aCXJB6r6nuF6H8mRgI4r4vkMRsOlAhJ
DnZPNeweboo+K3Q3lwbgbmOKD/z9svk8N/+hUTBtIX0fRtbG+PLIw3xWRHGrMSn2
BzsPVuDLAgMBAAECggEACiJUj2hfJkKv47Dc3es3Zex67A5uDVjXmxfox2Xhdupn
fAcNqAptV6fXt0SPUNbhUxbBKNbshoJGufFwXPli1SXnpzvkdU4Hyco4zgbhXFsE
RNYjYfOGzTPwdBLpNMB6k3Tp4RHse6dNrlH0jDhpioL8cQEBdBJyVF5X0wymEbmV
mC0jgH/MxsBAPWW6ZKicg9ULMlWiAZ3MRAZPjHHgpYkAAsUWKAbCBwVQcVjGO59W
jfZjzTX5pQtVVH68ruciH88DTZCwjCkjBhxg+OIkJBLE5wkh82jIHSivZ63flwLw
z+E0+HhELSZJrn2MY6Jxmik3qNNUOF/Z+3msdj2luQKBgQDjwlC/3jxp8zJy6P8o
JQKv7TdvMwUj4VSWOHZBHLv4evJaaia0uQjIo1UDa8AYitqhX1NmCCehGH8yuXj/
v6V3CzMKDkmRr1NrONnSz5QsndQ04Z6ihAQlPmJ96g4wKtgoC7AYpyP0g1a+4/sj
b1+o3YQI4pD/F71c+qaztH7PRwKBgQDdc23yNmT3+Jyptf0fKjEvONK+xwUKzi9c
L/OzBq5yOIC1Pz2T85gOe1i8kwZws+xlpG6uBT6lmIJELd0k59FyupNu4dPvX5SD
6GGqdx4jk9KvI74usGeOBohmF0phTHkrWKBxXiyT0oS8zjnJlEn8ysIpGgO28jjr
LpaHNZ/MXQKBgQDfLNcnS0LzpsS2aK0tzyZU8SMyqVHOGMxj7quhneBq2T6FbiLD
T9TVlYaGNZ0j71vQaLI19qOubWymbautH0Op5KV8owdf4+bf1/NJaPIOzhDUSIjD
Qo01WW31Z9XDSRhKFTnWzmCjBdeIcajyzf10YKsycaAW9lItu8aBrMndnQKBgQDb
nNp/JyRwqjOrNljk7DHEs+SD39kHQzzCfqd+dnTPv2sc06+cpym3yulQcbokULpy
fmRo3bin/pvJQ3aZX/Bdh9woTXqhXDdrrSwWInVYMQPyPk8f/D9mIOJp5FUWMwHD
U+whIZSxsEeE+jtixlWtheKRYkQmzQZXbWdIhYyI3QKBgD+F/6wcZ85QW8nAUykA
3WrSIx/3cwDGdm4NRGct8ZOZjTHjiy9ojMOD1L7iMhRQ/3k3hUsin5LDMp/ryWGG
x4uIaLat40kiC7T4I66DM7P59euqdz3w0PD+VU+h7GSivvsFDdySUt7bNK0AUVLh
dMJfWxDN8QV0b5p3WuWH1U8B
-----END PRIVATE KEY-----
Private-Key: (2048 bit)
modulus:
    00:c5:05:71:d1:23:81:d5:28:08:61:de:c7:a2:72:
    2a:28:8b:30:91:4d:b2:5e:d7:e6:2c:c4:d4:e3:6b:
    85:f2:2b:2a:55:18:81:56:0c:68:59:b3:8e:05:08:
    79:4f:38:e4:95:27:e3:6a:3f:be:30:f7:aa:0c:ec:
    33:d2:df:1a:3d:91:a4:32:64:11:67:d9:81:29:d8:
    40:6a:e6:f7:f7:d3:b2:87:35:19:99:65:49:a4:9f:
    4c:c7:39:21:29:36:66:36:7c:cc:48:48:1c:5e:c2:
    5c:51:14:09:e2:c2:64:9d:ff:c4:c3:bc:72:4c:63:
    d1:6f:00:8b:d6:b9:3b:2f:e6:5d:2d:24:a9:3e:6b:
    dd:4a:e3:eb:4e:dd:47:43:47:b4:a7:a3:95:97:13:
    17:ec:06:b5:b7:83:5c:9d:a3:74:c1:b3:1f:22:e7:
    f6:22:54:e7:0d:02:9c:bb:81:ed:bf:16:2c:18:dd:
    a0:97:24:1e:ab:ea:7b:85:e8:7f:26:46:02:38:af:
    8b:e4:31:1b:0e:94:08:49:0e:76:4f:35:ec:1e:6e:
    8a:3e:2b:74:37:97:06:e0:6e:63:8a:0f:fc:fd:b2:
    f9:3c:37:ff:a1:51:30:6d:21:7d:1f:46:d6:c6:f8:
```

```
    f2:c8:c3:7c:56:44:71:ab:31:29:f6:07:3b:0f:56:
    e0:cb
publicExponent: 65537 (0x10001)
privateExponent:
    0a:22:54:8f:68:5f:26:42:af:e3:b0:dc:dd:eb:37:
    65:ec:7a:ec:0e:6e:0d:58:d7:9b:17:e8:c7:65:e1:
    76:ea:67:7c:07:0d:a8:0a:6d:57:a7:d7:b7:44:8f:
    50:d6:e1:53:16:c1:28:d6:ec:86:82:46:b9:f1:70:
    5c:f9:62:d5:25:e7:a7:3b:e4:75:4e:07:c9:ca:38:
    ce:06:e1:5c:5b:04:44:d6:23:61:f3:86:cd:33:f0:
    74:12:e9:34:c0:7a:93:74:e9:e1:11:ec:7b:a7:4d:
    ae:51:f4:8c:38:69:8a:82:fc:71:01:01:74:12:72:
    54:5e:57:d3:0c:a6:11:b9:95:98:2d:23:80:7f:cc:
    c6:c0:40:3d:65:ba:64:a8:9c:83:d5:0b:32:55:a2:
    01:9d:cc:44:06:4f:8c:71:e0:a5:89:00:02:c5:16:
    28:06:c2:07:05:50:71:58:c6:3b:9f:56:8d:f6:63:
    cd:35:f9:a5:0b:55:54:7e:bc:ae:e7:22:1f:cf:03:
    4d:90:b0:8c:29:23:06:1c:60:f8:e2:24:24:12:c4:
    e7:09:21:f3:68:c8:1d:28:af:67:ad:df:97:02:f0:
    cf:e1:34:f8:78:44:2d:26:49:ae:7d:8c:63:a2:71:
    9a:29:37:a8:d3:54:38:5f:d9:fb:79:ac:76:3d:a5:
    b9
prime1:
    00:e3:c2:50:bf:de:3c:69:f3:32:72:e8:ff:28:25:
    02:af:ed:37:6f:33:05:23:e1:54:96:38:76:41:1c:
    bb:f8:7a:f2:5a:6a:26:b4:b9:08:c8:a3:55:03:6b:
    c0:18:8a:da:a1:5f:53:66:08:27:a1:18:7f:32:b9:
    78:ff:bf:a5:77:0b:33:0a:0e:49:91:af:53:6b:38:
    d9:d2:cf:94:2c:9d:d4:34:e1:9e:a2:84:04:25:3e:
    62:7d:ea:0e:30:2a:d8:28:0b:b0:18:a7:23:f4:83:
    56:be:e3:fb:23:6f:5f:a8:dd:84:08:e2:90:ff:17:
    bd:5c:fa:a6:b3:b4:7e:cf:47
prime2:
    00:dd:73:6d:f2:36:64:f7:f8:9c:a9:b5:fd:1f:2a:
    31:2f:38:d2:be:c7:05:0a:ce:2f:5c:2f:f3:b3:06:
    ae:72:38:80:b5:3f:3d:93:f3:98:0e:7b:58:bc:93:
    06:70:b3:ec:65:a4:6e:ae:05:3e:a5:98:82:44:2d:
    dd:24:e7:d1:72:ba:93:6e:e1:d3:ef:5f:94:83:e8:
    61:aa:77:1e:23:93:d2:af:23:be:2e:b0:67:8e:06:
    88:66:17:4a:61:4c:79:2b:58:a0:71:5e:2c:93:d2:
    84:bc:ce:39:c9:94:49:fc:ca:c2:29:1a:03:b6:f2:
    38:eb:2e:96:87:35:9f:cc:5d
exponent1:
    00:df:2c:d7:27:4b:42:f3:a6:c4:b6:68:ad:2d:cf:
```

```
    26:54:f1:23:32:a9:51:ce:18:cc:63:ee:ab:a1:9d:
    e0:6a:d9:3e:85:6e:22:c3:4f:d4:d5:95:86:86:35:
    9d:23:ef:5b:d0:68:b2:35:f6:a3:ae:6d:6c:a6:6d:
    ab:ad:1f:43:a9:e4:a5:7c:a3:07:5f:e3:e6:df:d7:
    f3:49:68:f2:0e:ce:10:d4:48:88:c3:42:8d:35:59:
    6d:f5:67:d5:c3:49:18:4a:15:39:d6:ce:60:a3:05:
    d7:88:71:a8:f2:cd:fd:74:60:ab:32:71:a0:16:f6:
    52:2d:bb:c6:81:ac:c9:dd:9d
exponent2:
    00:db:9c:da:7f:27:24:70:aa:33:ab:36:58:e4:ec:
    31:c4:b3:e4:83:df:d9:07:43:3c:c2:7e:a7:7e:76:
    74:cf:bf:6b:1c:d3:af:9c:a7:29:b7:ca:e9:50:71:
    ba:24:50:ba:72:7e:64:68:dd:b8:a7:fe:9b:c9:43:
    76:99:5f:f0:5d:87:dc:28:4d:7a:a1:5c:37:6b:ad:
    2c:16:22:75:58:31:03:f2:3e:4f:1f:fc:3f:66:20:
    e2:69:e4:55:16:33:01:c3:53:ec:21:21:94:b1:b0:
    47:84:fa:3b:62:c6:55:ad:85:e2:91:62:44:26:cd:
    06:57:6d:67:48:85:8c:88:dd
coefficient:
    3f:85:ff:ac:1c:67:ce:50:5b:c9:c0:53:29:00:dd:
    6a:d2:23:1f:f7:73:00:c6:76:6e:0d:44:67:2d:f1:
    93:99:8d:31:e3:8b:2f:68:8c:c3:83:d4:be:e2:32:
    14:50:ff:79:37:85:4b:22:9f:92:c3:32:9f:eb:c9:
    61:86:c7:8b:88:68:b6:ad:e3:49:22:0b:b4:f8:23:
    ae:83:33:b3:f9:f5:eb:aa:77:3d:f0:d0:f0:fe:55:
    4f:a1:ec:64:a2:be:fb:05:0d:dc:92:52:de:db:34:
    ad:00:51:52:e1:74:c2:5f:5b:10:cd:f1:05:74:6f:
    9a:77:5a:e5:87:d5:4f:01
```

プライベートキーは使用しないときは安全な場所に保管してください。

2. 以前のステップで作成したプライベートキーを使用して、X.509 証明書を生成します。この例では証明書は 365 日で期限切れになり、それ以降は信頼されません。有効期限は適切に設定してください。証明書はプロビジョニングプロセスの間のみ有効である必要があります。プロビジョニングが完了したら、RIR の記録から証明書を削除できます。`tr -d "\n"` コマンドは出力から改行文字 (改行) を削除します。プロンプトが表示されたら、共通名を指定する必要がありますが、その他のフィールドは空白のままにしておくことができます。

```
$ openssl req -new -x509 -key private-key.pem -days 365 | tr -d "\n" >
 certificate.pem
```

この結果、次のような出力が得られます。

```
Enter pass phrase for private-key.pem: xxxxxxx
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) []:
State or Province Name (full name) []:
Locality Name (eg, city) []:
Organization Name (eg, company) []:
Organizational Unit Name (eg, section) []:
Common Name (eg, fully qualified host name) []:example.com
Email Address []:
```

> **Note**
>
> AWS プロビジョニングには共通名は必要ありません。内部ドメイン名またはパブリックドメイン名は任意です。

次のコマンドを使用して、証明書を取得できます。

```
$ cat certificate.pem
```

出力は改行のない長い PEM エンコード文字列で、先頭が `-----BEGIN CERTIFICATE-----` で、その後に `-----END CERTIFICATE-----` が続きます。

X.509 証明書を RIR の RDAP レコードにアップロードする

以前に作成した証明書を、RIR の RDAP レコードに追加します。エンコードされた部分の前後の `-----BEGIN CERTIFICATE-----` および `-----END CERTIFICATE-----` 文字列を、必ず含めます。このコンテンツはすべて、長い 1 行にする必要があります。RDAP を更新する手順はご使用の RIR によって異なります。

- ARIN の場合はAccount Manager ポータルを使用して、アドレス範囲を表す「ネットワーク情報」オブジェクトの「パブリックコメント」セクションに証明書を追加してください。組織の [comments] セクションには追加しないでください。
- RIPE の場合は証明書を新しい「descr」フィールドとして、アドレス範囲を表す「inetnum」または「inet6num」オブジェクトに追加します。これらは通常、RIPE Database ポータルの「マイリソース」セクションにあります。組織の [コメント] セクションや上記オブジェクトの「備考」フィールドには追加しないでください。
- APNIC の場合、「inetnum」または「inet6num」レコードに備考を編集して証明書を追加します。

以下のプロビジョニング段階が完了したら、RIR の記録から証明書を削除できます。

RIR に ROA オブジェクトを作成する

アドレス範囲をアドバタイズするために Amazon ASN 16509 および 14618 を承認し、また、アドレス範囲をアドバタイズすることが現在許可されている ASN も承認するために、ROA オブジェクトを作成します。AWS GovCloud (US) Regions については16509 および 14618 ではなく ASN 8987 を承認してください。持ち込む CIDR のサイズに最大長を設定する必要があります。持ち込める最も具体的な IPv4 プレフィクスは /24 です。提供できる最も具体的な IPv6 アドレス範囲はパブリックにアドバタイズ可能な CIDR の場合は /48、パブリックにアドバタイズ可能でない CIDR の場合は /60 です。

> **Important**
>
> Amazon VPC IP Address Manager (IPAM) 用の ROA オブジェクトを作成する場合、ROA を作成するときにはIPv4 CIDR に対して、/24 のIP アドレスのプレフィックスの最大長を設定する必要があります。IPv6 CIDR についてはアドバタイズ可能なプールに追加する場合、IP アドレスのプレフィックスの最大長は /48 である必要があります。これにより、パブリック IP アドレスを AWS リージョンごとに分割して利用する柔軟性がもたらされます。IPAM では設定した最大長が適用されます。IPAM への BYOIP アドレスの詳細については、「Amazon VPC IPAM ユーザーガイド」の「チュートリアル: IPAM への BYOIP アドレス CIDR」を参照してください。

ROA が Amazon で使用できるようになるまで最大 24 時間かかる場合があります。詳細についてはRIRを参照してください。

- ARIN — ROA のリクエスト数

- RIPE — ROA の管理
- APNIC— 経路管理

アドバタイズメントをオンプレミスのワークロードから AWS に移行する場合はAmazon の ASN の ROA を作成する前に、既存の ASN 向けの ROA を作成する必要があります。これを行わないと、既存のルーティングとアドバタイズメントに影響を与える可能性があります。

> **Important**
>
> Amazon がお客様の IP アドレス範囲をアドバタイズし、引き続きアドバタイズするには Amazon ASN の ROA が上記のガイドラインに準拠している必要があります。お客様の ROA が無効であるか、上記のガイドラインに準拠していない場合、Amazon はお客様の IP アドレス範囲のアドバタイズを停止する権利を留保します。

> **Note**
>
> このステップはパブリックにアドバタイズ可能でない IPv6 アドレス空間には必要ありません。

Amazon EC2 で使用するアドレス範囲をオンボードする

BYOIP のオンボーディングプロセスにはニーズに応じて次のタスクが含まれます。

タスク

- AWS でパブリックにアドバタイズ可能なアドレス範囲をプロビジョニングする
- パブリックにアドバタイズ可能でない IPv6 アドレス範囲をプロビジョニングする
- AWS を通じてアドレス範囲をアドバタイズする
- アドレス範囲のプロビジョニング解除
- BYOIP を検証する

AWS でパブリックにアドバタイズ可能なアドレス範囲をプロビジョニングする

AWS で使用するアドレス範囲をプロビジョニングする場合は当該範囲の管理者であることを証明し、Amazon による当該範囲のアドバタイズを承認します。また、署名済みの認可メッセージを使

用して、アドレス範囲を管理していることを確認します。このメッセージはX.509 証明書で RDAP レコードを更新するときに使用した自己署名 X.509 キーペアで署名されます。AWS にはRIR に提示する暗号署名付き認可メッセージが必要です。RIR はRDAP に追加した証明書に対して署名を認証し、ROA に対して認可の詳細をチェックします。

アドレス範囲をプロビジョニングするには

1. メッセージを構成する

   プレーンテキスト認可メッセージを作成します。メッセージの形式は以下のとおりです。日付はメッセージの有効期限日になります。

   ```
   1|aws|account|cidr|YYYYMMDD|SHA256|RSAPSS
   ```

   アカウント番号、アドレス範囲、および有効期限日を独自の値に置き換え、次のようなメッセージを作成します。

   ```
   text_message="1|aws|0123456789AB|198.51.100.0/24|20211231|SHA256|RSAPSS"
   ```

   これは ROA メッセージと外観が似ているので、混同しないでください。

2. メッセージに署名する

   以前に作成したプライベートキーを使用して、プレーンテキストメッセージに署名します。このコマンドが返す署名は長い文字列となります。また、次の手順で使用する必要があります。

   > **Important**
   >
   > このコマンドをコピーして貼り付けることをお勧めします。メッセージの内容を除いて、いずれの値も変更または置換しないでください。

   ```
   signed_message=$( echo -n $text_message | openssl dgst -sha256 -sigopt
    rsa_padding_mode:pss -sigopt rsa_pss_saltlen:-1 -sign private-key.pem -keyform PEM
    | openssl base64 | tr -- '+=/' '-_~' | tr -d "\n")
   ```

3. アドレスのプロビジョニング

   AWS CLIprovision-byoip-cidr コマンドを使用して、アドレス範囲をプロビジョニングします。--cidr-authorization-context オプションは以前に作成したメッセージと署名の文字列を使用します。

   > **Important**
   >
   > 「Configure the AWS CLI」で設定した Default region name とは異なるリージョンに BYOIP 範囲をプロビジョニングする場合はプロビジョニング先の AWS リージョンを指定する必要があります。

   ```
   aws ec2 provision-byoip-cidr --cidr address-range --cidr-authorization-context
    Message="$text_message",Signature="$signed_message" --region us-east-1
   ```

   アドレス範囲のプロビジョニングは非同期オペレーションであるため、呼び出しはすぐに戻りますが、アドレスの範囲はそのステータスが pending-provision から provisioned に変わるまで使用できません。

4. 進行状況をモニタリングする

   ほとんどのプロビジョニングは 2 時間以内に完了しますが、パブリックにアドバタイズ可能な範囲のプロビジョニングプロセスが完了するまでに最大 1 週間かかる場合があります。この例のように、describe-byoip-cidrs コマンドを使用して進行状況をモニタリングします。

   ```
   aws ec2 describe-byoip-cidrs --max-results 5 --region us-east-1
   ```

   プロビジョニング中に問題が発生してステータスが failed-provision になった場合は問題の解決後に provision-byoip-cidr コマンドを再度実行する必要があります。

パブリックにアドバタイズ可能でない IPv6 アドレス範囲をプロビジョニングする

デフォルトではアドレス範囲はインターネットにパブリックにアドバタイズ可能になるようにプロビジョニングされます。パブリックにアドバタイズ可能でない IPv6 アドレス範囲をプロビジョニングできます。パブリックにアドバタイズできないルートの場合、プロビジョニングプロセスは通常、数分以内に完了します。非パブリックアドレス範囲の IPv6 CIDR ブロックを VPC に関連付ける場

合、IPv6 CIDR にはDirect Connect、AWS Site-to-Site VPN、Amazon VPC トランジットゲートウェイなどの IPv6 をサポートするハイブリッド接続オプションを介してのみアクセスできます。

非パブリックアドレス範囲をプロビジョニングする場合、ROA は必要ありません。

> **Important**
>
> - ユーザーはプロビジョニング中にアドレス範囲がパブリックにアドバタイズ可能かどうかのみ指定できます。アドバタイズ可能なステータスは後で変更できません。
> - Amazon VPC はユニークローカルアドレス (ULA) CIDR をサポートしていません。すべての VPC には一意の IPv6 CIDR が必要です。2 つの VPC が同じ IPv6 CIDR 範囲を持つことはできません。

パブリックにアドバタイズ可能でない IPv6 アドレス範囲をプロビジョニングするには次の provision-byoip-cidr コマンドを使用します。

```
aws ec2 provision-byoip-cidr --cidr address-range --cidr-authorization-context
 Message="$text_message",Signature="$signed_message" --no-publicly-advertisable --
region us-east-1
```

AWS を通じてアドレス範囲をアドバタイズする

アドレス範囲をプロビジョニングすると、公開することができるようになります。プロビジョンした正確なアドレス範囲をアドバタイズする必要があります。プロビジョンしたアドレス範囲の一部のみアドバタイズすることはできません。

パブリックにアドバタイズされない IPv6 アドレス範囲をプロビジョニングした場合、このステップを実行する必要はありません。

AWS からアドバタイズする前に、アドレス範囲またはその範囲の一部を他の場所からアドバタイズするのを停止することをお勧めします。他の場所から IP アドレス範囲またはその一部をアドバタイズし続ける場合は当社でその IP アドレス範囲を高い信頼性でサポートしたり、問題をトラブルシューティングすることができなくなります。具体的にはそのアドレス範囲またはその範囲の一部へのトラフィックが当社のネットワークに入るのを保証できません。

ダウンタイムを最小限に抑えるにはアドレス範囲がアドバタイズされる前にご使用のアドレスプールからアドレスを使用するように AWS リソースを設定してから、同時に現在の場所からのアドバタイ

ズを停止して、AWS からのアドバタイズを開始します。アドレスプールからの Elastic IP アドレスの割り当ての詳細についてはElastic IP アドレスを割り当てるを参照してください。

制限事項

- アドレス範囲が毎回異なる場合でも、advertise-byoip-cidr コマンドは 10 秒ごとに最大 1 回しか実行できません。
- アドレス範囲が毎回異なる場合でも、withdraw-byoip-cidr コマンドは 10 秒ごとに最大 1 回しか実行できません。

アドレス範囲を公開するには以下のadvertise-byoip-cidrコマンドを使用します。

```
aws ec2 advertise-byoip-cidr --cidr address-range --region us-east-1
```

アドレス範囲の公開を停止するには以下のwithdraw-byoip-cidrコマンドを使用します。

```
aws ec2 withdraw-byoip-cidr --cidr address-range --region us-east-1
```

> **Note**
>
> eu-central-1-ist-1 ローカルゾーンでは、アドバタイズと取り消しは非同期です。アドレス範囲は、オペレーションが完了する前に pending-advertising または pending-withdrawal の状態になります。詳細については、「ローカルゾーンのアベイラビリティー」を参照してください。

アドレス範囲のプロビジョニング解除

AWS でアドレス範囲の使用を停止するにはまず Elastic IP アドレスをリリースし、アドレスプールからまだ割り当てられている IPv6 CIDR ブロックの関連付けを解除します。次に、アドレス範囲のアドバタイズを停止し、最後にアドレス範囲のプロビジョニングを解除します。

アドレス範囲のプロビジョニングを部分的に解除することはできません。AWS でより具体的なアドレス範囲を使用する場合はアドレス範囲全体のプロビジョニングを解除し、より具体的なアドレス範囲をプロビジョニングします。

(IPv4) 各 Elastic IP アドレスをリリースするには以下の release-address コマンドを使用します。

```
aws ec2 release-address --allocation-id eipalloc-12345678abcabcabc --region us-east-1
```

(IPv6) IPv6 CIDR ブロックの関連付けを解除するには次の disassociate-vpc-cidr-block コマンドを使用します。

```
aws ec2 disassociate-vpc-cidr-block --association-id vpc-cidr-assoc-12345abcd1234abc1
 --region us-east-1
```

アドレス範囲の公開を停止するには以下のwithdraw-byoip-cidrコマンドを使用します。

```
aws ec2 withdraw-byoip-cidr --cidr address-range --region us-east-1
```

アドレス範囲のプロビジョニングを解除するには以下のdeprovision-byoip-cidrコマンドを使用します。

```
aws ec2 deprovision-byoip-cidr --cidr address-range --region us-east-1
```

アドレス範囲のプロビジョニングを解除するには最長 1 日かかります。

BYOIP を検証する

1. 自己署名 x.509 キーペアを検証する

   whois コマンドで、証明書がアップロードされており、かつ、有効であることを確認します。

   ARIN の場合、whois -h whois.arin.net r + *2001:0DB8:6172::/48* を使用してアドレス範囲の RDAP レコードを検索します。コマンド出力の NetRange (ネットワーク範囲) については「Public Comments」セクションを確認してください。証明書はアドレス範囲の「Public Comments」セクションに追加する必要があります。

   次のコマンドを使用して、証明書を含む Public Comments を検査できます。

   ```
   whois -h whois.arin.net r + 2001:0DB8:6172::/48 | grep Comments | grep BEGIN
   ```

   これにより、キーの内容を含む出力が返されます。これは次のようになります。

   ```
   Public Comments:
   -----BEGIN CERTIFICATE-----
   MIID1zCCAr+gAwIBAgIUBkRPNSLrPqbRAFP8RDAHSP+I1TowDQYJKoZIhvcNAQE
   ```

```
LBQAwezELMAkGA1UEBhMCTloxETAPBgNVBAgMCEF1Y2tsYW5kMREwDwYDVQQHDA
hBdWNrbGFuZDEcMBoGA1UECgwTQW1hem9uIFdlYiBTZXJ2aWNlczETMBEGA1UEC
wwKQllPSVAgRGVtbzETMBEGA1UEAwwKQllPSVAgRGVtbzAeFw0yMTEyMDcyMDI0
NTRaFw0yMjEyMDcyMDI0NTRaMHsxCzAJBgNVBAYTAk5aMREwDwYDVQQIDAhBdWN
rbGFuZDERMA8GA1UEBwwIQXVja2xhbmQxHDAaBgNVBAoME0FtYXpvbiBXZWIgU2
VydmljZXMxEzARBgNVBAsMCkJZT0lQIERlbW8xEzARBgNVBAMMCkJZT0lQIERlb
W8wggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCfmacvDp0wZ0ceiXXc
R/q27mHI/U5HKt7SST4X2eAqufR9wXkfNanAEskgAseyFypwEEQr4CJijI/5hp9
prh+jsWHWwkFRoBRR9FBtwcU/45XDXLga7D3stsI5QesHVRwOaXUdprAnndaTug
mDPkD0vrl475JWDSIm+PUxGWLy+60aBqiaZq35wU/x+wXlAqBXg4MZK2KoUu27k
Yt2zhmy0S7Ky+oRfRJ9QbAiSu/RwhQbh5Mkp1ZnVIc7NqnhdeIW48QaYjhMlUEf
xdaqYUinzz8KpjfADZ4Hvqj9jWZ/eXo/9b2rGlHWkJsbhr0VEUyAGu1bwkgcdww
3A7NjOxQbAgMBAAGjUzBRMB0GA1UdDgQWBBStFyujN6SYBr2glHpGt0XGF7GbGT
AfBgNVHSMEGDAWgBStFyujN6SYBr2glHpGt0XGF7GbGTAPBgNVHRMBAf8EBTADA
QH/MA0GCSqGSIb3DQEBCwUAA4IBAQBX6nn6YLhz521lfyVfxY0t6o3410bQAeAF
08ud+ICtmQ4IO4A4B7zV3zIVYr0clrOOaFyLxngwMYN0XY5tVhDQqk4/gmDNEKS
Zy2QkX4Eg0YUWVzOyt6fPzjOvJLcsqc1hcF9wySL507XQz76Uk5cFypBOzbnk35
UkWrzA9KK97cXckfIESgK/k1N4ecwxwG6VQ8mBGqVpPpey+dXpzzzv1iBKN/VY4
ydjgH/LBfdTsVarmmy2vtWBxwrqkFvpdhSGCvRDl/qdO/GIDJi77dmZWkh/ic90
MNk1f38gs1jrCj8lThoar17Uo9y/Q5qJIsoNPyQrJRzqFU9F3FBjiPJF
-----END CERTIFICATE-----
```

RIPE の場合、whois -r -h whois.ripe.net *2001:0DB8:7269::/48* を使用してアドレス範囲の RDAP レコードを検索します。コマンド出力の inetnum オブジェクト (ネットワーク範囲) については「descr」セクションを確認してください。証明書はアドレス範囲の新しい descr フィールドとして追加する必要があります。

次のコマンドを使用して、証明書を含む descr を検査できます。

```
whois -r -h whois.ripe.net 2001:0DB8:7269::/48 | grep descr | grep BEGIN
```

これにより、キーの内容を含む出力が返されます。これは次のようになります。

```
descr:
-----BEGIN CERTIFICATE-----MIID1zCCAr+gAwIBAgIUBkRPNSLrPqbRAFP8
RDAHSP+I1TowDQYJKoZIhvcNAQELBQAwezELMAkGA1UEBhMCTloxETAPBgNVBAg
MCEF1Y2tsYW5kMREwDwYDVQQHDAhBdWNrbGFuZDEcMBoGA1UECgwTQW1hem9uIF
dlYiBTZXJ2aWNlczETMBEGA1UECwwKQllPSVAgRGVtbzETMBEGA1UEAwwKQllPS
VAgRGVtbzAeFw0yMTEyMDcyMDI0NTRaFw0yMjEyMDcyMDI0NTRaMHsxCzAJBgNV
BAYTAk5aMREwDwYDVQQIDAhBdWNrbGFuZDERMA8GA1UEBwwIQXVja2xhbmQxHDA
aBgNVBAoME0FtYXpvbiBXZWIgU2VydmljZXMxEzARBgNVBAsMCkJZT0lQIERlbW
8xEzARBgNVBAMMCkJZT0lQIERlbW8wggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwg
```

```
gEKAoIBAQCfmacvDp0wZ0ceiXXcR/q27mHI/U5HKt7SST4X2eAqufR9wXkfNanA
EskgAseyFypwEEQr4CJijI/5hp9prh+jsWHWwkFRoBRR9FBtwcU/45XDXLga7D3
stsI5QesHVRwOaXUdprAnndaTugmDPkD0vrl475JWDSIm+PUxGWLy+60aBqiaZq
35wU/x+wXlAqBXg4MZK2KoUu27kYt2zhmy0S7Ky+oRfRJ9QbAiSu/RwhQbh5Mkp
1ZnVIc7NqnhdeIW48QaYjhMlUEfxdaqYUinzz8KpjfADZ4Hvqj9jWZ/eXo/9b2r
GlHWkJsbhr0VEUyAGu1bwkgcdww3A7NjOxQbAgMBAAGjUzBRMB0GA1UdDgQWBBS
tFyujN6SYBr2glHpGt0XGF7GbGTAfBgNVHSMEGDAWgBStFyujN6SYBr2glHpGt0
XGF7GbGTAPBgNVHRMBAf8EBTADAQH/MA0GCSqGSIb3DQEBCwUAA4IBAQBX6nn6Y
Lhz521lfyVfxY0t6o3410bQAeAF08ud+ICtmQ4IO4A4B7zV3zIVYr0clrOOaFyL
xngwMYN0XY5tVhDQqk4/gmDNEKSZy2QkX4Eg0YUWVzOyt6fPzjOvJLcsqc1hcF9
wySL507XQz76Uk5cFypBOzbnk35UkWrzA9KK97cXckfIESgK/k1N4ecwxwG6VQ8
mBGqVpPpey+dXpzzzv1iBKN/VY4ydjgH/LBfdTsVarmmy2vtWBxwrqkFvpdhSGC
vRDl/qdO/GIDJi77dmZWkh/ic90MNk1f38gs1jrCj8lThoar17Uo9y/Q5qJIsoN
PyQrJRzqFU9F3FBjiPJF
-----END CERTIFICATE-----
```

APNIC の場合、whois -h whois.apnic.net *2001:0DB8:6170::/48* を使用して BYOIP アドレス範囲の RDAP レコードを検索します。コマンド出力の inetnum オブジェクト (ネットワーク範囲) については「remarks」セクションを確認してください。証明書はアドレス範囲の新しい remarks フィールドとして追加する必要があります。

次のコマンドを使用して、証明書を含む remarks を検査できます。

```
whois -h whois.apnic.net 2001:0DB8:6170::/48 | grep remarks | grep BEGIN
```

これにより、キーの内容を含む出力が返されます。これは次のようになります。

```
remarks:
-----BEGIN CERTIFICATE-----
MIID1zCCAr+gAwIBAgIUBkRPNSLrPqbRAFP8RDAHSP+I1TowDQYJKoZIhvcNAQE
LBQAwezELMAkGA1UEBhMCTloxETAPBgNVBAgMCEF1Y2tsYW5kMREwDwYDVQQHDA
hBdWNrbGFuZDEcMBoGA1UECgwTQW1hem9uIFdlYiBTZXJ2aWNlczETMBEGA1UEC
wwKQllPSVAgRGVtbzETMBEGA1UEAwwKQllPSVAgRGVtbzAeFw0yMTEyMDcyMDI0
NTRaFw0yMjEyMDcyMDI0NTRaMHsxCzAJBgNVBAYTAk5aMREwDwYDVQQIDAhBdWN
rbGFuZDERMA8GA1UEBwwIQXVja2xhbmQxHDAaBgNVBAoME0FtYXpvbiBXZWIgU2
VydmljZXMxEzARBgNVBAsMCkJZT0lQIERlbW8xEzARBgNVBAMMCkJZT0lQIERlb
W8wggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCfmacvDp0wZ0ceiXXc
R/q27mHI/U5HKt7SST4X2eAqufR9wXkfNanAEskgAseyFypwEEQr4CJijI/5hp9
prh+jsWHWwkFRoBRR9FBtwcU/45XDXLga7D3stsI5QesHVRwOaXUdprAnndaTug
mDPkD0vrl475JWDSIm+PUxGWLy+60aBqiaZq35wU/x+wXlAqBXg4MZK2KoUu27k
Yt2zhmy0S7Ky+oRfRJ9QbAiSu/RwhQbh5Mkp1ZnVIc7NqnhdeIW48QaYjhMlUEf
xdaqYUinzz8KpjfADZ4Hvqj9jWZ/eXo/9b2rGlHWkJsbhr0VEUyAGu1bwkgcdww
```

```
3A7NjOxQbAgMBAAGjUzBRMB0GA1UdDgQWBBStFyujN6SYBr2glHpGt0XGF7GbGT
AfBgNVHSMEGDAWgBStFyujN6SYBr2glHpGt0XGF7GbGTAPBgNVHRMBAf8EBTADA
QH/MA0GCSqGSIb3DQEBCwUAA4IBAQBX6nn6YLhz521lfyVfxY0t6o3410bQAeAF
08ud+ICtmQ4IO4A4B7zV3zIVYr0clrOOaFyLxngwMYN0XY5tVhDQqk4/gmDNEKS
Zy2QkX4Eg0YUWVzOyt6fPzjOvJLcsqc1hcF9wySL507XQz76Uk5cFypBOzbnk35
UkWrzA9KK97cXckfIESgK/k1N4ecwxwG6VQ8mBGqVpPpey+dXpzzzv1iBKN/VY4
ydjgH/LBfdTsVarmmy2vtWBxwrqkFvpdhSGCvRDl/qdO/GIDJi77dmZWkh/ic90
MNk1f38gs1jrCj8lThoar17Uo9y/Q5qJIsoNPyQrJRzqFU9F3FBjiPJF
-----END CERTIFICATE-----
```

2. ROA オブジェクトの作成を検証する

RIPEstat データ API コマンドを使用して、ROA オブジェクトが正常に作成されたことを検証します。Amazon ASN 16509 および 14618、ならびにそのアドレス範囲をアドバタイズすることが現在承認されている ASN に対して、アドレス範囲をテストしてください。

次のコマンドを使用して、アドレス範囲で異なる Amazon ASN の ROA オブジェクトを検査できます。

```
curl --location --request GET "https://stat.ripe.net/data/rpki-validation/data.json?
resource=ASN&prefix=CIDR
```

この出力例ではレスポンスには、Amazon ASN 16509 について "status": "valid" の結果があります。これはアドレス範囲についての ROA オブジェクトが正常に作成されたことを示します。

```
{
    "messages": [],
    "see_also": [],
    "version": "0.3",
    "data_call_name": "rpki-validation",
    "data_call_status": "supported",
    "cached": false,
    "data": {
        "validating_roas": [
            {
                "origin": "16509",
                "prefix": "2001:0DB8::/32",
                "max_length": 48,
                "validity": "valid"
            },
            {
```

```
                "origin": "14618",
                "prefix": "2001:0DB8::/32",
                "max_length": 48,
                "validity": "invalid_asn"
            },
            {
                "origin": "64496",
                "prefix": "2001:0DB8::/32",
                "max_length": 48,
                "validity": "invalid_asn"
            }
        ],
        "status": "valid",
        "validator": "routinator",
        "resource": "16509",
        "prefix": "2001:0DB8::/32"
    },
    "query_id": "20230224152430-81e6384e-21ba-4a86-852a-31850787105f",
    "process_time": 58,
    "server_id": "app116",
    "build_version": "live.2023.2.1.142",
    "status": "ok",
    "status_code": 200,
    "time": "2023-02-24T15:24:30.773654"
}
```


“unknown” の状態はアドレス範囲についての ROA オブジェクトが作成されていないことを示します。“invalid_asn” の状態はアドレス範囲についての ROA オブジェクトが正常に作成されなかったことを示します。

Amazon EC2 で BYOIP アドレス範囲を使用する

ユーザーはアカウントでプロビジョニングした IPv4 と IPv6 のアドレス範囲の表示と使用が可能です。詳細については「the section called “アドレス範囲をオンボードする”」を参照してください。

IPv4 アドレス範囲

IPv4 アドレスプールから Elastic IP アドレスを作成し、EC2 インスタンス、NAT ゲートウェイ、Network Load Balancer などの AWS リソースで使用できます。

アカウントでプロビジョニングした IPv4 アドレスプールに関する情報を表示するには次の describe-public-ipv4-pools コマンドを使用します。

```
aws ec2 describe-public-ipv4-pools --region us-east-1
```

IPv4 アドレスプールから Elastic IP アドレスを作成するにはallocate-address コマンドを使用します。--public-ipv4-poolオプションを使用して、describe-byoip-cidrsが返すアドレスプールの ID を指定したり、--address オプションを使用して、プロビジョニングしたアドレス範囲からのアドレスを指定したりすることができます。

IPv6 アドレス範囲

アカウントでプロビジョニングした IPv6 アドレスプールに関する情報を表示するには次の describe-ipv6-pools コマンドを使用します。

```
aws ec2 describe-ipv6-pools --region us-east-1
```

VPC を作成し、IPv6 アドレスプールから IPv6 CIDR を指定するには次の create-vpc コマンドを使用します。Amazon が IPv6 アドレスプールから IPv6 CIDR を選択できるようにするには[--ipv6-cidr-block] オプションを省略します。

```
aws ec2 create-vpc --cidr-block 10.0.0.0/16 --ipv6-cidr-block ipv6-cidr --ipv6-
pool pool-id --region us-east-1
```

IPv6 アドレスプールからの IPv6 CIDR ブロックを VPC に関連付けるには次の associate-vpc-cidr-block コマンドを使用します。Amazon が IPv6 アドレスプールから IPv6 CIDR を選択できるようにするには[--ipv6-cidr-block] オプションを省略します。

```
aws ec2 associate-vpc-cidr-block --vpc-id vpc-123456789abc123ab --ipv6-cidr-block ipv6-
cidr --ipv6-pool pool-id --region us-east-1
```

VPC および関連する IPv6 アドレスプール情報を表示するにはdescribe-vpcs コマンドを使用します。特定の IPv6 アドレスプールから関連付けられた IPv6 CIDR ブロックに関する情報を表示するには次の get-associated-ipv6-pool-cidrs コマンドを使用します。

```
aws ec2 get-associated-ipv6-pool-cidrs --pool-id pool-id --region us-east-1
```

VPC から IPv6 CIDR ブロックの関連付けを解除すると、IPv6 アドレスプールに戻されます。

Elastic IP アドレス

Elastic IP アドレスは動的なクラウドコンピューティングのために設計された静的 IPv4 アドレスです。Elastic IP アドレスはユーザーの AWS アカウントに割り当てられ、リリースするまでユーザーのアドレスになります。Elastic IP アドレスを使用すると、アドレスをアカウント内の別のインスタンスに迅速に再マッピングすることで、インスタンスやソフトウェアの障害をマスクできます。またはドメインがインスタンスを参照するように、ドメインの DNS レコードに Elastic IP アドレスを指定することもできます。詳細については使用しているドメインレジストラのドキュメントを参照してください。

Elastic IP アドレスはインターネットからアクセス可能なパブリック IPv4 アドレスです。パブリック IPv4 アドレスを持たないインスタンスに接続する必要がある場合はElastic IP アドレスをインスタンスに関連付けることで、インターネットとの通信を有効にできます。

内容

- Elastic IP アドレスの料金
- Elastic IP アドレスの基本
- Elastic IP アドレスのクォータ
- Elastic IP アドレスをインスタンスに関連付ける
- AWS アカウント間での Elastic IP アドレスの移管
- Elastic IP アドレスをリリース
- Amazon EC2 で E メールの逆引き DNS レコードを作成する

Elastic IP アドレスの料金

すべての Elastic IP アドレスでは、使用中か (EC2 インスタンスなどのリソースに割り当てられているか)、アイドル状態か (アカウントに作成されているが割り当てられてはいない) にかかわらず、料金が発生します。

AWS は、実行中のインスタンスに関連付けられているパブリック IPv4 アドレスや Elastic IP アドレスを含めたすべてのパブリック IPv4 アドレスの料金を請求します。詳細については「Amazon VPC の料金」ページの「パブリック IPv4 アドレス」タブを参照してください。

Elastic IP アドレスの基本

Elastic IP アドレスの基本的な特徴を次に示します。

- Elastic IP アドレスは静的であり、時間の経過とともに変わることはありません。
- Elastic IP アドレスは特定のリージョン専用であり、別のリージョンに移動することはできません。
- Elastic IP アドレスはAmazon が持っている IPv4 アドレスのプールまたはお客様が AWS アカウント に持ち込んだカスタム IPv4 アドレスのプールから割り当てることができます。IPv6 に対する Elastic IP アドレスはサポートされていません。
- Elastic IP アドレスを使用するにはまずアカウントに 1 つ割り当ててから、それをインスタンスまたはネットワークインターフェイスに関連付けます。
- Elastic IP アドレスをインスタンスと関連付けると、インスタンスのプライマリネットワークインターフェイスとも関連付けられます。Elastic IP アドレスをインスタンスにアタッチされたネットワークインターフェイスと関連付けると、インスタンスとも関連付けられます。
- Elastic IP アドレスをインスタンスまたはそのプライマリネットワークインターフェイスに関連付けると、インスタンスに既にパブリック IPv4 アドレスが関連付けられている場合、そのパブリック IPv4 アドレスは Amazon のパブリック IPv4 アドレスのプールに解放され、Elastic IP アドレスは代わりにインスタンスに関連付けられます。以前にインスタンスに関連付けられたパブリック IPv4 アドレスを再利用することはできず、そのパブリック IPv4 アドレスを Elastic IP アドレスに変換することもできません。詳細については「パブリック IPv4 アドレス」を参照してください。
- リソースから Elastic IP アドレスの関連付けを解除し、別のリソースと関連付けることができます。予期しない動作を避けるため、変更を行う前に、既存の関連付けで指定されたリソースへのアクティブな接続をすべて閉じていることを確認してください。Elastic IP アドレスを別のリソースに関連付けた後、新しく関連付けられたリソースへの接続を再度開くことができます。
- 関連付けが解除された Elastic IP アドレスは明示的にリリースするまでアカウントに割り当てられたままです。インスタンスに関連付けられているか、関連付けが解除されているかにかかわらず、アカウント内のすべての Elastic IP アドレスに対して課金されます。詳細については「Amazon VPC の料金」ページの「パブリック IPv4 アドレス」タブを参照してください。
- パブリック IPv4 アドレスが前回割り当てられたインスタンスに Elastic IP アドレスを関連付けると、インスタンスのパブリック DNS ホスト名はElastic IP アドレスに一致するように変更されます。
- パブリック DNS ホスト名を解決すると、インスタンスのパブリック IPv4 アドレスまたは Elastic IP アドレス (インスタンスのネットワークの外部の場合)、およびインスタンスのプライベート IPv4 アドレス (インスタンスのネットワーク内からの場合) となります。
- AWS アカウントに持ち込んだ IP アドレスプールから Elastic IP アドレスを割り当てた場合、Elastic IP アドレス制限にカウントされません。詳細についてはElastic IP アドレスのクォータを参照してください。

- Elastic IP アドレスを割り当てると、Elastic IP アドレスをネットワークボーダーグループに関連付けることができます。これはCIDR ブロックをアドバタイズする場所です。ネットワークボーダーグループを設定すると、CIDR ブロックがこのグループに制限されます。ネットワークボーダーグループを指定しない場合はリージョン (us-west-2 など) のすべてのアベイラビリティーゾーンを含むボーダーグループが自動的に設定されます。
- Elastic IP アドレスはネットワークボーダーグループ別に専用になっています。

Elastic IP アドレスのクォータ

デフォルトではすべての AWS アカウント でリージョンあたり 5 つの Elastic IP アドレスが割り当てられています。これはパブリック (IPv4) インターネットアドレスが数に限りのあるパブリックリソースであるためです。インスタンスに障害が発生した場合にアドレスを他のインスタンスに再マップする機能のために主に Elastic IP アドレスを使用し、他のすべてのノード間通信には DNS ホスト名を使用することを強くお勧めします。

お客様のアーキテクチャで追加の Elastic IP アドレスが必要な場合、クォータの引き上げを Service Quotas コンソールから直接リクエストできます。クォータの引き上げをリクエストするには[アカウントレベルでの引き上げをリクエスト] を選択してください。詳細については「Amazon EC2 の Service Quotas」を参照してください。

Elastic IP アドレスをインスタンスに関連付ける

Elastic IP アドレスを割り当てると、EC2 インスタンス、NAT ゲートウェイ、Network Load Balancer などの AWS リソースに関連付けることができます。後で Elastic IP アドレスを別の AWS リソースに関連付けるには現在のリソースから関連付けを解除してから、新しいリソースに関連付けます。

タスク

- Elastic IP アドレスを割り当てる
- Elastic IP アドレスの関連付け
- Elastic IP アドレスの関連付けを解除する

Elastic IP アドレスを割り当てる

リージョンで使用するための Elastic IP アドレスを割り当てることができます。すべての Elastic IP アドレスでは、使用中か (EC2 インスタンスなどのリソースに関連付けられている)、アイドル状態か (アカウントに作成されているが関連付けられてはいない) にかかわらず、料金が発生します。

Console

Elastic IP アドレスを割り当てるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Network & Security]、[Elastic IPs] の順に選択してください。
3. [Allocate Elastic IP address] を選択してください。
4. (オプション) Elastic IP アドレス (EIP) を割り当てるときはEIP を割り当てるネットワークボーダーグループを選択してください。ネットワークボーダーグループはAWS がパブリック IP アドレスをアドバタイズするアベイラビリティーゾーン (AZ)、Local Zones、または Wavelength Zones のコレクションです。Local Zones と Wavelength Zones はAWS ネットワークとこれらのゾーンのリソースにアクセスする顧客との間のレイテンシーや物理的距離を最小限に抑えるために、リージョン内の AZ とは異なるネットワークボーダーグループを持つ場合があります。

> **Important**
>
> EIP に関連付ける AWS リソースと同じネットワークボーダーグループに EIP を割り当てる必要があります。あるネットワークボーダーグループ内の EIP はそのネットワークボーダーグループ内のゾーンでのみアドバタイズでき、他のネットワークボーダーグループで表される他のゾーンではアドバタイズできません。

Local Zones または Wavelength Zones を有効にしている場合 (詳細については「Local Zone を有効にする」または「Wavelength Zones を有効にする」を参照)、AZ、Local Zones、または Wavelength Zones のネットワークボーダーグループを選択できます。EIP とそれが関連付けられている AWS リソースは同じネットワークボーダーグループに属している必要があるため、ネットワークボーダーグループは慎重に選択してください。EC2 コンソールを使用して、アベイラビリティーゾーン、ローカルゾーン、または Wavelength Zones が属するネットワークボーダーグループを表示できます。通常、リージョン内のすべてのアベイラビリティーゾーンは同じネットワークボーダーグループに属しますが、Local Zones や Wavelength Zones はそれぞれ別のネットワークボーダーグループに属します。

Local Zones または Wavelength Zones が有効になっていない場合、EIP を割り当てると、リージョン (`us-west-2` など) のすべての AZ を表すネットワークボーダーグループが定義済みになり、変更することはできません。つまり、このネットワークボーダーグループに割り当てた EIP は現在のリージョンのすべての AZ でアドバタイズされます。

5. [Public IPv4 address pool (パブリック IPv4 アドレスのプール)] で、以下のいずれかを選択してください。

   - [Amazon's pool of IPv4 addresses (Amazon の IP アドレスのプール)] — Amazon の IPv4 アドレスのプールから IPv4 アドレスを割り当てる場合。
   - [AWS アカウントに持ち込むパブリック IPv4 アドレス] - AWS アカウントに持ち込んだ IP アドレスプールから非連続 (非シーケンシャル) パブリック IPv4 アドレスを割り当てる場合。IP アドレスプールがない場合、このオプションは無効になります。AWS アカウントへの独自の IP アドレス範囲の持ち込みの詳細については Amazon EC2 に自分の IP アドレスを持ち込む (BYOIP) をご参照ください。
   - ユーザー所有の IPv4 アドレスのプール - AWS Outpost で使用するために、オンプレミスネットワークから作成したプールから IPv4 アドレスを割り当てる場合。AWS Outpost がない場合、このオプションは無効になります。
   - [IPAM IPv4 プールを使用して割り当てる]: IPAM プール内の連続したパブリック IPv4 ブロックからシーケンシャル Elastic IP アドレスを割り当てる場合。シーケンシャル Elastic IP アドレスを割り当てると、セキュリティアクセスコントロールリストの管理オーバーヘッドが大幅に削減され、AWS でスケーリングする企業の IP アドレスの割り当てと追跡が簡素化できます。詳細については「Amazon VPC IPAM ユーザーガイド」の「Allocate sequential Elastic IP addresses from an IPAM pool」を参照してください。

6. (オプション) タグを追加するには[新しいタグを追加] を選択し、タグキーとタグ値を入力してください。

AWS CLI

Elastic IP アドレスを割り当てるには

allocate-address AWS CLI コマンドを使用します。

次の例では、Amazon EC2 が Amazon のアドレスプールからアドレスを選択します。

```
aws ec2 allocate-address
```

次の例では、ユーザーが BYOIP を使用して AWS に持ち込んだ指定のプールから Amazon EC2 がアドレスを選択します。

```
aws ec2 allocate-address \
    --public-ipv4-pool ipv4pool-ec2-012345abcdef67890
```

次の例は、指定された IPv4 IPAM プールからのアドレスを指定します。

```
aws ec2 allocate-address \
    --ipam-pool-id ipam-pool-1234567890abcdef0 \
    --address 192.0.2.0
```

PowerShell

Elastic IP アドレスを割り当てるには

New-EC2Address コマンドレットを使用します。

次の例では、Amazon EC2 が Amazon のアドレスプールからアドレスを選択します。

```
New-EC2Address
```

次の例では、ユーザーが BYOIP を使用して AWS に持ち込んだ指定のプールから Amazon EC2 がアドレスを選択します。

```
New-EC2Address `
    -PublicIpv4Pool ipv4pool-ec2-012345abcdef67890
```

次の例は、指定された IPv4 IPAM プールからのアドレスを指定します。

```
New-EC2Address `
    -IpamPoolId ipam-pool-1234567890abcdef0 `
    -Address 192.0.2.0
```

Elastic IP アドレスの関連付け

Elastic IP アドレスをインスタンスに関連付けてインターネットとの通信を有効にする場合、インスタンスがパブリックサブネットに属していることも確認する必要があります。詳細については「Amazon VPC ユーザーガイド」の「Enable internet access using an internet gateway」を参照してください。

Console

Elastic IP アドレスをインスタンスに関連付けるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

2. ナビゲーションペインで [Elastic IP] を選択してください。
3. 関連付ける Elastic IP アドレスを選択してから、[Actions (アクション)]、[Associate Elastic IP address (Elastic IP アドレスの関連付け)] の順に選択してください。
4. [リソースタイプ] で、[Instance (インスタンス)] を選択してください。
5. 例えば、Elastic IP アドレスを関連付けるインスタンスを選択してください。テキストを入力して特定のインスタンスを検索することもできます。
6. (オプション) [プライベート IP アドレス] で、Elastic IP アドレスを関連付けるプライベート IP アドレスを指定します。
7. [Associate] を選択してください。

Elastic IP アドレスとネットワークインターフェイスを関連付けるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Elastic IP] を選択してください。
3. 関連付ける Elastic IP アドレスを選択してから、[Actions (アクション)]、[Associate Elastic IP address (Elastic IP アドレスの関連付け)] の順に選択してください。
4. [リソースタイプ] で、[ネットワークインターフェイス] を選択してください。
5. [ネットワークインターフェイス] で、Elastic IP アドレスを関連付けるネットワークインターフェイスを選択してください。テキストを入力して、特定のネットワークインターフェイスを検索することもできます。
6. (オプション) [プライベート IP アドレス] で、Elastic IP アドレスを関連付けるプライベート IP アドレスを指定します。
7. [Associate] を選択してください。

AWS CLI

Elastic IP アドレスを関連付けるには

associate-address AWS CLI コマンドを使用します。

```
aws ec2 associate-address \
    --instance-id i-0b263919b6498b123 \
    --allocation-id eipalloc-64d5890a
```

PowerShell

Elastic IP アドレスを関連付けるには

Register-EC2Address コマンドレットを使用します。

```
Register-EC2Address `
    -InstanceId i-0b263919b6498b123 `
    -AllocationId eipalloc-64d5890a
```

Elastic IP アドレスの関連付けを解除する

インスタンスまたはネットワークインターフェイスから Elastic IP アドレスの関連付けをいつでも解除できます。Elastic IP アドレスの関連付けを解除した後、そのアドレスを別のリソースに関連付けることができます。

Console

Elastic IP アドレスの関連付けを解除して再度関連付けするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Elastic IP] を選択してください。
3. 関連付けを解除する Elastic IP アドレスを選択してから、[Actions (アクション)]、[Elastic IP アドレスの関連付けの解除] の順に選択してください。
4. [関連付け解除] を選択してください。

AWS CLI

Elastic IP アドレスの関連付けを解除するには

disassociate-address AWS CLI コマンドを使用します。

```
aws ec2 disassociate-address --association-id eipassoc-12345678
```

PowerShell

Elastic IP アドレスの関連付けを解除するには

Unregister-EC2Address コマンドレットを使用します。

```
Unregister-EC2Address -AssociationId eipassoc-12345678
```

AWS アカウント間での Elastic IP アドレスの移管

Elastic IP アドレスを AWS アカウント間で移管できます。これは次のような状況で役立つ場合があります。

- ディザスタリカバリ – 緊急時に、一般向けインターネットワークロードの IP アドレスをすばやく再マッピングします。
- 組織の再構築 - AWS アカウント間でワークロードをすばやく移動します。アドレスの移管により、セキュリティグループやネットワーク ACL によって新しい Elastic IP アドレスが許可されるのを待つ必要がなくなります。
- 一元的なセキュリティ管理 - 一元化された AWS セキュリティアカウントを使用して、セキュリティコンプライアンスのために精査された Elastic IP アドレスを追跡および移管できます。

料金

Elastic IP アドレスの移管には料金はかかりません。

タスク

- Elastic IP アドレスの移管を有効にする
- 移管された Elastic IP アドレスを承諾する
- Elastic IP アドレスの移管を無効にする

Elastic IP アドレスの移管を有効にする

このセクションでは移管された Elastic IP アドレスを承諾する方法について説明します。Elastic IP アドレスの移管を有効にする際には以下の制限に注意してください。

- 任意の AWS アカウント (ソースアカウント) から同じ AWS リージョン内の他の AWS アカウント (転送先アカウント) に Elastic IP アドレスを転送できます。Elastic IP アドレスを別のリージョンに転送することはできません。
- Elastic IP アドレスを転送する場合、AWS アカウント の間で 2 段階のハンドシェイクが行われます。ソースアカウントが移管を開始してから 7 日間は転送先アカウントが Elastic IP アドレス移管を受け入れることができます。この 7 日間、ソースアカウントは保留中の移管をAWS コンソール

やコマンドの describe-address-transfers などを使用して) 確認できます。7 日後、移管の有効期限が切れ、Elastic IP アドレスの所有権がソースアカウントに戻ります。

- 移管が受け入れられてから 14 日間、ソースアカウントは受け入れられた移管をAWS コンソールや コマンド describe-address-transfers などを使用して) 表示できます。
- AWS は保留中の Elastic IP アドレス転送リクエストについて、転送先アカウントに通知しません。ソースアカウントの所有者は承諾する必要がある Elastic IP アドレス転送リクエストがあることを転送先アカウントの所有者に通知する必要があります。
- 転送中の Elastic IP アドレスに関連付けられているタグは転送が完了するとリセットされます。
- AWS アカウント に持ち込んだパブリック IPv4 アドレスプール (一般的に Bring-Your-Own-IP (BYOIP) アドレスプールと呼ばれる) から割り当てられた Elastic IP アドレスは転送できません。
- Amazon が提供する連続したパブリック IPv4 Amazon VPC IP Address Manager (IPAM) プールから割り当てられた Elastic IP アドレスを転送することはできません。代わりに、IPAM ではIPAM を AWS Organizations と統合し、AWS RAM を使用することで、AWS アカウント間で IPAM プールを共有できます。詳細については「Amazon VPC IPAM ユーザーガイド」の「Allocate sequential Elastic IP addresses from an IPAM pool」を参照してください。
- リバース DNS レコードが関連付けられている Elastic IP アドレスを移管しようとする場合、移管プロセスを開始することはできますが、関連付けられている DNS レコードが削除されるまで、転送先アカウントは移管を受け入れることができません。
- AWS Outposts を有効にして設定している場合はカスタマー所有の IP アドレスプール (CoIP) から Elastic IP アドレスを割り当てている可能性があります。CoIP から割り当てられた Elastic IP アドレスを転送することはできません。ただし、AWS RAM を使用して CoIP を別のアカウントと共有することはできます。CoIP の詳細についてはAWS Outposts ユーザーガイドの「カスタマー所有 IP アドレス」を参照してください 。
- Amazon VPC IPAM を使用して、AWS Organizations から組織内のアカウントへの Elastic IP アドレスの転送を追跡することができます。詳細については「IP アドレスの履歴の表示」を参照してください。Elastic IP アドレスが組織外の AWS アカウント に転送されると、その Elastic IP アドレスの IPAM 監査履歴は失われます。

これらのステップはソースアカウントで実行する必要があります。

Console

Elastic IP アドレスの移管を有効にするには

1. 転送元となる AWS アカウントを使用していることを確認してください。

2. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
3. ナビゲーションペインで [Elastic IP] を選択してください。
4. 移管を有効にする Elastic IP アドレスを 1 つ以上選択し、[Action] (アクション)、[Enable transfer] (移管を有効にする) を選択してください。
5. 複数の Elastic IP アドレスを移管する場合は[Transfer type] (転送タイプ) オプションが表示されます。以下のオプションのいずれかを選択してください。
   - Elastic IP アドレスを単一の AWS アカウントに移管する場合は[Single account] (単一アカウント) を選択してください。
   - Elastic IP アドレスを複数の AWS アカウントに移管する場合は[Multiple accounts] (複数アカウント) を選択してください。
6. [Transfer account ID] (アカウント ID の移管) に、Elastic IP アドレスの転送先の AWS アカウント ID を入力してください。
7. テキストボックスに「**enable**」と入力して移管を確定します。
8. [送信] を選択してください。
9. 移管を承諾するには「移管された Elastic IP アドレスを承諾する」を参照してください。移管を無効にするには「Elastic IP アドレスの移管を無効にする」を参照してください。

AWS CLI

Elastic IP アドレスの移管を有効にするには

enable-address-transfer コマンドを使用します。

```
aws ec2 enable-address-transfer \
    --allocation-id eipalloc-09ad461b0d03f6aaf \
    --transfer-account-id 123456789012
```

PowerShell

Elastic IP アドレスの移管を有効にするには

Enable-EC2AddressTransfer コマンドレットを使用します。

```
Enable-EC2AddressTransfer `
    -AllocationId eipalloc-09ad461b0d03f6aaf `
    -TransferAccountId 123456789012
```

移管された Elastic IP アドレスを承諾する

このセクションでは移管された Elastic IP アドレスを承諾する方法について説明します。

Elastic IP アドレスを転送する場合、AWS アカウント の間で 2 段階のハンドシェイクが行われます。ソースアカウントが移管を開始してから 7 日間は転送先アカウントが Elastic IP アドレス移管を受け入れることができます。この 7 日間、ソースアカウントは保留中の移管をAWS コンソールやコマンドの describe-address-transfers などを使用して) 確認できます。7 日後、移管の有効期限が切れ、Elastic IP アドレスの所有権がソースアカウントに戻ります。

転送を承諾する際に発生する可能性のある例外と、解決する方法は次のとおりです。

- AddressLimitExceeded: 転送先アカウントが Elastic IP アドレスのクォータを超えている場合、ソースアカウントは Elastic IP アドレス移管を有効にできますが、この例外は転送先アカウントが移管を承諾しようとした場合に発生します。デフォルトではすべての AWS アカウントはリージョンあたり 5 つの Elastic IP アドレスに制限されています。制限を増やす方法については「Elastic IP アドレスのクォータ」を参照してください。
- InvalidTransfer.addressCustomPtrSet: お客様または組織内の誰かが、移管しようとしている Elastic IP アドレスをリバース DNS ルックアップを使用するように設定している場合、ソースアカウントは Elastic IP アドレスの移管を有効にできますが、転送元アカウントが転送を受け入れようとするとこの例外が発生します。この問題を解決するには転送元アカウントで Elastic IP アドレスの DNS レコードを削除する必要があります。詳細については「Amazon EC2 で E メールの逆引き DNS レコードを作成する」を参照してください。
- InvalidTransfer.AddressAssociated: Elastic IP アドレスが ENI や EC2 インスタンスと関連付けられている場合、転送元アカウントはその Elastic IP アドレスに対して移管を有効にできますが、転送元アカウントが移管を受け入れようとするとこの例外が発生します。この問題を解決するにはソースアカウントが Elastic IP アドレスの関連付けを解除する必要があります。詳細については「Elastic IP アドレスの関連付けを解除する」を参照してください。

その他の例外についてはサポート にお問い合わせください。

これらのステップは転送先アカウントで実行する必要があります。

Console

Elastic IP アドレスの移管を承諾するには

1. 転送先アカウントを使用していることを確認してください。
2. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

3. ナビゲーションペインで [Elastic IP] を選択してください。
4. [Action] (アクション)、[Accept transfer] (移管を許可する) を選択してください。
5. 転送を受け入れると、移管される Elastic IP アドレスに関連付けられたタグは転送されません。承諾する Elastic IP アドレスの [Name] (名前) タグを定義する場合は[Create a tag with a key of 'Name' and a value that you specify] ('Name'のキーと指定した値を使用してタグを作成) を選択してください。
6. 移管する Elastic IP アドレスを入力してください。
7. 複数の移管された Elastic IP アドレスを受け入れる場合は[Add address] (アドレスを追加) を選択して追加の Elastic IP アドレスを入力してください。
8. [送信] を選択してください。

AWS CLI

Elastic IP アドレスの移管を承諾するには

accept-address-transfer コマンドを使用します。

```
aws ec2 accept-address-transfer --address 100.21.184.216
```

PowerShell

Elastic IP アドレスの移管を承諾するには

Approve-EC2AddressTransfer コマンドレットを使用します。

```
Approve-EC2AddressTransfer -Address 100.21.184.216
```

Elastic IP アドレスの移管を無効にする

このセクションではElastic IP 移管を有効にした後に Elastic IP 転送を無効にする方法について説明します。

これらのステップは移管を有効にしたソースアカウントが実行する必要があります。

Console

Elastic IP アドレスの移行を無効にするには

1. 転送元となる AWS アカウントを使用していることを確認してください。

2. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
3. ナビゲーションペインで [Elastic IP] を選択してください。
4. Elastic IP のリソースリストで、[Transfer status] (移管ステータス) 列を表示するプロパティが有効になっていることを確認します。
5. [Transfer status] (移管ステータス) が [Pending] (保留中) の Elastic IP アドレスを 1 つ以上選択し、[Action] (アクション)、[Disable transfer] (移管を無効にする) を選択してください。
6. テキストボックスに「**disable**」と入力して確認します。
7. [送信] を選択してください。

AWS CLI

Elastic IP アドレスの移管を無効にするには

disable-address-transfer コマンドを使用します。

```
aws ec2 disable-address-transfer --allocation-id eipalloc-09ad461b0d03f6aaf
```

PowerShell

Elastic IP アドレスの移管を無効にするには

Disable-EC2AddressTransfer コマンドレットを使用します。

```
Disable-EC2AddressTransfer -AllocationId eipalloc-09ad461b0d03f6aaf
```

Elastic IP アドレスをリリース

Elastic IP アドレスが不要になった場合は解放することをお勧めします。解放する Elastic IP アドレスは現在、AWS リソースに関連付けられていてはなりません。

Console

Elastic IP アドレスを解放するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Elastic IP] を選択してください。
3. リリースする Elastic IP アドレスを選択してから、[アクション]、 [Elastic IP アドレスのリリース] の順に選択してください。

4. [リリース] を選択してください。

AWS CLI

Elastic IP アドレスをリリースするには

release-address AWS CLI コマンドを使用します。

```
aws ec2 release-address --allocation-id eipalloc-64d5890a
```

PowerShell

Elastic IP アドレスをリリースするには

Remove-EC2Address コマンドレットを使用します。

```
Remove-EC2Address -AllocationId eipalloc-64d5890a
```

Elastic IP アドレスを解放した後に、復元できる場合があります。以下のルールが適用されます。

- Elastic IP アドレスが別の AWS アカウントに割り当てられている場合や Elastic IP アドレスの制限を超過する場合はElastic IP アドレスを復元できません。
- Elastic IP アドレスに関連付けられたタグを復元することはできません。

AWS CLI

Elastic IP アドレスを復元するには

allocate-address コマンドレットを使用します。

```
aws ec2 allocate-address \
    --domain vpc \
    --address 203.0.113.3
```

PowerShell

Elastic IP アドレスを復元するには

New-EC2Address コマンドレットを使用します。

```
New-EC2Address `
    -Address 203.0.113.3 `
    -Domain vpc `
    -Region us-east-1
```

Amazon EC2 で E メールの逆引き DNS レコードを作成する

EC2 インスタンスから第三者に E メールを送信する場合は1 つ以上の Elastic IP アドレスをプロビジョニングし、E メールの送信に使用する Elastic IP アドレスに静的な逆引き DNS レコードを割り当てることをお勧めします。これにより、電子メールが一部のスパム対策組織によりスパムとしてフラグ付けされるのを防ぐことができます。AWSはISP およびインターネットアンチスパム組織と協力して、これらのアドレスから送信された E メールにスパムのフラグが付く可能性を減らしています。

考慮事項

- 逆引き DNS レコードを作成する前に、お使いのアカウントの Elastic IP アドレスに解決される対応するフォワード DNS レコード (レコードタイプ A) を設定しておく必要があります。ベストプラクティスとして、逆引き DNS レコードを作成する予定のすべての Elastic IP アドレスをフォワード DNS レコードに含めておくことをお勧めします。
- 逆引き DNS レコードが Elastic IP アドレスに関連付けられている場合、その Elastic IP アドレスはアカウントにロックされ、レコードが削除されるまでアカウントからリリースすることはできません。
- サポートに問い合わせて Elastic IP アドレスの逆引き DNS を設定する場合、逆引き DNS を削除することはできますが、Elastic IP アドレスを解放することはできません。これはElastic IP アドレスが サポートによってロックされているためです。Elastic IP アドレスのロックを解除するにはAWS サポート にお問い合わせください。Elastic IP アドレスはロックを解除した後に解放することができます。
- [AWS GovCloud (US) Region] 逆引き DNS レコードを作成することはできません。AWS から静的な逆引き DNS レコードが割り当てられる必要があります。サポートケースを開いて、逆引き DNS と E メール送信の制限を削除します。エラスチック IP アドレスと逆引き DNS レコードを指定する必要があります。

逆引き DNS レコードを作成する

Elastic IP アドレスの逆引き DNS レコードを作成するには次の手順を実行します。

Console

逆引き DNS レコードを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Elastic IP] を選択してください。
3. Elastic IP アドレスを選択し、[Actions]、[Update reverse DNS] の順に選択してください。
4. [Reverse DNS domain name] (リバース DNS ドメイン名) の場合、ドメイン名を入力してください。
5. **update** を入力して確定します。
6. [更新] を選択してください。

AWS CLI

逆引き DNS レコードを作成するには

modify-address-attribute コマンドを使用します。

```
aws ec2 modify-address-attribute \
    --allocation-id eipalloc-abcdef01234567890 \
    --domain-name example.com
```

以下は出力の例です。

```
{
    "Addresses": [
        {
            "PublicIp": "192.0.2.0",
            "AllocationId": "eipalloc-abcdef01234567890",
            "PtrRecord": "example.net.",
            "PtrRecordUpdate": {
                "Value": "example.com.",
                "Status": "PENDING"
            }
        }
    ]
}
```

PowerShell

逆引き DNS レコードを作成するには

Edit-EC2AddressAttribute コマンドレットを使用します。

```
Edit-EC2AddressAttribute `
    -AllocationId 'eipalloc-abcdef01234567890' `
    -DomainName 'example.com' |
Format-List `
    AllocationId, PtrRecord, PublicIp,
    @{Name='PtrRecordUpdate';Expression={$_.PtrRecordUpdate | Format-List | Out-
String}}
```

以下は出力の例です。

```
AllocationId    : eipalloc-abcdef01234567890
PtrRecord       : example.net.
PublicIp        : 192.0.2.0
PtrRecordUpdate :
                  Reason :
                  Status : PENDING
                  Value  : example.com.
```

逆引き DNS レコードを削除する

Elastic IP アドレスから逆引き DNS レコードを削除するには次の手順を実行します。

次のエラーが表示された場合はサポートに E メール送信制限の解除リクエストを送信することで、サポートを受けることができます。

```
The address cannot be released because it is locked to your account.
```

Console

逆引き DNS レコードを削除するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Elastic IP] を選択してください。
3. Elastic IP アドレスを選択し、[Actions]、[Update reverse DNS] の順に選択してください。

4. [Reverse DNS domain name] (リバース DNS ドメイン名) の場合、ドメイン名をクリアします。
5. **update** を入力して確定します。
6. [更新] を選択してください。

AWS CLI

逆引き DNS レコードを削除するには

reset-address-attribute コマンドを使用します。

```
aws ec2 reset-address-attribute \
    --allocation-id eipalloc-abcdef01234567890 \
    --attribute domain-name
```

以下は出力の例です。

```
{
    "Addresses": [
        {
            "PublicIp": "192.0.2.0",
            "AllocationId": "eipalloc-abcdef01234567890",
            "PtrRecord": "example.com.",
            "PtrRecordUpdate": {
                "Value": "example.net.",
                "Status": "PENDING"
            }
        }
    ]
}
```

PowerShell

逆引き DNS レコードを削除するには

Reset-EC2AddressAttribute コマンドレットを使用します。

```
Reset-EC2AddressAttribute `
    -AllocationId 'eipalloc-abcdef01234567890' `
    -Attribute domain-name |
Format-List `
```

```
    AllocationId, PtrRecord, PublicIp,
    @{Name='PtrRecordUpdate';Expression={$_.PtrRecordUpdate | Format-List | Out-
String}}
```

以下は出力の例です。

```
AllocationId    : eipalloc-abcdef01234567890
PtrRecord       : example.com.
PublicIp        : 192.0.2.0
PtrRecordUpdate :
                  Reason :
                  Status : PENDING
                  Value  : example.net.
```

Elastic Network Interface

Elastic Network Interface は仮想ネットワークカードを表す VPC 内の論理ネットワーキングコンポーネントです。ネットワークインターフェイスを作成および設定して、同じアベイラビリティーゾーンで起動するインスタンスにアタッチできます。ネットワークインターフェイスをインスタンスにアタッチしたり、インスタンスからデタッチして別のインスタンスに再アタッチしたりするときにはネットワークインターフェイスの属性が保持されます。インスタンス間でネットワークインターフェイスを移動すると、ネットワークトラフィックは元のインスタンスから新しいインスタンスにリダイレクトされます。

この AWS リソースはAWS マネジメントコンソール および Amazon EC2 API ではネットワークインターフェイスと呼ばれているので注意してください。したがって、このドキュメントでは Elastic Network Interface ではなく「ネットワークインターフェイス」を使用します。このドキュメントの「ネットワークインターフェイス」という用語は常に Elastic Network Interface を意味します。

ネットワークインターフェイスの属性

ネットワークインターフェイスには以下の属性を含めることができます。

- サブネットの IPv4 アドレス範囲からのプライマリプライベート IPv4 アドレス
- サブネットの IPv6 アドレス範囲からのプライマリ IPv6 アドレス
- サブネットの IPv4 アドレス範囲からのセカンダリプライベート IPv4 アドレス
- プライベート IPv4 アドレスごとに 1 つの Elastic IP アドレス (IPv4)
- 1 つのパブリック IPv4 アドレス

- セカンダリ IPv6 アドレス
- セキュリティグループ
- MAC アドレス
- 送信元/送信先チェックフラグ
- 説明

トラフィックのモニタリング

ネットワークインターフェイスで VPC フローログを有効にして、ネットワークインターフェイスとの間で行き来するトラフィックに関する情報をキャプチャできます。フローログを作成すると、そのデータを Amazon CloudWatch Logs で表示し、取得できます。詳細についてはAmazon VPC ユーザーガイド のVPC フローログを参照してください。

内容

- ネットワークインターフェイスの概念
- ネットワークカード
- ネットワークインターフェイスあたりの IP アドレスの最大数
- EC2 インスタンス用のネットワークインターフェイスを作成する
- EC2 インスタンスのネットワークインターフェイスのアタッチ
- ネットワークインターフェイスの IP アドレスを管理する
- ネットワークインターフェイス属性の変更
- Amazon EC2 インスタンス用の複数のネットワークインターフェイス
- リクエスタマネージド型のネットワークインターフェイス
- Amazon EC2 ネットワークインターフェイスのプレフィックス委任
- ネットワークインターフェイスの削除

ネットワークインターフェイスの概念

ネットワークインターフェイスの使用を開始する際に理解しておくべき重要な概念を次に示します。

プライマリネットワークインターフェイス

各インスタンスにはプライマリネットワークインターフェイスと呼ばれるデフォルトのネットワークインターフェイスがあります。プライマリネットワークインターフェイスをインスタンスからデタッチすることはできません。

セカンダリネットワークインターフェイス

セカンダリネットワークインターフェイスを作成して、インスタンスにアタッチできます。ネットワークインターフェイスの最大数はインスタンスタイプによって異なります。詳細については「ネットワークインターフェイスあたりの IP アドレスの最大数」を参照してください。

ネットワークインターフェイスの IPv4 アドレス

EC2 インスタンスを IPv4 専用サブネットまたはデュアルスタックサブネットで起動すると、インスタンスはサブネットの IPv4 アドレス範囲からプライマリプライベート IP アドレスを受け取ります。追加のプライベート IPv4 アドレス (セカンダリプライベート IPv4 アドレス) も指定できます。プライマリプライベート IP アドレスとは異なり、セカンダリプライベート IP アドレスは別のインスタンスに割り当て直すことができます。

ネットワークインターフェイスのパブリック IPv4 アドレス

すべてのサブネットに、そのサブネットで作成されるネットワークインターフェイス (結果的にそのサブネットで起動されるインスタンス) にパブリック IPv4 アドレスを割り当てるかどうかを決定する、変更可能な属性があります。詳細については「Amazon VPC ユーザーガイド」の「サブネット設定」を参照してください。インスタンスを起動すると、プライマリネットワークインターフェイスに IP アドレスが割り当てられます。インスタンスの起動時に既存のネットワークインターフェイスをプライマリネットワークインターフェイスとして指定した場合、パブリック IPv4 アドレスはこのネットワークインターフェイスによって決まります。

ネットワークインターフェイスを作成すると、サブネットからパブリック IPv4 アドレス指定属性を継承します。後でサブネットのパブリック IPv4 アドレス指定属性を変更しても、ネットワークインターフェイスでは作成時に有効だった設定が保持されます。

インスタンスが停止、休止状態、または終了すると、インスタンスのパブリック IP アドレスが開放されます。Elastic IP アドレスに関連付けられたセカンダリネットワークインターフェイスまたはセカンダリプライベート IPv4 アドレスがない限り、停止または休止したインスタンスを起動すると、新しいパブリック IP アドレスが割り当てられます。

ネットワークインターフェイスの IPv6 アドレス

IPv6 CIDR ブロックを VPC とサブネットに関連付けると、サブネットの範囲から IPv6 アドレスをネットワークインターフェイスに割り当てることができます。各 IPv6 アドレスは1 つのネットワークインターフェイスに割り当てることができます。

すべてのサブネットにはそのサブネットで作成されるネットワークインターフェイス (結果的にそのサブネットで起動されるインスタンス) にサブネットの範囲から IPv6 アドレスを自動的に割

り当てるかどうかを決定する、変更可能な属性があります。インスタンスを起動すると、プライマリネットワークインターフェイスに IPv6 アドレスが割り当てられます。

ネットワークインターフェイスの Elastic IP アドレス

Elastic IP アドレスはネットワークインターフェイスの 1 つのプライベート IPv4 アドレスに関連付けることができます。1 つの Elastic IP アドレスと各プライベート IPv4 アドレスを関連付けることができます。ネットワークインターフェイスから Elastic IP アドレスを関連付け解除すると、その Elastic IP アドレスは解放することも他のインスタンスに関連付けることもできます。

終了動作

インスタンスにアタッチされているネットワークインターフェイスの終了動作を設定できます。アタッチしたインスタンスの終了時に、ネットワークインターフェイスを自動的に削除するかどうかを指定できます。

送信元/送信先チェック

送信元/送信先チェックを有効または無効にできます。これにより、受信するすべてのトラフィックに関して、そのインスタンスが送信元なのか、あるいは送信先であるのかを確認できます。送信元/送信先チェックはデフォルトで有効化されています。ネットワークアドレス変換、ルーティング、ファイアウォールなどのサービスを実行するインスタンスでは送信元/送信先チェックを無効にする必要があります。

リクエスタマネージド型のネットワークインターフェイス

これらのネットワークインターフェイスは AWS のサービスによって作成および管理され、ユーザーがリソースやサービスを使用できるようになります。これらのネットワークインターフェイスをユーザー自身が管理することはできません。詳細については「リクエスタマネージド型のネットワークインターフェイス」を参照してください。

プレフィックスの委任

プレフィックスとはインスタンスに関連付けられたネットワークインターフェイスに自動または手動で割り当てるために確保された、予約済みのプライベート IPv4 または IPv6 CIDR 範囲です。委任プレフィクスを使用すると、IP アドレスの範囲を単一のプレフィクスとして割り当てることで、サービスを迅速に起動できます。

マネージド型のネットワークインターフェイス

マネージドネットワークインターフェイスは、Amazon EKS Auto Mode などのサービスプロバイダーによって管理されます。管理されたネットワークインターフェイスの設定を直接変更することはできません。管理されたネットワークインターフェイスは 管理されたフィールドの真の値に

よって識別されます。詳細については「Amazon EC2 マネージドインスタンス」を参照してください。

ネットワークカード

ほとんどのインスタンスタイプは1 つのネットワークカードをサポートします。複数のネットワークカードをサポートするインスタンスタイプは100 Gbps を超える帯域幅機能やパケットレートのパフォーマンスの向上など、より高いネットワークパフォーマンスを提供します。複数のネットワークカードをサポートするインスタンスにネットワークインターフェイスをアタッチすると、ネットワークインターフェイスのネットワークカードを選択できます。プライマリネットワークインターフェースはネットワークカードインデックス 0 に割り当てる必要があります。

EFA および EFA のみのネットワークインターフェイスはネットワークインターフェイスとしてカウントされます。ネットワークカードごとに割り当てることができる EFA または EFA のみのネットワークインターフェイスは 1 つのみです。EFA のみのネットワークインターフェースを、プライマリネットワークインターフェースにすることはできません。

次のインスタンスタイプは複数のネットワークカードをサポートします。インスタンスタイプがサポートしているネットワークインターフェイスの数については「ネットワークインターフェイスあたりの IP アドレスの最大数」を参照してください。

インスタンスタイプ	ネットワークカードの数
c6in.32xlarge	2
c6in.metal	2
c8gb.48xlarge	2
c8gb.metal-48xl	2
c8gn.48xlarge	2
c8gn.metal-48xl	2
c8in.96xlarge	2
c8in.metal-96xl	2

インスタンスタイプ	ネットワークカードの数
c8ib.96xlarge	2
c8ib.metal-96xl	2
dl1.24xlarge	4
g6e.24xlarge	2
g6e.48xlarge	4
g7.48xlarge	2
g7e.24xlarge	2
g7e.48xlarge	4
hpc6id.32xlarge	2
hpc7a.12xlarge	2
hpc7a.24xlarge	2
hpc7a.48xlarge	2
hpc7a.96xlarge	2
hpc8a.96xlarge	2
m6idn.32xlarge	2
m6idn.metal	2
m6in.32xlarge	2
m6in.metal	2
m8gb.48xlarge	2
m8gb.metal-48xl	2

インスタンスタイプ	ネットワークカードの数
m8gn.48xlarge	2
m8gn.metal-48xl	2
m8in.96xlarge	2
m8in.metal-96xl	2
m8idn.96xlarge	2
m8idn.metal-96xl	2
m8ib.96xlarge	2
m8ib.metal-96xl	2
m8idb.96xlarge	2
m8idb.metal-96xl	2
p4d.24xlarge	4
p4de.24xlarge	4
p5.48xlarge	32
p5e.48xlarge	32
p5en.48xlarge	16
p6-b200.48xlarge	8
p6-b300.48xlarge	17
p6e-gb200.36xlarge	17
r8gb.48xlarge	2
r8gb.metal-48xl	2

インスタンスタイプ	ネットワークカードの数
r8gn.48xlarge	2
r8gn.metal-48xl	2
r8in.96xlarge	2
r8in.metal-96xl	2
r8idn.96xlarge	2
r8idn.metal-96xl	2
r8ib.96xlarge	2
r8ib.metal-96xl	2
r8idb.96xlarge	2
r8idb.metal-96xl	2
r6idn.32xlarge	2
r6idn.metal	2
r6in.32xlarge	2
r6in.metal	2
trn1.32xlarge	8
trn1n.32xlarge	16
trn2.48xlarge	16
trn2u.48xlarge	16
u7in-16tb.224xlarge	2
u7in-24tb.224xlarge	2

インスタンスタイプ	ネットワークカードの数
u7in-32tb.224xlarge	2
u7inh-32tb.480xlarge	2

ネットワークインターフェイスあたりの IP アドレスの最大数

各インスタンスタイプはネットワークインターフェイスの最大数、ネットワークインターフェイスあたりのプライベート IPv4 アドレスの最大数、ネットワークインターフェイスあたりの IPv6 アドレスの最大数をサポートしています。ネットワークインターフェイスあたりの IPv6 アドレスとプライベート IPv4 アドレスの制限は異なります。C1, M1, M2, M3 を除くすべてのインスタンスタイプが IPv6 アドレス指定をサポートしていることに注意してください。 T1

利用可能なネットワークインターフェイス

「Amazon EC2 Instance Types ガイド」には各インスタンスタイプで利用できるネットワークインターフェイスに関する情報が記載されています。詳細については次を参照してください:

- ネットワーク仕様 — 汎用
- ネットワーク仕様 — コンピューティング最適化
- ネットワーク仕様 — メモリ最適化
- ネットワーク仕様 — ストレージ最適化
- ネットワーク仕様 — 高速コンピューティング
- ネットワーク仕様 — ハイパフォーマンスコンピューティング
- ネットワーク仕様 — 旧世代

Console

最大ネットワークインターフェイスを取得するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンスタイプ]を選択します。
3. フィルターを追加して、インスタンスタイプ (Instance type=c5.12xlarge) またはインスタンスファミリー (Instance family=c5) を指定します。

4. (オプション) [設定] アイコンをクリックして、[ネットワークインターフェイスの最大数] をオンにします。この列には、各インスタンスタイプの、ネットワークインターフェイスの最大数が表示されています。
5. (オプション) インスタンスタイプを選択します。[ネットワーク] タブで、[ネットワークインターフェイスの最大数] を見つけます。

AWS CLI

最大ネットワークインターフェイスを取得するには

describe-instance-types コマンドを使用して、サポートされるネットワークインターフェイスやインターフェイスごとの IP アドレスなど、インスタンスタイプに関する情報を表示できます。次の例ではすべての C8i インスタンスでこれらの情報を表示します。

```
{ echo -e "InstanceType\tMaximumNetworkInterfaces\tIpv4AddressesPerInterface"; \
aws ec2 describe-instance-types \
    --filters "Name=instance-type,Values=c8i.*" \
    --query 'InstanceTypes[*].[InstanceType, NetworkInfo.MaximumNetworkInterfaces,
 NetworkInfo.Ipv4AddressesPerInterface]' \
    --output text | sort -k2 -n; } | column -t
```

以下は出力の例です。

```
InstanceType    MaximumNetworkInterfaces  Ipv4AddressesPerInterface
c8i.large       3                         20
c8i.2xlarge     4                         30
c8i.xlarge      4                         30
c8i.4xlarge     8                         50
c8i.8xlarge     10                        50
c8i.12xlarge    12                        50
c8i.16xlarge    16                        64
c8i.24xlarge    16                        64
c8i.32xlarge    24                        64
c8i.48xlarge    24                        64
c8i.96xlarge    24                        64
c8i.metal-48xl  24                        64
c8i.metal-96xl  24                        64
```

PowerShell

最大ネットワークインターフェイスを取得するには

Get-EC2InstanceType PowerShell コマンドを使用して、サポートされているネットワークインターフェイスやインターフェイスごとの IP アドレスなど、インスタンスタイプに関する情報を表示できます。次の例ではすべての C8i インスタンスでこれらの情報を表示します。

```
Get-EC2InstanceType -Filter @{Name="instance-type"; Values="c8i.*"} |
Select-Object `
    InstanceType,
    @{Name='MaximumNetworkInterfaces';
 Expression={$_.NetworkInfo.MaximumNetworkInterfaces}},
    @{Name='Ipv4AddressesPerInterface';
 Expression={$_.NetworkInfo.Ipv4AddressesPerInterface}} |
Sort-Object MaximumNetworkInterfaces |
Format-Table -AutoSize
```

以下は出力の例です。

```
InstanceType   MaximumNetworkInterfaces Ipv4AddressesPerInterface
------------   ------------------------ -------------------------
c8i.large                             3                        20
c8i.xlarge                            4                        30
c8i.2xlarge                           4                        30
c8i.4xlarge                           8                        50
c8i.8xlarge                          10                        50
c8i.12xlarge                         12                        50
c8i.24xlarge                         16                        64
c8i.16xlarge                         16                        64
c8i.96xlarge                         24                        64
c8i.48xlarge                         24                        64
c8i.metal-96xl                       24                        64
c8i.32xlarge                         24                        64
c8i.metal-48xl                       24                        64
```

EC2 インスタンス用のネットワークインターフェイスを作成する

EC2 インスタンスで使用するためのネットワークインターフェイスを作成できます。ネットワークインターフェイスを作成する際に、作成先のサブネットを指定します。ネットワークインターフェイスは作成後に別のサブネットに移動することはできません。ネットワークインターフェイスは同じアベイラビリティーゾーンのインスタンスにアタッチする必要があります。セカンダリネットワークインターフェイスをインスタンスからデタッチし、同じアベイラビリティーゾーンの別のインスタンスにアタッチできます。プライマリネットワークインターフェイスをインスタンスからデタッチするこ

とはできません。詳細については「the section called “ネットワークインターフェイスのアタッチ”」を参照してください。

Console

ネットワークインターフェイスを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Network Interfaces] を選択してください。
3. [ネットワークインターフェイスの作成] をクリックしてください。
4. (オプション) [説明] に分かりやすい名前を入力してください。
5. [Subnet (サブネット)] で、サブネットを選択してください。以降のステップで使用できるオプションは選択したサブネットのタイプ (IPv4 専用、IPv6 専用、またはデュアルスタック (IPv4 および IPv6)) によって異なります。
6. [インターフェイスタイプ] で、以下のいずれかを選択します。
   - ENA: CPU 使用率を最小限に抑えながら、TCP/IP プロトコルの高スループットとパケット毎秒のレートを処理するように設計された高性能ネットワークインターフェイス。これは、デフォルト値です。ENA の詳細については、「Elastic Network Adapter」を参照してください。
   - ENA を使用した EFA: ENA デバイスと EFA デバイス両方のサポートにより、従来の TCP/IP ベースのトランスポートと SRD ベースのトランスポートを提供するネットワークインターフェイス。ENA を使用した EFA を選択する場合、アタッチするインスタンスも EFA をサポートしている必要があります。EFA の詳細については、「Elastic Fabric Adapter」を参照してください。
   - EFA のみ: オペレーティングシステムスタックをバイパスしながら、SRD ベーストランスポートでの高スループットで低レイテンシーのノード間通信を処理するように設計された高性能ネットワークインターフェイス。このオプションを選択した場合、アタッチするインスタンスも EFA をサポートしている必要があります。EFA 専用ネットワークインターフェイスは IP アドレスをサポートしていません。EFA の詳細については、「Elastic Fabric Adapter」を参照してください。
7. [プライベート IPv4 アドレス] で、次のいずれかの操作を実行します。
   - サブネットからの IPv4 アドレスの選択を Amazon EC2 に許可するには[自動割り当て] を選択してください。
   - [カスタム] を選択した場合は続いてサブネットから選択した IPv4 アドレスを入力してください。

8. (IPv6 アドレスを持つサブネットのみ) [IPv6 アドレス] で、次のいずれかの操作を行います。
   - ネットワークインターフェイスに IPv6 アドレスを割り当てたくない場合は[なし] を選択してください。
   - サブネットからの IPv6 アドレスの選択を Amazon EC2 に許可するには[自動割り当て] を選択してください。
   - [カスタム] を選択した場合は続いてサブネットから選択した IPv6 アドレスを入力してください。
9. (オプション) デュアルスタックまたは IPv6 専用サブネットにネットワーク インターフェイスを作成している場合はプライマリ IPv6 IP の割り当てるオプションがあります。これにより、プライマリ IPv6 グローバルユニキャストアドレス (GUA) がネットワークインターフェイスに割り当てられます。プライマリ IPv6 アドレスを割り当てると、インスタンスまたは ENI へのトラフィックの中断を回避できます。この ENI が接続されるインスタンスが IPv6 アドレスが変更されないことに依存する場合は[有効化] を選択してください。 AWS はインスタンスにアタッチされている ENI に関連付けられた IPv6 アドレスをプライマリ IPv6 アドレスとして自動的に割り当てます。IPv6 GUA アドレスをプライマリ IPv6 として有効にすると、無効にすることはできません。IPv6 GUA アドレスをプライマリ IPv6 にすることを有効にすると、インスタンスが終了するか、ネットワークインターフェースがデタッチされるまで、最初の IPv6 GUA がプライマリ IPv6 アドレスになります。インスタンスに複数の IPv6 アドレスがアタッチされていて、プライマリ IPv6 アドレスを有効にすると、ENI に関連付けられた最初の IPv6 GUA アドレスがプライマリ IPv6 アドレスになります。
10. (オプション) Elastic Fabric Adapter を作成するにはElastic Fabric Adapter、[有効化] の順にクリックしてください。
11. (オプション) [詳細設定] で、オプションで IP プレフィックス委任を設定できます。詳細については「プレフィックスの委任」を参照してください。
    - 自動割り当て — AWS が VPC サブネットの IPv4 または IPv6 CIDR ブロックからプレフィクスを選択し、ネットワークインターフェイスに割り当てます。
    - カスタム — サブネットの IPv4 または IPv6 CIDR ブロックからプレフィクスを指定します。AWS は、プレフィックスをネットワーク インターフェイスに割り当てる前に、そのプレフィックスが他のリソースにまだ割り当てられていないことを確認します。
12. (オプション) [詳細設定] の [アイドル接続追跡タイムアウト] で、デフォルトのアイドル接続のタイムアウトを変更します。詳細については「アイドル接続追跡タイムアウト」を参照してください。
    - TCP 確立タイムアウト: 確立された状態のアイドル TCP 接続のタイムアウト (秒単位)。

- 最小: 60 秒
- 最大: 432000 秒
- デフォルト: Nitro v6 インスタンスタイプ (P6e-GB200 を除く) の場合は 350 秒、その他のすべてのインスタンスタイプ (P6e-GB200 を含む) の場合は 432000 秒。
- 推奨: 432000 秒未満

- UDP タイムアウト: 単一方向、または 1 つのリクエスト-レスポンストランザクションのみのトラフィックが発生した、アイドル状態にある UDP フローのタイムアウト (秒単位)。
  - 最小: 30 秒
  - 最大: 60 秒
  - デフォルト: 30 秒
- UDP ストリームタイムアウト: 複数のリクエスト-レスポンストランザクションが発生したストリームとして分類される、アイドル状態にある UDP フローのタイムアウト (秒単位)。
  - 最小: 60 秒
  - 最大: 180 秒
  - デフォルト: 180 秒

13. [Security groups] で、1 つまたは複数のセキュリティグループを選択してください。
14. (オプション) タグごとに [新しいタグを追加] をクリックし、タグキーとオプションのタグ値を入力してください。
15. [ネットワークインターフェイスの作成] をクリックしてください。

AWS CLI

例 1: Amazon EC2 で選択された IP アドレスを持つネットワークインターフェイスを作成するには

次の create-network-interface コマンドを使用します。こちらの例では、パブリック IPv4 アドレスと、Amazon EC2 が選択した IPv6 アドレスを持つネットワークインターフェイスを作成します。

```
aws ec2 create-network-interface \
    --subnet-id subnet-0abcdef1234567890 \
    --description "my dual-stack network interface" \
    --ipv6-address-count 1 \
```

```
    --groups sg-1234567890abcdef0
```

例 2: 特定の IP アドレスを持つネットワークインターフェイスを作成するには

次の create-network-interface コマンドを使用します。

```
aws ec2 create-network-interface \
    --subnet-id subnet-0abcdef1234567890 \
    --description "my dual-stack network interface" \
    --private-ip-address 10.251.50.12 \
    --ipv6-addresses 2001:db8::1234:5678:1.2.3.4 \
    --groups sg-1234567890abcdef0
```

例 3: セカンダリ IP アドレスの数でネットワークインターフェイスを作成するには

次の create-network-interface コマンドを使用します。こちらの例では、Amazon EC2 はプライマリ IP アドレスとセカンダリ IP アドレスの両方を選択します。

```
aws ec2 create-network-interface \
    --subnet-id subnet-0abcdef1234567890 \
    --description "my network interface" \
    --secondary-private-ip-address-count 2 \
    --groups sg-1234567890abcdef0
```

例 4: 特定のセカンダリ IP アドレスを持つネットワークインターフェイスを作成するには

次の create-network-interface コマンドを使用します。こちらの例では、プライマリ IP アドレスとセカンダリ IP アドレスを指定します。

```
aws ec2 create-network-interface \
    --subnet-id subnet-0abcdef1234567890 \
    --description "my network interface" \
    --private-ip-addresses PrivateIpAddress=10.0.1.30,Primary=true \
                           PrivateIpAddress=10.0.1.31,Primary=false
    --groups sg-1234567890abcdef0
```

PowerShell

例 1: Amazon EC2 で選択された IP アドレスを持つネットワークインターフェイスを作成するには

New-EC2NetworkInterface コマンドレットを使用します。こちらの例では、パブリック IPv4 アドレスと、Amazon EC2 が選択した IPv6 アドレスを持つネットワークインターフェイスを作成します。

```
New-EC2NetworkInterface `
    -SubnetId subnet-0abcdef1234567890 `
    -Description "my dual-stack network interface" `
    -Ipv6AddresCount 1 `
    -Group sg-1234567890abcdef0
```

例 2: 特定の IP アドレスを持つネットワークインターフェイスを作成するには

New-EC2NetworkInterface コマンドレットを使用します。

```
New-EC2NetworkInterface `
    -SubnetId subnet-0abcdef1234567890 `
    -Description "my dual-stack network interface" `
    -PrivateIpAddress 10.251.50.12 `
    -Ipv6Address $ipv6addr `
    -Group sg-1234567890abcdef0
```

IPv6 アドレスを次のように定義します。

```
$ipv6addr = New-Object Amazon.EC2.Model.InstanceIpv6Address
$ipv6addr1.Ipv6Address = "2001:db8::1234:5678:1.2.3.4"
```

例 3: セカンダリ IP アドレスの数でネットワークインターフェイスを作成するには

New-EC2NetworkInterface コマンドレットを使用します。こちらの例では、Amazon EC2 はプライマリ IP アドレスとセカンダリ IP アドレスの両方を選択します。

```
New-EC2NetworkInterface `
    -SubnetId subnet-0abcdef1234567890 `
    -Description "my network interface" `
    -SecondaryPrivateIpAddressCount 2 `
    -Group sg-1234567890abcdef0
```

例 4: 特定のセカンダリ IP アドレスを持つネットワークインターフェイスを作成するには

New-EC2NetworkInterface コマンドレットを使用します。こちらの例では、プライマリ IP アドレスとセカンダリ IP アドレスを指定します。

```
New-EC2NetworkInterface `
    -SubnetId subnet-0abcdef1234567890 `
    -Description "my network interface" `
    -PrivateIpAddresses @($primary, $secondary) `
    -Group sg-1234567890abcdef0
```

セカンダリアドレスを次のように定義します。

```
$primary = New-Object Amazon.EC2.Model.PrivateIpAddressSpecification
$primary.PrivateIpAddress = "10.0.1.30"
$primary.Primary = $true
$secondary = New-Object Amazon.EC2.Model.PrivateIpAddressSpecification
$secondary.PrivateIpAddress = "10.0.1.31"
$secondary.Primary = $false
```

EC2 インスタンスのネットワークインターフェイスのアタッチ

EC2 インスタンスがプライマリまたはセカンダリのネットワークインターフェイスとして使用するネットワークインターフェイスを作成できます。ネットワークインターフェイスがネットワークインターフェイスと同じアベイラビリティーゾーンにある場合、そのネットワークインターフェイスを EC2 インスタンスにアタッチする必要があります。インスタンスのインスタンスタイプによって、インスタンスにアタッチできるネットワークインターフェイスの数が決まります。詳細については「the section called “ネットワークインターフェイスあたりの IP アドレス数”」を参照してください。

考慮事項

- ネットワークインターフェイスはインスタンスの実行中、インスタンスの停止中、インスタンスの起動中にインスタンスにアタッチできます (それぞれ、ホットアタッチ、ウォームアタッチ、コールドアタッチと呼ばれています)。
- セカンダリネットワークインターフェイスはインスタンスの実行中または停止中にデタッチできます。ただし、プライマリネットワークインターフェイスをデタッチすることはできません。
- セカンダリネットワークインターフェイスをあるインスタンスからデタッチし、別のインスタンスにアタッチできます。
- CLI、API、または SDK を使用してインスタンスを起動する場合、プライマリネットワークインターフェイスおよび追加のネットワークインターフェイスを指定できます。起動時にセカンダリ

ネットワークインターフェイスを追加する場合、パブリック IPv4 アドレスの自動割り当てを有効にできません。

- 複数のネットワークインターフェイスを使用して Amazon Linux または Windows Server インスタンスを起動すると、インスタンスのオペレーティングシステム上でインターフェイス、プライベート IPv4 アドレス、ルートテーブルが自動的に設定されます。
- 追加ネットワークインターフェイスをウォームアタッチまたはホットアタッチする際、場合によっては手動で 2 つ目のインターフェイスを起動し、プライベート IPv4 アドレスを設定し、ルートテーブルを適宜変更する必要が生じます。Amazon Linux または Windows Server を実行するインスタンスはウォームアタッチまたはホットアタッチを自動的に認識し、それらのインスタンス自体を設定します。
- 別のネットワークインターフェイスをインスタンスにアタッチする (NIC チーミング設定など) ことで、デュアルホーム接続インスタンスとの間のネットワーク帯域幅を増加または倍増させることはできません。
- 同じサブネットから複数のネットワークインターフェイスをインスタンスにアタッチする場合、非対称ルーティングなどのネットワーク問題が発生する場合があります。可能であれば、代わりにプライマリネットワークインターフェイス上にセカンダリプライベート IPv4 アドレスを追加します。
- IPv6 専用サブネット内の EC2 インスタンスの場合、セカンダリネットワークインターフェイスをアタッチすると、セカンダリネットワークインターフェイスのプライベート DNS ホスト名はプライマリネットワークインターフェイスのプライマリ IPv6 アドレスに解決されます。
- [Windows インスタンス] – 複数のネットワークインターフェイスをインスタンスに追加する場合、静的ルーティングを使用するようにネットワークインターフェイスを設定する必要があります。

ネットワークインターフェイスのアタッチ

同じアベイラビリティーゾーンにあるインスタンスであれば、ネットワークインターフェイスをアタッチできます。アタッチするにはAmazon EC2 コンソールの [インスタンス] または [ネットワークインターフェイス] ページを開きます。またはインスタンスを起動する際に、既存のネットワークインターフェイスを指定することもできます。

インスタンスのパブリック IPv4 アドレスが解放される場合、複数のネットワークインターフェイスがそのインスタンスにアタッチされていると、インスタンスに新しいパブリック IP アドレスは送信されません。パブリック IPv4 アドレスの動作の詳細についてはパブリック IPv4 アドレスを参照してください。

Console

インスタンスページを使用してネットワークインターフェイスをインスタンスにアタッチするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスのチェックボックスをオンにします。
4. [Actions (アクション)]、[Networking (ネットワーク)]、[Attach Network Interface (ネットワークインターフェイスのアタッチ)] の順に選択してください。
5. VPC を選択してください。ネットワークインターフェイスがインスタンスと同じアベイラビリティゾーンにある場合に限り、ネットワークインターフェイスはインスタンスと同じ VPC または所有する別の VPC に所在することができます。これにより、ネットワークとセキュリティの設定が異なる VPC にまたがるマルチホームインスタンスを作成できます。
6. ネットワークインターフェイスを選択してください。インスタンスが複数のネットワークカードをサポートしている場合はネットワークカードを選択できます。
7. [アタッチ] を選択してください。

ネットワークインターフェイスページを使用してネットワークインターフェイスをアタッチするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Network Interfaces] を選択してください。
3. ネットワークインターフェイスのチェックボックスをオンにします。
4. [アクション]、[アタッチ] の順にクリックしてください。
5. インスタンスを選択してください。インスタンスが複数のネットワークカードをサポートしている場合はネットワークカードを選択できます。
6. [アタッチ] を選択してください。

AWS CLI

ネットワークインターフェイスをアタッチするには

次の attach-network-interface コマンドを使用します。

```
aws ec2 attach-network-interface \
```

```
    --network-interface-id eni-1234567890abcdef0 \
    --instance-id i-1234567890abcdef0 \
    --device-index 1
```

PowerShell

ネットワークインターフェイスをアタッチするには

Add-EC2NetworkInterface コマンドレットを使用します。

```
Add-EC2NetworkInterface `
    -NetworkInterfaceId eni-1234567890abcdef0 `
    -InstanceId i-1234567890abcdef0 `
    -DeviceIndex 1
```

ネットワークインターフェイスをデタッチする

ある時点で EC2 インスタンスにアタッチされているセカンダリネットワークインターフェイスは Amazon EC2 コンソールの [インスタンス] ページまたは [ネットワークインターフェイス] ページを使用して、いつでもデタッチできます。

Elastic Load Balancing ロードバランサー、Lambda 関数、WorkSpace、NAT ゲートウェイなど、別のサービスからリソースにアタッチされているネットワークインターフェイスをデタッチしようとすると、そのリソースに対するアクセス許可がないことを知らせるエラーが表示されます。ネットワークインターフェイスにアタッチされているリソースを作成したサービスを特定するにはそのネットワークインターフェイスの説明を確認します。リソースを削除すると、そのネットワークインターフェイスも削除されます。

Console

インスタンスページを使用してネットワークインターフェイスをデタッチするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスのチェックボックスをオンにします。[ネットワーキング] タブにある [ネットワークインタフェース] セクションを選択して、そのネットワークインターフェイスが、セカンダリネットワークインターフェイスとしてインスタンスにアタッチされていることを確認します。

4. [Actions (アクション)]、[Networking (ネットワーク)]、[Detach Network Interface (ネットワークインターフェイスのデタッチ)] の順に選択してください。
5. ネットワークインターフェイスを選択し、[デタッチ] を選択してください。

ネットワークインターフェイスページを使用してネットワークインターフェイスをデタッチするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Network Interfaces] を選択してください。
3. ネットワークインターフェイスのチェックボックスをオンにします。[詳細] タブの [インスタンスの詳細] セクションを選択し、そのネットワークインターフェイスが、セカンダリネットワークインターフェイスとしてインスタンスにアタッチされていることを確認します。
4. [アクション]、[デタッチ] の順にクリックしてください。
5. 確認を求められたら、[デタッチ] を選択してください。
6. ネットワークインターフェイスをインスタンスからデタッチできなかった場合は[強制デタッチ]、[有効化] の順にクリックし再試行します。強制デタッチは最終的手段としてのみご使用になることをお勧めします。デタッチを強制すると、インスタンスを再起動するまで、同じインデックスに別のネットワークインターフェイスをアタッチできなくなります。また、インスタンスを再起動するまで、ネットワークインターフェイスがデタッチされたことをインスタンスメタデータに反映しないようにできます。

AWS CLI

ネットワークインターフェースをデタッチするには

次の detach-network-interface コマンドを使用します。

```
aws ec2 detach-network-interface --attachment-id eni-attach-016c93267131892c9
```

PowerShell

ネットワークインターフェースをデタッチするには

Dismount-EC2NetworkInterface コマンドレットを使用します。

```
Dismount-EC2NetworkInterface -AttachmentId eni-attach-016c93267131892c9
```

ネットワークインターフェイスの IP アドレスを管理する

ネットワークインターフェースの次の IP アドレスを管理できます。

- Elastic IP アドレス (プライベート IPv4 アドレスごとに 1 つ)
- IPv4 アドレス
- IPv6 アドレス

Console

ネットワークインターフェイスの IP アドレスを管理するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Network Interfaces] を選択してください。
3. ネットワークインターフェイスのチェックボックスをオンにします。
4. IPv4 アドレスと IPv6 アドレスを管理するには、次の操作を行います。
   a. [アクション]、[IP アドレスの管理] の順にクリックしてください。
   b. ネットワークインターフェイスを展開します。
   c. [IPv4 アドレス] で、必要に応じて IP アドレスを修正します。追加の IPv4 アドレスを割り当てるには、[新しい IP アドレスの割り当て] をクリックし、サブネット範囲にある IPv4 アドレスを指定するか、AWS に自動選択させます。
   d. (デュアルスタックまたは IPv6 のみ) IPv6 アドレスについては、必要に応じて IP アドレスを修正します。追加の IPv6 アドレスを割り当てるには、[新しい IP アドレスの割り当て] をクリックし、サブネット範囲にある IPv6 アドレスを指定するか、AWS に自動選択させます。
   e. ネットワークインターフェイスにパブリック IPv4 アドレスを割り当てるまたは割り当て解除するには[パブリック IP の自動割り当て] を選択してください。こちらは任意のネットワークインターフェイスで有効または無効にできますが、影響が及ぶのはプライマリネットワークインターフェイスのみです。
   f. (デュアルスタックまたは IPv6 のみの場合) [プライマリ IPv6 IP の割り当て] で、[有効] を選択してプライマリ IPv6 IP アドレスを割り当てます。ネットワークインターフェイスに関連付けられた最初の GUA が、プライマリ IPv6 アドレスとして選択されます。プライマリ IPv6 アドレスを割り当てた後は、これを変更することはできません。インスタンスが終了するか、ネットワークインターフェイスがデタッチされるまで、このアドレスがプライマリ IPv6 アドレスとなります。

g. [保存] を選択します。

5. Elastic IP アドレスを関連付けるには次の操作を行います。

a. [アクション]、[アドレスの関連付け] の順にクリックしてください。

b. [Elastic IP アドレス] で、Elastic IP アドレスを選択してください。

c. [プライベート IPv4 アドレス] で、Elastic IP アドレスに関連付けるプライベート IPv4 アドレスを選択してください。

d. (オプション) 現在、ネットワークインターフェイスが別のインスタンスまたはネットワークインターフェイスに関連付けられている場合は[Elastic IP アドレスの再関連付けを許可する] をクリックしてください。

e. [Associate] を選択してください。

6. Elastic IP アドレスの関連付けを解除するには次の手順を実行します。

a. [Actions]、[Disassociate address] の順に選択してください。

b. [パブリック IP アドレス] で、Elastic IP アドレスを選択してください。

c. [関連付け解除] を選択してください。

AWS CLI

IPv4 アドレスを管理するには

IPv4 アドレスを割り当てるには、次の assign-private-ip-addresses コマンドを使用します。

```
aws ec2 assign-private-ip-addresses \
    --network-interface-id eni-1234567890abcdef0 \
    --private-ip-addresses 10.0.0.82
```

IPv4 アドレスの割り当てを解除するには、次の unassign-private-ip-addresses コマンドを使用します。

```
aws ec2 unassign-private-ip-addresses \
    --network-interface-id eni-1234567890abcdef0 \
    --private-ip-addresses 10.0.0.82
```

IPv6 アドレスを管理するには

IPv6 アドレスを割り当てるには、次の assign-ipv6-addresses コマンドを使用します。

```
aws ec2 assign-ipv6-addresses \
    --network-interface-id eni-1234567890abcdef0 \
    --ipv6-addresses 2001:db8:1234:1a00:9691:9503:25ad:1761
```

IPv6 アドレスの割り当てを解除するには、次の unassign-ipv6-addresses コマンドを使用します。

```
aws ec2 unassign-ipv6-addresses \
    --network-interface-id eni-1234567890abcdef0 \
    --ipv6-addresses 2001:db8:1234:1a00:9691:9503:25ad:1761
```

プライマリプライベート IPv4 アドレスの Elastic IP アドレスを管理するには

次の associate-address コマンドを使用して、Elastic IP アドレスをプライマリプライベート IPv4 アドレスに関連付けます。

```
aws ec2 associate-address \
    --allocation-id eipalloc-0b263919b6EXAMPLE \
    --network-interface-id eni-1234567890abcdef0
```

次の disassociate-address コマンドを使用して、Elastic IP アドレスとプライマリプライベート IPv4 アドレスの関連付けを解除します。

```
aws ec2 disassociate-address --association-id eipassoc-2bebb745a1EXAMPLE
```

PowerShell

IPv4 アドレスを管理するには

Register-EC2PrivateIpAddress コマンドレットを使用して、IPv4 アドレスを割り当てます。

```
Register-EC2PrivateIpAddress `
    -NetworkInterfaceId eni-1234567890abcdef0 `
    -PrivateIpAddress 10.0.0.82
```

Unregister-EC2PrivateIpAddress コマンドレットを使用して、IPv4 アドレスの割り当てを解除します。

```
Unregister-EC2PrivateIpAddress `
```

```
    -NetworkInterfaceId eni-1234567890abcdef0 `
    -PrivateIpAddress 10.0.0.82
```

IPv6 アドレスを管理するには

Register-EC2Ipv6AddressList コマンドレットを使用して、IPv6 アドレスを割り当てます。

```
Register-EC2Ipv6AddressList `
    -NetworkInterfaceId eni-1234567890abcdef0 `
    -Ipv6Address 2001:db8:1234:1a00:9691:9503:25ad:1761
```

Unregister-EC2Ipv6AddressList コマンドレットを使用して、IPv6 アドレスの割り当てを解除します。

```
Unregister-EC2Ipv6AddressList `
    -NetworkInterfaceId eni-1234567890abcdef0 `
    -Ipv6Address 2001:db8:1234:1a00:9691:9503:25ad:1761
```

プライマリプライベート IPv4 アドレスの Elastic IP アドレスを管理するには

Register-EC2Address コマンドレットを使用して、Elastic IP アドレスをプライマリプライベート IPv4 アドレスに関連付けます。

```
Register-EC2Address `
    -NetworkInterfaceId eni-1234567890abcdef0 `
    -AllocationId eipalloc-0b263919b6EXAMPLE
```

Unregister-EC2Address コマンドレットを使用して、Elastic IP アドレスとプライマリプライベート IPv4 アドレスの関連付けを解除します。

```
Unregister-EC2Address -AssociationId eipassoc-2bebb745a1EXAMPLE
```

ネットワークインターフェイス属性の変更

次のネットワークインターフェース属性を変更できます。

- 説明
- セキュリティグループ

- 終了時に削除
- 送信元/送信先チェック
- アイドル接続追跡タイムアウト

考慮事項

リクエスタマネージド型のネットワークインターフェイスの属性は、変更できません。

Console

ネットワークインターフェイスの属性を変更するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Network Interfaces] を選択してください。
3. ネットワークインターフェイスのチェックボックスをオンにします。
4. 説明を変更するには、以下を実行します。
   a. [アクション]、[説明の変更] の順にクリックしてください。
   b. [説明] に説明を入力します。
   c. [保存] を選択します。
5. セキュリティグループを変更するには、以下を実行します。
   a. [アクション]、[セキュリティグループの変更] の順にクリックしてください。
   b. [関連付けられたセキュリティグループ] で、必要に応じてセキュリティグループを追加および削除します。セキュリティグループとネットワークインターフェイスは同じ VPC に対して作成する必要があります。
   c. [保存] を選択します。
6. 終了動作を変更するには、以下を実行します。
   a. [アクション]、[終了時の動作を変更] の順にクリックしてください。
   b. [終了時に削除] を選択するか、解除して [有効] にします。
   c. [保存] を選択します。
7. 送信元/送信先チェックを変更するには、以下を実行します。
   a. [アクション]、[送信元/送信先チェックの変更] の順にクリックしてください。
   b. [送信元/送信先チェック] を選択するか、解除して [有効] にします。

c. [保存] を選択します。

8. [アイドル接続追跡タイムアウト] を変更するには、以下を実行します。

a. [アクション]、[アイドル接続追跡タイムアウトを変更] の順に選択します。

b. 必要に応じてタイムアウト値を変更します。詳細については「アイドル接続追跡タイムアウト」を参照してください。

- TCP 確立タイムアウト: 確立された状態のアイドル TCP 接続のタイムアウト (秒単位)。
  - 最小: 60 秒
  - 最大: 432000 秒
  - デフォルト: Nitro v6 インスタンスタイプ (P6e-GB200 を除く) の場合は 350 秒、その他のすべてのインスタンスタイプ (P6e-GB200 を含む) の場合は 432000 秒。
  - 推奨: 432000 秒未満
- UDP タイムアウト: 単一方向、または 1 つのリクエスト-レスポンストランザクションのみのトラフィックが発生した、アイドル状態にある UDP フローのタイムアウト (秒単位)。
  - 最小: 30 秒
  - 最大: 60 秒
  - デフォルト: 30 秒
- UDP ストリームタイムアウト: 複数のリクエスト-レスポンストランザクションが発生したストリームとして分類される、アイドル状態にある UDP フローのタイムアウト (秒単位)。
  - 最小: 60 秒
  - 最大: 180 秒
  - デフォルト: 180 秒

c. [保存] を選択します。

AWS CLI

Example例: 説明を変更するには

次の modify-network-interface-attribute コマンドを使用します。

```
aws ec2 modify-network-interface-attribute \
```

```
    --network-interface-id eni-1234567890abcdef0 \
    --description "my updated description"
```

Example例: セキュリティグループを変更するには

次の modify-network-interface-attribute コマンドを使用します。

```
aws ec2 modify-network-interface-attribute \
    --network-interface-id eni-1234567890abcdef0 \
    --groups sg-1234567890abcdef0
```

Example例: 終了動作を変更するには

次の modify-network-interface-attribute コマンドを使用します。

```
aws ec2 modify-network-interface-attribute \
    --network-interface-id eni-1234567890abcdef0 \
    --attachment AttachmentId=eni-attach-43348162abEXAMPLE,DeleteOnTermination=false
```

Example例: 送信元/送信先チェックを有効にするには

次の modify-network-interface-attribute コマンドを使用します。

```
aws ec2 modify-network-interface-attribute \
    --network-interface-id eni-1234567890abcdef0 \
    --source-dest-check
```

Example例: アイドル接続追跡タイムアウトを変更するには

次の modify-network-interface-attribute コマンドを使用します。詳細については「アイドル接続追跡タイムアウト」を参照してください。

```
aws ec2 modify-network-interface-attribute \
    --network-interface-id eni-1234567890abcdef0 \
    --connection-tracking-specification
 TcpEstablishedTimeout=172800,UdpStreamTimeout=90,UdpTimeout=60
```

PowerShell

Example例: 説明を変更するには

Edit-EC2NetworkInterfaceAttribute コマンドレットを使用します。

```
Edit-EC2NetworkInterfaceAttribute `
    -NetworkInterfaceId eni-1234567890abcdef0 `
    -Description "my updated description"
```

Example例: セキュリティグループを変更するには

Edit-EC2NetworkInterfaceAttribute コマンドレットを使用します。

```
Edit-EC2NetworkInterfaceAttribute `
    -NetworkInterfaceId eni-1234567890abcdef0 `
    -Group sg-1234567890abcdef0
```

Example例: 終了動作を変更するには

Edit-EC2NetworkInterfaceAttribute コマンドレットを使用します。

```
Edit-EC2NetworkInterfaceAttribute `
    -NetworkInterfaceId eni-1234567890abcdef0 `
    -Attachment_AttachmentId eni-attach-43348162abEXAMPLE `
    -Attachment_DeleteOnTermination $false
```

Example例: 送信元/送信先チェックを有効にするには

Edit-EC2NetworkInterfaceAttribute コマンドレットを使用します。

```
Edit-EC2NetworkInterfaceAttribute `
    -NetworkInterfaceId eni-1234567890abcdef0 `
    -SourceDestCheck $true
```

Example例: アイドル接続追跡タイムアウトを変更するには

Edit-EC2NetworkInterfaceAttribute コマンドレットを使用します。詳細については「アイドル接続追跡タイムアウト」を参照してください。

```
Edit-EC2NetworkInterfaceAttribute `
    -NetworkInterfaceId eni-1234567890abcdef0 `
    -ConnectionTrackingSpecification_TcpEstablishedTimeout 172800 `
    -ConnectionTrackingSpecification_UdpStreamTimeout 90 `
    -ConnectionTrackingSpecification_UdpTimeout 60
```

Amazon EC2 インスタンス用の複数のネットワークインターフェイス

以下が必要な場合、1 つのインスタンスに複数のネットワークインターフェイスをアタッチすると便利です。

- 管理ネットワーク。
- ネットワークアプライアンスとセキュリティアプライアンス。
- 異なるサブネットまたは VPC にワークロードを持つデュアルホーム接続インスタンス。
- 低予算で可用性の高いソリューション。

管理ネットワーク

次の概要では複数のネットワークインターフェイスを使用して作成された管理ネットワークについて説明します。

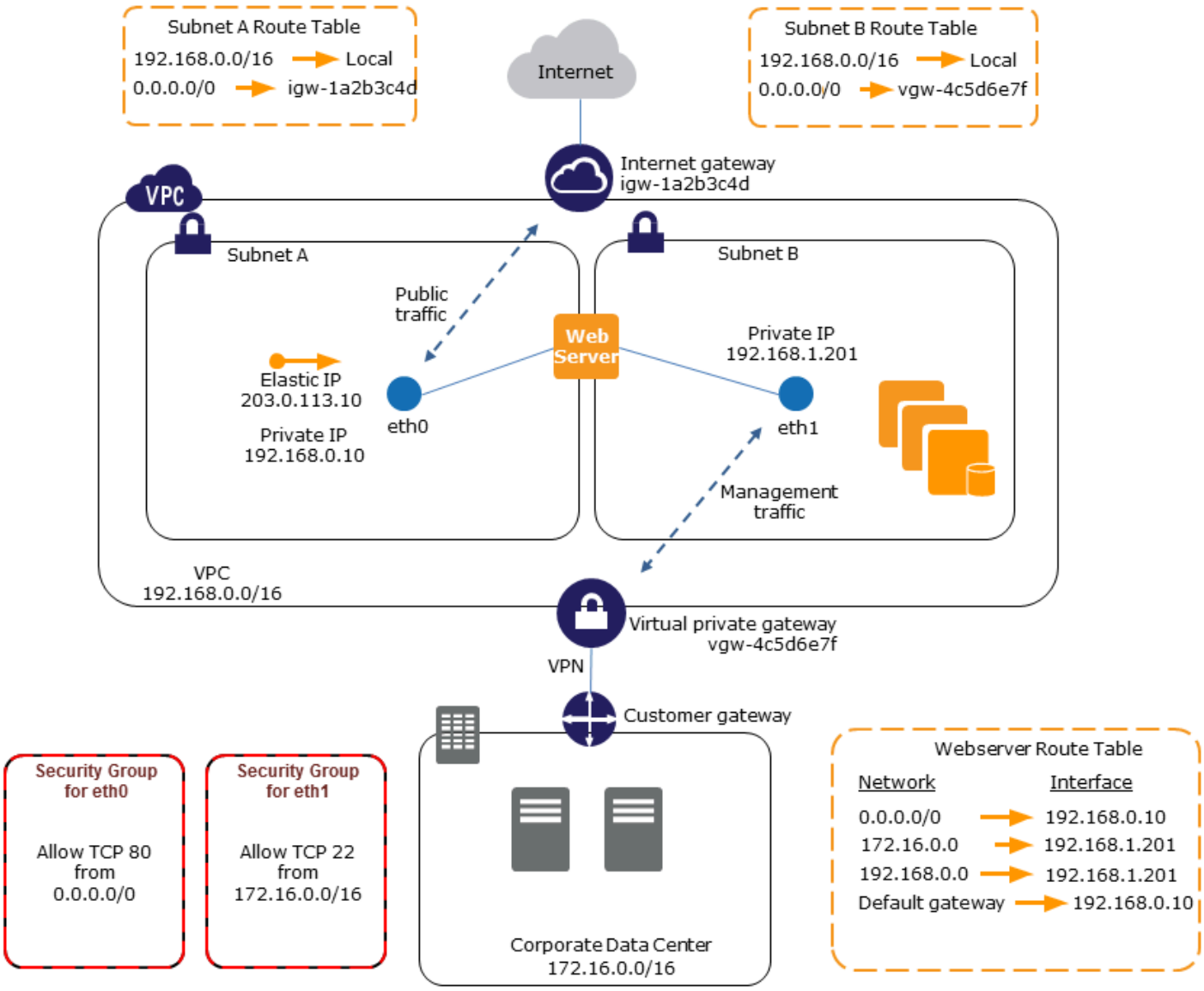
条件

- インスタンスのプライマリネットワークインターフェイス (例えば、eth0) がパブリックトラフィックを処理します。
- インスタンスのセカンダリネットワークインターフェイス (例えばeth1) がバックエンド管理トラフィックを処理します。より厳格なアクセス制御を使用した個別のサブネットに接続されており、プライマリネットワークインターフェイスと同じアベイラビリティーゾーン内にあります。

設定

- プライマリネットワークインターフェイスはロードバランサーの背後にある場合とそうでない場合があり、インターネットからのサーバーへのアクセスを許可するセキュリティグループが関連付けられています。例えば、0.0.0.0/0 またはロードバランサーからの TCP ポート 80 と 443 を許可します。
- セカンダリネットワークインターフェイスには次のいずれかの場所から開始された SSH アクセスのみを許可するセキュリティグループが関連付けられています。
  - VPC 内またはインターネット上の IP アドレスの許容範囲。
  - プライマリネットワークインターフェイスと同じアベイラビリティーゾーン内のプライベートサブネット。
  - 仮想プライベートゲートウェイ。

> **Note**
>
> フェイルオーバー機能が確実に動作するように、ネットワークインターフェイスの受信トラフィックに対してセカンダリプライベート IPv4 を使用することをお勧めします。インスタンスに障害が発生した場合はインターフェイスまたはセカンダリプライベート IPv4 アドレスあるいはその両方をスタンバイ用のインスタンスに移行できます。



ネットワークアプライアンスとセキュリティアプライアンス

ロードバランサー、ネットワークアドレス変換 (NAT) サーバー、プロキシサーバーなど、ネットワークアプライアンスやセキュリティアプライアンスの中には複数のネットワークインターフェイスを使用した構成が優先されるものがあります。セカンダリネットワークインターフェイスを作成して、これらのタイプのアプリケーションを実行するインスタンスにアタッチし、専用のパブリック IP アドレスとプライベート IP アドレス、セキュリティグループ、およびソース/デスティネーションチェックを使用するインターフェイスを、追加で構成することができます。

異なるサブネットにワークロードを持つデュアルホーム接続インスタンス

アプリケーションサーバーが存在するミッドティアネットワークに接続する Web サーバーのそれぞれにネットワークインターフェイスを置くことができます。このアプリケーションサーバーはデータベースサーバーが存在するバックエンドネットワーク (サブネット) にデュアルホーム接続することもできます。デュアルホーム接続されたインスタンスを介してネットワークパケットをルーティングする代わりに、デュアルホーム接続された各インスタンスはフロントエンドでリクエストを受信して処理し、バックエンドとの接続を開始して、バックエンドネットワーク上のサーバーにリクエストを送信します。

同じアカウントで異なる VPC にワークロードを持つデュアルホーム接続インスタンス

ネットワークインターフェイスがインスタンスと同じアベイラビリティーゾーンにある場合に限り、1 つの VPC で EC2 インスタンスを起動して別の VPC からセカンダリ ENI をアタッチできます。これにより、ネットワークとセキュリティの設定が異なる VPC にまたがるマルチホームインスタンスを作成できます。異なる AWS アカウントの VPC 間でマルチホームインスタンスを作成することはできません。

VPC にまたがるデュアルホームインスタンスは次のユースケースで使用できます。

- 相互にピアリングできない 2 つの VPC 間における CIDR の重複を克服: VPC のセカンダリ CIDR を活用して、重複しない 2 つの IP 範囲でインスタンスが通信できるようにします。
- 単一アカウント内で複数の VPC を接続: 通常は VPC の境界で区切られている個々のリソース間における通信を可能にします。

低予算で可用性の高いソリューション

特定の機能にサービスを提供しているインスタンスのいずれかが機能しなくなった場合はそのネットワークインターフェイスを同じ役割で構成された交換用またはホットスタンバイ用のインスタンスに

アタッチすることで、サービスを迅速に回復できます。例えば、データベースインスタンスや NAT インスタンスなどの重要なサービスに対するプライマリまたはセカンダリのネットワークインターフェイスとしてネットワークインターフェイスを使用することができます。そのインスタンスが機能しなくなった場合、お客様 (通常はお客様に代わって実行されるコード) がネットワークインターフェイスをホットスタンバイ用のインスタンスにアタッチすることができます。インターフェイスではプライベート IP アドレス、Elastic IP アドレス、および MAC アドレスがそのまま維持されるため、交換用のインスタンスにネットワークインターフェイスを接続するとすぐに、ネットワークトラフィックはスタンバイ用のインスタンスに流れ始めます。インスタンスに障害が発生してから、ネットワークインターフェイスがスタンバイ用のインスタンスにアタッチされるまで一時的な接続断が発生しますが、ルートテーブルや DNS サーバーに変更を加える必要はありません。

リクエスタマネージド型のネットワークインターフェイス

リクエスタマネージド型ネットワークインターフェイスはAWS のサービスがユーザーに代わって VPC 内に作成するネットワークインターフェイスです。ネットワークインターフェイスは、Amazon RDS の DB インスタンス、NAT ゲートウェイ、または AWS PrivateLink のインターフェイス VPC エンドポイントなど、別のサービスのリソースに関連付けられています。

考慮事項

- アカウントでリクエスタマネージドネットワークインターフェイスを表示し、タグを追加または削除できます。ただし、リクエスタマネージドネットワークインターフェイスでは、次のアクションを実行することはできません。
  - ネットワークインターフェイス属性の変更
  - ネットワークインターフェイスのアタッチ
  - Elastic IP アドレスを関連付けまたは関連付け解除
  - プライベート IP アドレスの割り当てまたは割り当て解除
  - インスタンスの起動時にネットワークインターフェイスを指定する

  ネットワークインターフェイス属性は、変更できなくてもリセットできます。
- リクエスタマネジード型ネットワークインターフェイスに関連付けられているリソースを削除すると、AWS のサービスはネットワークインターフェイスをデタッチして削除します。サービスがネットワークインターフェースをデタッチしたが、削除しなかった場合はデタッチされたネットワークインターフェイスを削除できます。

Console

リクエスタマネージド型のネットワークインターフェイスを表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Network & Security] (ネットワークとセキュリティ)、[Network Interfaces] (ネットワークインターフェイス) を選択してください。
3. ネットワークインターフェイスの ID を選択して、その詳細ページを開きます。
4. ネットワークインターフェイスの目的を特定するために使用できる主要なフィールドを次に示します。

   - [Description] (説明): インターフェイスを作成した AWS のサービスによって提供される説明。例えば、「VPC Endpoint Interface vpce 089f2123488812123」です。
   - Requester-managed：ネットワークインターフェイスが AWS によって管理されているかどうかを示します。
   - [Requester ID] (リクエスタ ID)：ネットワークインターフェイスを作成したプリンシパルまたはサービスのエイリアスまたはAWSアカウント ID ネットワークインターフェイスを作成した場合、これはユーザのAWS アカウントID です。それ以外の場合は別のプリンシパルまたはサービスによって作成されています。

AWS CLI

リクエスタマネージド型のネットワークインターフェイスを表示するには

次のように、describe-network-interfaces コマンドを使用します。

```
aws ec2 describe-network-interfaces \
    --filters "Name=requester-managed,Values=true" \
    --query "NetworkInterfaces[*].[Description, InterfaceType]" \
    --output table
```

ネットワークインターフェイス Description および InterfaceType の目的を決定するために使用できるキーフィールドを示す出力例を次に示します。

```
-------------------------------------------------------------------------------
|                          DescribeNetworkInterfaces                          |
+-----------------------------------------------+-------------------------+
|  VPC Endpoint Interface: vpce-0f00567fa8477a1e6   |  interface              |
```

```
|  VPC Endpoint Interface vpce-0d8ddce4be80e4474    |  interface               |
|  VPC Endpoint Interface vpce-078221a1e27d1ea5b    |  vpc_endpoint            |
|  Resource Gateway Interface rgw-0bba03f3d56060135 |  interface               |
|  VPC Endpoint Interface: vpce-0cc199f605eaeace7   |  interface               |
|  VPC Endpoint Interface vpce-019b90d6f16d4f958    |  interface               |
+---------------------------------------------------+--------------------------+
```

PowerShell

リクエスタマネージド型のネットワークインターフェイスを表示するには

次のように、Get-EC2NetworkInterface cmdlet を使用します。

```
Get-EC2NetworkInterface -Filter @{Name="requester-managed"; Values="true"} | Select
 Description, InterfaceType
```

以下は、ネットワークインターフェイス Description および InterfaceType の目的を特定する際に役立つキーフィールドを表示した出力の例です。

```
Description                                       InterfaceType
-----------                                       -------------
VPC Endpoint Interface: vpce-0f00567fa8477a1e6    interface
VPC Endpoint Interface vpce-0d8ddce4be80e4474     interface
VPC Endpoint Interface vpce-078221a1e27d1ea5b     vpc_endpoint
Resource Gateway Interface rgw-0bba03f3d56060135  interface
VPC Endpoint Interface: vpce-0cc199f605eaeace7    interface
VPC Endpoint Interface vpce-019b90d6f16d4f958     interface
```

Amazon EC2 ネットワークインターフェイスのプレフィックス委任

プライベート IPv4 または IPv6 CIDR 範囲は自動または手動で、ネットワークインターフェイスに割り当てることができます。プレフィクスを割り当てることにより、インスタンス上で複数の IP アドレスを必要とするコンテナおよびネットワークアプリケーションなど、アプリケーションの管理を拡張および簡素化します。IPv4 アドレスと IPv6 アドレスの詳細については「Amazon EC2 インスタンスの IP アドレス指定」を参照してください。

以下の割り当てオプションが利用できます。

- 自動割り当て — AWS がプレフィックスを選択し、ネットワークインターフェイスに割り当てます。ネットワークインターフェイスのサブネットにタイプ prefix のサブネット CIDR 予約があ

る場合、サブネット CIDR 予約からプレフィックスが選択されます。そうでない場合、サブネット CIDR 範囲から選択されます。

- 手動割り当て — ユーザーがプレフィックスを指定し、AWS はそのプレフィックスが他のリソースにまだ割り当てられていないことを確認してから、ネットワークインターフェイスに割り当てます。

プレフィクスの割り当てには次の利点があります。

- ネットワークインターフェイスの IP アドレスの増加 — プレフィクスを使用する場合は個々の IP アドレスではなく IP アドレスのブロックを割り当てます。これにより、ネットワークインターフェイスの IP アドレスの数が増加します。
- コンテナの VPC 管理の簡素化 — コンテナアプリケーションでは各コンテナに一意の IP アドレスが必要です。インスタンスにプレフィクスを割り当てることで、個々の IP 割り当てに対して Amazon EC2 API を呼び出すことなくコンテナを起動および終了できるため、VPC の管理が簡素化されます。

目次

- 基本
- 考慮事項
- ネットワークインターフェイスのプレフィックスを管理する
  - ネットワークインターフェイスの作成時にプレフィクスを割り当てる
  - 既存のネットワークインターフェイスにプレフィックスを割り当てる
  - ネットワークインターフェイスからプレフィクスを削除する

基本

- 新しいネットワークインターフェイスまたは既存のネットワークインターフェイスにプレフィクスを割り当てることができます。
- プレフィクスを使用するにはネットワークインターフェイスにプレフィクスを割り当て、次にそのネットワークインターフェイスをインスタンスにアタッチしてから、オペレーティングシステムを設定します。
- プレフィクスを指定するオプションを選択する場合、プレフィクスは次の要件を満たしている必要があります。
  - 指定できる IPv4 プレフィクスは /28。

  - 指定できる IPv6 プレフィクスは /80。
  - プレフィクスがネットワークインターフェイスのサブネット CIDR にあり、サブネット内の既存のリソースに割り当てられた他のプレフィクスまたは IP アドレスと重複していないこと。
- プレフィクスはプライマリまたはセカンダリのネットワークインターフェイスに割り当てることができます。
- プレフィクスが割り当てられているネットワークインターフェイスに Elastic IP アドレスを割り当てることができます。
- 割り当てられたプレフィクスの IP アドレス部分に Elastic IP アドレスを割り当てることもできます。
- インスタンスのプライベート DNS ホスト名をプライマリのプライベート IPv4 アドレスに解決します。
- プレフィクスからのものを含め、ネットワークインターフェイスの各プライベート IPv4 アドレスは次の形式を使用して割り当てます。
  - us-east-1 リージョン

```
ip-private-ipv4-address.ec2.internal
```

  - その他のすべてのリージョン

```
ip-private-ipv4-address.region.compute.internal
```

考慮事項

プレフィックスを使用する場合は次の点を考慮してください:

- プレフィックス付きのネットワークインターフェイスはNitro ベースのインスタンスでサポートされています。
- ネットワークインターフェイスのプレフィックスは、IPv6 アドレスまたはプライベート IPv4 アドレスに制限されます。
- ネットワークインターフェイスに割り当てることができる IP アドレスの数はインスタンスタイプによって異なります。ネットワークインターフェイスに割り当てる各プレフィクスが、1 つの IP アドレスとしてカウントされます。例えば、c5.large インスタンスで、ネットワークインターフェイスあたりの IPv4 アドレス数が 10 に制限されているとします。このインスタンスの各ネットワークインターフェイスに、プライマリ IPv4 アドレスが存在します。ネットワークインターフェイスにセカンダリ IPv4 アドレスがない場合はネットワークインターフェイスに最大 9 つのプ

レフィクスを割り当てることができます。ネットワークインターフェイスに割り当てる IPv4 アドレスを追加する度に、ネットワークインターフェイスに割り当てるプレフィクスの数が 1 つ少なくなります。詳細についてはネットワークインターフェイスあたりの IP アドレスの最大数 を参照してください。

- プレフィクスは送信元/送信先チェックに含まれます。
- プレフィクス付きのネットワークインターフェイス用にオペレーティングシステムを設定する必要があります。次の点に注意してください。
  - 一部の Amazon Linux AMI にはAWS によってインストールされる追加のスクリプト (`ec2-net-utils` と呼ばれる) が含まれます。これらのスクリプトはオプションで、ネットワークインターフェイスの設定を自動化します。これらは Amazon Linux 専用です。
  - コンテナの場合はKubernetes 用 Container Network Interface (CNI) プラグインを使用できます。またはDocker を使用してコンテナを管理している場合は`dockerd` を使用できます。

ネットワークインターフェイスのプレフィックスを管理する

ネットワークインターフェイスにプレフィックスを割り当てる際、プレフィックスを自動的に割り当てるか、カスタムプレフィックスを指定するかを選択できます。プレフィックスを自動的に割り当てて、ネットワークインターフェイスのサブネットにタイプ `prefix` のサブネット CIDR 予約がある場合、サブネッ CIDR 予約からプレフィックスが選択されます。そうでない場合、サブネット CIDR 範囲から選択されます。

タスク

- ネットワークインターフェイスの作成時にプレフィクスを割り当てる
- 既存のネットワークインターフェイスにプレフィックスを割り当てる
- ネットワークインターフェイスからプレフィクスを削除する

ネットワークインターフェイスの作成時にプレフィクスを割り当てる

ネットワークインターフェイスの作成時に、自動プレフィックスまたはカスタムプレフィックスを割り当てることができます。

Console

ネットワークインターフェース作成時に自動的にプレフィクスを割り当てるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

2. ナビゲーションペインで、[Network Interfaces] を選択してください。
3. [ネットワークインターフェイスの作成] をクリックしてください。
4. ネットワークインターフェイスの説明を入力し、ネットワークインターフェイスを作成するサブネットを選択し、プライベート IPv4 アドレスと IPv6 アドレスを設定します。
5. [詳細設定] を展開します。
6. [IPv4 プレフィックス委任] で、次のいずれかを実行します。
   - IPv4 プレフィックスを自動的に割り当てるには[自動割り当て] を選択してください。[IPv4 プレフィックスの数] で、割り当てるプレフィックスの数を入力してください。
   - 特定の IPv4 プレフィックスを割り当てるには[カスタム] を選択してください。[新しいプレフィックスの追加] を選択し、プレフィックスを入力してください。
7. [IPv6 プレフィックス委任] で、次のいずれかを実行します。
   - IPv6 プレフィックスを自動的に割り当てるには[自動割り当て] を選択してください。[IPv6 プレフィックスの数] で、割り当てるプレフィックスの数を入力してください。
   - 特定の IPv6 プレフィックスを割り当てるには[カスタム] を選択してください。[新しいプレフィックスの追加] を選択し、プレフィックスを入力してください。

   > **Note**
   >
   > [IPv6 プレフィックスの委任]は選択したサブネットがIPv6 に対して有効になっている場合にのみ表示されます。

8. ネットワークインターフェイスに関連付けるセキュリティグループを選択し、必要に応じてリソースタグを割り当てます。
9. [ネットワークインターフェイスの作成] をクリックしてください。

AWS CLI

ネットワークインターフェース作成時に自動的に IPv4 プレフィクスを割り当てるには

create-network-interface コマンドを使用して、AWS が割り当てる IPv4 プレフィックスの数を `--ipv4-prefix-count` に設定します。次の例ではAWS は 1 つの IPv4 プレフィクスを割り当てています。

```
aws ec2 create-network-interface \
    --subnet-id subnet-047cfed18eEXAMPLE \
```

```
    --description "IPv4 automatic example" \
    --ipv4-prefix-count 1
```

ネットワークインターフェース作成時に特定の IPv4 プレフィクスを割り当てるには

create-network-interface コマンドを使用し、--ipv4-prefixes にプレフィクスを設定します。AWS はこの範囲から IPv4 アドレスを選択してください。次の例ではプレフィクス CIDR は 10.0.0.208/28 です。

```
aws ec2 create-network-interface \
    --subnet-id subnet-047cfed18eEXAMPLE \
    --description "IPv4 manual example" \
    --ipv4-prefixes Ipv4Prefix=10.0.0.208/28
```

ネットワークインターフェース作成時に自動的に IPv6 プレフィクスを割り当てるには

create-network-interface コマンドを使用し、AWS が割り当てる IPv6 プレフィクスの数を --ipv6-prefix-count に設定します。次の例ではAWS は 1 つの IPv6 プレフィクスを割り当てています。

```
aws ec2 create-network-interface \
    --subnet-id subnet-047cfed18eEXAMPLE \
    --description "IPv6 automatic example" \
    --ipv6-prefix-count 1
```

ネットワークインターフェース作成時に特定の IPv6 プレフィクスを割り当てるには

create-network-interface コマンドを使用し、--ipv6-prefixes にプレフィクスを設定します。AWS はこの範囲から IPv6 アドレスを選択してください。次の例ではプレフィクス CIDR は 2600:1f13:fc2:a700:1768::/80 です。

```
aws ec2 create-network-interface \
    --subnet-id subnet-047cfed18eEXAMPLE \
    --description "IPv6 manual example" \
    --ipv6-prefixes Ipv6Prefix=2600:1f13:fc2:a700:1768::/80
```

PowerShell

ネットワークインターフェース作成時に自動的に IPv4 プレフィクスを割り当てるには

New-EC2NetworkInterface コマンドレットを使用して、AWS が割り当てる IPv4 プレフィックスの数を Ipv4PrefixCount に設定します。次の例ではAWS は 1 つの IPv4 プレフィクスを割り当てています。

```
New-EC2NetworkInterface `
    -SubnetId 'subnet-047cfed18eEXAMPLE' `
    -Description 'IPv4 automatic example' `
    -Ipv4PrefixCount 1
```

ネットワークインターフェース作成時に特定の IPv4 プレフィクスを割り当てるには

New-EC2NetworkInterface コマンドレットを使用して、Ipv4Prefix をプレフィックスに設定します。AWS はこの範囲から IPv4 アドレスを選択します。次の例ではプレフィクス CIDR は 10.0.0.208/28 です。

```
Import-Module AWS.Tools.EC2
New-EC2NetworkInterface `
    -SubnetId 'subnet-047cfed18eEXAMPLE' `
    -Description 'IPv4 manual example' `
    -Ipv4Prefix (New-Object `
        -TypeName Amazon.EC2.Model.Ipv4PrefixSpecificationRequest `
        -Property @{Ipv4Prefix = '10.0.0.208/28'})
```

ネットワークインターフェース作成時に自動的に IPv6 プレフィクスを割り当てるには

New-EC2NetworkInterface コマンドレットを使用して、AWS が割り当てる IPv6 プレフィックスの数を Ipv6PrefixCount に設定します。次の例ではAWS は 1 つの IPv6 プレフィクスを割り当てています。

```
New-EC2NetworkInterface `
    -SubnetId 'subnet-047cfed18eEXAMPLE' `
    -Description 'IPv6 automatic example' `
    -Ipv6PrefixCount 1
```

ネットワークインターフェース作成時に特定の IPv6 プレフィクスを割り当てるには

New-EC2NetworkInterface コマンドレットを使用して、Ipv6Prefixes をプレフィックスに設定します。AWS はこの範囲から IPv6 アドレスを選択します。次の例ではプレフィクス CIDR は 2600:1f13:fc2:a700:1768::/80 です。

```
Import-Module AWS.Tools.EC2
```

```
New-EC2NetworkInterface `
    -SubnetId 'subnet-047cfed18eEXAMPLE' `
    -Description 'IPv6 manual example' `
    -Ipv6Prefix (New-Object `
        -TypeName Amazon.EC2.Model.Ipv6PrefixSpecificationRequest `
        -Property @{Ipv6Prefix = '2600:1f13:fc2:a700:1768::/80'})
```

既存のネットワークインターフェイスにプレフィックスを割り当てる

既存のネットワークインターフェイスに、自動プレフィックスまたはカスタムプレフィクスを割り当てることができます。

Console

既存のネットワークインターフェイスに自動的にプレフィクスを割り当てるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[ネットワークインターフェイス] を選択してください。
3. プレフィクスを割り当てるネットワークインターフェイスを選択し、[アクション],[プレフィクスの管理]を選択してください。
4. [IPv4 プレフィックス委任] で、次のいずれかを実行します。
   - IPv4 プレフィックスを自動的に割り当てるには[自動割り当て] を選択してください。[IPv4 プレフィックスの数] で、割り当てるプレフィクスの数を入力してください。
   - 特定の IPv4 プレフィックスを割り当てるには[カスタム] を選択してください。[新しいプレフィックスの追加] を選択し、プレフィックスを入力してください。
5. [IPv6 プレフィックス委任] で、次のいずれかを実行します。
   - IPv6 プレフィクスを自動的に割り当てるには[自動割り当て] を選択してください。[IPv6 プレフィックスの数] で、割り当てるプレフィックスの数を入力してください。
   - 特定の IPv6 プレフィックスを割り当てるには[カスタム] を選択してください。[新しいプレフィックスの追加] を選択し、プレフィックスを入力してください。

   > **Note**
   >
   > [IPv6 プレフィクスの委任]はIPv6 に対して選択したサブネットが有効になっている場合にのみ表示されます。

6. [保存] を選択します。

AWS CLI

既存のネットワーク インターフェイスに IPv6 プレフィックスを割り当てるには、assign-ipv6-addresses コマンドを使用し、IPv4 プレフィックスを割り当てるには、assign-private-ip-addresses コマンドを使用します。

既存のネットワークインターフェイスに自動的に IPv4 プレフィクスを割り当てるには

assign-private-ip-addresses コマンドを使用し、AWS が割り当てる IPv4 プレフィクスの数を --ipv4-prefix-count に設定します。次の例ではAWS は 1 つの IPv4 プレフィクスを割り当てています。

```
aws ec2 assign-private-ip-addresses \
    --network-interface-id eni-081fbb4095EXAMPLE \
    --ipv4-prefix-count 1
```

既存のネットワークインターフェイスに特定の IPv4 プレフィックスを割り当てるには

assign-private-ip-addresses コマンドを使用し、--ipv4-prefixes にプレフィクスを設定します。AWS はこの範囲から IPv4 アドレスを選択してください。次の例ではプレフィクス CIDR は 10.0.0.208/28 です。

```
aws ec2 assign-private-ip-addresses \
    --network-interface-id eni-081fbb4095EXAMPLE \
    --ipv4-prefixes 10.0.0.208/28
```

既存のネットワークインターフェイスに自動的に IPv6 プレフィクスを割り当てるには

assign-ipv6-addresses コマンドを使用し、AWS が割り当てる IPv6 プレフィクスの数を --ipv6-prefix-count に設定します。次の例ではAWS は 1 つの IPv6 プレフィクスを割り当てています。

```
aws ec2 assign-ipv6-addresses \
    --network-interface-id eni-00d577338cEXAMPLE \
    --ipv6-prefix-count 1
```

既存のネットワークインターフェイスに特定の IPv6 プレフィクスを割り当てるには

assign-ipv6-addresses コマンドを使用し、--ipv6-prefixes にプレフィクスを設定します。AWS はこの範囲から IPv6 アドレスを選択してください。次の例ではプレフィクス CIDR は 2600:1f13:fc2:a700:18bb::/80 です。

```
aws ec2 assign-ipv6-addresses \
    --network-interface-id eni-00d577338cEXAMPLE \
    --ipv6-prefixes 2600:1f13:fc2:a700:18bb::/80
```

PowerShell

既存のネットワークインターフェイスに自動的に IPv4 プレフィクスを割り当てるには

Register-EC2PrivateIpAddress コマンドレットを使用し、 AWS が割り当てる IPv4 プレフィックスの数を Ipv4PrefixCount に設定します。次の例ではAWS は 1 つの IPv4 プレフィクスを割り当てています。

```
Register-EC2PrivateIpAddress `
    -NetworkInterfaceId 'eni-00d577338cEXAMPLE' `
    -Ipv4PrefixCount 1
```

既存のネットワークインターフェイスに特定の IPv4 プレフィックスを割り当てるには

Register-EC2PrivateIpAddress コマンドレットを使用して、Ipv4Prefix を プレフィックスに設定します。AWS はこの範囲から IPv4 アドレスを選択します。次の例ではプレフィクス CIDR は 10.0.0.208/28 です。

```
Register-EC2PrivateIpAddress `
    -NetworkInterfaceId 'eni-00d577338cEXAMPLE' `
    -Ipv4Prefix '10.0.0.208/28'
```

既存のネットワークインターフェイスに自動的に IPv6 プレフィクスを割り当てるには

Register-EC2Ipv6AddressList コマンドレットを使用し、AWS が割り当てる IPv4 プレフィックスの数を Ipv6PrefixCount に設定します。次の例ではAWS は 1 つの IPv6 プレフィクスを割り当てています。

```
Register-EC2Ipv6AddressList `
    -NetworkInterfaceId 'eni-00d577338cEXAMPLE' `
    -Ipv6PrefixCount 1
```

既存のネットワークインターフェイスに特定の IPv6 プレフィクスを割り当てるには

Register-EC2Ipv6AddressList コマンドレットを使用して、Ipv6Prefix をプレフィックスに設定します。AWS はこの範囲から IPv6 アドレスを選択します。次の例ではプレフィクス CIDR は 2600:1f13:fc2:a700:18bb::/80 です。

```
Register-EC2Ipv6AddressList `
    -NetworkInterfaceId 'eni-00d577338cEXAMPLE' `
    -Ipv6Prefix '2600:1f13:fc2:a700:18bb::/80'
```

ネットワークインターフェイスからプレフィクスを削除する

既存のネットワークインターフェイスからプレフィックスを削除できます。

Console

ネットワークインターフェイスからプレフィクスを削除するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Network Interfaces] を選択してください。
3. ネットワークインターフェイスを選択してください。
4. [アクション]、[プレフィックスを管理] を選択してください。
5. [IPv4 プレフィックス委任] で、特定のプレフィックスを削除するには削除するプレフィックスの横にある [割り当て解除] を選択してください。すべてのプレフィックスを削除するには[割り当てない] を選択してください。
6. [IPv6 プレフィックス委任] で、特定のプレフィックスを削除するには削除するプレフィックスの横にある [割り当て解除] を選択してください。すべてのプレフィックスを削除するには[割り当てない] を選択してください。

   > **Note**
   >
   > [IPv6 プレフィクスの委任]はIPv6 に対して選択したサブネットが有効になっている場合にのみ表示されます。

7. [Save] を選択してください。

AWS CLI

unassign-ipv6-addresses コマンドで IPv6 プレフィクスを削除し、unassign-private-ip-addresses コマンドで既存のネットワークインターフェイスから IPv4 プレフィクスを削除することができます。

ネットワークインターフェイスから IPv4 プレフィクスを削除するには

unassign-private-ip-addresses コマンドを使用して、削除するプレフィックス CIDR に `--ipv4-prefix` を設定します。

```
aws ec2 unassign-private-ip-addresses \
    --network-interface-id eni-081fbb4095EXAMPLE \
    --ipv4-prefixes 10.0.0.176/28
```

ネットワークインターフェイスから IPv6 プレフィクスを削除するには

unassign-ipv6-addresses コマンドを使用し、削除するプレフィックス CIDR に `--ipv6-prefix` を設定します。

```
aws ec2 unassign-ipv6-addresses \
    --network-interface-id eni-00d577338cEXAMPLE \
    --ipv6-prefix 2600:1f13:fc2:a700:18bb::/80
```

PowerShell

ネットワークインターフェイスから IPv4 プレフィクスを削除するには

Unregister-EC2PrivateIpAddress コマンドレットを使用して、削除するプレフィックス CIDR に `Ipv4Prefix` を設定します。

```
Unregister-EC2PrivateIpAddress `
    -NetworkInterfaceId 'eni-00d577338cEXAMPLE' `
    -Ipv4Prefix '10.0.0.208/28'
```

ネットワークインターフェイスから IPv6 プレフィクスを削除するには

Unregister-EC2Ipv6AddressList コマンドレットを使用して、削除するプレフィックス CIDR に `Ipv6Prefix` を設定します。

```
Unregister-EC2Ipv6AddressList `
    -NetworkInterfaceId 'eni-00d577338cEXAMPLE' `
    -Ipv6Prefix '2600:1f13:fc2:a700:18bb::/80'
```

ネットワークインターフェイスの削除

ネットワークインターフェイスを削除すると、そのインターフェイスに関連付けられているすべての属性がリリースされ、別のインスタンスで使用できるように、プライベート IP アドレスまたは Elastic IP アドレスがリリースされます。

使用中のネットワークインターフェイスを削除することはできません。先に、ネットワークインターフェイスをデタッチする必要があります。

Console

ネットワークインターフェイスを削除するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Network Interfaces] を選択してください。
3. ネットワークインターフェイスのチェックボックスをオンにし、[アクション]、[削除] の順にクリックしてください。
4. 確認を求めるメッセージが表示されたら、[削除] を選択してください。

AWS CLI

ネットワークインターフェイスを削除するには

次の delete-network-interface コマンドを使用します。

```
aws ec2 delete-network-interface --network-interface-id eni-1234567890abcdef0
```

PowerShell

ネットワークインターフェイスを削除するには

Remove-EC2NetworkInterface コマンドレットを使用します。

```
Remove-EC2NetworkInterface -NetworkInterfaceId eni-1234567890abcdef0
```

Amazon EC2 インスタンスのネットワーク帯域幅

インスタンスの帯域幅の仕様はインスタンスのインバウンドトラフィックとアウトバウンドトラフィックの両方に適用されます。例えば、インスタンスが最大 10 Gbps の帯域幅を指定している場

合、それはインバウンドトラフィックに対して最大 10 Gbps の帯域幅があり、同時にアウトバウンドトラフィックに対しても最大 10 Gbps の帯域幅があることを意味します。EC2 インスタンスで使用できるネットワーク帯域幅は次のようにいくつかの要因によって異なります。

マルチフロートラフィック

トラフィックがインターネットゲートウェイまたはローカルゲートウェイを通過する場合、マルチフロートラフィックに使用できる帯域幅はインスタンスタイプによって異なります。

インスタンスのタイプ	使用可能な帯域幅
vCPU が 32 未満のインスタンスタイプ	5 Gbps に制限
vCPU が 32 を超えるインスタンスタイプ	インスタンスタイプで使用可能な帯域幅の 50% に制限
C8in、C8ine、M8in、M8ine、M8idn、R8in、R8idn のインスタンス	インスタンスタイプのベースライン帯域幅に制限

シングルフロートラフィック

インスタンスが同じクラスタープレイスメントグループにない場合、シングルフロートラフィックの帯域幅は 5 Gbps に制限されます。レイテンシーを減らし、シングルフロー帯域幅を増やすには以下のいずれかをお試しください。

- クラスタープレイスメントグループを使用すると、同じプレイスメントグループ内のインスタンスで最大 10 Gbps の帯域幅を実現できます。
- または任意の 2 つのエンドポイント間に複数のパスを設定することで、Multipath TCP (MPTCP) を使用して高帯域幅を実現できます。
- 同じアベイラビリティーゾーン内の対象インスタンスに ENA Express を設定して、それらのインスタンス間で最大 25 Gbps を実現します。

> **Note**
>
> シングルフローは一意の 5 タプル TCP または UDP フローと見なされます。IP ヘッダーに続く他のプロトコル (GRE や IPsec など) では送信元 IP、送信先 IP、次のプロトコルの 3 つのタプルを使用してフローを定義します。

使用可能なインスタンスの帯域幅

使用可能なインスタンスのネットワーク帯域幅はその vCPU の数によって異なります。例えば、m5.8xlarge インスタンスには 32 個の vCPU と 10 Gbps のネットワーク帯域幅があり、m5.16xlarge インスタンスには 64 個の vCPU と 20 Gbps のネットワーク帯域幅があります。ただし、インスタンスがこの帯域幅を達成できない場合があります。例えば、インスタンスレベルでネットワーク許容量 (1 秒あたりのパケット数や追跡される接続数など) を超えた場合などです。トラフィックが使用できる帯域幅の量はvCPUsの数と宛先によって異なります。例えば、m5.16xlarge インスタンスは 64 vCPUs のため、リージョン内の別のインスタンスへのトラフィックは使用可能な全帯域幅 (20 Gbps) を利用できます。ただし、インターネットゲートウェイまたはローカルゲートウェイを通過するトラフィックは利用可能な帯域幅 (10 Gbps) の 50% しか利用できません。

通常、vCPU が 16 個以下のインスタンス (サイズ 4xlarge 以下) の場合、指定の帯域幅に「最大」と文書化されています。例えば、最大 10 Gbpsなどです。これらのインスタンスにはベースライン帯域幅があります。追加需要を満たすために、ネットワーク I/O クレジットメカニズムを使用して、ベースライン帯域幅を超えてバーストすることができます。インスタンスはインスタンスのサイズに応じて、バースト帯域幅を限られた期間 (通常は 5～60 分) 使用できます。

インスタンスは起動時にネットワーク I/O クレジットの最大数を受け取ります。インスタンスがネットワーク I/O クレジットを使い果たすと、ベースライン帯域幅に戻ります。実行中のインスタンスはベースライン帯域幅よりも少ないネットワーク帯域幅を使用するたびに、ネットワーク I/O クレジットを取得します。停止したインスタンスはネットワーク I/O クレジットを取得しません。バースト帯域幅は共有リソースであるため、インスタンスにクレジットが使用可能な場合でも、インスタンスのバーストはベストエフォートベースで行われます。

インバウンドトラフィックとアウトバウンドトラフィックには個別のネットワーク I/O クレジットバケットがあります。

ベースおよびバーストネットワークパフォーマンス

「Amazon EC2 Instance Types ガイド」では各インスタンスタイプのネットワークパフォーマンスに加え、バースト帯域幅を使用できるインスタンスで利用可能な、ベースラインネットワーク帯域幅について説明しています。詳細については次を参照してください:

- ネットワーク仕様 — 汎用
- ネットワーク仕様 — コンピューティング最適化
- ネットワーク仕様 — メモリ最適化
- ネットワーク仕様 — ストレージ最適化

- ネットワーク仕様 — 高速コンピューティング
- ネットワーク仕様 — ハイパフォーマンスコンピューティング
- ネットワーク仕様 — 旧世代

あるいはコマンドラインツールを使用して、この情報を取得することもできます。Amazon EC2 コンソールには、インスタンスタイプのベースラインネットワーク帯域幅は表示されません。

AWS CLI

describe-instance-types マンドを使用して、インスタンスタイプに関する情報を表示できます。次の例ではすべての C5 インスタンスのネットワークパフォーマンス情報を表示します。

```
aws ec2 describe-instance-types \
    --filters "Name=instance-type,Values=c5.*" \
    --query "InstanceTypes[].[InstanceType, NetworkInfo.NetworkPerformance,
 NetworkInfo.NetworkCards[0].BaselineBandwidthInGbps] | sort_by(@,&[2])" \
    --output table
```

以下は出力の例です。出力にベースライン帯域幅がない場合はAWS CLI の最新バージョンに更新します。

```
----------------------------------------------
|           DescribeInstanceTypes            |
+--------------+---------------------+-------+
|  c5.large    |  Up to 10 Gigabit   |  0.75 |
|  c5.xlarge   |  Up to 10 Gigabit   |  1.25 |
|  c5.2xlarge  |  Up to 10 Gigabit   |  2.5  |
|  c5.4xlarge  |  Up to 10 Gigabit   |  5.0  |
|  c5.9xlarge  |  12 Gigabit         |  12.0 |
|  c5.12xlarge |  12 Gigabit         |  12.0 |
|  c5.18xlarge |  25 Gigabit         |  25.0 |
|  c5.24xlarge |  25 Gigabit         |  25.0 |
|  c5.metal    |  25 Gigabit         |  25.0 |
+--------------+---------------------+-------+
```

PowerShell

Get-EC2InstanceType PowerShell コマンドを使用して、インスタンスタイプに関する情報を表示できます。次の例ではすべての C5 インスタンスのネットワークパフォーマンス情報を表示します。

```
Get-EC2InstanceType -Filter @{Name = "instance-type"; Values = "c5.*" } | `
    Select-Object `
    InstanceType,
    @{Name = 'NetworkPerformance'; Expression =
 {($_.Networkinfo.NetworkCards.NetworkPerformance)}},
    @{Name = 'BaselineBandwidthInGbps'; Expression =
 {($_.Networkinfo.NetworkCards.BaselineBandwidthInGbps)}} | `
Format-Table -AutoSize
```

以下は出力の例です。

```
InstanceType NetworkPerformance BaselineBandwidthInGbps
------------ ------------------ ----------------------
c5.4xlarge   Up to 10 Gigabit                     5.00
c5.xlarge    Up to 10 Gigabit                     1.25
c5.12xlarge  12 Gigabit                          12.00
c5.9xlarge   12 Gigabit                          12.00
c5.24xlarge  25 Gigabit                          25.00
c5.metal     25 Gigabit                          25.00
c5.2xlarge   Up to 10 Gigabit                     2.50
c5.large     Up to 10 Gigabit                     0.75
c5.18xlarge  25 Gigabit                          25.00
```

EC2 インスタンスの帯域幅の重み設定

一部のインスタンスタイプはネットワーク処理または EBS オペレーションを優先するベースライン帯域幅の重み付けを選択できる、設定可能な帯域幅の重み付けをサポートしています。ベースライン帯域幅のデフォルト設定はインスタンスタイプによって決まります。起動時に帯域幅の重み付けを設定することも、次の重み設定を使用してインスタンス設定を変更することもできます。

- デフォルト – このオプションはインスタンスタイプの標準帯域幅設定を使用します。
- vpc-1 – このオプションはネットワークに使用できるベースライン帯域幅を増やし、EBS オペレーションのベースライン帯域幅を減らします。
- ebs-1 – このオプションはEBS オペレーションで使用できるベースライン帯域幅を増やし、ネットワーク用のベースライン帯域幅を減らします。

帯域幅の重み付けに関する考慮事項

以下は帯域幅の重み付け戦略に影響する可能性のある考慮事項です。

- 帯域幅の重み付けの設定は帯域幅の仕様にのみ影響します。1 秒あたりのネットワークパケット (PPS) と EBS の 1 秒あたりの入出力オペレーション (IOPS) の仕様は変更されません。
- ネットワークと EBS 間の帯域幅の仕様は変更されません。帯域幅の重み付け設定を選択すると、選択したオプションで使用可能なベースライン帯域幅が増加し、残りのオプションのベースライン帯域幅は同じ絶対量だけ減少します。Flex インスタンス以外のすべてのインスタンスでは、選択したオプションで使用可能なバースト帯域幅は変わりません。残りのオプションでは縮小されます。4xlarge までの Flex インスタンスの場合、バースト帯域幅は変更されません。Flex インスタンス 8xlarge 以上の場合、バースト帯域幅はベースライン帯域幅と同じ量だけ増加します。
- 帯域幅割り当ての変更が EBS の I/O パフォーマンスにどのように影響するかを理解することが重要です。vpc-1 設定 (ネットワーク帯域幅の増加) を持つ EC2 インスタンスではIOPS 制限に達する前に EBS 帯域幅制限に達すると、EBS ボリュームの IOPS が低下する可能性があります。これはI/O サイズが大きいほど顕著です。

  例えば、通常 16 KiB の I/O サイズで 240,000 IOPS をサポートするインスタンスタイプではvpc-1重み付けを選択すると、調整された EBS ベースライン帯域幅制限が原因で達成可能な IOPS が減少する可能性があります。

  ワークロードを計画するときはI/O のサイズとパターンを考慮してください。I/O サイズが小さいほど帯域幅制限の影響を受けにくくなりますが、I/O サイズやシーケンシャルワークロードが大きいほど、帯域幅の変化による影響が大きくなる可能性があります。選択した設定で最適なパフォーマンスを確保するために、常に特定のワークロードをテストしてください。
- インターネットゲートウェイまたはローカルゲートウェイを通過するトラフィックのネットワークマルチフロー帯域幅仕様は設定済みオプションのベースライン帯域幅の 50%、または該当する場合は 5 Gbps に調整されます。詳細については「Amazon EC2 インスタンスのネットワーク帯域幅」を参照してください。

  次の例はデフォルトのベースライン帯域幅が 40 Gbps、デフォルトの境界帯域幅が 20 Gbps のインスタンスタイプに基づいています。このインスタンスのvpc-1帯域幅の重み付けを選択すると、重み付けされたベースライン帯域幅は 50 Gbps に、境界帯域幅は 25 Gbps に変わります。
- この機能はEC2 インスタンスの可用性とサポートに合わせて、すべての商用リージョンで使用できます。
- この機能はEC2 インスタンスに追加のコストはかかりません。

帯域幅の重み付けでサポートされているインスタンスタイプ

次のインスタンスタイプファミリーのインスタンスタイプは、設定可能な帯域幅の重み付けをサポートしています。

- 汎用: M8a、M8g、M8gd、M8i、M8id、M8i-flex、M9g、M9gd
- コンピューティング最適化: C8a、C8g、C8gd、C8i、C8id、C8i-flex、C9g、C9gd
- メモリ最適化: R8a、R8g、R8gd、R8i、R8id、R8i-flex、X8g、X8aedz、X8i

現在の帯域幅設定を確認する

インスタンスの現在の帯域幅設定を確認するにはいずれかのタブを選択して手順を確認してください。

Console

インスタンスの帯域幅設定を取得するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. リストからチェックするインスタンスを選択し、ネットワークタブに移動します。現在の設定は設定済み帯域幅フィールドに表示されます。帯域幅が特定の値に設定されていない場合、Amazon EC2 はインスタンスタイプのデフォルト設定を使用します。

AWS CLI

インスタンスの帯域幅設定を取得するには

describe-instances コマンドを使用します。

```
aws ec2 describe-instances \
    --instance-ids i-1234567890abcdef0 \
    --query Reservations[].Instances[].NetworkPerformanceOptions.BandwidthWeighting
 \
    --output text
```

以下は出力の例です。

```
default
```

この例では、帯域幅をさらに増やすため、帯域幅の重み設定が vpc-1 に設定されているすべてのインスタンスを一覧表示します。

```
aws ec2 describe-instances \
    --filters "Name=network-performance-options.bandwidth-weighting,Values=vpc-1" \
    --query Reservations[].Instances[].InstanceId \
    --output text
```

PowerShell

インスタンスの帯域幅設定を取得するには

Get-EC2Instance コマンドレットを使用します。

```
(Get-EC2Instance `
    -
InstanceId i-1234567890abcdef0).Instances.NetworkPerformanceOptions.BandwidthWeighting.Value
```

以下は出力の例です。

```
default
```

この例では、帯域幅をさらに増やすため、帯域幅の重み設定が vpc-1 に設定されているすべてのインスタンスを一覧表示します。

```
(Get-EC2Instance `
    -Filter @{Name="network-performance-options.bandwidth-
weighting";Values="vpc-1"}).Instances.InstanceId
```

インスタンスの帯域幅の重み付けを設定する

帯域幅の重み付けは起動時に設定するか、EC2 コンソール、API/SDKs、または CLI から既存のインスタンスを変更することで設定できます。

インスタンスの起動時に帯域幅の重み付けを設定する

インスタンスの起動時に帯域幅設定を構成するにはいずれかのタブを選択して手順を確認してください。

起動テンプレートで帯域幅の重み付けを指定することもできます。起動テンプレートを作成するにはAmazon EC2 起動テンプレートを作成する 設定するパラメータはコンソールから直接インスタンスを起動する場合と同じ場所にあります。詳細セクションを展開し、インスタンス帯域幅設定を設定します。

起動テンプレートを使用してインスタンスを起動するには起動テンプレートを使用した EC2 インスタンスの起動を参照してください。

Console

設定可能な帯域幅の重み付けを使用してインスタンスを起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. [Launch Instances] (インスタンスの起動) を選択してください。これにより、[インスタンスを起動する] ページが開きます。設定によりますが、起動ダイアログにアクセスする方法は他にいくつかあります。例えば、AMI から直接、または Amazon EC2 ダッシュボード自体からインスタンスを起動できます。
4. から起動する Amazon マシンイメージ (AMI) はArmアーキテクチャに基づいている必要があります。多くのクイックスタートイメージは x86 と Arm の両方のアーキテクチャをサポートしています。インスタンスのオペレーティングシステムを選択したら、[アーキテクチャ] リストから Arm オプションを選択してください。
5. インスタンスタイプはこの機能サポートされるインスタンスタイプの の 1 つである必要があります。
6. 詳細セクションを展開すると、下にスクロールしてインスタンス帯域幅の設定を見つけることができます。インスタンスの帯域幅設定オプションを選択してください。
7. インスタンスの他のすべての設定を通常どおりに設定し、インスタンスの起動を選択してください。

AWS CLI

設定可能な帯域幅の重み付けを使用してインスタンスを起動するには

次の run-instances コマンドを次のオプションで使用して、ネットワーク帯域幅をさらに増やすように設定されたインスタンスを起動します。

```
--network-performance-options BandwidthWeighting=vpc-1
```

次の run-instances コマンドを次のオプションで使用して、EBS 帯域幅をさらに増やすように設定されたインスタンスを起動します。

```
--network-performance-options BandwidthWeighting=ebs-1
```

PowerShell

設定可能な帯域幅の重み付けを使用してインスタンスを起動するには

次の New-EC2Instance コマンドレットを次のパラメータと共に使用して、ネットワーク帯域幅の重み付けをさらに増やすように設定されたインスタンスを起動します。

```
-NetworkPerformanceOptions_BandwidthWeighting vpc-1
```

次の New-EC2Instance コマンドレットを次のパラメータと共に使用して、EBS 帯域幅の重み付けをさらに増やすように設定されたインスタンスを起動します。

```
-NetworkPerformanceOptions_BandwidthWeighting ebs-1
```

既存のインスタンスの帯域幅の重み付けを更新する

既存のインスタンスの帯域幅の重み付けを更新するにはインスタンスが Stopped状態である必要があります。

Console

帯域幅の重み付けを更新するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. 更新するインスタンスをリストから選択してください。
4. 帯域幅設定を変更する前に、インスタンスが Stopped状態になっている必要があります。インスタンスが実行されている場合はインスタンス状態メニューからインスタンスを停止を選択してください。
5. アクション > ネットワークメニューから帯域幅の管理を選択してください。帯域幅の管理ダイアログが開きます。

> **Note**
>
> インスタンスタイプが帯域幅の重み付けの設定をサポートしていない場合、そのメニュー項目は無効になります。

6. インスタンスを更新するオプションを選択し、変更を選択して設定を保存します。

AWS CLI

帯域幅の重み付けを更新するには

指定したインスタンスのネットワーク帯域幅の重み付けをさらに増やすには、modify-instance-network-performance-options コマンドを使用します。

```
aws ec2 modify-instance-network-performance-options \
    --instance-id i-1234567890abcdef0 \
    --bandwidth-weighting=vpc-1
```

次の の例では指定されたインスタンスに対してより高い EBS 帯域幅の重み付けを設定します。

```
aws ec2 modify-instance-network-performance-options \
    --instance-id i-1234567890abcdef0 \
    --bandwidth-weighting=ebs-1
```

PowerShell

帯域幅の重み付けを更新するには

指定したインスタンスのネットワーク帯域幅の重み付けをさらに増やすには、Edit-EC2InstanceNetworkPerformanceOption コマンドレットを使用します。

```
Edit-EC2InstanceNetworkPerformanceOption `
    -InstanceId i-1234567890abcdef0 `
    -BandwidthWeighting vpc-1
```

次の の例では指定されたインスタンスに対してより高い EBS 帯域幅の重み付けを設定します。

```
Edit-EC2InstanceNetworkPerformanceOption `
    -InstanceId i-1234567890abcdef0 `
    -BandwidthWeighting ebs-1
```

ネットワークパフォーマンス

次の表では、default、vpc-1、ebs-1 の設定で達成できるネットワークパフォーマンスを Gbps で示しています。

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
c8a.medium	0.52 / 12.5	0.65 / 12.5	0.438 / 10.0
c8a.large	0.937/12.5	1.172 / 12.5	0.774 / 10.0
c8a.xlarge	1.875 / 12.5	2.344 / 12.5	1.562 / 10.0
c8a.2xlarge	3.75 / 15.0	4.688 / 15.0	3.125 / 12.5
c8a.4xlarge	7.5/15.0	9.375 / 15.0	6.25 / 12.5
c8a.8xlarge	15.0	18.75	12.5
c8a.12xlarge	22.5	28.125	18.75
c8a.16xlarge	30.0	37.5	25.0
c8a.24xlarge	40.0	50.0	32.5
c8a.48xlarge	75.0	93.75	60.0
c8a.metal-24xl	40.0	50.0	32.5
c8a.metal-48xl	75.0	93.75	60.0
c8g.medium	0.52 / 12.5	0.65 / 12.5	0.441 / 10.0
c8g.large	0.937 / 12.5	1.171 / 12.5	0.779 / 10.0
c8g.xlarge	1.875 / 12.5	2.344 / 12.5	1.562 / 10.0
c8g.2xlarge	3.75 / 15.0	4.688 / 15.0	3.125 / 12.5

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
c8g.4xlarge	7.5 / 15.0	9.375 / 15.0	6.25 / 12.5
c8g.8xlarge	15.0	18.75	12.5
c8g.12xlarge	22.5	28.125	18.75
c8g.16xlarge	30.0	37.5	25.0
c8g.24xlarge	40.0	50.0	32.5
c8g.48xlarge	50.0	62.5	40.0
c8g.metal-24xl	40.0	50.0	32.5
c8g.metal-48xl	50.0	62.5	40.0
c8gd.medium	0.52 / 12.5	0.65 / 12.5	0.441 / 10.0
c8gd.large	0.937 / 12.5	1.171 / 12.5	0.779 / 10.0
c8gd.xlarge	1.875 / 12.5	2.344 / 12.5	1.562 / 10.0
c8gd.2xlarge	3.75 / 15.0	4.688 / 15.0	3.125 / 12.5
c8gd.4xlarge	7.5 / 15.0	9.375 / 15.0	6.25 / 12.5
c8gd.8xlarge	15.0	18.75	12.5
c8gd.12xlarge	22.5	28.125	18.75
c8gd.16xlarge	30.0	37.5	25.0
c8gd.24xlarge	40.0	50.0	32.5
c8gd.48xlarge	50.0	62.5	40.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
c8gd.metal-24xl	40.0	50.0	32.5
c8gd.metal-48xl	50.0	62.5	40.0
c8i.large	0.937/12.5	1.172 / 12.5	0.774 / 10.0
c8i.xlarge	1.875 / 12.5	2.344 / 12.5	1.562 / 10.0
c8i.2xlarge	3.75 / 15.0	4.688 / 15.0	3.125 / 12.5
c8i.4xlarge	7.5/15.0	9.375 / 15.0	6.25 / 12.5
c8i.8xlarge	15.0	18.75	12.5
c8i.12xlarge	22.5	28.125	18.75
c8i.16xlarge	30.0	37.5	25.0
c8i.24xlarge	40.0	50.0	32.5
c8i.32xlarge	50.0	62.5	40.0
c8i.48xlarge	75.0	93.75	60.0
c8i.96xlarge	100.0	125.0	80.0
c8i.metal-48xl	75.0	93.75	60.0
c8i.metal-96xl	100.0	125.0	80.0
c8id.large	0.937/12.5	1.172 / 12.5	0.774 / 10.0
c8id.xlarge	1.875 / 12.5	2.344 / 12.5	1.562 / 10.0
c8id.2xlarge	3.75 / 15.0	4.688 / 15.0	3.125 / 12.5

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
c8id.4xlarge	7.5/15.0	9.375 / 15.0	6.25 / 12.5
c8id.8xlarge	15.0	18.75	12.5
c8id.12xlarge	22.5	28.125	18.75
c8id.16xlarge	30.0	37.5	25.0
c8id.24xlarge	40.0	50.0	32.5
c8id.32xlarge	50.0	62.5	40.0
c8id.48xlarge	75.0	93.75	60.0
c8id.96xlarge	100.0	125.0	80.0
c8id.metal-48xl	75.0	93.75	60.0
c8id.metal-96xl	100.0	125.0	80.0
c8i-flex.large	0.468 / 12.5	0.585 / 12.5	0.389 / 10.0
c8i-flex.xlarge	0.937/12.5	1.172 / 12.5	0.779 / 10.0
c8i-flex.2xlarge	1.875 / 15.0	2.344 / 15.0	1.562 / 12.5
c8i-flex.4xlarge	3.75 / 15.0	4.688 / 15.0	3.125 / 12.5
c8i-flex.8xlarge	7.5/15.0	9.375 / 18.75	6.25 / 12.5
c8i-flex.12xlarge	11.25 / 22.5	14.063 / 28.125	9.375 / 18.75
c8i-flex.16xlarge	15.0 / 30.0	18.75 / 37.5	12.5 / 25.0
c9g.medium	0.55 / 15.0	0.688 / 15.0	0.455 / 12.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
c9g.large	1.0 / 15.0	1.25 / 15.0	0.81 / 12.0
c9g.xlarge	2.1 / 15.0	2.625 / 15.0	1.725 / 12.0
c9g.2xlarge	4.25 / 17.0	5.313 / 17.0	3.5 / 14.0
c9g.4xlarge	8.5 / 17.0	10.625 / 17.0	7.0 / 14.0
c9g.8xlarge	17.0	21.25	14.0
c9g.12xlarge	25.0	31.25	20.5
c9g.16xlarge	34.0	42.5	28.0
c9g.24xlarge	50.0	62.5	41.0
c9g.48xlarge	100.0	125.0	82.0
c9g.metal-48xl	100.0	125.0	82.0
c9gd.medium	0.55 / 15.0	0.688 / 15.0	0.455 / 12.0
c9gd.large	1.0 / 15.0	1.25 / 15.0	0.81 / 12.0
c9gd.xlarge	2.1 / 15.0	2.625 / 15.0	1.725 / 12.0
c9gd.2xlarge	4.25 / 17.0	5.313 / 17.0	3.5 / 14.0
c9gd.4xlarge	8.5 / 17.0	10.625 / 17.0	7.0 / 14.0
c9gd.8xlarge	17.0	21.25	14.0
c9gd.12xlarge	25.0	31.25	20.5
c9gd.16xlarge	34.0	42.5	28.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
c9gd.24xlarge	50.0	62.5	41.0
c9gd.48xlarge	100.0	125.0	82.0
c9gd.metal-48xl	100.0	125.0	82.0
m8a.medium	0.52 / 12.5	0.65 / 12.5	0.438 / 10.0
m8a.large	0.937/12.5	1.172 / 12.5	0.774 / 10.0
m8a.xlarge	1.875 / 12.5	2.344 / 12.5	1.562 / 10.0
m8a.2xlarge	3.75 / 15.0	4.688 / 15.0	3.125 / 12.5
m8a.4xlarge	7.5/15.0	9.375 / 15.0	6.25 / 12.5
m8a.8xlarge	15.0	18.75	12.5
m8a.12xlarge	22.5	28.125	18.75
m8a.16xlarge	30.0	37.5	25.0
m8a.24xlarge	40.0	50.0	32.5
m8a.48xlarge	75.0	93.75	60.0
m8a.metal-24xl	40.0	50.0	32.5
m8a.metal-48xl	75.0	93.75	60.0
m8g.medium	0.52 / 12.5	0.65 / 12.5	0.441 / 10.0
m8g.large	0.937 / 12.5	1.171 / 12.5	0.779 / 10.0
m8g.xlarge	1.875 / 12.5	2.344 / 12.5	1.562 / 10.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
m8g.2xlarge	3.75 / 15.0	4.688 / 15.0	3.125 / 12.5
m8g.4xlarge	7.5 / 15.0	9.375 / 15.0	6.25 / 12.5
m8g.8xlarge	15.0	18.75	12.5
m8g.12xlarge	22.5	28.125	18.75
m8g.16xlarge	30.0	37.5	25.0
m8g.24xlarge	40.0	50.0	32.5
m8g.48xlarge	50.0	62.5	40.0
m8g.metal-24xl	40.0	50.0	32.5
m8g.metal-48xl	50.0	62.5	40.0
m8gd.medium	0.52 / 12.5	0.65 / 12.5	0.441 / 10.0
m8gd.large	0.937 / 12.5	1.171 / 12.5	0.779 / 10.0
m8gd.xlarge	1.875 / 12.5	2.344 / 12.5	1.562 / 10.0
m8gd.2xlarge	3.75 / 15.0	4.688 / 15.0	3.125 / 12.5
m8gd.4xlarge	7.5 / 15.0	9.375 / 15.0	6.25 / 12.5
m8gd.8xlarge	15.0	18.75	12.5
m8gd.12xlarge	22.5	28.125	18.75
m8gd.16xlarge	30.0	37.5	25.0
m8gd.24xlarge	40.0	50.0	32.5

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
m8gd.48xlarge	50.0	62.5	40.0
m8gd.metal-24xl	40.0	50.0	32.5
m8gd.metal-48xl	50.0	62.5	40.0
m8i.large	0.937/12.5	1.172 / 12.5	0.774 / 10.0
m8i.xlarge	1.875 / 12.5	2.344 / 12.5	1.562 / 10.0
m8i.2xlarge	3.75 / 15.0	4.688 / 15.0	3.125 / 12.5
m8i.4xlarge	7.5/15.0	9.375 / 15.0	6.25 / 12.5
m8i.8xlarge	15.0	18.75	12.5
m8i.12xlarge	22.5	28.125	18.75
m8i.16xlarge	30.0	37.5	25.0
m8i.24xlarge	40.0	50.0	32.5
m8i.32xlarge	50.0	62.5	40.0
m8i.48xlarge	75.0	93.75	60.0
m8i.96xlarge	100.0	125.0	80.0
m8i.metal-48xl	75.0	93.75	60.0
m8i.metal-96xl	100.0	125.0	80.0
m8id.large	0.937/12.5	1.172 / 12.5	0.774 / 10.0
m8id.xlarge	1.875 / 12.5	2.344 / 12.5	1.562 / 10.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
m8id.2xlarge	3.75 / 15.0	4.688 / 15.0	3.125 / 12.5
m8id.4xlarge	7.5/15.0	9.375 / 15.0	6.25 / 12.5
m8id.8xlarge	15.0	18.75	12.5
m8id.12xlarge	22.5	28.125	18.75
m8id.16xlarge	30.0	37.5	25.0
m8id.24xlarge	40.0	50.0	32.5
m8id.32xlarge	50.0	62.5	40.0
m8id.48xlarge	75.0	93.75	60.0
m8id.96xlarge	100.0	125.0	80.0
m8id.metal-48xl	75.0	93.75	60.0
m8id.metal-96xl	100.0	125.0	80.0
m8i-flex.large	0.468 / 12.5	0.585 / 12.5	0.389 / 10.0
m8i-flex.xlarge	0.937/12.5	1.172 / 12.5	0.779 / 10.0
m8i-flex.2xlarge	1.875 / 15.0	2.344 / 15.0	1.562 / 12.5
m8i-flex.4xlarge	3.75 / 15.0	4.688 / 15.0	3.125 / 12.5
m8i-flex.8xlarge	7.5/15.0	9.375 / 18.75	6.25 / 12.5
m8i-flex.12xlarge	11.25 / 22.5	14.063 / 28.125	9.375 / 18.75
m8i-flex.16xlarge	15.0 / 30.0	18.75 / 37.5	12.5 / 25.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
m9g.medium	0.55 / 15.0	0.688 / 15.0	0.455 / 12.0
m9g.large	1.0 / 15.0	1.25 / 15.0	0.81 / 12.0
m9g.xlarge	2.1 / 15.0	2.625 / 15.0	1.725 / 12.0
m9g.2xlarge	4.25 / 17.0	5.313 / 17.0	3.5 / 14.0
m9g.4xlarge	8.5 / 17.0	10.625 / 17.0	7.0 / 14.0
m9g.8xlarge	17.0	21.25	14.0
m9g.12xlarge	25.0	31.25	20.5
m9g.16xlarge	34.0	42.5	28.0
m9g.24xlarge	50.0	62.5	41.0
m9g.48xlarge	100.0	125.0	82.0
m9g.metal-48xl	100.0	125.0	82.0
m9gd.medium	0.55 / 15.0	0.688 / 15.0	0.455 / 12.0
m9gd.large	1.0 / 15.0	1.25 / 15.0	0.81 / 12.0
m9gd.xlarge	2.1 / 15.0	2.625 / 15.0	1.725 / 12.0
m9gd.2xlarge	4.25 / 17.0	5.313 / 17.0	3.5 / 14.0
m9gd.4xlarge	8.5 / 17.0	10.625 / 17.0	7.0 / 14.0
m9gd.8xlarge	17.0	21.25	14.0
m9gd.12xlarge	25.0	31.25	20.5

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
m9gd.16xlarge	34.0	42.5	28.0
m9gd.24xlarge	50.0	62.5	41.0
m9gd.48xlarge	100.0	125.0	82.0
m9gd.metal-48xl	100.0	125.0	82.0
r8a.medium	0.52 / 12.5	0.65 / 12.5	0.438 / 10.0
r8a.large	0.937/12.5	1.172 / 12.5	0.774 / 10.0
r8a.xlarge	1.875 / 12.5	2.344 / 12.5	1.562 / 10.0
r8a.2xlarge	3.75 / 15.0	4.688 / 15.0	3.125 / 12.5
r8a.4xlarge	7.5/15.0	9.375 / 15.0	6.25 / 12.5
r8a.8xlarge	15.0	18.75	12.5
r8a.12xlarge	22.5	28.125	18.75
r8a.16xlarge	30.0	37.5	25.0
r8a.24xlarge	40.0	50.0	32.5
r8a.48xlarge	75.0	93.75	60.0
r8a.metal-24xl	40.0	50.0	32.5
r8a.metal-48xl	75.0	93.75	60.0
r8g.medium	0.52 / 12.5	0.65 / 12.5	0.441 / 10.0
r8g.large	0.937 / 12.5	1.171 / 12.5	0.779 / 10.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
r8g.xlarge	1.875 / 12.5	2.344 / 12.5	1.562 / 10.0
r8g.2xlarge	3.75 / 15.0	4.688 / 15.0	3.125 / 12.5
r8g.4xlarge	7.5 / 15.0	9.375 / 15.0	6.25 / 12.5
r8g.8xlarge	15.0	18.75	12.5
r8g.12xlarge	22.5	28.125	18.75
r8g.16xlarge	30.0	37.5	25.0
r8g.24xlarge	40.0	50.0	32.5
r8g.48xlarge	50.0	62.5	40.0
r8g.metal-24xl	40.0	50.0	32.5
r8g.metal-48xl	50.0	62.5	40.0
r8gd.medium	0.52 / 12.5	0.65 / 12.5	0.441 / 10.0
r8gd.large	0.937 / 12.5	1.171 / 12.5	0.779 / 10.0
r8gd.xlarge	1.875 / 12.5	2.344 / 12.5	1.562 / 10.0
r8gd.2xlarge	3.75 / 15.0	4.688 / 15.0	3.125 / 12.5
r8gd.4xlarge	7.5 / 15.0	9.375 / 15.0	6.25 / 12.5
r8gd.8xlarge	15.0	18.75	12.5
r8gd.12xlarge	22.5	28.125	18.75
r8gd.16xlarge	30.0	37.5	25.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
r8gd.24xlarge	40.0	50.0	32.5
r8gd.48xlarge	50.0	62.5	40.0
r8gd.metal-24xl	40.0	50.0	32.5
r8gd.metal-48xl	50.0	62.5	40.0
r8i.large	0.937 / 12.5	1.172 / 12.5	0.774 / 10.0
r8i.xlarge	1.875 / 12.5	2.344 / 12.5	1.562 / 10.0
r8i.2xlarge	3.75 / 15.0	4.688 / 15.0	3.125 / 12.5
r8i.4xlarge	7.5 / 15.0	9.375 / 15.0	6.25 / 12.5
r8i.8xlarge	15.0	18.75	12.5
r8i.12xlarge	22.5	28.125	18.75
r8i.16xlarge	30.0	37.5	25.0
r8i.24xlarge	40.0	50.0	32.5
r8i.32xlarge	50.0	62.5	40.0
r8i.48xlarge	75.0	93.75	60.0
r8i.96xlarge	100.0	125.0	80.0
r8i.metal-48xl	75.0	93.75	60.0
r8i.metal-96xl	100.0	125.0	80.0
r8id.large	0.937/12.5	1.172 / 12.5	0.774 / 10.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
r8id.xlarge	1.875 / 12.5	2.344 / 12.5	1.562 / 10.0
r8id.2xlarge	3.75 / 15.0	4.688 / 15.0	3.125 / 12.5
r8id.4xlarge	7.5/15.0	9.375 / 15.0	6.25 / 12.5
r8id.8xlarge	15.0	18.75	12.5
r8id.12xlarge	22.5	28.125	18.75
r8id.16xlarge	30.0	37.5	25.0
r8id.24xlarge	40.0	50.0	32.5
r8id.32xlarge	50.0	62.5	40.0
r8id.48xlarge	75.0	93.75	60.0
r8id.96xlarge	100.0	125.0	80.0
r8id.metal-48xl	75.0	93.75	60.0
r8id.metal-96xl	100.0	125.0	80.0
r8i-flex.large	0.468 / 12.5	0.585 / 12.5	0.389 / 10.0
r8i-flex.xlarge	0.937 / 12.5	1.172 / 12.5	0.779 / 10.0
r8i-flex.2xlarge	1.875 / 15.0	2.344 / 15.0	1.562 / 12.5
r8i-flex.4xlarge	3.75 / 15.0	4.688 / 15.0	3.125 / 12.5
r8i-flex.8xlarge	7.5 / 15.0	9.375 / 18.75	6.25 / 12.5
r8i-flex.12xlarge	11.25 / 22.5	14.063 / 28.125	9.375 / 18.75

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
r8i-flex.16xlarge	15.0 / 30.0	18.75 / 37.5	12.5 / 25.0
x8g.medium	0.52 / 12.5	0.65 / 12.5	0.441 / 10.0
x8g.large	0.937 / 12.5	1.171 / 12.5	0.779 / 10.0
x8g.xlarge	1.875 / 12.5	2.344 / 12.5	1.562 / 10.0
x8g.2xlarge	3.75 / 15.0	4.688 / 15.0	3.125 / 12.5
x8g.4xlarge	7.5 / 15.0	9.375 / 15.0	6.25 / 12.5
x8g.8xlarge	15.0	18.75	12.5
x8g.12xlarge	22.5	28.125	18.75
x8g.16xlarge	30.0	37.5	25.0
x8g.24xlarge	40.0	50.0	32.5
x8g.48xlarge	50.0	62.5	40.0
x8g.metal-24xl	40.0	50.0	32.5
x8g.metal-48xl	50.0	62.5	40.0
x8aedz.large	1.562 / 18.75	1.953 / 18.75	1.249 / 15.0
x8aedz.xlarge	3.125 / 18.75	3.907 / 18.75	2.5 / 15.0
x8aedz.3xlarge	9.375 / 18.75	11.719 / 18.75	7.5/15.0
x8aedz.6xlarge	18.75	23.438	15.0
x8aedz.12xlarge	37.5	46.875	30.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
x8aedz.24xlarge	75.0	93.75	60.0
x8aedz.metal-12xl	37.5	46.875	30.0
x8aedz.metal-24xl	75.0	93.75	60.0
x8i.large	0.937/12.5	1.172 / 12.5	0.774 / 10.0
x8i.xlarge	1.875 / 12.5	2.344 / 12.5	1.562 / 10.0
x8i.2xlarge	3.75 / 15.0	4.688 / 15.0	3.125 / 12.5
x8i.4xlarge	7.5/15.0	9.375 / 15.0	6.25 / 12.5
x8i.8xlarge	15.0	18.75	12.5
x8i.12xlarge	22.5	28.125	18.75
x8i.16xlarge	30.0	37.5	25.0
x8i.24xlarge	40.0	50.0	32.5
x8i.32xlarge	50.0	62.5	40.0
x8i.48xlarge	75.0	93.75	60.0
x8i.64xlarge	80.0	100.0	62.5
x8i.96xlarge	100.0	125.0	80.0
x8i.metal-48xl	75.0	93.75	60.0
x8i.metal-96xl	100.0	125.0	80.0

Amazon EBS パフォーマンス

次の表では、`default`、`vpc-1`、`ebs-1` の設定で達成できる Amazon EBS のパフォーマンスを Gbps で示しています。

インスタンスタイプ	**`default`** (ベースライン/バースト)	**`vpc-1`** (ベースライン/バースト)	**`ebs-1`** (ベースライン/バースト)
c8a.medium	0.325 / 10.0	0.195 / 6.25	0.407 / 10.0
c8a.large	0.65 / 10.0	0.415 / 6.25	0.813 / 10.0
c8a.xlarge	1.25 / 10.0	0.781 / 6.25	1.563 / 10.0
c8a.2xlarge	2.5 / 10.0	1.562 / 6.25	3.125 / 10.0
c8a.4xlarge	5.0 / 10.0	3.125 / 6.25	6.25 / 10.0
c8a.8xlarge	10.0	6.25	12.5
c8a.12xlarge	15.0	9.375	18.75
c8a.16xlarge	20.0	12.5	25.0
c8a.24xlarge	30.0	20.0	37.5
c8a.48xlarge	60.0	41.25	75.0
c8a.metal-24xl	30.0	20.0	37.5
c8a.metal-48xl	60.0	41.25	75.0
c8g.medium	0.315 / 10.0	0.185 / 6.25	0.394 / 10.0
c8g.large	0.63 / 10.0	0.396 / 6.25	0.788 / 10.0
c8g.xlarge	1.25 / 10.0	0.781 / 6.25	1.563 / 10.0
c8g.2xlarge	2.5 / 10.0	1.562 / 6.25	3.125 / 10.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
c8g.4xlarge	5.0 / 10.0	3.125 / 6.25	6.25 / 10.0
c8g.8xlarge	10.0	6.25	12.5
c8g.12xlarge	15.0	9.375	18.75
c8g.16xlarge	20.0	12.5	25.0
c8g.24xlarge	30.0	20.0	37.5
c8g.48xlarge	40.0	27.5	50.0
c8g.metal-24xl	30.0	20.0	37.5
c8g.metal-48xl	40.0	27.5	50.0
c8gd.medium	0.315 / 10.0	0.185 / 6.25	0.394 / 10.0
c8gd.large	0.63 / 10.0	0.396 / 6.25	0.788 / 10.0
c8gd.xlarge	1.25 / 10.0	0.781 / 6.25	1.563 / 10.0
c8gd.2xlarge	2.5 / 10.0	1.562 / 6.25	3.125 / 10.0
c8gd.4xlarge	5.0 / 10.0	3.125 / 6.25	6.25 / 10.0
c8gd.8xlarge	10.0	6.25	12.5
c8gd.12xlarge	15.0	9.375	18.75
c8gd.16xlarge	20.0	12.5	25.0
c8gd.24xlarge	30.0	20.0	37.5
c8gd.48xlarge	40.0	27.5	50.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
c8gd.metal-24xl	30.0	20.0	37.5
c8gd.metal-48xl	40.0	27.5	50.0
c8i.large	0.65 / 10.0	0.415 / 6.25	0.813 / 10.0
c8i.xlarge	1.25 / 10.0	0.781 / 6.25	1.563 / 10.0
c8i.2xlarge	2.5 / 10.0	1.562 / 6.25	3.125 / 10.0
c8i.4xlarge	5.0 / 10.0	3.125 / 6.25	6.25 / 10.0
c8i.8xlarge	10.0	6.25	12.5
c8i.12xlarge	15.0	9.375	18.75
c8i.16xlarge	20.0	12.5	25.0
c8i.24xlarge	30.0	20.0	37.5
c8i.32xlarge	40.0	27.5	50.0
c8i.48xlarge	60.0	41.25	75.0
c8i.96xlarge	80.0	55.0	100.0
c8i.metal-48xl	60.0	41.25	75.0
c8i.metal-96xl	80.0	55.0	100.0
c8id.large	0.65 / 10.0	0.415 / 6.25	0.813 / 10.0
c8id.xlarge	1.25 / 10.0	0.781 / 6.25	1.563 / 10.0
c8id.2xlarge	2.5 / 10.0	1.562 / 6.25	3.125 / 10.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
c8id.4xlarge	5.0 / 10.0	3.125 / 6.25	6.25 / 10.0
c8id.8xlarge	10.0	6.25	12.5
c8id.12xlarge	15.0	9.375	18.75
c8id.16xlarge	20.0	12.5	25.0
c8id.24xlarge	30.0	20.0	37.5
c8id.32xlarge	40.0	27.5	50.0
c8id.48xlarge	60.0	41.25	75.0
c8id.96xlarge	80.0	55.0	100.0
c8id.metal-48xl	60.0	41.25	75.0
c8id.metal-96xl	80.0	55.0	100.0
c8i-flex.large	0.315 / 10.0	0.198 / 6.25	0.394 / 10.0
c8i-flex.xlarge	0.63 / 10.0	0.395 / 6.25	0.788 / 10.0
c8i-flex.2xlarge	1.25 / 10.0	0.781 / 6.25	1.563 / 10.0
c8i-flex.4xlarge	2.5 / 10.0	1.562 / 6.25	3.125 / 10.0
c8i-flex.8xlarge	5.0 / 10.0	3.125 / 6.25	6.25 / 12.5
c8i-flex.12xlarge	7.5 / 15.0	4.687 / 9.375	9.375 / 18.75
c8i-flex.16xlarge	10.0 / 20.0	6.25 / 12.5	12.5 / 25.0
c9g.medium	0.38 / 12.0	0.242 / 7.75	0.475 / 12.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
c9g.large	0.76 / 12.0	0.51 / 7.75	0.95 / 12.0
c9g.xlarge	1.5 / 12.0	0.975 / 7.75	1.875 / 12.0
c9g.2xlarge	3.0 / 12.0	1.937 / 7.75	3.75 / 12.0
c9g.4xlarge	6.0 / 12.0	3.875 / 7.75	7.5 / 12.0
c9g.8xlarge	12.0	7.75	15.0
c9g.12xlarge	18.0	11.75	22.5
c9g.16xlarge	24.0	15.5	30.0
c9g.24xlarge	36.0	23.5	45.0
c9g.48xlarge	72.0	47.0	90.0
c9g.metal-48xl	72.0	47.0	90.0
c9gd.medium	0.38 / 12.0	0.242 / 7.75	0.475 / 12.0
c9gd.large	0.76 / 12.0	0.51 / 7.75	0.95 / 12.0
c9gd.xlarge	1.5 / 12.0	0.975 / 7.75	1.875 / 12.0
c9gd.2xlarge	3.0 / 12.0	1.937 / 7.75	3.75 / 12.0
c9gd.4xlarge	6.0 / 12.0	3.875 / 7.75	7.5 / 12.0
c9gd.8xlarge	12.0	7.75	15.0
c9gd.12xlarge	18.0	11.75	22.5
c9gd.16xlarge	24.0	15.5	30.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
c9gd.24xlarge	36.0	23.5	45.0
c9gd.48xlarge	72.0	47.0	90.0
c9gd.metal-48xl	72.0	47.0	90.0
m8a.medium	0.325 / 10.0	0.195 / 6.25	0.407 / 10.0
m8a.large	0.65 / 10.0	0.415 / 6.25	0.813 / 10.0
m8a.xlarge	1.25 / 10.0	0.781 / 6.25	1.563 / 10.0
m8a.2xlarge	2.5 / 10.0	1.562 / 6.25	3.125 / 10.0
m8a.4xlarge	5.0 / 10.0	3.125 / 6.25	6.25 / 10.0
m8a.8xlarge	10.0	6.25	12.5
m8a.12xlarge	15.0	9.375	18.75
m8a.16xlarge	20.0	12.5	25.0
m8a.24xlarge	30.0	20.0	37.5
m8a.48xlarge	60.0	41.25	75.0
m8a.metal-24xl	30.0	20.0	37.5
m8a.metal-48xl	60.0	41.25	75.0
m8g.medium	0.315 / 10.0	0.185 / 6.25	0.394 / 10.0
m8g.large	0.63 / 10.0	0.396 / 6.25	0.788 / 10.0
m8g.xlarge	1.25 / 10.0	0.781 / 6.25	1.563 / 10.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
m8g.2xlarge	2.5 / 10.0	1.562 / 6.25	3.125 / 10.0
m8g.4xlarge	5.0 / 10.0	3.125 / 6.25	6.25 / 10.0
m8g.8xlarge	10.0	6.25	12.5
m8g.12xlarge	15.0	9.375	18.75
m8g.16xlarge	20.0	12.5	25.0
m8g.24xlarge	30.0	20.0	37.5
m8g.48xlarge	40.0	27.5	50.0
m8g.metal-24xl	30.0	20.0	37.5
m8g.metal-48xl	40.0	27.5	50.0
m8gd.medium	0.315 / 10.0	0.185 / 6.25	0.394 / 10.0
m8gd.large	0.63 / 10.0	0.396 / 6.25	0.788 / 10.0
m8gd.xlarge	1.25 / 10.0	0.781 / 6.25	1.563 / 10.0
m8gd.2xlarge	2.5 / 10.0	1.562 / 6.25	3.125 / 10.0
m8gd.4xlarge	5.0 / 10.0	3.125 / 6.25	6.25 / 10.0
m8gd.8xlarge	10.0	6.25	12.5
m8gd.12xlarge	15.0	9.375	18.75
m8gd.16xlarge	20.0	12.5	25.0
m8gd.24xlarge	30.0	20.0	37.5

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
m8gd.48xlarge	40.0	27.5	50.0
m8gd.metal-24xl	30.0	20.0	37.5
m8gd.metal-48xl	40.0	27.5	50.0
m8i.large	0.65 / 10.0	0.415 / 6.25	0.813 / 10.0
m8i.xlarge	1.25 / 10.0	0.781 / 6.25	1.563 / 10.0
m8i.2xlarge	2.5 / 10.0	1.562 / 6.25	3.125 / 10.0
m8i.4xlarge	5.0 / 10.0	3.125 / 6.25	6.25 / 10.0
m8i.8xlarge	10.0	6.25	12.5
m8i.12xlarge	15.0	9.375	18.75
m8i.16xlarge	20.0	12.5	25.0
m8i.24xlarge	30.0	20.0	37.5
m8i.32xlarge	40.0	27.5	50.0
m8i.48xlarge	60.0	41.25	75.0
m8i.96xlarge	80.0	55.0	100.0
m8i.metal-48xl	60.0	41.25	75.0
m8i.metal-96xl	80.0	55.0	100.0
m8id.large	0.65 / 10.0	0.415 / 6.25	0.813 / 10.0
m8id.xlarge	1.25 / 10.0	0.781 / 6.25	1.563 / 10.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
m8id.2xlarge	2.5 / 10.0	1.562 / 6.25	3.125 / 10.0
m8id.4xlarge	5.0 / 10.0	3.125 / 6.25	6.25 / 10.0
m8id.8xlarge	10.0	6.25	12.5
m8id.12xlarge	15.0	9.375	18.75
m8id.16xlarge	20.0	12.5	25.0
m8id.24xlarge	30.0	20.0	37.5
m8id.32xlarge	40.0	27.5	50.0
m8id.48xlarge	60.0	41.25	75.0
m8id.96xlarge	80.0	55.0	100.0
m8id.metal-48xl	60.0	41.25	75.0
m8id.metal-96xl	80.0	55.0	100.0
m8i-flex.large	0.315 / 10.0	0.198 / 6.25	0.394 / 10.0
m8i-flex.xlarge	0.63 / 10.0	0.395 / 6.25	0.788 / 10.0
m8i-flex.2xlarge	1.25 / 10.0	0.781 / 6.25	1.563 / 10.0
m8i-flex.4xlarge	2.5 / 10.0	1.562 / 6.25	3.125 / 10.0
m8i-flex.8xlarge	5.0 / 10.0	3.125 / 6.25	6.25 / 12.5
m8i-flex.12xlarge	7.5 / 15.0	4.687 / 9.375	9.375 / 18.75
m8i-flex.16xlarge	10.0 / 20.0	6.25 / 12.5	12.5 / 25.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
m9g.medium	0.38 / 12.0	0.242 / 7.75	0.475 / 12.0
m9g.large	0.76 / 12.0	0.51 / 7.75	0.95 / 12.0
m9g.xlarge	1.5 / 12.0	0.975 / 7.75	1.875 / 12.0
m9g.2xlarge	3.0 / 12.0	1.937 / 7.75	3.75 / 12.0
m9g.4xlarge	6.0 / 12.0	3.875 / 7.75	7.5 / 12.0
m9g.8xlarge	12.0	7.75	15.0
m9g.12xlarge	18.0	11.75	22.5
m9g.16xlarge	24.0	15.5	30.0
m9g.24xlarge	36.0	23.5	45.0
m9g.48xlarge	72.0	47.0	90.0
m9g.metal-48xl	72.0	47.0	90.0
m9gd.medium	0.38 / 12.0	0.242 / 7.75	0.475 / 12.0
m9gd.large	0.76 / 12.0	0.51 / 7.75	0.95 / 12.0
m9gd.xlarge	1.5 / 12.0	0.975 / 7.75	1.875 / 12.0
m9gd.2xlarge	3.0 / 12.0	1.937 / 7.75	3.75 / 12.0
m9gd.4xlarge	6.0 / 12.0	3.875 / 7.75	7.5 / 12.0
m9gd.8xlarge	12.0	7.75	15.0
m9gd.12xlarge	18.0	11.75	22.5

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
m9gd.16xlarge	24.0	15.5	30.0
m9gd.24xlarge	36.0	23.5	45.0
m9gd.48xlarge	72.0	47.0	90.0
m9gd.metal-48xl	72.0	47.0	90.0
r8a.medium	0.325 / 10.0	0.195 / 6.25	0.407 / 10.0
r8a.large	0.65 / 10.0	0.415 / 6.25	0.813 / 10.0
r8a.xlarge	1.25 / 10.0	0.781 / 6.25	1.563 / 10.0
r8a.2xlarge	2.5 / 10.0	1.562 / 6.25	3.125 / 10.0
r8a.4xlarge	5.0 / 10.0	3.125 / 6.25	6.25 / 10.0
r8a.8xlarge	10.0	6.25	12.5
r8a.12xlarge	15.0	9.375	18.75
r8a.16xlarge	20.0	12.5	25.0
r8a.24xlarge	30.0	20.0	37.5
r8a.48xlarge	60.0	41.25	75.0
r8a.metal-24xl	30.0	20.0	37.5
r8a.metal-48xl	60.0	41.25	75.0
r8g.medium	0.315 / 10.0	0.185 / 6.25	0.394 / 10.0
r8g.large	0.63 / 10.0	0.396 / 6.25	0.788 / 10.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
r8g.xlarge	1.25 / 10.0	0.781 / 6.25	1.563 / 10.0
r8g.2xlarge	2.5 / 10.0	1.562 / 6.25	3.125 / 10.0
r8g.4xlarge	5.0 / 10.0	3.125 / 6.25	6.25 / 10.0
r8g.8xlarge	10.0	6.25	12.5
r8g.12xlarge	15.0	9.375	18.75
r8g.16xlarge	20.0	12.5	25.0
r8g.24xlarge	30.0	20.0	37.5
r8g.48xlarge	40.0	27.5	50.0
r8g.metal-24xl	30.0	20.0	37.5
r8g.metal-48xl	40.0	27.5	50.0
r8gd.medium	0.315 / 10.0	0.185 / 6.25	0.394 / 10.0
r8gd.large	0.63 / 10.0	0.396 / 6.25	0.788 / 10.0
r8gd.xlarge	1.25 / 10.0	0.781 / 6.25	1.563 / 10.0
r8gd.2xlarge	2.5 / 10.0	1.562 / 6.25	3.125 / 10.0
r8gd.4xlarge	5.0 / 10.0	3.125 / 6.25	6.25 / 10.0
r8gd.8xlarge	10.0	6.25	12.5
r8gd.12xlarge	15.0	9.375	18.75
r8gd.16xlarge	20.0	12.5	25.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
r8gd.24xlarge	30.0	20.0	37.5
r8gd.48xlarge	40.0	27.5	50.0
r8gd.metal-24xl	30.0	20.0	37.5
r8gd.metal-48xl	40.0	27.5	50.0
r8i.large	0.65 / 10.0	0.415 / 6.25	0.813 / 10.0
r8i.xlarge	1.25 / 10.0	0.781 / 6.25	1.563 / 10.0
r8i.2xlarge	2.5 / 10.0	1.562 / 6.25	3.125 / 10.0
r8i.4xlarge	5.0 / 10.0	3.125 / 6.25	6.25 / 10.0
r8i.8xlarge	10.0	6.25	12.5
r8i.12xlarge	15.0	9.375	18.75
r8i.16xlarge	20.0	12.5	25.0
r8i.24xlarge	30.0	20.0	37.5
r8i.32xlarge	40.0	27.5	50.0
r8i.48xlarge	60.0	41.25	75.0
r8i.96xlarge	80.0	55.0	100.0
r8i.metal-48xl	60.0	41.25	75.0
r8i.metal-96xl	80.0	55.0	100.0
r8id.large	0.65 / 10.0	0.415 / 6.25	0.813 / 10.0

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
r8id.xlarge	1.25 / 10.0	0.781 / 6.25	1.563 / 10.0
r8id.2xlarge	2.5 / 10.0	1.562 / 6.25	3.125 / 10.0
r8id.4xlarge	5.0 / 10.0	3.125 / 6.25	6.25 / 10.0
r8id.8xlarge	10.0	6.25	12.5
r8id.12xlarge	15.0	9.375	18.75
r8id.16xlarge	20.0	12.5	25.0
r8id.24xlarge	30.0	20.0	37.5
r8id.32xlarge	40.0	27.5	50.0
r8id.48xlarge	60.0	41.25	75.0
r8id.96xlarge	80.0	55.0	100.0
r8id.metal-48xl	60.0	41.25	75.0
r8id.metal-96xl	80.0	55.0	100.0
r8i-flex.large	0.315 / 10.0	0.198 / 6.25	0.394 / 10.0
r8i-flex.xlarge	0.63 / 10.0	0.395 / 6.25	0.788 / 10.0
r8i-flex.2xlarge	1.25 / 10.0	0.781 / 6.25	1.563 / 10.0
r8i-flex.4xlarge	2.5 / 10.0	1.562 / 6.25	3.125 / 10.0
r8i-flex.8xlarge	5.0 / 10.0	3.125 / 6.25	6.25 / 12.5
r8i-flex.12xlarge	7.5 / 15.0	4.687 / 9.375	9.375 / 18.75

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
r8i-flex.16xlarge	10.0 / 20.0	6.25 / 12.5	12.5 / 25.0
x8g.medium	0.315 / 10.0	0.185 / 6.25	0.394 / 10.0
x8g.large	0.63 / 10.0	0.396 / 6.25	0.788 / 10.0
x8g.xlarge	1.25 / 10.0	0.781 / 6.25	1.563 / 10.0
x8g.2xlarge	2.5 / 10.0	1.562 / 6.25	3.125 / 10.0
x8g.4xlarge	5.0 / 10.0	3.125 / 6.25	6.25 / 10.0
x8g.8xlarge	10.0	6.25	12.5
x8g.12xlarge	15.0	9.375	18.75
x8g.16xlarge	20.0	12.5	25.0
x8g.24xlarge	30.0	20.0	37.5
x8g.48xlarge	40.0	27.5	50.0
x8g.metal-24xl	30.0	20.0	37.5
x8g.metal-48xl	40.0	27.5	50.0
x8aedz.large	1.25 / 15.0	0.859 / 10.312	1.563 / 15.0
x8aedz.xlarge	2.5 / 15.0	1.718 / 10.312	3.125 / 15.0
x8aedz.3xlarge	7.5/15.0	5.156 / 10.312	9.375 / 15.0
x8aedz.6xlarge	15.0	10.312	18.75
x8aedz.12xlarge	30.0	20.625	37.5

インスタンスタイプ	**default** (ベースライン/バースト)	**vpc-1** (ベースライン/バースト)	**ebs-1** (ベースライン/バースト)
x8aedz.24xlarge	60.0	41.25	75.0
x8aedz.metal-12xl	30.0	20.625	37.5
x8aedz.metal-24xl	60.0	41.25	75.0
x8i.large	0.65 / 10.0	0.415 / 6.25	0.813 / 10.0
x8i.xlarge	1.25 / 10.0	0.781 / 6.25	1.563 / 10.0
x8i.2xlarge	2.5 / 10.0	1.562 / 6.25	3.125 / 10.0
x8i.4xlarge	5.0 / 10.0	3.125 / 6.25	6.25 / 10.0
x8i.8xlarge	10.0	6.25	12.5
x8i.12xlarge	15.0	9.375	18.75
x8i.16xlarge	20.0	12.5	25.0
x8i.24xlarge	30.0	20.0	37.5
x8i.32xlarge	40.0	27.5	50.0
x8i.48xlarge	60.0	41.25	75.0
x8i.64xlarge	70.0	50.0	87.5
x8i.96xlarge	80.0	55.0	100.0
x8i.metal-48xl	60.0	41.25	75.0
x8i.metal-96xl	80.0	55.0	100.0

インスタンスの帯域幅をモニタリングします。

CloudWatch メトリクスを使用して、インスタンスのネットワーク帯域幅と送受信されたパケットをモニタリングできます。Elastic Network Adapter (ENA) ドライバーが提供するネットワークパフォーマンスメトリクスを使用して、トラフィックが Amazon EC2 がインスタンスレベルで定義するネットワーク許容量を超えるかモニタリングできます。

Amazon EC2 がインスタンスのメトリクスデータを CloudWatch に送信するときに、1 分単位か 5 分単位かを設定できます。CloudWatch インスタンスメトリクスでは許容量を超え、パケットがドロップされたことがネットワークパフォーマンスメトリクスに表示されることがあります。これはインスタンスのネットワークリソースに対する需要が短時間で急増し (マイクロバーストと呼ばれる)、CloudWatch メトリクスがこのようなマイクロ秒単位の急増を反映するのに十分な粒度を持っていない場合に発生します。

詳細情報

- インスタンスメトリクス
- ネットワークパフォーマンスのモニタリング

Amazon EC2 インスタンスでの拡張ネットワーキング

拡張ネットワーキングでは、シングルルート I/O 仮想化 (SR-IOV) を使用して、サポートされるインスタンスタイプにおける高性能ネットワーキング機能が提供されます。SR-IOV は、従来の仮想化ネットワークインターフェイスと比較し、I/O パフォーマンスが高く、CPU 利用率が低いデバイス仮想化の手法です。拡張ネットワーキングは、より高い帯域幅、1 秒あたりのパケット (PPS) のより高いパフォーマンス、常により低いインスタンス間レイテンシーを実現します。拡張ネットワーキングは追加料金なしで使用できます。

各インスタンスタイプでサポートされているネットワーク速度については、Amazon EC2インスタンスタイプを参照してください。

次のいずれかのメカニズムを使用して、拡張ネットワークを有効にすることができます。

Elastic Network Adapter (ENA)

> Elastic Network Adapter (ENA) は、サポート対象のインスタンスタイプに対して最大 100 Gbps のネットワーク速度をサポートします。

すべての Nitro ベースのインスタンスが ENA を使用してネットワーキングを強化しています。さらに、次の Xen ベースのインスタンスタイプは ENA (H1、I3、G3、`m4.16xlarge`、P3、P3dn、R4) を使用しています。

詳細については、「EC2 インスタンスで ENA による拡張ネットワーキングを有効にする」を参照してください。

Intel 82599 Virtual Function (VF) インターフェイス

Intel 82599 Virtual Function インターフェイスでは、サポートされているインスタンスタイプについて最大 10 Gbps のネットワーク速度がサポートされています。

インスタンスタイプ C3、C4、D2、I2、M4 (m4.16xlarge を除く)、R3 では、拡張ネットワーキングに Intel 82599 VF インターフェイスが使用されます。

詳細については、「Intel 82599 VF インターフェイスを使用する拡張ネットワーキング」を参照してください。

内容

- EC2 インスタンスで ENA による拡張ネットワーキングを有効にする
- ENA Express を使用して EC2 インスタンス間のネットワークパフォーマンスを高める
- Intel 82599 VF インターフェイスを使用する拡張ネットワーキング
- EC2 インスタンスでの ENA 設定のネットワークパフォーマンスのモニタリング
- Linux ベースの EC2 インスタンスのネットワークレイテンシーを改善する
- Nitro System のパフォーマンスチューニングに関する考慮事項
- EC2 Windows インスタンスでのネットワークパフォーマンスを最適化する

EC2 インスタンスで ENA による拡張ネットワーキングを有効にする

Amazon EC2 は、Elastic Network Adapter (ENA) を介してネットワーキング機能を提供します。拡張ネットワーキングを使用するには、必要な ENA ドライバーを定義した AMI を使用するか、必要な ENA ドライバーを手動でインストールする必要があります。その後、インスタンスで ENA サポートを有効にできます。

ENA ドライバーのリリースノートやインストール手順を確認するには、ご使用のインスタンスオペレーティングシステムプラットフォームに応じたタブを参照してください。

Linux

GitHub では、以下のドキュメントを参照できます。

- GitHub で「ENA Linux kernel driver release notes」を確認してください。
- インストール手順が含まれている ENA Linux カーネルドライバーの概要については、GitHub で「Elastic Network Adapter (ENA) ファミリー用Linuxカーネルドライバ」を参照してください。

Windows

このガイドの「Manage device drivers」セクションから、以下のドキュメントを確認できます。

- ENA Windows ドライバーのバージョンリリースを追跡する.
- EC2 Windows インスタンスに ENA ドライバーをインストールする.

Nitro ベースのインスタンスの場合、拡張ネットワーキング機能はインスタンスタイプが実装する Nitro のバージョンによって異なります。

インスタンスのネットワーク仕様を確認するには、インスタンスタイプのインスタンスファミリーリンクを選択してください。適用されるインスタンスファミリーがわからない場合は、「Amazon EC2 インスタンスタイプ」ガイドの「Naming conventions」を参照してください。

- 高速コンピューティングインスタンスのネットワーク仕様
- コンピューティング最適化インスタンスのネットワーク仕様
- 汎用インスタンスのネットワーク仕様
- ハイパフォーマンスコンピューティングインスタンスのネットワーク仕様
- メモリ最適化インスタンスのネットワーク仕様
- ストレージ最適化インスタンスのネットワーク仕様

内容

- ENA を使用した拡張ネットワーキングの前提条件
- 拡張ネットワーキングが有効化されているかどうかのテスト
- インスタンスでの拡張ネットワーキングの有効化
- ENA キュー

- Linux での ENA カーネルドライバーのトラブルシューティング
- エラスティックネットワークアダプタ Windows ドライバーのトラブルシューティング

ENA を使用した拡張ネットワーキングの前提条件

ENA を使用した拡張ネットワーキングを準備するには、次のようにインスタンスをセットアップします。

- Nitro ベースのインスタンスを起動します。
- インスタンスがインターネットに接続されていることを確認します。
- 保持する必要がある重要なデータがインスタンスにある場合、インスタンスから AMI を作成してそのデータをバックアップする必要があります。enaSupport 属性を有効にすると共に、ENA カーネルドライバーを更新すると、互換性のないインスタンスがレンダリングされたり、オペレーティングシステムに接続できなくなったりする可能性があります。最近のバックアップがある場合はこれが発生してもデータは保持されます。
- Linux インスタンス – インスタンスに対して ENA 拡張ネットワーキングが自動的に有効化されるように、サポートされているバージョンの Linux カーネルとサポートされているディストリビューションを使用してインスタンスを起動します。詳細については、ENA Linux Kernel Driver リリースノートを参照してください。
- Windows インスタンス – インスタンスで Windows Server 2008 R2 SP1 を実行している場合は、SHA-2 コード署名サポートが更新されていることを確認します。
- 選択した任意のコンピュータ、できればローカルのデスクトップまたはノートパソコンで、AWS マネジメントコンソール から AWS CloudShell を使用するか、AWS CLI もしくは AWS Tools for Windows PowerShell をインストールし設定します。詳細についてはAmazon EC2 へのアクセスもしくは AWS CloudShell ユーザーガイドを参照してください。拡張ネットワーキングは、Amazon EC2 コンソールから管理することはできません。

拡張ネットワーキングが有効化されているかどうかのテスト

インスタンスまたは AMI で拡張ネットワーキングが有効になっているかどうかをテストできます。

インスタンス属性

enaSupport インスタンス属性の値を確認します。

AWS CLI

describe-instances コマンドを使用します。

```
aws ec2 describe-instances \
    --instance-ids i-1234567890abcdef0 \
    --query "Reservations[].Instances[].EnaSupport"
```

拡張ネットワーキングが有効になっている場合、出力は次のようになります。

```
[
    true
]
```

PowerShell

Get-EC2Instance コマンドレットを使用します。

```
(Get-EC2Instance -InstanceId i-1234567890abcdef0).Instances.EnaSupport
```

拡張ネットワーキングが有効になっている場合、出力は次のようになります。

```
True
```

イメージ属性

enaSupport イメージ属性の値を確認します。

AWS CLI

describe-images コマンドを使用します。

```
aws ec2 describe-images \
    --image-id ami-0abcdef1234567890 \
    --query "Images[].EnaSupport"
```

拡張ネットワーキングが有効になっている場合、出力は次のようになります。

```
[
    true
```

```
]
```

PowerShell

Get-EC2Image コマンドレットを使用します。

```
(Get-EC2Image -ImageId ami-0abcdef1234567890).EnaSupport
```

拡張ネットワーキングが有効になっている場合、出力は次のようになります。

```
True
```

Linux ネットワークインターフェイスドライバー

以下のコマンドを使用して、ena カーネルドライバーが特定のインターフェイスで使用されていることを確認し、チェックするインターフェイス名に置き換えます。単一のインターフェイス (デフォルト) を使用している場合はeth0 です。Linux ディストリビューションが予測可能なネットワーク名をサポートしている場合、これは ens5 のような名前である可能性があります。詳細については、インスタンスでの拡張ネットワーキングの有効化 の RHEL、SUSE、および CentOS のセクションを展開します。

以下の例では、リストされているドライバーは vif であるため、ena カーネルドライバーはロードされません。

```
[ec2-user ~]$ ethtool -i eth0
driver: vif
version:
firmware-version:
bus-info: vif-0
supports-statistics: yes
supports-test: no
supports-eeprom-access: no
supports-register-dump: no
supports-priv-flags: no
```

この例では、ena カーネルドライバーが最小推奨バージョンでロードされています。このインスタンスでは拡張ネットワーキングが適切に設定されています。

```
[ec2-user ~]$ ethtool -i eth0
driver: ena
```

```
version: 1.5.0g
firmware-version:
expansion-rom-version:
bus-info: 0000:00:05.0
supports-statistics: yes
supports-test: no
supports-eeprom-access: no
supports-register-dump: no
supports-priv-flags: no
```

インスタンスでの拡張ネットワーキングの有効化

使用する手順はインスタンスのオペレーティングシステムによって異なります。

Amazon Linux

拡張ネットワーキングには ENA 搭載のカーネルドライバーが必要であり、Amazon Linux 向け AMI であれば、そうしたドライバーが含まれ、ENA にも対応しています。したがって、サポートされるインスタンスタイプで HVM バージョンの Amazon Linux を使用してインスタンスを起動した場合、拡張ネットワーキングは既にインスタンスで有効になっています。詳細については「拡張ネットワーキングが有効化されているかどうかのテスト」を参照してください。

Ubuntu

拡張ネットワーキングには ENA 搭載のカーネルドライバーが必要であり、最新の Ubuntu HVM AMI であれば、そうしたドライバーが含まれ、ENA にも対応しています。したがって、サポートされるインスタンスタイプで最新の Ubuntu HVM AMI を使用してインスタンスを起動した場合、拡張ネットワーキングは既にインスタンスで有効になっています。詳細については「拡張ネットワーキングが有効化されているかどうかのテスト」を参照してください。

以前の AMI を使用してインスタンスを起動した場合、まだ拡張ネットワーキングが有効になっていなければ、`linux-aws` カーネルパッケージをインストールして最新の拡張ネットワーキングドライバーを取得して、必要な属性を更新できます。

`linux-aws` カーネルパッケージをインストールするには (Ubuntu 16.04 以降)

Ubuntu 16.04 および 18.04 には、Ubuntu カスタムカーネル (`linux-aws` カーネルパッケージ) が付属しています。別のカーネルを使用するには、サポート にお問い合わせください。

`linux-aws` カーネルパッケージをインストールするには (Ubuntu Trusty 14.04)

1. インスタンスに接続します。

2. パッケージキャッシュおよびパッケージを更新します。

```
ubuntu:~$ sudo apt-get update && sudo apt-get upgrade -y linux-aws
```

> **Important**
>
> 更新プロセス中に grub をインストールするよう求められた場合は、/dev/xvda のインストール先として grub を使用し、現在のバージョンの /boot/grub/menu.lst を保持することを選択してください。

3. [EBS-backed インスタンス] ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを停止します。stop-instances (AWS CLI)、Stop-EC2Instance (AWS Tools for Windows PowerShell)。

   [Instance store-backed インスタンス] インスタンスを停止して属性を変更することはできません。代わりに、この手順に進んでください: Ubuntu で拡張ネットワーキングを有効にするには (Instance store-backed インスタンス)。

4. ローカルコンピュータから、次のいずれかのコマンドを使用して拡張ネットワーキングの属性を有効化します。

   - modify-instance-attribute (AWS CLI)

     ```
     aws ec2 modify-instance-attribute --instance-id i-1234567890abcdef0 --ena-support
     ```

   - Edit-EC2InstanceAttribute (Windows PowerShell 用のツール)

     ```
     Edit-EC2InstanceAttribute -InstanceId i-1234567890abcdef0 -EnaSupport $true
     ```

5. (オプション) Amazon EBS-backed AMI を作成するの説明に従って、インスタンスから AMI を作成します。AMI は、インスタンスから拡張ネットワーキング enaSupport 属性を継承します。このため、この AMI を使用することで、拡張ネットワーキングがデフォルトで有効になっている別のインスタンスを起動できます。

6. ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを開始します。start-instances (AWS CLI)、Start-EC2Instance (AWS Tools for Windows PowerShell)。

Ubuntu で拡張ネットワーキングを有効にするには (Instance store-backed インスタンス)

インスタンスを停止するステップまで、前の手順に従います。Amazon S3-backed AMI を作成するに記述されているように、新しい AMI を作成します。AMI を登録するときに拡張ネットワーキング属性を有効にしてください。

- register-image (AWS CLI)

```
aws ec2 register-image --ena-support ...
```

- Register-EC2Image (AWS Tools for Windows PowerShell)

```
Register-EC2Image -EnaSupport $true ...
```

RHEL、SUSE、CentOS

拡張ネットワーキングには ENA 搭載のカーネルドライバーが必要であり、Red Hat Enterprise Linux、SUSE Linux Enterprise Server、および CentOS 用の最新の AMI であれば、そうしたドライバーが含まれ、ENA にも対応しています。したがって、サポートされるインスタンスタイプで最新の AMI を使用してインスタンスを起動した場合、拡張ネットワーキングは既にインスタンスで有効になっています。詳細については「拡張ネットワーキングが有効化されているかどうかのテスト」を参照してください。

次の手順では、Amazon Linux AMI または Ubuntu 以外の Linux ディストリビューションで拡張ネットワーキングを有効にするための一般的なステップを説明します。コマンドの詳細な構文、ファイルの場所、パッケージやツールのサポートなどの詳細については、使用する Linux ディストリビューションのドキュメントを参照してください。

Linux で拡張ネットワーキングを有効化するには

1. インスタンスに接続します。
2. https://github.com/amzn/amzn-drivers の GitHub からインスタンスで ena カーネルドライバーのソースコードのクローンを作成します。(SUSE Linux Enterprise Server 12 SP2 以降には、デフォルトで ENA 2.02 が含まれているため、ENA ドライバーをダウンロードしてコンパイルする必要はありません。SUSE Linux Enterprise Server 12 SP2 以降では、必要なドライバーバージョンを標準カーネルに追加するためリクエストを申請する必要があります。)

```
git clone https://github.com/amzn/amzn-drivers
```

3. インスタンスで ena カーネルドライバーをコンパイルし、インストールします。これらの手順は Linux ディストリビューションによって異なります。Red Hat Enterprise Linux でのカーネルドライバーのコンパイルの詳細については、「RHEL を実行する Amazon EC2 インスタンスに拡張ネットワークサポート用の最新の ENS ドライバーをインストールする方法」を参照してください。
4. sudo depmod コマンドを実行して、カーネルドライバーの依存関係を更新します。
5. 起動時に新しいカーネルドライバーがロードされるように、インスタンスの initramfs を更新します。例えば、ディストリビューションで dracut がサポートされる場合、次のコマンドを使用できます。

```
dracut -f -v
```

6. システムがデフォルトで予測可能なネットワークインターフェイス名を使用するかどうかを確認します。systemd または udev のバージョン 197 以上を使用するシステムの場合、イーサネットデバイスの名前を変更でき、単一ネットワークインターフェイスの名前が eth0 になることは保証されません。この動作はインスタンスに接続する際に問題の原因となる可能性があります。詳細と他の設定オプションについては、freedesktop.org ウェブサイトでPredictable Network Interface Names/を参照してください。

   a. 次のコマンドを使用して、RPM ベースのシステムで systemd または udev のバージョンを確認できます。

   ```
   rpm -qa | grep -e '^systemd-[0-9]\+\|^udev-[0-9]\+'
   systemd-208-11.el7_0.2.x86_64
   ```

   上記の Red Hat Enterprise Linux 7 の例ではsystemd のバージョンは 208 であるため、予測可能なネットワークインターフェイス名は無効になっている必要があります。

   b. net.ifnames=0 オプションを GRUB_CMDLINE_LINUX の /etc/default/grub 行に追加することによって、予測可能なネットワークインターフェイス名を無効にします。

   ```
   sudo sed -i '/^GRUB\_CMDLINE\_LINUX/s/\"$/\ net\.ifnames\=0\"/' /etc/default/
   grub
   ```

   c. grub の設定ファイルを再ビルドします。

   ```
   sudo grub2-mkconfig -o /boot/grub2/grub.cfg
   ```

7. [EBS-backed インスタンス] ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを停止します。stop-instances (AWS CLI)、Stop-EC2Instance (AWS Tools for Windows PowerShell)。

   [Instance store-backed インスタンス] インスタンスを停止して属性を変更することはできません。代わりに、この手順に進んでください: Linux で拡張ネットワーキングを有効にするには (Instance store-backed インスタンス)。

8. ローカルコンピュータから、次のいずれかのコマンドを使用して拡張ネットワーキングの `enaSupport` 属性を有効化します。

   - modify-instance-attribute (AWS CLI)

     ```
     aws ec2 modify-instance-attribute --instance-id i-1234567890abcdef0 --ena-support
     ```

   - Edit-EC2InstanceAttribute (Windows PowerShell 用のツール)

     ```
     Edit-EC2InstanceAttribute -InstanceId i-1234567890abcdef0 -EnaSupport $true
     ```

9. (オプション) Amazon EBS-backed AMI を作成するの説明に従って、インスタンスから AMI を作成します。AMI は、インスタンスから拡張ネットワーキング `enaSupport` 属性を継承します。このため、この AMI を使用することで、拡張ネットワーキングがデフォルトで有効になっている別のインスタンスを起動できます。

   インスタンスオペレーティングシステムに `/etc/udev/rules.d/70-persistent-net.rules` が含まれている場合にはAMI を作成する前にそれを削除する必要があります。このファイルには元のインスタンスのイーサネットアダプターの MAC アドレスが保存されています。別のインスタンスがこのファイルを使用して起動した場合、オペレーティングシステムがそのデバイスを検出できなくなり、`eth0` が失敗して、起動に関する問題が発生することがあります。このファイルは次の起動サイクルで再び生成され、AMI から起動されるインスタンスごとに独自のバージョンが作成されます。

10. ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを開始します。start-instances (AWS CLI)、Start-EC2Instance (AWS Tools for Windows PowerShell)。

11. (オプション) インスタンスに接続し、カーネルドライバーがインストールされていることを確認します。

    拡張ネットワーキングを有効にした後にインスタンスに接続できない場合、Linux での ENA カーネルドライバーのトラブルシューティングを参照してください。

Linux で拡張ネットワーキングを有効にするには (Instance store-backed インスタンス)

インスタンスを停止するステップまで、前の手順に従います。Amazon S3-backed AMI を作成するに記述されているように、新しい AMI を作成します。AMI を登録するときに拡張ネットワーキング属性を有効にしてください。

- register-image (AWS CLI)

```
aws ec2 register-image --ena-support ...
```

- Register-EC2Image (AWS Tools for Windows PowerShell)

```
Register-EC2Image -EnaSupport ...
```

DKMS を備えた Ubuntu

この方法は、テストおよびフィードバックのみを目的としています。本番稼働用デプロイによる使用を目的としていません。本番稼働デプロイについては、Ubuntuを参照してください。

> **Important**
>
> DKMS を使用すると、サブスクリプションのサポート契約が無効になります。本稼働環境では使用しないでください。

Ubuntu で ENA を使用した拡張ネットワーキングを有効にするには (EBS-backed インスタンス)

1. Ubuntu のステップ 1 および 2 を行います。
2. `build-essential` パッケージをインストールしてカーネルドライバーと `dkms` パッケージをコンパイルし、カーネルが更新されるたびに `ena` カーネルドライバーが再構築されるようにします。

```
ubuntu:~$ sudo apt-get install -y build-essential dkms
```

3. https://github.com/amzn/amzn-drivers の GitHub からインスタンスで `ena` カーネルドライバーのソースのクローンを作成します。

```
ubuntu:~$ git clone https://github.com/amzn/amzn-drivers
```

4. amzn-drivers パッケージを /usr/src/ ディレクトリに移動して、カーネルの更新のたびに DKMS がこのパッケージを見つけて構築できるようにします。ソースコードのバージョン番号 (現在のバージョン番号はリリースノートにあります) をディレクトリ名に付加します。例えば、バージョン 1.0.0 は以下のようになります。

```
ubuntu:~$ sudo mv amzn-drivers /usr/src/amzn-drivers-1.0.0
```

5. 以下の値を使用して DKMS 設定ファイルを作成し、ena のバージョンに置き換えます。

   ファイルを作成します。

```
ubuntu:~$ sudo touch /usr/src/amzn-drivers-1.0.0/dkms.conf
```

   ファイルを編集し、次の値を追加します。

```
ubuntu:~$ sudo vim /usr/src/amzn-drivers-1.0.0/dkms.conf
PACKAGE_NAME="ena"
PACKAGE_VERSION="1.0.0"
CLEAN="make -C kernel/linux/ena clean"
MAKE="make -C kernel/linux/ena/ BUILD_KERNEL=${kernelver}"
BUILT_MODULE_NAME[0]="ena"
BUILT_MODULE_LOCATION="kernel/linux/ena"
DEST_MODULE_LOCATION[0]="/updates"
DEST_MODULE_NAME[0]="ena"
AUTOINSTALL="yes"
```

6. DKMS を使用して、インスタンスで ena カーネルドライバーを追加、構築、インストールします。

   DKMS にカーネルドライバーを追加します。

```
ubuntu:~$ sudo dkms add -m amzn-drivers -v 1.0.0
```

   dkms コマンドでカーネルドライバーを構築します。

```
ubuntu:~$ sudo dkms build -m amzn-drivers -v 1.0.0
```

   dkms を使用してカーネルドライバーをインストールします。

```
ubuntu:~$ sudo dkms install -m amzn-drivers -v 1.0.0
```

7. 起動時に正しいカーネルドライバーがロードされるように、initramfs を再構築します。

```
ubuntu:~$ sudo update-initramfs -u -k all
```

8. 拡張ネットワーキングが有効化されているかどうかのテスト から modinfo ena コマンドを使用して、ena カーネルドライバーがインストールされていることを確認します。

```
ubuntu:~$ modinfo ena
filename:    /lib/modules/3.13.0-74-generic/updates/dkms/ena.ko
version:  1.0.0
license:  GPL
description: Elastic Network Adapter (ENA)
author:   Amazon.com, Inc. or its affiliates
srcversion:  9693C876C54CA64AE48F0CA
alias:    pci:v00001D0Fd0000EC21sv*sd*bc*sc*i*
alias:    pci:v00001D0Fd0000EC20sv*sd*bc*sc*i*
alias:    pci:v00001D0Fd00001EC2sv*sd*bc*sc*i*
alias:    pci:v00001D0Fd00000EC2sv*sd*bc*sc*i*
depends:
vermagic:    3.13.0-74-generic SMP mod_unload modversions
parm:     debug:Debug level (0=none,...,16=all) (int)
parm:     push_mode:Descriptor / header push mode (0=automatic,1=disable,3=enable)
    0 - Automatically choose according to device capability (default)
    1 - Don't push anything to device memory
    3 - Push descriptors and header buffer to device memory (int)
parm:     enable_wd:Enable keepalive watchdog (0=disable,1=enable,default=1) (int)
parm:     enable_missing_tx_detection:Enable missing Tx completions. (default=1)
 (int)
parm:     numa_node_override_array:Numa node override map
(array of int)
parm:     numa_node_override:Enable/Disable numa node override (0=disable)
(int)
```

9. Ubuntu のステップ 3 に進みます。

Windows の拡張ネットワーキングの有効化

インスタンスを起動し、すでに拡張ネットワーキングが有効になっていない場合、必要なネットワークアダプタードライバーをダウンロードしてインスタンスにインストールし、拡張ネットワーキングを有効にするように enaSupport インスタンス属性を設定する必要があります。

拡張ネットワーキングを有効にするには

1. インスタンスに接続してローカル管理者としてログインします。
2. [Windows Server 2016 と 2019 のみ] 以下の EC2Launch PowerShell スクリプトを実行して、ドライバーのインストール後にインスタンスを設定します。

```
PS C:\> C:\ProgramData\Amazon\EC2-Windows\Launch\Scripts\InitializeInstance.ps1 -
Schedule
```

3. インスタンスから、次のようにドライバーをインストールします。
   a. インスタンスに最新のドライバーをダウンロードします。
   b. zip アーカイブを展開します。
   c. `install.ps1` PowerShell スクリプトを実行してドライバーをインストールします。

> **Note**
> 実行ポリシーエラーが発生した場合は、ポリシーを `Unrestricted` に設定します (デフォルトでは、`Restricted` または `RemoteSigned` に設定されています)。コマンドラインで、`Set-ExecutionPolicy -ExecutionPolicy Unrestricted` を実行し、次に PowerShell スクリプト `install.ps1` を再度実行します。

4. ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを停止します。stop-instances (AWS CLI)/Stop-EC2Instance、 (AWS Tools for Windows PowerShell)。
5. 次のように、インスタンスの ENA サポートを有効にします。
   a. ローカルコンピュータから、次のいずれかのコマンドを実行して、インスタンスの EC2 インスタンス ENA サポート属性を確認します。属性が有効になっていない場合、出力は「[]」 または空白です。`EnaSupport`はデフォルトで `false` に設定されます。
      - describe-instances (AWS CLI)

```
aws ec2 describe-instances --instance-ids i-1234567890abcdef0 --query
 "Reservations[].Instances[].EnaSupport"
```

- Get-EC2Instance (Windows PowerShell 用のツール)

```
(Get-EC2Instance -InstanceId i-1234567890abcdef0).Instances.EnaSupport
```

b. ENA サポートを有効にするには、次のいずれかのコマンドを実行します:

- modify-instance-attribute (AWS CLI)

```
aws ec2 modify-instance-attribute --instance-id i-1234567890abcdef0 --ena-
support
```

- Edit-EC2InstanceAttribute (AWS Tools for Windows PowerShell)

```
Edit-EC2InstanceAttribute -InstanceId i-1234567890abcdef0 -EnaSupport $true
```

インスタンスを再開するときに問題が発生した場合は、次のいずれかのコマンドを使用して ENA サポートを無効にすることもできます。

- modify-instance-attribute (AWS CLI)

```
aws ec2 modify-instance-attribute --instance-id i-1234567890abcdef0 --no-ena-
support
```

- Edit-EC2InstanceAttribute (AWS Tools for Windows PowerShell)

```
Edit-EC2InstanceAttribute -InstanceId i-1234567890abcdef0 -EnaSupport $false
```

c. 前述のように describe-instances または Get-EC2Instance を使用して、属性が `true` に設定されていることを確認します。次のような出力が表示されます。

```
[
 true
]
```

6. ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを開始します。start-instances (AWS CLI)、Start-EC2Instance (AWS Tools for Windows PowerShell)。
7. インスタンスで、次のように ENA ドライバーがインストールされて有効であることを検証します。

   a. ネットワークアイコンを右クリックして、[Open Network and Sharing Center] を選択してください。

   b. イーサネットアダプター ([Ethernet 2] など) を選択してください。

   c. [Details] を選択してください。[Network Connection Details] で、[Description] が [Amazon Elastic Network Adapter] であることを確認します。
8. (オプション) インスタンスから AMI を作成します。AMI は、インスタンスから `enaSupport` 属性を継承します。したがって、この AMI を使用して、ENA がデフォルトで有効になっている別のインスタンスを起動できます。

ENA キュー

ENA キューは、インスタンスのタイプとサイズに基づくデフォルトの静的制限を用いてネットワークインターフェイスに割り当てられます。サポートされているインスタンスタイプでは、これらのキューを Elastic Network Interface (ENI) 全体に動的に割り当てることができます。インスタンスあたりのキューの合計数はインスタンスのタイプとサイズによって異なりますが、ENI キューは、ENI とインスタンスの最大キュー数に到達するまで複数の ENI に設定できます。

柔軟な ENA キューの割り当てはリソース配分を最適化するため、vCPU を最大限に活用できます。通常、ネットワークパフォーマンスの高いワークロードには複数の ENA キューが必要です。特定のワークロードニーズに応じてキュー数を調整することで、ネットワークパフォーマンスと 1 秒あたりのパケット数 (PPS) をファインチューニングできます。例えば、ネットワーク集約型アプリケーションには CPU 集約型アプリケーションよりも多くのキューが必要になる場合があります。

トピック

- サポートされているインスタンス
- キュー数の変更

サポートされているインスタンス

以下のインスタンスは、複数の ENA キューの動的な割り当てをサポートしています。

汎用

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
M6i			
m6i.large	2	2	6
m6i.xlarge	4	4	16
m6i.2xlarge	8	8	32
m6i.4xlarge	8	16	64
m6i.8xlarge	8	32	64
m6i.12xlarge	8	32	64
m6i.16xlarge	8	32	120
m6i.24xlarge	8	32	120
m6i.32xlarge	8	32	120
M6id			
m6id.large	2	2	6
m6id.xlarge	4	4	16
m6id.2xlarge	8	8	32
m6id.4xlarge	8	16	64
m6id.8xlarge	8	32	64
m6id.12xlarge	8	32	64

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
m6id.16xlarge	8	32	120
m6id.24xlarge	8	32	120
m6id.32xlarge	8	32	120
M6idn			
m6idn.large	2	2	6
m6idn.xlarge	4	4	16
m6idn.2xlarge	8	8	32
m6idn.4xlarge	8	16	64
m6idn.8xlarge	16	32	128
m6idn.12xlarge	16	32	128
m6idn.16xlarge	16	32	240
m6idn.24xlarge	32	32	480
m6idn.32xlarge	32	32	512 *
M6in			
m6in.large	2	2	6
m6in.xlarge	4	4	16
m6in.2xlarge	8	8	32
m6in.4xlarge	8	16	64

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
m6in.8xlarge	16	32	128
m6in.12xlarge	16	32	128
m6in.16xlarge	16	32	240
m6in.24xlarge	32	32	480
m6in.32xlarge	32	32	512 *
M8a			
m8a.medium	1	1	3
m8a.large	2	2	6
m8a.xlarge	4	4	16
m8a.2xlarge	8	8	32
m8a.4xlarge	8	16	64
m8a.8xlarge	8	32	128
m8a.12xlarge	16	64	192
m8a.16xlarge	16	64	256
m8a.24xlarge	16	128	384
m8a.48xlarge	32	128	768
m8a.metal-24xl	16	128	384
m8a.metal-48xl	32	128	768

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
M8azn			
m8azn.medium	1	1	3
m8azn.large	2	2	8
m8azn.xlarge	4	4	16
m8azn.3xlarge	4	16	48
m8azn.6xlarge	8	32	96
m8azn.12xlarge	8	64	192
m8azn.24xlarge	16	128	384
m8azn.metal-12xl	8	64	192
m8azn.metal-24xl	16	128	384
M8gb			
m8gb.medium	1	1	2
m8gb.large	2	2	6
m8gb.xlarge	4	4	16
m8gb.2xlarge	8	8	32
m8gb.4xlarge	8	16	64
m8gb.8xlarge	8	32	128
m8gb.12xlarge	16	64	192

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
m8gb.16xlarge	16	64	256
m8gb.24xlarge	16	128	384
m8gb.48xlarge	32	128	768 *
m8gb.metal-24xl	32	128	768
m8gb.metal-48xl	32	128	768 *
M8gn			
m8gn.medium	1	1	2
m8gn.large	2	2	6
m8gn.xlarge	4	4	16
m8gn.2xlarge	8	8	32
m8gn.4xlarge	8	16	64
m8gn.8xlarge	8	32	128
m8gn.12xlarge	16	64	192
m8gn.16xlarge	16	64	256
m8gn.24xlarge	16	128	384
m8gn.48xlarge	32	128	768 *
m8gn.metal-24xl	32	128	768
m8gn.metal-48xl	32	128	768 *

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
M8i			
m8i.large	2	2	6
m8i.xlarge	4	4	16
m8i.2xlarge	8	8	32
m8i.4xlarge	8	16	64
m8i.8xlarge	8	32	128
m8i.12xlarge	16	64	192
m8i.16xlarge	16	64	256
m8i.24xlarge	16	128	384
m8i.32xlarge	16	128	512
m8i.48xlarge	32	128	768
m8i.96xlarge	32	128	1536
m8i.metal-48xl	32	128	768
m8i.metal-96xl	32	128	1536
M8id			
m8id.large	2	2	6
m8id.xlarge	4	4	16
m8id.2xlarge	8	8	32

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
m8id.4xlarge	8	16	64
m8id.8xlarge	8	32	128
m8id.12xlarge	16	64	192
m8id.16xlarge	16	64	256
m8id.24xlarge	16	128	384
m8id.32xlarge	16	128	512
m8id.48xlarge	32	128	768
m8id.96xlarge	32	128	1536
m8id.metal-48xl	32	128	768
m8id.metal-96xl	32	128	1536
M8i-flex			
m8i-flex.large	1	1	3
m8i-flex.xlarge	2	2	8
m8i-flex.2xlarge	4	4	16
m8i-flex.4xlarge	4	8	32
m8i-flex.8xlarge	4	16	64
m8i-flex.12xlarge	8	32	96
m8i-flex.16xlarge	8	32	128

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
M8in			
m8in.large	2	2	8
m8in.xlarge	4	4	16
m8in.2xlarge	8	8	32
m8in.4xlarge	8	16	64
m8in.8xlarge	16	32	128
m8in.12xlarge	16	64	192
m8in.16xlarge	16	64	256
m8in.24xlarge	16	128	256
m8in.32xlarge	32	128	512
m8in.48xlarge	32	128	768
m8in.96xlarge	32	128	1536 *
m8in.metal-48xl	32	128	768
m8in.metal-96xl	32	128	1536 *
M8idn			
m8idn.large	2	2	8
m8idn.xlarge	4	4	16
m8idn.2xlarge	8	8	32

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
m8idn.4xlarge	8	16	64
m8idn.8xlarge	16	32	128
m8idn.12xlarge	16	64	192
m8idn.16xlarge	16	64	256
m8idn.24xlarge	16	128	256
m8idn.32xlarge	32	128	512
m8idn.48xlarge	32	128	768
m8idn.96xlarge	32	128	1536 *
m8idn.metal-48xl	32	128	768
m8idn.metal-96xl	32	128	1536 *
M8ine			
m8ine.large	2	2	8
m8ine.xlarge	4	4	16
m8ine.2xlarge	8	8	32
m8ine.4xlarge	16	16	128
m8ine.8xlarge	32	32	256
m8ine.12xlarge	32	64	384
M8ib			

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
m8ib.large	2	2	8
m8ib.xlarge	4	4	16
m8ib.2xlarge	8	8	32
m8ib.4xlarge	8	16	64
m8ib.8xlarge	16	32	128
m8ib.12xlarge	16	64	192
m8ib.16xlarge	16	64	256
m8ib.24xlarge	16	128	256
m8ib.32xlarge	32	128	512
m8ib.48xlarge	32	128	768
m8ib.96xlarge	32	128	1536 *
m8ib.metal-48xl	32	128	768
m8ib.metal-96xl	32	128	1536 *
M8idb			
m8idb.large	2	2	8
m8idb.xlarge	4	4	16
m8idb.2xlarge	8	8	32
m8idb.4xlarge	8	16	64

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
m8idb.8xlarge	16	32	128
m8idb.12xlarge	16	64	192
m8idb.16xlarge	16	64	256
m8idb.24xlarge	16	128	256
m8idb.32xlarge	32	128	512
m8idb.48xlarge	32	128	768
m8idb.96xlarge	32	128	1536 *
m8idb.metal-48xl	32	128	768
m8idb.metal-96xl	32	128	1536 *
M9g			
m9g.medium	1	1	2
m9g.large	2	2	6
m9g.xlarge	4	4	16
m9g.2xlarge	8	8	32
m9g.4xlarge	8	16	64
m9g.8xlarge	8	32	128
m9g.12xlarge	16	64	192
m9g.16xlarge	16	64	256

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
m9g.24xlarge	16	128	384
m9g.48xlarge	32	128	768
m9g.metal-48xl	32	128	768
M9gd			
m9gd.medium	1	1	2
m9gd.large	2	2	6
m9gd.xlarge	4	4	16
m9gd.2xlarge	8	8	32
m9gd.4xlarge	8	16	64
m9gd.8xlarge	8	32	128
m9gd.12xlarge	16	64	192
m9gd.16xlarge	16	64	256
m9gd.24xlarge	16	128	384
m9gd.48xlarge	32	128	768
m9gd.metal-48xl	32	128	768

> **Note**
>
> * これらのインスタンスタイプには複数のネットワークカードが搭載されています。他のインスタンスタイプには単一のネットワークカードが搭載されています。詳細については「ネットワークカード」を参照してください。

コンピューティングの最適化

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
C6i			
c6i.large	2	2	6
c6i.xlarge	4	4	16
c6i.2xlarge	8	8	32
c6i.4xlarge	8	16	64
c6i.8xlarge	8	32	64
c6i.12xlarge	8	32	64
c6i.16xlarge	8	32	120
c6i.24xlarge	8	32	120
c6i.32xlarge	8	32	120
C6id			
c6id.large	2	2	6
c6id.xlarge	4	4	16

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
c6id.2xlarge	8	8	32
c6id.4xlarge	8	16	64
c6id.8xlarge	8	32	64
c6id.12xlarge	8	32	64
c6id.16xlarge	8	32	120
c6id.24xlarge	8	32	120
c6id.32xlarge	8	32	120
C6in			
c6in.large	2	2	6
c6in.xlarge	4	4	16
c6in.2xlarge	8	8	32
c6in.4xlarge	8	16	64
c6in.8xlarge	16	32	128
c6in.12xlarge	16	32	128
c6in.16xlarge	16	32	240
c6in.24xlarge	32	32	480
c6in.32xlarge	32	32	512 *
C8a			

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
c8a.medium	1	1	3
c8a.large	2	2	6
c8a.xlarge	4	4	16
c8a.2xlarge	8	8	32
c8a.4xlarge	8	16	64
c8a.8xlarge	8	32	128
c8a.12xlarge	16	64	192
c8a.16xlarge	16	64	256
c8a.24xlarge	16	128	384
c8a.48xlarge	32	128	768
c8a.metal-24xl	16	128	384
c8a.metal-48xl	32	128	768
C8gb			
c8gb.medium	1	1	2
c8gb.large	2	2	6
c8gb.xlarge	4	4	16
c8gb.2xlarge	8	8	32
c8gb.4xlarge	8	16	64

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
c8gb.8xlarge	8	32	128
c8gb.12xlarge	16	64	192
c8gb.16xlarge	16	64	256
c8gb.24xlarge	16	128	384
c8gb.48xlarge	32	128	768 *
c8gb.metal-24xl	32	128	768
c8gb.metal-48xl	32	128	768 *
C8gn			
c8gn.medium	1	1	2
c8gn.large	2	2	6
c8gn.xlarge	4	4	16
c8gn.2xlarge	8	8	32
c8gn.4xlarge	8	16	64
c8gn.8xlarge	8	32	128
c8gn.12xlarge	16	64	192
c8gn.16xlarge	16	64	256
c8gn.24xlarge	16	128	384
c8gn.48xlarge	32	128	768 *

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
c8gn.metal-24xl	32	128	768
c8gn.metal-48xl	32	128	768 *
C8i			
c8i.large	2	2	6
c8i.xlarge	4	4	16
c8i.2xlarge	8	8	32
c8i.4xlarge	8	16	64
c8i.8xlarge	8	32	128
c8i.12xlarge	16	64	192
c8i.16xlarge	16	64	256
c8i.24xlarge	16	128	384
c8i.32xlarge	16	128	512
c8i.48xlarge	32	128	768
c8i.96xlarge	32	128	1536
c8i.metal-48xl	32	128	768
c8i.metal-96xl	32	128	1536
C8id			
c8id.large	2	2	6

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
c8id.xlarge	4	4	16
c8id.2xlarge	8	8	32
c8id.4xlarge	8	16	64
c8id.8xlarge	8	32	128
c8id.12xlarge	16	64	192
c8id.16xlarge	16	64	256
c8id.24xlarge	16	128	384
c8id.32xlarge	16	128	512
c8id.48xlarge	32	128	768
c8id.96xlarge	32	128	1536
c8id.metal-48xl	32	128	768
c8id.metal-96xl	32	128	1536
C8i-flex			
c8i-flex.large	1	1	3
c8i-flex.xlarge	2	2	8
c8i-flex.2xlarge	4	4	16
c8i-flex.4xlarge	4	8	32
c8i-flex.8xlarge	4	16	64

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
c8i-flex.12xlarge	8	32	96
c8i-flex.16xlarge	8	32	128
C8in			
c8in.large	2	2	8
c8in.xlarge	4	4	16
c8in.2xlarge	8	8	32
c8in.4xlarge	8	16	64
c8in.8xlarge	16	32	128
c8in.12xlarge	16	64	192
c8in.16xlarge	16	64	256
c8in.24xlarge	16	128	256
c8in.32xlarge	32	128	512
c8in.48xlarge	32	128	768
c8in.96xlarge	32	128	1536 *
c8in.metal-48xl	32	128	768
c8in.metal-96xl	32	128	1536 *
C8ine			
c8ine.large	2	2	8

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
c8ine.xlarge	4	4	16
c8ine.2xlarge	8	8	32
c8ine.4xlarge	16	16	128
c8ine.8xlarge	32	32	256
c8ine.12xlarge	32	64	384
C8ib			
c8ib.large	2	2	8
c8ib.xlarge	4	4	16
c8ib.2xlarge	8	8	32
c8ib.4xlarge	8	16	64
c8ib.8xlarge	16	32	128
c8ib.12xlarge	16	64	192
c8ib.16xlarge	16	64	256
c8ib.24xlarge	16	128	256
c8ib.32xlarge	32	128	512
c8ib.48xlarge	32	128	768
c8ib.96xlarge	32	128	1536 *
c8ib.metal-48xl	32	128	768

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
c8ib.metal-96xl	32	128	1536 *
C9g			
c9g.medium	1	1	2
c9g.large	2	2	6
c9g.xlarge	4	4	16
c9g.2xlarge	8	8	32
c9g.4xlarge	8	16	64
c9g.8xlarge	8	32	128
c9g.12xlarge	16	64	192
c9g.16xlarge	16	64	256
c9g.24xlarge	16	128	384
c9g.48xlarge	32	128	768
c9g.metal-48xl	32	128	768
C9gd			
c9gd.medium	1	1	2
c9gd.large	2	2	6
c9gd.xlarge	4	4	16
c9gd.2xlarge	8	8	32

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
c9gd.4xlarge	8	16	64
c9gd.8xlarge	8	32	128
c9gd.12xlarge	16	64	192
c9gd.16xlarge	16	64	256
c9gd.24xlarge	16	128	384
c9gd.48xlarge	32	128	768
c9gd.metal-48xl	32	128	768

> **Note**
>
> * これらのインスタンスタイプには複数のネットワークカードが搭載されています。他のインスタンスタイプには単一のネットワークカードが搭載されています。詳細については「ネットワークカード」を参照してください。

メモリ最適化

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
	R6i		
r6i.large	2	2	6
r6i.xlarge	4	4	16

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
r6i.2xlarge	8	8	32
r6i.4xlarge	8	16	64
r6i.8xlarge	8	32	64
r6i.12xlarge	8	32	64
r6i.16xlarge	8	32	120
r6i.24xlarge	8	32	120
r6i.32xlarge	8	32	120
R6id			
r6id.large	2	2	6
r6id.xlarge	4	4	16
r6id.2xlarge	8	8	32
r6id.4xlarge	8	16	64
r6id.8xlarge	8	32	64
r6id.12xlarge	8	32	64
r6id.16xlarge	8	32	120
r6id.24xlarge	8	32	120
r6id.32xlarge	8	32	120
R6idn			

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
r6idn.large	2	2	6
r6idn.xlarge	4	4	16
r6idn.2xlarge	8	8	32
r6idn.4xlarge	8	16	64
r6idn.8xlarge	16	32	128
r6idn.12xlarge	16	32	128
r6idn.16xlarge	16	32	240
r6idn.24xlarge	32	32	480
r6idn.32xlarge	32	32	512 *
	R6in		
r6in.large	2	2	6
r6in.xlarge	4	4	16
r6in.2xlarge	8	8	32
r6in.4xlarge	8	16	64
r6in.8xlarge	16	32	128
r6in.12xlarge	16	32	128
r6in.16xlarge	16	32	240
r6in.24xlarge	32	32	480

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
r6in.32xlarge	32	32	512 *
R8a			
r8a.medium	1	1	3
r8a.large	2	2	6
r8a.xlarge	4	4	16
r8a.2xlarge	8	8	32
r8a.4xlarge	8	16	64
r8a.8xlarge	8	32	128
r8a.12xlarge	16	64	192
r8a.16xlarge	16	64	256
r8a.24xlarge	16	128	384
r8a.48xlarge	32	128	768
r8a.metal-24xl	16	128	384
r8a.metal-48xl	32	128	768
R8gb			
r8gb.medium	1	1	2
r8gb.large	2	2	6
r8gb.xlarge	4	4	16

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
r8gb.2xlarge	8	8	32
r8gb.4xlarge	8	16	64
r8gb.8xlarge	8	32	128
r8gb.12xlarge	16	64	192
r8gb.16xlarge	16	64	256
r8gb.24xlarge	16	128	384
r8gb.48xlarge	32	128	768 *
r8gb.metal-24xl	32	128	768
r8gb.metal-48xl	32	128	768 *
R8gn			
r8gn.medium	1	1	2
r8gn.large	2	2	6
r8gn.xlarge	4	4	16
r8gn.2xlarge	8	8	32
r8gn.4xlarge	8	16	64
r8gn.8xlarge	8	32	128
r8gn.12xlarge	16	64	192
r8gn.16xlarge	16	64	256

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
r8gn.24xlarge	16	128	384
r8gn.48xlarge	32	128	768 *
r8gn.metal-24xl	32	128	768
r8gn.metal-48xl	32	128	768 *
R8i			
r8i.large	2	2	6
r8i.xlarge	4	4	16
r8i.2xlarge	8	8	32
r8i.4xlarge	8	16	64
r8i.8xlarge	8	32	128
r8i.12xlarge	16	64	192
r8i.16xlarge	16	64	256
r8i.24xlarge	16	128	384
r8i.32xlarge	16	128	512
r8i.48xlarge	32	128	768
r8i.96xlarge	32	128	1536
r8i.metal-48xl	32	128	768
r8i.metal-96xl	32	128	1536

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
R8id			
r8id.large	2	2	6
r8id.xlarge	4	4	16
r8id.2xlarge	8	8	32
r8id.4xlarge	8	16	64
r8id.8xlarge	8	32	128
r8id.12xlarge	16	64	192
r8id.16xlarge	16	64	256
r8id.24xlarge	16	128	384
r8id.32xlarge	16	128	512
r8id.48xlarge	32	128	768
r8id.96xlarge	32	128	1536
r8id.metal-48xl	32	128	768
r8id.metal-96xl	32	128	1536
R8i-flex			
r8i-flex.large	1	1	3
r8i-flex.xlarge	2	2	8
r8i-flex.2xlarge	4	4	16

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
r8i-flex.4xlarge	4	8	32
r8i-flex.8xlarge	4	16	64
r8i-flex.12xlarge	8	32	96
r8i-flex.16xlarge	8	32	128
R8in			
r8in.large	2	2	8
r8in.xlarge	4	4	16
r8in.2xlarge	8	8	32
r8in.4xlarge	8	16	64
r8in.8xlarge	16	32	128
r8in.12xlarge	16	64	192
r8in.16xlarge	16	64	256
r8in.24xlarge	16	128	256
r8in.32xlarge	32	128	512
r8in.48xlarge	32	128	768
r8in.96xlarge	32	128	1536 *
r8in.metal-48xl	32	128	768
r8in.metal-96xl	32	128	1536 *

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
R8idn			
r8idn.large	2	2	8
r8idn.xlarge	4	4	16
r8idn.2xlarge	8	8	32
r8idn.4xlarge	8	16	64
r8idn.8xlarge	16	32	128
r8idn.12xlarge	16	64	192
r8idn.16xlarge	16	64	256
r8idn.24xlarge	16	128	256
r8idn.32xlarge	32	128	512
r8idn.48xlarge	32	128	768
r8idn.96xlarge	32	128	1536 *
r8idn.metal-48xl	32	128	768
r8idn.metal-96xl	32	128	1536 *
R8ib			
r8ib.large	2	2	8
r8ib.xlarge	4	4	16
r8ib.2xlarge	8	8	32

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
r8ib.4xlarge	8	16	64
r8ib.8xlarge	16	32	128
r8ib.12xlarge	16	64	192
r8ib.16xlarge	16	64	256
r8ib.24xlarge	16	128	256
r8ib.32xlarge	32	128	512
r8ib.48xlarge	32	128	768
r8ib.96xlarge	32	128	1536 *
r8ib.metal-48xl	32	128	768
r8ib.metal-96xl	32	128	1536 *
R8idb			
r8idb.large	2	2	8
r8idb.xlarge	4	4	16
r8idb.2xlarge	8	8	32
r8idb.4xlarge	8	16	64
r8idb.8xlarge	16	32	128
r8idb.12xlarge	16	64	192
r8idb.16xlarge	16	64	256

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
r8idb.24xlarge	16	128	256
r8idb.32xlarge	32	128	512
r8idb.48xlarge	32	128	768
r8idb.96xlarge	32	128	1536 *
r8idb.metal-48xl	32	128	768
r8idb.metal-96xl	32	128	1536 *
X8aedz			
x8aedz.large	2	2	8
x8aedz.xlarge	4	4	16
x8aedz.3xlarge	4	16	48
x8aedz.6xlarge	8	32	96
x8aedz.12xlarge	8	64	192
x8aedz.24xlarge	16	128	384
x8aedz.metal-12xl	8	64	192
x8aedz.metal-24xl	16	128	384
X8i			
x8i.large	2	2	6
x8i.xlarge	4	4	16

インスタンスタイプ	インターフェイスあたりのデフォルトの ENA キュー	インターフェイスあたりの最大 ENA キュー	インスタンスあたりの最大 ENA キュー
x8i.2xlarge	8	8	32
x8i.4xlarge	8	16	64
x8i.8xlarge	8	32	128
x8i.12xlarge	16	64	192
x8i.16xlarge	16	64	256
x8i.24xlarge	16	128	384
x8i.32xlarge	16	128	512
x8i.48xlarge	32	128	768
x8i.64xlarge	32	128	1024
x8i.96xlarge	32	128	1536
x8i.metal-48xl	32	128	768
x8i.metal-96xl	32	128	1536

Note

* これらのインスタンスタイプには複数のネットワークカードが搭載されています。他のインスタンスタイプには単一のネットワークカードが搭載されています。詳細については「ネットワークカード」を参照してください。

キュー数の変更

ENA キューの数は、AWS マネジメントコンソール、AWS CLI、PowerShell を使用して変更できます。AWS マネジメントコンソールでは、各 [ネットワークインターフェイス] 設定に ENA キューの設定があります。

> **Note**
>
> - ENA キューの数を変更する前に、インスタンスを停止しておく必要があります。
> - ENA キューの値は、2 のべき乗 (1、2、4、8、16、32 など) にする必要があります。
> - 単一の ENI に割り当てるキューの数は、インスタンスで利用できる vCPU の数を超えない必要があります。

キュー数を変更する前に、以下のコマンドを使用して現在のキュー数を確認してください。

AWS CLI

```
aws ec2 describe-instances --instance-id i-1234567890abcdef0
```

PowerShell

```
(Get-EC2Instance -InstanceId i-1234567890abcdef0).Instances.NetworkInterfaces |
  Select-Object NetworkInterfaceId,
    @{N='DeviceIndex';E={$_.Attachment.DeviceIndex}},
    @{N='AttachmentId';E={$_.Attachment.AttachmentId}},
    @{N='EnaQueueCount';E={$_.Attachment.EnaQueueCount}}
```

ENA キューでネットワークインターフェイスをアタッチする

以下の例では、ENI に 16 個の ENA キューが設定されます。

AWS CLI

ENA キューでネットワークインターフェイスをアタッチするには

attach-network-interface コマンドを使用します。

```
aws ec2 attach-network-interface \
```

```
  --network-interface-id eni-abcdef01234567890 \
  --instance-id i-1234567890abcdef0 \
  --device-index 1 \
  --ena-queue-count 16
```

PowerShell

ENA キューでネットワークインターフェイスをアタッチするには

Add-EC2NetworkInterface コマンドレットを使用します。

```
Add-EC2NetworkInterface `
  -NetworkInterfaceId eni-abcdef01234567890 `
  -InstanceId i-1234567890abcdef0 `
  -DeviceIndex 1 `
  -EnaQueueCount 16
```

ENA キューを使用してインスタンスを起動する

以下の例では、3 個の ENI にそれぞれ 16 個の ENA キューが設定されます。

AWS CLI

ENA キューを使用してインスタンスを起動するには

run-instances コマンドを使用します。

```
aws ec2 run-instances \
  --image-id ami-1234567890abcdef0 \
  --instance-type c8i.4xlarge \
  --network-interfaces \
    "[{\"DeviceIndex\":0,\"SubnetId\":\"subnet-abcdef01234567890\",\"EnaQueueCount
\":16},
      {\"DeviceIndex\":1,\"SubnetId\":\"subnet-abcdef01234567890\",\"EnaQueueCount
\":16},
      {\"DeviceIndex\":2,\"SubnetId\":\"subnet-abcdef01234567890\",\"EnaQueueCount
\":16}]"
```

PowerShell

ENA キューを使用してインスタンスを起動するには

New-EC2Instance コマンドレットを使用します。

```
New-EC2Instance `
  -ImageId ami-1234567890abcdef0 `
  -InstanceType c8i.4xlarge `
  -NetworkInterface @(
    @{DeviceIndex=0; SubnetId="subnet-abcdef01234567890"; EnaQueueCount=16},
    @{DeviceIndex=1; SubnetId="subnet-abcdef01234567890"; EnaQueueCount=16},
    @{DeviceIndex=2; SubnetId="subnet-abcdef01234567890"; EnaQueueCount=16}
  )
```

既存のネットワークインターフェイスで ENA キューを変更する

以下の例では、ENI に 16 個の ENA キューが設定されます。

AWS CLI

ネットワークインターフェイスで ENA キューを変更するには

modify-network-interface-attribute コマンドを使用します。

```
aws ec2 modify-network-interface-attribute \
  --network-interface-id eni-1234567890abcdef0 \
  --attachment AttachmentId=eni-attach-1234567890abcdef0,EnaQueueCount=16
```

PowerShell

ネットワークインターフェイスで ENA キューを変更するには

Edit-EC2NetworkInterfaceAttribute コマンドレットを使用します。

```
Edit-EC2NetworkInterfaceAttribute `
  -NetworkInterfaceId eni-1234567890abcdef0 `
  -Attachment_AttachmentId eni-attach-1234567890abcdef0 `
  -Attachment_EnaQueueCount 16
```

以下の例では、ENA 数がデフォルト値にリセットされます。

AWS CLI

```
aws ec2 modify-network-interface-attribute \
  --network-interface-id eni-1234567890abcdef0 \
```

```
  --attachment AttachmentId=eni-attach-1234567890abcdef0,DefaultEnaQueueCount=true
```

PowerShell

```
Edit-EC2NetworkInterfaceAttribute `
  -NetworkInterfaceId eni-1234567890abcdef0 `
  -Attachment_AttachmentId eni-attach-1234567890abcdef0 `
  -Attachment_DefaultEnaQueueCount $true
```

Linux での ENA カーネルドライバーのトラブルシューティング

Elastic Network Adapter (ENA) は、オペレーティング システムのヘルスを向上し、予期しないハードウェア動作や障害による長期的な停止の可能性を減らすように設計されています。ENA アーキテクチャでは、デバイスやドライバーの障害がシステムに対して可能な限り透過的に保持されます。このトピックでは、ENA のトラブルシューティングについて説明します。

インスタンスに接続できない場合は、まず接続に関する問題のトラブルシューティングセクションを参照してください。

第 6 世代のインスタンスタイプに移行した後にパフォーマンスの低下が発生した場合は、記事「ネットワークのパフォーマンスを最大限に引き出すには、EC2 インスタンスを第 6 世代インスタンスに移行する前、何をする必要がありますか?」を参照してください。

インスタンスに接続できる場合、このトピックの以降のセクションに記載されている障害検出/復旧メカニズムを使用して診断情報を収集することができます。

コンテンツ

- 接続に関する問題のトラブルシューティング
- キープアライブメカニズム
- 読み取りタイムアウトの登録
- 統計
- syslog のドライバーエラーログ
- 最適とは言えない構成に関する通知

接続に関する問題のトラブルシューティング

拡張ネットワーキングを有効化しているときに接続が失われると、ena モジュールとインスタンスの現在実行中のカーネルの互換性が保たれない可能性があります。これは、特定のカーネルバージョ

ンのモジュールをインストール (dkms を使用しないか、不適切な設定の dkms.conf ファイルを使用) したため、インスタンスカーネルが更新された場合に発生します。起動時にロードされるインスタンスカーネルにより、ena モジュールが正しくインストールされない場合、インスタンスがネットワークアダプタを認識せず、インスタンスが到達不可能になります。

PV インスタンスまたは AMI で拡張ネットワーキングを有効にすると、お使いのインスタンスにも到達できなくなります。

ENA を使用して拡張ネットワーキングを有効した後インスタンスが到達不可能になった場合、インスタンスの enaSupport 属性を無効にすると、ストックネットワークアダプタにフォールバックできます。

ENA を使用して拡張ネットワーキングを無効にするには (EBS-backed インスタンス)

1. ローカルコンピュータから、Amazon EC2 コンソール、stop-instances コマンド (AWS CLI)、Stop-EC2Instance コマンドレット (AWS Tools for PowerShell) のいずれかを使用して、インスタンスを停止します。
2. ローカルコンピュータから、modify-instance-attribute コマンドを `--no-ena-support`オプションと共に使用するか、Edit-EC2InstanceAttribute コマンドレットを `-EnaSupport $false` パラメータ共に使用して、拡張ネットワーキング属性を無効にします。
3. ローカルコンピュータから、Amazon EC2 コンソール、start-instances コマンド、Start-EC2Instance コマンドレットのいずれかを使用して、インスタンスを停止します。
4. (オプション) インスタンスに接続し、enaのステップに従って、現在のカーネルバージョンを使用して EC2 インスタンスで ENA による拡張ネットワーキングを有効にする モジュールの再インストールを試みます。

ENA を使用して拡張ネットワーキングを無効にするには (Instance store-backed インスタンス)

1. 「Amazon S3-backed AMI を作成する」の説明に従って、新しい AMI を作成します。
2. AMI を登録するときは、stop-instances コマンド (AWS CLI) に `--no-ena-support` オプションを、または Register-EC2Image コマンドレットに `-EnaSupport $false` パラメータを、必ず含めます。

キープアライブメカニズム

ENA デバイスは、キープアライブイベントを一定の速度 (通常は 1 秒に 1 回) で送信します。ENA ドライバーは、これらのキープアライブメッセージの存在を確認するウォッチドッグメカニズムを実

装します。メッセージが存在する場合、ウォッチドッグが再実装されます。存在しない場合、ドライバーはデバイスで障害が発生したと判断し、次の処理を行います。

- 現在の統計を syslog にダンプする
- ENA デバイスをリセットする
- ENA のドライバー状態をリセットする

上記のリセット手順を実行すると、トラフィックが短時間失われる可能性がありますが (TCP 接続は回復可能です)、ユーザーに影響は及びません。

ENA デバイスは、キープアライブ通知を送信しないことによりデバイスリセット手順を間接的にリクエストすることがあります。例えば、ENA デバイスが回復不可能な設定をロードした後に不明な状態になった場合などです。

リセット手順の例を以下に示します。

```
[18509.800135] ena 0000:00:07.0 eth1: Keep alive watchdog timeout. // The watchdog
 process initiates a reset
[18509.815244] ena 0000:00:07.0 eth1: Trigger reset is on
[18509.825589] ena 0000:00:07.0 eth1: tx_timeout: 0 // The driver logs the current
 statistics
[18509.834253] ena 0000:00:07.0 eth1: io_suspend: 0
[18509.842674] ena 0000:00:07.0 eth1: io_resume: 0
[18509.850275] ena 0000:00:07.0 eth1: wd_expired: 1
[18509.857855] ena 0000:00:07.0 eth1: interface_up: 1
[18509.865415] ena 0000:00:07.0 eth1: interface_down: 0
[18509.873468] ena 0000:00:07.0 eth1: admin_q_pause: 0
[18509.881075] ena 0000:00:07.0 eth1: queue_0_tx_cnt: 0
[18509.888629] ena 0000:00:07.0 eth1: queue_0_tx_bytes: 0
[18509.895286] ena 0000:00:07.0 eth1: queue_0_tx_queue_stop: 0
.......
........
[18511.280972] ena 0000:00:07.0 eth1: free uncompleted tx skb qid 3 idx 0x7 // At the
 end of the down process, the driver discards incomplete packets.
[18511.420112] [ENA_COM: ena_com_validate_version] ena device version: 0.10 //The
 driver begins its up process
[18511.420119] [ENA_COM: ena_com_validate_version] ena controller version: 0.0.1
 implementation version 1
[18511.420127] [ENA_COM: ena_com_admin_init] ena_defs : Version:[b9692e8] Build date
 [Wed Apr  6 09:54:21 IDT 2016]
[18512.252108] ena 0000:00:07.0: Device watchdog is Enabled
```

```
[18512.674877] ena 0000:00:07.0: irq 46 for MSI/MSI-X
[18512.674933] ena 0000:00:07.0: irq 47 for MSI/MSI-X
[18512.674990] ena 0000:00:07.0: irq 48 for MSI/MSI-X
[18512.675037] ena 0000:00:07.0: irq 49 for MSI/MSI-X
[18512.675085] ena 0000:00:07.0: irq 50 for MSI/MSI-X
[18512.675141] ena 0000:00:07.0: irq 51 for MSI/MSI-X
[18512.675188] ena 0000:00:07.0: irq 52 for MSI/MSI-X
[18512.675233] ena 0000:00:07.0: irq 53 for MSI/MSI-X
[18512.675279] ena 0000:00:07.0: irq 54 for MSI/MSI-X
[18512.772641] [ENA_COM: ena_com_set_hash_function] Feature 10 isn't supported
[18512.772647] [ENA_COM: ena_com_set_hash_ctrl] Feature 18 isn't supported
[18512.775945] ena 0000:00:07.0: Device reset completed successfully // The reset
 process is complete
```

読み取りタイムアウトの登録

ENA アーキテクチャでは、Memory Mapped I/O (MMIO) の読み取りオペレーションの限定的に使用することが推奨されます。MMIO レジスタには、初期化手順中のみ ENA デバイスドライバーがアクセスします。

ドライバーログ (dmesg 出力にあります) が読み取りオペレーションの失敗を示している場合、互換性のないドライバーまたは適切にコンパイルされていないドライバー、ビジー状態のハードウェアドライバー、ハードウェア障害が原因の可能性があります。

読み取りオペレーションの失敗を示すログエントリが断続的に発生する場合は、問題とみなさないでください。この場合はドライバーによって再試行されます。ただし、読み取りの失敗を含むログエントリが連続して発生する場合は、ドライバーまたはハードウェアの問題を示しています。

タイムアウトによる読み取りオペレーション失敗を示すドライバーログエントリの例を以下に示します。

```
[ 47.113698] [ENA_COM: ena_com_reg_bar_read32] reading reg failed for timeout.
 expected: req id[1] offset[88] actual: req id[57006] offset[0]
[ 47.333715] [ENA_COM: ena_com_reg_bar_read32] reading reg failed for timeout.
 expected: req id[2] offset[8] actual: req id[57007] offset[0]
[ 47.346221] [ENA_COM: ena_com_dev_reset] Reg read32 timeout occurred
```

統計

ネットワークパフォーマンスが不十分な場合やレイテンシーの問題がある場合、デバイス統計情報を取得して調査する必要があります。これらの統計は、以下に示すように ethtool を使用して取得できます。

```
[ec2-user ~]$ ethtool -S ethN
NIC statistics:
 tx_timeout: 0
 suspend: 0
 resume: 0
 wd_expired: 0
 interface_up: 1
 interface_down: 0
 admin_q_pause: 0
 bw_in_allowance_exceeded: 0
 bw_out_allowance_exceeded: 0
 pps_allowance_exceeded: 0
 conntrack_allowance_available: 450878
 conntrack_allowance_exceeded: 0
 linklocal_allowance_exceeded: 0
 queue_0_tx_cnt: 4329
 queue_0_tx_bytes: 1075749
 queue_0_tx_queue_stop: 0
 ...
```

次のコマンド出力パラメータの説明を以下に示します。

`tx_timeout`: *N*

Netdev ウォッチドッグがアクティブになった回数。

`suspend`: *N*

ドライバーが停止操作を実行した回数。

`resume`: *N*

ドライバーが再開操作を実行した回数。

`wd_expired`: *N*

ドライバーが直近 3 秒以内にキープアライブイベントを受け取らなかった回数。

`interface_up`: *N*

ENA インターフェイスが起動された回数。

`interface_down`: *N*

ENA インターフェイスが停止された回数。

admin_q_pause: *N*

管理キューが実行状態で見つからなかった回数。

bw_in_allowance_exceeded: *N*

インバウンド集計帯域幅がインスタンスの最大値を超えたためにキューまたはドロップされたパケットの数。

bw_out_allowance_exceeded: *N*

アウトバウンド集計帯域幅がインスタンスの最大値を超えたためにキューまたはドロップされたパケットの数。

pps_allowance_exceeded: *N*

双方向 PPS がインスタンスの最大値を超えたためにキューまたはドロップされたパケットの数。*

conntrack_allowance_available: *N*

そのインスタンスタイプの Connections Tracked 許容量に達する前にインスタンスが確立できる接続トラッキング数。Nitro ベースのインスタンスでのみ使用できます。FreeBSD インスタンスまたは DPDK 環境ではサポートされません。

conntrack_allowance_exceeded: *N*

接続トラッキングがインスタンスの最大数を超え、新しい接続を確立できなかったためにドロップされたパケットの数。これにより、インスタンスとの間で送受信されるトラフィックのパケット損失が発生する可能性があります。

linklocal_allowance_exceeded: *N*

ローカルプロキシサービスへのトラフィックの PPS がネットワークインターフェイスの最大値を超えたためにドロップされたパケットの数。これは、Amazon DNS サービス、インスタンスメタデータサービス、および Amazon Time Sync Service へのトラフィックに影響しますが、カスタム DNS リゾルバーへのトラフィックには影響しません。

queue_*N*_tx_cnt: *N*

このキューの送信パケット数。

queue_*N*_tx_bytes: *N*

このキューの送信バイト数。

queue_*N*_tx_queue_stop: *N*

キュー *N* がいっぱいになって停止された回数。

queue_*N*_tx_queue_wakeup: *N*

停止後にキュー *N* が再開された回数。

queue_*N*_tx_dma_mapping_err: *N*

直接メモリアクセスエラーの数。この値が 0 の場合は、システムリソースが低いことを示しています。

queue_*N*_tx_linearize: *N*

このキューに SKB 線形化が試行された回数。

queue_*N*_tx_linearize_failed: *N*

このキューで SKB 線形化が失敗した回数。

queue_*N*_tx_napi_comp: *N*

napi ハンドラーがこのキューの napi_complete を呼び出した回数。

queue_*N*_tx_tx_poll: *N*

napi ハンドラーがこのキューにスケジュールされた回数。

queue_*N*_tx_doorbells: *N*

このキューの送信ドアベルの数。

queue_*N*_tx_prepare_ctx_err: *N*

このキューで ena_com_prepare_tx が失敗した回数。

queue_*N*_tx_bad_req_id: *N*

このキューの req_id が無効です。有効な req_id は 0、マイナス queue_size、マイナス 1 です。

queue_*N*_tx_llq_buffer_copy: *N*

このキューの llq エントリよりもヘッダーサイズが大きいパケットの数。

queue_*N*_tx_missed_tx: *N*

このキューの未処理のパケット数。

queue_*N*_tx_unmask_interrupt: *N*

このキューで tx 割り込みがマスク解除された回数。

queue_*N*_rx_cnt: *N*

このキューで受信したパケット数。

queue_*N*_rx_bytes: *N*

このキューの受信バイト数。

queue_*N*_rx_rx_copybreak_pkt: *N*

rx キューが、このキューの rx_copybreak パケットサイズより小さいパケットを受信した回数。

queue_*N*_rx_csum_good: *N*

rx キューが、チェックサムをチェックし、このキューに対して正しいパケットを受信した回数。

queue_*N*_rx_refil_partial: *N*

ドライバーが rx キューの空いている部分にこのキューのバッファーを補充できなかった回数。この値が 0 でない場合、メモリリソースが低いことを示しています。

queue_*N*_rx_bad_csum: *N*

rx キューに、このキューの不良なチェックサムがあった回数 (rx チェックサムオフロードがサポートされている場合のみ)。

queue_*N*_rx_page_alloc_fail: *N*

このキューのページ割り当てに失敗した回数。この値が 0 でない場合、メモリリソースが低いことを示しています。

queue_*N*_rx_skb_alloc_fail: *N*

このキューの SKB 割り当てに失敗した回数。この値が 0 でない場合、システムリソースが低いことを示しています。

queue_*N*_rx_dma_mapping_err: *N*

直接メモリアクセスエラーの数。この値が 0 の場合は、システムリソースが低いことを示しています。

queue_*N*_rx_bad_desc_num: *N*

パケットあたりのバッファーが多すぎます。この値が 0 でない場合、バッファーの使用量が非常に少ないことを示しています。

queue_*N*_rx_bad_req_id: *N*

このキューの req_id は無効です。有効な req_id は [0, queue_size - 1] です。

queue_*N*_rx_empty_rx_ring: *N*

このキューの rx キューが空だった回数。

queue_*N*_rx_csum_unchecked: *N*

rx キューが、このキューに対してチェックサムがチェックされなかったパケットを受信した回数。

queue_*N*_rx_xdp_aborted: *N*

XDP パケットが XDP_ABORT として分類された回数。

queue_*N*_rx_xdp_drop: *N*

XDP パケットが XDP_DROP として分類された回数。

queue_*N*_rx_xdp_pass: *N*

XDP パケットが XDP_PASS として分類された回数。

queue_*N*_rx_xdp_tx: *N*

XDP パケットが XDP_TX として分類された回数。

queue_*N*_rx_xdp_invalid: *N*

パケットの XDP リターンコードが無効な回数。

queue_*N*_rx_xdp_redirect: *N*

XDP パケットが XDP_REDIRECT として分類された回数。

queue_*N*_xdp_tx_cnt: *N*

このキューの送信パケット数。

queue_*N*_xdp_tx_bytes: *N*

このキューの送信バイト数。

queue_*N*_xdp_tx_queue_stop: *N*

このキューがいっぱいになって停止した回数。

queue_*N*_xdp_tx_queue_wakeup: *N*

停止後にこのキューが再開された回数。

queue_*N*_xdp_tx_dma_mapping_err: *N*

直接メモリアクセスエラーの数。この値が 0 の場合は、システムリソースが低いことを示しています。

queue_*N*_xdp_tx_linearize: *N*

このキューに XDP バッファ線形化が試行された回数。

queue_*N*_xdp_tx_linearize_failed: *N*

このキューで XDP バッファ線形化が失敗した回数。

queue_*N*_xdp_tx_napi_comp: *N*

このキューで napi ハンドラーが napi_complete を呼び出した回数。

queue_*N*_xdp_tx_tx_poll: *N*

このキューで napi ハンドラーがスケジュールされた回数。

queue_*N*_xdp_tx_doorbells: *N*

このキューの送信ドアベルの数。

queue_*N*_xdp_tx_prepare_ctx_err: *N*

このキューで ena_com_prepare_tx が失敗した回数。この値は、常に 0 になる必要があります。そうでない場合はドライバーログを参照してください。

queue_*N*_xdp_tx_bad_req_id: *N*

このキューの req_id は無効です。有効な req_id は [0, queue_size - 1] です。

queue_*N*_xdp_tx_llq_buffer_copy: *N*

このキューの llq バッファコピーを使用してヘッダーをコピーしたパケット数。

queue_*N*_xdp_tx_missed_tx: *N*

tx キューエントリがこのキューの完了タイムアウトを逃した回数。

queue_*N*_xdp_tx_unmask_interrupt: *N*

このキューで tx 割り込みがマスク解除された回数。

ena_admin_q_aborted_cmd: *N*

中断された管理コマンドの数。これは、通常自動リカバリ手順中に発生します。

ena_admin_q_submitted_cmd: *N*

管理者キューのドアベルの数。

ena_admin_q_completed_cmd: *N*

管理者キューの完了数。

ena_admin_q_out_of_space: *N*

ドライバーが新しい管理コマンドの送信を試みたが、キューがいっぱいであった回数。

ena_admin_q_no_completion: *N*

ドライバーが管理コマンドの完了を取得しなかった回数。

syslog のドライバーエラーログ

ENA ドライバーは、システム起動時にログメッセージを syslog に書き込みます。問題が発生した場合、これらのログを調べてエラーを探すことができます。システム起動時に ENA ドライバーにより syslog に記録される情報の例と、特定のメッセージの注釈の一部を以下に示します。

```
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  478.416939] [ENA_COM:
 ena_com_validate_version] ena device version: 0.10
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  478.420915] [ENA_COM:
 ena_com_validate_version] ena controller version: 0.0.1 implementation version 1
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  479.256831] ena 0000:00:03.0: Device
 watchdog is Enabled
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  479.672947] ena 0000:00:03.0: creating 8 io
 queues. queue size: 1024
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  479.680885] [ENA_COM:
 ena_com_init_interrupt_moderation] Feature 20 isn't supported  // Interrupt moderation
 is not supported by the device
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  479.691609] [ENA_COM:
 ena_com_get_feature_ex] Feature 10 isn't supported // RSS HASH function configuration
 is not supported by the
 device
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  479.694583] [ENA_COM:
 ena_com_get_feature_ex] Feature 18 isn't supported //RSS HASH input source
 configuration is not supported by the device
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  479.697433] [ENA_COM:
 ena_com_set_host_attributes] Set host attribute isn't supported
```

```
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  479.701064] ena 0000:00:03.0 (unnamed
 net_device) (uninitialized): Cannot set host attributes
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  479.704917] ena 0000:00:03.0: Elastic
 Network Adapter (ENA) found at mem f3000000, mac addr 02:8a:3c:1e:13:b5 Queues 8
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  480.805037] EXT4-fs (xvda1): re-mounted.
 Opts: (null)
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  481.025842] NET: Registered protocol family
 10
```

無視可能なエラー

システムのエラーログに記録される可能性がある以下のエラーは、Elastic Network Adapter では無視できます。

ホスト属性の設定がサポートされない

ホスト属性は、このデバイスではサポートされていません。

rx キューのバッファの割り当てに失敗した

これは復元可能なエラーであり、エラーがスローされたときにメモリプレッシャーの問題が発生した可能性があることを示します。

機能 *X* はサポートされていない

言及されている機能は、Elastic Network Adapter ではサポートされていません。*X* に指定できる値は、以下のとおりです。

- 10: RSS ハッシュ関数設定は、このデバイスではサポートされていません。
- 12: RSS 間接テーブル設定は、このデバイスではサポートされていません。
- 18: RSS ハッシュ入力設定は、このデバイスではサポートされていません。
- 20: 割り込みモデレーションは、このデバイスではサポートされていません。
- 27: Elastic Network Adapter ドライバーは、snmpd によるイーサネット機能のポーリングをサポートしていません。

AENQ の設定に失敗した

Elastic Network Adapter では、AENQ 設定がサポートされていません。

サポートされていない AENQ のイベントを設定しようとしている

このエラーは、Elastic Network Adapter によりサポートされていない AENQ イベントグループを設定しようとしたことを示しています。

最適とは言えない構成に関する通知

ENA デバイスは、変更可能なドライバー内の最適ではない構成設定を検出します。デバイスは ENA ドライバーに通知し、コンソールに警告を記録します。次の例は、警告メッセージの形式を示しています。

```
Sub-optimal configuration notification code: 1. Refer to AWS ENA documentation for
 additional details and mitigation options.
```

次のリストは、通知コードの詳細と、最適ではない構成が検出された場合の推奨アクションを示しています。

- コード 1: ワイド LLQ 設定の ENA Express は推奨されません

  ENA Express ENI はワイド LLQ で設定されています。この構成は最適とは言えず、ENA Express のパフォーマンスに影響を与える可能性があります。ENA Express ENI を使用するときは、次のようにワイド LLQ 設定を無効にすることをお勧めします。

  ```
  sudo rmmod ena && sudo modprobe ena force_large_llq_header=0
  ```

  ENA Express の最適な構成の詳細については、「ENA Express を使用して EC2 インスタンス間のネットワークパフォーマンスを高める」を参照してください。
- コード 2: Tx キューの深さが最適ではない ENA Express ENI は推奨されません

  ENA Express ENI が最適ではない Tx キューの深さで設定されています。この設定はENA Express のパフォーマンスに影響を与える可能性があります。ENA Express ENI を使用する際は次のようにすべての Tx キューをネットワークインターフェイスの最大値に拡大することをお勧めします。

  以下の ethtool コマンドを実行すると、LLQ サイズを調整できます。Wide-LLQ を制御、クエリ、有効化する方法の詳細については、Amazon Drivers GitHub リポジトリで、ENA 用 Linux カーネルドライバードキュメントの「ラージローレイテンシーキュー (Large LLQ)」トピックを参照してください。

  ```
  ethtool -g interface
  ```

  Tx キューを最大の深度に設定します。

  ```
  ethtool -G interface tx depth
  ```

ENA Express の最適な構成の詳細については、「ENA Express を使用して EC2 インスタンス間のネットワークパフォーマンスを高める」を参照してください。

- コード 3: 通常の LLQ サイズの ENA および Tx パケットトラフィックは、ヘッダーがサポートする最大サイズを超えています

  デフォルトでは、ENA LLQ は最大 96 バイトの Tx パケットヘッダーサイズをサポートします。パケットヘッダーのサイズが 96 バイトを超えると、パケットはドロップされます。この問題を軽減するには、Wide-LLQ を有効にすることをお勧めします。これにより、サポートされる Tx パケットヘッダーサイズが最大 224 バイトに増加します。

  ただし、Wide-LLQ を有効にすると、Tx リングの最大サイズは 1000 エントリから 512 エントリに減少します。Wide-LLQ は Nitro v4 以降のすべてのインスタンスタイプでデフォルトで有効になっています。

  - Nitro v4 インスタンスタイプのデフォルトの最大 Wide-LLQ Tx リングサイズは 512 エントリで、これを変更することはできません。
  - Nitro v5 インスタンスタイプのデフォルトの最大 Wide-LLQ Tx リングサイズは 512 エントリで、最大 1000 エントリまで増加できます。

  以下の ethtool コマンドを実行すると、LLQ サイズを調整できます。Wide-LLQ を制御、クエリ、有効化する方法の詳細については、Amazon Drivers GitHub リポジトリで、ENA 用 Linux カーネルドライバードキュメントの「ラージローレイテンシーキュー (Large LLQ)」トピックを参照してください。

  Tx キューの最大深度を特定します。

```
ethtool -g interface
```

  Tx キューを最大の深度に設定します。

```
ethtool -G interface tx depth
```

エラスティックネットワークアダプタ Windows ドライバーのトラブルシューティング

エラスティックネットワークアダプタ (ENA) はオペレーティングシステムのヘルスを向上し、Windows インスタンスのオペレーションを中断させる可能性のある予期しないハードウェア動

作や障害を減らすように設計されています。ENA アーキテクチャではデバイスやドライバーの障害がオペレーティングシステムに対して可能な限り透過的に保持されます。

インスタンスの診断情報を収集する

Windows オペレーティングシステム (OS) ツールを開く手順はインスタンスにインストールされている OS のバージョンによって異なります。以下のセクションでは[Run] (実行) ダイアログでツールを開きます。このツールはすべての OS バージョンで同じ動作をします。ただし、これらのツールには任意の方法を使用してアクセスできます。

[Run] (実行) ダイアログにアクセスする

- Windows ロゴのキーの組み合わせを使用する: `Windows + R`
- 検索バーを使用する:
  - 検索バーに「`run`」と入力してください。
  - 検索結果から [Run] (実行) アプリケーションを選択してください。

一部の手順ではプロパティまたはコンテキスト依存アクションにアクセスするためにコンテキストメニューが必要です。OS のバージョンとハードウェアに応じて、いくつかの方法があります。

コンテキストメニューにアクセスする

- マウスを使用する: 項目を右クリックしてコンテキストメニューを表示します。
- キーボードを使用する:
  - お使いの OS のバージョンに応じて、`Shift + F10`、または `Ctrl + Shift + F10` を使用します。
  - キーボードにコンテキストキー (ボックス内の 3 本の水平線) がある場合は目的の項目を選択し、コンテキストキーを押します。

インスタンスに接続できる場合は次の方法を使用してトラブルシューティング用の診断情報を収集します。

ENA デバイスのステータスを確認する

Windows デバイスマネージャーを使用して ENA Windows ドライバーのステータスを確認するには次の手順に従います。

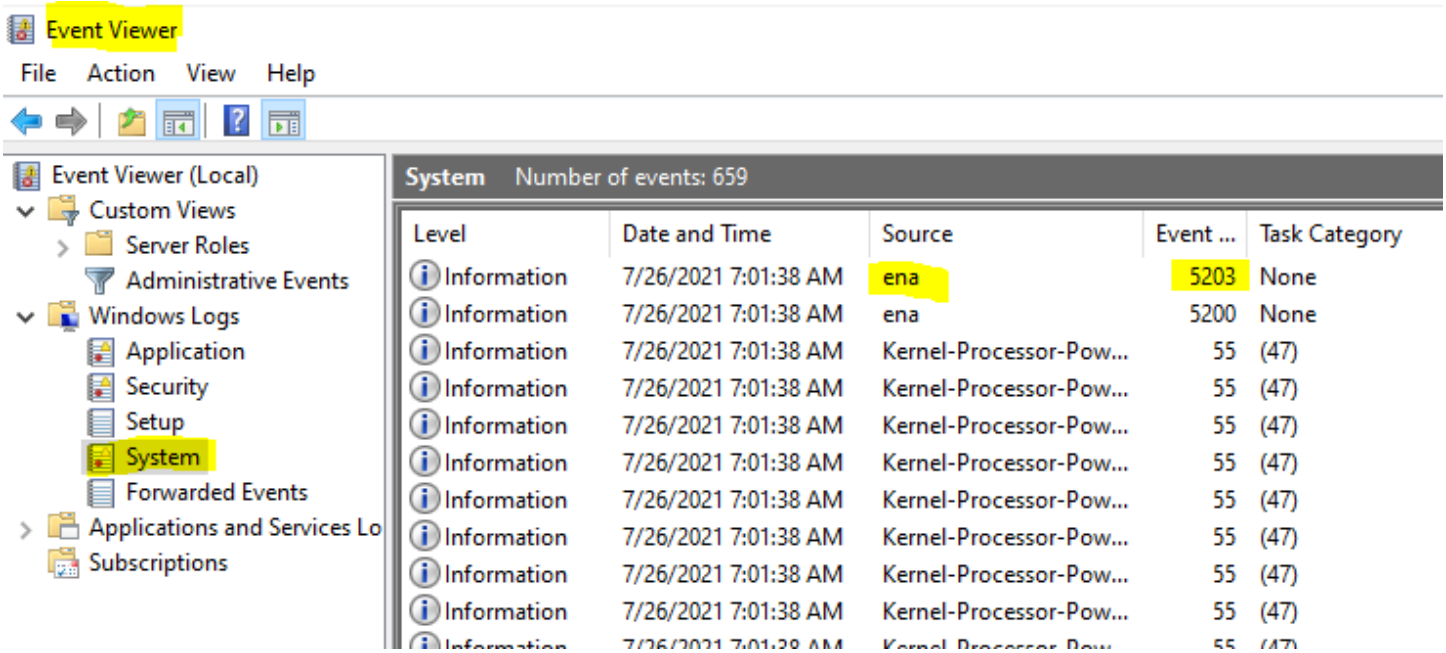
1. 前のセクションで説明されているいずれかの方法を使用して [Run] (実行) ダイアログを開きます。
2. Windows デバイスマネージャーを開くには[Run] (実行) ボックスに `devmgmt.msc` と入力してください。
3. [OK] を選択してください。デバイスマネージャーウィンドウが開きます。
4. [ネットワークアダプター] の左側にある矢印を選択してリストを展開します。
5. 名前を選択するか、[Amazonエラスティックネットワークアダプター] のコンテキストメニューを開き、[プロパティ] を選択してください。これにより、[Amazon エラスティックネットワークアダプタ のプロパティ] ダイアログが開きます。
6. [全般] タブに「このデバイスは正常に動作しています」というメッセージが表示されていることを確認します。

ドライバーイベントメッセージを調査する

Windows イベントビューアーを使用して ENA Windows ドライバーイベントログを確認するには次の手順を実行します。

1. 前のセクションで説明されているいずれかの方法を使用して [Run] (実行) ダイアログを開きます。
2. Windows イベントビューアーを開くには[Run] (実行) ボックスに `eventvwr.msc` と入力してください。
3. [OK] を選択してください。これにより、イベントビューアー ウィンドウが開きます。
4. [Windows Logs] (Windows ログ) メニューを展開し、[System] (システム) を選択してください。
5. [アクションs] (アクション) の右上のパネルで、[Filter Current Log] (現在のログをフィルター) を選択してください。これにより、フィルタリングダイアログが表示されます。
6. [Event sources] (イベントソース) ボックスに `ena` と入力してください。これにより、ENA Windows ドライバーによって生成されたイベントに結果が制限されます。
7. [OK] を選択してください。これにより、フィルターされたイベントログの結果がウィンドウの詳細セクションに表示されます。
8. 詳細をドリルダウンするにはリストからイベントメッセージを選択してください。

次の例はWindows イベントビューアーのシステムイベントリスト内の ENA ドライバーイベントを示しています。



イベントメッセージの概要

次の表に、ENA Windows ドライバーが生成するイベントメッセージを示します。

Input

Event ID	ENA ドライバーイベントの説明	タイプ
5001	ハードウェアのリソースが不足しています	エラー
5002	アダプタがハードウェアエラーを検出しました	エラー
5005	タイムリーに完了しなかった NDIS 操作でアダプタがタイムアウトしました	エラー
5032	アダプタがデバイスをリセットできませんでした	エラー
5200	アダプタが初期化されました	情報
5201	アダプタが停止されました	情報

Event ID	ENA ドライバーイベントの説明	タイプ
5202	アダプタが一時停止されました	情報
5203	アダプタが再起動されました	情報
5204	アダプタがシャットダウンされました	情報
5205	アダプタがリセットされました	エラー
5206	アダプタが突然取り外されました	エラー
5208	アダプタ初期化ルーチンが失敗しました	エラー
5210	アダプタが内部問題を検出し、正常に回復しました	エラー

パフォーマンスメトリクスを確認する

ENA Windows ドライバーはメトリクスが有効になっているインスタンスからネットワークパフォーマンスメトリクスを発行します。ネイティブのパフォーマンスモニターアプリケーションを使用して、インスタンスのメトリクスを表示および有効化できます。ENA Windows ドライバーが生成するメトリクスの詳細については[EC2 インスタンスでの ENA 設定のネットワークパフォーマンスのモニタリング]を参照してください。

ENA メトリクスが有効で、Amazon CloudWatch エージェントがインストールされているインスタンスではCloudWatch は Windows パフォーマンスモニターのカウンターに関連付けられているメトリクスと ENA の高度なメトリクスを収集します。これらのメトリクスはEC2 インスタンスでデフォルトで有効になっているメトリクスに加えて収集されます。これらのメトリクスの詳細については「Amazon CloudWatch ユーザーガイド」の[「CloudWatch エージェントにより収集されるメトリクス」]を参照してください。

> **Note**
>
> パフォーマンスメトリクスはENA ドライバーのバージョン 2.4.0 以降 (バージョン 2.2.3 でも使用可能) で使用できます。ENA ドライバーのバージョン 2.2.4 は第 6 世代 EC2 インスタンスでパフォーマンスが低下する可能性があるためロールバックされました。新バージョンに更新されていることを確認するため、ドライバーを最新バージョンにアップグレードすることをお勧めします。

パフォーマンスメトリクスを使用できる方法には次のようなものがあります。

- インスタンスのパフォーマンスの問題をトラブルシューティングします。
- ワークロードに適したインスタンスサイズを選択してください。
- スケーリングアクティビティをプロアクティブに計画します。
- アプリケーションをベンチマークして、インスタンスで使用可能なパフォーマンスを最大化するかどうかを判断します。

更新レート

デフォルトではドライバーは 1 秒間隔でメトリクスを更新します。ただし、メトリクスを取得するアプリケーションはポーリングに別の間隔を使用する場合があります。更新間隔はデバイスマネージャーで、ドライバーの詳細プロパティを使用して変更できます。

ENA Windows ドライバーのメトリクスの更新間隔を変更するには次の手順を実行します。

1. 前のセクションで説明されているいずれかの方法を使用して [Run] (実行) ダイアログを開きます。
2. Windows デバイスマネージャーを開くには[Run] (実行) ボックスに `devmgmt.msc` と入力してください。
3. [OK] を選択してください。デバイスマネージャーウィンドウが開きます。
4. [ネットワークアダプター] の左側にある矢印を選択してリストを展開します。
5. 名前を選択するか、[Amazonエラスティックネットワークアダプター] のコンテキストメニューを開き、[プロパティ] を選択してください。これにより、[Amazon エラスティックネットワークアダプタ のプロパティ] ダイアログが開きます。
6. ポップアップウィンドウで [Advanced] (詳細) タブを開きます。

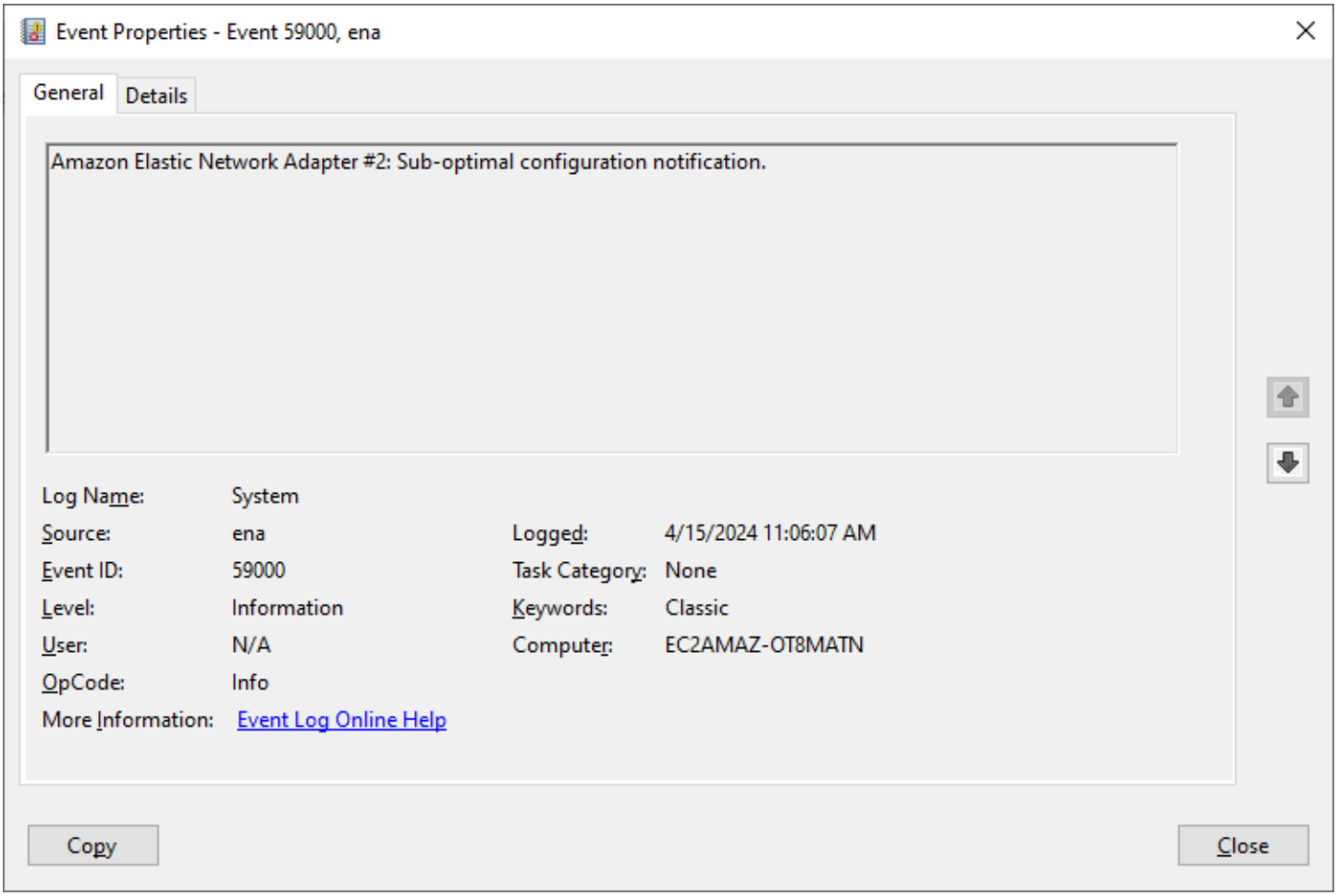
7. [Property] (プロパティ) リストから、[Metrics Refresh Interval] (メトリクス更新間隔) を選択して値を変更します。
8. 終了したら、[OK] を選択してください。

最適とは言えない設定に関する通知を確認する

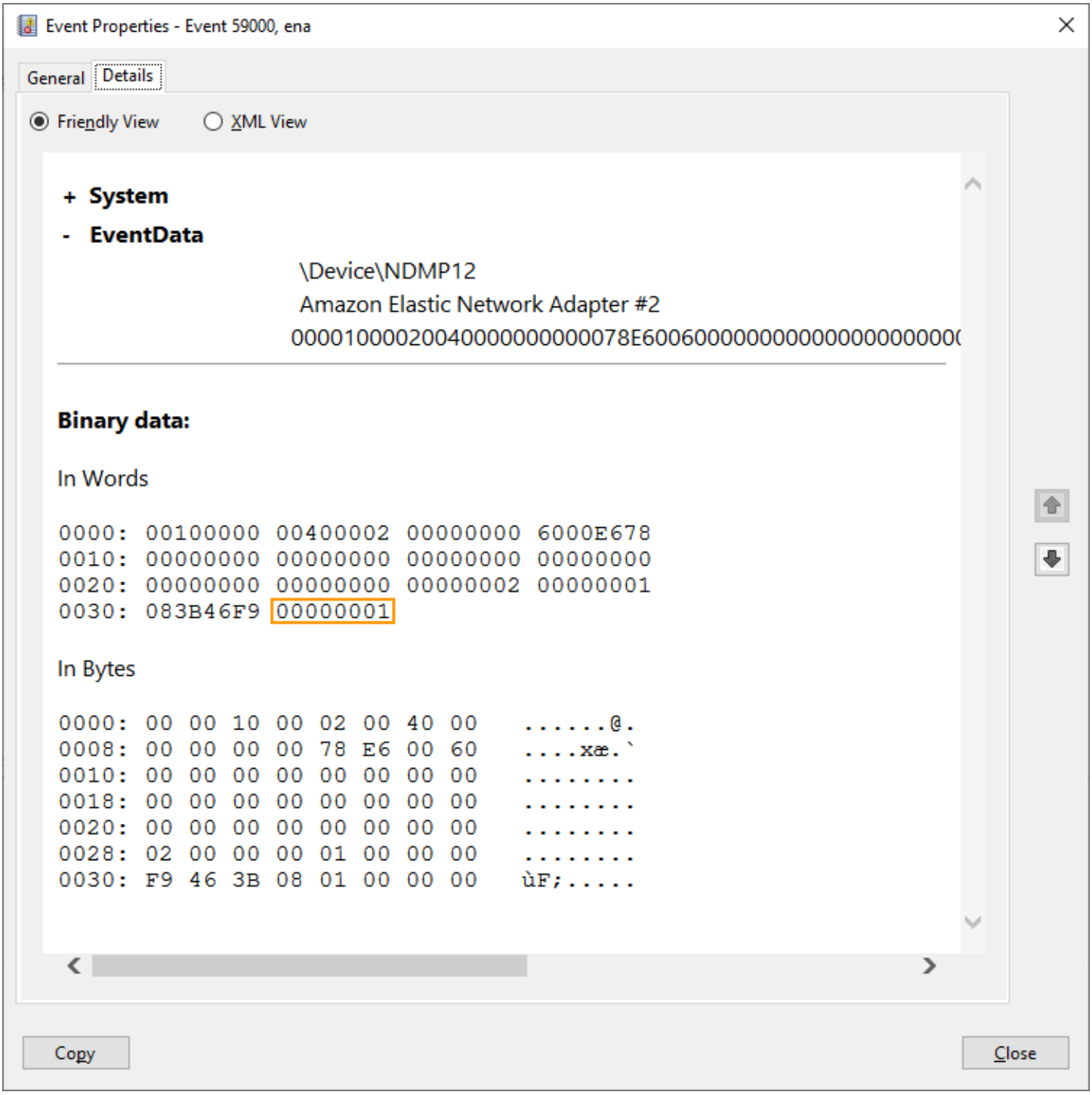
ENA デバイスは変更可能なドライバー内の最適ではない構成設定を検出します。デバイスは ENA ドライバーに通知し、イベント通知をログに記録します。Windows イベントビューアーで最適でないイベントを確認するには

1. 前のセクションで説明されているいずれかの方法を使用して [Run] (実行) ダイアログを開きます。
2. Windows イベントビューアーを開くには[Run] (実行) ボックスに `eventvwr.msc` と入力してください。
3. [OK] を選択してください。これにより、イベントビューアー ウィンドウが開きます。
4. [Windows Logs] (Windows ログ) メニューを展開し、[System] (システム) を選択してください。
5. [アクションs] (アクション) の右上のパネルで、[Filter Current Log] (現在のログをフィルター) を選択してください。これにより、フィルタリングダイアログが表示されます。
6. [Event sources] (イベントソース) ボックスに `ena` と入力してください。これにより、ENA Windows ドライバーによって生成されたイベントに結果が制限されます。
7. [OK] を選択してください。これにより、フィルターされたイベントログの結果がウィンドウの詳細セクションに表示されます。

ID `59000` のイベントは最適ではない設定に関する検出結果を通知します。イベントを右クリックして [イベントのプロパティ] を選択すると詳細ビューが開きます。または[表示] メニューから [プレビューペイン] を選択すると、同じ詳細を表示できます。



イベントコードを表示するには[詳細] タブを開きます。[バイナリデータ: 単語] セクションでは最後の単語がコードです。



次のリストは通知コードの詳細と、最適ではない構成が検出された場合の推奨アクションを示しています。

- コード **1**: ワイド LLQ 構成の ENA Express は推奨されません。

ENA Express ENI はワイド LLQ で設定されています。この構成は最適とは言えず、ENA Express のパフォーマンスに影響を与える可能性があります。ENA Express ENI を使用するときは次のようにワイド LLQ 設定を無効にすることをお勧めします。

1. Windows デバイスマネージャーを開くには[Run] (実行) ボックスに `devmgmt.msc` と入力してください。
2. [OK] を選択してください。デバイスマネージャーウィンドウが開きます。
3. [Network adapters] (ネットワークアダプタ) の左側にある矢印を選択してリストを展開します。
4. `Amazon Elastic Network Adapter` のデバイスプロパティを開きます。
5. そこから、[詳細設定] タブを開いて変更を加えます。
6. [LLQ ヘッダーサイズポリシー] プロパティを選択し、その値を `Normal (128 Bytes)` に設定します。
7. OK を選択して変更を保存します。

- コード **2**: Tx キューの深さが最適ではない ENA Express ENI は推奨されません

  ENA Express ENI が最適ではない Tx キューの深さで設定されています。この設定はENA Express のパフォーマンスに影響を与える可能性があります。ENA Express ENI を使用する際は次のようにすべての Tx キューをネットワークインターフェイスの最大値に拡大することをお勧めします。

  これらのステップに従って、Tx キューを最大の深さまで拡大します:

  1. Windows デバイスマネージャーを開くには[Run] (実行) ボックスに `devmgmt.msc` と入力してください。
  2. [OK] を選択してください。デバイスマネージャーウィンドウが開きます。
  3. [Network adapters] (ネットワークアダプタ) の左側にある矢印を選択してリストを展開します。
  4. `Amazon Elastic Network Adapter` のデバイスプロパティを開きます。
  5. そこから、[詳細設定] タブを開いて変更を加えます。
  6. [送信バッファ] プロパティを選択し、サポートされている最大値にその値を設定します。
  7. OK を選択して変更を保存します。

ENA アダプタのリセット

リセットプロセスはENA Windows ドライバーがアダプタのエラーを検出し、アダプタを異常としてマークすると開始されます。ドライバー自体をリセットできないため、アダプタのヘルスステータスを確認し、ENA Windows ドライバーのリセットハンドルを呼び出すのはオペレーティングシステムによって異なります。リセットプロセスでは短時間、トラフィック損失が発生することがあります。ただし、TCP 接続は回復できるはずです。

ENA アダプタはキープアライブ通知の送信に失敗して、間接的にデバイスのリセット手順を要求することもあります。例えば、ENA アダプタが回復不可能な設定をロードした後に不明な状態になった場合、ENA アダプタがキープアライブ通知の送信を停止することがあります。

ENA アダプタのリセットの一般的な原因

- キープアライブメッセージが見つからない

  ENA アダプタはキープアライブイベントを一定の速度 (通常は 1 秒に 1 回) で送信します。ENA Windows ドライバーはこれらのキープアライブメッセージの存在を定期的に確認するウォッチドッグメカニズムを実装します。前回チェックしてから新しいメッセージを 1 つ以上検出すると、成功した結果が記録されます。それ以外の場合、ドライバーはデバイスに障害が発生したと結論付け、リセットシーケンスを開始します。
- パケットが送信キューにスタックしている

  ENA アダプタはパケットが送信キューを予期したとおりに流れていることを確認します。ENA Windows ドライバーはパケットがスタックしているかどうかを検出し、パケットがスタックしている場合はリセットシーケンスを開始します。
- Memory Mapped I/O (MMIO) レジスターの読み取りタイムアウト

  Memory Mapped I/O (MMIO) の読み取りオペレーションを制限するために、ENA Windows ドライバーは初期化およびリセットプロセス中にのみ MMIO レジスターにアクセスします。ドライバーがタイムアウトを検出すると、実行中のプロセスに応じて、次のいずれかのアクションが実行されます。

  - 初期化中にタイムアウトが検出されると、フローが失敗し、Windows デバイスマネージャーで ENA アダプタによってドライバーに黄色の感嘆符が表示されます。
  - リセット中にタイムアウトが検出されると、フローは失敗します。その後、OS は ENA アダプタの突然の取り外しを開始し、取り外したアダプタを停止して起動することで回復します。ネットワークインターフェイスカード (NIC) の突然の取り外しの詳細については「Microsoft

Windows ハードウェア開発者向けドキュメント」の「NIC の突然の取り外しの処理」を参照してください。

トラブルシューティングシナリオ

ENA Windows ドライバーで発生する可能性のある問題のトラブルシューティングには以下のシナリオが役立ちます。最新バージョンがない場合はENA ドライバーのアップグレードから始めることをお勧めします。Windows OS のバージョン用の最新のドライバーを検索するにはENA Windows ドライバーのバージョンリリースを追跡するを参照してください。

予期しない ENA ドライバーのバージョンがインストールされました

説明

特定のバージョンの ENA ドライバーをインストールするステップを実行すると、Windows デバイスマネージャーはWindows が別のバージョンの ENA ドライバーをインストールしたことを表示します。

原因

ドライバーパッケージのインストールを実行すると、Windows は開始前にローカルドライバーストア内の特定のデバイスのために、有効なすべてのドライバーパッケージをランク付けします。その後、ランクの値が最も低いパッケージが最適なものとして選択されます。これはインストールする予定のパッケージとは異なる場合があります。デバイスドライバーパッケージの選択プロセスの詳細についてはMicrosoft ドキュメントウェブサイトの「Windows がデバイスのドライバーパッケージを選択する方法」を参照してください。

ソリューション

選択したドライバーパッケージバージョンを Windows が確実にインストールするようにするにはPnPUtil コマンドラインツールを使用して、ドライバーストアから下位ランクのドライバーパッケージを削除します。

ENA ドライバーを更新するには次のステップを実行します。

1. インスタンスに接続してローカル管理者としてログインします。
2. 「ENA デバイスのステータスを確認する」セクションの説明に従って、[Device Manager] (デバイスマネージャー) プロパティウィンドウを開きます。これにより、[Amazon エラスティックネットワークアダプタ のプロパティ] ウィンドウの [全般] タブが開きます。

3. [Driver] (ドライバー) タブを開きます。
4. [更新 Driver] を選択してください。これにより、[ドライバーソフトウェアを更新 - Amazon エラスティックネットワークアダプタ] ダイアログボックスが開きます。

   a. [ドライバーソフトウェアをどのように検索しますか?] セクションで、[コンピュータを参照してドライバーソフトウェアを探す] を選択してください。
   b. [コンピュータ上のドライバーソフトウェアを参照] ページで、検索バーの下にある [コンピュータ上のデバイスドライバーのリストから選択] を選択してください。
   c. [このハードウェア用にインストールするデバイスドライバーを選択] ページで、[ディスク使用...] を選択してください。
   d. [ディスクからインストール] ウィンドウで、ドロップダウンリストからファイルの場所の横にある [参照...] を選択してください。
   e. ターゲット ENA ドライバーパッケージをダウンロードした場所に移動します。`ena.inf` という名前のファイルを選択し、[開く] を選択してください。
   f. インストールを開始するには[OK]、[次へ] の順に選択してください。
5. インストーラーがインスタンスを自動的に再起動しない場合はRestart-Computer PowerShell コマンドレットを実行します。

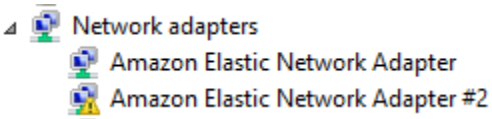
```
PS C:\> Restart-Computer
```

ENA ドライバーのデバイス警告

説明

デバイスマネージャーの [Network adapters] (ネットワークアダプタ) セクションの ENA アダプタアイコンには警告記号 (感嘆符が入った黄色の三角形) が表示されます。

次の例はWindows デバイスマネージャーで警告アイコンが表示された ENA アダプタを示しています。



原因

このデバイスの警告は一般に、環境の問題によって引き起こされ、さらに調査が必要になる可能性があり、根本的な原因を特定するために消去プロセスが必要になることがよくあります。デバイス エ

ラーの完全なリストについては、Microsoft ドキュメントの Device Manager Error Messages を参照してください。

ソリューション

このデバイス警告の解決策は根本原因によって異なります。ここで説明する消去プロセスには単純な解決策がある可能性のある最も一般的な問題を特定して解決するためのいくつかの基本的なステップが含まれています。これらのステップで問題が解決しない場合は追加の根本原因分析が必要です。

一般的な問題を特定して解決するには次のステップを実行します。

1. デバイスを停止して起動する

   「ENA デバイスのステータスを確認する」セクションの説明に従って、[Device Manager] (デバイスマネージャー) プロパティウィンドウを開きます。これにより、[Amazon エラスティックネットワークアダプタ Properties] (Amazon エラスティックネットワークアダプタ のプロパティ) ウィンドウの [General] (全般) タブが開きます。その [Device status] (デバイスの状態) にエラーコードと短いメッセージが表示されます。

   a. [Driver] (ドライバー) タブを開きます。
   b. [Disable Device] (デバイスを無効にする) を選択し、表示される警告メッセージに対して [Yes] (はい) を選択してください。
   c. [Enable Device] (デバイスを有効にする) を選択してください。
2. EC2 インスタンスを停止して起動する

   依然としてデバイスマネージャーでアダプタに警告アイコンが表示されている場合は次のステップとして、EC2 インスタンスを停止して起動します。これにより、ほとんどの場合、別のハードウェアでインスタンスが再起動されます。
3. インスタンスリソースの問題の可能性を調査する

   EC2 インスタンスを停止して起動しても問題が解決しない場合はメモリ不足など、インスタンスのリソースの問題を示している可能性があります。

アダプタのリセットによる接続タイムアウト (エラーコード 5007、5205)

説明

Windows イベントビューアーにはENA アダプタの併用で発生する、アダプタのタイムアウトイベントとリセットイベントが表示されます。メッセージは次の例のように表示されます。

- Event ID 5007 (イベント ID 5007): Amazon エラスティックネットワークアダプタ : Timed out during an operation. (操作中にタイムアウトしました。)
- Event ID 5205 (イベント ID 5205): Amazon エラスティックネットワークアダプタ : Adapter reset has been started. (アダプタのリセットが開始されました。)

アダプタのリセットにより、最小限のトラフィックの中断が発生します。複数回リセットされても、重大なネットワークの中断を引き起こすのは異常です。

原因

この一連のイベントはENA Windows ドライバーが、応答しなくなった ENA アダプタのリセットを開始したことを示します。ただし、デバイスドライバーがこの問題を検出するために使用するメカニズムはCPU 0 の枯渇による誤検出の影響を受けます。

ソリューション

このようなエラーの組み合わせが頻繁に発生する場合はリソースの割り当てを調べて、どこを調整するのがよいか確認してください。

1. 前のセクションで説明されているいずれかの方法を使用して [Run] (実行) ダイアログを開きます。
2. Windows リソースモニターを開くには[Run] (実行) ボックスに `resmon` と入力してください。
3. [OK] を選択してください。これにより、[Resource Monitor] (リソースモニター) ウィンドウが開きます。
4. [CPU] タブを開きます。CPU ごとの使用率グラフは[Resource Monitor] (リソースモニター) ウィンドウの右側に表示されます。
5. CPU 0 の使用率レベルをチェックして、それらが高すぎるかどうかを確認します。

大きなインスタンスタイプ (16 vCPU より大きい) では ENA アダプタの CPU 0 を除外するように RSS を設定することをお勧めします。インスタンスタイプが小さい場合はRSS を設定すると エクスペリエンスが向上する可能性がありますが、使用可能なコア数が少なくなるため、CPU コアの制約がパフォーマンスに悪影響を及ぼさないようにするためのテストが必要です。

以下の例に示すように、Set-NetAdapterRss コマンドを使用して ENA アダプタの RSS を設定します。

```
Set-NetAdapterRss -name (Get-NetAdapter | Where-Object {$_.InterfaceDescription -like
 "*Elastic*"}).Name -Baseprocessorgroup 0 -BaseProcessorNumber 1
```

第 6 世代のインスタンスインフラストラクチャへの移行がパフォーマンスまたはアタッチメントに影響する

説明

第 6 世代の EC2 インスタンスに移行すると、ENA Windows ドライバーのバージョンを更新していない場合、パフォーマンスが低下したり、ENA アタッチメントエラーが発生したりする可能性があります。

原因

インスタンスのオペレーティングシステム (OS) に基づき、第 6 世代の EC2 インスタンスタイプには次の ENA Windows ドライバーの最小バージョンが必要です。

最小バージョン

Windows サーバー バージョン	ENA ドライバーバージョン
Windows サーバー 2008 R2	2.2.3 または 2.4.0
Windows サーバー 2012 以降	2.2.3 以降
Windows ワークステーション	2.2.3 以降

ソリューション

第 6 世代の EC2 インスタンスにアップグレードする前に、起動する AMI に前の表に示したインスタンス OS に基づく互換性のあるドライバーがあることを確認してください。詳細についてはAWS re:Post「ナレッジセンター」の「ネットワークのパフォーマンスを最大限に引き出すにはEC2 インスタンスを第 6 世代インスタンスに移行する前、何をする必要がありますか?」を参照してください。

Elastic Network Interface の最適でないパフォーマンス

説明

ENA インターフェイスが期待どおりに動作していません。

原因

パフォーマンス問題の根本原因の分析は消去プロセスです。関連する変数が多すぎて一般的な原因を挙げることはできません。

ソリューション

根本原因の分析の最初のステップとして、期待どおりに動作していないインスタンスの診断情報を確認し、問題の原因となっている可能性のあるエラーがあるかどうかを判断します。詳細については「インスタンスの診断情報を収集する」セクションを参照してください。

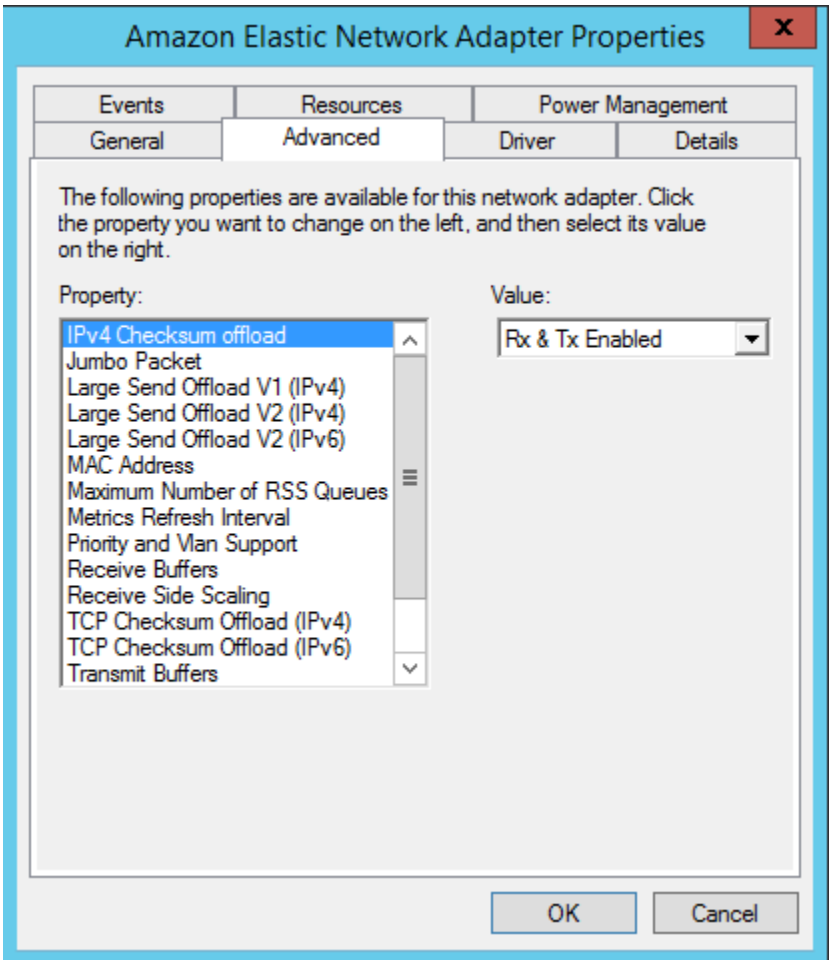
ネットワーキングが拡張されたインスタンスで最大のネットワークパフォーマンスを実現するにはデフォルトのオペレーティングシステムの設定を変更することが必要になる場合があります。いくつかの最適化 (チェックサムオフロードをオンにして RSS を有効にするなど) は公式の Windows AMI でデフォルトで設定されています。ENA アダプタに適用できるその他の最適化については「ENA アダプタのパフォーマンス調整」に示すパフォーマンス調整を参照してください。

慎重に作業を進め、デバイスプロパティの調整を、このセクションに記載されている内容や、AWS サポートチームが推奨する特定の変更に限定することをお勧めします。

ENA アダプタのプロパティを変更するには次のステップに従います。

1. 前のセクションで説明されているいずれかの方法を使用して [Run] (実行) ダイアログを開きます。
2. Windows デバイスマネージャーを開くには[Run] (実行) ボックスに `devmgmt.msc` と入力してください。
3. [OK] を選択してください。デバイスマネージャーウィンドウが開きます。
4. [ネットワークアダプター] の左側にある矢印を選択してリストを展開します。
5. 名前を選択するか、[Amazonエラスティックネットワークアダプター] のコンテキストメニューを開き、[プロパティ] を選択してください。これにより、[Amazon エラスティックネットワークアダプタ のプロパティ] ダイアログが開きます。
6. 変更するには[詳細設定] タブを開きます。
7. 完了したら、[OK] を選択して変更を保存します。

次の例はWindows デバイスマネージャーの ENA アダプタプロパティを示しています。



ENA アダプタのパフォーマンス調整

次の表に、ENA インターフェイスのパフォーマンスを向上させるために調整できるプロパティを示します。

Input

プロパティ	説明	デフォルトの値	調整
Receive Buffers	ソフトウェア受信キューのエントリ数を制御します。	1024	最大 8192 まで増やすことができます。
Receive Side Scaling (RSS)	マルチプロセッサシステムの複数の CPU	有効	負荷を複数のプロセッサに分散させる

プロパティ	説明	デフォルトの値	調整
	間でネットワーク受信処理を効率的に配分できます。		ことができます。詳細についてはEC2 Windows インスタンスでのネットワークパフォーマンスを最適化するを参照してください。

プロパティ	説明	デフォルトの値	調整
RSS キューの最大数	RSS が有効な場合に許可する RSS キューの最大数を設定します。	32	RSS キューの数はドライバーの初期化中に決定され、(特に) 次のような制限があります。 • このプロパティで設定される RSS キューの制限 • インスタンスの制限 (vCPU 数) • ハードウェアの世代の制限 (ENAv1 では最大 8 個の RSS キュー、ENAv2 では最大 32 個の RSS キュー) インスタンスとハードウェアの世代の制限に応じて、値を 1 ~ 32 に設定できます。詳細については[EC2 Windows インスタンスでのネットワークパフォーマンスを最適化する]を参照してください。

プロパティ	説明	デフォルトの値	調整
Jumbo packet	ジャンボイーサネットフレーム (1500 バイトを超えるペイロード) の使用を有効にします。	無効 (ペイロードは 1500 バイト以下に制限されます)	値は最大 9015 まで設定できます。これは 9001 バイトのペイロードに相当します。これがジャンボイーサネットフレームの最大ペイロードとなります。ジャンボイーサネットフレームの使用に関する考慮事項を参照してください。

ジャンボイーサネットフレームの使用に関する考慮事項

ジャンボフレームではパケットあたりのペイロードサイズを拡張し、パケットオーバーヘッド以外のパケットの割合を高めることによって、1500 バイトを超えるデータを送信できます。同じ量の使用可能なデータを少ないパケットで送信することができます。ただし次の場合にはトラフィックの MTU は最大 1500 に制限されます。

- EC2 Classic 用の特定の AWS リージョン外部にあるトラフィック
- 単一の VPC の外部にあるトラフィック
- リージョン間 VPC ピアリング接続経由のトラフィック
- VPN 接続経由のトラフィック
- インターネットゲートウェイ経由のトラフィック

> **Note**
>
> 1500 バイトを超えるパケットはフラグメント化されます。IP ヘッダーに `Don't Fragment` フラグが設定されている場合、それらのパケットはドロップされます。
> ジャンボフレームを、インターネットバウンドトラフィックや VPC を出るトラフィックに使用する場合には慎重に行ってください。パケットは中間システムによってフラグメント化されるため、このトラフィックの速度が低下します。VPC から出るアウトバウンドトラ

フィックに影響を与えずに VPC 内でジャンボフレームを使用するには次のいずれかのオプションを試してください。

- ルートごとに MTU サイズを設定します。
- 異なる MTU サイズと異なるルートを持つ複数のネットワークインターフェイスを使用します。

ジャンボフレームの推奨ユースケース

ジャンボフレームはVPC 内および VPC 間のトラフィックに役立ちます。次のユースケースにはジャンボフレームを使用することをお勧めします。

- クラスタープレイスメントグループ内にコロケーションされたインスタンスでは考えられる最大のネットワークスループットを実現するうえでジャンボフレームが役立ちます。詳細についてはAmazon EC2 インスタンスのプレイスメントグループを参照してください。
- Direct Connect を経由した VPC とオンプレミスのネットワーク間のトラフィックにはジャンボフレームを使用できます。Direct Connect の使用、およびジャンボフレーム機能の検証の詳細についてはDirect Connect「ユーザーガイド」の「プライベート仮想インターフェイスまたはトランジット仮想インターフェイスのネットワーク MTU の設定」を参照してください。
- トランジットゲートウェイでサポートされる MTU サイズの詳細については「Amazon VPC トランジットゲートウェイ」の「トランジットゲートウェイ のクォータ」を参照してください。

ENA Express を使用して EC2 インスタンス間のネットワークパフォーマンスを高める

ENA Express は、AWS Scalable Reliable Datagram (SRD) テクノロジーを搭載しています。SRD は、動的ルーティングを使用してスループットを向上させ、テールレイテンシーを最小限に抑える高性能なネットワークトランスポートプロトコルです。ENA Express を使用すると、同じアベイラビリティゾーン内、または同じリージョン内の異なるアベイラビリティゾーンにある 2 つの EC2 インスタンス間で通信を行うことができます。

ENA Express の利点

- 同一リージョン内において、単一フローが利用可能な最大帯域幅を 5Gbps から 25Gbps まで引き上げ、インスタンス全体の合計上限値まで適用可能にします。

- 同じアベイラビリティゾーン内の EC2 インスタンス間のネットワークトラフィックのテールレイテンシを低減します。特にネットワーク負荷が高い時間帯において効果的です。
- 混雑したネットワークパスを検出して回避します。
- 受信側でのパケットの並べ替えや、必要とされるほとんどの再送信など、一部のタスクをネットワーク層で直接処理します。これにより、アプリケーション層が解放され、他の作業に充てることができるようになります。

> **Note**
>
> - アプリケーションで 1 秒あたりのパケット数が多く、輻輳していない時間帯の遅延を最適化する必要がある場合は、拡張ネットワーク の方が適しているかもしれません。
> - ローカルゾーン内で ENA Express トラフィックを送信することはできません。
> - 南米 (サンパウロ)、中東 (バーレーン)、および中東 (UAE) では、アベイラビリティゾーン間のトラフィックに対する ENA Express のサポートは利用できません。

インスタンスのネットワークインターフェイスアタッチメントで ENA Express を有効にした後、送信側インスタンスは受信側インスタンスとの通信を開始し、SRD は 送信側インスタンスと受信側インスタンスの両方で ENA Express が動作しているかどうかを検出します。ENA Express が動作している場合、通信に SRD 送信を使用できます。ENA Express が動作していない場合、通信は標準の ENA 送信にフォールバックします。

ネットワークトラフィックが少ない時間帯には、パケットが ENA Express を使用する場合、パケットの遅延の中央値がわずかに増加 (数十マイクロ秒) することがあります。このような場合、特定のネットワークパフォーマンス特性を優先するアプリケーションには、次のような ENA Express の利点があります。

- 同一リージョン内において、インスタンスの合計上限まで、最大単一フロー帯域幅が 5 Gbps から 25 Gbps に拡大されることで、プロセスのパフォーマンス向上が期待できます。例えば、特定のインスタンスタイプが最大 12.5 Gbps までサポートする場合、シングルフローの帯域幅も 12.5 Gbps までに制限されます。
- 同じアベイラビリティゾーン内で長時間実行されているプロセスでは、ネットワークが混雑している期間中、テールレイテンシが低減されます。
- プロセスには、ネットワークの応答時間をよりスムーズに、より標準的にディストリビューションできるという利点があります。

トピック

- ENA Express の仕組み
- ENA Express でサポートされるインスタンスタイプ
- Linux インスタンスで ENA Express 設定のパフォーマンスをチューニングする
- EC2 インスタンスの ENA Express 設定を確認する
- ご使用の EC2 インスタンスに適した ENA Express を設定する

ENA Express の仕組み

ENA Express は、AWS スケーラブルで信頼性の高いデータグラム (SRD) テクノロジーを搭載しています。各ネットワークフローのパケットをさまざまな AWS ネットワークパスに分散し、輻輳の兆候を検出すると配信を動的に調整します。また、受信側でのパケットの並べ替えも管理します。

ENA Express がネットワークトラフィックを意図したとおりに管理できるようにするには、送受信インスタンスと受信側インスタンス間の通信が次の要件をすべて満たしている必要があります。

- 送信側と受信側の両方のインスタンスタイプがサポートされています。詳細については、ENA Express でサポートされるインスタンスタイプの表を参照してください。
- 送信側と受信側の両方のインスタンスに ENA Express が設定されている必要があります。設定に違いがあると、トラフィックがデフォルトで標準の ENA 送信になる状況が発生する可能性があります。発生し得る状況を次のシナリオで説明します。

  シナリオ: 設定の違い

インスタンス	ENA Express が有効になっている	UDP は ENA Express を使用する
インスタンス 1	はい	はい
インスタンス 2	はい	いいえ

  この場合、2 つのインスタンスが ENA Express を有効にするので、両方のインスタンス間の TCP トラフィックで、ENA Express を使用できます。ただし、一方のインスタンスは UDP トラフィックに ENA Express を使用しないため、これら 2 つのインスタンス間の UDP 経由の通信には標準の ENA 送信が使用されます。
- 送信側と受信側のインスタンスは、同じリージョンで実行されている必要があります。

- インスタンス間のネットワークパスには、ミドルウェアボックスを含めないようにしてください。ENA Express は現在、ミドルウェアボックスをサポートしていません。
- (Linux インスタンスのみ) 帯域幅を最大限に活用するには、ドライバーバージョン 2.2.9 以降を使用します。
- (Linux インスタンスのみ) メトリクスを生成するには、ドライバーバージョン 2.8 以降を使用します。

いずれかの要件が満たされていない場合、インスタンスは標準の TCP/UDP プロトコルを使用して通信しますが、SRD は使用しません。

インスタンスのネットワークドライバーが最適なパフォーマンスを発揮できるように構成するには、ENA ドライバーの推奨ベストプラクティスを確認してください。これらのベストプラクティスは ENA Express にも当てはまります。詳細については、GitHub ウェブサイトの「ENA Linux ドライバーのベストプラクティスとパフォーマンス最適化ガイド」を参照してください。

> **Note**
>
> Amazon EC2 では、インスタンスとそれにアタッチされたネットワークインターフェイスとの関係をアタッチメントと呼びます。ENA Express の設定がアタッチメントに適用されます。ネットワークインターフェースがインスタンスからデタッチされると、アタッチメントは存在しなくなり、そのアタッチメントに適用されていた ENA Express 設定は無効になります。ネットワークインターフェースが残っていても、インスタンスが終了した場合、同様になります。

送信側と受信側の両方のインスタンスでネットワークインターフェイスをアタッチできるように ENA Express を有効にすると、インスタンスが SRD テクノロジーで実現されるパフォーマンス向上を全面的に享受できることを、ENA Express メトリクスを使用して容易に確認できます。ENA Express のメトリクスに関する詳細は、ENA Express のメトリクスを参照してください。

ENA Express でサポートされるインスタンスタイプ

次のインスタンスタイプが ENA Express をサポートしています。

General purpose

インスタンスタイプ	アーキテクチャ
m6a.12xlarge	x86_64
m6a.16xlarge	x86_64
m6a.24xlarge	x86_64
m6a.32xlarge	x86_64
m6a.48xlarge	x86_64
m6a.metal	x86_64
m6i.8xlarge	x86_64
m6i.12xlarge	x86_64
m6i.16xlarge	x86_64
m6i.24xlarge	x86_64
m6i.32xlarge	x86_64
m6i.metal	x86_64
m6id.8xlarge	x86_64
m6id.12xlarge	x86_64
m6id.16xlarge	x86_64
m6id.24xlarge	x86_64
m6id.32xlarge	x86_64
m6id.metal	x86_64
m6idn.8xlarge	x86_64

インスタンスタイプ	アーキテクチャ
m6idn.12xlarge	x86_64
m6idn.16xlarge	x86_64
m6idn.24xlarge	x86_64
m6idn.32xlarge	x86_64
m6idn.metal	x86_64
m6in.8xlarge	x86_64
m6in.12xlarge	x86_64
m6in.16xlarge	x86_64
m6in.24xlarge	x86_64
m6in.32xlarge	x86_64
m6in.metal	x86_64
m7a.12xlarge	x86_64
m7a.16xlarge	x86_64
m7a.24xlarge	x86_64
m7a.32xlarge	x86_64
m7a.48xlarge	x86_64
m7a.metal-48xl	x86_64
m7g.12xlarge	arm64
m7g.16xlarge	arm64
m7g.metal	arm64

インスタンスタイプ	アーキテクチャ
m7gd.12xlarge	arm64
m7gd.16xlarge	arm64
m7gd.metal	arm64
m7i.12xlarge	x86_64
m7i.16xlarge	x86_64
m7i.24xlarge	x86_64
m7i.48xlarge	x86_64
m7i.metal-24xl	x86_64
m7i.metal-48xl	x86_64
m8a.16xlarge	x86_64
m8a.24xlarge	x86_64
m8a.48xlarge	x86_64
m8a.metal-24xl	x86_64
m8a.metal-48xl	x86_64
m8azn.12xlarge	x86_64
m8azn.24xlarge	x86_64
m8azn.metal-12xl	x86_64
m8azn.metal-24xl	x86_64
m8g.8xlarge	arm64
m8g.12xlarge	arm64

インスタンスタイプ	アーキテクチャ
m8g.16xlarge	arm64
m8g.24xlarge	arm64
m8g.48xlarge	arm64
m8g.metal-24xl	arm64
m8g.metal-48xl	arm64
m8gb.4xlarge	arm64
m8gb.8xlarge	arm64
m8gb.12xlarge	arm64
m8gb.16xlarge	arm64
m8gb.24xlarge	arm64
m8gb.48xlarge	arm64
m8gb.metal-24xl	arm64
m8gb.metal-48xl	arm64
m8gd.8xlarge	arm64
m8gd.12xlarge	arm64
m8gd.16xlarge	arm64
m8gd.24xlarge	arm64
m8gd.48xlarge	arm64
m8gd.metal-24xl	arm64
m8gd.metal-48xl	arm64

インスタンスタイプ	アーキテクチャ
m8gn.4xlarge	arm64
m8gn.8xlarge	arm64
m8gn.12xlarge	arm64
m8gn.16xlarge	arm64
m8gn.24xlarge	arm64
m8gn.48xlarge	arm64
m8gn.metal-24xl	arm64
m8gn.metal-48xl	arm64
m8i.8xlarge	x86_64
m8i.12xlarge	x86_64
m8i.16xlarge	x86_64
m8i.24xlarge	x86_64
m8i.32xlarge	x86_64
m8i.48xlarge	x86_64
m8i.96xlarge	x86_64
m8i.metal-48xl	x86_64
m8i.metal-96xl	x86_64
m8id.8xlarge	x86_64
m8id.12xlarge	x86_64
m8id.16xlarge	x86_64

インスタンスタイプ	アーキテクチャ
m8id.24xlarge	x86_64
m8id.32xlarge	x86_64
m8id.48xlarge	x86_64
m8id.96xlarge	x86_64
m8id.metal-48xl	x86_64
m8id.metal-96xl	x86_64
m8in.8xlarge	x86_64
m8in.12xlarge	x86_64
m8in.16xlarge	x86_64
m8in.24xlarge	x86_64
m8in.32xlarge	x86_64
m8in.48xlarge	x86_64
m8in.96xlarge	x86_64
m8in.metal-48xl	x86_64
m8in.metal-96xl	x86_64
m8idn.8xlarge	x86_64
m8idn.12xlarge	x86_64
m8idn.16xlarge	x86_64
m8idn.24xlarge	x86_64
m8idn.32xlarge	x86_64

インスタンスタイプ	アーキテクチャ
m8idn.48xlarge	x86_64
m8idn.96xlarge	x86_64
m8idn.metal-48xl	x86_64
m8idn.metal-96xl	x86_64
m8ib.8xlarge	x86_64
m8ib.12xlarge	x86_64
m8ib.16xlarge	x86_64
m8ib.24xlarge	x86_64
m8ib.32xlarge	x86_64
m8ib.48xlarge	x86_64
m8ib.96xlarge	x86_64
m8ib.metal-48xl	x86_64
m8ib.metal-96xl	x86_64
m8idb.8xlarge	x86_64
m8idb.12xlarge	x86_64
m8idb.16xlarge	x86_64
m8idb.24xlarge	x86_64
m8idb.32xlarge	x86_64
m8idb.48xlarge	x86_64
m8idb.96xlarge	x86_64

インスタンスタイプ	アーキテクチャ
m8idb.metal-48xl	x86_64
m8idb.metal-96xl	x86_64
m9g.8xlarge	arm64
m9g.12xlarge	arm64
m9g.16xlarge	arm64
m9g.24xlarge	arm64
m9g.48xlarge	arm64
m9g.metal-48xl	arm64
m9gd.8xlarge	arm64
m9gd.12xlarge	arm64
m9gd.16xlarge	arm64
m9gd.24xlarge	arm64
m9gd.48xlarge	arm64
m9gd.metal-48xl	arm64

Compute optimized

インスタンスタイプ	アーキテクチャ
c6a.12xlarge	x86_64
c6a.16xlarge	x86_64
c6a.24xlarge	x86_64

インスタンスタイプ	アーキテクチャ
c6a.32xlarge	x86_64
c6a.48xlarge	x86_64
c6a.metal	x86_64
c6gn.4xlarge	arm64
c6gn.8xlarge	arm64
c6gn.12xlarge	arm64
c6gn.16xlarge	arm64
c6i.8xlarge	x86_64
c6i.12xlarge	x86_64
c6i.16xlarge	x86_64
c6i.24xlarge	x86_64
c6i.32xlarge	x86_64
c6i.metal	x86_64
c6id.8xlarge	x86_64
c6id.12xlarge	x86_64
c6id.16xlarge	x86_64
c6id.24xlarge	x86_64
c6id.32xlarge	x86_64
c6id.metal	x86_64
c6in.8xlarge	x86_64

インスタンスタイプ	アーキテクチャ
c6in.12xlarge	x86_64
c6in.16xlarge	x86_64
c6in.24xlarge	x86_64
c6in.32xlarge	x86_64
c6in.metal	x86_64
c7a.12xlarge	x86_64
c7a.16xlarge	x86_64
c7a.24xlarge	x86_64
c7a.32xlarge	x86_64
c7a.48xlarge	x86_64
c7a.metal-48xl	x86_64
c7g.12xlarge	arm64
c7g.16xlarge	arm64
c7g.metal	arm64
c7gd.12xlarge	arm64
c7gd.16xlarge	arm64
c7gd.metal	arm64
c7gn.4xlarge	arm64
c7gn.8xlarge	arm64
c7gn.12xlarge	arm64

インスタンスタイプ	アーキテクチャ
c7gn.16xlarge	arm64
c7gn.metal	arm64
c7i.12xlarge	x86_64
c7i.16xlarge	x86_64
c7i.24xlarge	x86_64
c7i.48xlarge	x86_64
c7i.metal-24xl	x86_64
c7i.metal-48xl	x86_64
c8a.16xlarge	x86_64
c8a.24xlarge	x86_64
c8a.48xlarge	x86_64
c8a.metal-24xl	x86_64
c8a.metal-48xl	x86_64
c8g.8xlarge	arm64
c8g.12xlarge	arm64
c8g.16xlarge	arm64
c8g.24xlarge	arm64
c8g.48xlarge	arm64
c8g.metal-24xl	arm64
c8g.metal-48xl	arm64

インスタンスタイプ	アーキテクチャ
c8gb.4xlarge	arm64
c8gb.8xlarge	arm64
c8gb.12xlarge	arm64
c8gb.16xlarge	arm64
c8gb.24xlarge	arm64
c8gb.48xlarge	arm64
c8gb.metal-24xl	arm64
c8gb.metal-48xl	arm64
c8gd.8xlarge	arm64
c8gd.12xlarge	arm64
c8gd.16xlarge	arm64
c8gd.24xlarge	arm64
c8gd.48xlarge	arm64
c8gd.metal-24xl	arm64
c8gd.metal-48xl	arm64
c8gn.4xlarge	arm64
c8gn.8xlarge	arm64
c8gn.12xlarge	arm64
c8gn.16xlarge	arm64
c8gn.24xlarge	arm64

インスタンスタイプ	アーキテクチャ
c8gn.48xlarge	arm64
c8gn.metal-24xl	arm64
c8gn.metal-48xl	arm64
c8i.8xlarge	x86_64
c8i.12xlarge	x86_64
c8i.16xlarge	x86_64
c8i.24xlarge	x86_64
c8i.32xlarge	x86_64
c8i.48xlarge	x86_64
c8i.96xlarge	x86_64
c8i.metal-48xl	x86_64
c8i.metal-96xl	x86_64
c8id.8xlarge	x86_64
c8id.12xlarge	x86_64
c8id.16xlarge	x86_64
c8id.24xlarge	x86_64
c8id.32xlarge	x86_64
c8id.48xlarge	x86_64
c8id.96xlarge	x86_64
c8id.metal-48xl	x86_64

インスタンスタイプ	アーキテクチャ
c8id.metal-96xl	x86_64
c8in.8xlarge	x86_64
c8in.12xlarge	x86_64
c8in.16xlarge	x86_64
c8in.24xlarge	x86_64
c8in.32xlarge	x86_64
c8in.48xlarge	x86_64
c8in.96xlarge	x86_64
c8in.metal-48xl	x86_64
c8in.metal-96xl	x86_64
c8ib.8xlarge	x86_64
c8ib.12xlarge	x86_64
c8ib.16xlarge	x86_64
c8ib.24xlarge	x86_64
c8ib.32xlarge	x86_64
c8ib.48xlarge	x86_64
c8ib.96xlarge	x86_64
c8ib.metal-48xl	x86_64
c8ib.metal-96xl	x86_64
c9g.8xlarge	arm64

インスタンスタイプ	アーキテクチャ
c9g.12xlarge	arm64
c9g.16xlarge	arm64
c9g.24xlarge	arm64
c9g.48xlarge	arm64
c9g.metal-48xl	arm64
c9gd.8xlarge	arm64
c9gd.12xlarge	arm64
c9gd.16xlarge	arm64
c9gd.24xlarge	arm64
c9gd.48xlarge	arm64
c9gd.metal-48xl	arm64

Memory optimized

インスタンスタイプ	アーキテクチャ
r6a.12xlarge	x86_64
r6a.16xlarge	x86_64
r6a.24xlarge	x86_64
r6a.32xlarge	x86_64
r6a.48xlarge	x86_64
r6a.metal	x86_64

インスタンスタイプ	アーキテクチャ
r6i.8xlarge	x86_64
r6i.12xlarge	x86_64
r6i.16xlarge	x86_64
r6i.24xlarge	x86_64
r6i.32xlarge	x86_64
r6i.metal	x86_64
r6id.8xlarge	x86_64
r6id.12xlarge	x86_64
r6id.16xlarge	x86_64
r6id.24xlarge	x86_64
r6id.32xlarge	x86_64
r6id.metal	x86_64
r6idn.8xlarge	x86_64
r6idn.12xlarge	x86_64
r6idn.16xlarge	x86_64
r6idn.24xlarge	x86_64
r6idn.32xlarge	x86_64
r6idn.metal	x86_64
r6in.8xlarge	x86_64
r6in.12xlarge	x86_64

インスタンスタイプ	アーキテクチャ
r6in.16xlarge	x86_64
r6in.24xlarge	x86_64
r6in.32xlarge	x86_64
r6in.metal	x86_64
r7a.12xlarge	x86_64
r7a.16xlarge	x86_64
r7a.24xlarge	x86_64
r7a.32xlarge	x86_64
r7a.48xlarge	x86_64
r7a.metal-48xl	x86_64
r7g.12xlarge	arm64
r7g.16xlarge	arm64
r7g.metal	arm64
r7gd.12xlarge	arm64
r7gd.16xlarge	arm64
r7gd.metal	arm64
r7i.12xlarge	x86_64
r7i.16xlarge	x86_64
r7i.24xlarge	x86_64
r7i.48xlarge	x86_64

インスタンスタイプ	アーキテクチャ
r7i.metal-24xl	x86_64
r7i.metal-48xl	x86_64
r7iz.8xlarge	x86_64
r7iz.12xlarge	x86_64
r7iz.16xlarge	x86_64
r7iz.32xlarge	x86_64
r7iz.metal-16xl	x86_64
r7iz.metal-32xl	x86_64
r8a.16xlarge	x86_64
r8a.24xlarge	x86_64
r8a.48xlarge	x86_64
r8a.metal-24xl	x86_64
r8a.metal-48xl	x86_64
r8g.8xlarge	arm64
r8g.12xlarge	arm64
r8g.16xlarge	arm64
r8g.24xlarge	arm64
r8g.48xlarge	arm64
r8g.metal-24xl	arm64
r8g.metal-48xl	arm64

インスタンスタイプ	アーキテクチャ
r8gb.4xlarge	arm64
r8gb.8xlarge	arm64
r8gb.12xlarge	arm64
r8gb.16xlarge	arm64
r8gb.24xlarge	arm64
r8gb.48xlarge	arm64
r8gb.metal-24xl	arm64
r8gb.metal-48xl	arm64
r8gd.8xlarge	arm64
r8gd.12xlarge	arm64
r8gd.16xlarge	arm64
r8gd.24xlarge	arm64
r8gd.48xlarge	arm64
r8gd.metal-24xl	arm64
r8gd.metal-48xl	arm64
r8gn.4xlarge	arm64
r8gn.8xlarge	arm64
r8gn.12xlarge	arm64
r8gn.16xlarge	arm64
r8gn.24xlarge	arm64

インスタンスタイプ	アーキテクチャ
r8gn.48xlarge	arm64
r8gn.metal-24xl	arm64
r8gn.metal-48xl	arm64
r8i.8xlarge	x86_64
r8i.12xlarge	x86_64
r8i.16xlarge	x86_64
r8i.24xlarge	x86_64
r8i.32xlarge	x86_64
r8i.48xlarge	x86_64
r8i.96xlarge	x86_64
r8i.metal-48xl	x86_64
r8i.metal-96xl	x86_64
r8id.8xlarge	x86_64
r8id.12xlarge	x86_64
r8id.16xlarge	x86_64
r8id.24xlarge	x86_64
r8id.32xlarge	x86_64
r8id.48xlarge	x86_64
r8id.96xlarge	x86_64
r8id.metal-48xl	x86_64

インスタンスタイプ	アーキテクチャ
r8id.metal-96xl	x86_64
r8in.8xlarge	x86_64
r8in.12xlarge	x86_64
r8in.16xlarge	x86_64
r8in.24xlarge	x86_64
r8in.32xlarge	x86_64
r8in.48xlarge	x86_64
r8in.96xlarge	x86_64
r8in.metal-48xl	x86_64
r8in.metal-96xl	x86_64
r8idn.8xlarge	x86_64
r8idn.12xlarge	x86_64
r8idn.16xlarge	x86_64
r8idn.24xlarge	x86_64
r8idn.32xlarge	x86_64
r8idn.48xlarge	x86_64
r8idn.96xlarge	x86_64
r8idn.metal-48xl	x86_64
r8idn.metal-96xl	x86_64
r8ib.8xlarge	x86_64

インスタンスタイプ	アーキテクチャ
r8ib.12xlarge	x86_64
r8ib.16xlarge	x86_64
r8ib.24xlarge	x86_64
r8ib.32xlarge	x86_64
r8ib.48xlarge	x86_64
r8ib.96xlarge	x86_64
r8ib.metal-48xl	x86_64
r8ib.metal-96xl	x86_64
r8idb.8xlarge	x86_64
r8idb.12xlarge	x86_64
r8idb.16xlarge	x86_64
r8idb.24xlarge	x86_64
r8idb.32xlarge	x86_64
r8idb.48xlarge	x86_64
r8idb.96xlarge	x86_64
r8idb.metal-48xl	x86_64
r8idb.metal-96xl	x86_64
u7i-6tb.112xlarge	x86_64
u7i-8tb.112xlarge	x86_64
u7i-12tb.224xlarge	x86_64

インスタンスタイプ	アーキテクチャ
u7in-16tb.224xlarge	x86_64
u7in-24tb.224xlarge	x86_64
u7in-32tb.224xlarge	x86_64
u7inh-32tb.480xlarge	x86_64
x2idn.16xlarge	x86_64
x2idn.24xlarge	x86_64
x2idn.32xlarge	x86_64
x2idn.metal	x86_64
x2iedn.8xlarge	x86_64
x2iedn.16xlarge	x86_64
x2iedn.24xlarge	x86_64
x2iedn.32xlarge	x86_64
x2iedn.metal	x86_64
x8g.8xlarge	arm64
x8g.12xlarge	arm64
x8g.16xlarge	arm64
x8g.24xlarge	arm64
x8g.48xlarge	arm64
x8g.metal-24xl	arm64
x8g.metal-48xl	arm64

インスタンスタイプ	アーキテクチャ
x8aedz.24xlarge	x86_64
x8aedz.metal-24xl	x86_64
x8i.8xlarge	x86_64
x8i.12xlarge	x86_64
x8i.16xlarge	x86_64
x8i.24xlarge	x86_64
x8i.32xlarge	x86_64
x8i.48xlarge	x86_64
x8i.64xlarge	x86_64
x8i.96xlarge	x86_64
x8i.metal-48xl	x86_64
x8i.metal-96xl	x86_64

Accelerated computing

インスタンスタイプ	アーキテクチャ
g6.48xlarge	x86_64
g6e.12xlarge	x86_64
g6e.24xlarge	x86_64
g6e.48xlarge	x86_64
g7.24xlarge	x86_64

インスタンスタイプ	アーキテクチャ
g7.48xlarge	x86_64
g7e.12xlarge	x86_64
g7e.24xlarge	x86_64
g7e.48xlarge	x86_64
p5.4xlarge	x86_64
p5.48xlarge	x86_64
p5e.48xlarge	x86_64
p5en.48xlarge	x86_64
p6-b200.48xlarge	x86_64
p6-b300.48xlarge	x86_64
p6e-gb200.36xlarge	arm64

Storage optimized

インスタンスタイプ	アーキテクチャ
i4g.4xlarge	arm64
i4g.8xlarge	arm64
i4g.16xlarge	arm64
i4i.8xlarge	x86_64
i4i.12xlarge	x86_64
i4i.16xlarge	x86_64

インスタンスタイプ	アーキテクチャ
i4i.24xlarge	x86_64
i4i.32xlarge	x86_64
i4i.metal	x86_64
i7i.12xlarge	x86_64
i7i.16xlarge	x86_64
i7i.24xlarge	x86_64
i7i.48xlarge	x86_64
i7i.metal-24xl	x86_64
i7i.metal-48xl	x86_64
i7ie.6xlarge	x86_64
i7ie.12xlarge	x86_64
i7ie.18xlarge	x86_64
i7ie.24xlarge	x86_64
i7ie.48xlarge	x86_64
i7ie.metal-24xl	x86_64
i7ie.metal-48xl	x86_64
i8g.8xlarge	arm64
i8g.12xlarge	arm64
i8g.16xlarge	arm64
i8g.24xlarge	arm64

インスタンスタイプ	アーキテクチャ
i8g.48xlarge	arm64
i8g.metal-24xl	arm64
i8g.metal-48xl	arm64
i8ge.6xlarge	arm64
i8ge.12xlarge	arm64
i8ge.18xlarge	arm64
i8ge.24xlarge	arm64
i8ge.48xlarge	arm64
i8ge.metal-24xl	arm64
i8ge.metal-48xl	arm64
im4gn.4xlarge	arm64
im4gn.8xlarge	arm64
im4gn.16xlarge	arm64

Linux インスタンスで ENA Express 設定のパフォーマンスをチューニングする

ENA Express を効果的に運用するためには、お使いの Linux インスタンスがいくつかのネットワーク設定要件を満たしている必要があります。

各設定を手動で設定する代わりに、Amazon GitHub リポジトリから ENA Express の設定チェックスクリプトをダウンロードして実行することができます。このスクリプトは、お使いのインスタンスが ENA Express の必須設定および推奨設定を満たしているかどうかを確認し、検出された問題を修正するための具体的なコマンドを出力します。

https://github.com/amzn/amzn-ec2-ena-utilities/blob/main/ena-express/check-ena-express-settings.sh

このスクリプトは、以下の設定と構成を確認します:

- MTU サイズ – ENA Express では、追加の AWS SRD ヘッダーを収容するために、デフォルトよりも小さい MTU が必要です。この問題を軽減するため、新しく確立された TCP 接続では MSS が自動的に制限されますが、UDP トラフィックでは依然としてより低い MTU が必要です。
- TCP 出力キューのサイズ制限 – ソケットごとの処理中のバイト数の上限が、高いスループットを維持するのに十分であるかを確認します。ネットワークの遅延が大きい環境では、より高い上限値が必要です。
- バイトキュー制限 — ネットワークインターフェースでバイトキュー制限 (BQL) が無効になっていることを確認します。BQL は、デバイスレベルでの送信のためにキューに入れられるデータ量を制限することができ、これにより ENA Express のパフォーマンスが低下します。

> **Note**
>
> Amazon Linux ディストリビューションの ENA ドライバーは、デフォルトでバイトキュー制限を無効にします。

- TCP オートコーキング — TCP オートコーキングが無効になっているかどうかを確認します。オートコーキングを無効にすると、リクエスト・レスポンス型のワークロードなど、特定の ENA Express TCP トラフィックパターンにおいてレイテンシを低減できる場合があります。これにより、パケット処理のオーバーヘッドがわずかに増加する可能性があります。
- TX キューサイズと Large LLQ – ネットワークインターフェースの送信キューサイズが、最適なパフォーマンスを発揮するのに十分な大きさであることを確認します。また、このスクリプトでは、ENA モジュールのパラメータによって Large Low Latency Queue (Large LLQ) 機能が明示的に無効化されていないかどうかも確認します。これは、この機能が利用可能な TX キューの深さを減少させる可能性があるためです。Large LLQ および TX キューサイズへの影響に関する詳細については、GitHub の「Large Low Latency Queue (Large LLQ)」を参照してください。
- RX キューサイズ – ネットワークインターフェースの受信リングバッファが、着信トラフィックを効率的に処理し、負荷がかかった際にパケットがドロップされないように十分な容量を確保しているかを確認します。
- TCP およびネットワークソケットバッファサイズ – TCP の受信および送信バッファの最大サイズ、ならびにコアネットワークソケットのバッファのデフォルト値および最大値が、高いスループットを維持するのに十分な大きさであることを検証します。これらの設定は、ネットワークの遅延が大きくなる環境において、接続を有効に活用するために大きなバッファが必要となる場合に重要です。
- TCP 輻輳制御 – ネットワーク遅延が大きい環境において、TCP 輻輳制御の設定が ENA Express での使用に最適化されていることを確認します。

このスクリプトは、ENA ドライバのバージョン、ENA SRD の統計情報、割り込み調整設定、キュー構成、ソケットバッファサイズなど、その他の診断情報も報告します。この情報は、ENA Express のパフォーマンスに関する問題のトラブルシューティングに役立ちます。

インスタンスのネットワークドライバが最適なパフォーマンスを発揮するように設定されていることを確認するため、GitHub にある「ENA Linux ドライバーのベストプラクティスとパフォーマンス最適化ガイド」も確認してください。

EC2 インスタンスの ENA Express 設定を確認する

ENA Express 設定は、インスタンスまたはネットワークインターフェイスで確認できます。ENA Express 設定を更新するには、the section called “インスタンスの設定を行う”を参照してください。

Console

ネットワークインターフェイスの ENA Express 設定を表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左側のナビゲーションペインで、[ネットワークインターフェース] を選択してください。
3. ネットワークインターフェイスを選択すると、そのインスタンスの詳細が表示されます。[ネットワークインターフェース ID] リンクを選択して詳細ページを開くか、リストの左側のチェックボックスを選択してページ下部の詳細ペインで詳細を表示できます。
4. [詳細] タブまたは詳細ページの [ネットワークインターフェイスのアタッチメント] セクションで、[ENA Express] と [ENA Express UDP] の設定を確認します。

インスタンスの ENA Express 設定を表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左のナビゲーションペインで、[インスタンス] をクリックしてください。
3. インスタンスを選択すると、そのインスタンスの詳細が表示されます。[インスタンス ID] リンクを選択して詳細ページを開くことも、リストの左側のチェックボックスを選択してページ下部の詳細ペインに詳細を表示することもできます。
4. [ネットワーキング] タブの [ネットワークインターフェース] セクションを右にスクロールして [ENA Express] と [ENA Express UDP] の設定を確認します。

AWS CLI

インスタンスの ENA Express 設定を取得するには

describe-instances コマンドを使用します。このコマンド例は、--instance-ids パラメータで指定された実行中の各インスタンスに接続されているネットワークインターフェイスの ENA Express 設定のリストを返します。

```
aws ec2 describe-instances \
    --instance-ids i-1234567890abcdef0 i-0598c7d356eba48d7 \
    --query 'Reservations[*].Instances[*].[InstanceId,
 NetworkInterfaces[*].Attachment.EnaSrdSpecification]'
```

以下は出力の例です。

```
[
    [
        [
            "i-1234567890abcdef0",
            [
                {
                    "EnaSrdEnabled": true,
                    "EnaSrdUdpSpecification": {
                        "EnaSrdUdpEnabled": false
                    }
                }
            ]
        ]
    ],
    [
        [
            "i-0598c7d356eba48d7",
            [
            {
                    "EnaSrdEnabled": true,
                    "EnaSrdUdpSpecification": {
                        "EnaSrdUdpEnabled": false
                    }
                }
            ]
        ]
    ]
]
```

ネットワークインターフェイスの ENA Express 設定を取得するには

describe-network-interfaces コマンドを使用します。

```
aws ec2 describe-network-interfaces \
    --network-interface-ids eni-1234567890abcdef0 \
    --query NetworkInterfaces[].[NetworkInterfaceId,Attachment.EnaSrdSpecification]
```

以下は出力の例です。

```
[
    [
        "eni-1234567890abcdef0",
        {
            "EnaSrdEnabled": true,
            "EnaSrdUdpSpecification": {
                "EnaSrdUdpEnabled": false
            }
        }
    ]
]
```

PowerShell

ネットワークインターフェイスの ENA Express 設定を取得するには

Get-EC2NetworkInterface コマンドレットを使用します。

```
Get-EC2NetworkInterface `
    -NetworkInterfaceId eni-1234567890abcdef0 | `
Select-Object `
    Association,
    NetworkInterfaceId,
    OwnerId,
    @{Name = 'AttachTime'; Expression = { $_.Attachment.AttachTime } },
    @{Name = 'AttachmentId'; Expression = { $_.Attachment.AttachmentId } },
    @{Name = 'DeleteOnTermination'; Expression =
 { $_.Attachment.DeleteOnTermination } },
    @{Name = 'NetworkCardIndex'; Expression = { $_.Attachment.NetworkCardIndex } },
    @{Name = 'InstanceId'; Expression = { $_.Attachment.InstanceId } },
    @{Name = 'InstanceOwnerId'; Expression = { $_.Attachment.InstanceOwnerId } },
    @{Name = 'Status'; Expression = { $_.Attachment.Status } },
    @{Name = 'EnaSrdEnabled'; Expression =
 { $_.Attachment.EnaSrdSpecification.EnaSrdEnabled } },
```

```
    @{Name = 'EnaSrdUdpEnabled'; Expression =
{ $_.Attachment.EnaSrdSpecification.EnaSrdUdpSpecification.EnaSrdUdpEnabled } }
```

以下は出力の例です。

```
Association         :
NetworkInterfaceId  : eni-0d1234e5f6a78901b
OwnerId             : 111122223333
AttachTime          : 6/11/2022 1:13:11 AM
AttachmentId        : eni-attach-0d1234e5f6a78901b
DeleteOnTermination : True
NetworkCardIndex    : 0
InstanceId          : i-1234567890abcdef0
InstanceOwnerId     : 111122223333
Status              : attached
EnaSrdEnabled       : True
EnaSrdUdpEnabled    : False
```

ご使用の EC2 インスタンスに適した ENA Express を設定する

ENA Express は、対応する EC2 インスタンスタイプに対して、追加のソフトウェアをインストールすることなく設定できます。詳細については、the section called “ENA Express でサポートされるインスタンスタイプ”を参照してください。

Console

ネットワークインターフェイスの ENA Express を管理するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左側のナビゲーションペインで、[ネットワークインターフェース] を選択してください。
3. インスタンスにアタッチされるネットワークインターフェースを指定します。[ネットワークインターフェース ID] リンクを選択して詳細ページを開くことも、リストの左側にあるチェックボックスを選択することもできます。
4. ページ右上の [アクション] メニューから [ENA Express の管理] を選択してください。これにより、選択したネットワークインターフェイス ID と現在の設定が表示された [ENA Express の管理] ダイアログが開きます。

   選択したネットワークインターフェースがインスタンスに接続されていない場合、このアクションはメニューに表示されません。

5. [ENA Express] を使用するには、[有効化] チェックボックスをオンにします。
6. ENA Express が有効になっている場合、UDP 設定を構成できます。[ENA Express UDP] を使用するには、[有効化] チェックボックスをオンにします。
7. 設定を保存するには [保存] を選択してください。

インスタンスの ENA Express を管理するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左のナビゲーションペインで、[インスタンス] をクリックしてください。
3. 管理するインスタンスを選択してください。[インスタンス ID] を選択して詳細ページを開くか、リストの左側にあるチェックボックスを選択してください。
4. インスタンスに設定する[ネットワークインターフェイス] を選択してください。
5. ページ右上の [アクション] メニューから [ENA Express の管理] を選択してください。
6. インスタンスにアタッチされているネットワークインターフェイスに ENA Express を設定するには、[ネットワークインターフェイス] リストから選択してください。
7. 選択したネットワークインターフェイスアタッチメントに [ENA Express] を使用するには、[有効化] チェックボックスをオンにします。
8. ENA Express が有効になっている場合、UDP 設定を構成できます。[ENA Express UDP] を使用するには、[有効化] チェックボックスをオンにします。
9. 設定を保存するには [保存] を選択してください。

ネットワークインターフェイスをアタッチする際に ENA Express を設定するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左側のナビゲーションペインで、[ネットワークインターフェース] を選択してください。
3. インスタンスにアタッチされていないネットワークインターフェイスを選択します ([ステータス] が [利用可] のもの)。[ネットワークインターフェース ID] リンクを選択して詳細ページを開くことも、リストの左側にあるチェックボックスを選択することもできます。
4. アタッチする [インスタンス] を選択してください。
5. ネットワークインターフェイスをインスタンスにアタッチした後に [ENA Express] を使用するには、[有効化] チェックボックスをオンにします。
6. ENA Express が有効になっている場合、UDP 設定を構成できます。[ENA Express UDP] を使用するには、[有効化] チェックボックスをオンにします。

7. ネットワークインターフェイスをインスタンスにアタッチし、ENA Express の設定を保存するには、[アタッチ] を選択してください。

AWS CLI

ネットワークインターフェイスをアタッチする際に ENA Express を設定するには

次の例に示すように attach-network-interface コマンドを使用します。

例 1: TCP トラフィックには ENA Express を使用するが、UDP トラフィックには使用しない

この例では、EnaSrdEnabled を true に設定し、EnaSrdUdpEnabled をデフォルトで false になるよう設定します。

```
aws ec2 attach-network-interface \
    --network-interface-id eni-1234567890abcdef0 \
    --instance-id i-1234567890abcdef0 \
    --device-index 1 \
    --ena-srd-specification 'EnaSrdEnabled=true'
```

例 2: TCP トラフィックと UDP トラフィックの両方に ENA Express を使用する

この例では、EnaSrdEnabled と EnaSrdUdpEnabled の両方を true に設定します。

```
aws ec2 attach-network-interface \
    --network-interface-id eni-1234567890abcdef0 \
    --instance-id i-1234567890abcdef0 \
    --device-index 1 \
    --ena-srd-specification
 'EnaSrdEnabled=true,EnaSrdUdpSpecification={EnaSrdUdpEnabled=true}'
```

ネットワークインターフェイスアタッチメントの ENA Express 設定を更新するには

次の例に示すように modify-network-interface-attribute コマンドを使用します。

例 1: TCP トラフィックには ENA Express を使用するが、UDP トラフィックには使用しない

この例では、EnaSrdEnabled を true に設定し、以前に設定したことがない場合は EnaSrdUdpEnabled をデフォルトで false になるよう設定します。

```
aws ec2 modify-network-interface-attribute \
    --network-interface-id eni-1234567890abcdef0 \
```

```
    --ena-srd-specification 'EnaSrdEnabled=true'
```

例 2: TCP トラフィックと UDP トラフィックの両方に ENA Express を使用する

この例では、EnaSrdEnabled と EnaSrdUdpEnabled の両方を true に設定します。

```
aws ec2 modify-network-interface-attribute \
    --network-interface-id eni-1234567890abcdef0 \
    --ena-srd-specification
 'EnaSrdEnabled=true,EnaSrdUdpSpecification={EnaSrdUdpEnabled=true}'
```

例 3: UDP トラフィックでの ENA Express の使用を停止する

この例では、EnaSrdUdpEnabled を false に設定します。

```
aws ec2 modify-network-interface-attribute \
    --network-interface-id eni-1234567890abcdef0 \
    --ena-srd-specification 'EnaSrdUdpSpecification={EnaSrdUdpEnabled=false}'
```

PowerShell

ネットワークインターフェイスをアタッチする際に ENA Express を設定するには

次の例に示すように、Add-EC2NetworkInterface コマンドレットを使用します。

例 1: TCP トラフィックには ENA Express を使用するが、UDP トラフィックには使用しない

この例では、EnaSrdEnabled を true に設定し、EnaSrdUdpEnabled をデフォルトで false になるよう設定します。

```
Add-EC2NetworkInterface `
    -NetworkInterfaceId eni-1234567890abcdef0 `
    -InstanceId i-1234567890abcdef0 `
    -DeviceIndex 1 `
    -EnaSrdSpecification_EnaSrdEnabled $true
```

例 2: TCP トラフィックと UDP トラフィックの両方に ENA Express を使用する

この例では、EnaSrdEnabled と EnaSrdUdpEnabled の両方を true に設定します。

```
Add-EC2NetworkInterface `
    -NetworkInterfaceId eni-1234567890abcdef0 `
```

```
    -InstanceId i-1234567890abcdef0 `
    -DeviceIndex 1 `
    -EnaSrdSpecification_EnaSrdEnabled $true `
    -EnaSrdUdpSpecification_EnaSrdUdpEnabled $true
```

ネットワークインターフェイスアタッチメントの ENA Express 設定を設定するには

次の例に示すように、Edit-EC2NetworkInterfaceAttribute コマンドレットを使用します。

例 1: TCP トラフィックには ENA Express を使用するが、UDP トラフィックには使用しない

この例では、EnaSrdEnabled を true に設定し、以前に設定したことがない場合は EnaSrdUdpEnabled をデフォルトで false になるよう設定します。

```
Edit-EC2NetworkInterfaceAttribute `
    -NetworkInterfaceId eni-1234567890abcdef0 `
    -EnaSrdSpecification_EnaSrdEnabled $true ;
Get-EC2NetworkInterface -NetworkInterfaceId eni-0123f4567890a1b23 | `
Select-Object `
    NetworkInterfaceId,
    @{Name = 'EnaSrdEnabled'; Expression =
 { $_.Attachment.EnaSrdSpecification.EnaSrdEnabled }},
    @{Name = 'EnaSrdUdpEnabled'; Expression =
 { $_.Attachment.EnaSrdSpecification.EnaSrdUdpSpecification.EnaSrdUdpEnabled }} | `
Format-List
```

例 2: TCP トラフィックと UDP トラフィックの両方に ENA Express を使用する

この例では、EnaSrdEnabled と EnaSrdUdpEnabled の両方を true に設定します。

```
Edit-EC2NetworkInterfaceAttribute `
    -NetworkInterfaceId eni-1234567890abcdef0 `
    -EnaSrdSpecification_EnaSrdEnabled $true `
    -EnaSrdSpecification_EnaSrdUdpSpecification_EnaSrdUdpEnabled $true ;
Get-EC2NetworkInterface -NetworkInterfaceId eni-1234567890abcdef0 | `
Select-Object `
    NetworkInterfaceId,
    @{Name = 'EnaSrdEnabled'; Expression =
 { $_.Attachment.EnaSrdSpecification.EnaSrdEnabled }},
    @{Name = 'EnaSrdUdpEnabled'; Expression =
 { $_.Attachment.EnaSrdSpecification.EnaSrdUdpSpecification.EnaSrdUdpEnabled }} | `
Format-List
```

例 3: UDP トラフィックでの ENA Express の使用を停止する

この例では、EnaSrdUdpEnabled を false に設定します。

```
Edit-EC2NetworkInterfaceAttribute `
    -NetworkInterfaceId eni-0123f4567890a1b23 `
    -EnaSrdSpecification_EnaSrdUdpSpecification_EnaSrdUdpEnabled $false ;
Get-EC2NetworkInterface -NetworkInterfaceId eni-0123f4567890a1b23 | `
Select-Object `
    NetworkInterfaceId,
    @{Name = 'EnaSrdEnabled'; Expression =
 { $_.Attachment.EnaSrdSpecification.EnaSrdEnabled }},
    @{Name = 'EnaSrdUdpEnabled'; Expression =
 { $_.Attachment.EnaSrdSpecification.EnaSrdUdpSpecification.EnaSrdUdpEnabled }} | `
Format-List
```

起動時に ENA Express を設定する

インスタンスを起動するときに、以下のいずれかの方法を使用して ENA Express を直接設定できます。設定されているリンクをたどると、AWS マネジメントコンソールからこれらの方法を行う手順を確認できます。

- インスタンス起動ウィザード – インスタンス起動ウィザードでは、起動時に ENA Express を設定できます。詳細については、インスタンス起動ウィザードの ネットワーク設定、「Advanced network configuration」を参照してください。
- 起動テンプレート – 起動テンプレートを使用すると、起動時に ENA Express を設定できます。詳細については、「Amazon EC2 起動テンプレートを作成する」ページを参照し、[ネットワーク設定] セクションを展開して、[高度なネットワーク設定] を確認してください。

Intel 82599 VF インターフェイスを使用する拡張ネットワーキング

Xen ベースのインスタンス の場合、Intel 82599 仮想関数 (VF) インターフェイスは拡張ネットワーキング機能を提供します。インターフェイスは Intel ixgbevf ドライバーを使用します。

次のタブはインスタンスオペレーティングシステムにインストールされているネットワークアダプタードライバーを確認する方法を示しています。

Linux

Linux ネットワークインターフェイスドライバー

次のコマンドを使用して、モジュールが特定のインターフェイスで使用されていることを確認し、確認するインターフェイス名に置き換えます。単一のインターフェイス (デフォルト) を使用している場合はeth0 です。オペレーティングシステムで予測可能なネットワーク名がサポートされている場合はens5 のような名前にすることができます。

次の例で、リストされているドライバーは vif であるため、ixgbevf モジュールはロードされていません。

```
[ec2-user ~]$ ethtool -i eth0
driver: vif
version:
firmware-version:
bus-info: vif-0
supports-statistics: yes
supports-test: no
supports-eeprom-access: no
supports-register-dump: no
supports-priv-flags: no
```

この例ではixgbevf モジュールがロードされます。このインスタンスでは拡張ネットワーキングが適切に設定されています。

```
[ec2-user ~]$ ethtool -i eth0
driver: ixgbevf
version: 4.0.3
firmware-version: N/A
bus-info: 0000:00:03.0
supports-statistics: yes
supports-test: yes
supports-eeprom-access: no
supports-register-dump: yes
supports-priv-flags: no
```

Windows

Windows ネットワークアダプター

ドライバーがインストールされていることを確認するにはインスタンスに接続し、デバイスマネージャーを開きます。[ネットワークアダプター] の下に Intel(R) 82599 Virtual Function が表示されるはずです。

内容

- 拡張ネットワーキング用のインスタンスの準備
- 拡張ネットワーキングが有効化されているかどうかのテスト
- インスタンスでの拡張ネットワーキングの有効化
- 接続に関する問題のトラブルシューティング

拡張ネットワーキング用のインスタンスの準備

Intel 82599 VF インターフェイスを使用した拡張ネットワーキングを準備するには次のようにインスタンスをセットアップします。

- インスタンスタイプが C3、C4、D2、I2、M4 (`m4.16xlarge` を除く)、R3 のいずれかであることを確認します。
- インスタンスがインターネットに接続されていることを確認します。
- 保持する必要がある重要なデータがインスタンスにある場合、インスタンスから AMI を作成してそのデータをバックアップする必要があります。`sriovNetSupport` 属性を有効にするとともに、カーネルおよびカーネルモジュールを更新すると、互換性のないインスタンスがレンダリングされたり、オペレーティングシステムに接続できなくなったりする可能性があります。最近のバックアップがある場合はこれが発生してもデータは保持されます。
- Linux インスタンス – Linux カーネルバージョン 2.6.32 以降を使用して、HVM AMI からインスタンスを起動します。最新の Amazon Linux HVM AMI では拡張ネットワーキングに必要なモジュールがインストールされており、必要な属性も設定されています。したがって、最新の Amazon Linux HVM AMI を使用して、拡張ネットワーキングがサポートされている Amazon EBS–バックインスタンスを起動した場合はインスタンスで拡張ネットワーキングが既に有効化されています。

  > **Warning**
  >
  > 拡張ネットワーキングはHVM インスタンスでのみサポートされています。PV インスタンスで拡張ネットワーキングを有効にすると、このインスタンスに到達できなくなります。また、適切なモジュールまたはモジュールバージョンを使用せずにこの属性を設定すると、インスタンスにアクセスできなくなる場合があります。

- Windows インスタンス – 64 ビット HVM AMI からインスタンスを起動します。Windows Server 2008 では拡張ネットワーキングを有効にできません。Windows Server 2012 R2 および Windows Server 2016 以降の AMI では拡張ネットワーキングが既に有効になっています。Windows Server

2012 R2 にはインテルドライバー 1.0.15.3 が含まれており、Pnputil.exe ユーティリティを使用してそのドライバーを最新のバージョンにアップグレードすることをお勧めします。

- 選択した任意のコンピュータ、できればローカルのデスクトップまたはノートパソコンで、AWS マネジメントコンソール から AWS CloudShell を使用するか、AWS CLI もしくは AWS Tools for Windows PowerShell をインストールし設定します。詳細についてはAmazon EC2 へのアクセスもしくは AWS CloudShell ユーザーガイドを参照してください。拡張ネットワーキングは、Amazon EC2 コンソールから管理することはできません。

拡張ネットワーキングが有効化されているかどうかのテスト

`sriovNetSupport` 属性がインスタンスまたはイメージに設定されていることを確認します。

AWS CLI

インスタンス属性をチェックするには (sriovNetSupport)

次の describe-instance-attribute コマンドを使用します。属性が設定されている場合、値は `simple` になります。

```
aws ec2 describe-instance-attribute \
    --instance-id i-1234567890abcdef0 \
    --attribute sriovNetSupport
```

イメージ属性をチェックするには (sriovNetSupport)

次の describe-images コマンドを使用します。属性が設定されている場合、値は `simple` になります。

```
aws ec2 describe-images \
    --image-id ami-0abcdef1234567890 \
    --query "Images[].SriovNetSupport"
```

PowerShell

インスタンス属性をチェックするには (sriovNetSupport)

Get-EC2InstanceAttribute コマンドレットを使用します。属性が設定されている場合、値は `simple` になります。

```
Get-EC2InstanceAttribute `
```

```
    -InstanceId i-1234567890abcdef0 `
    -Attribute sriovNetSupport
```

イメージ属性をチェックするには (sriovNetSupport)

次の describe-images コマンドを使用します。属性が設定されている場合、値は simple になります。

```
(Get-EC2Image -ImageId ami-0abcdef1234567890).SriovNetSupport
```

インスタンスでの拡張ネットワーキングの有効化

使用する手順はインスタンスのオペレーティングシステムによって異なります。

> **Warning**
> 拡張ネットワーキング属性はいったん有効にすると無効にする方法はありません。

Amazon Linux

最新の Amazon Linux HVM AMI では拡張ネットワーキングに必要な ixgbevf モジュールがインストールされており、必要な sriovNetSupport 属性も設定されています。したがって、最新の Amazon Linux HVM AMI を使用してインスタンスタイプを起動した場合は拡張ネットワーキングが既にインスタンスに対して有効になっています。詳細については拡張ネットワーキングが有効化されているかどうかのテストを参照してください。

以前の Amazon Linux AMI を使用してインスタンスを起動し、まだ拡張ネットワーキングが有効になっていない場合、拡張ネットワーキングを有効にするには次の手順を実行します。

拡張ネットワーキングを有効にするには

1. インスタンスに接続します。
2. インスタンスから、次のコマンドを実行して、ixgbevf を含む最新のカーネルとカーネルモジュールでインスタンスを更新します。

```
[ec2-user ~]$ sudo yum update
```

3. ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを再起動します。reboot-instances (AWS CLI)、Restart-EC2Instance (AWS Tools for Windows PowerShell)。
4. インスタンスに再接続し、ixgbevfの modinfo ixgbevf コマンドを使用して、拡張ネットワーキングが有効化されているかどうかのテスト モジュールがインストールされ、最小推奨バージョンであることを確認します。
5. [EBS-backed インスタンス] ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを停止します。stop-instances (AWS CLI)、Stop-EC2Instance (AWS Tools for Windows PowerShell)。

   [Instance store-バック インスタンス] インスタンスを停止して属性を変更することはできません。代わりに、次の手順に進みます。
6. ローカルコンピュータから、次のいずれかのコマンドを使用して拡張ネットワーキングの属性を有効化します。

   AWS CLI

   modify-instance-attribute コマンドを次のように使用します。

   ```
   aws ec2 modify-instance-attribute \
       --instance-id i-1234567890abcdef0 \
       --sriov-net-support simple
   ```

   PowerShell

   Edit-EC2InstanceAttribute コマンドレットを次のように使用します。

   ```
   Edit-EC2InstanceAttribute `
       -InstanceId i-1234567890abcdef0 `
       -SriovNetSupport "simple"
   ```
7. (オプション) Amazon EBS-backed AMI を作成するの説明に従って、インスタンスから AMI を作成します。AMI はインスタンスから拡張ネットワーキング属性を継承します。このため、この AMI を使用することで、拡張ネットワーキングがデフォルトで有効になっている別のインスタンスを起動できます。
8. ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを開始します。start-instances (AWS CLI)、Start-EC2Instance (AWS Tools for Windows PowerShell)。

9. インスタンスに接続し、拡張ネットワーキングが有効化されているかどうかのテスト の ethtool -i eth*n* コマンドを使用して、ixgbevf モジュールがインストールされ、ネットワークインターフェイスにロードされていることを確認します。

拡張ネットワーキングを有効にするには (Instance store-バック インスタンス)

インスタンスを停止するステップまで、前の手順に従います。Amazon S3-backed AMI を作成するに記述されているように、新しい AMI を作成します。AMI を登録するときに拡張ネットワーキング属性を有効にしてください。

AWS CLI

register-image コマンドを次のように使用します。

```
aws ec2 register-image --sriov-net-support simple ...
```

PowerShell

Register-EC2Image を次のように使用します。

```
Register-EC2Image -SriovNetSupport "simple" ...
```

Ubuntu

開始する前に、インスタンスで拡張ネットワーキングがすでに有効になっているかどうかを確認します。

クイックスタート Ubuntu HVM AMI には拡張ネットワーキングに必要なドライバーが搭載されています。ixgbevf 2.16.4 より前のバージョンを使用している場合はlinux-aws カーネルパッケージをインストールして最新の拡張ネットワーキングドライバーを取得できます。

以下の手順はUbuntu インスタンスで ixgbevf モジュールをコンパイルするための一般的なステップを示しています。

linux-aws カーネルパッケージをインストールするには

1. インスタンスに接続します。
2. パッケージキャッシュおよびパッケージを更新します。

```
ubuntu:~$ sudo apt-get update && sudo apt-get upgrade -y linux-aws
```

> **Important**
>
> 更新プロセス中に grub をインストールするよう求められた場合は/dev/xvda のインストール先として grub を使用し、現在のバージョンの /boot/grub/menu.lst を保持することを選択してください。

他の Linux ディストリビューション

開始する前に、インスタンスで拡張ネットワーキングがすでに有効になっているかどうかを確認します。最新のクイックスタート HVM AMI には拡張ネットワーキングに必要なドライバーが含まれているため、追加ステップを実行する必要はありません。

次の手順ではAmazon Linux または Ubuntu 以外の Linux ディストリビューションで Intel 82599 VF インターフェイスを使用した拡張ネットワーキングを有効にする必要がある場合の一般的なステップを説明します。コマンドの詳細な構文、ファイルの場所、パッケージやツールのサポートなどの詳細については使用する Linux ディストリビューションのドキュメントを参照してください。

Linux で拡張ネットワーキングを有効化するには

1. インスタンスに接続します。
2. Sourceforge (https://sourceforge.net/projects/e1000/files/ixgbevf%20stable/) からインスタンスに ixgbevf モジュールのソースをダウンロードします。

   ixgbevf の 2.16.4 より前のバージョン (バージョン 2.14.2 を含む) は一部の Linux ディストリビューション (特定のバージョンの Ubuntu など) では適切にビルドされません。
3. インスタンスで ixgbevf モジュールをコンパイルし、インストールします。

> **Warning**
>
> 現在のカーネルに ixgbevf モジュールをコンパイルし、新しいカーネルをドライバを再構築しないで更新すると、システムは次回の再起動の際にディストリビューション固有の ixgbevf モジュールに戻る場合があります。これにより、ディストリビューション固有のバージョンが拡張ネットワーキングと互換性がない場合に、システムに接続できなくなります。

4. sudo depmod コマンドを実行して、モジュールの依存関係を更新します。
5. 起動時に新しいモジュールがロードされるように、インスタンスの `initramfs` を更新します。
6. システムがデフォルトで予測可能なネットワークインターフェイス名を使用するかどうかを確認します。systemd または udev のバージョン 197 以上を使用するシステムの場合、イーサネットデバイスの名前を変更でき、単一ネットワークインターフェイスの名前が `eth0` になることは保証されません。この動作はインスタンスに接続する際に問題の原因となる可能性があります。詳細と他の設定オプションについてはfreedesktop.org ウェブサイトでPredictable Network Interface Names/を参照してください。

   a. 次のコマンドを使用して、RPM ベースのシステムで systemd または udev のバージョンを確認できます。

   ```
   [ec2-user ~]$ rpm -qa | grep -e '^systemd-[0-9]\+\|^udev-[0-9]\+'
       systemd-208-11.el7_0.2.x86_64
   ```

   上記の Red Hat Enterprise Linux 7 の例ではsystemd のバージョンは 208 であるため、予測可能なネットワークインターフェイス名は無効になっている必要があります。

   b. `net.ifnames=0` オプションを `GRUB_CMDLINE_LINUX` の `/etc/default/grub` 行に追加することによって、予測可能なネットワークインターフェイス名を無効にします。

   ```
   [ec2-user ~]$ sudo sed -i '/^GRUB\_CMDLINE\_LINUX/s/\"$/\ net\.ifnames\=0\"/' /
   etc/default/grub
   ```

   c. grub の設定ファイルを再ビルドします。

   ```
   [ec2-user ~]$ sudo grub2-mkconfig -o /boot/grub2/grub.cfg
   ```

7. [EBS-backed インスタンス] ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用してインスタンスを停止します。stop-instances (AWS CLI)/Stop-EC2Instance)、 (AWS Tools for Windows PowerShell)。

   [Instance store-バック インスタンス] インスタンスを停止して属性を変更することはできません。代わりに、次の手順に進みます。

8. ローカルコンピュータから、次のいずれかのコマンドを使用して拡張ネットワーキングの属性を有効化します。

AWS CLI

modify-instance-attribute コマンドを次のように使用します。

```
aws ec2 modify-instance-attribute \
    --instance-id i-1234567890abcdef0 -\
    -sriov-net-support simple
```

PowerShell

Edit-EC2InstanceAttribute コマンドレットを次のように使用します。

```
Edit-EC2InstanceAttribute `
    -InstanceId i-1234567890abcdef0 `
    -SriovNetSupport "simple"
```

9. (オプション) Amazon EBS-backed AMI を作成するの説明に従って、インスタンスから AMI を作成します。AMI はインスタンスから拡張ネットワーキング属性を継承します。このため、この AMI を使用することで、拡張ネットワーキングがデフォルトで有効になっている別のインスタンスを起動できます。

   インスタンスオペレーティングシステムに /etc/udev/rules.d/70-persistent-net.rules が含まれている場合にはAMI を作成する前にそれを削除する必要があります。このファイルには元のインスタンスのイーサネットアダプターの MAC アドレスが保存されています。別のインスタンスがこのファイルを使用して起動した場合、オペレーティングシステムがそのデバイスを検出できなくなり、eth0 が失敗して、起動に関する問題が発生することがあります。このファイルは次の起動サイクルで再び生成され、AMI から起動されるインスタンスごとに独自のバージョンが作成されます。

10. ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを開始します。start-instances (AWS CLI)、Start-EC2Instance (AWS Tools for Windows PowerShell)。
11. (オプション) インスタンスに接続し、モジュールがインストールされていることを確認します。

拡張ネットワーキングを有効にするには (Instance store-バック インスタンス)

インスタンスを停止するステップまで、前の手順に従います。Amazon S3-backed AMI を作成するに記述されているように、新しい AMI を作成します。AMI を登録するときに拡張ネットワーキング属性を有効にしてください。

AWS CLI

register-image コマンドを次のように使用します。

```
aws ec2 register-image --sriov-net-support simple ...
```

PowerShell

Register-EC2Image を次のように使用します。

```
Register-EC2Image -SriovNetSupport "simple" ...
```

Server

インスタンスを起動し、すでに拡張ネットワーキングが有効になっていない場合、必要なネットワークアダプタードライバーをダウンロードしてインスタンスにインストールし、拡張ネットワーキングを有効にするように sriovNetSupport インスタンス属性を設定する必要があります。この属性を有効にできるのはサポートされるインスタンスタイプのみです。詳細については「Amazon EC2 インスタンスでの拡張ネットワーキング」を参照してください。

> **Important**
>
> Windows AMI の最新のドライバーアップデートを確認するにはAWS Windows AMI リファレンスの「Windows AMI のバージョン履歴」を参照してください。

拡張ネットワーキングを有効にするには

1. インスタンスに接続してローカル管理者としてログインします。
2. [Windows Server 2016 以降] ドライバーがインストールされたら以下の EC2 Launch PowerShell スクリプトを実行し、インスタンスを設定します。

```
PS C:\> C:\ProgramData\Amazon\EC2-Windows\Launch\Scripts\InitializeInstance.ps1 -
Schedule
```

> **Important**
>
> 管理者パスワードはインスタンス初期化 EC2 Launch スクリプトを有効にするとリセットされます。初期化タスクの設定で指定することで、管理者パスワードのリセットを無効にするように設定ファイルを変更できます。

3. インスタンスから、OS 用の Intel ネットワークアダプタードライバーをダウンロードします。

   - [Windows Server 2022]

     ダウンロードページとWired_driver_*version*_x64.zip のダウンロードを見てください。

   - Windows Server 2019 Server バージョン 1809 以降を含む*

     ダウンロードページとWired_driver_*version*_x64.zip のダウンロードを見てください。

   - Windows Server 2016 Server バージョン 1803 以前を含む*

     ダウンロードページとWired_driver_*version*_x64.zip のダウンロードを見てください。

   - [Windows Server 2012 R2

     ダウンロードページとWired_driver_*version*_x64.zip のダウンロードを見てください。

   - Windows Server 2012

     ダウンロードページとWired_driver_*version*_x64.zip のダウンロードを見てください。

   - Windows Server 2008 R2

     ダウンロードページとPROWinx64Legacy.exe のダウンロードを見てください。

   *Server バージョン 1803 以前および 1809 以降はIntel のドライバーおよびソフトウェアのページでは特に扱われていません。

4. OS 用の Intel ネットワークアダプタードライバーをインストールします。

   - Windows Server 2008 R2

1. ダウンロードフォルダで、PROWinx64Legacy.exe ファイルを見つけて、名前を PROWinx64Legacy.zip に変更します。
2. PROWinx64Legacy.zip ファイルの内容を展開します。
3. コマンドラインを開き、抽出されたフォルダに移動し、pnputil ユーティリティを使用して次のコマンドを実行して、ドライバーストアで INF ファイルを追加およびインストールします。

```
C:\> pnputil -a PROXGB\Winx64\NDIS62\vxn62x64.inf
```

- Windows Server 2022、Windows Server 2019、Windows Server 2016、Windows Server 2012 R2、および Windows Server 2012

1. ダウンロードフォルダで、Wired_driver_*version*_x64.zip ファイルの内容を展開します。
2. コマンドラインを開き、抽出されたフォルダに移動し、pnputil ユーティリティを使用して次のコマンドのいずれかを実行して、ドライバーストアで INF ファイルを追加およびインストールします。

- Windows Server 2022

```
pnputil -i -a PROXGB\Winx64\NDIS68\vxn68x64.inf
```

- Windows Server 2019

```
pnputil -i -a PROXGB\Winx64\NDIS68\vxn68x64.inf
```

- Windows Server 2016

```
pnputil -i -a PROXGB\Winx64\NDIS65\vxn65x64.inf
```

- Windows Server 2012 R2

```
pnputil -i -a PROXGB\Winx64\NDIS64\vxn64x64.inf
```

- Windows Server 2012

```
pnputil -i -a PROXGB\Winx64\NDIS63\vxn63x64.inf
```

5. ローカルコンピュータから、次のいずれかのコマンドを使用して拡張ネットワーキングの属性を有効化します。

AWS CLI

modify-instance-attribute コマンドを次のように使用します。

```
aws ec2 modify-instance-attribute \
    --instance-id i-1234567890abcdef0 \
    --sriov-net-support simple
```

PowerShell

Edit-EC2InstanceAttribute コマンドレットを次のように使用します。

```
Edit-EC2InstanceAttribute `
    -InstanceId i-1234567890abcdef0 `
    -SriovNetSupport "simple"
```

6. (オプション) Amazon EBS-backed AMI を作成するの説明に従って、インスタンスから AMI を作成します。AMI はインスタンスから拡張ネットワーキング属性を継承します。このため、この AMI を使用することで、拡張ネットワーキングがデフォルトで有効になっている別のインスタンスを起動できます。
7. ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを開始します。start-instances (AWS CLI)、Start-EC2Instance (AWS Tools for Windows PowerShell)。

接続に関する問題のトラブルシューティング

拡張ネットワーキングを有効化しているときに接続が失われると、ixgbevf モジュールとカーネルの互換性が保たれない可能性があります。この場合、インスタンスの Linux ディストリビューションに含まれる ixgbevf モジュールのバージョンをインストールしてみます。

PV インスタンスまたは AMI で拡張ネットワーキングを有効にすると、お使いのインスタンスに到達できなくなります。

詳細については「EC2 インスタンスで拡張ネットワーキングを有効化および設定する方法」を参照してください。

EC2 インスタンスでの ENA 設定のネットワークパフォーマンスのモニタリング

Elastic Network Adapter (ENA) ドライバーは、有効になっているインスタンスからネットワークパフォーマンスメトリクスを公開します。このようなメトリクスを使用して、インスタンスのパフォーマンスの問題のトラブルシューティング、ワークロードに適したインスタンスサイズの選択、スケーリングアクティビティの事前計画、およびアプリケーションのベンチマークにより、メトリクスがインスタンスで利用できるパフォーマンスを最大化するかどうかを判断できます。

Amazon EC2 は、インスタンスレベルでネットワーク最大値を定義し、インスタンスサイズ全体で一貫したネットワークパフォーマンスを含め、質の高いネットワークエクスペリエンスを実現します。AWS は、各インスタンスに次の最大値を提供します。

- 帯域幅機能 - 各 EC2 インスタンスには、インスタンスタイプとサイズに基づいて、集計したインバウンドトラフィックとアウトバウンドトラフィックの最大帯域幅があります。インスタンスの一部は、ネットワーク I/O クレジットメカニズムを使用して、平均帯域幅使用率に基づいて、ネットワーク帯域幅を割り当てます。また、Amazon EC2 には、Direct Connect およびインターネットへのトラフィックに最大帯域幅があります。詳細については、「Amazon EC2 インスタンスのネットワーク帯域幅」を参照してください。
- Packet-per-second (PPS) パフォーマンス - 各 EC2 インスタンスには、インスタンスタイプとサイズに基づいて、最大 PPS パフォーマンスがあります。
- 追跡された接続 - セキュリティグループは、確立された各接続を追跡し、リターンパケットが期待どおりに配信されることを確認します。インスタンスごとに追跡できる接続の最大数があります。詳細については、Amazon EC2 セキュリティグループの接続の追跡を参照してください。
- リンクローカルサービスアクセス - Amazon EC2 は、Amazon DNS サービス、インスタンスメタデータサービス、Amazon Time Sync Service などのローカルプロキシサービスへのトラフィックに対して、ネットワークインターフェイスごとに最大 PPS を提供します。

インスタンスのネットワークトラフィックが最大値を超えると、AWS はネットワークパケットをキューイングしてから破棄することによって、最大値を超えるトラフィックを調整します。ネットワークパフォーマンスメトリクスを使用して、トラフィックが最大値を超えるタイミングをモニタリングできます。これらのメトリクスは、ネットワークトラフィックへの影響、およびネットワークパフォーマンスの問題の可能性をリアルタイムで通知します。

内容

- 要件

- ENA ドライバーのメトリクス
- インスタンスのネットワークパフォーマンスメトリクスを表示します。
- ENA Express のメトリクス
- ENA 用の DPDK ドライバーを備えたネットワークパフォーマンスメトリクス
- FreeBSD を実行しているインスタンスのメトリクス

要件

Linux インスタンス

- ENA ドライババージョン 2.2.10 以降をインストールします。インストールしたバージョンを検証するには、ethtool コマンドを使用します。次の例では、バージョンは最小要件を満たしています。

```
[ec2-user ~]$ ethtool -i eth0 | grep version
version: 2.2.10
```

  ENA ドライバーをアップグレードするには、拡張ネットワーキングを参照してください。
- これらのメトリクスを Amazon CloudWatch にインポートするには、CloudWatch エージェントをインストールします。詳細については、Amazon CloudWatch ユーザーガイドのネットワークパフォーマンスメトリクスの収集を参照してください。
- `conntrack_allowance_available` メトリクスをサポートするには、ENA ドライバーのバージョン 2.8.1 以降をインストールします。
- エグレスフラグメント PPS の制限である 1024 を上書きするには、ENA ドライバーのバージョン 2.13.3 以降をインストールします。

Windows インスタンス

- ENA ドライバーバージョン 2.2.2 以降をインストールします。インストールされているバージョンを確認するには、次のようにデバイスマネージャーを使用します。
  1. `devmgmt.msc` を実行して、デバイスマネージャを開きます。
  2. [Network Adapters] を展開します。
  3. [Amazon Elastic Network Adapter]、[Properties] を選択します。
  4. [Driver] タブで、[Driver Version] を探します。

ENA ドライバーをアップグレードするには、拡張ネットワーキングを参照してください。

- これらのメトリクスを Amazon CloudWatch にインポートするには、CloudWatch エージェントをインストールします。詳細については、Amazon CloudWatch ユーザーガイドの高度なネットワークメトリクスの収集を参照してください。

ENA ドライバーのメトリクス

ENA ドライバーは、次のメトリクスをリアルタイムでインスタンスに配信します。前回のドライバーのリセット以降に、各ネットワークインターフェイスでキューまたはドロップされたパケットの累積数を示します。

メトリクス	説明	以下でサポートされます
bw_in_allowance_exceeded	インバウンド集計帯域幅がインスタンスの最大値を超えたためにキューまたはドロップされたパケットの数。	すべてのインスタンスタイプ
bw_out_allowance_exceeded	アウトバウンド集計帯域幅がインスタンスの最大値を超えたためにキューまたはドロップされたパケットの数。	すべてのインスタンスタイプ
conntrack_allowance_exceeded	接続トラッキングがインスタンスの最大数を超え、新しい接続を確立できなかったためにドロップされたパケットの数。これにより、インスタンスとの間で送受信されるトラフィックのパケット損失が発生する可能性があります。	すべてのインスタンスタイプ
conntrack_allowance_available	そのインスタンスタイプの Connections Tracked 許容量に達する前にインスタンスが確立できる接続トラッキング数。	Nitro ベースのインスタンスのみ

メトリクス	説明	以下でサポートされます
linklocal_allowance_exceeded	ローカルプロキシサービスへのトラフィックの PPS がネットワークインターフェイスの最大値を超えたためにドロップされたパケットの数。これは、Amazon DNS サービス、インスタンスメタデータサービス、および Amazon Time Sync Service へのトラフィックに影響しますが、カスタム DNS リゾルバーへのトラフィックには影響しません。	すべてのインスタンスタイプ
pps_allowance_exceeded	双方向 PPS がインスタンスの最大値を超えたためにキューまたはドロップされたパケットの数。*	すべてのインスタンスタイプ

* ENA Linux ドライバーのバージョン 2.13.3 以降では、フラグメントプロキシモードの設定によっては、この制限にネットワークインターフェイスの 1024 PPS を超えるエグレスフラグメントドロップが含まれる場合があります。Linux ドライバーでフラグメントプロキシモードが有効になっている場合、エグレスフラグメントドロップは、通常であれば適用される 1024 PPS の制限を回避し、標準の PPS 許容値内でカウントされます。フラグメントプロキシモードはデフォルトで無効になっています。

インスタンスのネットワークパフォーマンスメトリクスを表示します。

使用する手順は、インスタンスのオペレーティングシステムによって異なります。

Linux インスタンス

メトリクスをお気に入りのツールに公開して、メトリクスデータを視覚化できます。例えば、CloudWatch エージェントを使用してメトリックスを Amazon CloudWatch に公開できます。エージェントにより、個々のメトリクスを選択し、公開を制御できます。

ethtool を使用して、次のように eth0 などの各ネットワークインターフェイスのメトリクスを取得することもできます。

```
[ec2-user ~]$ ethtool -S eth0
     bw_in_allowance_exceeded: 0
     bw_out_allowance_exceeded: 0
     pps_allowance_exceeded: 0
     conntrack_allowance_exceeded: 0
     linklocal_allowance_exceeded: 0
     conntrack_allowance_available: 136812
```

Windows インスタンス

Windows パフォーマンスカウンターの任意のコンシューマーを使用して、メトリクスを表示できます。EnaPerfCounters マニフェストに従って、データを解析できます。これは、パフォーマンスカウンタープロバイダーとそのカウンターセットを定義する XML ファイルです。

マニフェストをインストールするには

ENA ドライバー 2.2.2 以降を含む AMI を使用してインスタンスを起動した場合、または ENA ドライバー 2.2.2 のドライバーパッケージにインストールスクリプトを使用した場合、マニフェストは既にインストールされています。マニフェストを手動でインストールするには、次の手順を実行します。

1. 次のコマンドを使用して、既存のマニフェストを削除します。

   ```
   unlodctr /m:EnaPerfCounters.man
   ```

2. マニフェストファイル EnaPerfCounters.man をドライバインストールパッケージから %SystemRoot%\System32\drivers にコピーします。
3. 次のコマンドを使用して、新しいマニフェストをインストールします。

   ```
   lodctr /m:EnaPerfCounters.man
   ```

パフォーマンスモニタを使用してメトリクスを表示するには

1. パフォーマンスモニタを開きます。
2. Ctrl+N キーを押して、新しいカウンターを追加します。
3. リストから [ENA Packets Shaping] を選択します。

4. モニタリングするインスタンスを選択し、[Add] を選択します。
5. [OK] を選択してください。

ENA Express のメトリクス

ENA Express は、AWS Scalable Reliable Datagram (SRD) テクノロジーを搭載しています。SRD は、動的ルーティングを使用してスループットを向上させ、テールレイテンシーを最小限に抑える高性能なネットワークトランスポートプロトコルです。送信側と受信側の両方のインスタンスでネットワークインターフェイスをアタッチできるように ENA Express を有効にすると、インスタンスが SRD テクノロジーで実現されるパフォーマンス向上を全面的に享受できることを、ENA Express メトリクスを使用して容易に確認できます。例えば、次のようになります。

- より多くの SRD 接続を確立するのに十分な容量があることを確認するために、リソースを評価します。
- 対象となる送信パケットで SRD を使用できない原因となる潜在的な問題がある箇所を特定します。
- インスタンスに SRD を使用する送信トラフィックの割合を計算します。
- インスタンスに SRD を使用する受信トラフィックの割合を計算します。

> **Note**
>
> メトリクスを生成するには、ドライバーバージョン 2.8 以降を使用してください。

ENA Express 用にフィルタリングされた Linux インスタンスのメトリクスのリストを表示するには、ネットワークインターフェイスで以下の ethtool コマンドを実行します (ここでは `eth0` として表示されています)。`ena_srd_mode` メトリクスの値に注意してください。

```
[ec2-user ~]$ ethtool -S eth0 | grep ena_srd
NIC statistics:
 ena_srd_mode: 1
 ena_srd_tx_pkts: 0
 ena_srd_eligible_tx_pkts: 0
 ena_srd_rx_pkts: 0
 ena_srd_resource_utilization: 0
```

次のメトリクスは、ENA Express が有効になっているすべてのインスタンスで利用できます。

ena_srd_mode

ENA Express のどの機能が有効になっているかを説明します。値は次のとおりです。

- 0 = ENA Express がオフ、UDP がオフ
- 1 = ENA Express がオン、UDP がオフ
- 2 = ENA Express がオフ、UDP がオン

> **Note**
>
> これは、ENA Express が最初に有効になっていて、UDP がそれを使用するように設定されている場合にのみ発生します。UDP トラフィックの以前の値は保持されます。

- 3 = ENA Express がオン、UDP がオン

ena_srd_eligible_tx_pkts

ネットワークの数。次のとおりです。

- 送信側と受信側の両方のインスタンスタイプがサポートされています。詳細については、ENA Express でサポートされるインスタンスタイプの表を参照してください。
- 送信側と受信側の両方のインスタンスに ENA Express が設定されている必要があります。
- 送信側と受信側のインスタンスは、同じリージョンで実行されている必要があります。
- インスタンス間のネットワークパスには、ミドルウェアボックスを含めないようにしてください。ENA Express は現在、ミドルウェアボックスをサポートしていません。

> **Note**
>
> ENA Express の適格性メトリクスには、ソースと送信先の要件、および 2 つのエンドポイント間のネットワークが含まれます。対象となるパケットは、既にカウントされた後でも失格となる可能性があります。例えば、対象となるパケットが最大送信単位 (MTU) の制限を超えている場合、そのパケットはカウンターに適格として反映されますが、標準の ENA 送信にフォールバックします。

ena_srd_tx_pkts

一定期間内に送信した SRD パケット数。

ena_srd_rx_pkts

一定期間内に受信した SRD パケット数。

ena_srd_resource_utilization

インスタンスが消費した同時 SRD 接続の最大許容メモリ使用量の割合。

パケット送信に SRD が使用されているかどうかを確認するには、対象パケットの数 (ena_srd_eligible_tx_pkts メトリクス) と特定の期間に送信された SRD パケットの数 (ena_srd_tx_pkts メトリクス) を比較します。

エグレストラフィック (送信パケット)

エグレストラフィックが想定どおりに SRD を使用するようにするには、SRD 送信に適格なパケットの数 (ena_srd_eligible_tx_pkts) と、特定の期間に送信された SRD パケットの数 (ena_srd_tx_pkts) を比較します。

使用可能なパケット数と送信された SRD パケット数の差が大きい場合、リソース使用率の問題が原因である可能性が高いです。インスタンスにアタッチされたネットワークカードが最大リソースを使い果たしている場合、またはパケットが MTU 制限を超えている場合、適格なパケットでも SRD 経由で送信できないため、標準の ENA 送信にフォールバックする必要があります。ライブ移行中やライブサーバー更新中にもパケットがこのギャップに陥る可能性があります。根本原因を特定するには、追加のトラブルシューティングが必要です。

> **Note**
>
> 適格なパケット数と SRD パケット数のわずかな違いは無視してかまいません。これは、例えば、インスタンスが SRD トラフィック用に別のインスタンスへの接続を確立した場合に発生する可能性があります。

特定の期間における総エグレストラフィックの何パーセントで SRD が使用されているかを調べるには、送信された SRD パケット数 (ena_srd_tx_pkts) と、その期間にインスタンスに送信されたパケットの総数 (NetworkPacketOut) を比較します。

イングレストラフィック (受信パケット)

総イングレストラフィックの何パーセントで SRD が使用されているかを調べるには、特定の期間に受信した SRD パケット数 (ena_srd_rx_pkts) と、その期間にインスタンスで受信されたパケットの総数 (NetworkPacketIn) を比較します。

リソース使用率

リソース使用率は、1 つのインスタンスが一定時間に保持できる SRD の同時接続数に基づいています。リソース使用率メトリクス (ena_srd_resource_utilization) は、インスタンスの現在の使用率を追跡します。使用率が 100% に近づくと、パフォーマンスの問題が発生することが予想されます。ENA Express は SRD から標準の ENA 送信にフォールバックし、パケットドロップの可能性が高まります。リソース使用率が高い場合は、ネットワークパフォーマンスを向上させるためにインスタンスをスケールアウトする時期が来たと判断できます。

> **Note**
>
> インスタンスのネットワークトラフィックが最大値を超えると、AWS はネットワークパケットをキューイングしてから破棄することによって、最大値を超えるトラフィックを調整します。

永続的

エグレスメトリクスとイングレスメトリクスは、インスタンスで ENA Express が有効になっている間に発生します。ENA Express が非アクティブ化されるとメトリクスの発生しなくなりますが、インスタンスがまだ実行されている限り持続します。インスタンスが再起動または終了した場合、またはネットワークインターフェイスがインスタンスから切り離されると、メトリクスはリセットされます。

ENA 用の DPDK ドライバーを備えたネットワークパフォーマンスメトリクス

ENA ドライババージョン 2.2.0 以降では、ネットワークメトリクスのレポートがサポートされています。DPDK 20.11 には ENA ドライバー 2.2.0 が含まれており、この機能をサポートする最初の DPDK バージョンです。

DPDK ドライバー v25.03 以降では、フラグメントプロキシモードがサポートされています。DPDK ドライバーでフラグメントプロキシモードが有効になっている場合、エグレスフラグメントドロップは、通常であれば適用される 1024 PPS の制限を回避し、標準の PPS 許容値内でカウントされます。フラグメントプロキシモードはデフォルトで無効になっています。

サンプルアプリケーションを使用して、DPDK 統計を表示できます。サンプルアプリケーションの対話型バージョンを開始するには、次のコマンドを実行します。

```
./app/dpdk-testpmd -- -i
```

この対話型セッションでは、コマンドを入力してポートの拡張統計情報を取得できます。次のコマンド例では、ポート 0 の統計情報を取得します。

```
show port xstats 0
```

次に、DPDK サンプルアプリケーションとの対話型セッションの例を示します。

```
[root@ip-192.0.2.0 build]# ./app/dpdk-testpmd -- -i
        EAL: Detected 4 lcore(s)
        EAL: Detected 1 NUMA nodes
        EAL: Multi-process socket /var/run/dpdk/rte/mp_socket
        EAL: Selected IOVA mode 'PA'
        EAL: Probing VFIO support...
        EAL:   Invalid NUMA socket, default to 0
        EAL:   Invalid NUMA socket, default to 0
        EAL: Probe PCI driver: net_ena (1d0f:ec20) device: 0000:00:06.0
(socket 0)
        EAL: No legacy callbacks, legacy socket not created
        Interactive-mode selected

        Port 0: link state change event
        testpmd: create a new mbuf pool <mb_pool_0>: n=171456,
size=2176, socket=0
        testpmd: preferred mempool ops selected: ring_mp_mc

        Warning! port-topology=paired and odd forward ports number, the
last port will pair with itself.

        Configuring Port 0 (socket 0)
        Port 0: 02:C7:17:A2:60:B1
        Checking link statuses...
        Done
        Error during enabling promiscuous mode for port 0: Operation
not supported - ignore
        testpmd> show port xstats 0
        ###### NIC extended statistics for port 0
        rx_good_packets: 0
        tx_good_packets: 0
        rx_good_bytes: 0
        tx_good_bytes: 0
        rx_missed_errors: 0
        rx_errors: 0
        tx_errors: 0
```

```
        rx_mbuf_allocation_errors: 0
        rx_q0_packets: 0
        rx_q0_bytes: 0
        rx_q0_errors: 0
        tx_q0_packets: 0
        tx_q0_bytes: 0
        wd_expired: 0
        dev_start: 1
        dev_stop: 0
        tx_drops: 0
        bw_in_allowance_exceeded: 0
        bw_out_allowance_exceeded: 0
        pps_allowance_exceeded: 0
        conntrack_allowance_exceeded: 0
        linklocal_allowance_exceeded: 0
        rx_q0_cnt: 0
        rx_q0_bytes: 0
        rx_q0_refill_partial: 0
        rx_q0_bad_csum: 0
        rx_q0_mbuf_alloc_fail: 0
        rx_q0_bad_desc_num: 0
        rx_q0_bad_req_id: 0
        tx_q0_cnt: 0
        tx_q0_bytes: 0
        tx_q0_prepare_ctx_err: 0
        tx_q0_linearize: 0
        tx_q0_linearize_failed: 0
        tx_q0_tx_poll: 0
        tx_q0_doorbells: 0
        tx_q0_bad_req_id: 0
        tx_q0_available_desc: 1023
        testpmd>
```

サンプルアプリケーションとその使用による拡張統計情報の取得の詳細については、DPDK ドキュメントのTestpmd アプリケーションユーザーガイドを参照してください。

FreeBSD を実行しているインスタンスのメトリクス

ENA FreeBSD ドライバーのバージョン 2.3.0 以降では、FreeBSD を実行しているインスタンスでのネットワークパフォーマンスメトリクスの収集をサポートしています。FreeBSD メトリクスの収集を有効にするには、次のコマンドを入力し、[##]を 1 ~ 3600 の値に設定します。FreeBSD メトリクスを収集する頻度を秒単位で特定します。

```
sysctl dev.ena.network_interface.eni_metrics.sample_interval=interval
```

例えば、次のコマンドは、ネットワークインターフェイス 1 の FreeBSD メトリクスを 10 秒ごとに収集するようにドライバーを設定します。

```
sysctl dev.ena.1.eni_metrics.sample_interval=10
```

FreeBSD メトリクスの収集をオフにするには、上記のコマンドを実行し、[##]を 0 に指定します。

FreeBSD メトリクスの収集を有効にすると、次のコマンドを実行することで、収集されたメトリクスの最新セットを取得できます。

```
sysctl dev.ena.network_interface.eni_metrics
```

Linux ベースの EC2 インスタンスのネットワークレイテンシーを改善する

ネットワークレイテンシーとは、データのパケットが送信元から送信先に移動するまでにかかる時間です。ネットワークを介してデータを送信するアプリケーションは、快適なユーザーエクスペリエンスを提供するためにタイムリーな応答が不可欠です。ネットワークレイテンシーが長くなると、次のようなさまざまな問題が発生する可能性があります。

- ウェブページのロードが遅い
- ビデオストリーミングのタイムラグ
- オンラインリソースへのアクセスが難しい

このセクションでは、Linux で実行される Amazon EC2 インスタンスのネットワークレイテンシーを改善するために実行できる手順の概要を説明します。最適なレイテンシーを実現するには、以下の手順に従ってインスタンス、カーネル、および ENA ドライバーの設定を行います。その他の設定ガイダンスについては、GitHub の「ENA Linux ドライバーのベストプラクティスとパフォーマンス最適化ガイド」を参照してください。

> **Note**
>
> 手順と設定は、特定のネットワークハードウェア、インスタンスを起動した AMI、およびアプリケーションのユースケースによって若干異なる場合があります。変更を加える前に、ネットワークパフォーマンスを徹底的にテストおよびモニタリングして、望ましい結果が得られることを確認してください。

データパケットのネットワークホップの数を減らす

データパケットがルーター間を移動する (ホップする) たびにネットワークレイテンシーが増加します。通常、トラフィックが送信先に到達するには、複数のホップを経由する必要があります。Amazon EC2 インスタンスのネットワークホップを減らすには、次の 2 つの方法があります。

- クラスタープレイスメントグループ – クラスタープレイスメントグループを指定すると、Amazon EC2 は、同じアベイラビリティーゾーン (AZ) 内の物理的に互いに近いインスタンスをよりタイトなパッキングで起動します。グループ内のインスタンスが物理的に近接していることにより、高速接続を利用できるため、レイテンシーは低く、単一フローのスループットは高くなります。
- 専有ホスト – 専有ホストはお客様専用の物理サーバーです。専有ホストを使用すると、インスタンスを起動して同じ物理サーバー上で実行できます。同じ専有ホストで実行されるインスタンス間では、余分なネットワークホップなしで通信できます。

Linux カーネルの設定がレイテンシーに与える影響

Linux カーネル設定では、ネットワークレイテンシーを増減できます。レイテンシー最適化の目標を達成するには、ワークロードの特定の要件に応じて Linux カーネル設定を微調整することが重要です。

Linux カーネルには、ネットワークレイテンシーを減らすのに役立つ設定オプションが多数あります。最も影響の大きいオプションは次のとおりです。

- ビジーポーリングモードを有効にする – ビジーポーリングモードを使用すると、ネットワーク受信パスのレーテンシーが減少します。ビジーポーリングモードを有効にすると、ソケットレイヤーコードはネットワークデバイスの受信キューを直接ポーリングできます。ビジーポーリングの欠点は、タイトなループで新しいデータをポーリングすることにより、ホストの CPU 使用率が高くなることです。すべてのインターフェイスでパケットを待機するマイクロ秒数を制御するグローバル設定は 2 つあります。

  `busy_read`

  ソケット読み込み時の低レイテンシーのビジーポーリングタイムアウト。これにより、ソケット層がデバイスキューのパケットを読み取るまでのマイクロ秒数を制御します。sysctl コマンドを使用して機能をグローバルに有効にするには、Linux カーネル組織は 50 マイクロ秒の値を推奨しています。詳細については、「Linux カーネルユーザーおよび管理者ガイド」の「busy_read」を参照してください。

```
[ec2-user ~]$ sudo sysctl -w net.core.busy_read=50
```

busy_poll

ポーリングとセレクティングの低レイテンシーのビジーポーリングタイムアウト。これにより、イベント待ち時間のマイクロ秒数を制御します。推奨値は 50 ~ 100 マイクロ秒で、ポーリングするソケットの数によって異なります。追加するソケットが多いほど、数値は大きい必要があります。

```
[ec2-user ~]$ sudo sysctl -w net.core.busy_poll=50
```

- CPU 電源状態の設定 (C ステート) – C ステートは非アクティブ時のコアのスリープレベルを制御します。C ステートを制御して、システムのレイテンシーとパフォーマンスを調整することができます。より深い C ステートでは、CPU は基本的に「スリープ」状態になり、起動してアクティブ状態に戻るまでリクエストに応答できません。コアをスリープ状態にするには時間がかかります。また、スリープ状態のコアによって、別のコアが高い周波数で動作するための余裕が生まれますが、そのスリープ状態にあるコアが再び稼働し処理を実行するのにも時間がかかります。

  例えば、ネットワークパケットの中断を処理するように割り当てられたコアがスリープ状態である場合、その中断の処理に遅延が生じる可能性があります。より深い C ステートを使用しないようにシステムを設定できます。ただし、この設定では、プロセッサの反応レイテンシーは短縮されますが、他のコアの Turbo Boost 用のヘッドルームも減少します。

  プロセッサーの反応レイテンシーを短縮するために、C ステートが深くなるのを制限できます。詳細については、「Amazon Linux 2 ユーザーガイド」の「深い C ステートの制限による高パフォーマンスと低レイテンシー」を参照してください。

割り込みモデレーション

ENA ネットワークドライバーを使用すると、インスタンスとネットワーク間の通信が可能になります。ドライバーはネットワークパケットを処理し、ネットワークスタックまたは Nitro Card に渡します。ネットワークパケットが受信されると、Nitro Card は割り込みを生成して CPU がソフトウェアにイベントを通知します。

割り込み

割り込みは、デバイスまたはアプリケーションがプロセッサに送信する信号です。割り込みは、イベントが発生したこと、または即時の注意を要する条件が満たされたことをプロセッサに通知

します。割り込みにより、ネットワークインターフェイスからのデータ受信、ハードウェアイベントの処理、他のデバイスからのリクエストの処理など、時間的制約のあるタスクを処理できます。

割り込みモデレーション

割り込みモデレーションは、割り込みを集約または遅延させることにより、デバイスが生成する割り込みの数を減らす手法です。割り込みモデレーションの目的は、多数の割り込みの処理に伴うオーバーヘッドを削減することによってシステムパフォーマンスを向上させることです。割り込みが多すぎると CPU 使用率が高くなり、スループットに悪影響を及ぼします。一方、割り込みが少なすぎると、レイテンシーが長くなります。

動的割り込みモデレーション

動的割り込みモデレーションは、現在のシステム負荷とトラフィックパターンに基づいて割り込みレートを動的に調整する、割り込みモデレーションの拡張形です。割り込みオーバーヘッドの削減と、1 秒あたりのパケット数 (つまり帯域幅) とのバランスをとることを目的としています。

> **Note**
>
> 動的割り込みモデレーションは、一部の AMI ではデフォルトで有効になっています (ただし、すべての AMI で有効または無効にできます)。

ネットワークレイテンシーを最小限に抑えるために、割り込みモデレーションを無効にする必要があることがあります。ただし、これによって割り込み処理のオーバーヘッドが増加する可能性もあります。レイテンシーの低減とオーバーヘッドの最小化とのバランス点を見つけることが重要です。ethtool コマンドは割り込みモデレーションの設定に役立ちます。デフォルトでは、rx-usecs は 20 に設定され、tx-usecs は 64 に設定されています。

現在の割り込み変更設定を取得するには、次のコマンドを使用します。

```
[ec2-user ~]$ ethtool -c interface | egrep "rx-usecs:|tx-usecs:|Adaptive RX"
Adaptive RX: on  TX: off
rx-usecs: 20
tx-usecs: 64
```

割り込み変更および動的割り込みモデレーションを無効にするには、次のコマンドを使用します。

```
[ec2-user ~]$ sudo ethtool -C interface adaptive-rx off rx-usecs 0 tx-usecs 0
```

Nitro System のパフォーマンスチューニングに関する考慮事項

Nitro System は、AWS が構築した、高パフォーマンス、高可用性、高度なセキュリティを実現するハードウェアとソフトウェアコンポーネントのコレクションです。Nitro System は、ベアメタルのような機能を備えることで、仮想化オーバーヘッドを排除するとともに、ホストハードウェアへのフルアクセスを要求するワークロードをサポートします。詳細については、「AWS Nitro System」を参照してください。

現行世代の EC2 インスタンスタイプはすべて、ネットワークパケット処理を EC2 Nitro カードで実行します。このトピックでは、Nitro Card でのハイレベルなパケット処理、パケット処理のパフォーマンスに影響するネットワークアーキテクチャおよび設定の一般的側面、Nitro ベースインスタンスのパフォーマンスを最大化するために実行できるアクションについて説明します。

Nitro Card は、仮想プライベートクラウド (VPC) に必要な、すべての入出力 (I/O) インターフェイスを処理します。ネットワーク上で情報を送受信するすべてのコンポーネントで、Nitro Card は I/O トラフィック用の自己完結型のコンピューティングデバイスとして機能します。このコンピューティングデバイスは、顧客のワークロードを実行するシステムメインボードとは物理的に分離されています。

Nitro Card でのネットワークパケットフロー

Nitro System 上に構築された EC2 インスタンスには、1 秒あたりのパケット数 (PPS) スループットレートで測定されるよりも高速なパケット処理を可能にする、ハードウェアアクセラレーション機能があります。Nitro Card が新しいフローの初期評価を行うと、セキュリティグループ、アクセスコントロールリスト、ルートテーブルエントリなど、フロー内のすべてのパケットについて同じ情報が保存されます。同じフローの追加のパケットを処理する場合、保存した情報を使用することで、それらのパケットのオーバーヘッドを減らすことができます。

接続速度は 1 秒あたりの接続数 (CPS) で測定されます。新しい接続が発生するたびに、追加の処理オーバーヘッドが必要になるため、ワークロード能力の予測でそれを考慮する必要があります。ワークロードを設計するときは、CPS と PPS の両方のメトリクスを考慮することが重要です。

接続の確立方法

Nitro ベースのインスタンスと別のエンドポイントとの間で接続が確立されると、Nitro Card は 2 つのエンドポイント間で送受信される最初のパケットのフルフローを評価します。同じフローの後続のパケットについては、通常、完全な再評価は不要です。ただし、例外がいくつかあります。例外に関する詳細は、「ハードウェアアクセラレーションを使用しないパケット」を参照してください。

以下のプロパティは、2 つのエンドポイントとそれらの間のパケットフローを定義します。これら 5 つのプロパティを総合して、5 タプルフローと呼びます。

- [Source IP] (送信元 IP)
- ソースポート
- 送信先 IP
- 発信先ポート
- 通信プロトコル

パケットフローの方向は、イングレス (インバウンド) とエグレス (アウトバウンド) と呼びます。以下は、エンドツーエンドのネットワークパケットフローの概要を示したものです。

- イングレス — Nitro Card は、受信ネットワークパケットを処理するとき、パケットをステートフルファイアウォールルールとアクセス制御リストに照らして評価します。接続を追跡し、測定し、必要に応じてその他アクションを実行します。次に、パケットをホスト CPU 上の宛先に転送します。
- エグレス — Nitro Card は、アウトバウンドネットワークパケットを処理するとき、リモートインターフェイスの宛先を検索し、さまざまな VPC 機能を評価し、レート制限を適用して、適用されるその他アクションを実行します。次に、パケットをネットワーク上の次のホップの宛先に転送します。

最適なパフォーマンスが得られるようにネットワークを設計する

Nitro System のパフォーマンス機能を活用するには、ネットワーク処理のニーズを理解し、そしてそれらのニーズが Nitro リソースのワークロードにどのように影響するかを理解する必要があります。そうすれば、ネットワーク環境に最適なパフォーマンスを実現するように、設計することができます。インフラストラクチャ設定とアプリケーションワークロードの設計および構成は、パケット処理と接続速度の両方に影響を与える可能性があります。例えば、DNS サービス、ファイアウォール、仮想ルーターなど、アプリケーションの接続確立率が高いと、接続が確立された後にのみ発生するハードウェアアクセラレーションを利用できる機会が減ります。

アプリケーションとインフラストラクチャの構成を設定し、ワークロードを効率化して、ネットワークパフォーマンスを改善することができます。ただし、すべてのパケットがアクセラレーションの対象となるわけではありません。Nitro System は、新しい接続や、アクセラレーションの対象ではないパケットには、ネットワークフローをすべて使用します。

本セクションの残りの部分では、パケットが可能な限り高速パス内を流れるようにするための、アプリケーションとインフラストラクチャの、設計上の考慮事項に焦点を当てます。

Nitro システムのネットワーク設計に関して考慮すべき事項

インスタンスのネットワークトラフィックを設定するときは、PPS のパフォーマンスに影響するさまざまな側面を考慮する必要があります。フローが確立されると、定期的に送受信されるパケットの大半が、アクセラレーションの対象になります。ただし、インフラストラクチャ設計とパケットフローが引き続きプロトコル標準を満たすようにするため、いくつかの例外が存在します。

Nitro Card から最高のパフォーマンスを引き出すには、使用しているインフラストラクチャとアプリケーションに関する以下の設定の詳細の、長所と短所を慎重に検討する必要があります。

インフラストラクチャの考慮事項

インフラストラクチャの設定は、パケットフローと処理効率に影響を及ぼす可能性があります。以下は、重要な考慮事項の一覧です。

非対称ネットワークインターフェイス構成

セキュリティグループは、接続追跡を使用して、インスタンスに出入りするトラフィックの情報を追跡します。トラフィックが特定のネットワークインターフェースからインスタンスに入り、別のネットワークインターフェースから外に出る、非対称ルーティングでは、フローを追跡した場合に、インスタンスが達成できるピークパフォーマンスが低下する可能性があります。セキュリティグループの接続トラッキング、追跡されない接続、自動的に追跡される接続の詳細については、「Amazon EC2 セキュリティグループの接続の追跡」を参照してください。

ネットワークドライバー

ネットワークドライバーは、定期的に更新されリリースされています。ドライバーが古くなっていると、パフォーマンスが大幅に低下する可能性があります。最新のパッチを適用して、最新世代のドライバーのみで利用できるアクセラレーションパス機能などのパフォーマンス強化を活用できるよう、ドライバーを常に最新状態に保つようにします。旧世代のドライバーでは、アクセラレーションパス機能はサポートされていません。

アクセラレーションパス機能を利用するため、最新の ENA ドライバーをインスタンスにインストールすることが推奨されます。

Linux インスタンス – ENA Linux ドライバー 2.2.9 以降。Amazon Drivers GitHub リポジトリから ENA Linux ドライバーをインストールまたは更新するには、readme ファイルの「ドライバーのコンパイル」のセクションを参照してください。

Windows インスタンス – ENA Windows ドライバー 2.0.0 以降。ENA Windows ドライバーをインストールまたは更新するには、「EC2 Windows インスタンスに ENA ドライバーをインストールする」を参照してください。

エンドポイント間の距離

同じアベイラビリティーゾーン内の 2 つのインスタンス間の接続は、リージョン間の接続よりも 1 秒あたりで多くのパケットを処理できます。これは、アプリケーション層での TCP ウィンドウ処理により、いつでも送信できるデータの量が決定されるためです。インスタンス間の距離が長いとレイテンシーが長くなり、エンドポイントで処理できるパケットの数が減少します。

バイトキュー制限 (BQL)

BQL は、Nitro Card に渡されるバイト数を制限してキューイングを減らす機能です。BQL は、ENA ドライバー、Amazon Linux オペレーティングシステム、およびほとんどの Linux ディストリビューションでデフォルトで無効になっています。BQL とフラグメントプロキシの上書きの両方が有効になっている場合、すべてのフラグメントが処理される前に Nitro に渡されるバイト数が制限されることにより、パフォーマンスが低下する可能性があります。

アプリケーションの設計に関する考慮事項

アプリケーションの設計と構成には、処理効率に影響する要素がいくつか存在します。以下は、重要な考慮事項の一覧です。

バケットサイズ

パケットサイズを大きくすると、インスタンスがネットワーク上で送受信できるデータのスループットが増加します。Amazon EC2 では 9001 バイトのジャンボフレームがサポートされますが、他のサービスでは異なる制限が適用される場合があります。パケットサイズを小さくすると、パケットの処理速度は上がりますが、パケット数が PPS の許容値を超えたときに達成される最大帯域幅が、減少する可能性があります。

パケットのサイズがネットワークホップの最大転送単位 (MTU) を超えると、パス上のルーターがパケットをフラグメント化する可能性があります。生成されたパケットフラグメントは例外とみなされ、通常は標準速度 (高速ではない) で処理されます。これにより、パフォーマンスにばらつきが生じる可能性があります。ただし、フラグメントプロキシモード設定を使用して、アウトバウンドフラグメントパケットの標準の動作を上書きできます。詳細については、「Nitro System でのネットワークパフォーマンスを最大化する」を参照してください。MTU を設定するときは、トポロジを評価することが推奨されます。

プロトコルのトレードオフ

TCP のような信頼性の高いプロトコルは、UDP のような信頼性の低いプロトコルよりもオーバーヘッドが大きくなります。UDP トランスポートプロトコルの低いオーバーヘッドとシンプルなネットワーク処理により、PPS レートを高くすることができますが、信頼性の高いパケット配信が犠牲になります。アプリケーションにとって信頼性の高いパケット配信が重要でない場合は、UDP が適しているかもしれません。

マイクロバースティング

マイクロバーストは、トラフィックが均等に分散されているときではなく、短時間で許容値を超えたときに発生します。通常これは、マイクロ秒単位で発生します。

例えば、最大 10 Gbps を送信できるインスタンスがあり、アプリケーションが 10 Gb を 0.5 秒で送信するとします。このマイクロバーストは最初の 0.5 秒で許容値を超え、残りの時間には何も残りません。10 Gb を 1 秒以内に 送信したとしても、最初の 0.5 秒間に余裕があると、パケットがキューに入れられたりドロップされたりする可能性があります。

Linux Traffic Control などのネットワークスケジューラを使用すると、スループットを調整でき、マイクロバーストによってパケットがキューに入れられたりドロップされたりすることを、防ぐことができます。

フロー数

1 つのフローは、最大 10 Gbps をサポートするクラスタープレイスメントグループ内にある場合や、最大 25 Gbps をサポートする ENA Express を使用している場合を除いて、5 Gbps に制限されます。

同様に、Nitro Card は、1 つのフローを使用する場合と比べて、複数のフローで、より多くのパケットを処理できます。インスタンスあたりの、ピークのパケット処理速度を達成するには、総帯域幅が 100 Gbps 以上のインスタンスで、100 フロー以上にすることが推奨されます。総帯域幅の容量が増えると、ピーク処理速度の達成に必要なフローの数も増えます。ベンチマークは、ネットワークのピーク速度を達成するために、どのような構成が必要かを判断するのに役立ちます。

Elastic Network Adapter (ENA) キュー

ENA (Elastic Network Adapter) は、複数の受信 (Rx) キューと転送 (Tx) キュー (ENA キュー) を使用して、EC2 インスタンスのネットワークパフォーマンスとスケーラビリティを向上させます。これらのキューは、利用可能なキュー全体で送受信されるデータの負荷を分散させることで、ネットワークトラフィックを効率的に管理します。

詳細については、「the section called “ENA キュー”」を参照してください。

特徴量処理のオーバーヘッド

トラフィックミラーリングや ENA Express などの機能では、処理オーバーヘッドが増加して、パケット処理の絶対的パフォーマンスが低下する可能性があります。パケットの処理速度は、機能の使用を制限したり、無効にしたりすることで上げることができます。

状態を維持するための接続トラッキング

セキュリティグループは、接続トラッキングを使用して、インスタンスに出入りするトラフィックに関する情報を追跡します。接続トラッキングは、ネットワークトラフィックの各フローにルールを適用して、そのトラフィックが許可されているか拒否されているかを判定します。Nitro Card は、フロートラッキングを使用してフローの状態を維持します。適用されるセキュリティグループのルールが増えれば、フローを評価するための作業も増えます。

> **Note**
>
> トラフィックフローはすべて追跡されるわけではありません。セキュリティグループルールが 追跡されていない接続 で設定されている場合は、有効な応答パスが複数あるときに対称ルーティングを確保するため接続を自動的に追跡する場合を除いて、追加の作業は必要ありません。

ハードウェアアクセラレーションを使用しないパケット

ハードウェアアクセラレーションは、すべてのパケットで利用できるわけではありません。これらの例外の処理には、ネットワークフローの状態を確認するために必要な、処理オーバーヘッドが伴います。ネットワークフローは、プロトコル標準を確実に満たし、VPC 設計の変更に従い、許可された宛先にのみパケットを送信する必要があります。ただし、オーバーヘッドはパフォーマンスを低下させます。

パケットフラグメント

「アプリケーションの考慮事項」で述べたように、ネットワーク MTU を超えるパケットから発生するパケットフラグメントは、通常は例外として処理され、ハードウェアアクセラレーションを利用することはできません。ただし、ドライバーのバージョンによっては、フラグメントプロキシモードでエグレスフラグメントの制限を回避できます。詳細については、「Nitro System でのネットワークパフォーマンスを最大化する」セクションで実行可能なアクションを確認してください。

アイドル接続

接続がタイムアウト制限に達していなくても、接続に一定時間アクティビティがないと、システムはその優先順位を下げることができます。優先順位が下がった後にデータが入ってきた場合、システムは、再度接続するために、そのデータを例外として処理する必要があります。

接続を管理するには、接続トラッキングタイムアウトを使用してアイドル接続を終了します。TCP キープアライブを使用して、アイドル接続を開放しておくこともできます。詳細については、「アイドル接続追跡タイムアウト」を参照してください。

VPC ミューテーション

セキュリティグループ、ルートテーブル、アクセスコントロールリストの更新は、すべて処理パスで再評価して、ルートエントリとセキュリティグループのルールが、想定どおりに適用されることを確認する必要があります。

ICMP フロー

インターネット制御メッセージプロトコル (ICMP) は、ネットワーク通信の問題を診断する場合にネットワークデバイスで使用するネットワーク層プロトコルです。これらのパケットでは、常にフルフローが使用されます。

非対称 L2 フロー

NitroV3 以前のプラットフォームでは、同一サブネット内の 2 つの ENI 間のトラフィックにおいて、片方の ENI がデフォルトゲートウェイルーターを使用し、もう片方が使用しない構成の場合、ハードウェアアクセラレーションは有効になりません。NitroV4 以降のプラットフォームでは、このシナリオにおいてハードウェアアクセラレーションが有効になります。NitroV3 以前のプラットフォームでのパフォーマンスを向上させるには、両方の ENI 間で使用されているデフォルトのゲートウェイルーター、またはそれらの ENI が異なるサブネットにあることを確認してください。

Nitro System でのネットワークパフォーマンスを最大化する

Nitro システムのネットワークパフォーマンスは、ネットワーク設定を調整することで最大化できます。

トピック

- 考慮事項
- PPS パフォーマンスの調整

- ENA キュー割り当てを設定する
- Linux インスタンスのパフォーマンスを監視する

考慮事項

設計上の決定や、インスタンスのネットワーク設定の調整を行う際には、ベストな結果が得られるよう、事前に以下の手順を実行しておくことが推奨されます。

1. Nitro システムのネットワーク設計に関して考慮すべき事項 をレビューして、パフォーマンスの向上に役立つアクションの、長所と短所を理解しておきます。

   Linux でのインスタンス設定に関するその他の考慮事項とベストプラクティスについては、GitHub の「ENA Linux Driver Best Practices and Performance Optimization Guide」を参照してください。
2. ピーク時のアクティブフロー数でワークロードをベンチマークし、アプリケーションパフォーマンスのベースラインを決定します。パフォーマンスベースラインを使用すると、設定やアプリケーション設計のバリエーションをテストして、特にスケールアップやスケールアウトを計画している場合に、どの考慮事項が最も大きな影響を与えるかを判断することができます。

PPS パフォーマンスの調整

以下は、システムのニーズに応じて PPS のパフォーマンスを調整する際に活用できるアクションの一覧です。

- 2 つのインスタンス間の物理的な距離を短くします。送信側と受信側のインスタンスが同じアベイラビリティーゾーンにある場合や、クラスタープレイスメントグループを使用している場合は、パケットが 1 つのエンドポイントから別のエンドポイントに移動する際に必要になるホップの数を、減らすことができます。
- 追跡されていない接続 を使用します。
- ネットワークトラフィックには UDP プロトコルを使用します。
- 総帯域幅が 100 Gbps 以上の EC2 インスタンスでは、Nitro Card 全体に処理を均等に分散するために、100 以上の個別のフローにワークロードを分散します。
- EC2 インスタンスのエグレスフラグメント PPS 制限を克服するには、フラグメントプロキシモードを有効にします (これはドライバーのバージョンによってできる場合とできない場合があります)。この設定により、フラグメント化されたパケットが処理パスで評価されるようになるため、エグレス PPS 制限の 1024 を克服できます。ドライバーをロードするときは、次のいずれかのコマンドを実行して、フラグメントプロキシモードを有効または無効にします。

フラグメントプロキシモードを有効にする

```
sudo insmod ena.ko enable_frag_bypass=1
```

フラグメントプロキシモードを無効にする

```
sudo insmod ena.ko enable_frag_bypass=0
```

ENA キュー割り当てを設定する

サポートされているインスタンスタイプでは、これらのキューを Elastic Network Interface (ENI) 全体に動的に割り当てることができます。柔軟な ENA キューの割り当てはリソース配分を最適化するため、vCPU を最大限に活用できます。通常、ネットワークパフォーマンスの高いワークロードには複数の ENA キューが必要です。詳細については、「ENA キュー」を参照してください。

Linux インスタンスのパフォーマンスを監視する

Linux インスタンスの Ethtool メトリクスを使用することで、帯域幅、パケットレート、接続トラッキングなどの、インスタンスのネットワークパフォーマンス指標を監視することができます。詳細については、「EC2 インスタンスでの ENA 設定のネットワークパフォーマンスのモニタリング」を参照してください。

EC2 Windows インスタンスでのネットワークパフォーマンスを最適化する

ネットワーキングが拡張された Windows インスタンスで最良のネットワークパフォーマンスを実現するには、デフォルトのオペレーティングシステム設定を変更する必要がある場合があります。高いネットワークパフォーマンスを必要とするアプリケーションには、次の設定変更をお勧めします。その他の最適化 (チェックサムオフロードをオンにして RSS を有効にするなど) は、公式の Windows AMI で既に行われています。

> **Note**
>
> TCP Chimney オフロードはほとんどのユースケースで無効にする必要があり、Windows Server 2016 では廃止されています。

これらのオペレーティングシステムの最適化に加えて、ネットワークトラフィックの最大送信単位 (MTU) も考慮し、ワークロードとネットワークアーキテクチャーに応じて調整する必要がありま

す。詳細については、EC2 インスタンスのネットワークの最大送信単位 (MTU)を参照してください。

AWS では、99.9 パーセンタイルで 50us のクラスタープレイスメントグループで起動されたインスタンスと 200us のテールレイテンシーの間のラウンドトリップレイテンシーを定期的に測定しています。アプリケーションで一貫して低レイテンシーが必要な場合、Nitro System で構築された固定パフォーマンスインスタンスで最新バージョンの ENA ドライバーを使用することをお勧めします。

受信側スケーリング CPU アフィニティを設定する

受信側スケーリング (RSS) は、複数のプロセッサにネットワークトラフィック CPU 負荷を分配するために使用されます。デフォルトでは、公式 アマゾン Windows AMI は、RSS を有効にして設定され、RSS を許可します。ENA Elastic Network Interface では、最大 8 個の RSS キューを利用できます。RSS キューやその他のシステム処理の CPU アフィニティを定義することで、マルチコアシステムで CPU の負荷を分散することができ、より多くのネットワークトラフィックを処理できます。16 個を超える vCPU を備えたインスタンスタイプでは、`Set-NetAdapterRSS` PowerShell cmdlet を使用することをお勧めします。この cmdlet は、さまざまなシステムコンポーネントとの競合を防ぐために、すべての エラスティックネットワークインターフェイス の RSS 設定からブートプロセッサ (ハイパースレッディングが有効になっている場合は論理プロセッサ 0 と 1) を手動で除外します。

Windows はハイパースレッド対応であり、単一の Elastic Network Interface の RSS キューが常に異なる物理コアに配置されるようにします。したがって、ハイパースレッディングが無効になっていない限り、他の NIC との競合を完全に防ぐために、各 NIC の RSS 設定が 16 個の論理プロセッサの全体で分散されます。`Set-NetAdapterRss` cmdlet を使用すると、BaseProcessorGroup、BaseProcessorNumber、MaxProcessingGroup、MaxProcessorNumber、および NumaNode (オプション) の値を定義することによって、NIC ごとの有効な論理プロセッサの範囲を定義できます。 エラスティックネットワークインターフェイス 間の競合を完全に排除するのに十分な物理コアがない場合は、インターフェイスで想定されるワークロードに応じて、 エラスティックネットワークインターフェイス 範囲内の論理プロセッサの数を減らします (つまり、管理ネットワークインターフェイスが少量であれば、それほど多くの RSS キューを割り当てる必要はないということです)。また、前記のように、さまざまなコンポーネントを CPU 0 で実行する必要があるため、十分な vCPU が利用可能な場合は、すべての RSS 構成から除外することをお勧めします。

例えば、ハイパースレッディングが有効になっている NUMA ノードを 2 つ備えた 72 の vCPU インスタンスに Elastic Network Interface が 3 つある場合、以下のコマンドは 2 つの CPU 間でネットワーク負荷を重複なく分散させ、コア 0 の使用を完全に防ぎます。

```
Set-NetAdapterRss -Name NIC1 -BaseProcessorGroup 0 -BaseProcessorNumber 2 -
MaxProcessorNumber 16
Set-NetAdapterRss -Name NIC2 -BaseProcessorGroup 1 -BaseProcessorNumber 0 -
MaxProcessorNumber 14
Set-NetAdapterRss -Name NIC3 -BaseProcessorGroup 1 -BaseProcessorNumber 16 -
MaxProcessorNumber 30
```

これらの設定は各ネットワークアダプターに対して永続的であることに注意してください。インスタンスの vCPU 数が異なるインスタンスにサイズ変更された場合は、有効になっている Elastic Network Interface ごとに RSS 設定を再評価する必要があります。コマンドレットの完全な Microsoft ドキュメントは、Set-NetAdapterRss にあります。

SQL ワークロードに関する特記事項: 同じ CPU に対する I/O およびネットワークの競合を最小限に抑えるために、Elastic Network Interface RSS 構成と共に I/O スレッドアフィニティ設定を確認することをお勧めします。affinity mask Server Configuration Optionを参照してください。

Amazon EC2 の AI/ML および HPC ワークロード用の Elastic Fabric Adapter

Elastic Fabric Adapter (EFA) はAI、機械学習 (ML) およびハイパフォーマンスコンピューティング (HPC) のアプリケーションを高速化するために Amazon EC2 インスタンスにアタッチできるネットワークデバイスです。EFA ではAWS クラウドが提供するスケーラビリティ、柔軟性、伸縮性により、オンプレミス AI/ML または HPC クラスターのアプリケーションパフォーマンスを実現できます。

EFA ではクラウドベースの HPC システムで従来使用されていた TCP トランスポートよりも低く、一貫性の高いレイテンシーを提供し、高いスループットが得られます。AI/ML および HPC アプリケーションのスケーリングに不可欠なインスタンス間通信のパフォーマンスが向上します。既存の AWS ネットワークインフラストラクチャで動作するように最適化されており、アプリケーション要件に応じてスケールすることができます。

EFA は Libfabric と統合されており、AI および機械学習アプリケーション向けに Nvidia Collective Communications Library (NCCL) と NVIDIA Inference Xfer Library (NIXL)、および HPC アプリケーション向けに Open MPI 4.1 以降と Intel MPI 2019 Update 5 以降をサポートしています。NCCL と MPI は Libfabric 1.7.0 以降と統合されています。NIXL は Libfabric 1.21.0 以降と統合されています。

Nitro バージョン 4 以降でサポートされているほとんどのインスタンスタイプで、EFA は RDMA (リモートダイレクトメモリアクセス) 書き込みをサポートしています。RDMA 読み取りは、Nitro バー

ジョン 4 以降のすべてのインスタンスでサポートされています。詳細については「サポートされるインスタンスタイプ」を参照してください。

内容

- EFA の基本
- サポートされているインターフェイスとライブラリ
- サポートされるインスタンスタイプ
- サポートされるオペレーティングシステム
- EFA の制限事項
- EFA 価格設定
- Amazon EC2 の HPC ワークロードに対して EFA と MPI の使用を開始する
- Amazon EC2 の ML ワークロードに対して EFA と NCCL の使用を開始する
- Amazon EC2 の推論ワークロードに対して EFA と NIXL の使用を開始する
- 複数のネットワークカードを使用して Amazon EC2 インスタンスのネットワーク帯域幅を最大化する
- Elastic Fabric Adapter を作成して Amazon EC2 インスタンスにアタッチする
- Amazon EC2 インスタンスから EFA をデタッチおよび削除する
- Amazon EC2 で Elastic Fabric Adapter をモニタリングする
- チェックサムを使用した EFA インストーラの検証
- Elastic Fabric Adapter リリースノート

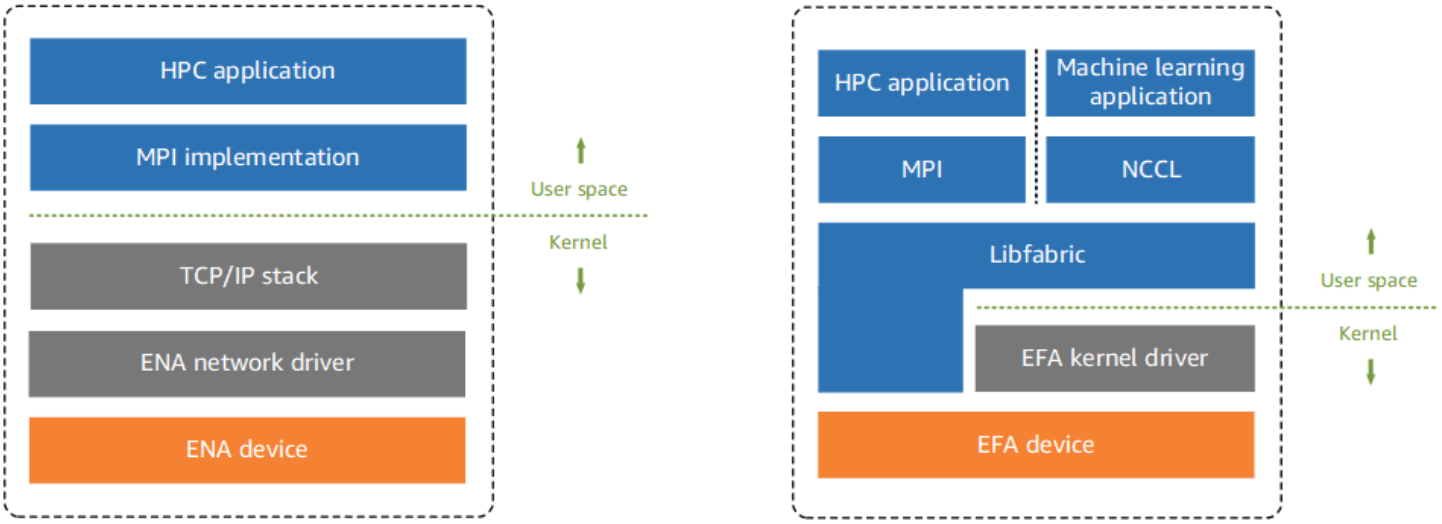
EFA の基本

EFA デバイスは次の 2 つの方法で EC2 インスタンスにアタッチできます。

1. EFA デバイスと ENA デバイスの両方を作成する EFA と ENA の組み合わせとも呼ばれる従来の EFA インターフェイスを使用します。
2. EFA のみのインターフェイスを使用して、EFA デバイスのみを作成します。

EFA デバイスはScalable Reliable Datagram (SRD) プロトコルを介した組み込み OS バイパスや輻輳制御などの機能を提供します。EFA デバイス機能により、低レイテンシーで信頼性の高いトランスポート機能が可能になり、EFA インターフェイスが Amazon EC2 の HPC および ML アプリケーションのパフォーマンスを向上させることができます。一方、ENA デバイスは従来の IP ネットワー

クを提供します。ユースケースに基づいて使用するインスタンス設定のガイダンスについては、「」を参照してくださいネットワーク帯域幅の最大化。



Traditional HPC software stack in EC2

HPC software stack in EC2 with EFA

従来、AI/ML アプリケーションは NCCL と NIXL (分散推論用) を使用します。HPC アプリケーションは、Message Passing Interface (MPI) を使用してシステムのネットワーク転送と連携します。AWS クラウドではアプリケーションが NCCL、NIXL または MPI と連携することを意味します。MPI はオペレーティングシステムの TCP/IP スタックと ENA デバイスドライバーを使用して、インスタンス間のネットワーク通信を行います。

従来の EFA (ENA を使用した EFA) または EFA 専用インターフェイスでは、AI/ML アプリケーションは NCCL と NIXL (分散推論用) を使用します。HPC アプリケーションは MPI を使用して Libfabric API と直接連携します。Libfabric API はオペレーティングシステムのカーネルをバイパスし、EFA デバイスと直接通信してパケットをネットワークに送ります。これにより、オーバーヘッドが削減され、AI/ML および HPC アプリケーションを効率的に実行できるようになります。

> **Note**
>
> Libfabric はOpenFabrics Interface (OFI) フレームワークのコアコンポーネントで、OFI のユーザースペース API を定義およびエクスポートします。詳細についてはLibfabric OpenFabricsウェブサイトを参照してください。

ENA、EFA、EFA のみのネットワークインターフェイスの違い

Amazon EC2 では次の 2 種類のネットワークインターフェイスを提供しています。

- ENA インターフェイスはVPC の IP ネットワークをサポートするために必要なすべての従来の IP ネットワークとルーティング機能を提供します。詳細については「EC2 インスタンスで ENA による拡張ネットワーキングを有効にする」を参照してください。
- EFA (EFA と ENA の組み合わせ) インターフェイスはIP ネットワーク用の ENA デバイスと、低レイテンシーで高スループット通信用の EFA デバイスの両方を提供します。
- EFA のみのインターフェイスは EFA デバイス機能のみをサポートし、従来の IP ネットワーク用の ENA デバイスはありません。

次の表はENA、EFA、および EFA のみのネットワークインターフェイスの比較を提示しています。

	ENA	EFA (EFA と ENA の組み合わせ)	EFA のみ
IP ネットワーク機能をサポート	はい	はい	いいえ
IPv4 アドレスまたは IPv6 アドレスを割り当て可能	はい	はい	いいえ
インスタンスのプライマリネットワークインターフェイスとして使用可能	はい	はい	いいえ
インスタンスの ENI アタッチメント制限へのカウント	はい	はい	はい

	ENA	EFA (EFA と ENA の組み合わせ)	EFA のみ
インスタンスタイプのサポート	すべての Nitro ベースのインスタンスタイプでサポートに対応	サポートされるインスタンスタイプ	サポートされるインスタンスタイプ
EC2 API のパラメータ命名	`interface`	`efa`	`efa-only`
EC2 コンソールのフィールド命名	選択なし	EFA と ENA の組み合わせ	EFA のみ

サポートされているインターフェイスとライブラリ

EFA は以下のインターフェイスとライブラリをサポートしています。

- Open MPI 4.1 以降
- Intel MPI 2019 Update 5 以降
- NVIDIA Collective Communications Library (NCCL) 2.4.2 以降
- NVIDIA Inference Xfer Library (NIXL) 1.0.0 以降
- AWS Neuron SDK バージョン 2.3 以降

サポートされるインスタンスタイプ

次のインスタンスタイプはすべて EFA をサポートしています。また、この表は、インスタンスタイプの RDMA 読み取りおよび RDMA 書き込みのサポートを示しています。

Nitro v6 (EFA v4)

インスタンスタイプ	RDMA 読み取りサポート	RDMA 書き込みサポート
汎用		

インスタンスタイプ	RDMA 読み取りサポート	RDMA 書き込みサポート
m8a.48xlarge	Yes	Yes
m8a.metal-48xl	Yes	Yes
m8azn.24xlarge	Yes	Yes
m8azn.metal-24xl	Yes	Yes
m8gb.16xlarge	Yes	Yes
m8gb.24xlarge	Yes	Yes
m8gb.48xlarge	Yes	Yes
m8gb.metal-24xl	Yes	Yes
m8gb.metal-48xl	Yes	Yes
m8gn.16xlarge	Yes	Yes
m8gn.24xlarge	Yes	Yes
m8gn.48xlarge	Yes	Yes
m8gn.metal-24xl	Yes	Yes
m8gn.metal-48xl	Yes	Yes
m8i.48xlarge	Yes	Yes
m8i.96xlarge	Yes	Yes
m8i.metal-48xl	Yes	Yes
m8i.metal-96xl	Yes	Yes
m8id.48xlarge	Yes	Yes
m8id.96xlarge	Yes	Yes
m8id.metal-48xl	Yes	Yes

インスタンスタイプ	RDMA 読み取りサポート	RDMA 書き込みサポート
m8id.metal-96xl	Yes	Yes
m8in.48xlarge	Yes	Yes
m8in.96xlarge	Yes	Yes
m8in.metal-48xl	Yes	Yes
m8in.metal-96xl	Yes	Yes
m8idn.48xlarge	Yes	Yes
m8idn.96xlarge	Yes	Yes
m8idn.metal-48xl	Yes	Yes
m8idn.metal-96xl	Yes	Yes
m8ib.48xlarge	Yes	Yes
m8ib.96xlarge	Yes	Yes
m8ib.metal-48xl	Yes	Yes
m8ib.metal-96xl	Yes	Yes
m8idb.48xlarge	Yes	Yes
m8idb.96xlarge	Yes	Yes
m8idb.metal-48xl	Yes	Yes
m8idb.metal-96xl	Yes	Yes
m9g.48xlarge	Yes	Yes
m9g.metal-48xl	Yes	Yes
m9gd.48xlarge	Yes	Yes
m9gd.metal-48xl	Yes	Yes

インスタンスタイプ	RDMA 読み取りサポート	RDMA 書き込みサポート
コンピューティング最適化		
c8a.48xlarge	Yes	Yes
c8a.metal-48xl	Yes	Yes
c8gb.16xlarge	Yes	Yes
c8gb.24xlarge	Yes	Yes
c8gb.48xlarge	Yes	Yes
c8gb.metal-24xl	Yes	Yes
c8gb.metal-48xl	Yes	Yes
c8gn.16xlarge	Yes	Yes
c8gn.24xlarge	Yes	Yes
c8gn.48xlarge	Yes	Yes
c8gn.metal-24xl	Yes	Yes
c8gn.metal-48xl	Yes	Yes
c8i.48xlarge	Yes	Yes
c8i.96xlarge	Yes	Yes
c8i.metal-48xl	Yes	Yes
c8i.metal-96xl	Yes	Yes
c8id.48xlarge	Yes	Yes
c8id.96xlarge	Yes	Yes
c8id.metal-48xl	Yes	Yes
c8id.metal-96xl	Yes	Yes

インスタンスタイプ	RDMA 読み取りサポート	RDMA 書き込みサポート
c8in.48xlarge	Yes	Yes
c8in.96xlarge	Yes	Yes
c8in.metal-48xl	Yes	Yes
c8in.metal-96xl	Yes	Yes
c8ib.48xlarge	Yes	Yes
c8ib.96xlarge	Yes	Yes
c8ib.metal-48xl	Yes	Yes
c8ib.metal-96xl	Yes	Yes
c9g.48xlarge	Yes	Yes
c9g.metal-48xl	Yes	Yes
c9gd.48xlarge	Yes	Yes
c9gd.metal-48xl	Yes	Yes
メモリを最適化		
r8a.48xlarge	Yes	Yes
r8a.metal-48xl	Yes	Yes
r8gb.16xlarge	Yes	Yes
r8gb.24xlarge	Yes	Yes
r8gb.48xlarge	Yes	Yes
r8gb.metal-24xl	Yes	Yes
r8gb.metal-48xl	Yes	Yes
r8gn.16xlarge	Yes	Yes

インスタンスタイプ	RDMA 読み取りサポート	RDMA 書き込みサポート
r8gn.24xlarge	Yes	Yes
r8gn.48xlarge	Yes	Yes
r8gn.metal-24xl	Yes	Yes
r8gn.metal-48xl	Yes	Yes
r8i.48xlarge	Yes	Yes
r8i.96xlarge	Yes	Yes
r8i.metal-48xl	Yes	Yes
r8i.metal-96xl	Yes	Yes
r8id.48xlarge	Yes	Yes
r8id.96xlarge	Yes	Yes
r8id.metal-48xl	Yes	Yes
r8id.metal-96xl	Yes	Yes
r8in.48xlarge	Yes	Yes
r8in.96xlarge	Yes	Yes
r8in.metal-48xl	Yes	Yes
r8in.metal-96xl	Yes	Yes
r8idn.48xlarge	Yes	Yes
r8idn.96xlarge	Yes	Yes
r8idn.metal-48xl	Yes	Yes
r8idn.metal-96xl	Yes	Yes
r8ib.48xlarge	Yes	Yes

インスタンスタイプ	RDMA 読み取りサポート	RDMA 書き込みサポート
r8ib.96xlarge	Yes	Yes
r8ib.metal-48xl	Yes	Yes
r8ib.metal-96xl	Yes	Yes
r8idb.48xlarge	Yes	Yes
r8idb.96xlarge	Yes	Yes
r8idb.metal-48xl	Yes	Yes
r8idb.metal-96xl	Yes	Yes
x8aedz.24xlarge	Yes	Yes
x8aedz.metal-24xl	Yes	Yes
x8i.48xlarge	Yes	Yes
x8i.64xlarge	Yes	Yes
x8i.96xlarge	Yes	Yes
x8i.metal-48xl	Yes	Yes
x8i.metal-96xl	Yes	Yes
ストレージの最適化		
i8ge.48xlarge	Yes	Yes
i8ge.metal-48xl	Yes	Yes
高速コンピューティング		
g7.8xlarge	Yes	Yes
g7.12xlarge	Yes	Yes
g7.24xlarge	Yes	Yes

インスタンスタイプ	RDMA 読み取りサポート	RDMA 書き込みサポート
g7.48xlarge	Yes	Yes
g7e.8xlarge	Yes	Yes
g7e.12xlarge	Yes	Yes
g7e.24xlarge	Yes	Yes
g7e.48xlarge	Yes	Yes
p6-b200.48xlarge	Yes	Yes
p6-b300.48xlarge	Yes	Yes
ハイパフォーマンスコンピューティング		
hpc8a.96xlarge	Yes	Yes

Nitro v5 (EFA v3)

インスタンスタイプ	RDMA 読み取りサポート	RDMA 書き込みサポート
汎用		
m8g.24xlarge	Yes	Yes
m8g.48xlarge	Yes	Yes
m8g.metal-24xl	Yes	Yes
m8g.metal-48xl	Yes	Yes
m8gd.24xlarge	Yes	Yes
m8gd.48xlarge	Yes	Yes
m8gd.metal-24xl	Yes	Yes
m8gd.metal-48xl	Yes	Yes

インスタンスタイプ	RDMA 読み取りサポート	RDMA 書き込みサポート
コンピューティング最適化		
c7gn.16xlarge	Yes	No
c7gn.metal	Yes	No
c8g.24xlarge	Yes	Yes
c8g.48xlarge	Yes	Yes
c8g.metal-24xl	Yes	Yes
c8g.metal-48xl	Yes	Yes
c8gd.24xlarge	Yes	Yes
c8gd.48xlarge	Yes	Yes
c8gd.metal-24xl	Yes	Yes
c8gd.metal-48xl	Yes	Yes
メモリを最適化		
r8g.24xlarge	Yes	Yes
r8g.48xlarge	Yes	Yes
r8g.metal-24xl	Yes	Yes
r8g.metal-48xl	Yes	Yes
r8gd.24xlarge	Yes	Yes
r8gd.48xlarge	Yes	Yes
r8gd.metal-24xl	Yes	Yes
r8gd.metal-48xl	Yes	Yes
x8g.24xlarge	Yes	Yes

インスタンスタイプ	RDMA 読み取りサポート	RDMA 書き込みサポート
x8g.48xlarge	Yes	Yes
x8g.metal-24xl	Yes	Yes
x8g.metal-48xl	Yes	Yes
ストレージの最適化		
i7ie.48xlarge	Yes	Yes
i7ie.metal-48xl	Yes	Yes
i8g.48xlarge	Yes	Yes
i8g.metal-48xl	Yes	Yes
高速コンピューティング		
p5en.48xlarge	Yes	Yes
p6e-gb200.36xlarge	Yes	Yes
trn2.3xlarge	Yes	Yes
trn2.48xlarge	Yes	Yes
trn2u.48xlarge	Yes	Yes
ハイパフォーマンスコンピューティング		
hpc7g.4xlarge	Yes	No
hpc7g.8xlarge	Yes	No
hpc7g.16xlarge	Yes	No

Nitro v4 (EFA v2)

インスタンスタイプ	RDMA 読み取りサポート	RDMA 書き込みサポート
汎用		
m6a.48xlarge	Yes	Yes
m6a.metal	Yes	Yes
m6i.32xlarge	Yes	Yes
m6i.metal	Yes	Yes
m6id.32xlarge	Yes	Yes
m6id.metal	Yes	Yes
m6idn.32xlarge	Yes	Yes
m6idn.metal	Yes	Yes
m6in.32xlarge	Yes	Yes
m6in.metal	Yes	Yes
m7a.48xlarge	Yes	Yes
m7a.metal-48xl	Yes	Yes
m7g.16xlarge	Yes	Yes
m7g.metal	Yes	Yes
m7gd.16xlarge	Yes	Yes
m7gd.metal	Yes	Yes
m7i.48xlarge	Yes	Yes
m7i.metal-48xl	Yes	Yes
コンピューティング最適化		

インスタンスタイプ	RDMA 読み取りサポート	RDMA 書き込みサポート
c6a.48xlarge	Yes	Yes
c6a.metal	Yes	Yes
c6gn.16xlarge	Yes	Yes
c6i.32xlarge	Yes	Yes
c6i.metal	Yes	Yes
c6id.32xlarge	Yes	Yes
c6id.metal	Yes	Yes
c6in.32xlarge	Yes	Yes
c6in.metal	Yes	Yes
c7a.48xlarge	Yes	Yes
c7a.metal-48xl	Yes	Yes
c7g.16xlarge	Yes	Yes
c7g.metal	Yes	Yes
c7gd.16xlarge	Yes	Yes
c7gd.metal	Yes	Yes
c7i.48xlarge	Yes	Yes
c7i.metal-48xl	Yes	Yes
メモリを最適化		
r6a.48xlarge	Yes	Yes
r6a.metal	Yes	Yes
r6i.32xlarge	Yes	Yes

インスタンスタイプ	RDMA 読み取りサポート	RDMA 書き込みサポート
r6i.metal	Yes	Yes
r6id.32xlarge	Yes	Yes
r6id.metal	Yes	Yes
r6idn.32xlarge	Yes	Yes
r6idn.metal	Yes	Yes
r6in.32xlarge	Yes	Yes
r6in.metal	Yes	Yes
r7a.48xlarge	Yes	Yes
r7a.metal-48xl	Yes	Yes
r7g.16xlarge	Yes	Yes
r7g.metal	Yes	Yes
r7gd.16xlarge	Yes	Yes
r7gd.metal	Yes	Yes
r7i.48xlarge	Yes	Yes
r7i.metal-48xl	Yes	Yes
r7iz.32xlarge	Yes	Yes
r7iz.metal-32xl	Yes	Yes
u7i-6tb.112xlarge	Yes	Yes
u7i-8tb.112xlarge	Yes	Yes
u7i-12tb.224xlarge	Yes	Yes
u7in-16tb.224xlarge	Yes	Yes

インスタンスタイプ	RDMA 読み取りサポート	RDMA 書き込みサポート
u7in-24tb.224xlarge	Yes	Yes
u7in-32tb.224xlarge	Yes	Yes
u7inh-32tb.480xlarge	Yes	Yes
x2idn.32xlarge	Yes	Yes
x2idn.metal	Yes	Yes
x2iedn.32xlarge	Yes	Yes
x2iedn.metal	Yes	Yes
ストレージの最適化		
i4g.16xlarge	Yes	Yes
i4i.32xlarge	Yes	Yes
i4i.metal	Yes	Yes
i7i.24xlarge	Yes	Yes
i7i.48xlarge	Yes	Yes
i7i.metal-48xl	Yes	Yes
im4gn.16xlarge	Yes	Yes
高速コンピューティング		
f2.48xlarge	Yes	Yes
g6.8xlarge	Yes	Yes
g6.12xlarge	Yes	Yes
g6.16xlarge	Yes	Yes
g6.24xlarge	Yes	Yes

インスタンスタイプ	RDMA 読み取りサポート	RDMA 書き込みサポート
g6.48xlarge	Yes	Yes
g6e.8xlarge	Yes	Yes
g6e.12xlarge	Yes	Yes
g6e.16xlarge	Yes	Yes
g6e.24xlarge	Yes	Yes
g6e.48xlarge	Yes	Yes
gr6.8xlarge	Yes	Yes
p5.4xlarge	Yes	Yes
p5.48xlarge	Yes	Yes
p5e.48xlarge	Yes	Yes
trn1.32xlarge	Yes	Yes
trn1n.32xlarge	Yes	Yes
ハイパフォーマンスコンピューティング		
hpc6a.48xlarge	Yes	Yes
hpc6id.32xlarge	Yes	Yes
hpc7a.12xlarge	Yes	Yes
hpc7a.24xlarge	Yes	Yes
hpc7a.48xlarge	Yes	Yes
hpc7a.96xlarge	Yes	Yes

Nitro v3 (EFA v1)

インスタンスタイプ	RDMA 読み取りサポート	RDMA 書き込みサポート
汎用		
m5dn.24xlarge	No	No
m5dn.metal	No	No
m5n.24xlarge	No	No
m5n.metal	No	No
m5zn.12xlarge	No	No
m5zn.metal	No	No
コンピューティング最適化		
c5n.9xlarge	No	No
c5n.18xlarge	No	No
c5n.metal	No	No
メモリを最適化		
r5dn.24xlarge	No	No
r5dn.metal	No	No
r5n.24xlarge	No	No
r5n.metal	No	No
x2iezn.12xlarge	No	No
x2iezn.metal	No	No
ストレージの最適化		
i3en.12xlarge	No	No

インスタンスタイプ	RDMA 読み取りサポート	RDMA 書き込みサポート
i3en.24xlarge	No	No
i3en.metal	No	No
高速コンピューティング		
dl2q.24xlarge	No	No
g4dn.8xlarge	No	No
g4dn.12xlarge	No	No
g4dn.16xlarge	No	No
g4dn.metal	No	No
g5.8xlarge	No	No
g5.12xlarge	No	No
g5.16xlarge	No	No
g5.24xlarge	No	No
g5.48xlarge	No	No
inf1.24xlarge	No	No
p3dn.24xlarge	No	No
p4d.24xlarge	Yes	No
p4de.24xlarge	Yes	No
vt1.24xlarge	No	No
前の世代		
p3dn.24xlarge	No	No

特定のリージョンで EFA をサポートする利用可能なインスタンスタイプを確認するには

利用可能なインスタンスタイプはリージョンごとに異なります。リージョンで EFA をサポートする使用可能なインスタンスタイプを確認するには--region パラメーターを指定して describe-instance-types コマンドを使用します。結果を EFA をサポートするインスタンスタイプにスコープする --filters パラメーターと、出力を InstanceType の値にスコープする --query パラメーターを含めます。

```
aws ec2 describe-instance-types \
    --region us-east-1  \
    --filters Name=network-info.efa-supported,Values=true \
    --query "InstanceTypes[*].[InstanceType]"  \
    --output text | sort
```

サポートされるオペレーティングシステム

オペレーティングシステムのサポートはプロセッサタイプによって異なります。次の表はサポートされているオペレーティングシステムを示しています。

オペレーティングシステム	Intel/AMD (**x86_64**) インスタンスタイプ	AWS Graviton (**arm64**) インスタンスタイプ
Amazon Linux 2023	✓	✓
RHEL 8、9、および 10	✓	✓
Debian 11、12、13	✓	✓
Rocky Linux 8 および 9	✓	✓
Ubuntu 22.04, 24.04, and 26.04	✓	✓

オペレーティングシステム	Intel/AMD (**x86_64**) インスタンスタイプ	AWS Graviton (**arm64**) インスタンスタイプ
SUSE Linux Enterprise 15 SP2 以降	✓	✓

> **Note**
>
> リストされているオペレーティングシステムの一部は、Intel MPI でサポートされていない場合があります。Intel MPI を使用している場合は、Intel MPI ドキュメントを参照してオペレーティングシステムのサポートを確認してください。

EFA の制限事項

EFA には次の制限があります。

- RDMA 書き込みは、すべてのインスタンスタイプでサポートされているわけではありません。詳細については「サポートされるインスタンスタイプ」を参照してください。
- P4d/P4de/DL1 インスタンスと他のインスタンスタイプ間の EFA トラフィック [1] は現在サポートされていません。
- 複数のネットワークカードをサポートするインスタンスタイプはネットワークカードごとに 1 つの EFA で設定できます。その他のサポートされているインスタンスタイプはすべて、インスタンスごとに 1 つの EFA のみをサポートしています。
- EFA がアタッチされている場合、`c7g.16xlarge`、`m7g.16xlarge`、および `r7g.16xlarge` のハードウェア専有インスタンスおよび専有ホストはサポートされません。
- EFA トラフィック [1] はアベイラビリティーゾーンまたは VPC をまたがることはできません。これはEFA インターフェイスの ENA デバイスからの通常の IP トラフィックには適用されません。
- EFA トラフィック [1] はルーティングできません。EFA インターフェイスの ENA デバイスからの通常の IP トラフィックはルーティング可能です。
- EFA は AWS Outposts ではサポートされていません。
- EFA (EFA と ENA の組み合わせ) インターフェイスの EFA デバイスは AWS Cloud Digital Interface Software Development Kit (AWS CDI SDK) ベースのアプリケーションでのみ Windows インスタンスでサポートされています。EFA (EFA と ENA の組み合わせ) インターフェイスを

Windows インスタンスに非 CDI SDK ベースのアプリケーション用として接続すると、EFA デバイスの機能を追加せずに、ENA インターフェースとして機能します。EFA のみのインターフェイスはWindows または Linux で AWS CDI ベースのアプリケーションではサポートされていません。詳細については「AWS Cloud Digital Interface Software Development Kit (AWS CDI SDK) ユーザーガイド」を参照してください。

[1]EFA トラフィックとは、EFA (EFA および ENA の組み合わせ) または EFA 専用のインターフェイスのいずれかの EFA デバイスを介して配信されるトラフィックを指します。

EFA 価格設定

EFA はオプションの Amazon EC2 ネットワーキング機能として利用でき、サポートされているどのインスタンスでも追加費用なしで有効にできます。

Amazon EC2 の HPC ワークロードに対して EFA と MPI の使用を開始する

本チュートリアルはEFA と HPC ワークロードの MPI 対応インスタンスクラスターの起動に役立ちます。

> **Note**
>
> `u7i-12tb.224xlarge`、`u7in-16tb.224xlarge`、`u7in-24tb.224xlarge`、および `u7in-32tb.224xlarge` インスタンスはOpen MPI では最大 128 の並列 MPI プロセスを実行でき、Intel MPI では最大 256 の並列 MPI プロセスを実行できます。

タスク

- ステップ 1: EFA 対応のセキュリティグループを準備する
- ステップ 2: 一時インスタンスを作成する
- ステップ 3: EFA ソフトウェアをインストールする
- ステップ 4: (オプション) Open MPI 5 を有効にする
- ステップ 5: (オプション) インテル MPI をインストールする
- ステップ 6: ptrace 保護を無効にする
- ステップ 7. インストールを確認する
- ステップ 8: HPC アプリケーションをインストールする
- ステップ 9: EFA 対応の AMI を作成する

- ステップ 10: クラスタープレイスメントグループで EFA 対応のインスタンスを作成する
- ステップ 11: 一時インスタンスを終了する
- ステップ 12: パスワードレス SSH を有効にする

ステップ 1: EFA 対応のセキュリティグループを準備する

EFA にはセキュリティグループ自体とのインバウンドおよびアウトバウンドのトラフィックをすべて許可するセキュリティグループが必要です。以下の手順ではセキュリティグループを作成します。このセキュリティグループではセキュリティグループ自体とのすべてのインバウンドおよびアウトバウンドのトラフィックと、SSH 接続用の任意の IPv4 アドレスからのインバウンド SSH トラフィックを許可します。

> **Important**
>
> このセキュリティグループはテストのみを目的としています。本番環境ではコンピュータの IP アドレスやローカルネットワークの IP アドレスの範囲など、接続元の IP アドレスからのトラフィックのみを許可するインバウンド SSH ルールを作成することをお勧めします。

その他のシナリオについてはさまざまなユースケースのセキュリティグループのルールを参照してください。

EFA 対応のセキュリティグループを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Security Groups] (セキュリティグループ) を選択して、[Create security group] (セキュリティグループの作成) を選択してください。
3. [Create security group] (セキュリティグループの作成) ウィンドウで、以下を行います。
   a. [セキュリティグループ名] に、`EFA-enabled security group` のような、分かりやすいセキュリティグループ名を入力してください。
   b. (オプション) [説明] に、セキュリティグループの簡単な説明を入力してください。
   c. [VPC] で、EFA 対応のインスタンスを起動する VPC を選択してください。
   d. [セキュリティグループの作成] を選択してください。
4. 作成したセキュリティグループを選択し、[Details] (詳細) タブで [Security group ID] (セキュリティグループ ID) をコピーします。

5. セキュリティグループが選択された状態で、[Actions] (アクション)、[Edit inbound rules] (インバウンドルールの編集) の順に選択し、次の手順を実行します。

   a. [Add rule] を選択してください。

   b. [Type] で、[All traffic] を選択してください。

   c. [Source type] (送信元タイプ) で、[Custom] (カスタム) を選択し、コピーしたセキュリティグループ ID をフィールドに貼り付けます。

   d. [ルールを追加] を選択してください。

   e. [タイプ] で SSH] を選択してください。

   f. [Source type] (ソースタイプ) で、[Anywhere-IPv4] を選択してください。

   g. [Save Rules] (ルールの保存) を選択してください。

6. セキュリティグループが選択された状態で、[Actions] (アクション)、[Edit outbound rules] (アウトバウンドルールの編集) の順に選択し、次の手順を実行します。

   a. [Add rule] を選択してください。

   b. [Type] で、[All traffic] を選択してください。

   c. [Destination type] (送信先タイプ) で、[Custom] (カスタム) を選択し、コピーしたセキュリティグループ ID をフィールドに貼り付けます。

   d. [Save Rules] (ルールの保存) を選択してください。

ステップ 2: 一時インスタンスを作成する

EFA ソフトウェアコンポーネントのインストールおよび設定に使用する一時インスタンスを起動します。このインスタンスを使用して、EFA 対応のインスタンスを起動する EFA 対応の AMI を作成します。

一時インスタンスを起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択し、[Launch Instances] (インスタンスの起動) を選択して、新しいインスタンス起動ウィザードを開きます。
3. (オプション) [Name and tags] (名前とタグ) セクションで、`EFA-instance` などのインスタンス名を指定します。指定した名前はリソースタグとしてインスタンスに割り当てられます (Name=*`EFA-instance`*)。

4. [Application and OS Images] (アプリケーションと OS イメージ) セクションで、サポートされるオペレーティングシステムを選択してください。
5. [Instance type] (インスタンスタイプ) セクションで、サポートされているインスタンスタイプを選択してください。
6. [Key pair] (キーペア) セクションで、インスタンスに使用するキーペアを選択してください。
7. [Network settings] (ネットワーク設定) セクションで、[Edit] (編集) を選択し、次の操作を行います。

   a. [サブネット] で、インスタンスを起動するサブネットを選択してください。

   > **Important**
   >
   > サブネットを選択する必要があります。サブネットを選択しない場合、EFA のインスタンスを有効にすることはできません。

   b. [Firewall (security groups)] (ファイアウォール (セキュリティグループ)) の場合、[Select existing security group] (既存のセキュリティグループの選択) を選択し、前のステップで作成したセキュリティグループを選択してください。

   c. [高度な設定] セクションを展開します。

   ネットワークインターフェイス 1 では[ネットワークカードインデックス = 0]、[デバイスインデックス = 0]、[インターフェイスタイプ = EFA と ENA の組み合わせ] を選択してください。

   (オプション) `p4d.24xlarge` または `p5.48xlarge` などのマルチカードインスタンスタイプを、必要な追加のネットワークインターフェイスごとに使用する場合は[ネットワークインターフェイスの追加] を選択し、ネットワークカードインデックスの場合は次の未使用インデックスを選択してから、[デバイスインデックス = 1]、[インターフェイスタイプ = EFA と ENA の組み合わせ] または [EFA のみ] を選択してください。
8. [Storage] (ストレージ) セクションで、必要に応じてボリュームを設定します。
9. 右側の [合計mary] (サマリー) パネルで、[Launch instance] (インスタンスの起動) を選択してください。

> **Note**
>
> 既にアカウントのデフォルトとして IMDSv2 を設定していない限り、一時インスタンスとステップ 9 で作成する AMI に IMDSv2 の使用を必須化することを検討してください。IMDSv2 設定ステップの詳細については「新規インスタンスのインスタンスメタデータオプションの設定」を参照してください。

ステップ 3: EFA ソフトウェアをインストールする

一時インスタンスで EFA をサポートするために必要な EFA 対応のカーネル、EFA ドライバー、Libfabric、および Open MPI スタックをインストールします。

このステップはEFA で Open MPI、Intel MPI、または Open MPI と Intel MPI のどれを使用するかによって異なります。

> **Note**
>
> 一部のオペレーティングシステムは、Intel MPI でサポートされていない場合があります。Intel MPI を使用している場合は、Intel MPI ドキュメントを参照してオペレーティングシステムのサポートを確認してください。

EFA ソフトウェアをインストールするには

1. 起動したインスタンスに接続します。詳細については「SSH を使用した Linux インスタンスへの接続」を参照してください。
2. すべてのソフトウェアパッケージが最新の状態であることを確認するため、インスタンスでソフトウェアの更新を実行します。このプロセスには数分かかることがあります。

   - Amazon Linux 2023、RHEL 8/9、Rocky Linux 8/9

     ```
     $ sudo yum update -y
     ```

   - Ubuntu と Debian

     ```
     $ sudo apt-get update && sudo apt-get upgrade -y
     ```

   - SUSE Linux Enterprise

```
$ sudo zypper update -y
```

3. インスタンスを再起動して、そのインスタンスに再接続します。
4. EFA ソフトウェアのインストールファイルをダウンロードします。ソフトウェアのインストールファイルは圧縮された tar (`.tar.gz`) ファイルにパッケージ化されています。次のコマンドを使用して、安定している最新バージョンをダウンロードします。

   前述のコマンドのバージョン番号を `latest` に置き換えることで最新バージョンを取得することもできます。

   ```
   $ curl -O https://efa-installer.amazonaws.com/aws-efa-installer-1.49.0.tar.gz
   ```

5. (オプション) EFA tarball (`.tar.gz`) ファイルの認証と完全性を検証します。

   ソフトウェアパブリッシャーの ID を検証し、発行後にファイルの改変や破損がないことを確認するために、これを行うことをお勧めします。tar ファイルを検証しない場合はこの手順をスキップします。

   > **Note**
   >
   > 代わりに、MD5 または SHA256 チェックサムを使用して tar ファイルを検証する場合はチェックサムを使用した EFA インストーラの検証を参照してください。

   a. パブリック GPG キーをダウンロードして、キーリングにインポートします。

      ```
      $ wget https://efa-installer.amazonaws.com/aws-efa-installer.key && gpg --
      import aws-efa-installer.key
      ```

      コマンドはキーの値を返します。次の手順で必要になるため、キーの値を書きとめておきます。

   b. GPG キーのフィンガープリントを検証します。次のコマンドを実行し、前のステップで作成したキーの値を指定します。

      ```
      $ gpg --fingerprint key_value
      ```

コマンドは4E90 91BC BB97 A96B 26B1 5E59 A054 80B1 DD2D 3CCC と同じフィンガープリントを返します。フィンガープリントが一致しない場合はEFA インストールスクリプトを実行せず、サポート にお問い合わせください。

c. 署名ファイルをダウンロードし、EFA tar ファイルの署名を検証します。

```
$ wget https://efa-installer.amazonaws.com/aws-efa-installer-1.49.0.tar.gz.sig
 && gpg --verify ./aws-efa-installer-1.49.0.tar.gz.sig
```

出力例を次に示します。

```
gpg: Signature made Wed 29 Jul 2020 12:50:13 AM UTC using RSA key ID DD2D3CCC
gpg: Good signature from "Amazon EC2 EFA <ec2-efa-maintainers@amazon.com>"
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: 4E90 91BC BB97 A96B 26B1  5E59 A054 80B1 DD2D 3CCC
```

結果に Good signature が含まれ、フィンガープリントが前のステップで返されたフィンガープリントと一致する場合は次のステップに進みます。そうでない場合はEFA インストールスクリプトを実行せず、サポート にお問い合わせください。

6. 圧縮された .tar.gz ファイルからファイルを展開し、展開されたディレクトリに移動します。

```
$ tar -xf aws-efa-installer-1.49.0.tar.gz && cd aws-efa-installer
```

7. (オプション) インストール中に個々のパッケージ署名を検証します。

EFA インストーラ 1.48.0 以降では、インストーラには GPG 署名の個々の RPM および DEB パッケージが含まれています。インストール中に個々のパッケージの信頼性と整合性を検証するには、--check-signatures フラグを使用します。このフラグを有効にすると、まず、インストーラはすべてのパッケージ署名を検証し、すべてのパッケージが検証に合格した場合にのみインストールを続行します。パッケージが検証に失敗すると、インストーラは何もインストールせずにすぐに終了します。

a. GPG 公開鍵をダウンロードします。

```
$ wget https://efa-installer.amazonaws.com/aws-efa-installer.key
```

b. キーパスをエクスポートします。次に、次のステップで、インストールコマンドに `--check-signatures` を追加し、sudo の代わりに `sudo -E` を使用して環境変数を保持します。

```
$ export EFA_INSTALLER_KEY=$(pwd)/aws-efa-installer.key
```

RPM ベースのシステム (Amazon Linux、RHEL、Rocky Linux、SUSE) では、インストーラは `rpm --checksig` を使用して各 RPM を検証します。DEB ベースのシステム (Ubuntu、Debian) では、インストーラは GPG 署名検証を使用して各 DEB を検証します。

パッケージの検証に失敗すると、インストールは直ちに中止され、壊れたパッケージや悪意のあるパッケージからシステムを保護します。

> **Note**
>
> `--check-signatures` フラグはオプションです。これがないと、インストーラは個々の署名検証を実行しません。

8. EFA ソフトウェアをインストールします。使用するユースケースに応じて、以下のいずれかを実行します。

> **Note**
>
> 前のオプションステップを完了してパッケージ署名の検証をセットアップした場合は、インストールコマンドに `--check-signatures` を追加して、sudo の代わりに `sudo -E` を使用します。例: `sudo -E ./efa_installer.sh -y --check-signatures`。

> **Note**
>
> EFA は SUSE Linux での NVIDIA GPUDirect をサポートしていません。SUSE Linux を使用している場合はさらに kmod のインストールを防止する `--skip-kmod` オプションを指定する必要があります。デフォルトではSUSE Linux はツリー外のカーネルモジュールを許可しません。

Open MPI and Intel MPI

EFA で Open MPI と Intel MPI を使用する場合はEFA ソフトウェアと共に Libfabric と Open MPI をインストールする必要があります。また、「ステップ 5: (オプション) インテル MPI をインストールする」を完了する必要があります。

Libfabric および Open MPI と共に EFA ソフトウェアをインストールするには次のコマンドを実行します。

> **Note**
>
> EFA 1.30.0 からは　Open MPI 4.1 と Open MPI 5 の両方がデフォルトでインストールされます。任意で、インストールする Open MPI のバージョンを指定できます。Open MPI 4.1 のみをインストールするには `--mpi=openmpi4` を含めてください。Open MPI 5 のみをインストールするには`--mpi=openmpi5` を含めてください。両方をインストールする場合は`--mpi` オプションを省略してください。

```
$ sudo ./efa_installer.sh -y
```

Libfabric は `/opt/amazon/efa` にインストールされます。Open MPI 4.1 は `/opt/amazon/openmpi` にインストールされます。Open MPI 5 は `/opt/amazon/openmpi5` にインストールされます。

Open MPI only

EFA で Open MPI のみを使用する場合はEFA ソフトウェアと共に Libfabric と Open MPI をインストールする必要があります。また、「ステップ 5: (オプション) インテル MPI をインストールする」はスキップできます。Libfabric および Open MPI と共に EFA ソフトウェアをインストールするには次のコマンドを実行します。

> **Note**
>
> EFA 1.30.0 からは　Open MPI 4.1 と Open MPI 5 の両方がデフォルトでインストールされます。任意で、インストールする Open MPI のバージョンを指定できます。Open MPI 4.1 のみをインストールするには `--mpi=openmpi4` を含めてくだ

さい。Open MPI 5 のみをインストールするには--mpi=openmpi5 を含めてください。両方をインストールする場合は--mpi オプションを省略してください。

```
$ sudo ./efa_installer.sh -y
```

Libfabric は /opt/amazon/efa にインストールされます。Open MPI 4.1 は /opt/amazon/openmpi にインストールされます。Open MPI 5 は /opt/amazon/openmpi5 にインストールされます。

Intel MPI only

EFA で Intel MPI のみを使用する場合はLibfabric および Open MPI を使用せずに EFA ソフトウェアをインストールできます。この場合、Intel MPI は埋め込まれている Libfabric を使用します。これを選択した場合は「ステップ 5: (オプション) インテル MPI をインストールする」を完了する必要があります。

Libfabric および Open MPI を使用せずに EFA ソフトウェアをインストールするには次のコマンドを実行します。

```
$ sudo ./efa_installer.sh -y --minimal
```

9. EFA インストーラーでインスタンスの再起動を求めるメッセージが表示された場合は再起動してからインスタンスに再接続します。それ以外の場合はインスタンスからログアウトし、再度ログインしてインストールを完了します。
10. 非圧縮 tarball と tarball 自体を削除してください。そうしないと、これらが作成する EFA 対応 AMI に含まれてしまい、サイズが増加します。

ステップ 4: (オプション) Open MPI 5 を有効にする

> **Note**
>
> このステップはOpen MPI 5 を使用する場合にのみ実行します。

EFA 1.30.0 からは　Open MPI 4.1 と Open MPI 5 の両方がデフォルトでインストールされます。あるいは、Open MPI 4.1 または Open MPI 5 のみをインストールするように選択することもできます。

「ステップ 3: EFA ソフトウェアをインストールする」で Open MPI 5 のインストールを選択し、これを使用する場合は次の手順を実行して有効にする必要があります。

Open MPI 5 を有効にするには

1. Open MPI 5 を PATH 環境変数に追加します。

```
$ module load openmpi5
```

2. Open MPI 5 の使用が有効になっていることを確認します。

```
$ which mpicc
```

このコマンドは Open MPI 5 のインストールディレクトリ - `/opt/amazon/openmpi5` を返すはずです。

3. (オプション) インスタンスが起動するたびに Open MPI 5 が PATH 環境変数に追加されるようにするには次の操作を行います。

bash shell

`module load openmpi5` を `/home/username/.bashrc` と `/home/username/.bash_profile` に追加します。

csh and tcsh shells

`/home/username/.cshrc` に `module load openmpi5` を追加します。

Open MPI 5 を PATH 環境変数から削除する必要がある場合は次のコマンドを実行して、シェルスタートアップスクリプトからそのコマンドを削除します。

```
$ module unload openmpi5
```

ステップ 5: (オプション) インテル MPI をインストールする

> **Important**
>
> このステップはIntel MPI を使用する場合にのみ実行します。Open MPI のみを使用する場合はこのステップをスキップしてください。

Intel MPI を使用するには追加のインストールと環境変数設定が必要です。

前提条件

以下のステップはsudo アクセス許可を持つユーザーが実行してください。

Intel MPI をインストールするには

1. インテル MPI のインストールスクリプトをダウンロードするには次の手順を実行します。
   a. インテルのウェブサイトにアクセスします。
   b. ウェブページの「Intel MPI Library」(インテル MPI ライブラリ) セクションで、Intel MPI Library for Linux Offline インストーラのリンクを選択してください。
2. 前のステップでダウンロードしたインストールスクリプトを実行します。

```
$ sudo bash installation_script_name.sh
```

3. インストーラで、[Accept & install] (承諾してインストール) を選択してください。
4. インテル Improvement Program を読み、適切なオプションを選択してから、[Begin Installation] (インストールを開始) を選択してください。
5. インストールが完了したら、[Close] を選択してください。
6. デフォルトではインテル MPI は埋め込まれている (内部) Libfabric を使用します。代わりに、EFA インストーラに同梱されている Libfabric を使用するようにインテル MPI を設定できます。通常、EFA インストーラにはインテル MPI よりも新しいバージョンの Libfabric が同梱されています。場合によってはEFA インストーラに同梱されている Libfabric はインテル MPI よりもパフォーマンスが高い場合があります。EFA インストーラに同梱されている Libfabric を使用するようにインテル MPI を設定するにはシェルに応じて次のいずれかを実行します。

   bash shells

   次のステートメントを /home/*username*/.bashrc と /home/*username*/.bash_profile に追加します。

   ```
   export I_MPI_OFI_LIBRARY_INTERNAL=0
   ```

   csh and tcsh shells

   次のステートメントを /home/*username*/.cshrc に追加します。

```
setenv I_MPI_OFI_LIBRARY_INTERNAL 0
```

7. 次のソースコマンドをシェルスクリプトに追加して、インストールディレクトリから vars.sh スクリプトを読み込み、インスタンスが開始するたびにコンパイラ環境をセットアップします。使用するシェルに応じて、以下のいずれかを実行します。

   bash shells

   次のステートメントを /home/*username*/.bashrc と /home/*username*/.bash_profile に追加します。

   ```
   source /opt/intel/oneapi/mpi/latest/env/vars.sh
   ```

   csh and tcsh shells

   次のステートメントを /home/*username*/.cshrc に追加します。

   ```
   source /opt/intel/oneapi/mpi/latest/env/vars.csh
   ```

8. デフォルトでは設定ミスにより EFA が使用できない場合、インテル MPI はデフォルトで TCP/IP ネットワークスタックを使用するため、アプリケーションのパフォーマンスが低下する可能性があります。I_MPI_OFI_PROVIDER を efa に設定することでこれを防ぐことができます。これにより、EFA が利用できない場合、インテル MPI は次のエラーで失敗します。

   ```
   Abort (XXXXXX) on node 0 (rank 0 in comm 0): Fatal error in PMPI_Init: OtherMPI
    error,
   MPIR_Init_thread (XXX).......:
   MPID_Init (XXXX).............:
   MPIDI_OFI_mpi_init_hook (XXXX):
   open_fabric (XXXX)...........:
   find_provider (XXXX).........:
   OFI fi_getinfo() failed (ofi_init.c:2684:find_provider:
   ```

   使用するシェルに応じて、以下のいずれかを実行します。

   bash shells

   次のステートメントを /home/*username*/.bashrc と /home/*username*/.bash_profile に追加します。

```
export I_MPI_OFI_PROVIDER=efa
```

csh and tcsh shells

次のステートメントを /home/*username*/.cshrc に追加します。

```
setenv I_MPI_OFI_PROVIDER efa
```

9. デフォルトではインテル MPI はデバッグ情報を出力しません。さまざまな詳細レベルを指定して、デバッグ情報を制御できます。可能な値は次のとおりです (提供される詳細の量の順): 0 (デフォルト)、1、2、3、4、5。レベル 1 以上は `libfabric version` と `libfabric provider` を出力します。インテル MPI が内部 Libfabric を使用しているか、または EFA インストーラに同梱されている Libfabric を使用しているかを確認するために `libfabric version` を使用します。内部 Libfabric を使用している場合、バージョンのサフィックスは `impi` です。インテル MPI が EFA または TCP/IP ネットワークを使用しているかどうかを確認するために `libfabric provider` を使用します。EFA を使用している場合、値は `efa` です。TCP/IP を使用している場合、値は `tcp;ofi_rxm` です。

   デバッグ情報を有効にするには使用するシェルに応じて、次のいずれかを実行します。

   bash shells

   次のステートメントを /home/*username*/.bashrc と /home/*username*/.bash_profile に追加します。

   ```
   export I_MPI_DEBUG=value
   ```

   csh and tcsh shells

   次のステートメントを /home/*username*/.cshrc に追加します。

   ```
   setenv I_MPI_DEBUG value
   ```

10. デフォルトではインテル MPI はノード内通信にはオペレーティングシステムの共有メモリ (shm) を使用し、ノード間通信にのみ Libfabric (`ofi`) を使用します。通常、この設定は最高のパフォーマンスを提供します。ただし、場合によってはインテル MPI shm ファブリックによって特定のアプリケーションが無期限にハングすることがあります。

この問題を解決するために、インテル MPI がノード内通信とノード間通信の両方に Libfabric を使用するように強制できます。これを実行するには使用するシェルに応じて、次のいずれかを実行します。

bash shells

次のステートメントを /home/*username*/.bashrc と /home/*username*/.bash_profile に追加します。

```
export I_MPI_FABRICS=ofi
```

csh and tcsh shells

次のステートメントを /home/*username*/.cshrc に追加します。

```
setenv I_MPI_FABRICS ofi
```

> **Note**
>
> EFA Libfabric プロバイダーはオペレーティングシステムの共有メモリをノード内通信に使用します。これはI_MPI_FABRICS を ofi に設定すると、デフォルトの shm:ofi 設定と同様のパフォーマンスが得られることを意味します。

11. インスタンスからログアウトしてからログインし直します。

Intel MPI が不要になった場合はシェル起動スクリプトから環境変数を削除してください。

ステップ 6: ptrace 保護を無効にする

HPC アプリケーションのパフォーマンスを向上させるために、Libfabric はプロセスが同じインスタンスで実行されている場合、プロセス間通信にインスタンスのローカルメモリを使用します。

共有メモリ機能ではptrace 保護ではサポートされない Cross-Memory Attach (CMA) が使用されます。Ubuntu など、ptrace 保護がデフォルトで有効になっている Linux ディストリビューションを使用している場合は無効にする必要があります。Linux ディストリビューションで ptrace 保護がデフォルトで有効になっていない場合はこのステップをスキップします。

ptrace 保護を無効にするには

次のいずれかを行ってください。

- テストのために ptrace 保護を一時的に無効にするには次のコマンドを実行します。

```
$ sudo sysctl -w kernel.yama.ptrace_scope=0
```

- ptrace 保護を完全に無効にするには`kernel.yama.ptrace_scope = 0` を `/etc/sysctl.d/10-ptrace.conf` に追加してインスタンスを再起動します。

ステップ 7. インストールを確認する

インストールが正常に完了したことを確認するには

1. MPI が正常にインストールされていることを確認するには次のコマンドを実行します。

   ```
   $ which mpicc
   ```

   - Open MPI の場合、返されるパスには `/opt/amazon/` が含まれている必要があります。
   - Intel MPI の場合、返されるパスには `/opt/intel/` が含まれている必要があります。期待どおりの出力が得られない場合はIntel MPI `vars.sh` スクリプトをソースとしていることを確認してください。

2. EFA ソフトウェアコンポーネントと Libfabric が正常にインストールされたことを確認するには以下のコマンドを実行します。

   ```
   $ fi_info -p efa -t FI_EP_RDM
   ```

   コマンドによって、Libfabric の EFA インターフェイスに関する情報が返ります。以下の例はコマンド出力を示しています。

   ```
   provider: efa
       fabric: EFA-fe80::94:3dff:fe89:1b70
       domain: efa_0-rdm
       version: 2.0
       type: FI_EP_RDM
       protocol: FI_PROTO_EFA
   ```

ステップ 8: HPC アプリケーションをインストールする

HPC アプリケーションを一時インスタンスにインストールします。インストール手順は特定の HPC アプリケーションによって異なります。詳細については、「Amazon Linux 2023 ユーザーガイド」の「OS 更新の管理」を参照してください。

> **Note**
>
> インストール手順については HPC アプリケーションのドキュメントを参照してください。

ステップ 9: EFA 対応の AMI を作成する

必要なソフトウェアコンポーネントのインストール後、EFA 対応のインスタンスの起動に再利用できる AMI を作成します。

一時インスタンスから AMI を作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. 作成した一時インスタンスを選択し、[アクション]、[イメージ]、[イメージの作成] の順に選択してください。
4. [イメージの作成] で、次を行います。
   a. [イメージ名] に、の分かりやすい AMI 名を入力してください。
   b. (オプション) [イメージの説明] に、AMI の簡単な説明を入力してください。
   c. [イメージを作成] を選択してください。
5. ナビゲーションペインで [AMI] を選択してください。
6. リストで作成した AMI を探します。ステータスが `pending` から `available` に変わるまで待ってから、次のステップに進みます。

ステップ 10: クラスタープレイスメントグループで EFA 対応のインスタンスを作成する

ステップ 7 で作成した EFA 対応の AMI と、ステップ 1 で作成した EFA 対応のセキュリティグループを使用して、EFA 対応のインスタンスをクラスタープレイスメントグループ内で起動します。

> **Note**
>
> - EFA 対応のインスタンスをクラスタープレイスメントグループに起動することは絶対的な要件ではありません。ただし、EFA 対応のインスタンスは 1 つのアベイラビリティーゾーン内の低レイテンシーグループに起動されるため、クラスタープレイスメントグループで実行することをお勧めします。
> - クラスターのインスタンスをスケールするときにキャパシティを使用できるようにするには、クラスタープレイスメントグループのキャパシティ予約を作成します。詳細については、「プレイスメントグループにキャパシティ予約を使用する」を参照してください。

インスタンスを起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択し、[Launch Instances] (インスタンスの起動) を選択して、新しいインスタンス起動ウィザードを開きます。
3. (オプション) [Name and tags] (名前とタグ) セクションで、`EFA-instance` などのインスタンス名を指定します。指定した名前はリソースタグとしてインスタンスに割り当てられます (`Name=`*`EFA-instance`*)。
4. [Application and OS Images] (アプリケーションと OS イメージ) セクションで、[My AMI] (マイ AMI) をクリックし、前のステップで作成した AMI を選択してください。
5. [Instance type] (インスタンスタイプ) セクションで、サポートされているインスタンスタイプを選択してください。
6. [Key pair] (キーペア) セクションで、インスタンスに使用するキーペアを選択してください。
7. [Network settings] (ネットワーク設定) セクションで、[Edit] (編集) を選択し、次の操作を行います。

   a. [サブネット] で、インスタンスを起動するサブネットを選択してください。

   > **Important**
   >
   > サブネットを選択する必要があります。サブネットを選択しない場合、EFA のインスタンスを有効にすることはできません。

b. [Firewall (security groups)] (ファイアウォール (セキュリティグループ)) の場合、[Select existing security group] (既存のセキュリティグループの選択) を選択し、前のステップで作成したセキュリティグループを選択してください。

c. [高度な設定] セクションを展開します。

ネットワークインターフェイス 1 では[ネットワークカードインデックス = 0]、[デバイスインデックス = 0]、[インターフェイスタイプ = EFA と ENA の組み合わせ] を選択してください。

(オプション) `p4d.24xlarge` または `p5.48xlarge` などのマルチカードインスタンスタイプを、必要な追加のネットワークインターフェイスごとに使用する場合は[ネットワークインターフェイスの追加] を選択し、ネットワークカードインデックスの場合は次の未使用インデックスを選択してから、[デバイスインデックス = 1]、[インターフェイスタイプ = EFA と ENA の組み合わせ] または [EFA のみ] を選択してください。

8. (オプション) [Storage] (ストレージ) セクションで、必要に応じてボリュームを設定します。
9. [Advanced details] (高度な詳細) セクションの [Placement group name] (プレイスメントグループ名) で、インスタンスを起動するクラスタープレイスメントグループを選択してください。新しいクラスタープレイスメントグループを作成する必要がある場合は[Create new placement group] (新しいプレイスメントグループの作成) を選択してください。
10. 右側の [合計mary] (サマリー) パネルで、[Number of instances] (インスタンス数) に、起動する EFA 対応のインスタンスの数を入力し、[Launch Instance] (インスタンスの起動) を選択してください。

ステップ 11: 一時インスタンスを終了する

この時点で、ステップ 2 で起動したインスタンスは不要になります。インスタンスを終了して、料金の発生を停止できます。

一時インスタンスを終了するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. 作成した一時インスタンスを選択し、[アクション]、[インスタンスの状態]、[インスタンスの終了 (削除)] の順に選択してください。
4. 確認を求めるメッセージが表示されたら、[終了 (削除)] を選択してください。

ステップ 12: パスワードレス SSH を有効にする

クラスター内のすべてのインスタンスでアプリケーションを実行できるようにするにはリーダーノードからメンバーノードへのパスワードなしの SSH アクセスを有効にする必要があります。リーダーノードはアプリケーションを実行するインスタンスです。クラスター内の残りのインスタンスはメンバーノードです。

クラスター内のインスタンス間でパスワードなしの SSH を有効にするには

1. クラスター内の 1 つのインスタンスをリーダーノードとして選択し、それに接続します。
2. リーダーノード上で strictHostKeyChecking を無効にし ForwardAgent を有効にします。任意のテキストエディタを使用して ~/.ssh/config ファイルを開き、以下を追加します。

```
Host *
    ForwardAgent yes
Host *
    StrictHostKeyChecking no
```

3. RSA キーペアを生成します。

```
$ ssh-keygen -t rsa -N "" -f ~/.ssh/id_rsa
```

キーペアは$HOME/.ssh/ ディレクトリで作成されます。

4. リーダーノードのプライベートキーの許可を変更します。

```
$ chmod 600 ~/.ssh/id_rsa
chmod 600 ~/.ssh/config
```

5. 任意のテキストエディタで ~/.ssh/id_rsa.pub を開き、キーをコピーします。
6. クラスター内の各メンバーノードについて、次の操作を行います。
   a. インスタンスに接続します。
   b. 任意のテキストエディタで ~/.ssh/authorized_keys を開き、前にコピーしたパブリックキーを追加します。
7. パスワードレス SSH が正常に機能していることをテストするにはリーダーノードに接続して、次のコマンドを実行します。

```
$ ssh member_node_private_ip
```

キーまたはパスワードの入力を求められずに、メンバーノードに接続できるはずです。

Amazon EC2 の ML ワークロードに対して EFA と NCCL の使用を開始する

NVIDIA Collective Communications Library (NCCL) は単一のノードまたは複数のノードの複数の GPU のための集合的な標準コミュニケーションルーチンのライブラリです。NCCL は各種の機械学習のワークロードをサポートするために、EFA、Libfabric、MPI と共に使用できます。詳細についてはNCCL のウェブサイトを参照してください。

要件

- 高速コンピューティング P シリーズのインスタンスタイプのみがサポートされています。詳細については、「Amazon EC2 の高速コンピューティングインスタンスの仕様」を参照してください。
- Amazon Linux 2023、Ubuntu 24.04、Ubuntu 22.04 ベースの AMI のみがサポートされています。
- NCCL EFA 以降のみが 2.4.2 でサポートされています。

AWS Deep Learning AMIs を使用して EFA および NCCL で機械学習ワークロードを実行する方法の詳細については、「AWS Deep Learning AMIs デベロッパーガイド」の「DLAMI での EFA の使用」を参照してください。

Steps

- ステップ 1: EFA 対応のセキュリティグループを準備する
- ステップ 2: 一時インスタンスを作成する
- ステップ 3: Nvidia GPU ドライバー、Nvidia CUDA ツールキットおよび cuDNN をインストールする
- ステップ 4: GDRCopy をインストールする
- ステップ 5: EFA ソフトウェアをインストールする
- ステップ 6: NCCL をインストールする
- ステップ 7: NCCL テストをインストールする
- ステップ 8: EFA と NCCL の設定をテストする
- ステップ 9: 機械学習アプリケーションをインストールする

- ステップ 10: EFA および NCCL 対応 AMI を作成する
- ステップ 11: 一時インスタンスを終了する
- ステップ 12: クラスタープレイスメントグループで EFA および NCCL 対応のインスタンスを作成する
- ステップ 13: パスワードレス SSH を有効にする

ステップ 1: EFA 対応のセキュリティグループを準備する

EFA にはセキュリティグループ自体とのインバウンドおよびアウトバウンドのトラフィックをすべて許可するセキュリティグループが必要です。以下の手順ではセキュリティグループを作成します。このセキュリティグループではセキュリティグループ自体とのすべてのインバウンドおよびアウトバウンドのトラフィックと、SSH 接続用の任意の IPv4 アドレスからのインバウンド SSH トラフィックを許可します。

> **Important**
>
> このセキュリティグループはテストのみを目的としています。本番環境ではコンピュータの IP アドレスやローカルネットワークの IP アドレスの範囲など、接続元の IP アドレスからのトラフィックのみを許可するインバウンド SSH ルールを作成することをお勧めします。

その他のシナリオについては[さまざまなユースケースのセキュリティグループのルール]を参照してください。

EFA 対応のセキュリティグループを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Security Groups] (セキュリティグループ) を選択して、[Create security group] (セキュリティグループの作成) を選択してください。
3. [Create security group] (セキュリティグループの作成) ウィンドウで、以下を行います。
   a. [セキュリティグループ名] に、`EFA-enabled security group` のような、分かりやすいセキュリティグループ名を入力してください。
   b. (オプション) [説明] に、セキュリティグループの簡単な説明を入力してください。
   c. [VPC] で、EFA 対応のインスタンスを起動する VPC を選択してください。
   d. [セキュリティグループの作成] を選択してください。

4. 作成したセキュリティグループを選択し、[Details] (詳細) タブで [Security group ID] (セキュリティグループ ID) をコピーします。
5. セキュリティグループが選択された状態で、[Actions] (アクション)、[Edit inbound rules] (インバウンドルールの編集) の順に選択し、次の手順を実行します。
   a. [Add rule] を選択してください。
   b. [Type] で、[All traffic] を選択してください。
   c. [Source type] (送信元タイプ) で、[Custom] (カスタム) を選択し、コピーしたセキュリティグループ ID をフィールドに貼り付けます。
   d. [ルールを追加] を選択してください。
   e. [タイプ] で SSH] を選択してください。
   f. [Source type] (ソースタイプ) で、[Anywhere-IPv4] を選択してください。
   g. [Save Rules] (ルールの保存) を選択してください。
6. セキュリティグループが選択された状態で、[Actions] (アクション)、[Edit outbound rules] (アウトバウンドルールの編集) の順に選択し、次の手順を実行します。
   a. [Add rule] を選択してください。
   b. [Type] で、[All traffic] を選択してください。
   c. [Destination type] (送信先タイプ) で、[Custom] (カスタム) を選択し、コピーしたセキュリティグループ ID をフィールドに貼り付けます。
   d. [Save Rules] (ルールの保存) を選択してください。

ステップ 2: 一時インスタンスを作成する

EFA ソフトウェアコンポーネントのインストールおよび設定に使用する一時インスタンスを起動します。このインスタンスを使用して、EFA 対応のインスタンスを起動する EFA 対応の AMI を作成します。

一時インスタンスを起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択し、[Launch Instances] (インスタンスの起動) を選択して、新しいインスタンス起動ウィザードを開きます。

3. (オプション) [Name and tags] (名前とタグ) セクションで、`EFA-instance` などのインスタンス名を指定します。指定した名前はリソースタグとしてインスタンスに割り当てられます (Name=*`EFA-instance`*)。
4. [アプリケーションと OS のイメージ] セクションで、サポートされるオペレーティングシステムのいずれかに対応する AMI を選択してください。
5. [インスタンスタイプ] セクションで、サポートされているインスタンスタイプを選択してください。
6. [Key pair] (キーペア) セクションで、インスタンスに使用するキーペアを選択してください。
7. [Network settings] (ネットワーク設定) セクションで、[Edit] (編集) を選択し、次の操作を行います。

   a. [サブネット] で、インスタンスを起動するサブネットを選択してください。

   > **Important**
   > サブネットを選択する必要があります。サブネットを選択しない場合、EFA のインスタンスを有効にすることはできません。

   b. [Firewall (security groups)] (ファイアウォール (セキュリティグループ)) の場合、[Select existing security group] (既存のセキュリティグループの選択) を選択し、前のステップで作成したセキュリティグループを選択してください。

   c. [高度な設定] セクションを展開します。

   ネットワークインターフェイス 1 では[ネットワークカードインデックス = 0]、[デバイスインデックス = 0]、[インターフェイスタイプ = EFA と ENA の組み合わせ] を選択してください。

   (オプション) `p4d.24xlarge` または `p5.48xlarge` などのマルチカードインスタンスタイプを、必要な追加のネットワークインターフェイスごとに使用する場合は[ネットワークインターフェイスの追加] を選択し、ネットワークカードインデックスの場合は次の未使用インデックスを選択してから、[デバイスインデックス = 1]、[インターフェイスタイプ = EFA と ENA の組み合わせ] または [EFA のみ] を選択してください。
8. [Storage] (ストレージ) セクションで、必要に応じてボリュームを設定します。

> **Note**
>
> Nvidia CUDA ツールキットには追加の 10 ~ 20 GiB のストレージをプロビジョニングする必要があります。十分な量のストレージをプロビジョニングしないと、Nvidia ドライバーと CUDA ツールキットをインストールしようとしたときに、`insufficient disk space` エラーが発生します。

9. 右側の [合計mary] (サマリー) パネルで、[Launch instance] (インスタンスの起動) を選択してください。

ステップ 3: Nvidia GPU ドライバー、Nvidia CUDA ツールキットおよび cuDNN をインストールする

Amazon Linux 2023

NVIDIA GPU ドライバー、NVIDIA CUDA ツールキットおよび cuDNN をインストールするには

1. すべてのソフトウェアパッケージが最新の状態であることを確認するため、インスタンスでソフトウェアの更新を実行します。

```
$ sudo yum upgrade -y && sudo reboot
```

   インスタンスの再起動後に、再接続します。

2. Nvidia GPU ドライバと Nvidia CUDA ツールキットをインストールするために必要なユーティリティをインストールします。

```
$ sudo yum groupinstall 'Development Tools' -y
```

3. nouveau オープンソースドライバーを無効にします。

   a. 必要なユーティリティ、および現在実行しているカーネルのバージョン用のカーネルヘッダーパッケージをインストールします。

```
$ sudo yum install -y wget kernel-devel-$(uname -r) kernel-headers-$(uname -r)
```

   b. nouveau 拒否リストファイルに `/etc/modprobe.d/blacklist.conf` を追加します。

```
$ cat << EOF | sudo tee --append /etc/modprobe.d/blacklist.conf
blacklist vga16fb
blacklist nouveau
blacklist rivafb
blacklist nvidiafb
blacklist rivatv
EOF
```

c. grub ファイルに GRUB_CMDLINE_LINUX="rdblacklist=nouveau" を追加し、Grub 設定を再構成します。

```
$ echo 'GRUB_CMDLINE_LINUX="rdblacklist=nouveau"' | sudo tee -a /etc/
default/grub \
&& sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

4. インスタンスを再起動して、そのインスタンスに再接続します。
5. 必要なリポジトリを準備する

   - ディストリビューション変数を設定し、CUDA ネットワークリポジトリを設定します。

```
$ distribution=$(. /etc/os-release;echo $ID$VERSION_ID) \
&& ARCH=$( /bin/arch ) \
&& sudo yum-config-manager --add-repo http://developer.download.nvidia.com/
compute/cuda/repos/$distribution/${ARCH}/cuda-$distribution.repo \
&& sudo yum clean expire-cache
```

6. NVIDIA GPU ドライバー、NVIDIA CUDA ツールキット、および cuDNN をインストールします。

```
$ sudo yum clean all \
&& sudo yum -y install nvidia-driver-latest-dkms \
&& sudo yum -y install cuda-drivers-fabricmanager cuda libcudnn8-devel
```

7. インスタンスを再起動して、そのインスタンスに再接続します。
8. (p4d.24xlarge および p5.48xlarge のみ) NVIDIA Fabric Manager サービスを開始し、インスタンスの起動時に自動的に起動することを確認します。NVIDIA Fabric Manager はNV Switch Management に必要です。

```
$ sudo systemctl enable nvidia-fabricmanager && sudo systemctl start nvidia-
fabricmanager
```

9. インスタンスが起動するたびに CUDA パスが設定されていることを確認します。

   - bash シェルの場合、次のステートメントを /home/*username*/.bashrc と /home/*username*/.bash_profile に追加します。

   ```
   export PATH=/usr/local/cuda/bin:$PATH
   export LD_LIBRARY_PATH=/usr/local/cuda/lib64:/usr/local/cuda/extras/CUPTI/
   lib64:$LD_LIBRARY_PATH
   ```

   - tcsh シェルの場合、次の文を /home/*username*/.cshrc に追加します。

   ```
   setenv PATH=/usr/local/cuda/bin:$PATH
   setenv LD_LIBRARY_PATH=/usr/local/cuda/lib64:/usr/local/cuda/extras/CUPTI/
   lib64:$LD_LIBRARY_PATH
   ```

10. 以下のコマンドを実行して、Nvidia GPU ドライバが機能することを確認します。

    ```
    $ nvidia-smi -q | head
    ```

    このコマンドはNvidia GPU、Nvidia GPU ドライバ、Nvidia CUDA ツールキットの情報を返します。

Ubuntu 24.04 and Ubuntu 22.04

NVIDIA GPU ドライバー、NVIDIA CUDA ツールキットおよび cuDNN をインストールするには

1. すべてのソフトウェアパッケージが最新の状態であることを確認するため、インスタンスでソフトウェアの更新を実行します。

   ```
   $ sudo apt-get update && sudo apt-get upgrade -y
   ```

2. Nvidia GPU ドライバと Nvidia CUDA ツールキットをインストールするために必要なユーティリティをインストールします。

   ```
   $ sudo apt-get update && sudo apt-get install build-essential -y
   ```

3. Nvidia GPU ドライバを使用するにはまず、nouveau オープンソースドライバを無効にする必要があります。

a. 必要なユーティリティ、および現在実行しているカーネルのバージョン用のカーネルヘッダーパッケージをインストールします。

```
$ sudo apt-get install -y gcc make linux-headers-$(uname -r)
```

b. nouveau 拒否リストファイルに /etc/modprobe.d/blacklist.conf を追加します。

```
$ cat << EOF | sudo tee --append /etc/modprobe.d/blacklist.conf
blacklist vga16fb
blacklist nouveau
blacklist rivafb
blacklist nvidiafb
blacklist rivatv
EOF
```

c. 任意のテキストエディタを使用して /etc/default/grub ファイルを開き、以下を追加します。

```
GRUB_CMDLINE_LINUX="rdblacklist=nouveau"
```

d. Grub 設定を再構築します。

```
$ sudo update-grub
```

4. インスタンスを再起動して、そのインスタンスに再接続します。
5. CUDA リポジトリを追加し、Nvidia GPU ドライバー、NVIDIA CUDA ツールキット、および cuDNN をインストールします。

   • p3dn.24xlarge

```
$ sudo apt-key adv --fetch-keys http://developer.download.nvidia.com/compute/
machine-learning/repos/ubuntu2004/x86_64/7fa2af80.pub \
&& wget -O /tmp/deeplearning.deb http://developer.download.nvidia.com/compute/
machine-learning/repos/ubuntu2004/x86_64/nvidia-machine-learning-repo-
ubuntu2004_1.0.0-1_amd64.deb \
&& sudo dpkg -i /tmp/deeplearning.deb \
&& wget -O /tmp/cuda.pin https://developer.download.nvidia.com/compute/cuda/
repos/ubuntu2004/x86_64/cuda-ubuntu2004.pin \
&& sudo mv /tmp/cuda.pin /etc/apt/preferences.d/cuda-repository-pin-600 \
```

```
&& sudo apt-key adv --fetch-keys https://developer.download.nvidia.com/
compute/cuda/repos/ubuntu2004/x86_64/3bf863cc.pub \
&& sudo add-apt-repository 'deb http://developer.download.nvidia.com/compute/
cuda/repos/ubuntu2004/x86_64/ /' \
&& sudo apt update \
&& sudo apt install nvidia-dkms-535 \
&& sudo apt install -o Dpkg::Options::='--force-overwrite' cuda-drivers-535
 cuda-toolkit-12-3 libcudnn8 libcudnn8-dev -y
```

- p4d.24xlarge および p5.48xlarge

```
$ sudo apt-key adv --fetch-keys http://developer.download.nvidia.com/compute/
machine-learning/repos/ubuntu2004/x86_64/7fa2af80.pub \
&& wget -O /tmp/deeplearning.deb http://developer.download.nvidia.com/compute/
machine-learning/repos/ubuntu2004/x86_64/nvidia-machine-learning-repo-
ubuntu2004_1.0.0-1_amd64.deb \
&& sudo dpkg -i /tmp/deeplearning.deb \
&& wget -O /tmp/cuda.pin https://developer.download.nvidia.com/compute/cuda/
repos/ubuntu2004/x86_64/cuda-ubuntu2004.pin \
&& sudo mv /tmp/cuda.pin /etc/apt/preferences.d/cuda-repository-pin-600 \
&& sudo apt-key adv --fetch-keys https://developer.download.nvidia.com/
compute/cuda/repos/ubuntu2004/x86_64/3bf863cc.pub \
&& sudo add-apt-repository 'deb http://developer.download.nvidia.com/compute/
cuda/repos/ubuntu2004/x86_64/ /' \
&& sudo apt update \
&& sudo apt install nvidia-kernel-open-535 \
&& sudo apt install -o Dpkg::Options::='--force-overwrite' cuda-drivers-535
 cuda-toolkit-12-3 libcudnn8 libcudnn8-dev -y
```

6. インスタンスを再起動して、そのインスタンスに再接続します。
7. (p4d.24xlarge および p5.48xlarge のみ) NVIDIA Fabric Manager をインストールします。

   a. 前の手順でインストールした Nvidia カーネルモジュールのバージョンと一致する Nvidia Fabric Manager のバージョンをインストールする必要があります。

      Nvidia カーネルモジュールのバージョンを確認するには次のコマンドを実行します。

      ```
      $ cat /proc/driver/nvidia/version | grep "Kernel Module"
      ```

      以下は出力の例です。

```
NVRM version: NVIDIA UNIX x86_64 Kernel Module  450.42.01  Tue Jun 15
 21:26:37 UTC 2021
```

上記の例ではメジャーバージョン 450 のカーネルモジュールがインストールされました。これはNvidia Fabric Manager のバージョン 450 をインストールする必要があることを意味します。

b. Nvidia Fabric Manager をインストールする 次のコマンドを、前の手順で識別されたメジャーバージョンを指定して実行します。

```
$ sudo apt install -o Dpkg::Options::='--force-overwrite' nvidia-
fabricmanager-major_version_number
```

例えば、メジャーバージョン 450 のカーネルモジュールがインストールされた場合、以下のコマンドを使用して、一致するバージョンの Nvidia Fabric Manager をインストールします。

```
$ sudo apt install -o Dpkg::Options::='--force-overwrite' nvidia-
fabricmanager-450
```

c. サービスを開始し、インスタンスの起動時に自動的に起動することを確認します。NVIDIA Fabric Manager はNV Switch Management に必要です。

```
$ sudo systemctl start nvidia-fabricmanager && sudo systemctl enable nvidia-
fabricmanager
```

8. インスタンスが起動するたびに CUDA パスが設定されていることを確認します。

- bash シェルの場合、次のステートメントを /home/*username*/.bashrc と /home/*username*/.bash_profile に追加します。

```
export PATH=/usr/local/cuda/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/cuda/lib64:/usr/local/cuda/extras/CUPTI/
lib64:$LD_LIBRARY_PATH
```

- tcsh シェルの場合、次の文を /home/*username*/.cshrc に追加します。

```
setenv PATH=/usr/local/cuda/bin:$PATH
```

```
setenv LD_LIBRARY_PATH=/usr/local/cuda/lib64:/usr/local/cuda/extras/CUPTI/
lib64:$LD_LIBRARY_PATH
```

9. 以下のコマンドを実行して、Nvidia GPU ドライバが機能することを確認します。

```
$ nvidia-smi -q | head
```

このコマンドはNvidia GPU、Nvidia GPU ドライバ、Nvidia CUDA ツールキットの情報を返します。

ステップ 4: GDRCopy をインストールする

GDRCopy をインストールして Libfabric のパフォーマンスを向上させます。GDRCopy の詳細については「GDRCopy レポジトリ」を参照してください。

Amazon Linux 2023

GDRCopy をインストールするには

1. 必要な依存ファイルをインストールします。

```
$ sudo yum -y install dkms rpm-build make check check-devel subunit subunit-
devel
```

2. GDRCopy パッケージをダウンロードして解凍します。

```
$ wget https://github.com/NVIDIA/gdrcopy/archive/refs/tags/v2.4.tar.gz \
&& tar xf v2.4.tar.gz ; cd gdrcopy-2.4/packages
```

3. GDRCopy RPM パッケージをビルドします。

```
$ CUDA=/usr/local/cuda ./build-rpm-packages.sh
```

4. GDRCopy RPM パッケージをインストールします。

```
$ sudo rpm -Uvh gdrcopy-kmod-2.4-1dkms.noarch*.rpm \
&& sudo rpm -Uvh gdrcopy-2.4-1.x86_64*.rpm \
&& sudo rpm -Uvh gdrcopy-devel-2.4-1.noarch*.rpm
```

Ubuntu 24.04 and Ubuntu 22.04

GDRCopy をインストールするには

1. 必要な依存ファイルをインストールします。

```
$ sudo apt -y install build-essential devscripts debhelper check libsubunit-dev
 fakeroot pkg-config dkms
```

2. GDRCopy パッケージをダウンロードして解凍します。

```
$ wget https://github.com/NVIDIA/gdrcopy/archive/refs/tags/v2.4.tar.gz \
&& tar xf v2.4.tar.gz \
&& cd gdrcopy-2.4/packages
```

3. GDRCopy RPM パッケージをビルドします。

```
$ CUDA=/usr/local/cuda ./build-deb-packages.sh
```

4. GDRCopy RPM パッケージをインストールします。

```
$ sudo dpkg -i gdrdrv-dkms_2.4-1_amd64.*.deb \
&& sudo dpkg -i libgdrapi_2.4-1_amd64.*.deb \
&& sudo dpkg -i gdrcopy-tests_2.4-1_amd64.*.deb \
&& sudo dpkg -i gdrcopy_2.4-1_amd64.*.deb
```

ステップ 5: EFA ソフトウェアをインストールする

一時インスタンスで EFA をサポートするために必要な EFA 対応のカーネル、EFA ドライバー、Libfabric、および Open MPI スタックをインストールします。

EFA ソフトウェアをインストールするには

1. 起動したインスタンスに接続します。詳細については「SSH を使用した Linux インスタンスへの接続」を参照してください。
2. EFA ソフトウェアのインストールファイルをダウンロードします。ソフトウェアのインストールファイルは圧縮された tar (`.tar.gz`) ファイルにパッケージ化されています。次のコマンドを使用して、安定している最新バージョンをダウンロードします。

前述のコマンドのバージョン番号を latest に置き換えることで最新バージョンを取得することもできます。

```
$ curl -O https://efa-installer.amazonaws.com/aws-efa-installer-1.49.0.tar.gz
```

3. (オプション) EFA tarball (.tar.gz) ファイルの認証と完全性を検証します。

   ソフトウェアパブリッシャーの ID を検証し、発行後にファイルの改変や破損がないことを確認するために、これを行うことをお勧めします。tar ファイルを検証しない場合はこの手順をスキップします。

   > **Note**
   >
   > 代わりに、MD5 または SHA256 チェックサムを使用して tar ファイルを検証する場合はチェックサムを使用した EFA インストーラの検証を参照してください。

   a. パブリック GPG キーをダウンロードして、キーリングにインポートします。

      ```
      $ wget https://efa-installer.amazonaws.com/aws-efa-installer.key && gpg --
      import aws-efa-installer.key
      ```

      コマンドはキーの値を返します。次の手順で必要になるため、キーの値を書きとめておきます。

   b. GPG キーのフィンガープリントを検証します。次のコマンドを実行し、前のステップで作成したキーの値を指定します。

      ```
      $ gpg --fingerprint key_value
      ```

      コマンドは4E90 91BC BB97 A96B 26B1 5E59 A054 80B1 DD2D 3CCC と同じフィンガープリントを返します。フィンガープリントが一致しない場合はEFA インストールスクリプトを実行せず、サポート にお問い合わせください。

   c. 署名ファイルをダウンロードし、EFA tar ファイルの署名を検証します。

      ```
      $ wget https://efa-installer.amazonaws.com/aws-efa-installer-1.49.0.tar.gz.sig
       && gpg --verify ./aws-efa-installer-1.49.0.tar.gz.sig
      ```

出力例を次に示します。

```
gpg: Signature made Wed 29 Jul 2020 12:50:13 AM UTC using RSA key ID DD2D3CCC
gpg: Good signature from "Amazon EC2 EFA <ec2-efa-maintainers@amazon.com>"
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: 4E90 91BC BB97 A96B 26B1  5E59 A054 80B1 DD2D 3CCC
```

結果に Good signature が含まれ、フィンガープリントが前のステップで返されたフィンガープリントと一致する場合は次のステップに進みます。そうでない場合はEFA インストールスクリプトを実行せず、サポート にお問い合わせください。

4. 圧縮された .tar.gz ファイルからファイルを展開し、展開されたディレクトリに移動します。

```
$ tar -xf aws-efa-installer-1.49.0.tar.gz && cd aws-efa-installer
```

5. (オプション) インストール中に個々のパッケージ署名を検証します。

EFA インストーラ 1.48.0 以降では、インストーラには GPG 署名の個々の RPM および DEB パッケージが含まれています。インストール中に個々のパッケージの信頼性と整合性を検証するには、--check-signatures フラグを使用します。このフラグを有効にすると、まず、インストーラはすべてのパッケージ署名を検証し、すべてのパッケージが検証に合格した場合にのみインストールを続行します。パッケージが検証に失敗すると、インストーラは何もインストールせずにすぐに終了します。

a. GPG 公開鍵をダウンロードします。

```
$ wget https://efa-installer.amazonaws.com/aws-efa-installer.key
```

b. キーパスをエクスポートします。次に、次のステップで、インストールコマンドに --check-signatures を追加し、sudo の代わりに sudo -E を使用して環境変数を保持します。

```
$ export EFA_INSTALLER_KEY=$(pwd)/aws-efa-installer.key
```

RPM ベースのシステム (Amazon Linux、RHEL、Rocky Linux、SUSE) では、インストーラは rpm --checksig を使用して各 RPM を検証します。DEB ベースのシステム (Ubuntu、Debian) では、インストーラは GPG 署名検証を使用して各 DEB を検証します。

パッケージの検証に失敗すると、インストールは直ちに中止され、壊れたパッケージや悪意のあるパッケージからシステムを保護します。

> **Note**
>
> `--check-signatures` フラグはオプションです。これがないと、インストーラは個々の署名検証を実行しません。

6. EFA ソフトウェアのインストールスクリプトを実行します。

> **Note**
>
> 前のオプションステップを完了してパッケージ署名の検証をセットアップした場合は、インストールコマンドに `--check-signatures` を追加して、sudo の代わりに sudo -E を使用します。例: `sudo -E ./efa_installer.sh -y --mpi=openmpi4 --check-signatures`。

> **Note**
>
> EFA 1.30.0 からは　Open MPI 4.1 と Open MPI 5 の両方がデフォルトでインストールされます。Open MPI 5 が必要でない限り、Open MPI 4.1 のみをインストールすることをお勧めします。以下のコマンドは Open MPI 4.1 のみをインストールします。Open MPI 4.1 と Open MPI 5 をインストールする場合は `--mpi=openmpi4` を削除してください。

```
$ sudo ./efa_installer.sh -y --mpi=openmpi4
```

Libfabric は `/opt/amazon/efa` ディレクトリにインストールされます。aws-ofi-nccl プラグインは `/opt/amazon/ofi-nccl` ディレクトリにインストールされます。オープン MPI は `/opt/amazon/openmpi` ディレクトリにインストールされます。

7. EFA インストーラーでインスタンスの再起動を求めるメッセージが表示された場合は再起動してからインスタンスに再接続します。それ以外の場合はインスタンスからログアウトし、再度ログインしてインストールを完了します。
8. EFA ソフトウェアコンポーネントが正常にインストールされたことを確認します。

```
$ fi_info -p efa -t FI_EP_RDM
```

コマンドによって、Libfabric の EFA インターフェイスに関する情報が返ります。以下の例はコマンド出力を示しています。

- 単一のネットワークインターフェイスを持つ p3dn.24xlarge

```
provider: efa
fabric: EFA-fe80::94:3dff:fe89:1b70
domain: efa_0-rdm
version: 2.0
type: FI_EP_RDM
protocol: FI_PROTO_EFA
```

- 複数のネットワークインターフェイスを持つ p4d.24xlarge および p5.48xlarge

```
provider: efa
fabric: EFA-fe80::c6e:8fff:fef6:e7ff
domain: efa_0-rdm
version: 111.0
type: FI_EP_RDM
protocol: FI_PROTO_EFA
provider: efa
fabric: EFA-fe80::c34:3eff:feb2:3c35
domain: efa_1-rdm
version: 111.0
type: FI_EP_RDM
protocol: FI_PROTO_EFA
provider: efa
fabric: EFA-fe80::c0f:7bff:fe68:a775
domain: efa_2-rdm
version: 111.0
type: FI_EP_RDM
protocol: FI_PROTO_EFA
provider: efa
fabric: EFA-fe80::ca7:b0ff:fea6:5e99
domain: efa_3-rdm
version: 111.0
type: FI_EP_RDM
protocol: FI_PROTO_EFA
```

ステップ 6: NCCL をインストールする

NCCL をインストールします。NCCL に関する詳細についてはNCCL repositoryを参照してください。

NCCL をインストールするには

1. /opt ディレクトリに移動します。

```
$ cd /opt
```

2. 公式の NCCL リポジトリをインスタンスにクローンし、ローカルのクローンされたリポジトリに移動します。

```
$ sudo git clone https://github.com/NVIDIA/nccl.git -b v2.23.4-1 && cd nccl
```

3. NCCL を構築およびインストールし、CUDA インストールディレクトリを指定します。

```
$ sudo make -j src.build CUDA_HOME=/usr/local/cuda
```

ステップ 7: NCCL テストをインストールする

NCCL テストをインストールします。NCCL テストではNCCL が適切にインストールされていることを確認し、想定どおりに機能していることを確認できます。NCCL テストに関する詳細についてはnccl-tests リポジトリを参照してください。

NCCL テストをインストールするには

1. ホームディレクトリに移動します。

```
$ cd $HOME
```

2. 公式の nccl-tests リポジトリをインスタンスにクローンし、ローカルのクローンされたリポジトリに移動します。

```
$ git clone https://github.com/NVIDIA/nccl-tests.git && cd nccl-tests
```

3. Libfabric ディレクトリを LD_LIBRARY_PATH 変数に追加します。

   - Amazon Linux 2023

```
$ export LD_LIBRARY_PATH=/opt/amazon/efa/lib64:$LD_LIBRARY_PATH
```

- Ubuntu 24.04 と Ubuntu 22.04

```
$ export LD_LIBRARY_PATH=/opt/amazon/efa/lib:$LD_LIBRARY_PATH
```

4. NCCL テストをインストールし、MPI、NCCL、CUDA インストールディレクトリを指定します。

```
$ make MPI=1 MPI_HOME=/opt/amazon/openmpi NCCL_HOME=/opt/nccl/build CUDA_HOME=/usr/local/cuda
```

ステップ 8: EFA と NCCL の設定をテストする

テストを実行し、EFA と NCCL に一時インスタンスが適切に設定されていることを確認します。

EFA と NCCL 設定をテストするには

1. テストを実行するホストを指定するホストファイルを作成します。以下のコマンドはインスタンス自体へのリファレンスを含む my-hosts と呼ばれるホストファイルを作成します。

   IMDSv2

   ```
   [ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H
    "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
   && curl -H "X-aws-ec2-metadata-token: $TOKEN" -v http://169.254.169.254/latest/meta-data/local-ipv4 >> my-hosts
   ```

   IMDSv1

   ```
   [ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/local-ipv4 >> my-hosts
   ```

2. テストを実行し、ホストファイル (--hostfile) と使用する GPU の数 (-n) を指定します。以下のコマンドはインスタンス自体の 8 つの GPU で all_reduce_perf テストを実行し、以下の環境変数を指定します。

   - FI_EFA_USE_DEVICE_RDMA=1 — (p4d.24xlarge のみ) 片側転送および両側転送にデバイスの RDMA 機能を使用します。

- NCCL_DEBUG=INFO – 詳細なデバッグ出力を有効にします。また、テストの開始時に NCCL バージョンのみをプリントするために VERSION を指定したり、エラーメッセージのみを受信するために WARN を指定したりすることもできます。

NCCL テスト引数に関する詳細は公式の nccl-tests リポジトリのNCCL Tests READMEを参照してください。

- p3dn.24xlarge

```
$ /opt/amazon/openmpi/bin/mpirun \
-x LD_LIBRARY_PATH=/opt/nccl/build/lib:/usr/local/cuda/lib64:/opt/amazon/efa/
lib:/opt/amazon/openmpi/lib:/opt/amazon/ofi-nccl/lib:$LD_LIBRARY_PATH \
-x NCCL_DEBUG=INFO \
--hostfile my-hosts -n 8 -N 8 \
--mca pml ^cm --mca btl tcp,self --mca btl_tcp_if_exclude lo,docker0 --bind-to
 none \
$HOME/nccl-tests/build/all_reduce_perf -b 8 -e 1G -f 2 -g 1 -c 1 -n 100
```

- p4d.24xlarge および p5.48xlarge

```
$ /opt/amazon/openmpi/bin/mpirun \
-x FI_EFA_USE_DEVICE_RDMA=1 \
-x LD_LIBRARY_PATH=/opt/nccl/build/lib:/usr/local/cuda/lib64:/opt/amazon/efa/
lib:/opt/amazon/openmpi/lib:/opt/amazon/ofi-nccl/lib:$LD_LIBRARY_PATH \
-x NCCL_DEBUG=INFO \
--hostfile my-hosts -n 8 -N 8 \
--mca pml ^cm --mca btl tcp,self --mca btl_tcp_if_exclude lo,docker0 --bind-to
 none \
$HOME/nccl-tests/build/all_reduce_perf -b 8 -e 1G -f 2 -g 1 -c 1 -n 100
```

3. NCCL_DEBUG ログが出力されるときに、EFA が NCCL の基盤となるプロバイダーとしてアクティブであることを確認できます。

```
ip-192-168-2-54:14:14 [0] NCCL INFO NET/OFI Selected Provider is efa*
```

p4d.24xlarge インスタンスの使用時に、次の追加情報が表示されます。

```
ip-192-168-2-54:14:14 [0] NCCL INFO NET/OFI Running on P4d platform, Setting
 NCCL_TOPO_FILE environment variable to /home/ec2-user/install/plugin/share/aws-
ofi-nccl/xml/p4d-24xl-topo.xml
```

ステップ 9: 機械学習アプリケーションをインストールする

機械学習アプリケーションを一時インスタンスにインストールします。インストール手順はそれぞれの機械学習アプリケーションによって異なります。お使いの Linux インスタンスへのソフトウェアインストールの詳細については、「Amazon Linux 2023 ユーザーガイド」の「OS 更新の管理」を参照してください。

> **Note**
>
> インストール手順については機械学習アプリケーションのドキュメントを参照してください。

ステップ 10: EFA および NCCL 対応 AMI を作成する

必要なソフトウェアコンポーネントのインストール後、EFA 対応のインスタンスの起動に再利用できる AMI を作成します。

一時インスタンスから AMI を作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. 作成した一時インスタンスを選択し、[アクション]、[イメージ]、[イメージの作成] の順に選択してください。
4. [イメージの作成] で、次を行います。
   a. [イメージ名] に、の分かりやすい AMI 名を入力してください。
   b. (オプション) [イメージの説明] に、AMI の簡単な説明を入力してください。
   c. [イメージを作成] を選択してください。
5. ナビゲーションペインで [AMI] を選択してください。
6. リストで作成した AMI を探します。ステータスが `pending` から `available` に変わるまで待ってから、次のステップに進みます。

ステップ 11: 一時インスタンスを終了する

この時点で、起動した一時インスタンスは不要になります。インスタンスを終了して、料金の発生を停止できます。

一時インスタンスを終了するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. 作成した一時インスタンスを選択し、[アクション]、[インスタンスの状態]、[インスタンスの終了] の順に選択してください。
4. 確認を求めるメッセージが表示されたら、[終了] を選択してください。

ステップ 12: クラスタープレイスメントグループで EFA および NCCL 対応のインスタンスを作成する

前に作成した EFA 対応の AMI と EFA 対応のセキュリティグループを使用して、EFA および NCCL 対応のインスタンスをクラスタープレイスメントグループ内で起動します。

> **Note**
>
> - EFA 対応のインスタンスをクラスタープレイスメントグループに起動することは絶対的な要件ではありません。ただし、EFA 対応のインスタンスは 1 つのアベイラビリティーゾーン内の低レイテンシーグループに起動されるため、クラスタープレイスメントグループで実行することをお勧めします。
> - クラスターのインスタンスをスケールするときにキャパシティを使用できるようにするにはクラスタープレイスメントグループのキャパシティ予約を作成します。詳細については「プレイスメントグループにキャパシティ予約を使用する」を参照してください。

New console

一時インスタンスを起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択し、[Launch Instances] (インスタンスの起動) を選択して、新しいインスタンス起動ウィザードを開きます。
3. (オプション) [Name and tags] (名前とタグ) セクションで、`EFA-instance` などのインスタンス名を指定します。指定した名前はリソースタグとしてインスタンスに割り当てられます (Name=*EFA-instance*)。

4. [Application and OS Images] (アプリケーションと OS イメージ) セクションで、[My AMI] (マイ AMI) をクリックし、前のステップで作成した AMI を選択してください。
5. [Instance type] (インスタンスタイプ) セクションで、`p3dn.24xlarge` または `p4d.24xlarge` のいずれかを選択してください。
6. [Key pair] (キーペア) セクションで、インスタンスに使用するキーペアを選択してください。
7. [Network settings] (ネットワーク設定) セクションで、[Edit] (編集) を選択し、次の操作を行います。

   a. [サブネット] で、インスタンスを起動するサブネットを選択してください。サブネットを選択しない場合、EFA のインスタンスを有効にすることはできません。

   b. [Firewall (security groups)] (ファイアウォール (セキュリティグループ)) の場合、[Select existing security group] (既存のセキュリティグループの選択) を選択し、前のステップで作成したセキュリティグループを選択してください。

   c. [高度な設定] セクションを展開します。

      ネットワークインターフェイス 1 では[ネットワークカードインデックス = 0]、[デバイスインデックス = 0]、および[インターフェイスタイプ = EFA と ENA の組み合わせ] を選択してください。

      (オプション) `p4d.24xlarge` または `p5.48xlarge` などのマルチカードインスタンスタイプを使用している場合は必要な追加のネットワークインターフェイスごとに[ネットワークインターフェイスの追加] を選択し、ネットワークカードインデックスの場合は次の未使用インデックスを選択してから、[デバイスインデックス = 1]、[インターフェイスタイプ = ENA を含む EFA] または [EFA のみ] を選択してください。
8. (オプション) [Storage] (ストレージ) セクションで、必要に応じてボリュームを設定します。
9. [Advanced details] (高度な詳細) セクションの [Placement group name] (プレイスメントグループ名) で、インスタンスを起動するクラスタープレイスメントグループを選択してください。新しいクラスタープレイスメントグループを作成する必要がある場合は[Create new placement group] (新しいプレイスメントグループの作成) を選択してください。
10. 右側の [合計mary] (サマリー) パネルで、[Number of instances] (インスタンス数) に、起動する EFA 対応のインスタンスの数を入力し、[Launch Instance] (インスタンスの起動) を選択してください。

Old console

EFA および NCCL 対応のインスタンスをクラスタープレイスメントグループに起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. [インスタンスの作成] を選択してください。
3. [AMI の選択] ページで、[マイ AMI] を選択し、前に作成した AMI を見つけて、[選択] をクリックしてください。
4. [インスタンスタイプの選択] ページで [p3dn.24xlarge] を選択し、[次へ: インスタンスの詳細の設定] を選択してください。
5. [インスタンスの詳細設定] ページで、以下の操作を実行します。
   a. [インスタンス数] に、起動する EFA および NCCL 対応のインスタンスの数を入力してください。
   b. [ネットワーク] および [サブネット] で、インスタンスを起動する VPC およびサブネットを選択してください。
   c. [プレイスメントグループ] で、[インスタンスをプレイスメントグループに追加します] チェックボックスをオンにします。
   d. [プレイスメントグループ名] で、[新しいプレイスメントグループに追加する] チェックボックスをオンにし、分かりやすいプレイスメントグループ名を入力してください。次に、[プレイスメントグループ戦略] で [クラスター] を選択してください。
   e. [EFA] で、[有効化] を選択してください。
   f. [ネットワークインターフェイス] セクションの [eth0] で、[新しいネットワークインターフェイス] を選択してください。必要に応じて、プライマリ IPv4 アドレスと 1 つ以上のセカンダリ IPv4 アドレスを指定できます。関連付けられている IPv6 CIDR ブロックを持つサブネットにインスタンスを起動する場合は必要に応じて、プライマリ IPv6 アドレスと 1 つ以上のセカンダリ IPv6 アドレスを指定できます。
   g. [次の手順: ストレージの追加] を選択してください。
6. [ストレージの追加] ページで、AMI が指定するボリューム (ルートデバイスボリュームなど) に加えて、インスタンスにアタッチするボリュームを指定します。次に、[次の手順: タグの追加] を選択してください。
7. [Add Tags] ページで、ユーザーフレンドリーな名前などを使ってインスタンスのタグを指定し、[Next: Configure Security Group] を選択してください。

8. [セキュリティグループの設定] ページの [セキュリティグループの割り当て] で、[既存のセキュリティグループの選択] を選択し、前に作成したセキュリティグループを選択してください。
9. [Review and Launch] を選択してください。
10. [インスタンス作成の確認] ページで設定を確認し、[起動] を選択してキーペアを選択し、インスタンスを起動します。

ステップ 13: パスワードレス SSH を有効にする

クラスター内のすべてのインスタンスでアプリケーションを実行できるようにするにはリーダーノードからメンバーノードへのパスワードなしの SSH アクセスを有効にする必要があります。リーダーノードはアプリケーションを実行するインスタンスです。クラスター内の残りのインスタンスはメンバーノードです。

クラスター内のインスタンス間でパスワードなしの SSH を有効にするには

1. クラスター内の 1 つのインスタンスをリーダーノードとして選択し、それに接続します。
2. リーダーノード上で `strictHostKeyChecking` を無効にし `ForwardAgent` を有効にします。任意のテキストエディタを使用して `~/.ssh/config` ファイルを開き、以下を追加します。

```
Host *
    ForwardAgent yes
Host *
    StrictHostKeyChecking no
```

3. RSA キーペアを生成します。

```
$ ssh-keygen -t rsa -N "" -f ~/.ssh/id_rsa
```

キーペアは`$HOME/.ssh/` ディレクトリで作成されます。

4. リーダーノードのプライベートキーの許可を変更します。

```
$ chmod 600 ~/.ssh/id_rsa
chmod 600 ~/.ssh/config
```

5. 任意のテキストエディタで `~/.ssh/id_rsa.pub` を開き、キーをコピーします。
6. クラスター内の各メンバーノードについて、次の操作を行います。

a. インスタンスに接続します。

b. 任意のテキストエディタで ~/.ssh/authorized_keys を開き、前にコピーしたパブリックキーを追加します。

7. パスワードレス SSH が正常に機能していることをテストするにはリーダーノードに接続して、次のコマンドを実行します。

```
$ ssh member_node_private_ip
```

キーまたはパスワードの入力を求められずに、メンバーノードに接続できるはずです。

Amazon EC2 の推論ワークロードに対して EFA と NIXL の使用を開始する

NVIDIA Inference Xfer Library (NIXL) は、分散推論ワークロード専用に設計された、高スループットで低レイテンシーの通信ライブラリです。NIXL は、EFA および Libfabric とともに使用すれば、プリフィルノードとデコードノード間の KV キャッシュ転送をサポートでき、さまざまなストレージレイヤー間での KV キャッシュ移動を効率化できます。詳細については、NIXL のウェブサイトを参照してください。

要件

- Ubuntu 24.04 と Ubuntu 22.04 ベースの AMI のみがサポートされています。
- EFA は NIXL 1.0.0 以降のみをサポートしています。

ステップ 1: EFA 対応のセキュリティグループを準備する

EFA にはセキュリティグループ自体とのインバウンドおよびアウトバウンドのトラフィックをすべて許可するセキュリティグループが必要です。以下の手順ではセキュリティグループを作成します。このセキュリティグループではセキュリティグループ自体とのすべてのインバウンドおよびアウトバウンドのトラフィックと、SSH 接続用の任意の IPv4 アドレスからのインバウンド SSH トラフィックを許可します。

> **Important**
>
> このセキュリティグループはテストのみを目的としています。本番環境ではコンピュータの IP アドレスやローカルネットワークの IP アドレスの範囲など、接続元の IP アドレスからのトラフィックのみを許可するインバウンド SSH ルールを作成することをお勧めします。

その他のシナリオについてはさまざまなユースケースのセキュリティグループのルールを参照してください。

EFA 対応のセキュリティグループを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Security Groups] (セキュリティグループ) を選択して、[Create security group] (セキュリティグループの作成) を選択してください。
3. [Create security group] (セキュリティグループの作成) ウィンドウで、以下を行います。
   a. [セキュリティグループ名] に、`EFA-enabled security group` のような、分かりやすいセキュリティグループ名を入力してください。
   b. (オプション) [説明] に、セキュリティグループの簡単な説明を入力してください。
   c. [VPC] で、EFA 対応のインスタンスを起動する VPC を選択してください。
   d. [セキュリティグループの作成] を選択してください。
4. 作成したセキュリティグループを選択し、[Details] (詳細) タブで [Security group ID] (セキュリティグループ ID) をコピーします。
5. セキュリティグループが選択された状態で、[Actions] (アクション)、[Edit inbound rules] (インバウンドルールの編集) の順に選択し、次の手順を実行します。
   a. [Add rule] を選択してください。
   b. [Type] で、[All traffic] を選択してください。
   c. [Source type] (送信元タイプ) で、[Custom] (カスタム) を選択し、コピーしたセキュリティグループ ID をフィールドに貼り付けます。
   d. [ルールを追加] を選択してください。
   e. [タイプ] で SSH] を選択してください。
   f. [Source type] (ソースタイプ) で、[Anywhere-IPv4] を選択してください。
   g. [Save Rules] (ルールの保存) を選択してください。
6. セキュリティグループが選択された状態で、[Actions] (アクション)、[Edit outbound rules] (アウトバウンドルールの編集) の順に選択し、次の手順を実行します。
   a. [Add rule] を選択してください。
   b. [Type] で、[All traffic] を選択してください。
   c. [Destination type] (送信先タイプ) で、[Custom] (カスタム) を選択し、コピーしたセキュリティグループ ID をフィールドに貼り付けます。

d. [Save Rules] (ルールの保存) を選択してください。

ステップ 2: 一時インスタンスを作成する

EFA ソフトウェアコンポーネントのインストールおよび設定に使用する一時インスタンスを起動します。このインスタンスを使用して、EFA 対応のインスタンスを起動する EFA 対応の AMI を作成します。

一時インスタンスを起動するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択し、[Launch Instances] (インスタンスの起動) を選択して、新しいインスタンス起動ウィザードを開きます。
3. (オプション) [Name and tags] (名前とタグ) セクションで、`EFA-instance` などのインスタンス名を指定します。指定した名前はリソースタグとしてインスタンスに割り当てられます (Name=*`EFA-instance`*)。
4. [アプリケーションと OS のイメージ] セクションで、サポートされるオペレーティングシステムのいずれかに対応する AMI を選択してください。DLAMI リリースノートページで、サポートされている DLAMI を選択することもできます。
5. [インスタンスタイプ] セクションで、サポートされているインスタンスタイプを選択してください。
6. [Key pair] (キーペア) セクションで、インスタンスに使用するキーペアを選択してください。
7. [Network settings] (ネットワーク設定) セクションで、[Edit] (編集) を選択し、次の操作を行います。

   a. [サブネット] で、インスタンスを起動するサブネットを選択してください。サブネットを選択しない場合、EFA のインスタンスを有効にすることはできません。

   b. [Firewall (security groups)] (ファイアウォール (セキュリティグループ)) の場合、[Select existing security group] (既存のセキュリティグループの選択) を選択し、前のステップで作成したセキュリティグループを選択してください。

   c. [高度な設定] セクションを展開します。

      ネットワークインターフェイス 1 では[ネットワークカードインデックス = 0]、[デバイスインデックス = 0]、[インターフェイスタイプ = EFA と ENA の組み合わせ] を選択してください。

(オプション) p4d.24xlarge または p5.48xlarge などのマルチカードインスタンスタイプを、必要な追加のネットワークインターフェイスごとに使用する場合は[ネットワークインターフェイスの追加] を選択し、ネットワークカードインデックスの場合は次の未使用インデックスを選択してから、[デバイスインデックス = 1]、[インターフェイスタイプ = EFA と ENA の組み合わせ] または [EFA のみ] を選択してください。

8. [Storage] (ストレージ) セクションで、必要に応じてボリュームを設定します。

> **Note**
>
> Nvidia CUDA ツールキットには追加の 10 ~ 20 GiB のストレージをプロビジョニングする必要があります。十分な量のストレージをプロビジョニングしないと、Nvidia ドライバーと CUDA ツールキットをインストールしようとしたときに、insufficient disk space エラーが発生します。

9. 右側の [合計mary] (サマリー) パネルで、[Launch instance] (インスタンスの起動) を選択してください。

> **Important**
>
> AMI に Nvidia GPU ドライバー、CUDA ツールキット、cuDNN がすでに含まれている場合、または非 GPU インスタンスを使用している場合は、ステップ 3 をスキップします。

ステップ 3: Nvidia GPU ドライバー、Nvidia CUDA ツールキットおよび cuDNN をインストールする

NVIDIA GPU ドライバー、NVIDIA CUDA ツールキットおよび cuDNN をインストールするには

1. すべてのソフトウェアパッケージが最新の状態であることを確認するため、インスタンスでソフトウェアの更新を実行します。

```
$ sudo apt-get update && sudo apt-get upgrade -y
```

2. Nvidia GPU ドライバと Nvidia CUDA ツールキットをインストールするために必要なユーティリティをインストールします。

```
$ sudo apt-get install build-essential -y
```

3. Nvidia GPU ドライバを使用するにはまず、nouveau オープンソースドライバを無効にする必要があります。

   a. 必要なユーティリティ、および現在実行しているカーネルのバージョン用のカーネルヘッダーパッケージをインストールします。

   ```
   $ sudo apt-get install -y gcc make linux-headers-$(uname -r)
   ```

   b. nouveau 拒否リストファイルに /etc/modprobe.d/blacklist.conf を追加します。

   ```
   $ cat << EOF | sudo tee --append /etc/modprobe.d/blacklist.conf
   blacklist vga16fb
   blacklist nouveau
   blacklist rivafb
   blacklist nvidiafb
   blacklist rivatv
   EOF
   ```

   c. 任意のテキストエディタを使用して /etc/default/grub ファイルを開き、以下を追加します。

   ```
   GRUB_CMDLINE_LINUX="rdblacklist=nouveau"
   ```

   d. Grub 設定を再構築します。

   ```
   $ sudo update-grub
   ```

4. インスタンスを再起動して、そのインスタンスに再接続します。

5. CUDA リポジトリを追加し、Nvidia GPU ドライバー、NVIDIA CUDA ツールキット、および cuDNN をインストールします。

   ```
   $ sudo apt-key adv --fetch-keys http://developer.download.nvidia.com/compute/
   machine-learning/repos/ubuntu2004/x86_64/7fa2af80.pub \
   && wget -O /tmp/deeplearning.deb http://developer.download.nvidia.com/
   compute/machine-learning/repos/ubuntu2004/x86_64/nvidia-machine-learning-repo-
   ubuntu2004_1.0.0-1_amd64.deb \
   && sudo dpkg -i /tmp/deeplearning.deb \
   && wget -O /tmp/cuda.pin https://developer.download.nvidia.com/compute/cuda/repos/
   ubuntu2004/x86_64/cuda-ubuntu2004.pin \
   && sudo mv /tmp/cuda.pin /etc/apt/preferences.d/cuda-repository-pin-600 \
   ```

```
&& sudo apt-key adv --fetch-keys https://developer.download.nvidia.com/compute/
cuda/repos/ubuntu2004/x86_64/3bf863cc.pub \
&& sudo add-apt-repository 'deb http://developer.download.nvidia.com/compute/cuda/
repos/ubuntu2004/x86_64/ /' \
&& sudo apt update \
&& sudo apt install nvidia-dkms-535 \
&& sudo apt install -o Dpkg::Options::='--force-overwrite' cuda-drivers-535 cuda-
toolkit-12-3 libcudnn8 libcudnn8-dev -y
```

6. インスタンスを再起動して、そのインスタンスに再接続します。
7. (p4d.24xlarge および p5.48xlarge のみ) NVIDIA Fabric Manager をインストールします。

   a. 前の手順でインストールした Nvidia カーネルモジュールのバージョンと一致する Nvidia Fabric Manager のバージョンをインストールする必要があります。

      Nvidia カーネルモジュールのバージョンを確認するには次のコマンドを実行します。

      ```
      $ cat /proc/driver/nvidia/version | grep "Kernel Module"
      ```

      以下は出力の例です。

      ```
      NVRM version: NVIDIA UNIX x86_64 Kernel Module  450.42.01  Tue Jun 15 21:26:37
       UTC 2021
      ```

      上記の例ではメジャーバージョン 450 のカーネルモジュールがインストールされました。これはNvidia Fabric Manager のバージョン 450 をインストールする必要があることを意味します。

   b. Nvidia Fabric Manager をインストールする 次のコマンドを、前の手順で識別されたメジャーバージョンを指定して実行します。

      ```
      $ sudo apt install -o Dpkg::Options::='--force-overwrite' nvidia-
      fabricmanager-major_version_number
      ```

      例えば、メジャーバージョン 450 のカーネルモジュールがインストールされた場合、以下のコマンドを使用して、一致するバージョンの Nvidia Fabric Manager をインストールします。

      ```
      $ sudo apt install -o Dpkg::Options::='--force-overwrite' nvidia-
      fabricmanager-450
      ```

c. サービスを開始し、インスタンスの起動時に自動的に起動することを確認します。NVIDIA Fabric Manager はNV Switch Management に必要です。

```
$ sudo systemctl start nvidia-fabricmanager && sudo systemctl enable nvidia-
fabricmanager
```

8. インスタンスが起動するたびに CUDA パスが設定されていることを確認します。

- bash シェルの場合、次のステートメントを /home/*username*/.bashrc と /home/*username*/.bash_profile に追加します。

```
export PATH=/usr/local/cuda/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/cuda/lib64:/usr/local/cuda/extras/CUPTI/lib64:
$LD_LIBRARY_PATH
```

- tcsh シェルの場合、次の文を /home/*username*/.cshrc に追加します。

```
setenv PATH=/usr/local/cuda/bin:$PATH
setenv LD_LIBRARY_PATH=/usr/local/cuda/lib64:/usr/local/cuda/extras/CUPTI/lib64:
$LD_LIBRARY_PATH
```

9. 以下のコマンドを実行して、Nvidia GPU ドライバが機能することを確認します。

```
$ nvidia-smi -q | head
```

このコマンドはNvidia GPU、Nvidia GPU ドライバ、Nvidia CUDA ツールキットの情報を返します。

> **Important**
>
> AMI にすでに GDRCopy が含まれている場合、または非 GPU インスタンスを使用している場合は、ステップ 4 をスキップします。

ステップ 4: GDRCopy をインストールする

GDRCopy をインストールして GPU ベースのプラットフォーム上で Libfabric のパフォーマンスを向上させます。GDRCopy の詳細については「GDRCopy レポジトリ」を参照してください。

GDRCopy をインストールするには

1. 必要な依存ファイルをインストールします。

```
$ sudo apt -y install build-essential devscripts debhelper check libsubunit-dev
 fakeroot pkg-config dkms
```

2. GDRCopy パッケージをダウンロードして解凍します。

```
$ wget https://github.com/NVIDIA/gdrcopy/archive/refs/tags/v2.4.tar.gz \
&& tar xf v2.4.tar.gz \
&& cd gdrcopy-2.4/packages
```

3. GDRCopy DEB パッケージをビルドします。

```
$ CUDA=/usr/local/cuda ./build-deb-packages.sh
```

4. GDRCopy DEB パッケージをインストールします。

```
$ sudo dpkg -i gdrdrv-dkms_2.4-1_amd64.*.deb \
&& sudo dpkg -i libgdrapi_2.4-1_amd64.*.deb \
&& sudo dpkg -i gdrcopy-tests_2.4-1_amd64.*.deb \
&& sudo dpkg -i gdrcopy_2.4-1_amd64.*.deb
```

> **Important**
>
> AMI に最新の EFA インストーラがすでに含まれている場合は、ステップ 5 をスキップします。

ステップ 5: EFA ソフトウェアをインストールする

インスタンスで EFA をサポートするために必要な EFA 対応のカーネル、EFA ドライバー、Libfabric スタックをインストールします。

EFA ソフトウェアをインストールするには

1. 起動したインスタンスに接続します。詳細については「SSH を使用した Linux インスタンスへの接続」を参照してください。

2. EFA ソフトウェアのインストールファイルをダウンロードします。ソフトウェアのインストールファイルは圧縮された tar (.tar.gz) ファイルにパッケージ化されています。次のコマンドを使用して、安定している最新バージョンをダウンロードします。

```
$ curl -O https://efa-installer.amazonaws.com/aws-efa-installer-1.49.0.tar.gz
```

3. 圧縮された .tar.gz ファイルからファイルを展開してターボールを削除し、展開されたディレクトリに移動します。

```
$ tar -xf aws-efa-installer-1.49.0.tar.gz && rm -rf aws-efa-installer-1.49.0.tar.gz
 && cd aws-efa-installer
```

4. (オプション) インストール中に個々のパッケージ署名を検証します。

   EFA インストーラ 1.48.0 以降では、インストーラには GPG 署名の個々の RPM および DEB パッケージが含まれています。インストール中に個々のパッケージの信頼性と整合性を検証するには、--check-signatures フラグを使用します。このフラグを有効にすると、まず、インストーラはすべてのパッケージ署名を検証し、すべてのパッケージが検証に合格した場合にのみインストールを続行します。パッケージが検証に失敗すると、インストーラは何もインストールせずにすぐに終了します。

   a. GPG 公開鍵をダウンロードします。

   ```
   $ wget https://efa-installer.amazonaws.com/aws-efa-installer.key
   ```

   b. キーパスをエクスポートします。次に、次のステップで、インストールコマンドに --check-signatures を追加し、sudo の代わりに sudo -E を使用して環境変数を保持します。

   ```
   $ export EFA_INSTALLER_KEY=$(pwd)/aws-efa-installer.key
   ```

   RPM ベースのシステム (Amazon Linux、RHEL、Rocky Linux、SUSE) では、インストーラは rpm --checksig を使用して各 RPM を検証します。DEB ベースのシステム (Ubuntu、Debian) では、インストーラは GPG 署名検証を使用して各 DEB を検証します。

   パッケージの検証に失敗すると、インストールは直ちに中止され、壊れたパッケージや悪意のあるパッケージからシステムを保護します。

> **Note**
>
> `--check-signatures` フラグはオプションです。これがないと、インストーラは個々の署名検証を実行しません。

5. EFA ソフトウェアのインストールスクリプトを実行します。

> **Note**
>
> 前のオプションステップを完了してパッケージ署名の検証をセットアップした場合は、インストールコマンドに `--check-signatures` を追加して、sudo の代わりに `sudo -E` を使用します。例: `sudo -E ./efa_installer.sh -y --check-signatures`。

```
$ sudo ./efa_installer.sh -y
```

Libfabric は `/opt/amazon/efa` ディレクトリにインストールされます。

6. EFA インストーラーでインスタンスの再起動を求めるメッセージが表示された場合は再起動してからインスタンスに再接続します。それ以外の場合はインスタンスからログアウトし、再度ログインしてインストールを完了します。
7. EFA ソフトウェアコンポーネントが正常にインストールされたことを確認します。

```
$ fi_info -p efa -t FI_EP_RDM
```

コマンドによって、Libfabric の EFA インターフェイスに関する情報が返ります。以下の例はコマンド出力を示しています。

- 単一のネットワークインターフェイスを持つ `p3dn.24xlarge`

```
provider: efa
fabric: EFA-fe80::94:3dff:fe89:1b70
domain: efa_0-rdm
version: 2.0
type: FI_EP_RDM
protocol: FI_PROTO_EFA
```

- 複数のネットワークインターフェイスを持つ p4d.24xlarge および p5.48xlarge

```
provider: efa
fabric: EFA-fe80::c6e:8fff:fef6:e7ff
domain: efa_0-rdm
version: 111.0
type: FI_EP_RDM
protocol: FI_PROTO_EFA
provider: efa
fabric: EFA-fe80::c34:3eff:feb2:3c35
domain: efa_1-rdm
version: 111.0
type: FI_EP_RDM
protocol: FI_PROTO_EFA
provider: efa
fabric: EFA-fe80::c0f:7bff:fe68:a775
domain: efa_2-rdm
version: 111.0
type: FI_EP_RDM
protocol: FI_PROTO_EFA
provider: efa
fabric: EFA-fe80::ca7:b0ff:fea6:5e99
domain: efa_3-rdm
version: 111.0
type: FI_EP_RDM
protocol: FI_PROTO_EFA
```

ステップ 6: NIXL をインストールする

NIXL をインストールします。NIXL の詳細については、NIXL リポジトリを参照してください。

Pre-built distributions

PyPI を使用して NIXL をインストールする方法

1. 必要な依存ファイルをインストールします。

```
$ sudo apt install pip
```

2. NIXL をインストールします。

```
$ pip install nixl
```

Build from source

ソースから NIXL をビルドしてインストールする方法

1. 必要な依存ファイルをインストールします。

```
$ sudo apt install cmake pkg-config meson pybind11-dev libaio-dev nvidia-cuda-
toolkit pip libhwloc-dev \
&& pip install meson ninja pybind11
```

2. ホームディレクトリに移動します。

```
$ cd $HOME
```

3. 公式の NIXL リポジトリをインスタンスにクローンし、ローカルのクローンされたリポジトリに移動します。

```
$ sudo git clone https://github.com/ai-dynamo/nixl.git && cd nixl
```

4. NIXL を構築してインストールし、Libfabric インストールディレクトリへのパスを指定します。

```
$ sudo meson setup . nixl --prefix=/usr/local/nixl -Dlibfabric_path=/opt/amazon/
efa
$ cd nixl && sudo ninja && sudo ninja install
```

ステップ 7: NIXL Benchmark をインストールして EFA と NIXL 設定をテストする

NIXL Benchmark をインストールし、テストを実行して EFA と NIXL に一時インスタンスが適切に設定されていることを確認します。NIXL Benchmark により、NIXL が適切にインストールされていることと、想定どおりに機能していることを確認できます。詳細については、nixlbench リポジトリを参照してください。

NIXL Benchmark (nixlbench) では、クライアントとサーバー間の調整に ETCD が必要です。ETCD と NIXL を併用するには、ETCD サーバーとクライアント、および ETCD CPP API が必要です。

Build from Docker

Docker を使用して NIXL Benchmark をインストールしてテストする方法

1. 公式の NIXL リポジトリのクローンを作成してインスタンスに投入し、nixlbench ビルドディレクトリに移動します。

```
$ git clone https://github.com/ai-dynamo/nixl.git
$ cd nixl/benchmark/nixlbench/contrib
```

2. コンテナを構築します。

```
$ ./build.sh
```

Docker ビルドオプションの詳細については、nixlbench リポジトリを参照してください。

3. Docker をインストールします。

```
$ sudo apt install docker.io -y
```

4. 調整のために ETCD サーバーを起動します。

```
$ docker run -d --name etcd-server \
    -p 2379:2379 -p 2380:2380 \
    quay.io/coreos/etcd:v3.5.18 \
    /usr/local/bin/etcd \
    --data-dir=/etcd-data \
    --listen-client-urls=http://0.0.0.0:2379 \
    --advertise-client-urls=http://0.0.0.0:2379 \
    --listen-peer-urls=http://0.0.0.0:2380 \
    --initial-advertise-peer-urls=http://0.0.0.0:2380 \
    --initial-cluster=default=http://0.0.0.0:2380
```

5. ETCD サーバーが実行されていることを確認します。

```
$ curl -L http://localhost:2379/health
```

正常な出力:

```
{"health":"true"}
```

6. インスタンスの 2 つのターミナルを開きます。両方のターミナルで、以下のコマンドを実行してインストールを確認します。コマンドは、同じインスタンス上の ETCD サーバーを使用し、バックエンドとして Libfabric を使用し、GPU メモリを使用して動作します。

```
$ docker run -it --gpus all --network host nixlbench:latest \
    nixlbench --etcd_endpoints http://localhost:2379 \
    --backend LIBFABRIC \
    --initiator_seg_type VRAM \
    --target_seg_type VRAM
```

> **Note**
>
> GPU 以外のインスタンスには VRAM の代わりに DRAM 値を使用します。

Build from source

> **Important**
>
> ステップ 6 で [ソースからビルドする] を選択した場合のみ、このタブに従います。

NIXL Benchmark をインストールする方法

1. 必要なシステム依存関係をインストールします。

```
$ sudo apt install libgflags-dev
```

2. ETCD サーバーとクライアントをインストールします。

```
$ sudo apt install -y etcd-server etcd-client
```

3. ETCD CPP API をインストールします。

   a. ETCD CPP API に必要な依存関係をインストールします。

```
$ sudo apt install libboost-all-dev libssl-dev libgrpc-dev libgrpc++-dev
 libprotobuf-dev protobuf-compiler-grpc libcpprest-dev
```

   b. ETCD CPP API のクローンを作成してインストールします。

```
$ cd $HOME
$ git clone https://github.com/etcd-cpp-apiv3/etcd-cpp-apiv3.git
$ cd etcd-cpp-apiv3
$ mkdir build && cd build
$ cmake ..
$ sudo make -j$(nproc) && sudo make install
```

4. nixlbench をビルドしてインストールします。

```
$ sudo meson setup . $HOME/nixl/benchmark/nixlbench -Dnixl_path=/usr/local/nixl/
$ sudo ninja && sudo ninja install
```

EFA と NIXL 設定をテストする方法

1. インスタンスで ETCD サーバーを起動します。

```
$ etcd --listen-client-urls "http://0.0.0.0:2379" \
    --advertise-client-urls "http://localhost:2379" &
```

2. ETCD サーバーが実行されていることを確認します。

```
$ curl -L http://localhost:2379/health
```

正常な出力:

```
{"health":"true"}
```

3. インスタンスの 2 つのターミナルを開きます。両方のターミナルで、以下の手順に従って nixlbench を実行します。

   a. nixlbench がインストールされているディレクトリに移動します。

   ```
   $ cd /usr/local/nixlbench/bin/
   ```

   b. テストを実行し、バックエンド、ETCD サーバーのアドレス、イニシエータセグメントタイプを指定します。次のコマンドは、同じインスタンス上の ETCD サーバーを使用し、バックエンドとして Libfabric を使用し、GPU メモリを使用して動作します。環境変数により以下が設定されます。

- NIXL_LOG_LEVEL=INFO — 詳細なデバッグ出力を有効にします。エラーメッセージのみを受信するように WARN を指定することもできます。
- LD_LIBRARY_PATH — NIXL ライブラリのパスを設定します。

NIXL Benchmark 引数の詳細については、公式の nixlbench リポジトリの NIXLbench README を参照してください。

```
$ export NIXL_LOG_LEVEL=INFO
$ export LD_LIBRARY_PATH=/usr/local/nixl/lib/$(gcc -dumpmachine):
$LD_LIBRARY_PATH

$ nixlbench --etcd-endpoints 'http://localhost:2379' \
    --backend 'LIBFABRIC' \
    --initiator_seg_type 'VRAM' \
    --target_seg_type 'VRAM'
```

> **Note**
>
> GPU 以外のインスタンスには VRAM の代わりに DRAM 値を使用します。

ステップ 8: 機械学習アプリケーションをインストールする

機械学習アプリケーションを一時インスタンスにインストールします。インストール手順はそれぞれの機械学習アプリケーションによって異なります。

> **Note**
>
> インストール手順については機械学習アプリケーションのドキュメントを参照してください。

ステップ 9: EFA および NIXL 対応 AMI を作成する

必要なソフトウェアコンポーネントのインストール後、EFA 対応のインスタンスの起動に再利用できる AMI を作成します。

一時インスタンスから AMI を作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. 作成した一時インスタンスを選択し、[アクション]、[イメージ]、[イメージの作成] の順に選択してください。
4. [イメージの作成] で、次を行います。
   a. [イメージ名] に、の分かりやすい AMI 名を入力してください。
   b. (オプション) [イメージの説明] に、AMI の簡単な説明を入力してください。
   c. [イメージを作成] を選択してください。
5. ナビゲーションペインで [AMI] を選択してください。
6. リストで作成した AMI を探します。ステータスが `pending` から `available` に変わるまで待ってから、次のステップに進みます。

ステップ 10: 一時インスタンスを終了する

この時点で、起動した一時インスタンスは不要になります。インスタンスを終了して、料金の発生を停止できます。

一時インスタンスを終了するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. 作成した一時インスタンスを選択し、[アクション]、[インスタンスの状態]、[インスタンスの終了] の順に選択してください。
4. 確認を求めるメッセージが表示されたら、[終了] を選択してください。

ステップ 11: EFA および NIXL 対応インスタンスを起動する

ステップ 9 で作成した EFA 対応の AMI と、ステップ 1 で作成した EFA 対応のセキュリティグループを使用して、EFA および NIXL 対応インスタンスを起動します。

EFA および NIXL 対応インスタンスを起動する方法

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

2. ナビゲーションペインで、[Instances] (インスタンス) を選択し、[Launch Instances] (インスタンスの起動) を選択して、新しいインスタンス起動ウィザードを開きます。
3. (オプション) [Name and tags] (名前とタグ) セクションで、`EFA-instance` などのインスタンス名を指定します。指定した名前はリソースタグとしてインスタンスに割り当てられます (Name=*`EFA-instance`*)。
4. [Application and OS Images] (アプリケーションと OS イメージ) セクションで、[My AMI] (マイ AMI) をクリックし、前のステップで作成した AMI を選択してください。
5. [インスタンスタイプ] セクションで、サポートされているインスタンスタイプを選択してください。
6. [Key pair] (キーペア) セクションで、インスタンスに使用するキーペアを選択してください。
7. [Network settings] (ネットワーク設定) セクションで、[Edit] (編集) を選択し、次の操作を行います。

   a. [サブネット] で、インスタンスを起動するサブネットを選択してください。サブネットを選択しない場合、EFA のインスタンスを有効にすることはできません。

   b. [ファイアウォール (セキュリティグループ)] の場合、[既存のセキュリティグループを選択] を選択し、ステップ 1で作成したセキュリティグループを選択してください。

   c. [高度な設定] セクションを展開します。

      ネットワークインターフェイス 1 では[ネットワークカードインデックス = 0]、[デバイスインデックス = 0]、[インターフェイスタイプ = EFA と ENA の組み合わせ] を選択してください。

      (オプション) `p4d.24xlarge` または `p5.48xlarge` などのマルチカードインスタンスタイプを、必要な追加のネットワークインターフェイスごとに使用する場合は[ネットワークインターフェイスの追加] を選択し、ネットワークカードインデックスの場合は次の未使用インデックスを選択してから、[デバイスインデックス = 1]、[インターフェイスタイプ = EFA と ENA の組み合わせ] または [EFA のみ] を選択してください。
8. (オプション) [Storage] (ストレージ) セクションで、必要に応じてボリュームを設定します。
9. 右側の [合計mary] (サマリー) パネルで、[Number of instances] (インスタンス数) に、起動する EFA 対応のインスタンスの数を入力し、[Launch Instance] (インスタンスの起動) を選択してください。

ステップ 12: パスワードレス SSH を有効にする

クラスター内のすべてのインスタンスでアプリケーションを実行できるようにするにはリーダーノードからメンバーノードへのパスワードなしの SSH アクセスを有効にする必要があります。リーダーノードはアプリケーションを実行するインスタンスです。クラスター内の残りのインスタンスはメンバーノードです。

クラスター内のインスタンス間でパスワードなしの SSH を有効にするには

1. クラスター内の 1 つのインスタンスをリーダーノードとして選択し、それに接続します。
2. リーダーノード上で strictHostKeyChecking を無効にし ForwardAgent を有効にします。任意のテキストエディタを使用して ~/.ssh/config ファイルを開き、以下を追加します。

```
Host *
    ForwardAgent yes
Host *
    StrictHostKeyChecking no
```

3. RSA キーペアを生成します。

```
$ ssh-keygen -t rsa -N "" -f ~/.ssh/id_rsa
```

キーペアは$HOME/.ssh/ ディレクトリで作成されます。

4. リーダーノードのプライベートキーの許可を変更します。

```
$ chmod 600 ~/.ssh/id_rsa
chmod 600 ~/.ssh/config
```

5. 任意のテキストエディタで ~/.ssh/id_rsa.pub を開き、キーをコピーします。
6. クラスター内の各メンバーノードについて、次の操作を行います。
   a. インスタンスに接続します。
   b. 任意のテキストエディタで ~/.ssh/authorized_keys を開き、前にコピーしたパブリックキーを追加します。
7. パスワードレス SSH が正常に機能していることをテストするにはリーダーノードに接続して、次のコマンドを実行します。

```
$ ssh member_node_private_ip
```

キーまたはパスワードの入力を求められずに、メンバーノードに接続できるはずです。

> **Important**
>
> ステップ 7 に従った場合にのみ、ステップ 13 に従ってください。

ステップ 13: インスタンス間で EFA と NIXL の設定をテストする

テストを実行し、EFA と NIXL に対して一時インスタンスが適切に設定されていることを確認します。

Build from Docker

Docker を使用してインスタンス間で EFA と NIXL の設定をテストする方法

1. nixlbench ベンチマークを実行するためにホストを 2 つ選択します。最初のホストの IP アドレスをメタデータ交換用の ETCD サーバー IP として使用します。
2. ホスト 1 で ETCD サーバーを起動します。

```
$ docker run -d --name etcd-server \
    -p 2379:2379 -p 2380:2380 \
    quay.io/coreos/etcd:v3.5.18 \
    /usr/local/bin/etcd \
    --data-dir=/etcd-data \
    --listen-client-urls=http://0.0.0.0:2379 \
    --advertise-client-urls=http://0.0.0.0:2379 \
    --listen-peer-urls=http://0.0.0.0:2380 \
    --initial-advertise-peer-urls=http://0.0.0.0:2380 \
    --initial-cluster=default=http://0.0.0.0:2380
```

3. ETCD サーバーが実行されていることを確認します。

```
$ curl -L http://localhost:2379/health
```

```
{"health":"true"}
```

4. ホスト 1 で nixlbench ベンチマークを実行します。

```
$ docker run -it --gpus all --network host nixlbench:latest \
    nixlbench --etcd_endpoints http://localhost:2379 \
    --backend LIBFABRIC \
    --initiator_seg_type VRAM
```

5. ホスト 2 で nixlbench ベンチマークを実行します。

```
$ docker run -it --gpus all --network host nixlbench:latest \
    nixlbench --etcd_endpoints http://ETCD_SERVER_IP:2379 \
    --backend LIBFABRIC \
    --initiator_seg_type VRAM
```

Build from source

> **Important**
>
> ステップ 6 で [ソースからビルドする] を選択した場合のみ、このタブに従います。

インスタンス間で EFA と NIXL の設定をテストする方法

1. nixlbench ベンチマークを実行するためにホストを 2 つ選択します。最初のホストの IP アドレスをメタデータ交換用の ETCD サーバー IP として使用します。
2. ホスト 1 で ETCD サーバーを起動します。

```
$ etcd --listen-client-urls "http://0.0.0.0:2379" \
    --advertise-client-urls "http://localhost:2379" &
```

3. ETCD サーバーが実行されていることを確認します。

```
$ curl -L http://localhost:2379/health
```

```
{"health":"true"}
```

4. ホスト 1 で nixlbench ベンチマークを実行します。

```
$ export NIXL_LOG_LEVEL=INFO
$ export LD_LIBRARY_PATH=$HOME/nixl/lib/x86_64-linux-gnu:$LD_LIBRARY_PATH
```

```
$ nixlbench \
    --etcd-endpoints http://localhost:2379 \
    --backend LIBFABRIC \
    --initiator_seg_type VRAM
```

5. ホスト 2 で nixlbench ベンチマークを実行します。

```
$ export NIXL_LOG_LEVEL=INFO
$ export LD_LIBRARY_PATH=$HOME/nixl/lib/x86_64-linux-gnu:$LD_LIBRARY_PATH

$ nixlbench \
    --etcd-endpoints http://ETCD_SERVER_IP:2379 \
    --backend LIBFABRIC \
    --initiator_seg_type VRAM
```

ステップ 14: vLLM で提供される分散推論をテストする (オプション)

NIXL をインストールしたら、LLM 推論を通じて NIXL を使用すると、vLLM、SGLang、TensorRT-LLM などのフレームワークを提供できます。

vLLM を使用して推論ワークロードを提供する方法

1. vLLM をインストールします。

```
$ pip install vllm
```

2. NIXL を使用して vLLM サーバーを起動します。次のサンプルコマンドにより、NIXL ハンドシェイク接続、KV コネクタ、KV ロール、トランスポートバックエンド用として、プリフィル (プロデューサー) インスタンスが 1 つとデコード (コンシューマー) インスタンスが 1 つ作成されます。詳細な例とスクリプトについては、「NIXLConnector 使用ガイド」を参照してください。

   NIXL を EFA とともに使用するには、セットアップとユースケースに基づいて環境変数を設定します。

   - プロデューサー (プリフィラー) の設定

```
$ vllm serve your-application \
    --port 8200 \
    --enforce-eager \
```

```
    --kv-transfer-config
 '{"kv_connector":"NixlConnector","kv_role":"kv_both","kv_buffer_device":"cuda","kv_conne
{"backends":["LIBFABRIC"]}}'
```

- コンシューマー (デコーダー) の設定

```
$ vllm serve your-application \
    --port 8200 \
    --enforce-eager \
    --kv-transfer-config
 '{"kv_connector":"NixlConnector","kv_role":"kv_both","kv_buffer_device":"cuda","kv_conne
{"backends":["LIBFABRIC"]}}'
```

前述のサンプル設定では、以下を設定します。

- kv_role を kv_both に設定すると、コネクタがプロデューサーとコンシューマーの両方として機能する対称機能が有効になります。これにより、実験的なセットアップや、ロールの区別が事前に決定されていないシナリオに柔軟性が生まれます。
- kv_buffer_device を cuda に設定すると、GPU メモリの使用が可能になります。
- NIXL バックエンドを LIBFABRIC に設定すると、NIXL トラフィックが EFA を通過できるようになります。

複数のネットワークカードを使用して Amazon EC2 インスタンスのネットワーク帯域幅を最大化する

EFA をサポートする多くのインスタンスタイプには複数のネットワークカードもあります。詳細については「ネットワークカード」を参照してください。これらのインスタンスタイプのいずれかで EFA を使用する予定がある場合は次の基本設定をお勧めします。

- プライマリネットワークインターフェイス (ネットワークカードインデックス 0、デバイスインデックス 0) には、EFA インターフェイスを作成します。EFA のみのネットワークインターフェイスをプライマリネットワークインターフェイスとして使用することはできません。
- ネットワークカードインデックス 0 が EFA をサポートする場合は、ネットワークカードインデックス 0、デバイスインデックス 1 に EFA 専用ネットワークインターフェイスを作成します。
- 追加される各ネットワークインターフェイスについては、ユースケース (ENA 帯域幅要件や IP アドレス空間など) に基づいて、EFA 専用ネットワークインターフェイスに次の未使用のネットワークカードインデックス、デバイスインデックス 0、および/または EFA ネットワークインターフェ

イスにデバイスインデックス 1 を使用します。ユースケース例については「P5 および P5e インスタンスの EFA 設定」を参照してください。

> **Note**
>
> P5 インスタンスでは最大ネットワーク帯域幅を有効にするために、特定の方法でネットワークインターフェイスを設定する必要があります。詳細については「P5 および P5e インスタンスの EFA 設定」を参照してください。

次の例はこれらの推奨事項に基づいてインスタンスを起動する方法を示しています。

Instance launch

インスタンス起動ウィザードを使用したインスタンスの起動時に EFA を指定するには

1. [ネットワーク設定] セクションで、[編集] をクリックしてください。
2. 高度なネットワーク設定の拡張。
3. プライマリネットワークインターフェイス (ネットワークカードインデックス 0、デバイスインデックス 0) には、EFA インターフェイスを作成します。EFA のみのネットワークインターフェイスをプライマリネットワークインターフェイスとして使用することはできません。
4. ネットワークカードインデックス 0 が EFA をサポートする場合は、ネットワークカードインデックス 0、デバイスインデックス 1 に EFA 専用ネットワークインターフェイスを作成します。
5. 追加される各ネットワークインターフェイスについては、ユースケース (ENA 帯域幅要件や IP アドレス空間など) に基づいて、EFA 専用ネットワークインターフェイスに次の未使用のネットワークカードインデックス、デバイスインデックス 0、および/または EFA ネットワークインターフェイスにデバイスインデックス 1 を使用します。ユースケース例については「P5 および P5e インスタンスの EFA 設定」を参照してください。

[run-instances] コマンドを使用してインスタンスの起動中に EFA を指定するには

`--network-interfaces` では必要な数のネットワークインターフェイスを指定します。プライマリネットワークインターフェイスに `NetworkCardIndex=0`、`DeviceIndex=0`、`InterfaceType=interface` を指定します。ネットワークカードインデックス 0 が EFA をサポートする場合

は、NetworkCardIndex=0、DeviceIndex=1、および InterfaceType=efa-only を指定します。追加のネットワークインターフェイスについては、NetworkCardIndex に次の未使用インデックス、InterfaceType=efa-only に DeviceIndex=0、および/または InterfaceType=interface に DeviceIndex=1 を指定してください。

次のコマンドスニペットの例は32 の EFA デバイスおよび 1 つの ENA デバイスによるリクエストを示しています。

```
$ aws ec2 run-instances \
 --instance-type p5.48xlarge \
 --count 1 \
 --key-name key_pair_name \
 --image-id ami-0abcdef1234567890 \
 --network-interfaces
 "NetworkCardIndex=0,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
 \
"NetworkCardIndex=0,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=
only" \
"NetworkCardIndex=1,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=
only" \
"NetworkCardIndex=2,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=
only" \
"NetworkCardIndex=3,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=
only" \
"NetworkCardIndex=4,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=
only" \
"NetworkCardIndex=5,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=
only" \
"NetworkCardIndex=6,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=
only" \
"NetworkCardIndex=7,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=
only" \
"NetworkCardIndex=8,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=
only" \
"NetworkCardIndex=9,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=
only" \
"NetworkCardIndex=10,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
"NetworkCardIndex=11,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
"NetworkCardIndex=12,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
```

```
"NetworkCardIndex=13,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
"NetworkCardIndex=14,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
"NetworkCardIndex=15,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
"NetworkCardIndex=16,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
"NetworkCardIndex=17,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
"NetworkCardIndex=18,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
"NetworkCardIndex=19,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
"NetworkCardIndex=20,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
"NetworkCardIndex=21,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
"NetworkCardIndex=22,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
"NetworkCardIndex=23,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
"NetworkCardIndex=24,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
"NetworkCardIndex=25,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
"NetworkCardIndex=26,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
"NetworkCardIndex=27,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
"NetworkCardIndex=28,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
"NetworkCardIndex=29,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
"NetworkCardIndex=30,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only" \
"NetworkCardIndex=31,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType
only"
...
```

Launch templates

Amazon EC2 コンソールを使用して起動テンプレートに EFA を追加するには

1. [ネットワーク設定] セクションで [高度なネットワーク設定] を展開します。
2. プライマリネットワークインターフェイス (ネットワークカードインデックス 0、デバイスインデックス 0) には、EFA インターフェイスを作成します。EFA のみのネットワークインターフェイスをプライマリネットワークインターフェイスとして使用することはできません。
3. ネットワークカードインデックス 0 が EFA をサポートする場合は、ネットワークカードインデックス 0、デバイスインデックス 1 に EFA 専用ネットワークインターフェイスを作成します。
4. 追加される各ネットワークインターフェイスについては、ユースケース (ENA 帯域幅要件や IP アドレス空間など) に基づいて、EFA 専用ネットワークインターフェイスに次の未使用のネットワークカードインデックス、デバイスインデックス 0、および/または EFA ネットワークインターフェイスにデバイスインデックス 1 を使用します。ユースケース例については「P5 および P5e インスタンスの EFA 設定」を参照してください。

[create-launch-template] コマンドを使用して起動テンプレートに EFA を追加するには

NetworkInterfaces では必要な数のネットワークインターフェイスを指定します。プライマリネットワークインターフェイスに NetworkCardIndex=0、DeviceIndex=0、InterfaceType=interface を指定します。ネットワークカードインデックス 0 が EFA をサポートする場合は、NetworkCardIndex=0、DeviceIndex=1、および InterfaceType=efa-only を指定します。追加のネットワークインターフェイスについては、NetworkCardIndex に次の未使用インデックス、InterfaceType=efa-only に DeviceIndex=0、および/または InterfaceType=interface に DeviceIndex=1 を指定してください。

次のスニペットは設定可能な 32 のネットワークインターフェイスのうち 3 つのネットワークインターフェイスを使用した例を示しています。

```
"NetworkInterfaces":[
{
  "NetworkCardIndex":0,
  "DeviceIndex":0,
  "InterfaceType": "interface",
  "AssociatePublicIpAddress":false,
```

```
  "Groups":[
    "security_group_id"
  ],
  "DeleteOnTermination":true
},
{
  "NetworkCardIndex": 0,
  "DeviceIndex": 1,
  "InterfaceType": "efa-only",
  "AssociatePublicIpAddress":false,
  "Groups":[
    "security_group_id"
  ],
  "DeleteOnTermination":true
},
{
  "NetworkCardIndex": 1,
  "DeviceIndex": 0,
  "InterfaceType": "efa-only",
  "AssociatePublicIpAddress":false,
  "Groups":[
    "security_group_id"
  ],
  "DeleteOnTermination":true
},
{
  "NetworkCardIndex": 2,
  "DeviceIndex": 0,
  "InterfaceType": "efa-only",
  "AssociatePublicIpAddress":false,
  "Groups":[
    "security_group_id"
  ],
  "DeleteOnTermination":true
},
{
  "NetworkCardIndex": 3,
  "DeviceIndex": 0,
  "InterfaceType": "efa-only",
  "AssociatePublicIpAddress":false,
  "Groups":[
    "security_group_id"
  ],
  "DeleteOnTermination":true
```

```
}
...
```

P5 および P5e インスタンスの EFA 設定

`p5.48xlarge` および `p5e.48xlarge` インスタンスは 32 個のネットワークカードをサポートします。ネットワーク帯域幅の合計キャパシティは 3,200 Gbps で、そのうち最大 800 Gbps を IP ネットワークトラフィックに使用できます。EFA ネットワークトラフィックと IP ネットワークトラフィックは同じ基盤となるリソースを共有するため、一方が使用する帯域幅は他方が利用できる帯域幅を減らします。つまり、総帯域幅が 3,200 Gbps を超えず、IP 帯域幅が 800 Gbps を超えない限り、EFA トラフィックと IP トラフィックの間でネットワーク帯域幅を任意の組み合わせで分散できます。例えば、IP 帯域幅で 400 Gbps を使用する場合、同時に最大 2,800 Gbps の EFA 帯域幅を実現できます。

ユースケース 1: IP アドレスを保存し、潜在的な Linux IP の問題を回避する

この設定では最大 3200 Gbps の EFA ネットワーク帯域幅と最大 100 Gbps の IP ネットワーク帯域幅を 1 つのプライベート IP アドレスで提供します。また、この設定はパブリック IP アドレスの自動割り当ての禁止や、インスタンスに複数のネットワークインターフェイスがある場合に発生する可能性のある IP ルーティングの課題 (ホスト名から IP アドレスへのマッピングの問題や送信元 IP アドレスの不一致) など、Linux IP の潜在的な問題を回避するのに役立ちます。

- プライマリネットワークインターフェイス (ネットワークカードインデックス 0、デバイスインデックス 0) には、ENA インターフェイスを使用します。
- ネットワークカードインデックス 0、デバイスインデックス 1 には、EFA 専用ネットワークインターフェイスを作成します。
- 残りのネットワークインターフェイス (ネットワークカードインデックス 1～31、デバイスインデックス 0) には、EFA 専用ネットワークインターフェイスを使用します。

ユースケース 2: 最大 EFA と IP ネットワーク帯域幅

この設定では最大 3200 Gbps の EFA ネットワーク帯域幅と最大 800 Gbps の IP ネットワーク帯域幅を 8 つのプライベート IP アドレスで提供します。この設定ではパブリック IP アドレスを自動割り当てすることはできません。ただし、インターネット接続の起動後に、Elastic IP アドレスをプライマリネットワークインターフェイス (ネットワークカードインデックス 0、デバイスインデックス 0) にアタッチすることはできます。

- プライマリネットワークインターフェイス (ネットワークカードインデックス 0、デバイスインデックス 0) には ENA ネットワークインターフェイスを使用します。
- 残りのインターフェイスについては次の手順を実行します。
  - ネットワークカードインデックス 0、デバイスインデックス 1 に EFA 専用ネットワークインターフェイスを指定し、デバイスインデックス 1、2、3 にはデバイスインデックス 0 を使用します。
  - 次のネットワークカードインデックスサブセットそれぞれに 1 つの ENA ネットワークインターフェイスと 4 つの EFA 専用ネットワークインターフェイスを指定し、ENA ネットワークインターフェイスにはデバイスインデックス 1、EFA 専用ネットワークインターフェイスにはデバイスインデックス 0 を使用します。
    - [4,5,6,7]
    - [8,9,10,11]
    - [12,13,14,15]
    - [16,17,18,19]
    - [20,21,22,23]
    - [24,25,26,27]
    - [28,29,30,31]

次の例ではこの設定を示します。

```
$ aws --region $REGION ec2 run-instances \
 --instance-type p5.48xlarge \
 --count 1 \
 --key-name key_pair_name \
 --image-id ami_id \
 --network-interfaces
 "NetworkCardIndex=0,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=in
 \
"NetworkCardIndex=0,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=1,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=2,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=3,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
```

```
"NetworkCardIndex=4,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=int
 \
"NetworkCardIndex=4,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=5,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=6,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=7,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=8,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=int
 \
"NetworkCardIndex=8,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=9,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=10,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=11,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=12,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=in
 \
"NetworkCardIndex=12,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=13,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=14,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=15,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=16,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=in
 \
"NetworkCardIndex=16,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=17,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=18,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=19,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=20,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=in
 \
"NetworkCardIndex=20,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
```

```
"NetworkCardIndex=21,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=22,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=23,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=24,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=in
 \
"NetworkCardIndex=24,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=25,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=26,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=27,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=28,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=in
 \
"NetworkCardIndex=28,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=29,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=30,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=31,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only"
...
```

P6-B200 インスタンスの EFA 設定

P6-B200 インスタンスのネットワーク帯域幅の総量は 3,200 Gbps であり、そのうち最大 1600 Gbps を ENA に使用できます。8 つの GPU と 8 つのネットワークカードがあり、各ネットワークカードは最大 400 Gbps の EFA 帯域幅と 200 Gbp の ENA 帯域幅をサポートします。EFA ネットワークトラフィックと ENA トラフィックは同じ基盤となるリソースを共有するため、一方で帯域幅を使用していれば、もう一方で使用できる帯域幅が減少します。

ユースケース 1: IP アドレスを保存する

この設定では、インスタンスごとに 1 個以上のプライベート IP アドレスを使用し、最大 3200 Gbp の EFA 帯域幅と 200 Gbp の ENA 帯域幅をサポートします。

- プライマリネットワークインターフェイス (ネットワークカードインデックス 0、デバイスインデックス 0) には、ENA インターフェイスを使用します。

- ネットワークカードインデックス 0、デバイスインデックス 1 には、EFA 専用ネットワークインターフェイスを作成します。
- 残りの 7 個のネットワークカード (ネットワークカードインデックス 1～7、デバイスインデックス 0) には、EFA 専用のネットワークインターフェイスを使用します。

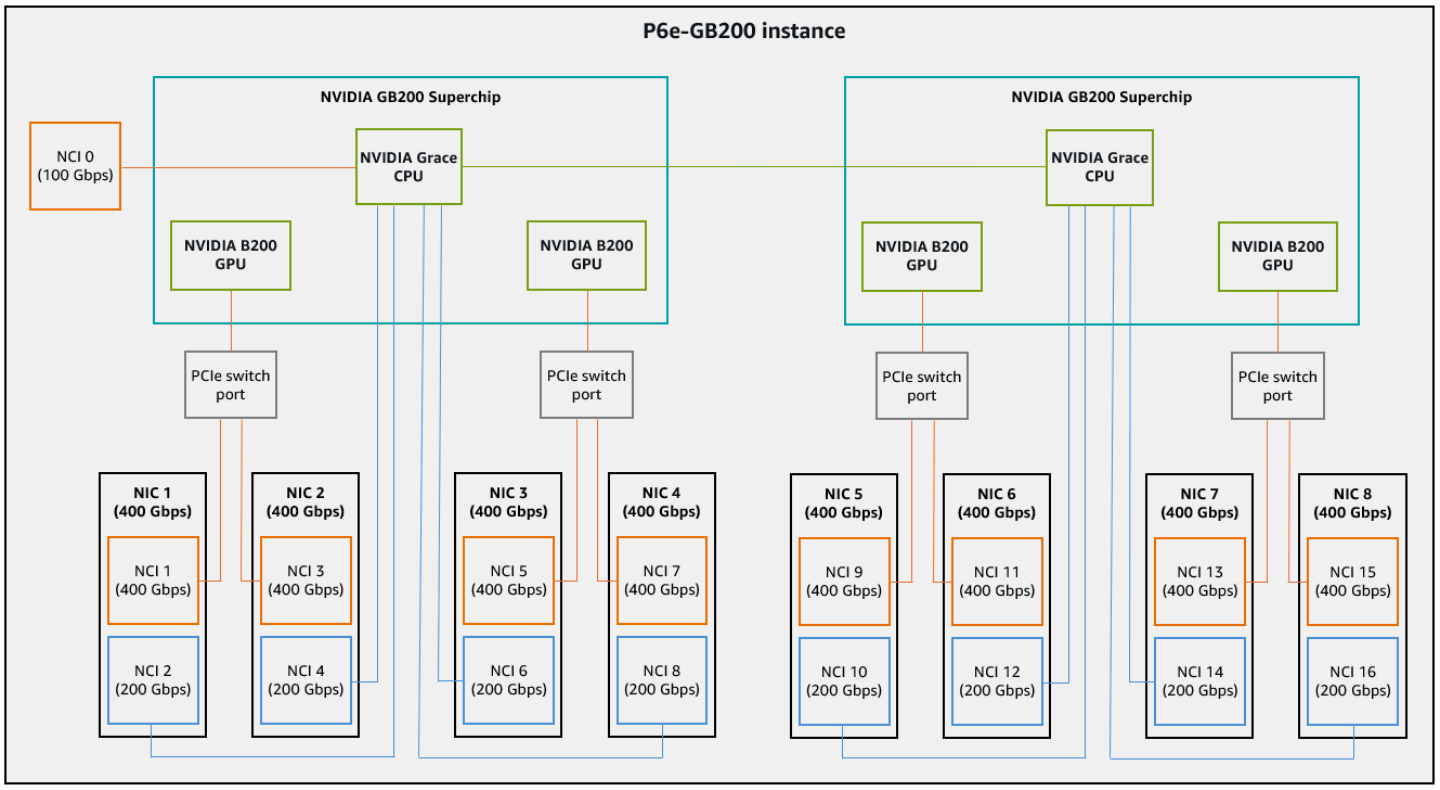
ユースケース 2: 最大 EFA と EFA 帯域幅

この設定では、インスタンスごとに 8 個以上のプライベート IP アドレスを使用し、最大 3200 Gbp の EFA 帯域幅と 1600 Gbp の ENA 帯域幅をサポートします。

- プライマリネットワークインターフェイス (ネットワークカードインデックス 0、デバイスインデックス 0) には、ENA インターフェイスを使用します。
- ネットワークカードインデックス 0、デバイスインデックス 1 には、EFA 専用ネットワークインターフェイスを作成します。
- 残りの 7 個のネットワークカード (ネットワークカードインデックス 1～7) については、デバイスインデックス 0 で EFA 専用インターフェイスを作成し、デバイスインデックス 1 で ENA ネットワークインターフェイスを作成します。

P6e-GB200 インスタンスの EFA 設定

P6e-GB200 インスタンスは、最大 17 個のネットワークカードで設定できます。次の図は、P6e-GB200 インスタンスの物理ネットワークインターフェイスカード (NIC) のレイアウトと、ネットワークカードインデックス (NCI) のマッピングを示しています。



プライマリ NCI (インデックス 0) は、最大 100 Gbps の ENA 帯域幅をサポートします。次のインデックスを持つ NCI は、EFA 専用ネットワークインターフェイスと 400 Gbps の EFA 帯域幅をサポートします: [1、3、5、7、9、11、13、15]。次のインデックスを持つ NCI は、最大 200 Gbps の ENA または EFA 帯域幅をサポートします: [2、4、6、8、10、12、14、16]。

次のグループの NCI は、ホスト上の基盤となる物理 NIC を共有します:

- [1 および 2]
- [3 および 4]
- [5 および 6]
- [7 および 8]
- [9 および 10]
- [11 および 12]
- [13 および 14]
- [15 および 16]

各物理 NIC は、最大 400 Gbps の帯域幅をサポートします。これらのグループの NCI は基盤となる同じ物理 NIC を共有するため、1 つが帯域幅を使用すると、他が利用できる帯域幅が減少します。

例えば、NCI 2 が 200 Gbps の ENA 帯域幅を使用する場合、NCI 1 は同時に最大 200 Gbps の EFA 帯域幅を使用できます。

ホスト上の基盤となる各 GPU は、次の NCI ペアを介して直接トラフィックを送信できます。

- [1 および 3]
- [5 および 7]
- [9 および 11]
- [13 および 15]

各 GPU は、最大 400 Gbps の EFA 帯域幅をサポートします。これらのグループのネットワークカードは同じ GPU を共有するため、1 つが帯域幅を使用すると、他が利用できる帯域幅が減少します。例えば、NCI 1 が 200 Gbps の EFA 帯域幅を使用する場合、NCI 3 は同時に最大 200 Gbps の EFA 帯域幅を使用できます。したがって、EFA の最大パフォーマンスを実現するには、次のいずれかを実行して合計 1,600 Gbps の EFA 帯域幅を達成することをお勧めします。

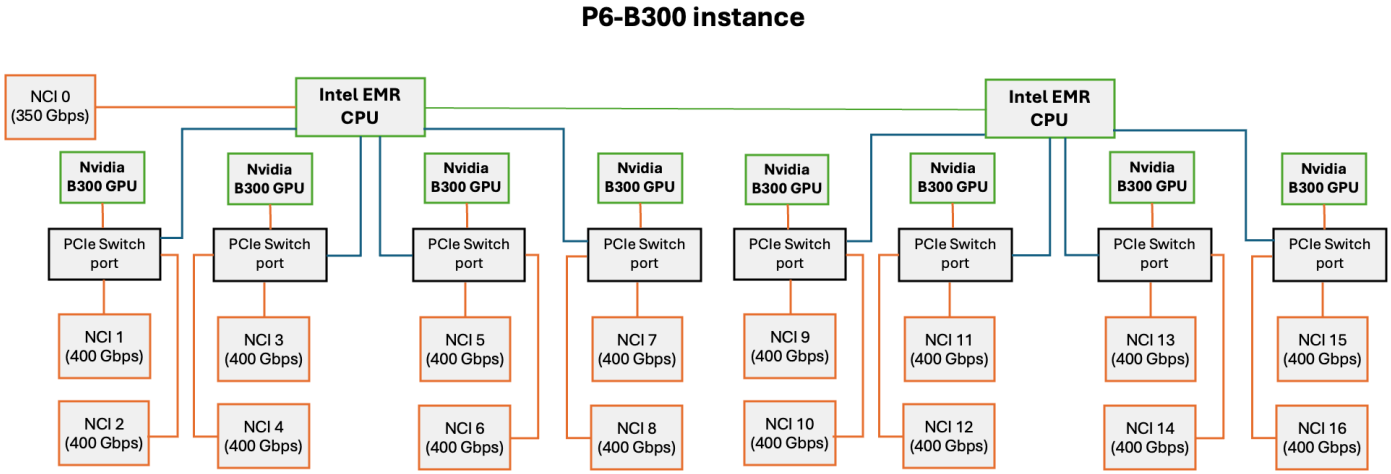
- EFA 専用ネットワークインターフェイスを各グループの 1 つの NCI のみに追加して、ネットワークインターフェイスあたり 400 Gbps (4 つの EFA ネットワークインターフェイス x 400 Gbps) を実現します。
- EFA 専用ネットワークインターフェイスを各グループの各 NCI に追加して、ネットワークインターフェイスあたり 200 Gbps (8 つの EFA ネットワークインターフェイス x 200 Gbps) を実現します。

例えば、次の設定では、各 NCI グループで単一の EFA 専用ネットワークインターフェイスを使用して最大 1,600 Gbps の EFA 帯域幅を提供し、プライマリ NCI (インデックス 0) のみを使用して最大 100 Gbps の ENA ネットワーク帯域幅を提供します。

- プライマリ NCI (ネットワークカードインデックス 0、デバイスインデックス 0) には ENA ネットワークインターフェイスを使用します。
- EFA 専用ネットワークインターフェイスを以下に追加します。
  - NCI 1、デバイスインデックス 0
  - NCI 5、デバイスインデックス 0
  - NCI 9、デバイスインデックス 0
  - NCI 13、デバイスインデックス 0

P6-B300 インスタンスの EFA 設定

P6-B300 インスタンスの合計ネットワーク帯域幅キャパシティは、EFA トラフィックでは最大 6,400 Gbps、ENA トラフィックでは最大 3,870 Gbps です。これらのインスタンスには 8 の GPU と 17 のネットワークカードがあり、プライマリネットワークカードは最大 350 Gbps の帯域幅を持つ ENA ネットワークインターフェイスのみをサポートします。セカンダリネットワークカードは、最大 400 Gbps の EFA と最大 220 Gbps の ENA 帯域幅をサポートします。EFA ネットワークトラフィックと ENA トラフィックは同じ基盤となるリソースを共有するため、一方で帯域幅を使用していれば、もう一方で使用できる帯域幅が減少します。



ユースケース 1: IP アドレスを保存する

この設定では、インスタンスごとに 1 つ以上のプライベート IP アドレスを使用し、最大 6,400 Gbp の EFA 帯域幅と最大 350 Gbp の ENA 帯域幅をサポートします。

- プライマリネットワークインターフェイス (ネットワークカードインデックス 0、デバイスインデックス 0) には ENA ネットワークインターフェイスを使用します。
- 残りのネットワークカード (ネットワークカードインデックス 1 ~ 16、デバイスインデックス 0) には、EFA 専用ネットワークインターフェイスを使用します。

```
--network-interfaces \
"NetworkCardIndex=0,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=int
 \
"NetworkCardIndex=1,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=2,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
```

```
"NetworkCardIndex=3,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=4,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=5,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=6,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=7,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=8,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=9,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=10,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=11,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=12,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=13,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=14,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=15,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=16,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only"
```

ユースケース 2: 最大 EFA と EFA 帯域幅

この設定では、インスタンスごとに 17 以上のプライベート IP アドレスを使用し、最大 6,400 Gbp の EFA 帯域幅と最大 3,870 Gbp の ENA 帯域幅をサポートします。

- プライマリネットワークインターフェイス (ネットワークカードインデックス 0、デバイスインデックス 0) では、ENA ネットワークインターフェイスを使用します。
- 残りのネットワークカードについては、EFA 専用インターフェイス (ネットワークカードインデックス 1 ~ 16、デバイスインデックス 0) と ENA インターフェイス (ネットワークカードインデックス 1 ~ 16、デバイスインデックス 1) を作成します。

```
--network-interfaces \
```

```
"NetworkCardIndex=0,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=int
 \
"NetworkCardIndex=1,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=2,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=3,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=4,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=5,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=6,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=7,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=8,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=9,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=efa
only" \
"NetworkCardIndex=10,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=11,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=12,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=13,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=14,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=15,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=16,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
only" \
"NetworkCardIndex=1,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=int
 \
"NetworkCardIndex=2,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=int
 \
"NetworkCardIndex=3,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=int
 \
"NetworkCardIndex=4,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=int
 \
"NetworkCardIndex=5,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=int
 \
```

```
"NetworkCardIndex=6,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=int
 \
"NetworkCardIndex=7,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=int
 \
"NetworkCardIndex=8,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=int
 \
"NetworkCardIndex=9,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=int
 \
"NetworkCardIndex=10,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=in
 \
"NetworkCardIndex=11,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=in
 \
"NetworkCardIndex=12,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=in
 \
"NetworkCardIndex=13,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=in
 \
"NetworkCardIndex=14,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=in
 \
"NetworkCardIndex=15,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=in
 \
"NetworkCardIndex=16,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=in
```

Elastic Fabric Adapter を作成して Amazon EC2 インスタンスにアタッチする

Amazon EC2 の他の Elastic Network Interface と同様に、EFA を作成して Amazon EC2 インスタンスにアタッチすることができます。ただし、Elastic Network Interface とは異なり、`running` 状態のインスタンスに対して EFA をアタッチまたはデタッチすることはできません。

考慮事項

- EFA に関連付けられているセキュリティグループは変更することができます。OS バイパス機能を有効にするにはEFA が、セキュリティグループ自体との間のインバウンドおよびアウトバウンドのトラフィックをすべて許可するセキュリティグループのメンバーである必要があります。詳細については「ステップ 1: EFA 対応のセキュリティグループを準備する」を参照してください。

  Elastic Network Interface に関連付けられているセキュリティグループを変更するのと同じ方法で、EFA に関連付けられているセキュリティグループを変更します。詳細については「the section called “ネットワークインターフェイス属性の変更”」を参照してください。

- IP アドレスを Elastic Network Interface に割り当てるのと同じ方法で、Elastic IP (IPv4) および IPv6 アドレスを EFA (EFA と ENA の組み合わせ) ネットワークインターフェイスに割り当てます。詳細については「IP アドレスの管理」を参照してください。

  EFA のみのネットワークインターフェイスに IP アドレスを割り当てることはできません。

タスク

- EFA の作成
- 停止したインスタンスへの EFA のアタッチ
- インスタンス起動時の EFA のアタッチ
- 起動テンプレートへの EFA 追加

EFA の作成

EFA はVPC のサブネットに作成することができます。作成後に EFA を別のサブネットに移動することはできません。また、アタッチできるのは同じアベイラビリティーゾーンの停止したインスタンスに限ります。

Console

EFA (EFA と ENA の組み合わせまたは ENA のみ) ネットワークインターフェイスを作成する方法

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [ネットワークインターフェース] 選択し、それから[ネットワークインターフェイス]を選択します。
3. [説明] に、EFA の分かりやすい名前を入力してください。
4. [サブネット] で、EFA を作成するサブネットを選択してください。
5. [インターフェイスタイプ] で、次のオプションのいずれかを選択します。
   - [EFA と ENA の組み合わせ] – ENA デバイスおよび EFA デバイスの両方をサポートするネットワークインターフェイスを作成します。
   - [EFA のみ] – EFA デバイスのみを使用してネットワークインターフェイスを作成します。
6. (EFA と ENA の組み合わせのみ) ネットワークインターフェイスの IP アドレスおよびプレフィックスの割り当てを設定します。割り当てることができる IP アドレスおよびプレフィックスのタイプは、選択したサブネットによって異なります。IPv4 のみのサブネットの場

合、IPv4 IP アドレスおよびプレフィックスのみを割り当てることができます。IPv6 のみのサブネットの場合、IPv6 IP アドレスおよびプレフィックスのみを割り当てることができます。デュアルスタックサブネットの場合、IPv4 および IPv6 の両方の IP アドレス、ならびにプレフィックスを割り当てることができます。

> **Note**
>
> EFA のみのネットワークインターフェイスに IP アドレスを割り当てることはできません。

a. [プライベート IPv4 アドレス] や [IPv6 アドレス] の場合、[自動割り当て] を選択し、Amazon EC2 が選択したサブネットから IP アドレスを自動的に割り当てるようにするか、[カスタム] を選択して割り当てる IP アドレスを手動で指定します。

b. IPv6 アドレスを割り当てる場合、必要に応じて [プライマリ IPv6 IP の割り当て] を有効にできます。これにより、プライマリ IPv6 グローバルユニキャストアドレス (GUA) がネットワークインターフェイスに割り当てられます。プライマリ IPv6 アドレスを割り当てると、インスタンスまたは ENI へのトラフィックの中断を回避できます。詳細については、「IPv6 アドレス」を参照してください。

c. [IPv4 プレフィックス委任] や [IPv6 プレフィックス委任] の場合、[自動割り当て] を選択し、Amazon EC2 がサブネットの CIDR ブロックからプレフィックスを自動的に割り当てるようにするか、[カスタム] を選択してサブネットの CIDR ブロックからプレフィックスを手動で指定します。プレフィックスを指定した場合、AWS は別のリソースにまだ割り当てられていないことを確認します。詳細については「Amazon EC2 ネットワークインターフェイスのプレフィックス委任」を参照してください。

d. (オプション) アイドル接続の追跡タイムアウト設定を構成します。詳細については「アイドル接続追跡タイムアウト」を参照してください。

   - [TCP 確立タイムアウト] – 確立された状態のアイドル TCP 接続のタイムアウト時間 (秒単位)。最小: 60 秒。最大: 432000 秒 (5 日間)。デフォルト: 432000 秒。推奨: 432000 秒未満。
   - [UDP タイムアウト] – 1 つの方向または 1 つのリクエストレスポンストランザクションのみでトラフィックが発生した、アイドル UDP フローのタイムアウト時間 (秒単位)。最小: 30 秒。最大: 60 秒。デフォルト: 30 秒。

- [UDP ストリームタイムアウト] – 複数のリクエストレスポンストランザクションが発生したストリームとして分類される、アイドル UDP フローのタイムアウト時間 (秒単位)。最小: 60 秒。最大: 180 秒 (3 分)。デフォルト: 180 秒。

7. [Security groups] で、1 つまたは複数のセキュリティグループを選択してください。
8. [ネットワークインターフェイスの作成] をクリックしてください。

AWS CLI

EFA を作成するには

create-network-interface コマンドを使用します。--interface-type では EFA ネットワークインターフェイス用の efa または EFA のみのネットワークインターフェイス用の efa-only を指定します。

```
aws ec2 create-network-interface \
    --subnet-id subnet-0abcdef1234567890 \
    --interface-type efa \
    --description "my efa"
```

PowerShell

EFA を作成するには

New-EC2NetworkInterface コマンドレットを使用します。-InterfaceType では EFA ネットワークインターフェイス用の efa または EFA のみのネットワークインターフェイス用の efa-only を指定します。

```
New-EC2NetworkInterface `
    -SubnetId subnet-0abcdef1234567890 `
    -InterfaceType efa `
    -Description "my efa"
```

停止したインスタンスへの EFA のアタッチ

EFA はサポート対象の stopped 状態のインスタンスにアタッチすることができます。running 状態のインスタンスに EFA をアタッチすることはできません。サポートされるインスタンスタイプの詳細についてはサポートされるインスタンスタイプを参照してください。

ネットワークインターフェイスをインスタンスにアタッチするのと同じ方法で、EFA をインスタンスにアタッチできます。詳細については「ネットワークインターフェイスのアタッチ」を参照してください。

インスタンス起動時の EFA のアタッチ

AWS CLI

インスタンス起動時に既存の EFA をアタッチするには

--network-interfaces オプションで run-instances コマンドを使用します。プライマリネットワークインターフェイスにはEFA ネットワークインターフェイス、NetworkCardIndex=0、DeviceIndex=0 を指定します。複数の EFA ネットワークインターフェイスをアタッチするには、「ネットワーク帯域幅の最大化」を参照してください。

```
--network-interfaces "NetworkCardIndex=0, \
    DeviceIndex=0, \
    NetworkInterfaceId=eni-1234567890abcdef0, \
    Groups=sg-1234567890abcdef0, \
    SubnetId=subnet-0abcdef1234567890"
```

インスタンス起動時に新しい EFA をアタッチするには

--network-interfaces オプションで run-instances コマンドを使用します。プライマリネットワークインターフェイスにはNetworkCardIndex=0、DeviceIndex=0、InterfaceType=efa を使用します。複数の EFA ネットワークインターフェイスをアタッチする場合は「ネットワーク帯域幅の最大化」を参照してください。

```
--network-interfaces "NetworkCardIndex=0, \
    DeviceIndex=0, \
    InterfaceType=efa, \
    Groups=sg-1234567890abcdef0, \
    SubnetId=subnet-0abcdef1234567890"
```

PowerShell

インスタンス起動時に既存の EFA をアタッチするには

-NetworkInterfaces パラメータで New-EC2Instance コマンドレットを使用します。

```
-NetworkInterface $networkInterface
```

ネットワークインターフェイスを次のように定義します。

```
$networkInterface = New-Object
 Amazon.EC2.Model.InstanceNetworkInterfaceSpecification
$networkInterface.DeviceIndex = 0
$networkInterface.NetworkInterfaceId = "eni-1234567890abcdef0"
$networkInterface.Groups = @("sg-1234567890abcdef0")
$networkInterface.SubnetId = "subnet-0abcdef1234567890"
```

インスタンス起動時に新しい EFA をアタッチするには

-NetworkInterfaces パラメータで New-EC2Instance コマンドレットを使用します。

```
-NetworkInterface $networkInterface
```

ネットワークインターフェイスを次のように定義します。

```
$networkInterface = New-Object
 Amazon.EC2.Model.InstanceNetworkInterfaceSpecification
$networkInterface.DeviceIndex = 0
$networkInterface.InterfaceType = "efa"
$networkInterface.Groups = @("sg-1234567890abcdef0")
$networkInterface.SubnetId = "subnet-0abcdef1234567890"
```

起動テンプレートへの EFA 追加

EFA 対応のインスタンスの起動に必要な設定情報を含む起動テンプレートを作成できます。起動テンプレートではEFA と EFA のみのネットワークインターフェイスの両方を指定できます。EFA 対応の起動テンプレートを作成するには新しい起動テンプレートを作成し、サポート対象のインスタンスタイプ、EFA 対応の AMI、および EFA 対応のセキュリティグループを指定します。NetworkInterfaces にはアタッチする EFA ネットワークインターフェイスを指定します。プライマリネットワークインターフェイスにはNetworkCardIndex=0、DeviceIndex=0、InterfaceType=efa を使用します。複数の EFA ネットワークインターフェイスをアタッチする場合は「複数のネットワークカードを使用して Amazon EC2 インスタンスのネットワーク帯域幅を最大化する」を参照してください。

起動テンプレートを利用すると、AWS や AWS Batch など他の AWS ParallelCluster サービスで EFA 対応のインスタンスを起動できます。

起動テンプレートの作成の詳細については Amazon EC2 起動テンプレートを作成する を参照してください。

Amazon EC2 インスタンスから EFA をデタッチおよび削除する

Amazon EC2 インスタンスから EFA をデタッチしたり、Amazon EC2 の他の Elastic Network Interface と同じ方法で削除したりできます。

EFA のデタッチ

EFA をインスタンスからデタッチするにはまずインスタンスを停止する必要があります。実行中状態のインスタンスから EFA をデタッチすることはできません。

インスタンスから Elastic Network Interface をデタッチするのと同じ方法で、EFA をインスタンスからデタッチします。詳細については「ネットワークインターフェイスをデタッチする」を参照してください。

EFA の削除

EFA を削除するにはまずインスタンスから削除する必要があります。インスタンスにアタッチされている場合はEFA を削除する必要があります。

Elastic Network Interface を削除するのと同じ方法で EFAs を削除します。詳細については「ネットワークインターフェイスの削除」を参照してください。

Amazon EC2 で Elastic Fabric Adapter をモニタリングする

Elastic Fabric Adapter のパフォーマンスをモニタリングするには次の機能を使用できます。

トピック

- Amazon EC2 インスタンスの EFA ドライバーメトリクス
- Amazon VPC フローログ
- Amazon CloudWatch

Amazon EC2 インスタンスの EFA ドライバーメトリクス

Elastic Fabric Adapter (EFA) ドライバーは、EFA インターフェイスがアタッチされているインスタンスからリアルタイムで複数のメトリクスを発行します。このようなメトリクスを使用して、アプリケーションパフォーマンスとネットワーキングの問題のトラブルシューティング、ワークロードに適したクラスターサイズの選択、スケーリングアクティビティの事前計画、およびアプリケーションのベンチマークを実施して、メトリクスがインスタンスで利用できる EFA パフォーマンスを最大化するかどうかを判断できます。

トピック

- 使用可能な EFA ドライバーメトリクス
- インスタンスの EFA ドライバーメトリクスを取得する

使用可能な EFA ドライバーメトリクス

EFA ドライバーは次のメトリクスをリアルタイムでインスタンスに発行します。これらは、インスタンスの起動以降または最後のドライバーのリセット以降に、アタッチされた EFA デバイスから送信、受信、再送信、またはドロップされた、エラー、接続イベント、パケットまたはバイトの累積数を提供します。

メトリクス	説明	サポートされるインスタンスタイプ
`tx_bytes`	送信されたバイト数。 単位: バイト	EFA をサポートしているすべてのインスタンスタイプ
`rx_bytes`	受信バイトの数。 単位: バイト	EFA をサポートしているすべてのインスタンスタイプ
`tx_pkts`	送信されたパケットの数。 単位: 個	EFA をサポートしているすべてのインスタンスタイプ
`rx_pkts`	受信されたパケットの数。 単位: 個	EFA をサポートしているすべてのインスタンスタイプ

メトリクス	説明	サポートされるインスタンスタイプ
rx_drops	受信後にドロップされたパケットの数。 単位: 個	EFA をサポートしているすべてのインスタンスタイプ
send_bytes	送信オペレーションを使用して送信されたバイト数。 単位: バイト	EFA をサポートしているすべてのインスタンスタイプ
recv_bytes	送信オペレーションによって受信されたバイト数。 単位: バイト	EFA をサポートしているすべてのインスタンスタイプ
send_wrs	送信オペレーションを使用して送信されたパケットの数。 単位: 個	EFA をサポートしているすべてのインスタンスタイプ
recv_wrs	送信オペレーションによって受信されたパケットの数。 単位: 個	EFA をサポートしているすべてのインスタンスタイプ
rdma_write_wrs	完了した rdma 書き込みオペレーションの数。 単位: 個	EFA をサポートしているすべてのインスタンスタイプ

メトリクス	説明	サポートされるインスタンスタイプ
rdma_read_wrs	完了した rdma 読み取りオペレーションの数。 単位: 個	EFA をサポートしているすべてのインスタンスタイプ
rdma_write_bytes	rdma 書き込みオペレーションを使用して他のインスタンスによって書き込まれたバイト数。 単位: バイト	EFA をサポートしているすべてのインスタンスタイプ
rdma_read_bytes	rdma 読み取りオペレーションを使用して受信したバイト数。 単位: バイト	EFA をサポートしているすべてのインスタンスタイプ
rdma_write_wr_err	ローカルエラーまたはリモートエラーが発生した rdma 書き込みオペレーションの数。 単位: 個	EFA をサポートしているすべてのインスタンスタイプ
rdma_read_wr_err	ローカルエラーまたはリモートエラーが発生した rdma 読み取りオペレーションの数。 単位: 個	EFA をサポートしているすべてのインスタンスタイプ
rdma_read_resp_bytes	rdma 読み取りオペレーションに応答して送信されたバイト数。 単位: バイト	EFA をサポートしているすべてのインスタンスタイプ

メトリクス	説明	サポートされるインスタンスタイプ
rdma_write_recv_bytes	rdma 書き込みオペレーションによって受信されたバイト数。 単位: バイト	EFA をサポートしているすべてのインスタンスタイプ
retrans_bytes	再送信された EFA SRD バイトの数。 単位: 個	EFA をサポートする Nitro v4 以降のインスタンスタイプ
retrans_pkts	再送信された EFA SRD パケットの数。 単位: バイト	EFA をサポートする Nitro v4 以降のインスタンスタイプ
retrans_timeout_events	EFA SRD トラフィックがタイムアウトしてネットワークパスが変更された回数。 単位: 個	EFA をサポートする Nitro v4 以降のインスタンスタイプ
impaired_remote_conn_events	EFA SRD 接続が障害状態になり、スループットレートの制限が低減した回数。 単位: 個	EFA をサポートする Nitro v4 以降のインスタンスタイプ
unresponsive_remote_events	EFA SRD リモート接続が応答しなかった回数。 単位: 個	EFA をサポートする Nitro v4 以降のインスタンスタイプ

EFA をサポートするインスタンスタイプの詳細については、「サポートされるインスタンスタイプ」を参照してください。

インスタンスの EFA ドライバーメトリクスを取得する

rdma-tool コマンドラインツールを使用して、次のようにインスタンスにアタッチされたすべての EFA インターフェイスのメトリクスを取得できます。

```
$ rdma -p statistic show
link rdmap0s31/1
    tx_bytes 0
    tx_pkts 0
    rx_bytes 0
    rx_pkts 0
    rx_drops 0
    send_bytes 0
    send_wrs 0
    recv_bytes 0
    recv_wrs 0
    rdma_read_wrs 0
    rdma_read_bytes 0
    rdma_read_wr_err 0
    rdma_read_resp_bytes 0
    rdma_write_wrs 0
    rdma_write_bytes 0
    rdma_write_wr_err 0
    retrans_bytes 0
    retrans_pkts 0
    retrans_timeout_events 0
    unresponsive_remote_events 0
    impaired_remote_conn_events 0
```

または次のコマンドを使用して、インスタンスにアタッチされた各 EFA インターフェイスのメトリクスを sys ファイルから取得することもできます。

```
$ more /sys/class/infiniband/device_number/ports/port_number/hw_counters/* | cat
```

例

```
$ more /sys/class/infiniband/rdmap0s31/ports/1/hw_counters/* | cat
::::::::::::::
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/lifespan
::::::::::::::
12
::::::::::::::
```

```
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/rdma_read_bytes
::::::::::::::
0
::::::::::::::
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/rdma_read_resp_bytes
::::::::::::::
0
::::::::::::::
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/rdma_read_wr_err
::::::::::::::
0
::::::::::::::
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/rdma_read_wrs
::::::::::::::
0
::::::::::::::
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/rdma_write_bytes
::::::::::::::
0
::::::::::::::
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/rdma_write_recv_bytes
::::::::::::::
0
::::::::::::::
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/rdma_write_wr_err
::::::::::::::
0
::::::::::::::
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/rdma_write_wrs
::::::::::::::
0
::::::::::::::
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/recv_bytes
::::::::::::::
0
::::::::::::::
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/recv_wrs
::::::::::::::
0
::::::::::::::
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/rx_bytes
::::::::::::::
0
::::::::::::::
```

```
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/rx_drops
::::::::::::::
0
::::::::::::::
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/rx_pkts
::::::::::::::
0
::::::::::::::
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/send_bytes
::::::::::::::
0
::::::::::::::
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/send_wrs
::::::::::::::
0
::::::::::::::
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/tx_bytes
::::::::::::::
0
::::::::::::::
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/tx_pkts
::::::::::::::
0
::::::::::::::
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/retrans_bytes
::::::::::::::
0
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/retrans_pkts
::::::::::::::
0
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/retrans_timeout_events
::::::::::::::
0
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/unresponsive_remote_events
::::::::::::::
0
/sys/class/infiniband/rdmap0s31/ports/1/hw_counters/impaired_remote_conn_events
::::::::::::::
0
```

Amazon VPC フローログ

Amazon VPC フローログを作成することで、EFA との間で送受信されるトラフィックに関する情報を取得できます。フローログデータは Amazon CloudWatch Logs と Amazon S3 に発行できます。フローログを作成したら、選択した送信先でそのデータを取得して表示できます。詳細については、「Amazon VPC ユーザーガイド」の「VPC フローログ」を参照してください。

EFA のフローログを作成する方法はElastic Network Interface のフローログを作成する場合と同じです。詳細については、「Amazon VPC ユーザーガイド」の「フローログの作成」を参照してください。

フローログエントリで、EFA エントリは`srcAddress` および `destAddress` で識別されます。次の例に示されているように、これらはいずれも MAC アドレス形式になります。

```
version accountId  eniId        srcAddress        destAddress       sourcePort destPort
 protocol packets bytes start      end        action log-status
2       3794735123 eni-10000001 01:23:45:67:89:ab 05:23:45:67:89:ab -          -
 -        9       5689  1521232534 1524512343 ACCEPT OK
```

Amazon CloudWatch

Amazon EKS クラスターで EFA を使用している場合は CloudWatch Container Insights を使用して EFA をモニタリングできます。Amazon CloudWatch Container Insights は、`retrans_bytes`、`retrans_pkts`、`retrans_timeout_events`、`unresponsive_remote_event` を除くすべての EFA ドライバーメトリクスをサポートしています。

詳細については「Amazon CloudWatch ユーザーガイド」の「Amazon ECS と Kubernetes のコンテナインサイトメトリクス」を参照してください。

チェックサムを使用した EFA インストーラの検証

オプションで、MD5 または SHA256 チェックサムを使用して EFA tarball (`.tar.gz` ファイル) を検証できます。ソフトウェアパブリッシャーの ID を確認し、発行後にアプリケーションの変更または破損がないことを確認するために、この操作を行うことをお勧めします。

tarball を検証するには

MD5 チェックサムには md5sum ユーティリティを使用します。SHA256 チェックサムには sha256sum ユーティリティを使用し、tarball のファイル名を指定します。このコマンドはtarball ファイルを保存したディレクトリから実行する必要があります。

- MD5

```
$  md5sum tarball_filename.tar.gz
```

- SHA256

```
$  sha256sum tarball_filename.tar.gz
```

このコマンドは次の形式でチェックサム値を返します。

```
checksum_value tarball_filename.tar.gz
```

コマンドによって返されるチェックサム値を、次の表に示すチェックサム値と比較します。チェックサムが一致する場合はインストールスクリプトを実行しても安全です。チェックサムが一致しない場合はインストールスクリプトを実行せず、サポート にお問い合わせください。

例えば、次のコマンドは SHA256 チェックサムを使用して EFA 1.9.4 tarball を検証します。

```
$  sha256sum aws-efa-installer-1.9.4.tar.gz
```

出力の例:

```
1009b5182693490d908ef0ed2c1dd4f813cc310a5d2062ce9619c4c12b5a7f14 aws-efa-
installer-1.9.4.tar.gz
```

次の表に、最新バージョンの EFA のチェックサムを示します。

バージョン	チェックサム
EFA 1.49.0	MD5: bb10de6856e08a7e7e8a6b52bea6f681 SHA256: cf2e9281a2328a243c76f911a490faed43ca0fecfe4733c25e34b2e92a32c309
EFA 1.48.0	MD5: d30960f1876fb16ac8238193a26c6b29

バージョン	チェックサム
	SHA256: f3a8aadc2b94288694 a9b9375bcad84a82f0f843e0eef afda7b158b2200549cc
EFA 1.47.0	MD5: c81d4caf24dabc04a6e48185906 20f5f SHA256: 2df4201e046833c7dc 8160907bee7f52b76ff80ed1473 76a2d0ed8a0dd66b2db
EFA 1.46.0	MD5:a88bbd9b71624d7ca401b54bc2fc0c19 SHA256: 8302bd7849afb95c90 3a875d7dcb6f85b3d7629e9a8b6 7d020031cfc6f4d0ee1
EFA 1.45.1	MD5:91c3c87e16bbcaca1513252c38b771bb SHA256: 9aeb20c645135b6039 cc08986d8f14e63280f7839e882 a74df5e83627ffeaa17
EFA 1.45.0	MD5:800aeddfa9d9c5f139a7b8f7c4fec627 SHA256: 25ba26a0877fe33173 90dc126aad2f23e27fc461cf0b9 40004f032cb342fa539
EFA 1.44.0	MD5:d024f6bebe080db42745103b84ca7c43 SHA256: f129a5b44a49d593d2 47e55a59eb9bcb57121566e1c2e 42b832a4e794fa83d8a

バージョン	チェックサム
EFA 1.43.3	MD5: 4dbc6eeecc516760253c10cbedb6319d SHA256: 6c470ebce254c7165347b5048895ac2996c88567271642297f4c597738300652
EFA 1.43.2	MD5: 7287b25a07c9747c0d4001e8fc5f59b2 SHA256: de15c5bdbc83b952afbde876110830c604ad0796680e5157c05f7c1979a41069
EFA 1.43.1	MD5: 7cfafc8debaea51dd4966fa0b2bba673 SHA256:54211eda0c193138ee8ed09b5fb41c41fc76fe0a77935fa4ec8d989466342740
EFA 1.43.0	MD5: f2b3dd7dc8670b541f7c23fd58e5e503 SHA256: 786df3458c499237be33bb8e50ffd4da7c18c20e254380ffc80fb90833d8cc73
EFA 1.42.0	MD5: 94b2b1db09da1dde08ec049db1f24370 SHA256: 4114fe612905ee05083ae5cb391a00a012510f3abfecc642d86c9a5ae4be9008

バージョン	チェックサム
EFA 1.41.0	MD5: 086181c3ee3f8da512fc6e1c795e8936 SHA256: 3506354cdfbe31ff552fe75f5d0d9bb7efd29cf79bd99457347d29c751c38f9f
EFA 1.40.0	MD5: f3ec6f73fbeaccba082327507581157c SHA256: 30491b0fe7c3470d4439594538855c981b05fa69862d74f8c05eb9b97912368a
EFA 1.39.0	MD5: c223d5954a85a7fbcd248c942b866e43 SHA256: 2cbc028c03064633bb990782b47c36156637769e2f48704417a9c700a7a32101
EFA 1.38.1	MD5: f112569e828ab65187777f794bab542c SHA256: 83923374afd388b1cfcf4b3a21a2b1ba7cf46a01a587f7b519b8386cb95e4f81
EFA 1.38.0	MD5: 43a2a446b33a2506f40853d55059f1ea SHA256: 4f436954f35ad53754b4d005fd8d0be63de3b4184de41a695b504bdce0fecb22

バージョン	チェックサム
EFA 1.37.0	MD5: 6328070192bae920eca45797ad4c1db1 SHA256: 2584fc3c8bb99f29b3285e275747ff09d67c18e162c2a652e36c976b72154bfb
EFA 1.36.0	MD5: 1bec83180fbffb23452ab6469ca21dfa SHA256: de183f333cfb58aeb7908a67bf9106985ba3ccb7f8638b851d2a0d8dbfacaec4
EFA 1.35.0	MD5: 252f03c978dca5f8e8d9f34e488b256e SHA256: 432b6ad4368ba0cd8b902729d14a908a97be7a3dcc5239422ea994a47f35a5e1
EFA 1.34.0	MD5: 5cd4b28d27a31677c16139b54c9acb45 SHA256: bd68839e741b0afd3ec2e37d50603803cfa7a279c120f0a736cc57c2ff2d7fdc
EFA 1.33.0	MD5: e2f61fccbcaa11e2ccfddd3660522276 SHA256: 0372877b87c6a7337bb7791d255e1053b907d030489fb2c3732ba70069185fce

バージョン	チェックサム
EFA 1.32.0	MD5: db8d65cc028d8d08b5a9f2d88881c1b1 SHA256: 5f7233760be57f6fee6de8c09acbfbf59238de848e06048dc54d156ef578fc66
EFA 1.31.0	MD5: 856352f12bef2ccbadcd75e35aa52aaf SHA256: 943325bd37902a4300ac9e5715163537d56ecb4e7b87b37827c3e547aa1897bf
EFA 1.30.0	MD5: 31f48e1a47fe93ede8ebd273fb747358 SHA256: 876ab9403e07a0c3c91a1a34685a52eced890ae052df94857f6081c5f6c78a0a
EFA 1.29.1	MD5: e1872ca815d752c1d7c2b5c175e52a16 SHA256: 178b263b8c25845b63dc93b25bcdff5870df5204ec509af26f43e8d283488744
EFA 1.29.0	MD5: 39d06a002154d94cd982ed348133f385 SHA256: 836655f87015547e733e7d9f7c760e4e24697f8bbc261bb5f3560abd4206bc36

バージョン	チェックサム
EFA 1.28.0	MD5: 9dc13b7446665822605e66febe074035 SHA256: 2e625d2d6d3e073b5178e8e861891273d896b66d03cb1a32244fd56789f1c435
EFA 1.27.0	MD5: 98bfb515ea3e8d93f554020f3837fa15 SHA256: 1d49a97b0bf8d964d91652a79ac851f2550e33a5bf9d0cf86ec9357ff6579aa3
EFA 1.26.1	MD5: 884e74671fdef4725501f7cd2d451d0c SHA256: c616994c924f54ebfabfab32b7fe8ac56947fae00a0ff453d975e298d174fc96
EFA 1.26.0	MD5: f8839f12ff2e3b9ba09ae8a82b30e663 SHA256: bc1abc1f76e97d204d3755d2a9ca307fc423e51c63141f798c2f15be3715aa11
EFA 1.25.1	MD5: 6d876b894547847a45bb8854d4431f18 SHA256: d2abc553d22b89a4ce92882052c1fa6de450d3a801fe005da718b7d4b9602b06

バージョン	チェックサム
EFA 1.25.0	MD5: 1993836ca749596051da04694ea0d00c SHA256: 98b7b26ce031a2d6a93de2297cc71b03af647194866369ca53b60d82d45ad342
EFA 1.24.1	MD5: 211b249f39d53086f3cb0c07665f4e6f SHA256: 120cfeec233af0955623ac7133b674143329f9561a9a8193e473060f596aec62
EFA 1.24.0	MD5: 7afe0187951e2dd2c9cc4b572e62f924 SHA256: 878623f819a0d9099d76ecd41cf4f569d4c3aac0c9bb7ba9536347c50b6bf88e
EFA 1.23.1	MD5: 22491e114b6ee7160a8290145dca0c28 SHA256: 5ca848d8e0ff4d1571cd443c36f8d27c8cdf2a0c97e9068ebf000c303fc40797
EFA 1.23.0	MD5: 38a6d7c1861f5038dba4e441ca7683ca SHA256: 555d497a60f22e3857fdeb3dfc53aa86d05926023c68c916d15d2dc3df6525bd

バージョン	チェックサム
EFA 1.22.1	MD5: 600c0ad7cdbc06e8e846cb763f92901b SHA256: f90f3d5f59c031b9a964466b5401e86fd0429272408f6c207c3f9048254e9665
EFA 1.22.0	MD5: 8f100c93dc8ab519c2aeb5dab89e98f8 SHA256: f329e7d54a86a03ea51da6ea9a5b68fb354fbae4a57a02f9592e21fce431dc3a
EFA 1.21.0	MD5: 959ccc3a4347461909ec02ed3ba7c372 SHA256: c64e6ca34ccfc3ebe8e82d08899ae8442b3ef552541cf5429c43d11a04333050
EFA 1.20.0	MD5: 7ebfbb8e85f1b94709df4ab3db47913b SHA256: aeefd2681ffd5c4c631d1502867db5b831621d6eb85b61fe3ec80df983d1dcf0
EFA 1.19.0	MD5: 2fd45324953347ec5518da7e3fefa0ec SHA256: 99b77821b9e72c8dea015cc92c96193e8db307deee05b91a58094cc331f16709

バージョン	チェックサム
EFA 1.18.0	MD5: fc2571a72f5d3c7b7b576ce2de38d91e SHA256: acb18a0808aedb9a5e485f1469225b9ac97f21db9af78e4cd6939700debe1cb6
EFA 1.17.3	MD5: 0517df4a190356ab559235147174cafd SHA256: 5130998b0d2883bbae189b21ab215ecbc1b01ae0231659a9b4a17b0a33ebc6ca
EFA 1.17.2	MD5: a329dedab53c4832df218a24449f4c9a SHA256: bca1fdde8b32b00346e175e597ffab32a09a08ee9ab136875fb38283cc4cd099
EFA 1.17.1	MD5: 733ae2cfc9d14b52017eaf0a2ab6b0ff SHA256: f29322640a88ae9279805993cb836276ea240623820848463ca686c8ce02136f
EFA 1.17.0	MD5: d430fc841563c11c3805c5f82a4746b1 SHA256: 75ab0cee4fb6bd38889dce313183f5d3a83bd233e0a6ef6205d8352821ea901d

バージョン	チェックサム
EFA 1.16.0	MD5: 399548d3b0d2e812d74dd67937b696b4 SHA256: cecec36495a1bc6fdc82f97761a541e4fb6c9a3cbf3cfcb145acf25ea5dbd45b
EFA 1.15.2	MD5: 955fea580d5170b05823d51acde7ca21 SHA256: 84df4fbc1b3741b6c073176287789a601a589313accc8e6653434e8d4c20bd49
EFA 1.15.1	MD5: c4610267039f72bbe4e35d7bf53519bc SHA256: be871781a1b9a15fca342a9d169219260069942a8bda7a8ad06d4baeb5e2efd7
EFA 1.15.0	MD5: 9861694e1cc00d884fadac07d22898be SHA256: b329862dd5729d2d098d0507fb486bf859d7c70ce18b61c302982234a3a5c88f
EFA 1.14.1	MD5: 50ba56397d359e57872fde1f74d4168a SHA256: c7b1b48e86fe4b3eaa4299d3600930919c4fe6d88cc6e2c7e4a408a3f16452c7

バージョン	チェックサム
EFA 1.14.0	MD5: 40805e7fd842c36ececb9fd7f921b1ae SHA256: 662d62c12de85116df33780d40e0533ef7dad92709f4f613907475a7a1b60a97
EFA 1.13.0	MD5: c91d16556f4fd53becadbb345828221e SHA256: ad6705eb23a3fce44af3afc0f7643091595653a723ad0374084f4f2b715192e1
EFA 1.12.3	MD5: 818aee81f097918cfaebd724eddea678 SHA256: 2c225321824788b8ca3fbc118207b944cdb096b847e1e0d1d853ef2f0d727172
EFA 1.12.2	MD5: 956bb1fc5ae0d6f0f87d2e481d49fccf SHA256: 083a868a2c212a5a4fcf3e4d732b685ce39cceb3ca7e5d50d0b74e7788d06259
EFA 1.12.1	MD5: f5bfe52779df435188b0a2874d0633ea SHA256: 5665795c2b4f09d5f3f767506d4d4c429695b36d4a17e5758b27f033aee58900

バージョン	チェックサム
EFA 1.12.0	MD5: d6c6b49fafb39b770297e1cc44fe68a6 SHA256: 28256c57e9ecc0b0778b41c1f777a9982b4e8eae782343dfe1246079933dca59
EFA 1.11.2	MD5: 2376cf18d1353a4551e35c33d269c404 SHA256: a25786f98a3628f7f54f7f74ee2b39bc6734ea9374720507d37d3e8bf8ee1371
EFA 1.11.1	MD5: 026b0d9a0a48780cc7406bd51997b1c0 SHA256: 6cb04baf5ffc58ddf319e956b5461289199c8dd805fe216f8f9ab8d102f6d02a
EFA 1.11.0	MD5: 7d9058e010ad65bf2e14259214a36949 SHA256: 7891f6d45ae33e822189511c4ea1d14c9d54d000f6696f97be54e915ce2c9dfa
EFA 1.10.1	MD5: 78521d3d668be22976f46c6fecc7b730 SHA256: 61564582de7320b21de319f532c3a677d26cc46785378eb3b95c636506b9bcb4

バージョン	チェックサム
EFA 1.10.0	MD5: 46f73f5a7afe41b4bb918c81888fefa9 SHA256: 136612f96f2a085a7d98296da0afb6fa807b38142e2fc0c548fa986c41186282
EFA 1.9.5	MD5: 95edb8a209c18ba8d250409846eb6ef4 SHA256: a4343308d7ea4dc943ccc21bcebed913e8868e59bfb2ac93599c61a7c87d7d25
EFA 1.9.4	MD5: f26dd5c350422c1a985e35947fa5aa28 SHA256: 1009b5182693490d908ef0ed2c1dd4f813cc310a5d2062ce9619c4c12b5a7f14
EFA 1.9.3	MD5: 95755765a097802d3e6d5018d1a5d3d6 SHA256: 46ce732d6f3fcc9edf6a6e9f9df0ad136054328e24675567f7029edab90c68f1
EFA 1.8.4	MD5: 85d594c41e831afc6c9305263140457e SHA256: 0d974655a09b213d7859e658965e56dc4f23a0eee2dc44bb41b6d039cc5bab45

Elastic Fabric Adapter リリースノート

以下の表には、Elastic Fabric Adapter ソフトウェアのバージョン履歴と変更ログが記載されています。

バージョン	変更	リリース日

1.49.0	• libfabric 2.4.0amzn5.0 にアップグレード
	• EFA driver 3.1.0 にアップグレード
	• rdma-core 63.0 にアップグレード
	• AWSOFI NCCL Plugin 1.20.0 にアップグレード
	• openSUSE Leap のサポートを中止します。
	• NGC インストールの更新とバグ修正:
	  • `--build-ngc` を使用する場合、欠落している libfabric-aws-bin/dev パッケージをインストールします。
	  • build-ngc ユースケースの自動検出を削除し、関連する `--disable-build-ngc` オプションを削除します。
	  • `--skip-rdma-core` を指定すると、`--build-ngc` は暗示されなくなります。
	  • バグ修正: `--build-ngc` と `--skip-rdma-core` が指定されていない場合、rdma-core が想定どおりにインストールされるようになりました。
	  • フルインストールは、NGC インストールのすべてのユースケースで回避されます。
	  •
		2026 年 6 月 27 日

バージョン	変更	リリース日
	`--skip-mpi` および `--skip-rdma-core` は、NGC ユースケースに限定されないグローバルな選択肢になりました。	

バージョン	変更	リリース日

1.48.0	• libfabric 2.4.0amzn3.0 にアップグレード
	  • RDM オペレーションで shm アドレスの取得を最適化
	  • DC パケットの完了、CQ ドレイン、AH ユニットテスト、および MR ログ記録の use-after-free バグを修正
	  • efa_rdm_ep_get_peer と QP ロックの競合状態を修正
	  • FI_EFA_ENABLE_SHM_TRANSFER=0 でセグフォールトを引き起こす初期化されていない記述子配列を修正
	  • fi_info、err_buf、EFA デバイスの破壊、およびユニットテストのメモリリークを修正
	  • 破棄された QP を処理するために、CQ ポーリングで base_ep の null チェックを追加
	  • efa_qp_create で qp->base_ep を設定し、efa_mr_reg_ibv_mr が NULL を返す場合にエラーを返す
	  • デバイスレベルのロックを使用して QP テーブルをドメインからデバイスに移動
	  • データパスの直接リクエスト ID に QP 生成を追加し、CQE で QP 番号を検証
	  • データパスダイレクトの qp->ibv_qp_ex->wr_id の設定を停止
	  • 受信 recv の完了時に longcts_send リストから txe を適切に削除
	  • efa_cq_handle_error のリファクタリングとクリーンアップを実施
		2026 年 4 月 14 日

バージョン	変更	リリース日
	• MR 登録カウンターをアトミックオペレーションに変換 • shm MR 登録コードを移行し、内部 mr regv 関数を導入 • ASAN で MR キャッシュをデフォルトで無効化 • p2p が使用できないが FI_HMEM がリクエストされている場合にのみ、zcpy_rx を無効化 • デバッグ以外のビルドの tx pkt プールを追跡しないように変更 • wait_send プロシージャとモック関数を改善 • efa-direct CQ err_data と efa_show_help のログレベルを調整 • ASAN でサニタイザーマクロを定義し、memhook を無効化 • ep enable エラーパスとエラー完了処理のユニットテストを追加 • libnccl-ofi 1.19.0 にアップグレード • ROCm/HSA によって作成された dma-buf ハンドルの処理エラーを修正し、ROCm プラットフォームで HSA ランタイムを使用する場合の dma-buf サポート検出を追加 • 潜在的なメモリリークを修正 • パッケージノードから切断された NUMA ノードによって正しくないトポロジ生成が発生した Docker コンテナでの GB200 の NCCL トポロジ生成を修正 • 初期化中の NIC 0 での QP 使用率を低減 •	

バージョン	変更	リリース日
	P6-B200 および P6-B300 インスタンスタイプのチューナーアルゴリズムの選択肢を改善 • NGC コンテナへの libnccl-ofi インストールをサポート • 個々の RPM/DEB パッケージ署名検証サポートを追加 • RHEL 10 のサポートを追加 • Debian 12 および RHEL 10 に OFI NCCL プラグインを追加	

バージョン	変更	リリース日

1.47.0	• libfabric 2.4.0amzn1.0 にアップグレード
	  • ピア管理のリファクタリング: endpoint→peer hashmap を AV レベルに移動
	  • パケットエントリ生成カウンターを追加して ABA の問題を防止
	  • 待機オブジェクトでのブロック CQ 読み取りサポート (fi_cq_sread) を実装
	  • eager/mulreq/longread RTM パケット処理のダブルフリーを修正
	  • デバイス RDMA 機能に基づいて MR 登録 IBV アクセスフラグを修正
	  • FI_OPT_EFA_SENDRECV_IN_ORDER_ALIGNED _128_BYTES を無効化 (-FI_EOPNOTSUPP を返す)
	  • 2 つの x86 キャッシュライン (128 バイト) に収まるようにパケットエントリを最適化
	  • efa_rdm_srx_start の常時成功と RTM/RTA エラーでの受信ウィンドウのスライド
	  • エンドポイントのクリーンアップ中のリリース前に RX パケットエントリのリンクを解除
	• libnccl-ofi 1.18.0 にアップグレード
	  • P6-B300 のサポート: P6-B300 のカスタムチューナーの決定を追加
	  • 小さいメッセージサイズのチャネル数を減らすことで P6-B200 での PAT のパフォーマンスを改善
		2026 年 1 月 29 日

バージョン	変更	リリース日
	• プロトコルのデフォルトを変更: Trn1 は RDMA プロトコル、g7e.8xlarge は SENDRECV プロトコル • 動的プラットフォームの選択: AWS NIC の存在に基づいてランタイムに AWS の最適化を有効にする機能を追加。これにより単一のプラグインバイナリを AWS プラットフォームと AWS 以外のプラットフォームの両方で使用できるようになります。AWS の最適化は引き続きコンパイル時に無効にできます。 • RDMA プロトコルでの非 FI_MR_VIRT_ADDR プロバイダーのサポートを修正 • NIC PCIe リンクの速度と幅の NCCL へのレポートを改善 • スレッドごとに個別の Libfabric ドメインを必要とせずにマルチスレッドアプリケーションをサポートするようにスレッドモデルを再設計。 • リソースを正しい順序でクリーンアップすることで FI_MR_ENDPOINT プロバイダー (SENDRECV プロトコルのみをサポート) のサポートを修正 • rdma-core 61.0 にアップグレード • スレッドドメインが指定されている場合のシングルスレッド CQ の使用 • CQE プロセスでの QP 番号の検証 • efa driver 3.0.0 にアップグレード • 管理者エラー処理の改善 • 完了ポーリング時の QP 番号の正確性の確認 • 最適ページサイズ検出結果の無条件バックポートを削除	

バージョン	変更	リリース日
	• エラーポインタの errno 文字列を出力 • CQ 作成フローのコードを簡素化 • umem 作成フローでの CQ のアップストリーム変更をバックポート • 新しい DKMS バージョンの DKMS 設定を調整 • Open MPI 5.0.9amzn1 にアップグレード • バグ修正: MPI スレッドのサポートに基づいてドメインスレッドレベルを設定 • バグ修正: すべてのデータ転送イベントが確実に完了するよう FFI_COMPLETION フラグをリクエスト • コア数が多いシステムでリソースの枯渇を修正するために作成されたドメインの合計数を削減するために BTL と MTL の間でドメインを共有	

バージョン	変更	リリース日

1.46.0	• Debian 13 のサポートを追加
	• efa-nv-peermem 1.2.3 にアップグレード
	  • 新しいカーネルと DKMS でビルドエラーと警告を修正
	• libfabric 2.3.1amzn4.0 にアップグレード
	  • efa-rdm プロトコルパスのデータパスダイレクトを有効化
	  • ROCr HMEM サポートを有効化
	  • 望ましくない送受信を無効にする新しい FI_OPT_EFA_USE_UNSOLICITED_WRITE_RECV オプション
	  • 接続確立時間を最適化
	• RHEL 9、Rocky Linux 9、および Debian 13 への libnccl-ofi を追加
		2025 年 12 月 12 日

1.45.1	• libfabric 2.3.1amzn3.0 にアップグレード • プロバイダーがアドバタイズした 4 バイトのサポート中に cq_data が 2 バイトに切り捨てられるバグを修正	2025 年 11 月 26 日

バージョン	変更	リリース日

1.45.0	• rdma-core 60.amzn0 にアップグレード
	  • ドアベルを返すように DV クエリ CQ を拡張
	• libfabric 2.3.1amzn2.0 にアップグレード
	  • efa-direct の FI_RX_CQ_DATA モードをサポート
	  • データパスの直接パスで WQE ポストを最適化
	  • cntr の進行状況の競合状態を修正
	  • 警告メッセージのログ記録を改善
	  • 削除されたピアの RMA 完了を正しく処理
	  • AH 制限に達したときに暗黙的な AV から AH エントリを削除
	• libnccl-ofi 1.17.2 にアップグレード
	  • Libfabric の初期化が失敗した場合の NCCL v2.28.x のクラッシュを修正
	  • g7e インスタンスファミリーのサポートを追加
	  • NCCL が DMA-BUF をサポートするプラットフォームで GPUDirect RDMA パスを誤って使用しようとする問題を修正
	  • エンドポイントごとのメモリ登録を必要とする NIC のシャットダウン順序の問題を修正 (Cray Slingshot)
		2025 年 11 月 17 日

バージョン	変更	リリース日

1.44.0	• rdma-core 59.amzn0 にアップグレード
	  • QP と CQ をスレッドドメインにバインドするためのサポートを追加
	  • 親ドメインを割り当てるためのサポートを追加
	  • スレッドドメインを割り当てるためのサポートを追加
	  • 直接動詞クエリ QP と CQ を追加
	  • 外部メモリを使用して CQ を作成するオプションを追加
	  • QP 破棄後の CQ ポーリングを修正
	• libfabric 2.3.1amzn1.0 にアップグレード
	• Open MPI 5.0.8amzn1 にアップグレード
	  • 機能拡張
	  • バグ修正
	• libnccl-ofi 1.17.1 にアップグレード
	  • NCCL v2.28.3-1 をサポートし、NCCL v2.17.1 以降との下位互換性を維持します。
	  • platform-aws を使用したビルドでは、Libfabric v1.22.0amzn4.0 以降が必要です。Libfabric v2.3.1amzn1.0 までのバージョンでテスト済みです。
	  • CUDA12 と CUDA13 間の互換性をサポートするよう修正。
		2025 年 10 月 29 日

バージョン	変更	リリース日

1.43.3	• efa ドライバー 2.17.3 にアップグレード
	  • NVIDIA 580 ドライバーで P2P をサポート
	  • 6.17 以降のカーネル用に MR 登録を調整
		2025 年 10 月 1 日

1.43.2	• libnccl-ofi 1.16.3 にアップグレード
	  • NCCL v2.27.7-1 をサポートし、NCCL v2.17.1 以降との下位互換性を維持します。
	  • platform-aws でビルドするには、Libfabric v1.22.0amzn4.0 以降が必要です。Libfabric 2.1.0amzn5.0 までのバージョンでテスト済みです。
	  • NCCL が複数のプロキシスレッドを作成する一部のアプリケーションのパフォーマンスを向上するため、すべての Amazon EC2 インスタンスタイプで domain-per-thread をデフォルトで有効にします。
		2025 年 8 月 15 日

1.43.1	• libnccl-ofi 1.16.2 にアップグレード
	  • NCCL v2.27.6-1 をサポートし、NCCL v2.17.1 以降との下位互換性を維持します。
	  • platform-aws を使用したビルドでは、Libfabric v1.22.0amzn4.0 以降が必要です。Libfabric 2.1.0amzn4.0 までのバージョンでテスト済みです。
	  • Amazon EC2 p5.4xlarge インスタンスタイプでの OFI NCCL プラグインの使用をサポートする新しいプラットフォーム設定を追加
	• libfabric 2.1.0amzn5.0 にアップグレード
		2025 年 7 月 31 日

バージョン	変更	リリース日

1.43.0	• libnccl-ofi 1.16.1 にアップグレード
	  • NCCL v2.27.6-1 をサポートし、NCCL v2.17.1 以降との下位互換性を維持します。
	  • platform-aws を使用したビルドでは、Libfabric v1.22.0amzn4.0 以降が必要です。Libfabric 2.1.0amzn3 までのバージョンでテスト済みです。
	  • トポロジファイルで報告された PCI リンク速度形式をカーネル 5.7 以降と一致するように更新
	  • 同じ PCI レベルのアクセラレーターを持たない libfabric NIC をスキップするために SKIP_NICS_WITHOUT_ACCEL_AT_SAME_PCI_LEVEL を追加
	• efa ドライバー 2.17.2 にアップグレード
	• efa-nv-peermem 1.2.2 にアップグレード
	• libfabric 2.1.0amzn4.0 にアップグレード
	• rdma-core 58.amzn0 にアップグレード
	  • 作業リクエストインデックスの二重使用を修正
	  • post_send トレースポイントに WQE のサイズ (長さ) を追加
	• Graviton プラットフォームの最適化を有効化
	• Ubuntu 20.04 のサポートを廃止
		2025 年 7 月 25 日

バージョン	変更	リリース日

1.42.0	• efa ドライバー 2.15.3 にアップグレード
	• efa-nv-peermem 1.2.1 にアップグレード
	• rdma-core 57.amzn1 にアップグレード
	  • 作業リクエストインデックスの二重使用を修正
	• libfabric 2.1.0amzn3.0 にアップグレード
	• libnccl-ofi 1.15.0 にアップグレード
	  • NCCL 2.26.6-1 をサポートし、NCCL v2.17.1 以降との下位互換性を維持します。
	  • platform-aws を使用したビルドでは、Libfabric v1.22.0amzn4.0 以降が必要です。Libfabric 2.1.0amzn3 までのバージョンでテスト済みです。
	  • ビルドシステムとプラットフォームのサポート
	  • Amazon EC2 P6-B200 プラットフォームのサポートを追加
	  • デフォルトのプラグインライブラリ名を libnccl-net-ofi.so に変更し、下位互換性を維持するために、デフォルトで libnccl-net-ofi.so から libnccl-net.so へのシンボリックリンクを作成しました。これにより、ユーザーは NCCL_NET_PLUGIN=ofi を設定して、NCCL が通信に OFI プラグインを使用するように強制できます。設定する --disable-nccl-net-symlink を指定すると、シンボリックリンクがスキップされ、同じコンテナに複数のプラグインをインストールできるようになります。
	  • チューニングとパフォーマンスの向上
	  • 0x0 および 0x7 ビットマスクの AllReduce、AllGather、および ReduceScatter 領域に対する P6-B200 でのチューナーサポートを追加
		2025 年 6 月 6 日

バージョン	変更	リリース日
	• 実験結果と分析に基づいて P5en および P6-B200 プラットフォームのデフォルトレイテンシーを更新 • 将来のトラフィック優先順位付けのために trafficClass パラメータをサポートする NCCL v10 API を使用するように更新 • プラグインコードベースを C から C++ に移行 • システム間で GPU あたりの NIC 数が異なるジョブのサポートを追加しました。詳細については、OFI_NCCL_FORCE_NUM_RAILS ランタイム環境変数のドキュメントを参照してください。	
1.41.0	• rdma-core 57.amzn0 にアップグレード	2025 年 5 月 16 日
1.40.0	• libfabric 2.1.0amzn2.0 にアップグレード • rdma-core 56.0 にアップグレード • efa-config 1.18 にアップグレード • Debian 10 のサポートを非推奨 • libnccl-ofi 1.14.2 にアップグレード • ARM for Ubuntu および Amazon Linux 2023 の Libfabric での CUDA サポートを有効化 • ARM for Ubuntu および Amazon Linux 2023 に libnccl-ofi を追加	2025 年 5 月 2 日
1.39.0	• libfabric 2.1.0 にアップグレード • Open MPI 5.0.6 にアップグレード • libnccl-ofi 1.14.1 にアップグレード • efa ドライバー 2.15.0 にアップグレード	2025 年 4 月 16 日
1.38.1	• `Libfabric 1.22.0amzn5.0` に更新	2025 年 3 月 3 日

バージョン	変更	リリース日
1.38.0	• `libnccl-ofi 1.13.2-1` を追加します。	2025 年 1 月 8 日
1.37.0	• `libfabric 1.22.0amzn4.0` にアップグレード • `rdma-core 54.amzn0` にアップグレード	2024 年 11 月 18 日
1.36.0	• Debian 12 のサポートを追加 • `/opt/amazon/efa/bin` のユーティリティスクリプトとして `efa_test.sh` を配布 • efa ドライバー 2.13.0 にアップグレード • `libfabric 1.22.0amzn3.0` にアップグレード • `rdma-core 54.0` にアップグレード • open mpi 4.1.7 にアップグレード	2024 年 11 月 7 日
1.35.0	• Open MPI 5.0.5 にアップグレード • PRRTE 3.0.6 にアップグレードし、RPM パッケージの名前を prrte-aws に変更 • OpenPMIx RPM パッケージの名前を pmix-aws に変更 • dpkg および RPM ビルドのビルドオプションを設定 • `Libfabric 1.22.0amzn2.0` にアップグレード • efa ドライバー 2.12.1 にアップグレード	2024 年 10 月 14 日
1.34.0	• CentOS 7 および RHEL 7 のサポートを終了 • `Libfabric 1.22.0amzn1.0` の取り込み • efa-config パッケージを 1.17 に更新	2024 年 8 月 6 日

バージョン	変更	リリース日
1.33.0	• efa ドライバー 2.10.0 にアップグレード • `rdma-core 52.0` にアップグレード • PRRTE 3.0.5 にアップグレード • Open MPI 5.0.3 にアップグレード • Ubuntu 24.04 LTS のサポートを追加 • OpenPMIx、PRRTE、Open MPI HTML のドキュメントを削除。ユーザーは公式ウェブサイトを参照してください。 • openSUSE 15.4 以前のサポートの終了	2024 年 6 月 20 日
1.32.0	• efa ドライバー 2.8.0 にアップグレード • efa-nv-peermem ドライバー 1.1.1 にアップグレード • efa-config パッケージを 1.16 にアップグレード • efa-profile パッケージを 1.7 にアップグレード • `rdma-core 50.0` にアップグレード • `libfabric 1.21.0amzn1.0` にアップグレード • `fi_pingpong` ポート選択で `efa_test.sh` を強化 • 必要に応じてシステムから新しい rdma-core をインストール	2024 年 4 月 18 日
1.31.0	• OpenPMIx 4.2.8 にアップグレード • PRRTE 3.0.3 にアップグレード • Open MPI 5.0.2 にアップグレードし、MCA コンポーネントを DSO に変換 • `Libfabric 1.20.1amzn1.0` にアップグレード	2024 年 3 月 7 日
1.30.0	• デフォルトで `/opt/amazon/pmix` にインストールされている OpenPMIx 4.2.7 を包含 • デフォルトで `/opt/amazon/prrte` にインストールされている PRRTE 3.0.2 を包含 • デフォルトで `/opt/amazon/openmpi5` にインストールされている Open MPI 5.0.0 を包含 • efa-profile を 1.6 に更新	2023 年 12 月

バージョン	変更	リリース日
1.29.1	• `libfabric 1.19.0amzn4.0` の取り込み	2023 年 12 月
1.29.0	• efa カーネルドライバー 2.6.0 の取り込み • `libfabric 1.19.0amzn3.0` の取り込み	2023 年 11 月
1.28.0	• Open MPI を 4.1.6 にアップグレード • `libfabric 1.19.0amzn2.0` の取り込み	2023 年 10 月
1.27.0	• インストーラに `Libfabric 1.19.0amzn1.0` を追加 • Open MPI 4 の GCC 組み込みアトミックを有効化	2023 年 9 月
1.26.1	• `efa_test.sh` ：無限再試行バグを修正	2023 年 9 月
1.26.0	• RHEL 9 のサポートを追加 • 複数のディストリビューションでデバッグビルドを修正 • `libfabric 1.18.2amzn1.0` の取り込み • `efa_test.sh` ：`fi_pingpong` の再試行を追加	2023 年 9 月
1.25.1	• `libfabric 1.18.1amzn1.0` の取り込み • EFA インストーラスクリプトのタイムアウトを更新	2023 年 9 月
1.25.0	• Amazon Linux 2023 のサポートを追加 • インストール後テストのバグ修正	2023 年 7 月
1.24.1	• libfabric を 1.18.1 にアップグレード • efa ドライバーを 2.5.0 にアップグレード	2023 年 7 月
1.24.0	• `rdma-core 46.0` の取り込み • efa ドライバー 2.4.1 の取り込み • Debian 11 をサポート	2023 年 6 月
1.23.1	• `libfabric 1.18.0amzn2.0` の取り込み	2023 年 6 月

バージョン	変更	リリース日
1.23.0	• Debian 10 のサポートを追加 • Ubuntu 18.04 LTS のサポートの終了 • efa-config パッケージを 1.14 にアップグレード • `libfabric 1.18.0amzn1.0` の取り込み	2023 年 5 月
1.22.1	• libfabric を 1.17.1 にアップグレード	2023 年 3 月
1.22.0	• Open MPI を 4.1.5 にアップグレード • libfabric を 1.17.0 にアップグレード • efa-config パッケージを 1.13 にアップグレード	2023 年 2 月
1.21.0	• Rocky Linux 9 OS のサポートを追加 • efa ドライバー 2.1.1 の取り込み • `libfabric 1.16.1amzn3.0` の取り込み • efa-config パッケージを 1.12 にアップグレード	2022 年 12 月
1.20.0	• Rocky Linux 8 OS のサポートを追加 • efa ドライバー 2.1.0 の取り込み • `rdma-core 43.0` の取り込み • `libfabric 1.16.1amzn1.0` の取り込み	2022 年 11 月
1.19.0	• `libfabric 1.16.0` の取り込み • `--enable-orterun-prefix-by-default` を使用して Open MPI を構築	2022 年 10 月
1.18.0	• Ubuntu 22.04 のサポートを追加	2022 年 8 月
1.17.3	• libfabric を `1.16.0~amzn4.0` に更新。~ は、それが libfabric 1.16.0 のプレリリースバージョンであることを示します。 • インストール後の pingpong テストタイムアウトを 20 秒に延長	2022 年 8 月

バージョン	変更	リリース日
1.17.2	• libfabric を `1.16.0~amzn3.0` に更新。~ は、それが libfabric 1.16.0 のプレリリースバージョンであることを示します。	2022 年 7 月
1.17.1	• libfabric を `1.16.0~amzn2.0` に更新。~ は、それが libfabric 1.16.0 のプレリリースバージョンであることを示します。 • libfabric の構築時に試験的ネットプロバイダーを無効化	2022 年 7 月
1.17.0	• `rdma-core` を v41.0 に更新 • Open MPI を 4.1.4 に更新 • libfabric を `1.16.0~amzn1.0` に更新。~ は、それが libfabric 1.16.0 のプレリリースバージョンであることを示します。	2022 年 7 月
1.16.0	• libfabric を `1.15.1amzn1.0` に更新し、neuron library の名前変更を包含 • efa-config を 1.10 にアップグレード • libfabric ビルドの opx プロバイダーと rxd プロバイダーを除外	2022 年 6 月
1.15.2	• libfabric を 1.14.1 に更新	2022 年 5 月
1.15.1	• libfabric を `1.14.0amzn1.0` に更新	2022 年 3 月

バージョン	変更	リリース日
1.15.0	• Open SuSE 15.3 でインストールが失敗する原因となっていたバグを修正 • Open SuSE 15.2 のサポートを終了 (Open SuSE 15.2 のライフサイクルが終了したため) • CentOS 8 のサポートを終了 (CentOS 8 のライフサイクルが終了したため) • libfabric を 1.14.0 に更新 • efa カーネルドライバーを 1.16.0 に更新 • `rdma-core` を v39.0 に更新 • Open MPI をバージョン 4.1.2 に更新	2022 年 2 月
1.14.1	• libfabric を `1.13.2amzn1.0` に更新	2021 年 10 月
1.14.0	• efa カーネルドライバー 1.14.2 の取り込み • 最新の efa カーネルドライバーがデフォルトで GDR サポートを有効にするため、`efa_installer.sh` の `-g, --enable-gdr` を no-op オプションに変更 • `rdma-core v37.0` の取り込み • `libfabric 1.13.2` の取り込み • 不明なパッケージがインストールされないように、パッケージリストを追加してインストール中に RPM/DEB とリストを比較 • EFA ドライバーの再ロード後に udev ルールが適用されるまで待機するため、インストーラスクリプトに sleep を追加	2021 年 10 月

バージョン	変更	リリース日
1.13.0	• `rdma-core` を v35.0 に更新 • `libfabric` を v1.13.0amzn1.0 に更新 • Gravition2 プラットフォームで CentOS/RHEL 8 の EFA サポートを追加 • システムに新しいバージョンのローカルパッケージがインストールされているときにローカルパッケージのインストールをスキップするため、インストーラスクリプトにバージョン比較ロジックを追加	2021 年 8 月
1.12.3	• EFA カーネルモジュールを 1.13.0 に更新 • efa-config パッケージをバージョン 1.9 に更新 大型のデフォルト huge page サイズを処理するために huge page 予約の計算を改善	2021 年 7 月
1.12.2	• EFA カーネルモジュールを 1.12.3 に更新 • `--with-libevent=external` と `--with-hwloc=external` を使用して Open MPI debian パッケージを構築 • MPI rpm ビルド ID を 2 にバージョンアップして、CentOS 8 での HWLOC の下位互換性問題を修正 • SLES15SP2 および openSUSE 15.2 にある kernel-devel および kernel-source パッケージのインストールを削除	2021 年 6 月
1.12.1	• Libfabric をバージョン 1.11.2amzon1.1 に更新 • EFA カーネルモジュールをバージョン 1.12.1 に更新	2021 年 5 月

バージョン	変更	リリース日
1.12.0	• Open MPI をバージョン 4.1.1 に更新 • Libfabric をバージョン 1.11.2amzn1.0 に更新 • AL2 rdma-core と同じパッケージング設定を使用して Amazon Linux 2 の rdma-core を構築 • Open MPI RPM での構築時に `-Wl,--enable-new-dtags` を強制しない • hwloc と libevent のシステムライブラリを使用して Open MPI を構築 • EFA カーネルモジュールをバージョン 1.12.0 に更新 • efa-config パッケージをバージョン 1.8 に更新 存続期間の長いインスタンスのための huge page 予約の計算を改善 • efa-profile パッケージをバージョン 1.5 に更新 P4d での Open MPI 4.1.0 ハングを修正するための回避策として機能していた open mpi collective tuning ファイルを削除 • `rdma-core` を v32.1 に更新 • Amazon Linux 1 と Ubuntu 16.04 のサポートを終了	2021 年 5 月
1.11.2	• 設定ファイル経由で MPI_BARRIER の実装に使用されるデフォルトのアルゴリズムを変更することで、P4d での Open MPI の使用時における Open MPI ハングを修正 • コンパイラ問題を回避するため、`--disable-builtin-atomics` を使用して ARM 上の Open MPI での組み込みアトミックの使用を無効化	2021 年 2 月
1.11.1	• Open MPI をバージョン 4.1.0 に更新 • efa-config パッケージをバージョン 1.7 に更新 huge page ページ予約数の計算を改善 • efa-profile パッケージをバージョン 1.3 に更新 Open MPI 4.1.0 が使用されるようになったため、不要な collectives decision ファイルを削除	2020 年 12 月

バージョン	変更	リリース日
1.11.0	• Gravition2 プラットフォームのサポートを追加 • `rdma-core` をバージョン 31.2amzn に更新 • Libfabric をバージョン 1.11.1amzn1.0 に更新 • efa-config をバージョン 1.6 に更新 • efa-profile をバージョン 1.2 に更新	2020 年 12 月
1.10.1	• CentOS/RHEL 8 のサポートを追加 • Ubuntu 20.04 のサポートを追加 • SUSE Linux Enterprise 15 のサポートを追加	2020 年 11 月
1.10.0	• P4d プラットフォームの GPUDirect RDMA サポートを追加 GDR 対応カーネルモジュールとユーザースペースをインストールするために `--enable-gdr` インストーラオプションを使用 • EFA カーネルモジュールをバージョン 1.10.2 に更新 • `rdma-core` をバージョン 31.amzn0 に更新 • Libfabric をバージョン 1.11.1 に更新 • Open MPI をバージョン 4.0.5 に更新 • efa-config をバージョン 1.5 に更新 • efa-profile をバージョン 1.1 に更新 改善された Open MPI collectives decision ファイルを包含	2020 年 10 月
1.9.5	• efa-config をバージョン 1.4 に更新 Open MPI collective decision ファイルのバグを修正	2020 年 9 月
1.9.4	• Open MPI をバージョン 4.0.3 に更新 • Libfabric をバージョン 1.10.1amazon1.1 に更新 • `rdma-core` をバージョン 28.amzn0 に更新	2020 年 7 月

バージョン	変更	リリース日
1.9.3	• EFA カーネルモジュールをバージョン 1.6.0 に更新 • `rdma-core` をバージョン 28.amzn0 に更新 • Libfabric をバージョン 1.10.1amzn1.1 に更新 • efa-config をバージョン 1.3 に更新 Open MPI の collectives tuning ファイルを追加 • dkms が既にインストール済みの場合に dkms のインストールをスキップ • `--skip-kmod` インストールモードが実際に機能するように修正	2020 年 6 月
1.8.4	• オペレーティングシステムパッケージマネージャーが設定ファイルを追跡できるように、これらのファイルを efa-config パッケージと efa-profile パッケージに移動 • Open MPI をバージョン 4.0.3 に更新	2020 年 4 月
1.8.3	• EFA カーネルモジュールをバージョン 1.5.1 に更新 • インストールの信頼性向上のため、EPEL リポジトリに依存するのではなく、いくつかのプラットフォームに DKMS を分散 • Amazon Linux 2 で構築された RPM の代わりに CentOS 7 で構築された RPM を RHEL 7 にインストール	2020 年 2 月
1.8.2	• カーネルモジュールと `rdma-core` 間のデバイス命名における不一致のため、`rdma-core` を 25 にバージョンダウン	2020 年 1 月
1.8.1	• Libfabric をバージョン 1.9.0amzn1.1 に更新	2020 年 1 月
1.8.0	• `rdma-core` をバージョン 27.0 に更新 • EFA カーネルモジュールをバージョン 1.5.0 に更新 • Libfabric をバージョン 1.9.0amzn1.0 に更新 • EFA カーネルモジュールと `rdma-core` のみをインストールするためのオプション `--minimal` を追加	2019 年 12 月
1.7.1	• Libfabric をバージョン 1.8.1amzn1.3 に更新	2019 年 12 月

バージョン	変更	リリース日
1.7.0	• Libfabric モジュールファイルを追加 • Libfabric をバージョン 1.8.1amzn1.1 に更新	2019 年 11 月
1.6.2	• Open MPI をバージョン 1.6.2 に更新	2019 年 10 月
1.6.1	• Libfabric をバージョン 1.8.1amzn1.0 に更新 • Open MPI をバージョン 4.0.1 に更新 • `rdma-core` をバージョン 26.0 に更新	2019 年 10 月
1.5.4	• EFA カーネルモジュールをバージョン 1.4.1 に更新	2019 年 9 月
1.5.3	• EFA カーネルモジュールをバージョン 1.3.1 に更新 • `kernel-devel` または `linux-headers` パッケージのインストールを回避 (カーネルドライバーをインストールする場合を除く)	2019 年 9 月
1.5.1	• EFA デバイスが存在する場合にのみ huge page を設定	2019 年 8 月
1.5.0	• Libfabric をバージョン 1.8.0amzn1.1 に更新 • `rmda-core` をバージョン 25.0 に更新	2019 年 8 月
1.4.1	• Open MPI と Libfabric の共有ライブラリが適切に配置されていることを確実にするため、Libfabric と Open MPI のライブラリパス (`/opt/amazon/efa/lib64` と `/opt/amazon/efa/openmpi/lib`) を `/etc/ld.so.conf.d/efa.conf` に追加	2019 年 7 月
1.4.0	• EFA カーネルモジュールをバージョン 1.3.0 に更新 • Libfabric をバージョン 1.8.0amzn1.0 に更新 • Intel MPI 2019 Update 4 をサポートするための初回リリース	2019 年 7 月

Amazon EC2 トポロジー

Amazon EC2 トポロジーは、コンピューティングキャパシティの相対的な近接性を階層的に示します。この情報を使用して、ハイパフォーマンスコンピューティング (HPC)、機械学習 (ML)、生成 AI のコンピューティングインフラストラクチャを大規模に管理できます。

使用可能な API

Amazon EC2 では、EC2 トポロジーを把握するための API が 2 つ用意されています。

- DescribeInstanceTopology
  - 実行中のインスタンスがネットワーク階層内で配置されている場所を相互に相対的に示します。
  - 既存のインスタンスでワークロードを実行する場所を最適化するのに役立ちます。
- DescribeCapacityReservationTopology
  - インスタンスを起動する前に、リザーブドキャパシティがネットワーク階層内で配置される場所を相互に相対的に示します。
  - インスタンスを起動する前にリザーブドキャパシティの配置を知らせることで、キャパシティプランニングを支援します。

主な利点

EC2 トポロジーには、主に以下のような利点があります。

- キャパシティ管理 – リソース使用率を最適化できます。
- ジョブスケジューリング – ワークロードの配置について、情報に基づいた意思決定を行えます。
- ノードランキング — 緊密に結合されたインスタンスでパフォーマンスを最適化するための相対的な近接性を理解できます。

考慮事項

- トポロジービューは次の場合にのみ使用できます。
  - インスタンスが `running` の状態である
  - キャパシティ予約が `pending` または `active` で状態にある
- 各トポロジービューは AWS アカウントごとに一異なります。
- AWS マネジメントコンソールではトポロジーの表示はサポートされていません。

- トポロジー情報はインスタンスの配置を理解するのに役立ちますが、既存のインスタンスに物理的に近い新しいインスタンスを起動するために使用することはできません。インスタンスの配置に影響を与えるにはクラスタープレイスメントグループでキャパシティ予約を作成できます。

料金

EC2 トポロジーを記述するのに追加のコストはかかりません。

内容

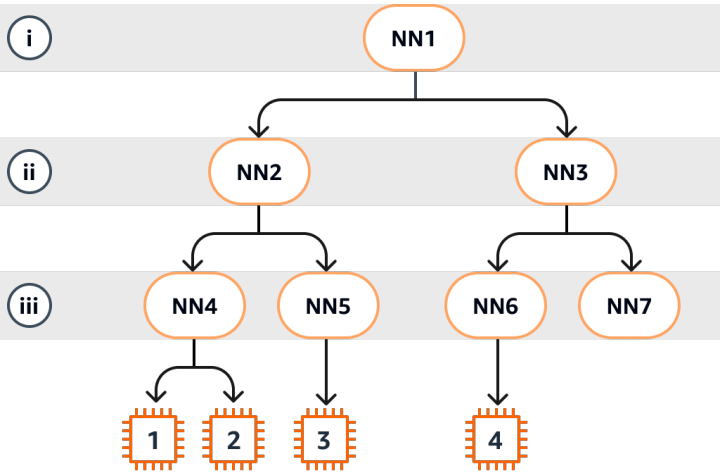
- Amazon EC2 トポロジーの仕組み
- Amazon EC2 トポロジーの前提条件
- Amazon EC2 インスタントポロジーの例

Amazon EC2 トポロジーの仕組み

AWS ネットワークはレイヤーの階層構造内に配置されています。EC2 インスタンスは、インスタンスのタイプに応じて、3 番目のレイヤー以下にあるネットワークに接続します。インスタンスのトポロジーは一連のノードによって記述され、ネットワークの各レイヤーに 1 つのノードがあります。DescribeInstanceTopology または DescribeCapacityReservationTopology API レスポンスにあるノードセットはネットワーク階層のトップダウンビューを提供し、最下部のノードがインスタンスに接続されます。

> **Note**
>
> インスタンスタイプには、ネットワーク内の 4 つのレイヤーを表すノードセットにある 4 つのネットワークノードで構成されているものもあれば、ネットワーク内の 3 つのレイヤーを表す 3 つのネットワークノードで構成されているものもあります。サポートされているインスタンスタイプについては、「インスタンスのタイプ」を参照してください。
> キャパシティ予約のタイプによっては、1 つ、2 つ、または 3 つのネットワークノードしか表示されない場合があります。

次の図は EC2 トポロジーを視覚的に表しており、EC2 トポロジーの理解に役立ちます。ネットワークノードは NN1 ~ NN7 として示されます。数字 i、ii、iii はネットワークレイヤーを示しています。数字 1、2、3、4 はEC2 インスタンスを示しています。インスタンスは、以下の図で iii と表示されている最下部レイヤー内のノードに接続します。複数のインスタンスが同じノードに接続できます。



この例では以下のようになっています。

- インスタンス 1 はレイヤー iii のネットワークノード 4 (NN4) に接続しています。NN4 はレイヤー ii のネットワークノード 2 (NN2) に接続しています。また NN2 はこの例でネットワーク階層の一番上にあるレイヤー i のネットワークノード 1 (NN1) に接続しています。ネットワークノードセットは NN1、NN2、NN4 で構成され、上位層から最下層まで階層的に示されています。
- インスタンス 2 はネットワークノード 4 (NN4) にも接続しています。インスタンス 1 とインスタンス 2 は同じネットワークノードセット (NN1、NN2、NN4) を共有しています。
- インスタンス 3 はネットワークノード 5 (NN5) に接続しています。NN5 は NN2 に接続しており、NN2 は NN1 に接続しています。インスタンス 3 に設定されているネットワークノードは NN1、NN2、NN5 です。
- インスタンス 4 はネットワークノード 6 (NN6) に接続しています。このネットワークノードセットは NN1、NN3、NN6 です。

インスタンス 1、2、3 の近接性を考えると、インスタンス 1 と 2 は同じネットワークノード (NN4) に接続しているため互いに近く、インスタンス 3 は別のネットワークノード (NN5) に接続しているため遠くなります。

この図のすべてのインスタンスの近接性を考えると、インスタンス 1、2、3 はネットワークノードセットで NN2 を共有しているため、インスタンス 4 よりも互いに近くなります。

原則として、いずれかの 2 つのインスタンスに接続されているネットワークノードが同じ場合、インスタンス 1 と 2 の場合と同様に、これらのインスタンスは互いに物理的に近くなります。さらに、ネットワークノード間のホップ数が少ないほど、インスタンスは互いに近くなります。例えば、インスタンス 1 と 3 ではインスタンス 4 との共通のネットワークノード (NN1) へよりも共通のネッ

トワークノード (NN2) への方がホップ数が少ないため、これらはインスタンス 4 よりも互いに近くなります。

この例ではネットワークノード 7 (NN7) ではインスタンスが実行されていないため、API 出力に NN7 は含まれません。

DescribeInstanceTopology の出力を解釈する方法

DescribeInstanceTopology API を使用してインスタンストポロジーを記述できます。出力ではインスタンスの基盤となるネットワークトポロジーが階層的に示されます。

次の出力例は上の図にある 4 つのインスタンスにおけるネットワークトポロジーの情報に対応します。この例のため、出力例にはコメントが含まれています。

出力に含まれる次の情報に注意してください。

- `NetworkNodes` では単一インスタンスのネットワークノードセットについて記述されます。
- 各ネットワークノードセットではネットワークノードは上から下に階層的に一覧表示されます。
- インスタンスに接続されているネットワークノードは一覧にある最後のネットワークノード (最下層) です。
- 互いに近いインスタンスを調べるにはまず、最下層にある共通のネットワークノードを見つけます。最下層に共通のネットワークノードがない場合、上位層で共通のネットワークノードを探します。

次の出力例で `i-1111111111example` と `i-2222222222example` は最下層に共通のネットワークノード `nn-4444444444example` があるため、この例における他のインスタンスと比較して互いに最も近い位置にあります。

> **Note**
>
> レスポンスには 3 つ以上のネットワークノードが含まれます。サポートされている各インスタンスタイプのレスポンスにあるネットワークノードの数については、「インスタンスのタイプ」を参照してください。

```
{
    "Instances": [
        {
            "InstanceId": "i-1111111111example",  //Corresponds to instance 1
```

```
            "InstanceType": "p4d.24xlarge",
            "GroupName": "ML-group",
            "NetworkNodes": [
                "nn-1111111111example",           //Corresponds to NN1 in layer i
                "nn-2222222222example",           //Corresponds to NN2 in layer ii
                "nn-4444444444example"            //Corresponds to NN4 in layer iii -
bottom layer, connected to the instance
            ],
            "CapacityBlockId": "null",
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        },
        {
            "InstanceId": "i-2222222222example",  //Corresponds to instance 2
            "InstanceType": "p4d.24xlarge",
            "NetworkNodes": [
                "nn-1111111111example",           //Corresponds to NN1 - layer i
                "nn-2222222222example",           //Corresponds to NN2 - layer ii
                "nn-4444444444example"            //Corresponds to NN4 - layer iii -
connected to instance
            ],
            "CapacityBlockId": "null",
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        },
        {
            "InstanceId": "i-3333333333example",  //Corresponds to instance 3
            "InstanceType": "trn1.32xlarge",
            "NetworkNodes": [
                "nn-1111111111example",           //Corresponds to NN1 - layer i
                "nn-2222222222example",           //Corresponds to NN2 - layer ii
                "nn-5555555555example"            //Corresponds to NN5 - layer iii -
connected to instance
            ],
            "CapacityBlockId": "null",
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        },
        {
            "InstanceId": "i-444444444example",  //Corresponds to instance 4
            "InstanceType": "trn1.2xlarge",
            "NetworkNodes": [
                "nn-1111111111example",          //Corresponds to NN1 - layer i
                "nn-3333333333example",          //Corresponds to NN3 - layer ii
```

```
                "nn-6666666666example"          //Corresponds to NN6 - layer iii -
 connected to instance
            ],
            "CapacityBlockId": "null",
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

DescribeCapacityReservationTopology の出力を解釈する方法

DescribeCapacityReservationTopology API を使用して、キャパシティ予約トポロジーを記述できます。出力ではリザーブドキャパシティの基盤となるネットワークトポロジーが階層的に示されます。

次の出力例は上の図のネットワークトポロジーの情報に対応します。この例のため、出力例にはコメントが含まれています。

出力に含まれる次の情報に注意してください。

- NetworkNodes は、単一のキャパシティ予約のネットワークノードセットを記述します。
- 各ネットワークノードセットではネットワークノードは上から下に階層的に一覧表示されます。
- キャパシティ予約に接続されているネットワークノードは一覧にある最後のネットワークノード (最下層) です。
- キャパシティ予約が互いに近いかどうかを調べるには、まず出力の最下層で共通のネットワークノードを見つけます。最下層に共通のネットワークノードがない場合、上位層で共通のネットワークノードを探します。

次の出力例では、cr-1111111111example は nn-2222222222example にあり、cr-2222222222exampleは nn-3333333333example にあります。キャパシティ予約は layer ii の異なるネットワークノード上にあるため、あるキャパシティ予約のインスタンスから別のキャパシティ予約のインスタンスへの通信は非効率になります。

> **Note**
>
> レスポンスには、キャパシティ予約のタイプに応じて 1 つ、2 つ、または 3 つのネットワークノードが含まれます。

```
{
    "CapacityReservations": [
        {
            "CapacityReservationId": "cr-1111111111example",
            "CapacityBlockId": "null",
            "State": "active",
            "InstanceType": "p4d.24xlarge",
            "NetworkNodes": [
                "nn-1111111111example",      //Corresponds to NN1 - layer i
                "nn-2222222222example"       //Corresponds to NN2 - layer ii
                // Visibility of additional nodes requires an instance launch and
                // the DescribeInstanceTopology API
            ],
            "AvailabilityZone": "us-west-2a"
        },
        {
            "CapacityReservationId": "cr-2222222222example",
            "CapacityBlockId": "null",
            "State": "active",
            "InstanceType": "trn1.2xlarge",
            "NetworkNodes": [
                "nn-1111111111example",      //Corresponds to NN1 - layer i
                "nn-3333333333example"       //Corresponds to NN3 - layer ii
                // Visibility of additional nodes requires an instance launch and
                // the DescribeInstanceTopology API
            ],
            "AvailabilityZone": "us-west-2a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

DescribeInstanceTopology と DescribeCapacityReservationTopology の違い

次の表は、DescribeInstanceTopology API と DescribeCapacityReservationTopology API の主な違いを比較しています。

比較ポイント	DescribeInstanceTopology	DescribeCapacityReservationTopology
使用フェーズ	起動後 (実行モード)	起動前 (プランニングおよび管理モード)
主な目的	実行中のインスタンスでのワークロードの最適化	インスタンス起動前のキャパシティプランニングとキャパシティ予約管理 (マージ、分割、割り当て)
ネットワークノードの数	実行中のインスタンスのすべてのノードを表示します。インスタンスがキャパシティ予約にある場合、最初のノードは対応するキャパシティ予約トポロジーと一致し、その後で追加のノードがインスタンスに接続します。	ノードの部分的なセットを表示します。これは、キャパシティ予約の状態 (pending または active) とタイプによって異なります。*
State	インスタンスは running の状態である必要があります。	キャパシティ予約は pending または active の状態である必要があります。
ユースケース	• ワークロードの最適化 • パフォーマンスチューニング • ランタイムトポロジーの分析	• 容量プランニング • キャパシティ予約の管理 (マージ/分割/割り当て) • 起動前のトポロジーの評価

* Ultraserver のキャパシティブロックの場合、active のキャパシティ予約またはその実行中のインスタンスのトポロジーを記述するとき、ネットワークノードセットは同じです。

Amazon EC2 トポロジーの前提条件

Amazon EC2 トポロジーを記述するには、インスタンスとキャパシティ予約が次の前提条件を満たしていることを確認します。

前提条件:

- AWS リージョン
- インスタンスのタイプ
- State
- IAM アクセス許可

AWS リージョン

サポートされている AWS リージョン:

- 米国東部 (バージニア北部)、米国東部 (オハイオ)、米国西部 (北カリフォルニア)、米国西部 (オレゴン)
- アフリカ (ケープタウン)
- アジアパシフィック (ジャカルタ)、アジアパシフィック (香港)、アジアパシフィック (ハイデラバード)、アジアパシフィック (メルボルン)、アジアパシフィック (ムンバイ)、アジアパシフィック (大阪)、アジアパシフィック (ソウル)、アジアパシフィック (シンガポール)、アジアパシフィック (シドニー)、アジアパシフィック (東京)
- カナダ (中部)
- 欧州 (フランクフルト)、欧州 (アイルランド)、欧州 (ロンドン)、欧州 (パリ)、欧州 (スペイン)、欧州 (ストックホルム)、欧州 (チューリッヒ)
- イスラエル (テルアビブ)
- 中東 (バーレーン)、中東 (アラブ首長国連邦)
- 南米 (サンパウロ)
- AWS GovCloud (米国西部)
- AWS European Sovereign Cloud (ドイツ)

DescribeCapacityReservationTopology API は、イスラエル (テルアビブ) および AWS GovCloud (米国西部) ではサポートされていません。

インスタンスのタイプ

サポートされるインスタンスタイプ:

- レスポンスで 3 つ*のネットワークノードを返す

  - g6e.xlarge | g6e.2xlarge | g6e.4xlarge | g6e.8xlarge | g6e.12xlarge | g6e.16xlarge | g6e.24xlarge | g6e.48xlarge | g7e.2xlarge | g7e.4xlarge | g7e.8xlarge | g7e.12xlarge | g7e.24xlarge | g7e.48xlarge
  - hpc6a.48xlarge | hpc6id.32xlarge | hpc7g.4xlarge | hpc7g.8xlarge | hpc7g.16xlarge | hpc7a.12xlarge | hpc7a.24xlarge | hpc7a.48xlarge | hpc7a.96xlarge | hpc8a.96xlarge
  - p3dn.24xlarge | p4d.24xlarge | p4de.24xlarge | p5.48xlarge | p5e.48xlarge | p5en.48xlarge | p6e-gb200.36xlarge
  - trn1.2xlarge | trn1.32xlarge | trn1n.32xlarge | trn2.48xlarge | trn2u.48xlarge
- レスポンスで 4 つ*のネットワークノードを返す
  - p6-b200.48xlarge | p6-b300.48xlarge

* 返されるネットワークノードの数は、DescribeInstanceTopology API を使用する場合にのみ適用されます。DescribeCapacityReservationTopology API の場合、返されるネットワークノードの数は、キャパシティ予約のタイプと状態によって異なります。

利用可能なインスタンスタイプはリージョンごとに異なります。詳細については、「リージョン別 Amazon EC2 インスタンスタイプ」を参照してください。

State

- DescribeInstanceTopology の場合 - インスタンスは running の状態である必要があります。
- DescribeCapacityReservationTopology の場合 – キャパシティ予約は pending または active の状態である必要があります。

他の状態のインスタンスまたはキャパシティ予約のトポロジー情報を取得することはできません。

IAM アクセス許可

IAM ID (ユーザー、ユーザーグループ、またはロール) には次のアクセス許可が必要です。

- ec2:DescribeInstanceTopology
- ec2:DescribeCapacityReservationTopology

Amazon EC2 インスタントポロジーの例

describe-instance-topology コマンドを使用して、EC2 インスタンスのトポロジーを記述できます。また、describe-capacity-reservation-topology コマンドを使用して、キャパシティ予約のトポロジーを記述できます。

パラメータやフィルターなしで `describe-instance-topology` コマンドまたは `describe-capacity-reservation-topology` コマンドを使用すると、指定したリージョン内のこのコマンドで利用可能なインスタンスタイプに一致する、すべてのインスタンスまたはキャパシティ予約 (使用するコマンドによる) が応答に含まれます。リージョンを設定するには `--region` パラメータを含めるかデフォルトのリージョンを設定できます。デフォルトのリージョンの設定についての詳細は「Amazon EC2 リソースのリージョンを選択する」を参照してください。

指定したインスタンス ID もしくはキャパシティ予約 ID、またはプレイスメントグループ名と一致するインスタンスまたはキャパシティ予約を返すパラメータを含めることができます。また、指定したインスタンスタイプやインスタンスファミリーに一致するインスタンスまたはキャパシティ予約、あるいは、指定したアベイラビリティーゾーンやローカルゾーン内のインスタンスまたはキャパシティ予約を返すフィルターを含めることもできます。1 つのパラメータまたはフィルター、もしくはパラメータとフィルターの組み合わせを含めることができます。

出力はページ分割されます。デフォルトでは、1 ページあたり最大 20のインスタンスまたはキャパシティ予約となっています。`--max-results` パラメータを使用すると、1 ページあたり最大 100 のインスタンスまたはキャパシティ予約を指定できます。

詳細については、「describe-instance-topology」および「describe-reservation-topology-topology」を参照してください。

必要なアクセス許可

以下のアクセス権限が必要です。

- `ec2:DescribeInstanceTopology` – インスタントポロジーの記述用
- `ec2:DescribeCapacityReservationTopology` – キャパシティ予約トポロジーの記述用

例

- 例 1: DescribeInstanceTopology - インスタンス ID
- 例 2: DescribeInstanceTopology - プレイスメントグループ名パラメータ
- 例 3: DescribeInstanceTopology - インスタンスタイプフィルター

- 例 3a — 指定したインスタンスタイプの完全一致フィルター
- 例 3b — インスタンスファミリーのワイルドカードフィルター
- 例 3c — インスタンスファミリーと完全一致フィルターの組み合わせ
- 例 4: DescribeInstanceTopology - ゾーン ID フィルター
  - 例 4a — アベイラビリティーゾーンフィルター
  - 例 4b — ローカルゾーンフィルター
  - 例 4c — アベイラビリティーゾーンフィルターとローカルゾーンフィルターの組み合わせ
- 例 5: DescribeInstanceTopology - インスタンスタイプとゾーン ID フィルター
- 例 6: DescribeCapacityReservationTopology - キャパシティ予約 ID
- 例 7: DescribeCapacityReservationTopology - インスタンスタイプフィルター

例 1: DescribeInstanceTopology - インスタンス ID

AWS CLI

特定のインスタンスのトポロジーを記述するには

`--instance-ids` パラメータを指定して、describe-instance-topology コマンドを使用します。出力には、指定したインスタンス ID と一致するインスタンスのみが含まれます。

```
aws ec2 describe-instance-topology \
    --region us-west-2 \
    --instance-ids i-1111111111example i-2222222222example
```

以下は出力の例です。

```
{
    "Instances": [
        {
            "InstanceId": "i-1111111111example",
            "InstanceType": "p4d.24xlarge",
            "GroupName": "ML-group",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3333333333example"
            ],
```

```
            "CapacityBlockId": "null",
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        },
        {
            "InstanceId": "i-2222222222example",
            "InstanceType": "trn1n.32xlarge",
            "GroupName": "HPC-group",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3214313214example"
            ],
            "CapacityBlockId": "null",
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

PowerShell

特定のインスタンスのトポロジーを記述するには

Get-EC2InstanceTopology コマンドレットを使用します。

```
Get-EC2InstanceTopology `
    -InstanceId i-1111111111example, i-2222222222example
```

例 2: DescribeInstanceTopology - プレイスメントグループ名パラメータ

AWS CLI

特定のプレイスメントグループ内にあるインスタンスのトポロジーを記述するには

group-names パラメータを指定して、describe-instance-topology コマンドを使用します。出力には、いずれかの指定したプレイスメントグループにあるインスタンスのみが含まれます。

```
aws ec2 describe-instance-topology \
    --region us-west-2 \
```

```
    --group-names ML-group HPC-group
```

以下は出力の例です。

```
{
    "Instances": [
        {
            "InstanceId": "i-1111111111example",
            "InstanceType": "p4d.24xlarge",
            "GroupName": "ML-group",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3333333333example"
            ],
            "CapacityBlockId": "null",
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        },
        {
            "InstanceId": "i-2222222222example",
            "InstanceType": "trn1n.32xlarge",
            "GroupName": "HPC-group",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3214313214example"
            ],
            "CapacityBlockId": "null",
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

PowerShell

特定のプレイスメントグループ内にあるインスタンスのトポロジーを記述するには

Get-EC2InstanceTopology コマンドレットを使用します。

```
Get-EC2InstanceTopology `
```

```
    -GroupName ML-group, HPC-group
```

例 3: DescribeInstanceTopology - インスタンスタイプフィルター

指定したインスタンスタイプ (完全一致)、またはインスタンスファミリーでフィルタリング (ワイルドカードを使用) できます。指定したインスタンスタイプフィルターとインスタンスファミリーのフィルターを組み合わせることもできます。

例 3a — 指定したインスタンスタイプの完全一致フィルター

AWS CLI

特定のインスタンスタイプを使用するインスタンスのトポロジーを記述するには

`instance-type` フィルターとともに describe-instance-topology CLI コマンドを使用します。出力には、指定したインスタンスタイプを使用するインスタンスのみが含まれます。

```
aws ec2 describe-instance-topology \
    --region us-west-2 \
    --filters Name=instance-type,Values=trn1n.32xlarge
```

以下は出力の例です。

```
{
    "Instances": [
        {
            "InstanceId": "i-2222222222example",
            "InstanceType": "trn1n.32xlarge",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3333333333example"
            ],
            "CapacityBlockId": "null",
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

PowerShell

特定のインスタンスタイプを使用するインスタンスのトポロジーを記述するには

Get-EC2InstanceTopology コマンドレットを使用します。

```
Get-EC2InstanceTopology `
    -Filter @{Name="instance-type"; Values="trn1n.32xlarge"}
```

例 3b — インスタンスファミリーのワイルドカードフィルター

AWS CLI

特定のインスタンスファミリーを使用するインスタンスのトポロジーを記述するには

instance-type フィルターとともに describe-instance-topology CLI コマンドを使用します。出力には、指定したインスタンスファミリーを使用するインスタンスのみが含まれます。

```
aws ec2 describe-instance-topology \
    --region us-west-2 \
    --filters Name=instance-type,Values=trn1*
```

以下は出力の例です。

```
{
    "Instances": [
        {
            "InstanceId": "i-2222222222example",
            "InstanceType": "trn1n.32xlarge",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3333333333example"
            ],
            "CapacityBlockId": "null",
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        },
        {
            "InstanceId": "i-3333333333example",
            "InstanceType": "trn1.32xlarge",
            "NetworkNodes": [
```

```
                "nn-1212121212example",
                "nn-1211122211example",
                "nn-1311133311example"
            ],
            "CapacityBlockId": "null",
            "ZoneId": "usw2-az4",
            "AvailabilityZone": "us-west-2d"
        },
        {
            "InstanceId": "i-444444444example",
            "InstanceType": "trn1.2xlarge",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-5434334334example",
                "nn-1235301234example"
            ],
            "CapacityBlockId": "null",
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

PowerShell

特定のインスタンスファミリーを使用するインスタンスのトポロジーを記述するには

Get-EC2InstanceTopology コマンドレットを使用します。

```
Get-EC2InstanceTopology `
    -Filter @{Name="instance-type"; Values="trn1*"}
```

例 3c — インスタンスファミリーと完全一致フィルターの組み合わせ

AWS CLI

インスタンスファミリーまたはインスタンスタイプを使用するインスタンスのトポロジを記述するには

`instance-type` フィルターとともに describe-instance-topology CLI コマンドを使用します。出力には、指定した条件を満たすインスタンスのみが含まれます。

```
aws ec2 describe-instance-topology \
    --region us-west-2 \
    --filters "Name=instance-type,Values=p4d*,trn1n.32xlarge"
```

以下は出力の例です。

```
{
    "Instances": [
        {
            "InstanceId": "i-1111111111example",
            "InstanceType": "p4d.24xlarge",
            "GroupName": "ML-group",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3333333333example"
            ],
            "CapacityBlockId": "null",
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        },
        {
            "InstanceId": "i-2222222222example",
            "InstanceType": "trn1n.32xlarge",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-4343434343example"
            ],
            "CapacityBlockId": "null",
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

PowerShell

インスタンスファミリーまたはインスタンスタイプを使用するインスタンスのトポロジを記述するには

Get-EC2InstanceTopology コマンドレットを使用します。

```
Get-EC2InstanceTopology `
    -Filter @{Name="instance-type"; Values="p4d*", "trn1n.32xlarge"}
```

例 4: DescribeInstanceTopology - ゾーン ID フィルター

zone-id フィルターを使用して、アベイラビリティーゾーンまたはローカルゾーンでフィルタリングできます。アベイラビリティーゾーンフィルターとローカルゾーンフィルターを組み合わせることもできます。

例 4a — アベイラビリティーゾーンフィルター

AWS CLI

特定のアベイラビリティーゾーンでインスタンスのトポロジーを記述するには

zone-id フィルターとともに describe-instance-topology CLI コマンドを使用します。出力には、指定したアベイラビリティーゾーンのインスタンスのみが含まれます。

```
aws ec2 describe-instance-topology \
    --region us-east-1 \
    --filters Name=zone-id,Values=use1-az1
```

以下は出力の例です。

```
{
    "Instances": [
        {
            "InstanceId": "i-2222222222example",
            "InstanceType": "trn1n.32xlarge",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3214313214example"
            ],
            "CapacityBlockId": "null",
            "ZoneId": "use1-az1",
            "AvailabilityZone": "us-east-1a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
```

```
}
```

PowerShell

特定のアベイラビリティーゾーンでインスタンスのトポロジーを記述するには

Get-EC2InstanceTopology コマンドレットを使用します。

```
Get-EC2InstanceTopology `
    -Filter @{Name="zone-id"; Values="use1-az1"}
```

例 4b — ローカルゾーンフィルター

AWS CLI

特定のローカルゾーンでインスタンスのトポロジーを記述するには

zone-id フィルターとともに describe-instance-topology CLI コマンドを使用します。出力には、指定したローカルゾーンのインスタンスのみが含まれます。

```
aws ec2 describe-instance-topology \
    --region us-east-1 \
    --filters Name=zone-id,Values=use1-atl2-az1
```

以下は出力の例です。

```
{
    "Instances": [
        {
            "InstanceId": "i-1111111111example",
            "InstanceType": "p4d.24xlarge",
            "GroupName": "ML-group",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3333333333example"
            ],
            "CapacityBlockId": "null",
            "ZoneId": "use1-atl2-az1",
            "AvailabilityZone": "us-east-1-atl-2a"
        }
```

```
    ],
    "NextToken": "SomeEncryptedToken"
}
```

PowerShell

特定のローカルゾーンでインスタンスのトポロジーを記述するには

Get-EC2InstanceTopology コマンドレットを使用します。

```
Get-EC2InstanceTopology `
    -Filter @{Name="zone-id"; Values="use1-atl2-az1"}
```

例 4c — アベイラビリティーゾーンフィルターとローカルゾーンフィルターの組み合わせ

AWS CLI

特定のゾーンにあるインスタンスのトポロジーを記述するには

zone-id フィルターとともに describe-instance-topology CLI コマンドを使用します。出力には、指定したゾーンのいずれかにあるインスタンスのみが含まれます。

```
aws ec2 describe-instance-topology \
    --region us-east-1 \
    --filters Name=zone-id,Values=use1-az1,use1-atl2-az1
```

以下は出力の例です。

```
{
    "Instances": [
        {
            "InstanceId": "i-1111111111example",
            "InstanceType": "p4d.24xlarge",
            "GroupName": "ML-group",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3333333333example"
            ],
            "CapacityBlockId": "null",
            "ZoneId": "use1-atl2-az1",
```

```
            "AvailabilityZone": "us-east-1-atl-2a"
        },
        {
            "InstanceId": "i-2222222222example",
            "InstanceType": "trn1n.32xlarge",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3214313214example"
            ],
            "CapacityBlockId": "null",
            "ZoneId": "use1-az1",
            "AvailabilityZone": "us-east-1a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

PowerShell

特定のゾーンにあるインスタンスのトポロジーを記述するには

Get-EC2InstanceTopology コマンドレットを使用します。

```
Get-EC2InstanceTopology `
    -Filter @{Name="zone-id"; Values="use1-az1", "use1-atl2-az1"}
```

例 5: DescribeInstanceTopology - インスタンスタイプとゾーン ID フィルター

1 つのコマンドで複数のフィルターを組み合わせることができます。

AWS CLI

特定のインスタンスタイプ、インスタンスファミリー、ゾーンを使用するインスタンスのトポロジーを記述するには

`instance-type` および `zone-id` フィルターとともに describe-instance-topology CLI コマンドを使用します。レスポンスには、指定したインスタンスタイプのいずれかを使用し、指定したゾーンのいずれかにあるインスタンスが含まれます。

```
aws ec2 describe-instance-topology \
```

```
    --region us-east-1 \
    --filters "Name=instance-type,Values=p4d*,trn1n.32xlarge" \
              "Name=zone-id,Values=use1-az1,use1-atl2-az1"
```

以下は出力の例です。

```
{
    "Instances": [
        {
            "InstanceId": "i-1111111111example",
            "InstanceType": "p4d.24xlarge",
            "GroupName": "ML-group",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3333333333example"
            ],
            "CapacityBlockId": "null",
            "ZoneId": "use1-atl2-az1",
            "AvailabilityZone": "us-east-1-atl-2a"
        },
        {
            "InstanceId": "i-2222222222example",
            "InstanceType": "trn1n.32xlarge",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3214313214example"
            ],
            "CapacityBlockId": "null",
            "ZoneId": "use1-az1",
            "AvailabilityZone": "us-east-1a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

PowerShell

特定のインスタンスタイプ、インスタンスファミリー、ゾーンを使用するインスタンスのトポロジーを記述するには

Get-EC2InstanceTopology コマンドレットを使用します。

```
Get-EC2InstanceTopology `
    -Filter @{Name="instance-type"; Values="p4d*", "trn1n.32xlarge"} `
            @{Name="zone-id"; Values="use1-az1", "use1-atl2-az1"}
```

例 6: DescribeCapacityReservationTopology - キャパシティ予約 ID

AWS CLI

特定のキャパシティ予約のトポロジーを記述するには

capacity-reservation-id パラメータを指定して、describe-capacity-reservation-topology コマンドを使用します。出力には、指定したキャパシティ予約 ID に一致するキャパシティ予約のみが含まれます。

```
aws ec2 describe-capacity-reservation-topology \
    --region us-east-1 \
    --capacity-reservation-id cr-1111111111example cr-2222222222example
```

以下は出力の例です。

```
{
    "CapacityReservations": [
        {
            "CapacityReservationId": "cr-1111111111example",
            "CapacityBlockId": "null",
            "State": "active",
            "InstanceType": "p5.48xlarge",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example"
            ],
            "AvailabilityZone": "us-east-1a"
        },
        {
            "CapacityReservationId": "cr-2222222222example",
            "CapacityBlockId": "null",
            "State": "active",
            "InstanceType": "p5en.48xlarge",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example"
```

```
            ],
            "AvailabilityZone": "us-east-1a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

PowerShell

特定のキャパシティ予約のトポロジーを記述するには

Get-EC2CapacityReservationTopology コマンドレットを使用します。

```
Get-EC2CapacityReservationTopology `
    -CapacityReservationId cr-1111111111example cr-2222222222example
```

例 7: DescribeCapacityReservationTopology - インスタンスタイプフィルター

指定したインスタンスタイプ (完全一致)、またはインスタンスファミリーでフィルタリング (ワイルドカードを使用) できます。指定したインスタンスタイプフィルターとインスタンスファミリーのフィルターを組み合わせることもできます。

AWS CLI

特定のインスタンスタイプのキャパシティ予約のトポロジーを記述するには

`instance-type` フィルターとともに describe-capacity-reservation-topology CLI コマンドを使用します。レスポンスには、指定したインスタンスタイプのインスタンスがすべて含まれます。

```
aws ec2 describe-capacity-reservation-topology \
    --region us-east-1 \
    --filters Name=instance-type,Values=p5en.48xlarge
```

以下は出力の例です。

```
{
    "CapacityReservations": [
        {
            "CapacityReservationId": "cr-2222222222example",
            "CapacityBlockId": "null",
```

```
            "State": "active",
            "InstanceType": "p5en.48xlarge",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example"
            ],
            "AvailabilityZone": "us-east-1a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

PowerShell

特定のインスタンスタイプのキャパシティ予約のトポロジーを記述するには

Get-EC2CapacityReservationTopology コマンドレットを使用します。

```
Get-EC2CapacityReservationTopology `
    -Filter @{Name="instance-type"; Values="p5en.48xlarge"}
```

Amazon EC2 インスタンスのプレイスメントグループ

ワークロードのニーズを対応するために、相互に依存する EC2 インスタンスのグループをプレイスメントグループ内に作成して、そのプレイスメントに影響を与えることができます。

ワークロードのタイプに応じて、以下のいずれかのプレイスメント戦略によりプレイスメントグループを作成できます。

- [クラスター] – アベイラビリティーゾーン内でインスタンスをまとめます。この戦略により、ワークロードは、ハイパフォーマンスコンピューティング (HPC) アプリケーションで典型的な緊密に組み合わされたノード間通信に必要な低レイテンシーネットワークパフォーマンスを実現できます。
- パーティション – インスタンスを複数の論理パーティションに分散させ、1 つのパーティション内のインスタンスのグループが基盤となるハードウェアを別のパーティション内のインスタンスのグループと共有しないようにします。この戦略は、Hadoop、Cassandra、Kafka などの大規模な分散および複製ワークロードで一般的に使用されます。
- スプレッド 相関性のエラーを減らすために、少数のインスタンスを基盤となるハードウェア全体に厳密に配置します。

- [精度時間] – AWS インフラストラクチャの高精度のタイムソースに直接アクセスを備えたサポートされているハードウェアにインスタンスを配置します。

プレイスメントグループは任意で選択します。インスタンスをリプレイスメントグループに作成しない場合、EC2 は、関連する障害を最小限に抑えるために、すべてのインスタンスが基盤となるハードウェア全体に分散されるような方法でインスタンスを配置しようとします。

料金

プレイスメントグループを作成するための料金は発生しません。

ルールと制限

プレイスメントグループを使用する前に、次のルールに注意してください。

- インスタンスは一度に 1 つのプレイスメントグループに配置することができます。1 つのインスタンスを複数のプレイスメントグループに配置することはできません。
- プレイスメントグループをマージすることはできません。
- オンデマンドキャパシティ予約およびゾーンリザーブドインスタンスを使用すると、アベイラビリティーゾーンの EC2 インスタンスに対してキャパシティを予約できます。インスタンスを起動するときに、インスタンス属性がオンデマンドキャパシティ予約またはゾーンリザーブドインスタンスで指定された属性と一致する場合、リザーブドキャパシティはインスタンスによって自動的に使用されます。これは、プレイスメントグループにインスタンスを起動する場合にも当てはまります。
- Dedicated Hosts をプレイスメントグループで起動することはできません。
- プレイスメントグループの中断時に停止または休止するように設定されたスポットインスタンスは起動できません。

内容

- プレイスメントグループのプレイスメント戦略
- EC2 インスタンスのプレイスメントグループを作成する
- EC2 インスタンスの配置を変更する
- プレイスメントグループの削除
- 共有プレイスメントグループ
- AWS Outposts のプレイスメントグループ

プレイスメントグループのプレイスメント戦略

EC2 インスタンスのプレイスメントグループは、次のいずれかのプレイスメント戦略を使用して作成できます。

プレイスメント戦略

- クラスタープレイスメントグループ
- パーティションプレイスメントグループ
- スプレッドプレイスメントグループ
- 精度タイムプレイスメントグループ

クラスタープレイスメントグループ

クラスタープレイスメントグループは、単一のアベイラビリティーゾーン内のインスタンスを論理的にグループ化したものです。インスタンスは単一ラックに分離されるものではありません。クラスタープレイスメントグループは、同じリージョン内の複数のピア接続 VPC にまたがることができます。同じクラスタープレイスメントグループ内のインスタンスは、TCP/IP トラフィックのフローあたりのスループット上限が高くなり、ネットワークの二分帯域幅の広い同じセグメントに配置されます。

次の図は、クラスタープレイスメントグループに配置されたインスタンスを示しています。

低いネットワークレイテンシー、高いネットワークスループット、またはその両方からメリットを受けるアプリケーションの場合は、クラスタープレイスメントグループの使用をお勧めします。また、ネットワークトラフィックの大部分がグループ内のインスタンス間で発生している場合にもお勧めします。プレイスメントグループで、最も低いレイテンシーと最も高いネットワークパフォーマンス

(1 秒あたりパケット数) を実現するためには、拡張ネットワーキングをサポートするインスタンスタイプを選択します。詳細については、拡張ネットワーキングを参照してください。

クラスタープレイスメントグループの作成時に親精度タイムプレイスメントグループを指定することで、クラスタープレイスメントグループで起動されたインスタンスが精度時間機能を持つハードウェアに配置できるようにします。クラスタープレイスメントグループを作成するときに `--parent-group-id` パラメータを使用して、この関係を確立します。

インスタンスは、次の方法で起動することをお勧めします。

- プレイスメントグループ内で必要な数のインスタンスを起動するには、1 つの起動リクエストを使用します。
- プレイスメントグループ内のすべてのインスタンスに同じインスタンスタイプを使用します。

後でプレイスメントグループにさらにインスタンスを追加しようとした場合、またはプレイスメントグループ内で複数のインスタンスタイプを起動しようとした場合、容量不足エラーが発生する可能性が高くなります。

プレイスメントグループ内のインスタンスを停止して再起動しても、そのインスタンスは同じプレイスメントグループ内で実行されます。ただし、インスタンスに対して十分な容量がない場合、起動は失敗します。

既にインスタンスを実行中のプレイスメントグループ内のインスタンスを起動するときに容量エラーを受け取った場合は、プレイスメントグループ内のすべてのインスタンスを停止して開始し、もう一度起動を試みてください。インスタンスを起動すると、すべてのリクエストしたインスタンスに応じた容量があるハードウェアにインスタンスが移行される場合があります。

ルールと制限

クラスタープレイスメントグループには、以下のルールが適用されます。

- 以下のインスタンスタイプがサポートされています。
  - 現行世代のインスタンス。バーストパフォーマンスインスタンス (T2 など)、Mac1 インスタンス、M7i-flex インスタンスを除く。
  - 以下は旧世代のインスタンスです: A1、C3、C4、I2、M4、R3、R4。
- クラスタープレイスメントグループを、複数のアベイラビリティーゾーンで設定することはできません。
- クラスタープレイスメントグループの 2 つのインスタンス間のトラフィックの最大ネットワークスループット速度は、2 つのインスタンスのうち遅い方に制限されます。高スループットの要件が

あるアプリケーションの場合、要件に適合するネットワーク接続を備えたインスタンスタイプを選択します。

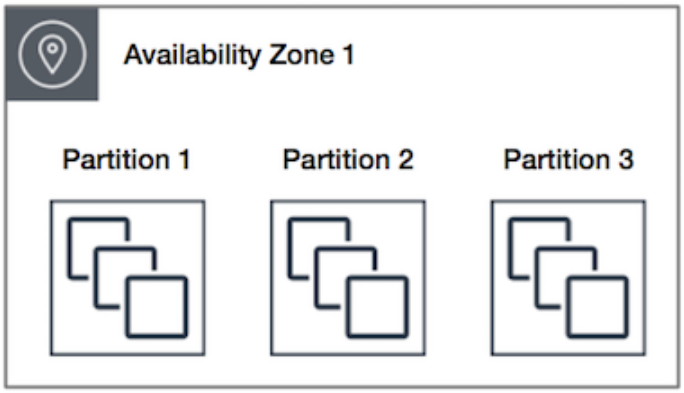
- 拡張ネットワーキングに対して有効になっているインスタンスには、以下のルールが適用されます。
  - クラスタープレイスメントグループ内のインスタンス間では、シングルフロートラフィックに最大 10 Gbps を使用できます。クラスタープレイスメントグループ内にないインスタンスは、シングルフロートラフィックに最大 5 Gbps を使用できます。
  - 同じリージョン内でのインスタンスと Amazon S3 バケットとの間では、パブリック IP アドレス空間または VPC エンドポイントを介したトラフィックに、使用可能なすべてのインスタンスの集計帯域幅を使用できます。
- 複数のインスタンスタイプをクラスタープレイスメントグループに起動できます。ただし、これにより起動に成功するために必要な容量が使用可能になる可能性が低くなります。クラスタープレイスメントグループ内ですべてのインスタンスで同じインスタンスタイプを使用することをお勧めします。
- クラスタープレイスメントグループ内にオンデマンドキャパシティ予約を作成することで、クラスタープレイスメントグループでキャパシティを明示的に予約することをお勧めします。ゾーンリザーブドインスタンスを使用してキャパシティを予約することはできないので注意してください。これは、ゾーンリザーブドインスタンスでは、プレイスメントグループでキャパシティを明示的に予約できないためです。
- インターネットへのネットワークトラフィックとオンプレミスリソースへの Direct Connect 接続は、クラスタープレイスメントグループに対して 5 Gbps に制限されます。

パーティションプレイスメントグループ

パーティションプレイスメントグループは、アプリケーションに関連するハードウェア障害の頻度を軽減するために役立ちます。パーティションプレイスメントグループを使用する場合、Amazon EC2 は各グループをパーティションと呼ばれる論理的なセグメントに分割します。Amazon EC2 では、プレイスメントグループ内の各パーティションにそれぞれ一連のラックがあります。各ラックには独自のネットワークおよび電源があります。プレイスメントグループ内のパーティションどうしが同じラックを共有することはありません。これにより、アプリケーション内でのハードウェア障害による影響を隔離できます。

次のイメージは、単一のアベイラビリティーゾーン内のパーティションプレイスメントグループのシンプルな描写を示しています。ここでは、3 つのパーティション (パーティション 1、パーティション 2、パーティション 3) があるパーティションプレイスメントグループに配置されたインスタンスを示しています。各パーティションは複数のインスタンスで構成されています。各パーティション内

のインスタンスは、他のパーティション内のラックを共有しないため、単一のハードウェア障害の影響は関連付けられたパーティションのみに留まります。



パーティションプレイスメントグループは、HDFS、HBase、Cassandra などの大規模な分散および複製ワークロードを異なるラック間でデプロイするために使用できます。インスタンスをパーティションプレイスメントグループに起動すると、Amazon EC2 は、指定したパーティション数全体にインスタンスを均等に分散しようとします。インスタンスを特定のパーティションに起動して、インスタンスの配置場所をより細かく制御することもできます。

パーティションプレイスメントグループは、同じリージョン内の複数のアベイラビリティーゾーンにパーティションを持つことができます。パーティションプレイスメントグループは、アベイラビリティーゾーンごとに最大 7 つのパーティションを持つことができます。パーティションプレイスメントグループで起動できるインスタンス数の制限は、アカウントの制限のみです。

また、パーティションプレイスメントグループでは各パーティションが可視化されるため、どのインスタンスがどのパーティションにあるかを確認できます。この情報は、HDFS、HBase、Cassandra などトポロジー対応アプリケーションと共有できます。これらのアプリケーションはこの情報を利用してインテリジェントなデータレプリケーションの決定を行い、データの可用性と耐久性を向上します。

パーティションプレイスメントグループでインスタンスを開始または起動し、リクエストを実行するための固有のハードウェアが不足している場合、そのリクエストは失敗します。Amazon EC2 では、時間の経過とともに、より明確なハードウェアを利用できるようになるため、後でリクエストを再試行できます。

ルールと制限

パーティションプレイスメントグループには、以下のルールが適用されます。

- パーティションプレイスメントグループは、アベイラビリティーゾーンごとに最大 7 つのパーティションをサポートします。パーティションプレイスメントグループで起動できるインスタンス数の制限は、アカウントの制限のみです。
- インスタンスをパーティションプレイスメントグループに起動すると、Amazon EC2 は、すべてのパーティションにインスタンスを均等に分散しようとします。Amazon EC2 では、すべてのパーティションにインスタンスが均等に分散されるとは限りません。
- ハードウェア専有インスタンス を持つパーティションプレイスメントグループは、最大 2 つのパーティションを持つことができます。
- [Capacity Reservations] (キャパシティー予約) を使用して、パーティションプレイスメントグループでキャパシティを予約することはできません。

スプレッドプレイスメントグループ

スプレッドプレイスメントグループは、それぞれ異なるハードウェアに配置されるインスタンスのグループです。

スプレッドプレイスメントグループは、少数の重要なインスタンスが互いに分離して保持される必要があるアプリケーションに推奨されます。スプレッドレベルのプレイスメントグループでインスタンスを起動すると、インスタンスが同じ機器を共有するときに発生し得る同時障害のリスクが軽減されます。スプレッドレベルのプレイスメントグループは、異なるハードウェアへのアクセスを提供するため、長時間のインスタンスタイプの混合やインスタンスの起動に適しています。

スプレッドプレイスメントグループでインスタンスを開始または起動し、リクエストを実行するための固有のハードウェアが不足している場合、そのリクエストは失敗します。Amazon EC2 では、時間の経過とともに、より明確なハードウェアを利用できるようになるため、後でリクエストを再試行できます。プレイスメントグループは、ラックまたはホスト全体でインスタンスを分散できます。ラックレベルのスプレッドプレイスメントグループは、AWS リージョンおよび AWS Outposts で使用できます。ホストレベルのスプレッドプレイスメントグループは、AWS Outposts を使用する場合にのみ使用できます。

ラックレベルのスプレッドプレイスメントグループ

次の図は、1 つのアベイラビリティーゾーン内の、スプレッドプレイスメントグループに配置された 7 つのインスタンスを示しています。7 つのインスタンスは、7 つの異なるラックに配置され、各ラックは独自のネットワークおよび電源を備えています。

ラックレベルのスプレッドプレイスメントグループは、同じリージョン内の複数のアベイラビリティーゾーンに分散できます。リージョンでは、ラックレベルのスプレッドプレイスメントグループについては、グループごとのアベイラビリティーゾーンごとに、最大 7 つの実行中のインスタンスを持つことができます。Outposts では、ラックレベルのスプレッドプレイスメントグループは、Outpost デプロイメント内のラックと同じ数のインスタンスを保持できます。

ホストレベルのスプレッドプレイスメントグループ

ホストレベルのスプレッドプレイスメントグループは、AWS Outposts を使用する場合にのみ使用できます。ホストスプレッドレベル配置グループは、Outpost デプロイメント内のホストと同じ数のインスタンスを保持できます。詳細については、「the section called “AWS Outposts のプレイスメントグループ”」を参照してください。

ルールと制限

スプレッドプレイスメントグループには、以下のルールが適用されます。

- ラックスプレッドプレイスメントグループは、アベイラビリティーゾーンごとに最大 7 つの実行インスタンスをサポートします。例えば、3 つのアベイラビリティーゾーンがあるリージョンでは、グループ内で合計 21 個のインスタンスを実行でき、各アベイラビリティーゾーンに 7 個のインスタンスがあります。同じアベイラビリティーゾーンと同じスプレッドプレイスメントグループで 8 番目のインスタンスを開始しようとすると、インスタンスは起動しません。アベイラビリティーゾーンに 7 個を超えるインスタンスが必要な場合は、複数のスプレッドプレイスメントグループを使用することをお勧めします。複数のプレイスメントグループに分散しても、グループ間でインスタンスが分散されるとは限りませんが、グループごとの分散が確実になされるようにできるため、特定の障害クラスからの影響は制限されます。
- ハードウェア専有インスタンス では、スプレッドプレイスメントグループはサポートされていません。
- ホストレベルのスプレッドプレイスメントグループは、AWS Outposts のプレイスメントグループでのみサポートされます。ホストレベルのスプレッドプレイスメントグループは、Outpost デプロイメント内のホストと同じ数のインスタンスを保持できます。

- リージョンでは、ラックレベルのスプレッドプレイスメントグループについては、グループごとのアベイラビリティーゾーンごとに、最大 7 つの実行中のインスタンスを持つことができます。AWS Outposts では、ラックレベルのスプレッドプレイスメントグループは、Outpost デプロイメント内のラックと同じ数のインスタンスを保持できます。
- [Capacity Reservations] (キャパシティー予約) を使用して、スプレッドプレイスメントグループでキャパシティを予約することはできません。

精度タイムプレイスメントグループ

精度タイムプレイスメントグループは、AWS インフラストラクチャの高精度のタイムソースに直接アクセス可能でサポートされているハードウェアにインスタンスを配置します。精度タイムプレイスメントグループで起動されたインスタンスは、拡張 Amazon Time Sync Service にアクセスできます。これにより、すべての Amazon EC2 インスタンスで利用可能な標準のローカル NTP エンドポイントと比較して、より正確なローカルタイムソースが提供されます。

精度タイムプレイスメントグループは、マイクロ秒単位の正確なクロック同期を必要とするアプリケーション (トランザクションの順序付けと整合性を必要とする分散データベース、正確なタイムスタンプを必要とする金融サービス、ローカルクロック読み取りを使用してイベントを順序付けする分散システムなど) に推奨されます。

精度タイムプレイスメントグループのインスタンスには、次の利点があります。

- 精度が向上した拡張ローカル NTP タイムソースは、追加の設定なしでサポートされているすべてのオペレーティングシステムに利用できます。
- マイクロ秒精度同期の PTP ハードウェアクロック (PHC) デバイスへのアクセス (Linux インスタンスのみ)。
- ナノ秒分解能のネットワーク測定値のハードウェアパケットタイムスタンプ (Linux インスタンスのみ)。

拡張タイムソースを使用するようにインスタンスを設定する方法については、「EC2 インスタンスのタイムリファレンスを、ローカル Amazon Time Sync Service を使用するように設定します。」を参照してください。

精度タイムプレイスメントグループには、次のルールが適用されます。

ルールと制限

- サポートされているインスタンスファミリーの現在のリストについては、「EC2 インスタンスのタイムリファレンスを、ローカル Amazon Time Sync Service を使用するように設定します。」を参照してください。
- 精度タイムプレイスメントグループでインスタンスを開始または起動し、拡張 Amazon Time Sync Service へのアクセスを提供するハードウェアが不足している場合、リクエストは失敗します。後でもう一度やり直すか、別のアベイラビリティーゾーンで試してください。
- クラスタープレイスメントグループの親である精度タイムプレイスメントグループは削除できません。
- 精度タイムプレイスメントグループの使用には、追加料金はかかりません。

EC2 インスタンスのプレイスメントグループを作成する

プレイスメントグループを使用して、インスタンス同士の相対的な配置を制御できます。プレイスメントグループを作成したら、そのプレイスメントグループ内でインスタンスを起動できます。

制限

リージョンあたり最大 500 個のプレイスメントグループを作成できます。

Console

プレイスメントグループを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Placement Groups] を選択します。
3. [プレイスメントグループを作成] を選択します。
4. [名前] には、グループの名前を入力します。
5. [配置戦略] には、グループに[クラスター]、[スプレッド]、[パーティション]、[精度時間] のいずれかの配置戦略を選択します。
   a. [スプレッド] を選択した場合、[スプレッドレベル] で [ホスト] または [ラック] を選択する必要があります。
   b. [パーティション] を選択した場合、[パーティションの数] でグループのパーティションの数を入力します。

c. [クラスター] を選択した場合は、オプションとして親精度タイムプレイスメントグループを指定して、クラスタープレイスメントグループのインスタンスが精度時間対応ハードウェアで起動できるようにします。

6. (オプション) タグを追加するには、[タグ] で [新しいタグを追加] を選択し、キーと値を入力します。
7. [グループの作成] を選択してください。

AWS CLI

create-placement-group コマンドを使用します。

クラスタープレイスメントグループを作成するには

次の例では、cluster プレイスメント戦略を使用するプレイスメントグループを作成し、キー purpose と値 production を持つタグを適用します。

```
aws ec2 create-placement-group \
    --group-name my-cluster \
    --strategy cluster \
    --tag-specifications 'ResourceType=placement-
group,Tags={Key=purpose,Value=production}'
```

精度タイムプレイスメントグループの親を使用してクラスタープレイスメントグループを作成するには

次の例では、--parent-group-id パラメータを使用して親精度タイムプレイスメントグループを指定するクラスタープレイスメントグループを作成します。

```
aws ec2 create-placement-group \
    --group-name my-cluster \
    --strategy cluster \
    --parent-group-id pg-0aaa1111111111111
```

パーティションプレイスメントグループを作成するには

次の例では、partition プレイスメント戦略を使用するプレイスメントグループを作成し、--partition-count パラメータを使用して 5 つのパーティションを指定します。

```
aws ec2 create-placement-group \
```

```
    --group-name HDFS-Group-A \
    --strategy partition \
    --partition-count 5
```

精度タイムプレイスメントグループを作成するには

次の例では、precision-time プレイスメント戦略を使用するプレイスメントグループを作成します。

```
aws ec2 create-placement-group \
    --group-name my-precision-time-pg \
    --strategy precision-time
```

PowerShell

New-EC2PlacementGroup コマンドレットを使用します。

クラスタープレイスメントグループを作成するには

以下の例では、クラスタープレイスメントグループを作成しています。

```
New-EC2PlacementGroup `
    -GroupName my-placement-group `
    -Strategy cluster
```

精度タイムプレイスメントグループの親を使用してクラスタープレイスメントグループを作成するには

次の例では、-ParentGroupId パラメータを使用して親精度タイムプレイスメントグループを指定するクラスタープレイスメントグループを作成します。

```
New-EC2PlacementGroup `
    -GroupName my-placement-group `
    -Strategy cluster `
    -ParentGroupId pg-0aaa1111111111111
```

パーティションプレイスメントグループを作成するには

以下の例では、パーティションプレイスメントグループを作成しています。

```
New-EC2PlacementGroup `
```

```
    -GroupName my-placement-group `
    -Strategy partition `
    -PartitionCount 5
```

精度タイムプレイスメントグループを作成するには

次の例では、精度タイムプレイスメントグループを作成しています。

```
New-EC2PlacementGroup `
    -GroupName my-precision-time-pg `
    -Strategy precision-time
```

EC2 インスタンスの配置を変更する

インスタンスのプレイスメントグループは、次の方法で変更できます。

- プレイスメントグループにインスタンスを追加する
- プレイスメントグループ間でインスタンスを移動する
- プレイスメントグループからインスタンスを削除する

要件

インスタンスのプレイスメントグループを変更する前に、インスタンスを stopped 状態にする必要があります。

Console

インスタンスの配置を変更するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択します。
4. [Actions (アクション)]、[Instance settings (インスタンスの設定)]、[Modify instance placement (インスタンスの配置の変更)] の順に選択します。
5. [プレイスメントグループ] で、次のいずれかを実行します。
   - プレイスメントグループにインスタンスを追加するには、プレイスメントグループを選択します。

- プレイスメントグループ間でインスタンスを移動するには、プレイスメントグループを選択します。
- プレイスメントグループからインスタンスを削除するには、[なし] を選択します。

6. [保存] を選択します。

AWS CLI

プレイスメントグループにインスタンスを移動するには

以下の modify-instance-placement コマンドを使用します。

```
aws ec2 modify-instance-placement \
    --instance-id i-0123a456700123456 \
    --group-name MySpreadGroup
```

プレイスメントグループからインスタンスを削除するには

以下の modify-instance-placement コマンドを使用します。プレイスメントグループ名に空の文字列を指定すると、現在のプレイスメントグループからインスタンスが削除されます。

```
aws ec2 modify-instance-placement \
    --instance-id i-0123a456700123456 \
    --group-name ""
```

PowerShell

プレイスメントグループにインスタンスを移動するには

Edit-EC2InstancePlacement cmdlet を使用し、プレイスメントグループの名前を指定します。

```
Edit-EC2InstancePlacement `
    -InstanceId i-0123a456700123456 `
    -GroupName MySpreadGroup
```

プレイスメントグループからインスタンスを削除するには

Edit-EC2InstancePlacement cmdlet を使用し、プレイスメントグループの名前に空の文字列を指定します。

```
Edit-EC2InstancePlacement `
    -InstanceId i-0123a456700123456 `
    -GroupName ""
```

プレイスメントグループの削除

プレイスメントグループを交換する必要がある場合、または不要になった場合は、そのプレイスメントグループを削除できます。削除するプレイスメントグループにはインスタンスが含まれていないことが必要です。インスタンスを終了し、そのインスタンスを別のプレイスメントグループに移動するか、プレイスメントグループから削除することができます。

クラスタープレイスメントグループの親であるプレイスメントグループは削除できません。最初にクラスタープレイスメントグループを削除します。

Console

プレイスメントグループを削除するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Placement Groups] を選択します。
3. プレイスメントグループを選択し、[Actions (アクション)]、[Delete (削除)] の順に選択します。
4. 確認を求められたら、**Delete**と入力し、[削除] を選択します。

AWS CLI

プレイスメントグループを削除するには

delete-placement-group コマンドを使用します。

```
aws ec2 delete-placement-group --group-name my-cluster
```

PowerShell

プレイスメントグループを削除するには

Remove-EC2PlacementGroup コマンドレットを使用します。

```
Remove-EC2PlacementGroup -GroupName my-cluster
```

共有プレイスメントグループ

プレイスメントグループを共有すると、別々の AWS アカウントが所有し、相互に依存しているインスタンスの配置に影響を与えることができます。所有者は、プレイスメントグループを複数の AWS アカウント 間で共有したり、自分の組織内で共有したりできます。参加者は、自分のアカウントと共有されているプレイスメントグループでインスタンスを起動できます。

プレイスメントグループの所有者は、プレイスメントグループを次の人と共有できます。

- 組織内または組織外の特定の AWS アカウント
- 組織内の組織単位
- 組織全体

VPC ピアリングを使用して別の AWS アカウントが所有するインスタンスを接続することで、共有クラスタープレイスメントグループが提供するレイテンシーの利点を最大限に活用できます。

内容

- ルールと制限
- 必要なアクセス許可
- アベイラビリティーゾーン間での共有
- プレイスメントグループの共有
- プレイスメントグループの共有解除

ルールと制限

プレイスメントグループを共有する場合、またはプレイスメントグループが自分と共有される場合は、次のルールと制限が適用されます。

- プレイメントグループを共有するには、AWS アカウント内で所有している必要があります。自分に共有されているプレイスメントグループは共有できません。
- パーティションまたはスプレッドプレイスメントグループを共有しても、プレイスメントグループの制限は変わりません。共有パーティションプレイスメントグループは、アベイラビリティーゾー

ンごとに最大 7 つのパーティションをサポートし、共有スプレッドプレイスメントグループは、アベイラビリティーゾーンごとに最大 7 つの実行インスタンスをサポートします。

- ユーザーの組織や組織内の組織単位とプレイスメントグループを共有するには、AWS Organizations との共有を有効にする必要があります。詳細については、「AWS リソースの共有」を参照してください。
- AWS マネジメントコンソールを使用してインスタンスを起動する場合は、自分と共有されていたプレイスメントグループを選択できます。AWS CLI を使用してインスタンスを起動する場合は、名前ではなく ID で共有プレイスメントグループを指定する必要があります。プレイスメントグループの名前は、ユーザーが、共有プレイスメントグループの所有者である場合にのみ使用できます。
- 共有プレイスメントグループで所有するインスタンスを管理する責任はお客様にあります。
- 共有プレイスメントグループに関連付けられているが、自分が所有していないインスタンスやキャパシティ予約を表示または変更することはできません。
- プレイスメントグループの Amazon リソースネーム (ARN) には、プレイスメントグループを所有しているアカウントの ID が含まれています。プレイスメントグループ ARN のアカウント ID 部分を使用することで、自分に共有されているプレイスメントグループの所有者を特定することができます。

必要なアクセス許可

プレイスメントグループを共有するには、ユーザーに次のアクションのアクセス許可が必要です。

- `ec2:PutResourcePolicy`
- `ec2:DeleteResourcePolicy`

アベイラビリティーゾーン間での共有

リソースがリージョンの複数のアベイラビリティーゾーンに分散されるようにするために、アベイラビリティーゾーンは各 アカウントの名前に個別にマッピングされます。このため、アカウントが異なると、アベイラビリティーゾーンの命名方法が異なる場合があります。例えば、`us-east-1a` アカウントのアベイラビリティーゾーン AWS の場所は、別の `us-east-1a` アカウントのアベイラビリティーゾーン AWS の場所と異なる可能性があります。

自己のアカウントを基準にして専有ホストの場所を指定するには、アベイラビリティーゾーン ID (AZ ID) を使用する必要があります。AZ ID は、すべての AWS アカウントで同じアベイラビリティーゾーンを一貫して示すための一意の識別子です。例えば、`use1-az1` は `us-east-1` リー

ジョンのアベイラビリティーゾーン ID であり、すべての AWS アカウントで同じ場所を示します。詳細については、「AZ ID」を参照してください。

プレイスメントグループの共有

プレイスメントグループを共有するには、リソース共有に追加する必要があります。リソース共有とは、AWS RAM アカウント間で自身のリソースを共有するための AWS リソースです。リソース共有では、共有対象のリソースと、共有先のコンシューマーを指定します。

AWS Organizations の組織に属しており、組織内での共有が有効化されている場合、組織内のコンシューマーに対し、共有プレイスメントグループへのアクセス権が付与されます。

プレイスメントグループが、組織外の AWS アカウントと共有されている場合、AWS アカウント所有者はリソース共有に参加するための招待状を受け取ります。招待を承諾すると、共有プレイスメントグループにアクセスできます。

プレイスメントグループは、AWS Resource Access Manager を使用して、AWS アカウント間で共有できます。詳細については、「AWS RAM ユーザーガイド」の「Creating a resource share」を参照してください。

プレイスメントグループの共有解除

プレイスメントグループの所有者は、共有プレイスメントグループをいつでも共有解除することができます。共有プレイスメントグループの共有を解除すると、次の変更が発生します。

- プレイスメントグループを共有していた AWS アカウントでは、インスタンスを起動したり、容量を予約したりできなくなります。
- 共有プレイスメントグループで実行されているインスタンスはすべて、プレイスメントグループとの関連付けが解除されますが、AWS アカウントでは引き続き実行されます。
- 共有プレイスメントグループのキャパシティ予約はプレイスメントグループとの関連付けが解除されますが、AWS アカウントで引き続き使用できます。

詳細については、「AWS RAM ユーザーガイド」の「Deleting a resource share」を参照してください。

AWS Outposts のプレイスメントグループ

AWS Outposts は、AWS のインフラストラクチャ、サービス、API、ツールをお客様のオンプレミスまで拡張するフル管理されたサービスです。AWS は、AWS Outposts 管理されたインフラストラク

チャへのローカルアクセスを提供することで、AWS リージョンと同じプログラミングインターフェイスを使用してオンプレミスでアプリケーションを構築して実行できるようにします。同時に、コンピューティングとストレージのローカルリソースを使用して、レイテンシーを短縮し、ローカルのデータ処理ニーズに対応します。

Outpost とはお客様のサイトにデプロイされる AWS のコンピューティングおよびストレージキャパシティーのプールです。AWS はAWS リージョンの一部としてこのキャパシティーを運営、監視、管理します。

ユーザーは、自分のアカウントで作成した Outposts にプレイスメントグループを作成できます。これにより、自分のサイトにある Outpost において、基盤となるハードウェア全体でインスタンスを分散できるようになります。通常のアベイラビリティーゾーンでプレイスメントグループを作成して使用するのと同じ方法で、Outposts でプレイスメントグループを作成して使用します。Outpost で分散戦略を使用してプレイスメントグループを作成する場合、プレイスメントグループがホストまたはラック全体でインスタンスを分散するように選択できます。ホスト全体でインスタンスを分散すると、単一ラックの Outpost で分散戦略を使用できます。

考慮事項

- ラックレベルのスプレッドプレイスメントグループは、Outpost デプロイメント内のラックと同じ数のインスタンスを保持できます。
- ホストレベルのスプレッドプレイスメントグループは、Outpost デプロイメント内のホストと同じ数のインスタンスを保持できます。

前提条件

Outpost は、自分のサイトにインストールする必要があります。詳細については、AWS Outposts ユーザーガイドのOutpost を作成し、Outpost 容量を注文するを参照してください。

Outpost でプレイスメントグループを使用するには

1. Outpost にサブネットを作成します。詳細については、AWS Outposts ユーザーガイドのサブネットの作成を参照してください。
2. Outpost の関連付けられたリージョンでプレイスメントグループを作成します。スプレッド戦略を使用してプレイスメントグループを作成する場合は、ホストまたはラックレベルのスプレッドを選択して、Outpost の基盤となるハードウェア全体にグループがインスタンスを分散する方法を決定できます。詳細については、「the section called “プレイスメントグループの作成”」を参照してください。

3. プレイスメントグループにインスタンスを起動します。[Subnet] (サブネット) には、ステップ 1 で作成したサブネットを選択し、[Placement group name] (プレイスメントグループ名) には、ステップ 2 で作成したプレイスメントグループを選択します。詳細については、AWS Outposts ユーザーガイドの、Outposts でインスタンスを起動するを参照してください。

EC2 インスタンスのネットワークの最大送信単位 (MTU)

ネットワーク接続の最大送信単位 (MTU) とは接続を介して渡すことができる最大許容パケットサイズ (バイト単位) です。接続の MTU が大きいほど、より多くのデータを単一のパケットで渡すことができます。イーサネットフレームはパケット (送信している実際のデータ) とそれを囲むネットワークオーバーヘッド情報で構成されています。

イーサネットフレームの形式はさまざまで、最も一般的な形式は標準イーサネット v2 フレーム形式です。これはインターネットのほとんどでサポートされている最大のイーサネットパケットサイズである 1500 MTU をサポートします。インスタンスでサポートされている最大 MTU はインスタンスタイプによって異なります。

EC2 インスタンスタイプはすべて 1500 MTU をサポートしています。

内容

- ジャンボフレーム (9001 MTU)
- パス MTU 検出
- Amazon EC2 インスタンスの MTU を設定する
- トラブルシューティング

ジャンボフレーム (9001 MTU)

ジャンボフレームではパケットあたりのペイロードサイズを拡張し、それによってパケットオーバーヘッド以外のパケットの割合を高めることができます。ジャンボフレームでは、同じ量の使用可能なデータを少ないパケットで送信することができます。ただし、特定のタイプのトラフィックには、次の最大ペイロードが適用されます。

MTU 制限 1500 バイト

- インターネットゲートウェイ経由のトラフィック
- VPN 接続経由のトラフィック

- AWS リージョン間のトラフィック (トランジットゲートウェイが使用されている場合を除く)

MTU 制限 8500 バイト

- リージョン間 VPC ピアリング接続経由のトラフィック

パケットが MTU 制限を超える場合は、フラグメント化されます。また、`Don't Fragment` フラグが IP ヘッダーに設定されている場合は削除されます。

ジャンボフレームを、インターネットバウンドトラフィックや VPC を出るトラフィックに使用する場合には慎重に行ってください。パケットは中間システムによってフラグメント化されるため、このトラフィックの速度が低下します。VPC 外に向かうトラフィックの速度を低下させずに VPC 内のジャンボ フレームを使用するにはルートごとに MTU サイズを設定するか、または MTU サイズやルートの異なる複数の Elastic ネットワークインターフェイスを使用します。

クラスタープレイスメントグループ内にコロケーションされたインスタンスでは考えられる最大のネットワークスループットの実現するうえでジャンボフレームが役立ちます。この場合はジャンボフレームを使用することが推奨されています。詳細については Amazon EC2 インスタンスのプレイスメントグループを参照してください。

Direct Connect を経由した VPC とオンプレミスのネットワーク間のトラフィックにはジャンボフレームを使用できます。詳細と、ジャンボフレーム機能の検証方法については、Direct Connect ユーザーガイドの「プライベート仮想インターフェイスまたはトランジット仮想インターフェイスの MTU」を参照してください。

現行世代のインスタンスはすべて、ジャンボフレームをサポートしています。以下の旧世代のインスタンスタイプは、A1、C3、I2、M3、R3 のジャンボフレームをサポートしています。

関連リソース

- NAT ゲートウェイについては「Amazon VPC ユーザーガイド」の「NAT ゲートウェイの基本」を参照してください。
- トランジットゲートウェイの詳細については「Amazon VPC Transit Gateway ユーザーガイド」の「Maximum transmission unit」を参照してください。
- ローカルゾーンについては「AWS ローカルゾーンユーザーガイド」の「考慮事項」を参照してください。
- AWS Wavelength については「AWS Wavelength ユーザーガイド」の「Maximum transmission unit」を参照してください。

- Outposts については「AWS Outposts ユーザーガイド」の「Service link maximum transmission unit requirements」を参照してください。

パス MTU 検出

2 つのデバイス間のパス MTU を判断するために、パス MTU 検出 (PMTUD) が使用されます。パス MTU は送信側ホストと受信側ホスト間のパスでサポートされている最大のパケットサイズです。2 つのホスト間のネットワークで MTU サイズに違いがある場合、PMTUD は受信側ホストが ICMP メッセージで送信側ホストに応答するのを可能にします。この ICMP メッセージは送信側ホストがネットワークパスに沿って最低の MTU サイズを使用し、リクエストを再送信するように指示します。このネゴシエーションがないと、リクエストが大きすぎて受信側ホストが受け取れないため、パケットドロップが発生する可能性があります。

IPv4 の場合、ホストがパスに沿って送信するパケットが、受信側ホストの MTU、あるいはデバイスの MTU よりも大きな場合、受信側ホストまたはデバイスはそのパケットをドロップし、次のような ICMP メッセージ `Destination Unreachable: Fragmentation Needed and Don't Fragment was Set` (タイプ 3、コード 4) を返します。このメッセージは送信側ホストに対し、ペイロードを複数の小さなパケットに分割し再送信することを指示します。

IPv6 プロトコルはネットワークのフラグメンテーションをサポートしていません。ホストがパスに沿って送信するパケットが、受信側ホストの MTU、あるいはデバイスの MTU よりも大きな場合、受信側ホストまたはデバイスはそのパケットをドロップし、次のような ICMP メッセージ `ICMPv6 Packet Too Big (PTB)` (タイプ 2) を返します。このメッセージは送信側ホストに対し、ペイロードを複数の小さなパケットに分割し再送信することを指示します。

NAT ゲートウェイやロードバランサーなどの一部のコンポーネントを介して行われる接続は自動追跡されます。つまり、セキュリティグループの追跡はアウトバウンド接続を試みると自動的に有効になります。接続が自動追跡される場合、追跡されていない場合、またはセキュリティグループのルールでインバウンド ICMP トラフィックが許可されている場合は PMTUD 応答を受信できます。

サブネットへの ICMP トラフィックを拒否するネットワークアクセスコントロールリストのエントリがある場合など、セキュリティグループレベルでトラフィックが許可されている場合でも、ICMP トラフィックはブロックされる可能性があることに注意してください。

> **Important**
>
> パス MTU 検出はジャンボフレームが一部のルーターによって破棄されないことを保証するものではありません。VPC のインターネットゲートウェイでは最大 1500 バイトのパケット

だけが転送されます。インターネットトラフィックではMTU が 1500 のパケットが推奨されています。

NAT ゲートウェイの MTU ルールについては「Amazon VPC ユーザーガイド」の「最大送信単位 (MTU)」を参照してください。Transit Gateway の MTU ルールについては「AWS Transit Gateway ユーザーガイド」の「最大送信単位 (MTU)」を参照してください。

Amazon EC2 インスタンスの MTU を設定する

ネットワーク接続の最大送信単位 (MTU) とは接続を介して渡すことができる最大許容パケットサイズ (バイト単位) です。すべての Amazon EC2 インスタンスで標準フレーム (1500 MTU) がサポートされており、すべての現行世代のインスタンスタイプでジャンボフレーム (9001 MTU) がサポートされています。

Amazon EC2 インスタンスの MTU を表示したり、インスタンスと別のホストの間のパス MTU を表示したり、標準フレームまたはジャンボフレームを使用するようにインスタンスを設定したりできます。

タスク

- 2 つのホスト間のパス MTU の確認
- インスタンスの MTU を確認する
- インスタンスの MTU を設定する

2 つのホスト間のパス MTU の確認

EC2 インスタンスと別のホストとの間のパス MTU を確認できます。宛先として DNS 名または IP アドレスを指定できます。宛先が別の EC2 インスタンスの場合、そのセキュリティグループによりインバウンド UDP トラフィックが許可されていることを確認します。

使用する手順はインスタンスのオペレーティングシステムによって異なります。

Linux インスタンス

インスタンスで tracepath コマンドを実行して、EC2 インスタンスと指定された宛先の間のパス MTU を確認します。このコマンドは `iputils` パッケージの一部であり、多くの Linux ディストリビューションでデフォルトで使用できます。

次の例ではEC2 インスタンスと amazon.com との間のパス MTU を確認しています。

```
[ec2-user ~]$ tracepath amazon.com
```

この出力例ではパス MTU は 1500 となっています。

```
 1?: [LOCALHOST]      pmtu 9001
 1:  ip-172-31-16-1.us-west-1.compute.internal (172.31.16.1)   0.187ms pmtu 1500
 1:  no reply
 2:  no reply
 3:  no reply
 4:  100.64.16.241 (100.64.16.241)                              0.574ms
 5:  72.21.222.221 (72.21.222.221)                             84.447ms asymm 21
 6:  205.251.229.97 (205.251.229.97)                           79.970ms asymm 19
 7:  72.21.222.194 (72.21.222.194)                             96.546ms asymm 16
 8:  72.21.222.239 (72.21.222.239)                             79.244ms asymm 15
 9:  205.251.225.73 (205.251.225.73)                           91.867ms asymm 16
...
31:  no reply
     Too many hops: pmtu 1500
     Resume: pmtu 1500
```

Windows インスタンス

mturoute を使用してパス MTU を確認するには

1. https://elifulkerson.com/projects/mturoute.php から EC2 インスタンスに mturoute.exe をダウンロードします。
2. コマンドプロンプトウィンドウを開いて、mturoute.exe をダウンロードしたディレクトリに移動します。
3. 次のコマンドを使用して、EC2 インスタンスと指定した宛先との間のパス MTU を確認します。次の例ではEC2 インスタンスと www.elifulkerson.com との間のパス MTU を確認しています。

   ```
   .\mturoute.exe www.elifulkerson.com
   ```

   この出力例ではパス MTU は 1500 となっています。

   ```
   * ICMP Fragmentation is not permitted. *
   * Speed optimization is enabled. *
   ```

```
* Maximum payload is 10000 bytes. *
+ ICMP payload of 1472 bytes succeeded.
- ICMP payload of 1473 bytes is too big.
Path MTU: 1500 bytes.
```

インスタンスの MTU を確認する

インスタンスの MTU 値を確認できます。一部のインスタンスではジャンボフレームを使用し、それ以外のドライバには標準フレームサイズを使用するように設定されています。

使用する手順はインスタンスのオペレーティングシステムによって異なります。

Linux インスタンス

Linux インスタンス上の MTU 設定を確認するには

EC2 インスタンスで次の ip コマンドを実行します。プライマリネットワークインターフェイスが eth0 でない場合はeth0 を使用しているネットワークインターフェイスに置き換えます。

```
[ec2-user ~]$ ip link show eth0
```

この出力例で、*mtu 9001* はこのインスタンスにジャンボフレームが使用されていることを示しています。

```
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc pfifo_fast state UP mode
 DEFAULT group default qlen 1000
    link/ether 02:90:c0:b7:9e:d1 brd ff:ff:ff:ff:ff:ff
```

Windows インスタンス

使用する手順はインスタンスのドライバーによって異なります。

ENA driver

バージョン 2.1.0 以降

MTU 値を取得するにはEC2 インスタンスで次の Get-NetAdapterAdvancedProperty コマンドを使用します。ワイルドカード (アスタリスク) を使用して、すべてのイーサネット名を取得します。インターフェイス名 *JumboPacket の出力を確認します。値 9015 はジャンボフレームが有効であることを示します。ジャンボフレームはデフォルトで無効化されています。

```
Get-NetAdapterAdvancedProperty -Name "Ethernet*"
```

バージョン 1.5 以前

MTU 値を取得するにはEC2 インスタンスで次の Get-NetAdapterAdvancedProperty コマンドを使用します。インターフェイス名 MTU の出力を確認します。値 9001 はジャンボフレームが有効であることを示します。ジャンボフレームはデフォルトで無効化されています。

```
Get-NetAdapterAdvancedProperty -Name "Ethernet"
```

Intel SRIOV 82599 driver

MTU 値を取得するにはEC2 インスタンスで次の Get-NetAdapterAdvancedProperty コマンドを使用します。インターフェイス名 *JumboPacket のエントリを確認します。値 9014 はジャンボフレームが有効であることを示します。(MTU のサイズにはヘッダーとペイロードが含まれる点に注意してください)。ジャンボフレームはデフォルトで無効化されています。

```
Get-NetAdapterAdvancedProperty -Name "Ethernet"
```

AWS PV driver

MTU 値を取得するにはEC2 インスタンスで次のコマンドを使用します。インターフェイスの名前は異なる場合があります。出力では「Ethernet」、「Ethernet 2」、または 「Local Area Connection」 という名前のエントリを探してください。ジャンボフレームを有効または無効にするにはインターフェイス名が必要です。値 9001 はジャンボフレームが有効であることを示します。

```
netsh interface ipv4 show subinterface
```

インスタンスの MTU を設定する

VPC 内のネットワークトラフィックにジャンボフレームを使用したり、インターネットトラフィックに標準フレームを使用したりする場合があるでしょう。どのようなユースケースでも、インスタンスが想定どおりに動作することを確認することをお勧めします。

使用する手順はインスタンスのオペレーティングシステムによって異なります。

Linux インスタンス

Linux インスタンス上の MTU 値を設定するには

1. インスタンスで次の ip コマンドを実行します。このコマンドで目的の MTU 値が 1500 に設定されますが、代わりに 9001 を使用することができます。プライマリネットワークインターフェイスが eth0 でない場合は eth0 を実際のネットワークインターフェイスに置き換えます。

```
[ec2-user ~]$ sudo ip link set dev eth0 mtu 1500
```

2. (オプション) 再起動後もネットワーク MTU 設定を維持するにはオペレーティングシステムのタイプに基づいて、次の設定ファイルを変更します。

   - Amazon Linux 2023 – 設定ファイルの [Link] セクションを変更します。デフォルトの設定ファイルは /usr/lib/systemd/network/80-ec2.network です。または、/run/systemd/network/ で作成された任意のカスタム設定ファイルを更新できます。ファイル名は *priority*-*interface*.network です。詳細については、Amazon Linux ドキュメントの「Networking service」を参照してください。

     ```
     MTUBytes=1500
     ```

   - Amazon Linux 2 – 次の行を /etc/sysconfig/network-scripts/ifcfg-*eth0* ファイルに追加します。

     ```
     MTU=1500
     ```

     次の行を /etc/dhcp/dhclient.conf ファイルに追加します。

     ```
     request subnet-mask, broadcast-address, time-offset, routers, domain-name,
      domain-search, domain-name-servers, host-name, nis-domain, nis-servers, ntp-
     servers;
     ```

   - その他の Linux ディストリビューション – それぞれに固有のドキュメントを参照してください。

3. (オプション) インスタンスを再起動し、MTU 設定が正しいことを確認します。

Windows インスタンス

使用する手順はインスタンスのドライバーによって異なります。

ENA driver

インスタンスでデバイスマネージャーまたは Set-NetAdapterAdvancedProperty コマンドを使用して、MTU を変更できます。

バージョン 2.1.0 以降

ジャンボフレームを有効にするには次のコマンドを使用します。

```
Set-NetAdapterAdvancedProperty -Name "Ethernet" -RegistryKeyword "*JumboPacket" -
RegistryValue 9015
```

ジャンボフレームを無効にするには次のコマンドを使用します。

```
Set-NetAdapterAdvancedProperty -Name "Ethernet" -RegistryKeyword "*JumboPacket" -
RegistryValue 1514
```

バージョン 1.5 以前

ジャンボフレームを有効にするには次のコマンドを使用します。

```
Set-NetAdapterAdvancedProperty -Name "Ethernet" -RegistryKeyword "MTU" -
RegistryValue 9001
```

ジャンボフレームを無効にするには次のコマンドを使用します。

```
Set-NetAdapterAdvancedProperty -Name "Ethernet" -RegistryKeyword "MTU" -
RegistryValue 1500
```

Intel SRIOV 82599 driver

インスタンスでデバイスマネージャーまたは Set-NetAdapterAdvancedProperty コマンドを使用して、MTU を変更できます。

ジャンボフレームを有効にするには次のコマンドを使用します。

```
Set-NetAdapterAdvancedProperty -Name "Ethernet" -RegistryKeyword "*JumboPacket" -
RegistryValue 9014
```

ジャンボフレームを無効にするには次のコマンドを使用します。

```
Set-NetAdapterAdvancedProperty -Name "Ethernet" -RegistryKeyword "*JumboPacket" -
RegistryValue 1514
```

AWS PV driver

インスタンスで netsh コマンドを使用することで MTU を変更できます。デバイスマネージャーでは MTU を変更できません。

ジャンボフレームを有効にするには次のコマンドを使用します。

```
netsh interface ipv4 set subinterface "Ethernet" mtu=9001
```

ジャンボフレームを無効にするには次のコマンドを使用します。

```
netsh interface ipv4 set subinterface "Ethernet" mtu=1500
```

トラブルシューティング

ジャンボフレームを使用したときに EC2 インスタンスと Amazon Redshift クラスターとの間で接続の問題が発生した場合は「Amazon Redshift 管理ガイド」の「クエリがハングしたようになる」を参照してください。

EC2 インスタンスの仮想プライベートクラウド

Amazon Virtual Private Cloud (Amazon VPC) を使用すると、AWS クラウドで論理的に分離された独自の領域内に、仮想プライベートクラウド (VPC) と呼ばれる仮想ネットワークを定義できます。AWS のリソース (Amazon EC2 インスタンスなど) を VPC のサブネット内に作成できます。VPC は、お客様自身のデータセンターで運用されている従来のネットワークによく似ていますが、AWS からスケーラブルなインフラストラクチャを使用できるというメリットがあります。お客様の VPC はお客様が設定できます。IP アドレスレンジの選択、サブネットの作成、ルートテーブル、ネットワークゲートウェイ、セキュリティの設定ができます。VPC のインスタンスをインターネットまたは独自のデータセンターに接続できます。

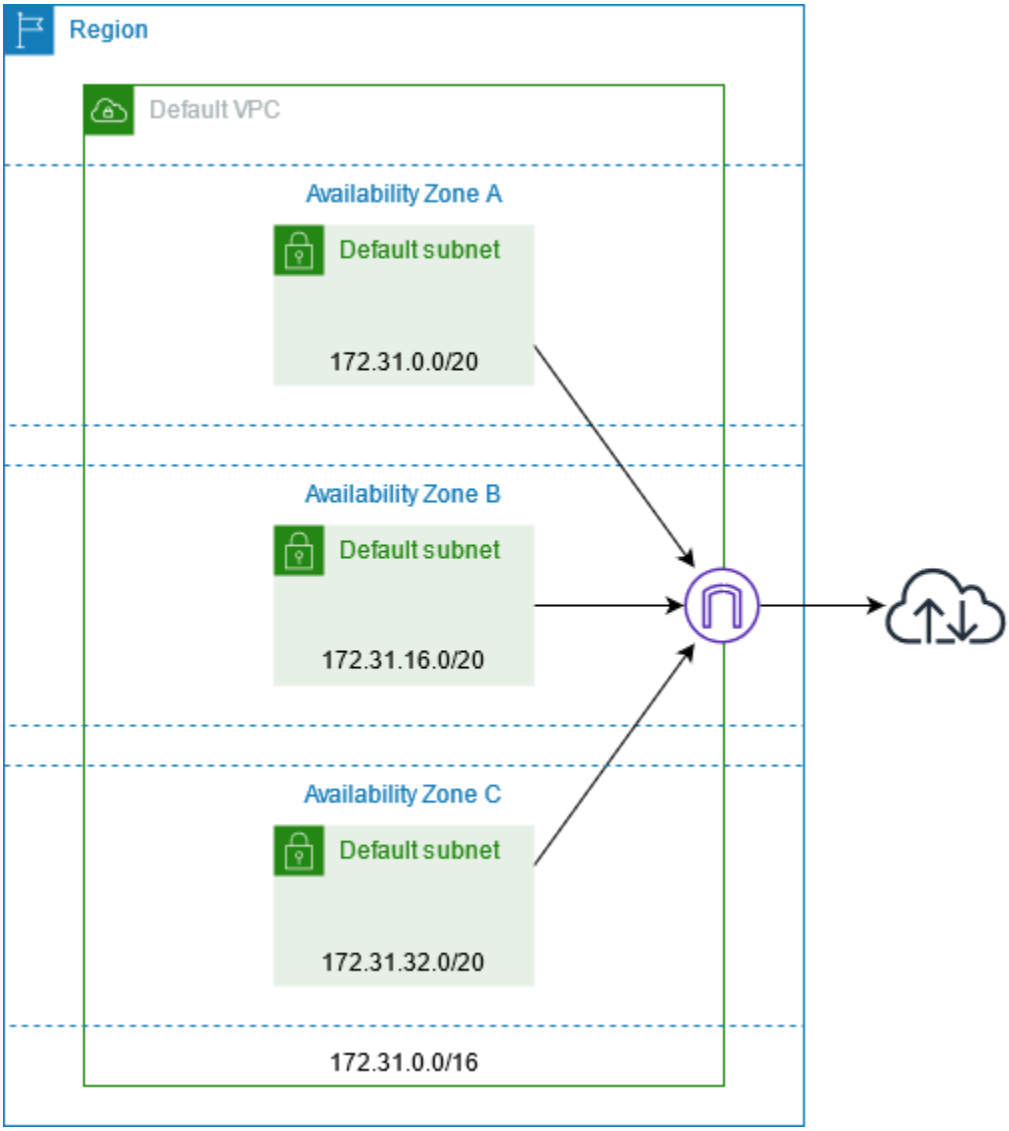
内容

- デフォルトの VPC
- 非デフォルト VPC

- インターネットアクセス
- 共有サブネット
- IPv6 専用サブネット

デフォルトの VPC

AWS アカウントを作成すると、各リージョンにデフォルト VPC が作成されます。デフォルトの VPC は、設定済みですぐに使用できる VPC です。例えば、それぞれのデフォルトの VPC では、各アベイラビリティーゾーンがデフォルトのサブネットを持ちます。この VPC には、インターネットゲートウェイがアタッチされ、メインルートテーブルでは、すべて (0.0.0.0/0) のトラフィックをインターネットゲートウェイに送信するルートが定義されています。必要に応じて、デフォルト VPC の設定を変更できます。例えば、サブネットとルートテーブルを追加できます。



非デフォルト VPC

リソースにデフォルト VPC を使用する代わりに、「Amazon VPC ユーザーガイド」の「VPC の作成」の説明に従って独自の VPC を作成できます。

EC2 インスタンスの VPC を作成する際には、考慮すべき点がいくつかあります。

- IPv4 CIDR ブロックには、デフォルトの提案を使用することも、アプリケーションまたはネットワークに必要な CIDR ブロックを入力することもできます。
- 高可用性を確保するには、複数のアベイラビリティーゾーンにサブネットを作成します。
- インスタンスをインターネットからアクセスできるようにするには、次のいずれかを実行します。
  - インスタンスをパブリックサブネットに配置できる場合は、パブリックサブネットを追加します。両方の DNS オプションを有効に設定したままにします。今すぐまたは後で、オプションでプライベートサブネットを追加することができます。
  - インスタンスをプライベートサブネットに配置する必要がある場合は、プライベートサブネットのみを追加します。NAT ゲートウェイを追加すると、プライベートサブネット内のインスタンスにインターネットアクセスを提供できます。インスタンスがアベイラビリティーゾーン間で大量のトラフィックを送受信する場合は、各アベイラビリティーゾーンに NAT ゲートウェイを作成します。それ以外の場合は、1 つのアベイラビリティーゾーンのみに NAT ゲートウェイを作成し、クロスゾーントラフィックを送受信するインスタンスを NAT ゲートウェイと同じアベイラビリティーゾーンで起動できます。

インターネットアクセス

デフォルト VPC はパブリック IP アドレスと DNS ホスト名を割り当てるように設定され、メインルートテーブルには VPC にアタッチされたインターネットゲートウェイへのルートが設定されているため、デフォルト VPC のデフォルトサブネットで起動されたインスタンスは、インターネットにアクセスすることが可能です。

デフォルト以外のサブネットおよび VPC で起動するインスタンスでは、以下のいずれかのオプションを使用することで、そのサブネットで起動したインスタンスをインターネットにアクセスできるようにすることができます。

- インターネットゲートウェイを設定します。詳細については、「Amazon VPC ユーザーガイド」の「インターネットゲートウェイを使用してインターネットに接続する」 を参照してください。
- パブリックな NAT ゲートウェイを設定します。詳細については、Amazon VPC ユーザーガイドのプライベートサブネットからインターネットにアクセスするを参照してください。

共有サブネット

EC2 インスタンスを共有 VPC サブネットで起動するときは、次の点に注意してください:

- 参加者は、共有サブネットの ID を指定することで、共有サブネットでインスタンスを実行できます。参加者は、指定するネットワークインターフェイスを所有する必要があります。
- 参加者は、共有サブネットで作成したインスタンスを起動、停止、終了、記述できます。参加者は、VPC 所有者が共有サブネットで作成したインスタンスを起動、停止、終了、記述できません。
- VPC 所有者は、参加者が共有サブネットで作成したインスタンスを起動、停止、終了、記述できません。
- 参加者は、EC2 Instance Connect Endpoint を使用して共有サブネット内のインスタンスに接続できます。参加者は、共有サブネットに EC2 Instance Connect Endpoint を作成する必要があります。参加者は、VPC 所有者が共有サブネットで作成した EC2 Instance Connect Endpoint を使用できません。

共有 Amazon EC2 リソースの詳細については、以下を参照してください。

- the section called “組織または OU の AMI 共有を管理する”
- the section called “共有キャパシティ予約”
- the section called “共有プレイスメントグループ”
- クロスアカウント Amazon EC2 専有ホストの共有

共有サブネットの詳細については、「Amazon VPC ユーザーガイド」の「VPC を他のアカウントと共有する」を参照してください。

IPv6 専用サブネット

IPv6 のみのサブネットで起動される EC2 インスタンスは、IPv6 アドレスを受信しますが、IPv4 アドレスは受信しません。IPv6 のみのサブネットで起動するインスタンスは、Nitro ベースのインスタンスである必要があります。

セカンダリネットワーク

セカンダリネットワークは、特殊なネットワークユースケース専用の仮想ネットワークです。これらのネットワークは、AWS クラウドのパーティション内で論理的に分離されています。セカンダリ

ネットワーク内には、セカンダリサブネットなどのリソースを作成できます。セカンダリネットワークは Amazon VPC と密接に結合されています。このため、一部のインスタンスはマルチホーム化されており、VPC とセカンダリネットワークの両方で起動されます。

現在、セカンダリネットワークは一部のインスタンスタイプで、コミットメントが長期的なキャパシティ予約を通じて利用できます。セカンダリネットワークがワークロードに役立つとお考えの場合は、担当のアカウントチームにお問い合わせください。

内容

- セカンダリネットワークとは
- 主要なコンセプト
- アーキテクチャ
- その他の考慮事項
- 開始方法
- セカンダリネットワークリソースの管理
- ネットワーク設計のベストプラクティス
- トラブルシューティング
- クォータと制限

セカンダリネットワークとは

分離された論理ネットワークを提供するセカンダリネットワークは VPC ネットワークと組み合わせて使用されるため、インスタンスが 2 つの独立したネットワークにマルチホーム化されます。セカンダリネットワークのメリットには以下が含まれます。

- ML ワークロードの East-West 接続などの特殊なユースケースやプロトコル向けの高性能ネットワーキング
- 論理的分離によるマルチテナントサポート
- VPC および AWS サービスとのシームレスなインスタンス統合

主要なコンセプト

セカンダリネットワーク

IPv4 CIDR ブロック (/28 ~ /12) に基づく論理レイヤー 3 ネットワークを提供する地域的なネットワーキングコンストラクト。セカンダリネットワークは、物理的にパーティション化されたネットワークインフラストラクチャ上で VPC とは別個に動作します。

セカンダリサブネット

セカンダリネットワーク内にあるアベイラビリティーゾーン固有のコンストラクトで、VPC サブネットに類似しています。セカンダリサブネットは、/28 から /12 までの CIDR ブロックをサポートします。

セカンダリインターフェイス

セカンダリネットワークカードにアタッチされたネットワークインターフェイスで、セカンダリサブネット内で East-West 接続を提供します。これらのインターフェイスは、Elastic Network Interface (ENI) から物理的かつ論理的に分離されています。

アーキテクチャ

セカンダリネットワークをサポートする EC2 インスタンスはマルチホーム化されます。これは、インスタンスが VPC 内とセカンダリネットワーク内の両方で同時に通信できることを意味します。

- VPC: AWS サービス、ストレージ、データベース、ネットワークサービス、インターネットに対する North-South TCP/IP 接続を提供
- セカンダリネットワーク: サポートされている特殊なインスタンス間の East-West 接続を提供

その他の考慮事項

- セカンダリインターフェイスは RunInstances 経由で管理されており、個別に作成または削除することはできません。
- インスタンスの起動後にセカンダリインターフェイスをアタッチ/デタッチすることはできません。
- 起動後にセカンダリインターフェイスの IP アドレスを変更することはできません。
- セカンダリネットワークでは、セキュリティグループ、NACL、フローログなどの VPC 機能がサポートされていません。

開始方法

前提条件

セカンダリネットワークを使用するインスタンスを起動する前に、対象リージョン内に VPC が設定されており、EC2 キャパシティの対象アベイラビリティーゾーン内にサブネットが設定されていることも確認してください。

ステップ 1: セカンダリネットワークを作成する

VPC と同一のリージョン内にセカンダリネットワークを作成します。これは、RDMA トラフィックを論理的に分離するリージョンリソースです。

```
aws ec2 create-secondary-network \
  --network-type rdma \
  --ipv4-cidr-block 172.31.0.0/16 \
  --region us-east-2
```

パラメータ:

- `--network-type`: ネットワークタイプ (現在サポートされているのは rdma のみです)
- `--ipv4-cidr-block`: /28 から /12 までの IPv4 CIDR ブロック
- `--region`: AWS リージョン (US-East-2)

> **Note**
>
> ベストプラクティス: インスタンスレベルでのルーティングを簡素化するため、VPC CIDR と重複しない CIDR ブロックを選択してください。

ステップ 2: セカンダリサブネットを作成する

VPC サブネットと同一のアベイラビリティーゾーン内にセカンダリサブネットを作成します。これは、アベイラビリティーゾーン固有のリソースです。

```
aws ec2 create-secondary-subnet \
  --secondary-network-id sn-1234567890abcdef0 \
  --ipv4-cidr-block 172.31.24.0/24 \
```

```
  --availability-zone us-east-2a
```

> **Note**
>
> IP アドレス予約: VPC サブネットと同様に、Amazon は各セカンダリサブネット内の最初の 4 つの IP アドレスと最後の IP アドレスを内部使用のために予約します。

ステップ 3: インスタンスを起動する

VPC サブネットとセカンダリサブネットの両方にインスタンスを起動します。インスタンスは、両方のネットワークに接続するマルチホーム型になります。

```
aws ec2 run-instances \
  --image-id ami-12345678 \
  --count 1 \
  --instance-type <instance> \
  --key-name MyKeyPair \
  --instance-market-options '{"MarketType": "capacity-block"}' \
  --capacity-reservation-specification '{"CapacityReservationTarget": \
  {"CapacityReservationId": "cr-1234567890abcdef0"}}' \
  --network-interfaces \
    "NetworkCardIndex=0,DeviceIndex=0,Groups=sg-12345678,\
    SubnetId=subnet-0987654321fedcba0,InterfaceType=interface" \
  --secondary-interfaces \
    "NetworkCardIndex=1,DeviceIndex=0,SecondarySubnetId=ss-98765421yxz,\
    InterfaceType=secondary,PrivateIpAddressCount=1,DeleteOnTermination=true", \
    "NetworkCardIndex=2,DeviceIndex=0,SecondarySubnetId=ss-98765421yxz,\
    InterfaceType=secondary,PrivateIpAddressCount=1,DeleteOnTermination=true", \
    "NetworkCardIndex=3,DeviceIndex=0,SecondarySubnetId=ss-98765421yxz,\
    InterfaceType=secondary,PrivateIpAddressCount=1,DeleteOnTermination=true", \
    "NetworkCardIndex=4,DeviceIndex=0,SecondarySubnetId=ss-98765421yxz,\
    InterfaceType=secondary,PrivateIpAddressCount=1,DeleteOnTermination=true", \
    "NetworkCardIndex=5,DeviceIndex=0,SecondarySubnetId=ss-98765421yxz,\
    InterfaceType=secondary,PrivateIpAddressCount=1,DeleteOnTermination=true", \
    "NetworkCardIndex=6,DeviceIndex=0,SecondarySubnetId=ss-98765421yxz,\
    InterfaceType=secondary,PrivateIpAddressCount=1,DeleteOnTermination=true", \
    "NetworkCardIndex=7,DeviceIndex=0,SecondarySubnetId=ss-98765421yxz,\
    InterfaceType=secondary,PrivateIpAddressCount=1,DeleteOnTermination=true", \
    "NetworkCardIndex=8,DeviceIndex=0,SecondarySubnetId=ss-98765421yxz,\
    InterfaceType=secondary,PrivateIpAddressCount=1,DeleteOnTermination=true"
```

主要パラメータ:

- `--network-interfaces`: VPC 接続用のプライマリ Nitro ENI を指定します (ネットワークカードインデックス 0)
- `--secondary-interfaces`: セカンダリサブネット内の East-West 接続用の 8 個のセカンダリインターフェイスを指定します (ネットワークカードインデックス 1～8)
- `InterfaceType=secondary`: セカンダリインターフェイスを示すものです

セカンダリネットワークリソースの管理

セカンダリネットワークの記述

セカンダリネットワークの詳細を確認します。

```
aws ec2 describe-secondary-networks \
  --secondary-network-id sn-1234567890abcdef0
```

セカンダリサブネットの記述

セカンダリサブネットの詳細を確認します。

```
aws ec2 describe-secondary-subnets \
  --secondary-subnet-id ss-98765421yxz
```

セカンダリインターフェイスの記述

インスタンスにアタッチされたセカンダリネットワークインターフェイスの詳細を確認します。

```
aws ec2 describe-secondary-interfaces \
  --filters "Name=attachment.instance-id,Values=i-1234567890abcdef0"
```

リソースの削除

セカンダリサブネットを削除します。

```
aws ec2 delete-secondary-subnet \
  --secondary-subnet-id ss-98765421yxz
```

セカンダリネットワークを削除します。

```
aws ec2 delete-secondary-network \
  --secondary-network-id sn-1234567890abcdef0
```

> **Important**
>
> セカンダリネットワークを削除する前に、すべてのインスタンスを終了し、すべてのセカンダリサブネットを削除しておく必要があります。

ネットワーク設計のベストプラクティス

CIDR 計画

CIDR の重複を避ける: セカンダリネットワークは VPC から物理的に分離されていますが、重複しない CIDR ブロックを使用することで、インスタンスのオペレーティングシステムレベルでのルーティング設定を簡素化できます。

> **Note**
>
> Amazon は、サブネットごとに 5 つの IP アドレスを予約します。

トラフィックの分離

セカンダリネットワークを使用して分離する: 異なるプロジェクト、チーム、またはセキュリティ境界ごとに個別のセカンダリネットワークを作成します。セカンダリネットワークは、インスタンス間を論理的に分離します。インスタンスが異なるセカンダリネットワーク間で通信することはできません。

複数のサブネットを使用する: セカンダリネットワーク内で複数のセカンダリサブネットを使用して、GPU インデックス、アベイラビリティーゾーン、またはワークロードタイプ別にトラフィックをセグメント化します。例えば、一般的なアーキテクチャパターンは、4 個または 8 個のセカンダリサブネットを持つ単一のセカンダリネットワークをデプロイすることです。この場合、各セカンダリサブネットは共通のインデックスを持つ GPU のグループに紐付けられます。

トラブルシューティング

インスタンス起動の失敗

問題: セカンダリネットワークインターフェイスを使用するインスタンスの起動が失敗する。

解決方法:

- AMI に適切なドライバーサポートが含まれていることを確認する
- セカンダリサブネットに使用可能な IP アドレスが十分に設定されていることを確認する
- キャパシティ予約が「アクティブ」状態であることを確認する
- セカンダリサブネットが VPC サブネットと同じアベイラビリティーゾーン内にあることを確認する

接続の問題

問題: インスタンス間の RDMA 接続を確立できない。

解決方法:

- すべてのインスタンスが同一のセカンダリネットワークとセカンダリサブネット内にあることを確認する
- セカンダリインターフェイスのドライバーがインスタンスに適切にロードされていることを確認する
- アプリケーションが正しいネットワークインターフェイスにバインドされていることを確認する
- 同一のセカンダリサブネット内にあるインスタンスにダイレクトルート経由でアクセスできる。DHCP 経由で提供される静的ルートを通じてサブネット間通信を実行できる。

API エラー

問題: セカンダリネットワークおよびセカンダリサブネットオペレーションに対する API コールが失敗する。

解決方法:

- ec2:CreateSecondaryNetwork、ec2:CreateSecondarySubnet などに対する IAM 許可を確認する
- CIDR ブロックがサポートされている範囲内 (/28 ~ /12) であることを確認する

- 正しいリージョンとアベイラビリティーゾーンを使用していることを確認する

クォータと制限

クォータの引き上げをリクエストするには、AWS Service Quotas を使用するか、AWS サポートにお問い合わせください。

セカンダリネットワークのクォータと制限

リソース	制限	引き上げ可能
リージョンあたりのセカンダリネットワーク数	5	はい
セカンダリネットワークあたりのセカンダリサブネット数	200	はい
CIDR ブロックサイズ	/28 ~ /12	いいえ

セキュリティとコンプライアンスの目標を満たすように Amazon EC2 を設定し、Amazon EC2 リソースの保護に役立つ他の サービスの使用方法を学びます。

AWS ではクラウドのセキュリティが最優先事項です。AWS のお客様はセキュリティを最も重視する組織の要件を満たすように構築されたデータセンターとネットワークアーキテクチャから利点を得られます。

セキュリティはAWS とお客様の間の共有責任です。責任共有モデルではこれをクラウドのセキュリティおよびクラウド内のセキュリティと説明しています。

- クラウドのセキュリティ - AWS は AWS Cloud で AWS のサービスを実行するインフラストラクチャを保護する責任を負います。また、AWS は使用するサービスを安全に提供します。AWSコンプライアンスプログラムの一環として、サードパーティーの監査が定期的にセキュリティの有効性をテストおよび検証しています。Amazon EC2 に適用するコンプライアンスプログラムの詳細についてはAWSコンプライアンスプログラムによる対象範囲の
- クラウド内のセキュリティ - お客様は以下の事項について責任を負います。
  - VPC とセキュリティグループの設定など、インスタンスへのネットワークアクセスの制御。詳細についてはネットワークトラフィックの制御を参照してください。
  - インスタンスへの接続に使用する認証情報の管理。
  - ゲスト OS と、ゲスト OS にデプロイされたソフトウェア (更新およびセキュリティパッチを含む) の管理。詳細についてはAmazon EC2 Windows インスタンスの更新管理を参照してください。
  - インスタンスにアタッチされた IAM ロールと、それらのロールに関連付けられたアクセス許可の設定。詳細についてはAmazon EC2 の IAM ロールを参照してください。

このドキュメントは Amazon EC2 使用時における責任共有モデルの適用法を理解するのに役立ちます。ここではセキュリティやコンプライアンスに関する目標を達成できるように Amazon EC2 を設定する方法について説明します。Amazon EC2リソースのモニタリングやセキュリティ確保に役立つ他の AWS サービスの使用方法についても説明します。

目次

- Amazon EC2 でのデータ保護
- Amazon EC2 でのインフラストラクチャセキュリティ

- Amazon EC2の耐障害性
- Amazon EC2 のコンプライアンス検証
- Amazon EC2 の Identity and Access Management
- Amazon EC2 Windows インスタンスの更新管理
- Windows インスタンスにおけるセキュリティのベストプラクティス
- Amazon EC2 のキーペアと Amazon EC2 インスタンス
- EC2 インスタンスの Amazon EC2 セキュリティグループ
- Amazon EC2 インスタンスの NitroTPM
- Amazon EC2 インスタンス構成証明
- Windows インスタンスの クレデンシャルガード
- インターフェイス VPC エンドポイントを使用して Amazon EC2 にアクセスします。

Amazon EC2 でのデータ保護

AWS 責任分担モデル このモデルで説明されているように、「AWS」は、「AWS クラウド」のすべてを実行するグローバルインフラストラクチャを保護する責任があります。ユーザーは、このインフラストラクチャでホストされるコンテンツに対する管理を維持する責任があります。また、使用する「AWS のサービス」のセキュリティ設定と管理タスクもユーザーの責任となります。データプライバシーに関する詳細については、 Data Privacy FAQ をご覧ください。欧州におけるデータ保護に関する情報については、General Data Protection Regulation (GDPR) Center を参照してください。

データを保護するため、「AWS アカウント」認証情報を保護し、AWS IAM アイデンティティセンター または AWS Identity and Access Management (IAM) を使用して個々のユーザーをセットアップすることをお勧めします。この方法により、それぞれのジョブを遂行するために必要な権限のみが各ユーザーに付与されます。また、次の方法でデータを保護することもお勧めします:

- 各アカウントで多要素認証 (MFA) を使用します。
- SSL/TLS を使用して AWS リソースと通信します。TLS 1.2 は必須ですが、TLS 1.3 を推奨します。
- AWS CloudTrail で API とユーザーアクティビティロギングを設定します。CloudTrail 証跡を使用して AWS アクティビティをキャプチャする方法については、「AWS CloudTrail ユーザーガイド」の「CloudTrail 証跡の使用」を参照してください。
- AWS のサービス内のすべてのデフォルトセキュリティコントロールに加え、AWS 暗号化ソリューションを使用します。

- Amazon Macie などの高度な管理されたセキュリティサービスを使用します。これらは、Amazon S3 に保存されている機密データの検出と保護を支援します。
- コマンドラインインターフェイスまたは API を使用して AWS にアクセスする際に FIPS 140-3 検証済みの暗号化モジュールが必要な場合は、FIPS エンドポイントを使用します。利用可能な FIPS エンドポイントの詳細については、「連邦情報処理規格 (FIPS) 140-3」を参照してください。

お客様の E メールアドレスなどの極秘または機密情報を、タグ、または [名前] フィールドなどの自由形式のテキストフィールドに含めないことを強くお勧めします。これには、コンソール、API、AWS CLI、または AWS SDK を使用して Amazon EC2 またはその他の AWS のサービスで作業する場合が含まれます。タグ、または名前に使用される自由記述のテキストフィールドに入力したデータは、請求または診断ログに使用される場合があります。外部サーバーに URL を提供する場合、そのサーバーへのリクエストを検証できるように、認証情報を URL に含めないことを強くお勧めします。

内容

- Amazon EBS のデータセキュリティ
- 保管中の暗号化
- 転送中の暗号化

Amazon EBS のデータセキュリティ

Amazon EBS ボリュームは、初期化されていない raw ブロックデバイスとして表示されます。これらのデバイスは、EBS インフラストラクチャ上に作成される論理デバイスであり、Amazon EBS サービスは、お客様による利用または再利用の前に、デバイスが論理的に空になっている (つまり、raw ブロックがゼロになっている、または暗号で擬似ランダムデータが含まれている) ようにします。

DoD 5220.22-M (National Industrial Security Program Operating Manual) や NIST 800-88 (Guidelines for Media Sanitization) に詳述されているような、使用後もしくは使用前 (またはその両方) に特定の方法を使用してすべてのデータを消去する必要がある手順がある場合、Amazon EBS でこれを行うことができます。ブロックレベルのアクティビティは、Amazon EBS サービス内の基盤となるストレージメディアに反映されます。

保管中の暗号化

EBS ボリューム

Amazon EBS暗号化は、EBS ボリュームおよびスナップショット向けの暗号化ソリューションです。それは AWS KMS keys を使用します。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS 暗号化」を参照してください。

[Windows インスタンス] フォルダレベルおよびファイルレベルの暗号化に Microsoft EFS および NTFS アクセス許可を使用することもできます。

インスタンスストアボリューム

NVMe インスタンスストアボリューム内のデータは、インスタンスのハードウェアモジュールに実装されている XTS-AES-256 暗号を使用して暗号化されます。ローカルに接続された NVMe ストレージデバイスに書き込まれるデータの暗号化に使用されるキーは、お客様ごと、ボリュームごとに異なります。キーはハードウェアモジュールによって生成され、ハードウェアモジュールの内部にのみ存在します。AWS ユーザーはハードウェアモジュールにはアクセスできません。暗号化キーは、インスタンスが停止または終了して復元できないときに破棄されます。この暗号化を無効にしたり、独自の暗号キーを指定したりすることはできません。

H1、D3、D3en インスタンス上にある HDD インスタンスストアボリュームのデータは、XTS-AES-256 とワンタイムキーを使用して暗号化されます。

インスタンスを、停止、休止、または終了するとき、インスタンスストアボリュームのストレージの各ブロックはリセットされます。そのため、別のインスタンスのインスタンスストアを通じてデータにアクセスすることはできません。

メモリ

メモリの暗号化は、次のインスタンスで有効になります。

- AWS Graviton2 以降の AWS Graviton プロセッサを搭載したインスタンスは、常時オンのメモリ暗号化をサポートします。暗号化キーは、ホストシステム内で安全に生成され、ホストシステムから離れることはなく、ホストの再起動または電源切断時に破棄されます。詳細については、「AWS Graviton プロセッサ」を参照してください。
- M6i インスタンスなどの第 3 世代 Intel Xeon スケーラブルプロセッサ (Ice Lake) と M7i インスタンスなどの第 4 世代 Intel Xeon スケーラブルプロセッサ (Sapphire Rapids) を搭載したインスタンス。これらのプロセッサーは、インテル· トータル· メモリー暗号化 (TME) を使用した常時オンのメモリー暗号化をサポートします。
- M6a インスタンスなどの第 3 世代 AMD EPYC プロセッサ (Milan) と M7a インスタンスなどの第 4 世代 AMD EPYC プロセッサ (Genoa) を搭載したインスタンス。これらのプロセッサーは、AMD Secure Memory Encryption (SME) を使用した常時オンのメモリー暗号化をサポートします。

- AMD Secure Encrypted Virtualization-Secure Nested Paging (SEV-SNP) は、一部の AMD ベースのインスタンスタイプでサポートされています。詳細については「AMD SEV-SNP をサポートする EC2 インスタンスタイプの検索」を参照してください。

転送中の暗号化

物理レイヤーでの暗号化

AWS グローバルネットワーク上の AWS リージョンを流れるすべてのデータは、AWS の安全な施設を離れる前に、物理層で自動的に暗号化されます。AZ 間のトラフィックはすべて暗号化されます。追加的な暗号化レイヤーでは、このセクションに記載されているもの以外にも、保護が提供されている場合があります。

Amazon VPC ピアリングおよび Transit Gateway のクロスリージョンピアリング接続によって得られる暗号化

Amazon VPC ピアリングおよび Transit Gateway のピアリング接続を使用する、すべてのクロスリージョントラフィックは、リージョンからの送信時に自動的に一括で暗号化されます。このセクションで先に述べたように、すべてのトラフィックにおける物理レイヤーには、そのトラフィックが AWS の保護された設備を離れる前に、追加の暗号化レイヤーが自動的に提供されています。

インスタンス間での暗号化

AWS では、すべてのタイプの EC2 インスタンス間において安全でプライベートな接続を提供しています。さらに、一部のインスタンスタイプでは、基盤となる Nitro System ハードウェアのオフロード機能を使用して、インスタンス間の転送中のトラフィックを自動的に暗号化します。この暗号化では、256 ビットの暗号化による関連データによる認証暗号化 (AEAD) アルゴリズムを使用します。ネットワークのパフォーマンスには影響しません。インスタンス間でこの追加の転送中トラフィック暗号化をサポートするには、次の要件を満たす必要があります。

- インスタンスは、次のインスタンスタイプを使用します。
  - 汎用: M5dn、M5n、M5zn、M6a、M6i、M6id、M6idn、M6in、M7a、M7g、M7gd、M7i、M7i-flex、M8a、M8azn、M8g、M8gb、M8gd、M8gn、M8i、M8id、M8i-flex、M8in、M8idn、M8ine、M8ib、M8idb、M9g、M9gd、Mac-m4、Mac-m4pro
  - コンピューティング最適化: C5n、C6a、C6gn、C6i、C6id、C6in、C7a、C7g、C7gd、C7gn、C7i、C7i-flex、C8a、C8g、C8gb、C8gd、C8gn、C8i、C8id、C8i-flex、C8in、C8ine、C8ib、C9g、C9gd

  - メモリ最適化:
    R5dn、R5n、R6a、R6i、R6id、R6idn、R6in、R7a、R7g、R7gd、R7i、R7iz、R8a、R8g、R8gb、R8
    flex、R8in、R8idn、R8ib、R8idb、U-3tb1、U-6tb1、U-9tb1、U-12tb1、U-18tb1、U-24tb1、U7i-6tb、U
  - ストレージ最適化: D3、D3en、I3en、I4g、I4i、I7i、I7ie、I8g、I8ge、Im4gn、Is4gen
  - 高速コンピューティング:
    DL1、DL2q、F2、G4ad、G4dn、G5、G6、G6e、G6f、Gr6、Gr6f、G7、G7e、Inf1、Inf2、P3dn、P4
    B200、P6-B300、P6e-GB200、Trn1、Trn1n、Trn2、Trn2u、VT1
  - ハイパフォーマンスコンピューティング: Hpc6a、Hpc6id、Hpc7a、Hpc7g、Hpc8a
- 各インスタンスは同じリージョンにあるものとします。
- 各インスタンスは同じ VPC 内、あるいはピア接続された VPC 内にあり、トラフィックは仮想ネットワークのデバイスもしくはサービス (ロードバランサーや Transit Gateway など) を通過しないものとします。

このセクションで先に述べたように、すべてのトラフィックにおける物理レイヤーには、そのトラフィックが AWS の保護された設備を離れる前に、追加の暗号化レイヤーが自動的に提供されています。

AWS CLIを使用してインスタンス間のトランジットトラフィックを暗号化するインスタンスタイプを表示するには、以下のようにします。

次の describe-instances コマンドを使用します。

```
aws ec2 describe-instance-types \
    --filters Name=network-info.encryption-in-transit-supported,Values=true \
    --query "InstanceTypes[*].[InstanceType]" \
    --output text | sort
```

AWS Outposts との間の暗号化

Outpost は、AWS ホームリージョンとの間にサービスリンクと呼ばれる特別なネットワーク接続を作成し、オプションとして指定した VPC サブネットとのプライベート接続も可能です。これらの接続上のすべてのトラフィックは完全に暗号化されます。詳細については、AWS Outposts ユーザーガイドのサービスリンクによる接続および転送中の暗号化を参照してください。

リモートアクセスの暗号化

SSH プロトコルおよび RDP プロトコルは、直接でも EC2 Instance Connect 経由でも、インスタンスへのリモートアクセスにおいてセキュアな通信チャネルを提供します。AWS Systems Manager

Session Manager または Run Command を使用したインスタンスへのリモートアクセスは、TLS 1.2 を使用して暗号化されます。また、接続確立のリクエストは SigV4 を使用して署名され、AWS Identity and Access Management により認証および許可されます。

クライアントと Amazon EC2 インスタンスの間の転送中の機密データを、Transport Layer Security (TLS) などの暗号化プロトコルを使用して暗号化することは、お客様の責任範囲です。

(Windows インスタンス) EC2 インスタンスと、AWS API エンドポイントなどの機密性の高いリモートネットワークサービスとの間では、必ず暗号化された接続のみを許可してください。これを適用するには、アウトバウンドセキュリティグループまたは Windows ファイアウォールのルールを使用します。

Amazon EC2 でのインフラストラクチャセキュリティ

マネージドサービスとして、Amazon Elastic Compute Cloud は AWS グローバルネットワークセキュリティによって保護されています。AWS セキュリティサービスと AWS がインフラストラクチャを保護する方法については「AWS クラウドセキュリティ」を参照してください。インフラストラクチャセキュリティのベストプラクティスを使用して AWS 環境を設計するには「セキュリティの柱 - AWS 適切なアーキテクチャを備えたフレームワーク」の「インフラストラクチャの保護」を参照してください。

AWS が公開している API コールを使用し、ネットワーク経由で Amazon EC2にアクセスします。クライアントは次をサポートする必要があります。

- Transport Layer Security (TLS)。TLS 1.2 が必須で、TLS 1.3 をお勧めします。
- DHE (楕円ディフィー・ヘルマン鍵共有) や ECDHE (楕円曲線ディフィー・ヘルマン鍵共有) などの完全前方秘匿性 (PFS) による暗号スイート。これらのモードは Java 7 以降など、ほとんどの最新システムでサポートされています。

詳細については、「セキュリティの柱 - AWS Well-Architected フレームワーク」の「インフラストラクチャの保護」を参照してください。

ネットワークの隔離

仮想プライベートクラウド (VPC) はAWS クラウド内の論理的に隔離された領域にある仮想ネットワークです。ワークロードまたは組織エンティティ単位でインフラストラクチャを隔離するには個別の VPC を使用します。

サブネットはある範囲の IP アドレスが示す VPC 内の領域です。インスタンスを起動する場合には VPC 内のあるサブネットにおいて起動することになります。サブネットを使用すると、単一の VPC 内で多階層ウェブアプリケーションの各階層 (ウェブサーバー、アプリケーションサーバーおよびデータベースサーバーなど) を隔離できます。インターネットからの直接アクセスを認めるべきでないインスタンスにはプライベートサブネットを使用します。

プライベート IP アドレスを使用して VPC から Amazon EC2 API を呼び出すにはAWS PrivateLink を使用します。詳細については「インターフェイス VPC エンドポイントを使用して Amazon EC2 にアクセスします。」を参照してください。

物理ホストでの分離

同じ物理ホストで実行される異なる EC2 インスタンスは個別の物理ホストで実行されるかのように隔離されます。ハイパーバイザーが CPU およびメモリを隔離し、各インスタンスには生ディスクデバイスへのアクセスに代わる仮想ディスクへのアクセスが提供されます。

インスタンスを停止または終了すると、そのインスタンスに割り当てられていたメモリをハイパーバイザーがスクラブ (ゼロに設定) し、そのメモリが新たなインスタンスに割り当てられ、すべてのストレージブロックがリセットされます。これはお客様のデータが誤って他のインスタンスに引き渡されないようにするための処理です。

ネットワーク MAC アドレスはAWS ネットワークインフラストラクチャが各インスタンスに対し動的に割り当てます。IP アドレスはAWS ネットワークインフラストラクチャが各インスタンスに対し動的に割り当てるか、要認証 API リクエストを介して EC2 管理者が割り当てます。AWS ネットワークはインスタンスは割り当てられた MAC および IP アドレスからのみトラフィックを送信できます。それ以外のトラフィックは除外されます。

デフォルトではインスタンスはそのインスタンス宛ではないトラフィックを受信することはできません。インスタンスにおいて、ネットワークアドレス変換 (NAT、network address translation)、ルーティングまたはファイアウォールといったサービスの実行が必要な場合にはネットワークインターフェースの送信元/送信先チェックを無効化できます。

ネットワークトラフィックの制御

EC2 インスタンスへのネットワークトラフックを制御するには以下のオプションを検討します。

- セキュリティグループを使用してインスタンスへのアクセスを制限します。最小限必要なネットワークトラフィックを許可するルールを設定します。例えば、企業ネットワークのアドレス範囲からのトラフィックのみ、または HTTPS など特定のプロトコルのトラフィックのみを許可すること

ができます。Windows インスタンスではWindows 管理トラフィックと最小限のアウトバウンド接続を許可します。

- Amazon EC2 インスタンスへのネットワークアクセスをコントロールするための主要なメカニズムとして、セキュリティグループを活用します。必要に応じて、ネットワーク ACL を控えめに使用して、ステートレスできめの粗いネットワークコントロールを提供します。セキュリティグループはステートフルなパケットフィルタ処理を実行でき、他のセキュリティグループを参照するルールを作成できるため、ネットワーク ACL よりも汎用性があります。ただし、ネットワーク ACL はトラフィックの特定のサブセットを拒否したり、高レベルのサブネットガードレールを提供したりするための、セカンダリコントロールとして効果的です。また、ネットワーク ACL はサブネット全体に適用されるため、多層防御として使用して、正しいセキュリティグループなしでインスタンスが意図せずに起動される事態に備えることができます。
- [Windows インスタンス] グループポリシーオブジェクト (GPO) を使用して Windows Firewall 設定を一元的に管理し、ネットワークコントロールをさらに強化します。顧客は多くの場合、Windows Firewall を使用してネットワークトラフィックをさらに可視化し、セキュリティグループフィルタを補完して、特定のアプリケーションからのネットワークへのアクセスをブロックしたり、サブセット IP アドレスからのトラフィックをフィルタ処理したりする高度なルールを作成します。例えば、Windows Firewall はEC2 メタデータサービスの IP アドレスへのアクセスを特定のユーザーまたはアプリケーションに制限できます。また、公開サービスではセキュリティグループを使用して、特定のポートへのトラフィックを制限し、Windows Firewall を使用して、明示的にブロックされた IP アドレスのリストを維持する場合があります。
- インターネットからの直接アクセスを認めるべきでないインスタンスにはプライベートサブネットを使用します。プライベートサブネット内にあるインスタンスからのインターネットアクセスに踏み台ホストまたは NAT ゲートウェイを使用します。
- [Windows インスタンス] SSL/TLS を介した RDP カプセル化などのセキュアな管理プロトコルを使用します。リモートデスクトップゲートウェイクイックスタートからはSSL/TLS を使用するように RDP を設定するなど、リモートデスクトップゲートウェイのデプロイに関するベストプラクティスを得られます。
- [Windows インスタンス] Active Directory または Directory Service を使用して、Windows インスタンスへのインタラクティブなユーザーアクセスおよびグループアクセスを厳密かつ一元的に制御およびモニタリングし、ローカルユーザーのアクセス許可を防ぎます。また、ドメイン管理者の使用を避け、代わりに、より細かいアプリケーション固有のロールベースのアカウントを作成します。Just Enough Administration (JEA) により、Windows インスタンスに対する変更をインタラクティブなアクセスや管理者のアクセスなしで管理できます。さらに、JEA を使用すると、組織はインスタンス管理に必要な Windows PowerShell コマンドのサブセットへの管理アクセスを口ッ

クダウンできます。追加の情報についてはAWS セキュリティのベストプラクティスホワイトペーパーの Amazon EC2 への「OS レベル」のアクセスの管理セクションを参照してください。

- [Windows インスタンス] システム管理者はアクセスが制限された Windows アカウントを使用して日常のアクティビティを実行し、特定の設定変更を実行する必要があるときにのみアクセスを昇格させる必要があります。さらに、絶対に必要な場合にのみ Windows インスタンスに直接アクセスするようにします。代わりに、EC2 Run Command、Systems Center Configuration Manager (SCCM)、Windows PowerShell DSC、または Amazon EC2 Systems Manager (SSM) などの一元設定管理システムを活用して、変更を Windows サーバーにプッシュします。
- 最小限必要なネットワークルートを使用して Amazon VPC サブネットルートテーブルを設定します。例えば、インターネットゲートウェイへのルートがあるサブネットに、インターネットへの直接アクセスを必要とする Amazon EC2 インスタンスのみを配置し、仮想プライベートゲートウェイへのルートがあるサブネットに、内部ネットワークへの直接アクセスを必要とする Amazon EC2 インスタンスのみを配置します。
- 追加のセキュリティグループまたはネットワークインターフェイスを使用して、Amazon EC2 インスタンス管理トラフィックを通常のアプリケーショントラフィックとは別に制御および監査することをご検討ください。このアプローチにより、顧客は変更管理用の特別な IAM ポリシーを実装できるため、セキュリティグループルールや自動化されたルール検証スクリプトに対する変更の監査が容易になります。複数のネットワークインターフェイスを使用すると、ホストベースのルーティングポリシーを作成したり、ネットワークインターフェイスの割り当てられたサブネットに基づいて異なる VPC サブネットルーティングルールを活用したりするなど、ネットワークトラフィックを制御するための他のオプションも利用できます。
- AWS Virtual Private Network または Direct Connectを使用して、リモートネットワークから VPC へのプライベート接続を確立します。詳細についてはネットワークから Amazon VPC への接続オプションを参照してください。
- VPC フローログを使用して、インスタンスに到達するトラフィックを監視します。
- GuardDuty Malware Protection はワークロードを危険にさらしたり、リソースを悪用したり、データに不正にアクセスしたりする、悪意のあるソフトウェアによるインスタンス上での不審な動作を特定するために使用します。
- GuardDuty Runtime Monitoring はインスタンスへの潜在的脅威を特定し、これに対処するために使用します。詳細については「How Runtime Monitoring works with Amazon EC2 instances」を参照してください。
- AWS Security Hub CSPM、Reachability Analyzer または Network Access Analyzer を使用して、インスタンスからの意図しないネットワークアクセスがないか確認します。
- EC2 Instance Connectを使用して、SSH キーの共有および管理が不要な Secure Shell (SSH) を使いインスタンスに接続します。

- インバウンド SSH または RDP ポートを開いてキーペアを管理する代わりに、AWS Systems Manager セッションマネージャーを使用してインスタンスにリモートアクセスします。
- インスタンスに接続する代わりに、AWS Systems Manager Run Command を使用して一般的な管理タスクを自動化します。
- [Windows インスタンス] Windows OS のロールと Microsoft のビジネスアプリケーションの多くにはIIS 内の IP アドレス範囲制限、Microsoft SQL Server の TCP/IP フィルタリングポリシー、Microsoft Exchange の接続フィルターポリシーなどの拡張機能も備わっています。アプリケーション層内のネットワーク制限機能により、重要なビジネスアプリケーションサーバーに追加の防御層を提供できます。

Amazon VPC はゲートウェイ、プロキシサーバー、ネットワークモニタリングのオプションなど、追加のネットワークセキュリティコントロールをサポートしています。詳細については「Amazon VPC ユーザーガイド」の「Control network traffic」を参照してください。

Amazon EC2の耐障害性

AWS のグローバルインフラストラクチャは、AWS リージョンとアベイラビリティーゾーンを中心として構築されています。リージョンには、低レイテンシー、高いスループット、そして高度の冗長ネットワークで接続されている複数の物理的に独立および隔離されたアベイラビリティーゾーンがあります。アベイラビリティーゾーンでは、ゾーン間で中断することなく自動的にフェイルオーバーするアプリケーションとデータベースを設計および運用することができます。アベイラビリティーゾーンは、従来の単一または複数のデータセンターインフラストラクチャよりも可用性、耐障害性、および拡張性が優れています。

データまたはアプリケーションをより広範な地理的距離にわたってレプリケートする必要がある場合は、AWS Local Zonesを使用します。AWS ローカルゾーンは AWS リージョンの拡張であり、ユーザーに近い場所に配置されます。Local Zones は、インターネットへの独自の接続を持ち、Direct Connect をサポートします。すべての AWS リージョンと同じように、AWS Local Zonesは他の AWS リージョンから完全に分離されています。

AWS ローカルゾーン内でデータまたはアプリケーションをレプリケートする必要がある場合、次のいずれかのゾーンをフェイルオーバーゾーンとして使用することを AWS はお勧めします。

- 別のローカルゾーン
- 親ゾーンではないリージョン内のアベイラビリティーゾーン。親ゾーンを確認するには、describe-availability-zones コマンドを使用できます。

AWS リージョンとアベイラビリティーゾーンの詳細については、AWS グローバルインフラストラクチャを参照してください。

Amazon EC2 は、AWS グローバルインフラストラクチャに加え、データ耐障害性をサポートする以下の機能を提供します。

- リージョンで AMI をコピーする機能
- リージョン間の EBS スナップショットをコピーする機能
- Amazon Data Lifecycle Manager を使用した EBS-backed AMI の自動化
- Amazon Data Lifecycle Managerを使用して EBS スナップショットを自動化する機能
- Amazon EC2 Auto Scalingを使用してフリートの健全性や可用性を維持する機能
- Elastic Load Balancing を使用して、単一のまたは複数のアベイラビリティーゾーンにある複数のインスタンスの間で受信トラフィックを分散する機能

Amazon EC2 のコンプライアンス検証

AWS のサービスが特定のコンプライアンスプログラムの対象であるかどうかを確認するには、「コンプライアンスプログラムによる AWS のサービス 対象範囲内」を参照して、関心のあるコンプライアンスプログラムを選択してください。一般的な情報については、AWSコンプライアンスプログラム を参照してください。

AWS Artifact を使用して、サードパーティの監査レポートをダウンロードできます。詳細については、「AWS Artifact でレポートをダウンロードする」「」を参照してください。

AWS のサービスを使用する際のお客様のコンプライアンス責任は、お客様のデータの機密性や貴社のコンプライアンス目的、適用可能な法律および規制によって決定されます。AWS のサービス を使用する際のコンプライアンス責任の詳細については、AWS セキュリティドキュメントを参照してください。

Amazon EC2 の Identity and Access Management

AWS Identity and Access Management (IAM) は管理者が AWS リソースへのアクセスを安全に制御するために役立つ AWS のサービスです。IAM 管理者は Amazon EC2 リソースの使用に「認証」(サインイン) されて「承認」(許可の付与) される人を管理します。IAM は追加費用なしで使用できる AWS のサービスです。

セキュリティ認証情報により、AWS サービスでユーザーの身分が証明され、Amazon EC2 リソースなどの AWS リソースへのアクセスが付与されます。Amazon EC2 および IAM の機能を使用して、他のユーザー、サービス、およびアプリケーションがユーザーの Amazon EC2 リソースを使用できるようにします。その際、ユーザーのセキュリティ認証情報は共有されません。他のユーザーが AWS アカウントのリソースを使用する方法を制御するには IAM を、Amazon EC2 インスタンスへのアクセスを制御するにはセキュリティグループを使用できます。Amazon EC2 のリソースの完全使用または制限付き使用のどちらを許可するか選択できます。

開発者はEC2 インスタンスで実行するアプリケーションに必要なセキュリティ認証情報を、IAM ロールを使用して管理できます。IAM ロールをインスタンスにアタッチすると、インスタンスで実行されているアプリケーションはインスタンスメタデータサービス (IMDS) から認証情報を取得できます。

IAM を使用して AWS リソースを保護するためのベストプラクティスについては「IAM ユーザーガイド」の「IAM でのセキュリティのベストプラクティス」を参照してください。

内容

- Amazon EC2 のアイデンティティベースのポリシー
- Amazon EC2 API へのアクセスを制御するポリシーの例
- Amazon EC2 コンソールへのアクセスを制御するポリシーの例
- Amazon EC2 の AWS マネージドポリシー
- Amazon EC2 の IAM ロール

Amazon EC2 のアイデンティティベースのポリシー

デフォルトではユーザーには Amazon EC2 リソースを作成または変更、または Amazon EC2 API、Amazon EC2 コンソールあるいは CLI を使用するタスクを実行する許可がありません。ユーザーがリソースを作成または変更、およびタスクを実行できるようにするにはIAM ポリシーを作成する必要があります。これによって、必要な特定のリソースおよび API アクションを使用するための許可をユーザーに付与し、その後、ポリシーをその許可が必要なユーザー、グループまたは IAM ロールにアタッチします。

ポリシーをユーザー、ユーザーのグループ、またはロールにアタッチする場合、ポリシーによって特定リソースの特定タスクを実行するユーザーの許可が許可または拒否されます。IAM ポリシーの一般的な情報については「IAM ユーザーガイド」の「IAM の許可とポリシー」を参照してください。&IAM; ポリシーの管理と作成の詳細については「IAM ポリシーの管理」を参照してください。

IAM ポリシーは1 つ以上の Amazon EC2 アクションを使用するアクセス許可を付与または拒否する必要があります。さらに、このアクションで使用できるリソース (すべてのリソースか、場合によっては特定のリソース) も指定する必要があります。このポリシーにはリソースに適用する条件も含めることができます。

最初に、Amazon EC2 の AWS マネージドポリシーがニーズを満たしているかどうかを確認できます。満たしていない場合は独自のカスタムポリシーを作成できます。詳細については、the section called “AWS 管理ポリシー”を参照してください。

内容

- ポリシー構文
- Amazon EC2 のアクション
- Amazon EC2 API アクションでサポートされるリソースレベルのアクセス許可
- Amazon EC2 用の Amazon リソースネーム (ARN)
- Amazon EC2 の条件キー
- 属性ベースのアクセスを使用するアクセスの制御
- ユーザー、グループ、およびロールに許可を付与する
- ユーザーが必要なアクセス許可を持っているかどうかの確認

ポリシー構文

IAM ポリシーは 1 つ以上のステートメントで構成される JSON ドキュメントです。各ステートメントは次のように構成されます。

```
{
  "Statement":[{
    "Effect":"effect",
    "Action":"action",
    "Resource":"arn",
    "Condition":{
      "condition":{
        "key":"value"
        }
      }
    }
  ]
}
```

ステートメントはさまざまなエレメントで構成されます。

- Effect: effect は`Allow` または `Deny` にすることができます。デフォルトでは ユーザーはリソースおよび API アクションを使用するアクセス許可がないため、リクエストはすべて拒否されます。明示的な許可はデフォルトに上書きされます。明示的な拒否はすべての許可に優先します。
- [Action]: action はアクセス許可を付与または拒否する対象とする、特定の API アクションです。action の指定についてはAmazon EC2 のアクションを参照してください。
- [Resource]: アクションによって影響を及ぼされるリソースです。Amazon EC2 API アクションの中にはアクションによって作成/変更できるリソースをポリシー内で特定できるものもあります。Amazon リソースネーム (ARN) を使用して、またはステートメントがすべてのリソースに適用されることを示すワイルドカード (*) を使用して、リソースを指定します。詳細については、Amazon EC2 API アクションでサポートされるリソースレベルのアクセス許可を参照してください。
- [Condition]: condition はオプションです。ポリシーの発効条件を指定するために使用します。Amazon EC2 の条件を指定する方法についてはAmazon EC2 の条件キーを参照してください。

ポリシー要件の詳細については「IAM ユーザーガイド」の「IAM JSON ポリシーリファレンス」を参照してください。Amazon EC2 の IAM ポリシーステートメントの例については「Amazon EC2 API へのアクセスを制御するポリシーの例」を参照してください。

Amazon EC2 のアクション

IAM ポリシーステートメントで、IAM をサポートするすべてのサービスから任意の API アクションを指定できます。Amazon EC2 の場合、API アクション `ec2:` の名前に次のプレフィクスを使用します。例: `ec2:RunInstances` および `ec2:CreateImage`。

単一のステートメントに複数のアクションを指定するには次のようにコンマで区切ります。

```
"Action": ["ec2:action1", "ec2:action2"]
```

ワイルドカードを使用して複数のアクションを指定することもできます。例えば、以下のようにDescribeという単語で始まる名前のすべてのアクションを指定できます。

```
"Action": "ec2:Describe*"
```

> **Note**
>
> 現在、すべての Amazon EC2 Describe* API アクションがリソースレベルのアクセス許可をサポートしているわけではありません。Amazon EC2のリソースレベルの許可の詳細についてはAmazon EC2 のアイデンティティベースのポリシーを参照してください。

Amazon EC2 API アクションをすべて指定するには* ワイルドカードを以下のように使用します。

```
"Action": "ec2:*"
```

Amazon EC2 アクションのリストを確認するには「Service Authorization Reference」の「Actions defined by Amazon EC2」を参照してください。

Amazon EC2 API アクションでサポートされるリソースレベルのアクセス許可

リソースレベルのアクセス許可とはユーザーがアクションを実行できるリソースを指定できる機能を意味します。Amazon EC2 はリソースレベルのアクセス許可を部分的にサポートします。これは特定の Amazon EC2 アクションでは満たす必要がある条件、またはユーザーが使用できる特定のリソースに基づいて、ユーザーがそれらのアクションをいつ使用できるかを制御できることを意味します。例えば、特定の AMI のみを使用して、特定のタイプのインスタンスだけを起動するアクセス許可をユーザーに付与できます。

IAM ポリシーステートメントでリソースを指定するには Amazon リソースネーム (ARN) を使用します。ARN 値の指定についてはAmazon EC2 用の Amazon リソースネーム (ARN)を参照してください。API アクションが個々の ARN をサポートしていない場合はワイルドカード (*) を使用して、アクションによってすべてのリソースが影響を受ける可能性があることを指定する必要があります。

リソースレベルのアクセス許可をサポートする Amazon EC2 API アクション、およびポリシーで使用できる ARN と条件キーがわかる表を見るにはAmazon EC2 のアクション、リソース、および条件キーを参照してください。

Amazon EC2 API アクションに対して使用する IAM ポリシーで、タグベースのリソースレベルアクセス許可を適用できます。これにより、ユーザーがどのリソースを作成、変更、または使用できるかを制御しやすくなります。詳細については、Amazon EC2 リソース作成時にタグ付けするアクセス許可の付与を参照してください。

Amazon EC2 用の Amazon リソースネーム (ARN)

各 IAM ポリシーステートメントはARN を使用して指定したリソースに適用されます。

ARN には以下の一般的な構文があります。

```
arn:aws:[service]:[region]:[account-id]:resourceType/resourcePath
```

service

サービス (例: ec2)。

region

リソースのリージョン (例: us-east-1)。

account-id

ハイフンなしの AWS アカウント ID (例: 123456789012)。

resourceType

リソースの種類 (例: instance)。

resourcePath

リソースを識別するパス。パスにワイルドカードの * が使用できます。

例えば、以下のように ARN を使用して、ステートメント内で特定のインスタンス (i-1234567890abcdef0) を指定することができます。

```
"Resource": "arn:aws:ec2:us-east-1:123456789012:instance/i-1234567890abcdef0"
```

以下のように * ワイルドカードを使用して、特定のアカウントに属するすべてのインスタンスを指定できます。

```
"Resource": "arn:aws:ec2:us-east-1:123456789012:instance/*"
```

また、以下のように * ワイルドカードを使用して、特定のアカウントに属するすべての Amazon EC2 リソースを指定することもできます。

```
"Resource": "arn:aws:ec2:us-east-1:123456789012:*"
```

すべてのリソースを指定する場合、または特定の API アクションが ARN をサポートしていない場合は以下のように、Resource エレメント内で * ワイルドカードを使用します。

```
"Resource": "*"
```

Amazon EC2 API アクションの多くが複数のリソースと関連します。例えば、AttachVolume では Amazon EBS ボリュームをインスタンスにアタッチするため、ユーザーはボリュームおよびインスタンスを使用する許可が必要です。1 つのステートメントで複数のリソースを指定するには次のように ARN をカンマで区切ります。

```
"Resource": ["arn1", "arn2"]
```

Amazon EC2 リソースの ARN のリストについては、Amazon EC2 で定義されるリソースタイプを参照してください。

Amazon EC2 の条件キー

ポリシーステートメントではオプションで有効になるタイミングを制御する条件を指定できます。各条件には 1 つ以上のキーと値のペアが含まれます。条件キーは大文字小文字を区別しません。AWS グローバル条件キーに加え、追加のサービス固有の条件キーも定義されています。

Amazon EC2 のサービス固有の条件キーのリストについては、Amazon EC2 の条件キーを参照してください。Amazon EC2 ではAWS グローバル条件キーも実装されています。詳細については、IAM ユーザーガイドのすべてのリクエストで利用可能な情報を参照してください。

すべての Amazon EC2 アクションは aws:RequestedRegion および ec2:Region 条件キーをサポートします。詳細については、例: 特定のリージョンへのアクセスの制限を参照してください。

IAM ポリシーで条件キーを使用するにはCondition ステートメントを使用します。例えば、次のポリシーはセキュリティグループのインバウンドルールとアウトバウンドルールを追加および削除するアクセス許可をユーザーに付与します。ec2:Vpc 条件キーを使用して、これらのアクションを実行できる対象は特定の VPC 内のセキュリティグループに限ることを指定します。

JSON

```
{
"Version":"2012-10-17",
  "Statement":[{
    "Effect":"Allow",
    "Action": [
       "ec2:AuthorizeSecurityGroupIngress",
```

```
        "ec2:AuthorizeSecurityGroupEgress",
        "ec2:RevokeSecurityGroupIngress",
        "ec2:RevokeSecurityGroupEgress"],
      "Resource": "arn:aws:ec2:us-east-1:111122223333:security-group/*",
       "Condition": {
         "StringEquals": {
           "ec2:Vpc": "arn:aws:ec2:us-east-1:111122223333:vpc/
vpc-11223344556677889"
         }
       }
    }
  ]
}
```

複数の条件、または単一の条件に複数のキーを指定する場合、論理 AND 演算を使用してそれらを評価します。1 つのキーに複数の値を使用して単一の条件を指定する場合、論理 OR 演算を使用して条件を評価します。アクセス許可が付与されるにはすべての条件を満たしている必要があります。

条件を指定する際にプレースホルダーも使用できます。詳細については、IAM ユーザーガイドのIAM ポリシーエレメント: 変数およびタグを参照してください。

> **Important**
>
> 多くの条件キーはリソースに固有のものであり、一部の API アクションでは複数のリソースを使用します。条件キーを使用してポリシーを作成する場合はポリシーステートメントの `Resource` 要素で、条件キーが適用されるリソースを指定します。指定しない場合、そのポリシーはユーザーに対してすべてのアクションの実行を禁止します。これは条件キーが適用されないリソースに対して条件チェックが失敗するためです。リソースを指定しない場合や、ポリシーの `Action` 要素に複数の API アクションを含めている場合は `...IfExists` 条件タイプを使用して、条件キーが適用されないリソースに対して無視されるようにする必要があります。詳細については、IAM ユーザーガイドの..IfExists 条件を参照してください。

条件キー

- ec2:Attribute 条件キー
- ec2:ResourceID 条件キー
- ec2:SourceInstanceARN 条件キー

ec2:Attribute 条件キー

`ec2:Attribute` 条件キーはリソースの属性によってアクセスをフィルタリングする条件に使用できます。

この条件キーはプリミティブデータ型 (文字列や整数など) のプロパティ、または Value プロパティのみを含む複雑な AttributeValue オブジェクト (ModifyImageAttribute API アクションの Description または ImdsSupport オブジェクトなど) に使用できます。条件キーはModifyImageAttribute の LaunchPermission オブジェクトなど、複数のプロパティを含む複雑なオブジェクトには使用できません。

例えば、次のポリシーではModifyImageAttributeAPI アクションの複雑な Description オブジェクトによるアクセスをフィルタリングするために `ec2:Attribute/Description` 条件キーを使用します。条件キーはイメージの説明を `Production` または `Development` のいずれかに変更するリクエストのみを許可します。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:ModifyImageAttribute",
      "Resource": "arn:aws:ec2:us-east-1::image/ami-*",
      "Condition": {
        "StringEquals": {
          "ec2:Attribute/Description": [
            "Production",
            "Development"
          ]
        }
      }
    }
  ]
}
```

次のポリシー例ではModifyImageAttribute API アクションのプリミティブな Attribute プロパティによるアクセスをフィルタリングするために `ec2:Attribute` 条件キーを使用します。この条件キーはイメージの説明を変更しようとするすべてのリクエストを拒否します。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "ec2:ModifyImageAttribute",
      "Resource": "arn:aws:ec2:us-east-1::image/ami-*",
      "Condition": {
        "StringEquals": {
          "ec2:Attribute": "Description"
        }
      }
    }
  ]
}
```

ec2:ResourceID 条件キー

指定された API アクションで次の `ec2:ResourceID` 条件キーを使用する場合、条件キーの値はAPI アクションによって作成される結果のリソースを指定するために使用されます。`ec2:ResourceID` 条件キーを使用して、API リクエストで指定されたソース リソースを指定することはできません。指定された API で次の `ec2:ResourceID` 条件キーのいずれかを使用する場合は常にワイルドカード (*) を指定する必要があります。別の値を指定した場合、条件はランタイム中に常に * に解決されます。例えば、CopyImage API で `ec2:ImageId` 条件キーを使用するには次のように条件キーを指定する必要があります。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:CopyImage",
      "Resource": "arn:aws:ec2:us-east-1::image/ami-*",
      "Condition": {
        "StringEquals": {
```

```
          "ec2:ImageID": "*"
        }
      }
    }
  ]
}
```

これらの API アクションではこれらの条件キーを使用しないことをお勧めします。

- ec2:DhcpOptionsID – CreateDhcpOptions
- ec2:ImageID – CopyImage、CreateImage、ImportImage、RegisterImage
- ec2:InstanceID – RunInstances および ImportInstance
- ec2:InternetGatewayID – CreateInternetGateway
- ec2:NetworkAclID – CreateNetworkAcl
- ec2:NetworkInterfaceID – CreateNetworkInterface
- ec2:PlacementGroupName – CreatePlacementGroup
- ec2:RouteTableID – CreateRouteTable
- ec2:SecurityGroupID – CreateSecurityGroup
- ec2:SnapshotID – CopySnapshot、CreateSnapshot、CreateSnapshots、ImportSnapshots
- ec2:SubnetID – CreateSubnet
- ec2:VolumeID – CreateVolume および ImportVolume
- ec2:VpcID – CreateVpc
- ec2:VpcPeeringConnectionID – CreateVpcPeeringConnection

特定のリソース ID に基づいてアクセスをフィルタリングするには次のように Resource ポリシー要素を使用することをお勧めします。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Action": "ec2:CopyImage",
      "Resource": "arn:aws:ec2:us-east-1::image/ami-01234567890abcdef"
    }
  ]
}
```

ec2:SourceInstanceARN 条件キー

ec2:SourceInstanceARN を使用して、リクエスト元のインスタンスの ARN を指定します。これは AWS グローバル条件キーであり、Amazon EC2 以外のサービスで使用できるということです。ポリシーの例については「例: 特定のインスタンスが他の AWS サービスでリソースを表示できるようにする」を参照してください。

属性ベースのアクセスを使用するアクセスの制御

EC2 リソースを使用するための許可をユーザーに付与する IAM ポリシーを作成する場合、ポリシーの Condition 要素にタグ情報を含めることで、タグに基づいてアクセスをコントロールできます。これは属性ベースのアクセス制御 (ABAC) と呼ばれます。ABAC を使用すると、ユーザーが変更、使用、または削除できるリソースをより適切に制御できます。詳細については、AWS の ABAC とはを参照してください。

例えば、インスタンスを終了することをユーザーに許可するが、インスタンスに environment=production タグが付いている場合はアクションを拒否するポリシーを作成できます。これを行うにはaws:ResourceTag 条件キーを使用し、リソースにアタッチされているタグに基づいてリソースへのアクセスを許可または拒否します。

```
"StringEquals": { "aws:ResourceTag/environment": "production" }
```

Amazon EC2 API アクションが aws:ResourceTag 条件キーを使用したアクセスの制御をサポートしているかどうかについてはAmazon EC2 のアクション、リソース、および条件キーを参照してください。Describe アクションはリソースレベルのアクセス権限をサポートしないため、条件のない別のステートメントでそれらのアクセス権限を指定する必要があることに注意してください。

IAM ポリシーの例はAmazon EC2 API へのアクセスを制御するポリシーの例を参照してください。

タグに基づいてリソースへのユーザーのアクセスを許可または拒否する場合はユーザーが同じリソースに対してそれらのタグを追加または削除することを明示的に拒否することを検討する必要があります。そうしないと、ユーザーはそのリソースのタグを変更することで、制限を回避してリソースにアクセスできてしまいます。

ユーザー、グループ、およびロールに許可を付与する

アクセス権限を付与するにはユーザー、グループ、またはロールにアクセス許可を追加します。

- AWS IAM アイデンティティセンター のユーザーとグループ:

  アクセス許可セットを作成します。「AWS IAM アイデンティティセンター ユーザーガイド」の「アクセス許可セットを作成する」の手順に従ってください。

- IAM 内で、ID プロバイダーによって管理されているユーザー:

  ID フェデレーションのロールを作成します。詳細については IAM ユーザーガイド の サードパーティー ID プロバイダー (フェデレーション) 用のロールを作成する を参照してください。

- IAM ユーザー:
  - ユーザーが担当できるロールを作成します。手順については IAM ユーザーガイド の IAM ユーザーのロールの作成 を参照してください。
  - (お奨めできない方法) ポリシーをユーザーに直接アタッチするか、ユーザーをユーザーグループに追加します。詳細については「IAM ユーザーガイド」の「ユーザー (コンソール) へのアクセス権限の追加」を参照してください。

ユーザーが必要なアクセス許可を持っているかどうかの確認

IAM ポリシーを作成したら、ポリシーを本稼働環境に置く前に、そのポリシーがユーザーに特定の API アクションおよび必要なリソースを使用するアクセス許可を付与しているかどうかを確認することをお勧めします。

まずテスト目的のユーザーを作成し、作成した IAM ポリシーをテストユーザーにアタッチします。次に、テストユーザーとしてリクエストを作成します。

テスト中の Amazon EC2 アクションがリソースを作成または変更する場合、`DryRun` パラメータ を使用してリクエストを作成する (または AWS CLI オプションで `--dry-run` コマンドを実行する) 必要があります。この場合、発信者は認可チェックを行いますが、操作は完了しません。例えば、実際に終了させることなく、ユーザーが特定のインスタンスを終了できるかどうかを確認できます。テストユーザーに必要なアクセス許可がある場合、リクエストで `DryRunOperation` が返されます。必要なアクセス許可がない場合は `UnauthorizedOperation` が返されます。

ポリシーが想定したアクセス許可をユーザーに付与していない場合、または過度に許可されている場合、必要に応じてポリシーを調整し、必要な結果を得るまで再テストできます。

> **Important**
>
> ポリシーの変更が反映され、有効になるには数分間かかります。したがって、ポリシーの更新をテストするには 5 分かかると見ておいてください。

認可チェックが失敗した場合、リクエストでは診断情報でエンコードされたメッセージが返されます。DecodeAuthorizationMessage アクションを使用してメッセージをデコードできます。詳細については、AWS Security Token Service API リファレンス の DecodeAuthorizationMessage、 および コマンドリファレンスのdecode-authorization-message.を参照してください。

Amazon EC2 API へのアクセスを制御するポリシーの例

IAM ポリシーを使用して、Amazon EC2 を操作するために必要なアクセス許可をユーザーに付与することができます。詳細な手順については「IAM ユーザーガイド」の「IAM ポリシーの作成」を参照してください。

以下の例ではAmazon EC2 を使用するアクセス許可をユーザーに付与するために使用できるポリシーステートメントを示しています。これらのポリシーはAWS CLI または AWS SDK を使用して行われたリクエスト向けに設計されています。次の例では###############をユーザー自身の情報で置き換えます。

例

- 例: 読み取り専用アクセス
- 例: 特定のリージョンへのアクセスの制限
- インスタンスの使用
- インスタンスの起動 (RunInstances)
- スポットインスタンス の操作
- 例: リザーブドインスタンス の操作
- 例: リソースのタグ付け
- 例: IAM ロールの使用
- 例: ルートテーブルの使用
- 例: 特定のインスタンスが他の AWS サービスでリソースを表示できるようにする
- 例: 起動テンプレートの使用
- インスタンスメタデータの使用

- Amazon EBS ボリュームとスナップショットの使用

Amazon EC2 コンソールで機能するポリシーの例については、「Amazon EC2 コンソールへのアクセスを制御するポリシーの例」を参照してください。

例: 読み取り専用アクセス

次のポリシーでは名前が Describe で始まるすべての Amazon EC2 API アクションを使用できるアクセス許可をユーザーに与えます。Resource エレメントにワイルドカードを使用します。これはユーザーが API アクションですべてのリソースを指定できることを示します。また、API アクションがリソースレベルのアクセス許可をサポートしていない場合も、* ワイルドカードが必要です。どの Amazon EC2 API アクションでどの ARN を使用できるかの詳細については、「Amazon EC2 のアクション、リソース、および条件キー」を参照してください。

デフォルトで API アクションを使用するアクセス許可が拒否されているため、ユーザーには (別のステートメントでアクセス許可が与えられない限り) そのリソースに対してアクションを実行するアクセス許可がありません。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:Describe*",
      "Resource": "*"
    }
   ]
}
```

例: 特定のリージョンへのアクセスの制限

次のポリシーではリージョンが 欧州 (フランクフルト) でない限り、すべての Amazon EC2 API アクションを使用するアクセス許可をユーザーに拒否します。これにはグローバル条件キー aws:RequestedRegion が使用され、このキーはすべての Amazon EC2 API アクションでサポートされています。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
       {
      "Effect": "Deny",
      "Action": "ec2:*",
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:RequestedRegion": "eu-central-1"
        }
      }
    }
  ]
}
```

または条件キー `ec2:Region` を使用することもできます。これはAmazon EC2 に固有のもので、すべての Amazon EC2 API アクションでサポートされています。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
       {
      "Effect": "Deny",
      "Action": "ec2:*",
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "ec2:Region": "eu-central-1"
        }
      }
    }
  ]
}
```

インスタンスの使用

例

- 例: すべてのインスタンスを記述、起動、停止、開始、および終了する
- 例: すべてのインスタンスを記述し、特定のインスタンスのみを停止、開始、および終了する

例: すべてのインスタンスを記述、起動、停止、開始、および終了する

次のポリシーではAction エレメントで指定された API アクションを使用するアクセス許可をユーザーに与えます。Resource エレメントでは * ワイルドカードを使用して、ユーザーが API アクションですべてのリソースを指定できることを示します。また、API アクションがリソースレベルのアクセス許可をサポートしていない場合も、* ワイルドカードが必要です。どの Amazon EC2 API アクションでどの ARN を使用できるかの詳細については、「Amazon EC2 のアクション、リソース、および条件キー」を参照してください。

ユーザーはデフォルトで API アクションを使用するアクセス許可を拒否されているため、ユーザーには (別のステートメントでユーザーにそのアクセス許可を与えない限り) その他の API アクションを使用するアクセス許可がありません。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances",
        "ec2:DescribeImages",
        "ec2:DescribeKeyPairs",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeAvailabilityZones",
        "ec2:RunInstances",
        "ec2:TerminateInstances",
        "ec2:StopInstances",
        "ec2:StartInstances"
      ],
      "Resource": "*"
    }
   ]
```

```
}
```

例: すべてのインスタンスを記述し、特定のインスタンスのみを停止、開始、および終了する

次のポリシーでは、ユーザーは「purpose=test」のリソースタグを使用してすべてのインスタンスを表示し、i-1234567890abcdef0 および i-0598c7d356eba48d7 のインスタンスのみを開始および停止し、us-east-1 リージョンでインスタンスのみを終了できるようにします。

最初のステートメントではResource エレメントに * ワイルドカードを使用して、ユーザーがそのアクションにすべてのリソースを指定できることを示しています。この場合、すべてのインスタンスをリストできます。また、API アクションがリソースレベルのアクセス許可をサポートしていない場合も、* ワイルドカードが必要です (この場合はec2:DescribeInstances)。どの Amazon EC2 API アクションでどの ARN を使用できるかの詳細については、「Amazon EC2 のアクション、リソース、および条件キー」を参照してください。

2 番目のステートメントではStopInstances および StartInstances アクションに対してリソースレベルのアクセス許可を使用しています。Resource エレメント内で、ARN によって特定のインスタンスが指定されています。

3 番目のステートメントでは、指定された AWS アカウントに属する us-east-1 リージョン内のすべてのインスタンスを終了する許可がユーザーに与えられます。ただし、インスタンスに "purpose=test" タグがある場合にのみ該当します。Condition エレメントはポリシーステートメントの発効条件を指定します。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
   {
   "Effect": "Allow",
      "Action": "ec2:DescribeInstances",
      "Resource": "*"
   },
   {
      "Effect": "Allow",
      "Action": [
        "ec2:StopInstances",
        "ec2:StartInstances"
      ],
```

```
      "Resource": [
        "arn:aws:ec2:us-east-1:111122223333:instance/i-1234567890abcdef0",
        "arn:aws:ec2:us-east-1:111122223333:instance/i-0598c7d356eba48d7"
      ]
    },
    {
      "Effect": "Allow",
      "Action": "ec2:TerminateInstances",
      "Resource": "arn:aws:ec2:us-east-1:111122223333:instance/*",
      "Condition": {
         "StringEquals": {
            "aws:ResourceTag/purpose": "test"
         }
      }
   }

   ]
}
```

インスタンスの起動 (RunInstances)

RunInstances API アクションは1 つ以上の オンデマンドインスタンス または 1 つ以上の スポットインスタンス を起動します。RunInstances は AMI を必要とし、インスタンスを作成します。ユーザーはリクエストでキーペアとセキュリティグループを指定できます。VPC 内に起動するにはサブネットが必要であり、起動されるとネットワークインターフェイスが作成されます。Amazon EBS-backed AMI から起動すると、ボリュームが作成されます。そのため、ユーザーにはこれらの Amazon EC2 リソースを使用するアクセス許可が必要です。ユーザーが RunInstances に対してオプションのパラメータを指定する必要がある、またはユーザーからパラメータの特定の値を制限するポリシーステートメントを作成できます。

インスタンスの起動に必要なリソースレベルのアクセス許可の詳細については、「Amazon EC2 のアクション、リソース、および条件キー」を参照してください。

デフォルトでは作成したインスタンスを記述、開始、停止、または終了するアクセス許可はユーザーに付与されていません。作成したインスタンスを管理するアクセス許可をユーザーに付与する 1 つの方法としてはインスタンスごとに特定のタグを作成し、そのタグでインスタンスを管理できるようにステートメントを作成します。詳細についてはインスタンスの使用を参照してください。

リソース

- AMI

- インスタンスタイプ
- サブネット
- EBS ボリューム
- タグ
- 起動テンプレートのタグ
- Elastic GPU
- 起動テンプレート

AMI

次のポリシーでは指定された AMI (ami-9e1670f7 および ami-45cf5c3c) のみを使用してインスタンスを起動できます。(別のステートメントでユーザーに起動するアクセス許可が付与されない限り) ユーザーはその他の AMI を使用してインスタンスを起動することはできません。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
   {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": [
        "arn:aws:ec2:us-east-1::image/ami-9e1670f7",
        "arn:aws:ec2:us-east-1::image/ami-45cf5c3c",
        "arn:aws:ec2:us-east-1:111122223333:instance/*",
        "arn:aws:ec2:us-east-1:111122223333:volume/*",
        "arn:aws:ec2:us-east-1:111122223333:key-pair/*",
        "arn:aws:ec2:us-east-1:111122223333:security-group/*",
        "arn:aws:ec2:us-east-1:111122223333:subnet/*",
        "arn:aws:ec2:us-east-1:111122223333:network-interface/*"
      ]
    }
   ]
}
```

または次のポリシーを使用すると、ユーザーは Amazon、または特定の信頼できる検証済みのパートナーが所有するすべての AMI からインスタンスを起動できます。最初のステートメントの

Condition エレメントはec2:Owner が amazon であるかどうかをテストします。(別のステートメントでユーザーに起動するアクセス許可が付与されない限り) ユーザーはその他の AMI を使用してインスタンスを起動することはできません。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
         {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": [
         "arn:aws:ec2:us-east-1::image/ami-*"
      ],
      "Condition": {
         "StringEquals": {
            "ec2:Owner": "amazon"
         }
      }
   },
   {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": [
         "arn:aws:ec2:us-east-1:111122223333:instance/*",
         "arn:aws:ec2:us-east-1:111122223333:subnet/*",
         "arn:aws:ec2:us-east-1:111122223333:volume/*",
         "arn:aws:ec2:us-east-1:111122223333:network-interface/*",
         "arn:aws:ec2:us-east-1:111122223333:key-pair/*",
         "arn:aws:ec2:us-east-1:111122223333:security-group/*"
         ]
      }
   ]
}
```

インスタンスタイプ

次のポリシーにより、ユーザーは t2.micro または t2.small インスタンスタイプのみを使用してインスタンスを起動できます。これにより、コストを管理することができます。最初のステートメン

トの Condition エレメントは ec2:InstanceType が t2.micro または t2.small のどちらであるかをテストするため、ユーザーは大きなインスタンスを起動することはできません。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
        {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": [
         "arn:aws:ec2:us-east-1:111122223333:instance/*"
      ],
      "Condition": {
         "StringEquals": {
            "ec2:InstanceType": ["t2.micro", "t2.small"]
         }
      }
   },
   {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": [
         "arn:aws:ec2:us-east-1::image/ami-*",
         "arn:aws:ec2:us-east-1:111122223333:subnet/*",
         "arn:aws:ec2:us-east-1:111122223333:network-interface/*",
         "arn:aws:ec2:us-east-1:111122223333:volume/*",
         "arn:aws:ec2:us-east-1:111122223333:key-pair/*",
         "arn:aws:ec2:us-east-1:111122223333:security-group/*"
         ]
      }
   ]
}
```

また、ユーザーが t2.micro と t2.small のインスタンスタイプ以外のすべてのインスタンス起動へのアクセスを拒否するポリシーを作成することもできます。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
        {
      "Effect": "Deny",
      "Action": "ec2:RunInstances",
      "Resource": [
         "arn:aws:ec2:us-east-1:111122223333:instance/*"
      ],
      "Condition": {
         "StringNotEquals": {
            "ec2:InstanceType": ["t2.micro", "t2.small"]
         }
      }
   },
   {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": [
         "arn:aws:ec2:us-east-1::image/ami-*",
         "arn:aws:ec2:us-east-1:111122223333:network-interface/*",
         "arn:aws:ec2:us-east-1:111122223333:instance/*",
         "arn:aws:ec2:us-east-1:111122223333:subnet/*",
         "arn:aws:ec2:us-east-1:111122223333:volume/*",
         "arn:aws:ec2:us-east-1:111122223333:key-pair/*",
         "arn:aws:ec2:us-east-1:111122223333:security-group/*"
         ]
      }
   ]
}
```

サブネット

次のポリシーにより、ユーザーは指定したサブネット subnet-*12345678* のみを使用してインスタンスを起動できます。グループはインスタンスを他のサブネットに起動することはできません (他のステートメントがそのような許可をユーザーに与えている場合はその限りではありません)。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": [
        "arn:aws:ec2:us-east-1:111122223333:subnet/subnet-12345678",
        "arn:aws:ec2:us-east-1:111122223333:network-interface/*",
        "arn:aws:ec2:us-east-1:111122223333:instance/*",
        "arn:aws:ec2:us-east-1:111122223333:volume/*",
        "arn:aws:ec2:us-east-1::image/ami-*",
        "arn:aws:ec2:us-east-1:111122223333:key-pair/*",
        "arn:aws:ec2:us-east-1:111122223333:security-group/*"
      ]
    }
   ]
}
```

また、ユーザーがその他のサブネットにインスタンスを起動するアクセス許可を拒否するポリシーを作成することもできます。ステートメントではサブネット subnet-*12345678* が指定されている場合以外はネットワークインターフェイスの作成を拒否することでこれを実行します。この拒否は他のサブネットへのインスタンスの起動を許可する他のすべてのポリシーよりも優先されます。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
         {
      "Effect": "Deny",
      "Action": "ec2:RunInstances",
      "Resource": [
         "arn:aws:ec2:us-east-1:111122223333:network-interface/*"
      ],
      "Condition": {
         "ArnNotEquals": {
```

```
            "ec2:Subnet": "arn:aws:ec2:us-east-1:111122223333:subnet/
subnet-12345678"
         }
      }
   },
   {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": [
         "arn:aws:ec2:us-east-1::image/ami-*",
         "arn:aws:ec2:us-east-1:111122223333:network-interface/*",
         "arn:aws:ec2:us-east-1:111122223333:instance/*",
         "arn:aws:ec2:us-east-1:111122223333:subnet/*",
         "arn:aws:ec2:us-east-1:111122223333:volume/*",
         "arn:aws:ec2:us-east-1:111122223333:key-pair/*",
         "arn:aws:ec2:us-east-1:111122223333:security-group/*"
         ]
      }
   ]
}
```

EBS ボリューム

次のポリシーではインスタンスの EBS ボリュームが暗号化されている場合のみユーザーがインスタンスを起動できます。ユーザーはルートボリュームが暗号化されるように、暗号化されたスナップショットを使用して作成された AMI からインスタンスを起動する必要があります。ユーザーが起動時にインスタンスにアタッチする追加ボリュームも暗号化されている必要があります。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
                {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:*:*:volume/*"
            ],
            "Condition": {
                "Bool": {
```

```
                    "ec2:Encrypted": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:*::image/ami-*",
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:instance/*",
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:key-pair/*",
                "arn:aws:ec2:*:*:security-group/*"
            ]
        }
    ]
}
```

タグ

インスタンスの作成時にタグを付ける

次のポリシーではユーザーがインスタンスを起動し、作成時にインスタンスにタグ付けすることができます。タグを適用するリソース作成アクションにはユーザーが CreateTags アクションを使用するアクセス権限を持っていることが必要です。2 番目のステートメントはec2:CreateAction条件キーを使用し、ユーザーが RunInstances のコンテキストでのみ、インスタンスに対してのみタグを作成できるようにします。ユーザーは既存のリソースにタグ付けすることができず、RunInstances リクエストを使用してボリュームにタグ付けすることもできません。

詳細についてはAmazon EC2 リソース作成時にタグ付けするアクセス許可の付与を参照してください。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
```

```
            "ec2:RunInstances"
         ],
         "Resource": "*"
      },
      {
         "Effect": "Allow",
         "Action": [
            "ec2:CreateTags"
         ],
         "Resource": "arn:aws:ec2:us-east-1:111122223333:instance/*",
         "Condition": {
            "StringEquals": {
                "ec2:CreateAction" : "RunInstances"
             }
          }
      }
   ]
}
```

インスタンスやボリュームの作成時に特定のタグを付ける

次のポリシーにはaws:RequestTag および RunInstances タグを使用して environment=production により作成されたすべてのインスタンスおよびボリュームへのタグ付けをユーザーに求める purpose=webserver 条件キーが含まれています。ユーザーがこれらのタグを渡さないか、タグをまったく指定しない場合、リクエストは失敗します。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
   {
      "Effect": "Allow",
      "Action": [
         "ec2:RunInstances"
      ],
      "Resource": [
         "arn:aws:ec2:us-east-1::image/*",
         "arn:aws:ec2:us-east-1:111122223333:subnet/*",
         "arn:aws:ec2:us-east-1:111122223333:network-interface/*",
         "arn:aws:ec2:us-east-1:111122223333:security-group/*",
```

```
            "arn:aws:ec2:us-east-1:111122223333:key-pair/*"
        ]
    },
    {
        "Effect": "Allow",
        "Action": [
            "ec2:RunInstances"
        ],
        "Resource": [
            "arn:aws:ec2:us-east-1:111122223333:volume/*",
            "arn:aws:ec2:us-east-1:111122223333:instance/*"
        ],
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/environment": "production" ,
                "aws:RequestTag/purpose": "webserver"
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": [
            "ec2:CreateTags"
        ],
        "Resource": "arn:aws:ec2:us-east-1:111122223333:*/*",
        "Condition": {
            "StringEquals": {
                "ec2:CreateAction" : "RunInstances"
            }
        }
    }
  ]
}
```

インスタンスやボリュームの作成時に特定のタグを少なくとも 1 つ付ける

次のポリシーはForAnyValue 条件で aws:TagKeys 修飾子を使用して、リクエストで少なくとも 1 つのタグが指定されている必要があり、キー environment または webserver が含まれている必要があることを示します。タグはインスタンスとボリュームの両方に適用される必要があります。リクエストでは任意のタグ値を指定できます。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
   {
      "Effect": "Allow",
      "Action": [
         "ec2:RunInstances"
      ],
      "Resource": [
         "arn:aws:ec2:us-east-1::image/*",
         "arn:aws:ec2:us-east-1:111122223333:subnet/*",
         "arn:aws:ec2:us-east-1:111122223333:network-interface/*",
         "arn:aws:ec2:us-east-1:111122223333:security-group/*",
         "arn:aws:ec2:us-east-1:111122223333:key-pair/*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
          "ec2:RunInstances"
      ],
      "Resource": [
          "arn:aws:ec2:us-east-1:111122223333:volume/*",
          "arn:aws:ec2:us-east-1:111122223333:instance/*"
      ],
      "Condition": {
          "ForAnyValue:StringEquals": {
              "aws:TagKeys": ["environment","webserver"]
          }
       }
    },
    {
      "Effect": "Allow",
      "Action": [
          "ec2:CreateTags"
      ],
      "Resource": "arn:aws:ec2:us-east-1:111122223333:*/*",
      "Condition": {
          "StringEquals": {
              "ec2:CreateAction" : "RunInstances"
          }
```

```
        }
      }
    ]
}
```

インスタンスの作成時にタグを付ける場合は特定のタグを使用してタグ付けする必要がある

次のポリシーではユーザーはリクエストでタグを指定する必要はありませんが、指定する場合は purpose=test タグを指定する必要があります。他のタグは許可されません。ユーザーはRunInstances リクエストでタグ付け可能なリソースにタグを適用できます。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
         "ec2:RunInstances"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
         "ec2:CreateTags"
      ],
      "Resource": "arn:aws:ec2:us-east-1:111122223333:*/*",
      "Condition": {
         "StringEquals": {
             "aws:RequestTag/purpose": "test",
             "ec2:CreateAction" : "RunInstances"
          },
          "ForAllValues:StringEquals": {
              "aws:TagKeys": "purpose"
          }
       }
    }
  ]
}
```

RunInstances の呼び出しで作成時のタグ付けをユーザーを禁止するには

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRun",
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1::image/*",
                "arn:aws:ec2:us-east-1:*:subnet/*",
                "arn:aws:ec2:us-east-1:*:network-interface/*",
                "arn:aws:ec2:us-east-1:*:security-group/*",
                "arn:aws:ec2:us-east-1:*:key-pair/*",
                "arn:aws:ec2:us-east-1:*:volume/*",
                "arn:aws:ec2:us-east-1:*:instance/*",
                "arn:aws:ec2:us-east-1:*:spot-instances-request/*"
            ]
        },
        {
            "Effect": "Deny",
            "Action": "ec2:CreateTags",
            "Resource": "*"
        }
    ]
}
```

spot-instances-request の特定のタグのみを許可します。ここで矛盾数 2 が意外な効果を発揮します。通常の状況ではタグを指定しないと非認証になります。spot-instances-request の場合、spot-instances-request タグがないと、このポリシーは評価されないため、タグなしの Spot on Run リクエストが成功します。

起動テンプレートのタグ

次の例で、ユーザーはインスタンスを起動できますが、特定の起動テンプレートを使用する場合に限ります (lt-09477bcd97b0d310e)。ec2:IsLaunchTemplateResource 条件キーはユーザーが起動テンプレートで指定されたリソースを上書きしないようにします。ステートメントの 2 番目の部分ではユーザーは作成時にインスタンスにタグ付けできます — ステートメントのこの部分は起動テンプレートでタグがインスタンスに対して指定されている場合に必要になります。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
   {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": "*",
      "Condition": {
         "ArnLike": {
             "ec2:LaunchTemplate": "arn:aws:ec2:us-east-1:111122223333:launch-
template/lt-09477bcd97b0d310e"
          },
          "Bool": {
             "ec2:IsLaunchTemplateResource": "true"
          }
       }
    },
    {
      "Effect": "Allow",
      "Action": [
         "ec2:CreateTags"
      ],
      "Resource": "arn:aws:ec2:us-east-1:111122223333:instance/*",
      "Condition": {
         "StringEquals": {
             "ec2:CreateAction" : "RunInstances"
          }
       }
    }
  ]
}
```

Elastic GPU

次のポリシーではユーザーはインスタンスを起動させ、インスタンスにアタッチする Elastic GPU を指定できます。ユーザーは任意のリージョンでインスタンスを起動できますが、Elastic GPU をアタッチできるのはその us-east-2 リージョンでの起動中に限られます。

ec2:ElasticGpuType条件キーはeg1.mediumeg1.largeインスタンスがまたはエラスティック GPU タイプのいずれかを使用することを保証します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
             {
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:*:111122223333:elastic-gpu/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:Region": "us-east-2",
                    "ec2:ElasticGpuType": [
                        "eg1.medium",
                        "eg1.large"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:*::image/ami-*",
                "arn:aws:ec2:*:111122223333:network-interface/*",
                "arn:aws:ec2:*:111122223333:instance/*",
                "arn:aws:ec2:*:111122223333:subnet/*",
                "arn:aws:ec2:*:111122223333:volume/*",
                "arn:aws:ec2:*:111122223333:key-pair/*",
```

```
                "arn:aws:ec2:*:111122223333:security-group/*"
            ]
        }
    ]
}
```

起動テンプレート

次の例で、ユーザーはインスタンスを起動できますが、特定の起動テンプレートを使用する場合に限ります (lt-09477bcd97b0d310e)。ユーザーはRunInstances アクションでパラメータを指定することで、起動テンプレートのパラメータを上書きできます。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
        {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": "*",
      "Condition": {
         "ArnLike": {
             "ec2:LaunchTemplate": "arn:aws:ec2:us-east-1:111122223333:launch-
template/lt-09477bcd97b0d310e"
          }
       }
    }
  ]
}
```

この例で、ユーザーは起動テンプレートを使用する場合に限りインスタンスを起動できます。ポリシーでは ec2:IsLaunchTemplateResource 条件キーを使用して、ユーザーが起動テンプレート内の既存の ARN を上書きできないようにします。

JSON

```
{
  "Version":"2012-10-17",
```

```
  "Statement": [
        {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": "*",
      "Condition": {
         "ArnLike": {
             "ec2:LaunchTemplate": "arn:aws:ec2:us-east-1:111122223333:launch-
template/*"
          },
          "Bool": {
             "ec2:IsLaunchTemplateResource": "true"
          }
       }
    }
  ]
}
```

次のサンプルポリシーによりユーザーはインスタンスを起動できますが、起動テンプレートを使用する場合に限ります。ユーザーはリクエストでサブネットおよびネットワークインターフェイスのパラメータを上書きすることはできません。これらのパラメータは起動テンプレートでのみ指定できます。ステートメントの最初の部分はNotResource 要素を使用して、サブネットやネットワークインターフェイスを除くその他のすべてのリソースを許可します。ステートメントの 2 番目の部分はサブネットおよびネットワークインターフェイスのリソースを許可しますが、これは起動テンプレートから取得された場合に限ります。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
        {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "NotResource": ["arn:aws:ec2:us-east-1:111122223333:subnet/*",
                      "arn:aws:ec2:us-east-1:111122223333:network-interface/*" ],
      "Condition": {
         "ArnLike": {
             "ec2:LaunchTemplate": "arn:aws:ec2:us-east-1:111122223333:launch-
template/*"
```

```
            }
         }
      },
      {
         "Effect": "Allow",
         "Action": "ec2:RunInstances",
         "Resource": ["arn:aws:ec2:us-east-1:111122223333:subnet/*",
                      "arn:aws:ec2:us-east-1:111122223333:network-interface/*" ],
         "Condition": {
            "ArnLike": {
               "ec2:LaunchTemplate": "arn:aws:ec2:us-east-1:111122223333:launch-
template/*"
            },
            "Bool": {
               "ec2:IsLaunchTemplateResource": "true"
            }
         }
      }
   ]
}
```

次の例では起動テンプレートを使用していて、また起動テンプレートにタグがある場合に限り、ユーザーはインスタンスを起動できるようになります Purpose=Webservers。ユーザーはRunInstances アクションで起動テンプレートパラメータを上書きすることはできません。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
        {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "NotResource": "arn:aws:ec2:us-east-1:111122223333:launch-template/*",
      "Condition": {
         "ArnLike": {
            "ec2:LaunchTemplate": "arn:aws:ec2:us-east-1:111122223333:launch-
template/*"
         },
         "Bool": {
            "ec2:IsLaunchTemplateResource": "true"
```

```
            }
          }
      },
      {
        "Effect": "Allow",
        "Action": "ec2:RunInstances",
        "Resource": "arn:aws:ec2:us-east-1:111122223333:launch-template/*",
        "Condition": {
         "StringEquals": {
             "aws:ResourceTag/Purpose": "Webservers"
          }
         }
       }
    ]
}
```

スポットインスタンス の操作

RunInstances アクションを使用してスポットインスタンスリクエストを作成し、作成時にスポットインスタンスリクエストにタグ付けできます。RunInstances に指定するリソースは `spot-instances-request` です。

`spot-instances-request` リソースはIAM ポリシーで次のように評価されます。

- スポットインスタンスリクエストの作成時にタグを付けない場合、Amazon EC2 は RunInstances ステートメント内の `spot-instances-request` リソースを評価しません。
- スポットインスタンスリクエストの作成時にタグを付けると、 RunInstances ステートメント内の `spot-instances-request` リソースが、Amazon EC2 により評価されます。

したがって、`spot-instances-request` リソースの場合、次のルールが IAM ポリシーに適用されます。

- RunInstances を使用してスポットインスタンスリクエストを作成し、その際リクエストにタグを付けない場合は`spot-instances-request` リソースを明示的に許可しなくても、その呼び出しは成功します。
- RunInstances を使用してスポットインスタンスリクエストを作成する際に、そのリクエストにタグを付ける場合にはRunInstances の許可ステートメントに `spot-instances-request` リソースを含める必要があります。これがない場合は呼び出しが失敗します。

- RunInstances を使用してスポットインスタンスリクエストを作成し、作成時にタグを付ける場合はCreateTags 許可ステートメントに `spot-instances-request` リソースまたは * ワイルドカードを指定する必要があります。指定しないと、呼び出しは失敗します。

スポットインスタンス はRunInstances または RequestSpotInstances を使用してリクエストできます。次の例の IAM ポリシーはRunInstances を使用して スポットインスタンス をリクエストする場合にのみ適用されます。

例: RunInstances を使用して スポットインスタンス をリクエストする

次のポリシーではRunInstances アクションを使用して スポットインスタンス をリクエストすることをユーザーに許可します。`spot-instances-request` リソースはRunInstances によって作成されます。このリソースは スポットインスタンス をリクエストします。

> **Note**
>
> RunInstances を使用してスポットインスタンスリクエストを作成し、作成時にタグを付けない場合は`spot-instances-request` リストから `Resource` を省略できます。これはスポットインスタンスリクエストの作成時にタグを付けない場合、Amazon EC2 は RunInstances ステートメント内の `spot-instances-request` リソースを評価しないためです。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRun",
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1::image/*",
                "arn:aws:ec2:us-east-1:*:subnet/*",
                "arn:aws:ec2:us-east-1:*:network-interface/*",
                "arn:aws:ec2:us-east-1:*:security-group/*",
                "arn:aws:ec2:us-east-1:*:key-pair/*",
```

```
                "arn:aws:ec2:us-east-1:*:volume/*",
                "arn:aws:ec2:us-east-1:*:instance/*",
                "arn:aws:ec2:us-east-1:*:spot-instances-request/*"
            ]
        }
    ]
}
```

⚠ Warning

サポート対象外 – 例: RunInstances を使用して スポットインスタンス をリクエストするためのアクセス許可をユーザーに拒否する
次のポリシーはspot-instances-requestリソースではサポートされません。
次のポリシーではユーザーに オンデマンドインスタンス を起動するためのアクセス許可を付与しますが、スポットインスタンス をリクエストするためのアクセス許可を拒否します。spot-instances-request リソースはRunInstances によって作成されます。このリソースは スポットインスタンス をリクエストします。2 番目のステートメントではspot-instances-request リソースに対する RunInstances アクションを拒否します。ただし、スポットインスタンスリクエストの作成時にタグを付けない場合、Amazon EC2 が RunInstances ステートメントの spot-instances-request リソースを評価しないため、この条件はサポートされません。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRun",
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1::image/*",
                "arn:aws:ec2:us-east-1:*:subnet/*",
                "arn:aws:ec2:us-east-1:*:network-interface/*",
                "arn:aws:ec2:us-east-1:*:security-group/*",
                "arn:aws:ec2:us-east-1:*:key-pair/*",
```

```
                "arn:aws:ec2:us-east-1:*:volume/*",
                "arn:aws:ec2:us-east-1:*:instance/*"
            ]
        },
        {
            "Sid": "DenySpotInstancesRequestsNOTSUPPORTEDDONOTUSE",
            "Effect": "Deny",
            "Action": "ec2:RunInstances",
            "Resource": "arn:aws:ec2:us-east-1:*:spot-instances-request/*"
        }
    ]
}
```

例: スポットインスタンスリクエストの作成時にタグを付ける

次のポリシーではインスタンスの起動時に作成されるすべてのリソースにタグを付けることをユーザーに許可します。最初のステートメントでは一覧表示されたリソースの作成を RunInstances に許可します。spot-instances-request リソースはRunInstances によって作成されます。このリソースは スポットインスタンス をリクエストします。2 番目のステートメントでは* ワイルドカードを指定し、インスタンスの起動時に作成されるすべてのリソースのタグ付けを許可します。

> **Note**
>
> スポットインスタンスリクエストの作成時にタグを付けると、 RunInstances ステートメント内の spot-instances-request リソースが、Amazon EC2 により評価されます。したがって、RunInstances アクションで spot-instances-request リソースを明示的に許可する必要があります。許可しないと、呼び出しは失敗します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRun",
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
```

```
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1::image/*",
                "arn:aws:ec2:us-east-1:*:subnet/*",
                "arn:aws:ec2:us-east-1:*:network-interface/*",
                "arn:aws:ec2:us-east-1:*:security-group/*",
                "arn:aws:ec2:us-east-1:*:key-pair/*",
                "arn:aws:ec2:us-east-1:*:volume/*",
                "arn:aws:ec2:us-east-1:*:instance/*",
                "arn:aws:ec2:us-east-1:*:spot-instances-request/*"
            ]
        },
        {
            "Sid": "TagResources",
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "*"
        }
    ]
}
```

例: スポットインスタンスリクエストの作成時にタグ付けを拒否する

次のポリシーではインスタンスの起動時に作成されるリソースにタグを付けるためのアクセス許可をユーザーに拒否します。

最初のステートメントでは一覧表示されたリソースの作成を RunInstances に許可します。spot-instances-request リソースはRunInstances によって作成されます。このリソースは スポットインスタンス をリクエストします。2 番目のステートメントでは* ワイルドカードを指定し、インスタンスの起動時に作成されるすべてのリソースのタグ付けを拒否します。spot-instances-request リソースまたは他のリソースの作成時にタグを付けた場合、RunInstances の呼び出しは失敗します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRun",
            "Effect": "Allow",
```

```
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1::image/*",
                "arn:aws:ec2:us-east-1:*:subnet/*",
                "arn:aws:ec2:us-east-1:*:network-interface/*",
                "arn:aws:ec2:us-east-1:*:security-group/*",
                "arn:aws:ec2:us-east-1:*:key-pair/*",
                "arn:aws:ec2:us-east-1:*:volume/*",
                "arn:aws:ec2:us-east-1:*:instance/*",
                "arn:aws:ec2:us-east-1:*:spot-instances-request/*"
            ]
        },
        {
            "Sid": "DenyTagResources",
            "Effect": "Deny",
            "Action": "ec2:CreateTags",
            "Resource": "*"
        }
    ]
}
```

Warning

サポート対象外 - 例: スポットインスタンスリクエストに特定のタグを割り当てる場合にのみ、リクエストの作成を許可する
次のポリシーは`spot-instances-request` リソースではサポートされません。
次のポリシーはスポットインスタンスリクエストに特定のタグを付ける場合にのみ、リクエストを作成するためのアクセス許可を RunInstances に付与することを想定しています。最初のステートメントでは一覧表示されたリソースの作成を RunInstances に許可します。2 番目のステートメントではスポットインスタンスリクエストにタグ `environment=production` が付いている場合にのみ、リクエストを作成するためのアクセス許可をユーザーに付与することを想定しています。この条件を RunInstances によって作成された他のリソースに適用する場合、タグを指定しないと、`Unauthenticated` エラーが発生します。ただし、スポットインスタンスリクエストにタグを指定しない場合、Amazon EC2 は RunInstances ステートメントの `spot-instances-request` リソースを評価しないため、RunInstances がタグなしのスポットインスタンスリクエストを作成します。

environment=production 以外の別のタグを指定すると、Unauthenticated エラーが発生することに注意してください。これはユーザーがスポットインスタンスリクエストにタグを付けると、Amazon EC2 が RunInstances ステートメントの spot-instances-request リソースを評価するためです。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRun",
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1::image/*",
                "arn:aws:ec2:us-east-1:*:subnet/*",
                "arn:aws:ec2:us-east-1:*:network-interface/*",
                "arn:aws:ec2:us-east-1:*:security-group/*",
                "arn:aws:ec2:us-east-1:*:key-pair/*",
                "arn:aws:ec2:us-east-1:*:volume/*",
                "arn:aws:ec2:us-east-1:*:instance/*"
            ]
        },
        {
            "Sid":
 "RequestSpotInstancesOnlyIfTagIsEnvironmentProductionNOTSUPPORTEDDONOTUSE",
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": "arn:aws:ec2:us-east-1:*:spot-instances-request/
*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/environment": "production"
                }
            }
        },
        {
            "Sid": "TagResources",
            "Effect": "Allow",
```

```
            "Action": "ec2:CreateTags",
            "Resource": "*"
        }

    ]
}
```

例: スポットインスタンスリクエストに特定のタグが割り当てられている場合、このリクエストの作成を拒否する

次のポリシーはスポットインスタンスリクエストにタグ environment=production が付いている場合、このリクエストを作成するためのアクセス許可を RunInstances に拒否します。

最初のステートメントでは一覧表示されたリソースの作成を RunInstances に許可します。

2 番目のステートメントではスポットインスタンスリクエストにタグ environment=production が付いている場合、このリクエストを作成するためのアクセス許可をユーザーに拒否します。environment=production をタグとして指定すると、Unauthenticated エラーが発生します。他のタグを指定するか、タグを指定しないと、スポットインスタンスリクエストが作成されます。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRun",
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1::image/*",
                "arn:aws:ec2:us-east-1:*:subnet/*",
                "arn:aws:ec2:us-east-1:*:network-interface/*",
                "arn:aws:ec2:us-east-1:*:security-group/*",
                "arn:aws:ec2:us-east-1:*:key-pair/*",
                "arn:aws:ec2:us-east-1:*:volume/*",
                "arn:aws:ec2:us-east-1:*:instance/*",
```

```
                "arn:aws:ec2:us-east-1:*:spot-instances-request/*"
            ]
        },
        {
            "Sid": "DenySpotInstancesRequests",
            "Effect": "Deny",
            "Action": "ec2:RunInstances",
            "Resource": "arn:aws:ec2:us-east-1:*:spot-instances-request/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/environment": "production"
                }
            }
        },
        {
            "Sid": "TagResources",
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "*"
        }
    ]
}
```

例: リザーブドインスタンス の操作

次のポリシーではアカウントで リザーブドインスタンス を表示、変更、購入するアクセス許可をユーザーに与えます。

個別の リザーブドインスタンス にリソースレベルのアクセス許可を設定することはできません。このポリシーはユーザーがアカウントのすべての リザーブドインスタンス にアクセスできることを意味します。

Resource 要素は * ワイルドカードを使用して、ユーザーがそのアクションにすべてのリソースを指定できることを示しています。この場合、アカウントのすべての リザーブドインスタンス をリストして変更できます。ユーザーはアカウント認証情報を使用して リザーブドインスタンス を購入することもできます。また、API アクションがリソースレベルのアクセス許可をサポートしていない場合も、* ワイルドカードが必要です。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeReservedInstances",
        "ec2:ModifyReservedInstances",
        "ec2:PurchaseReservedInstancesOffering",
        "ec2:DescribeAvailabilityZones",
        "ec2:DescribeReservedInstancesOfferings"
      ],
      "Resource": "*"
    }
   ]
}
```

次のコードではアカウント内の リザーブドインスタンス を表示および変更できるようにユーザーに許可しています。新しい リザーブドインスタンス の購入は許可していません。

JSON

```
{
  "Version":"2012-10-17",
   "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeReservedInstances",
        "ec2:ModifyReservedInstances",
        "ec2:DescribeAvailabilityZones"
      ],
      "Resource": "*"
    }
  ]
}
```

例: リソースのタグ付け

次のポリシーではタグにキー CreateTags および値 environment が含まれている場合のみ、ユーザーが production アクションを使用してインスタンスにタグを適用できます。他のタグは許可されず、ユーザーは他のリソースタイプをタグ付けすることはできません。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
              {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:us-east-1:111122223333:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/environment": "production"
                }
            }
        }
    ]
}
```

次のポリシーではユーザーは owner のキーとユーザー名の値を使用したタグが既に適用されているタグ付け可能なリソースにタグ付けできます。加えて、ユーザーはリクエストで anycompany:environment-type のキーと値 test または prod を持つタグを指定する必要があります。ユーザーはリクエストで追加のタグを指定できます。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
              {
            "Effect": "Allow",
```

```
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:us-east-1:111122223333:*/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/anycompany:environment-type":
 ["test","prod"],
                    "aws:ResourceTag/owner": "${aws:username}"
                }
            }
        }
    ]
}
```

ユーザーがリソースの特定のタグを指定できるようにする IAM ポリシーを作成できます。例えば、次のポリシーではリクエストで指定されたタグキーが environment または cost-center の場合、ユーザーがボリュームのタグを削除できます。タグにはどの値でも指定できますが、指定されたキーのいずれかにタグキーが一致する必要があります。

> **Note**
>
> リソースを削除すると、リソースに関連付けられているすべてのタグも削除されます。タグ付きのリソースを削除する場合、ユーザーは ec2:DeleteTags アクションを使用するためのアクセス許可は必要ありません。削除アクションを実行するためのアクセス許可のみが必要です。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
       {
      "Effect": "Allow",
      "Action": "ec2:DeleteTags",
      "Resource": "arn:aws:ec2:us-east-1:111122223333:volume/*",
      "Condition": {
        "ForAllValues:StringEquals": {
          "aws:TagKeys": ["environment","cost-center"]
```

```
        }
      }
    }
  ]
}
```

このポリシーではリソースが owner のキーとユーザー名の値で既にタグ付けされている場合のみ、ユーザーが任意のリソースで environment=prod タグのみ削除できます。ユーザーはリソースの他のタグを削除することはできません。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
      {
      "Effect": "Allow",
      "Action": [
        "ec2:DeleteTags"
      ],
      "Resource": "arn:aws:ec2:us-east-1:111122223333:*/*",
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/environment": "prod",
          "aws:ResourceTag/owner": "${aws:username}"
        },
        "ForAllValues:StringEquals": {
          "aws:TagKeys": ["environment"]
        }
      }
    }
  ]
}
```

例: IAM ロールの使用

次のポリシーではdepartment=test タグを持つインスタンス対して IAM ロールのアタッチ、置換、デタッチを行うことをユーザーに許可します。IAM ロールの置換またはデタッチには関連 ID が

必要であるため、ポリシーでは ec2:DescribeIamInstanceProfileAssociations アクションを使用するアクセス許可もユーザーに付与します。

ユーザーはロールをインスタンスに渡すために iam:PassRole アクションを使用するための許可が必要です。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:AssociateIamInstanceProfile",
        "ec2:ReplaceIamInstanceProfileAssociation",
        "ec2:DisassociateIamInstanceProfile"
      ],
      "Resource": "arn:aws:ec2:us-east-1:111122223333:instance/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/department":"test"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "ec2:DescribeIamInstanceProfileAssociations",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::111122223333:role/DevTeam*"
    }
  ]
}
```

次のポリシーではどのインスタンスに対しても IAM ロールのアタッチまたは置換を行うことをユーザーに許可します。ユーザーはTestRole- で始まる名前の IAM ロールのみアタッチまたは置換できます。iam:PassRole アクションではインスタンスプロファイルではなく IAM ロールの名前を指

定します (両方の名前が異なる場合)。詳細についてはインスタンスプロファイルを参照してください。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AssociateIamInstanceProfile",
                "ec2:ReplaceIamInstanceProfileAssociation"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeIamInstanceProfileAssociations",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::111122223333:role/TestRole-*"
        }
    ]
}
```

例: ルートテーブルの使用

次のポリシーではVPC (vpc-ec43eb89) のみに関連付けられているルートテーブルのルートの追加、削除、置換を行うことができます。ec2:Vpc 条件キーの VPC を指定するにはVPC の完全な ARN を指定する必要があります。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
            {
            "Effect": "Allow",
            "Action": [
                "ec2:DeleteRoute",
                "ec2:CreateRoute",
                "ec2:ReplaceRoute"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1:111122223333:route-table/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:Vpc": "arn:aws:ec2:us-east-1:111122223333:vpc/vpc-
ec43eb89"
                }
            }
        }
    ]
}
```

例: 特定のインスタンスが他の AWS サービスでリソースを表示できるようにする

次に示すのはIAM ロールにアタッチできるポリシーの例です。ポリシーにより、インスタンスは AWS サービスのさまざまなリソースを表示できるようになります。このポリシーではグローバル条件キー `ec2:SourceInstanceARN` を使用して、リクエスト元のインスタンスがインスタンス `i-093452212644b0dd6` である必要があることを指定します。同じ IAM ロールが別のインスタンスと関連付けられている場合、他のインスタンスはこれらのどのアクションも実行できません。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
            {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeVolumes",
                "s3:ListAllMyBuckets",
                "dynamodb:ListTables",
                "rds:DescribeDBInstances"
            ],
```

```
            "Resource": [
                "*"
            ],
            "Condition": {
                "ArnEquals": {
                    "ec2:SourceInstanceARN": "arn:aws:ec2:us-
east-1:111122223333:instance/i-0934522126440b0dd6"
                }
            }
        }
    ]
}
```

例: 起動テンプレートの使用

次のポリシーではユーザーは起動テンプレートのバージョンを作成して起動テンプレートを変更することができます。ただし、特定の起動テンプレートに限られます (`lt-09477bcd97b0d3abc`)。ユーザーは他の起動テンプレートを使用することはできません。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
   {
      "Action": [
        "ec2:CreateLaunchTemplateVersion",
        "ec2:ModifyLaunchTemplate"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:ec2:us-east-1:111122223333:launch-template/
lt-09477bcd97b0d3abc"
    }
  ]
}
```

次のポリシーではユーザーは任意の起動テンプレートと起動テンプレートのバージョンを削除できます。ただし、起動テンプレートに `Purpose=Testing` のタグがある場合に限ります。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
       {
      "Action": [
        "ec2:DeleteLaunchTemplate",
        "ec2:DeleteLaunchTemplateVersions"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:ec2:us-east-1:111122223333:launch-template/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Purpose": "Testing"
        }
      }
    }
  ]
}
```

インスタンスメタデータの使用

以下のポリシーではインスタンスメタデータサービスバージョン 2 (IMDSv2) を使用して、ユーザーがインスタンスメタデータのみを取得できるようにします。以下の 4 つのポリシーは4 つのステートメントを使用する 1 つのポリシーに結合できます。1 つのポリシーとして結合すると、このポリシーをサービスコントロールポリシー (SCP) として使用できます。これは既存の IAM ポリシーに適用する拒否ポリシーとして (既存のアクセス許可を削除して制限するために) 使用したり、アカウント、組織単位 (OU)、組織全体にグローバルに適用する SCP として使用したりすることもできます。

> **Note**
>
> 以下の RunInstances メタデータオプションポリシーはRunInstances を使用してインスタンスを起動するアクセス許可をプリンシパルに付与するポリシーと組み合わせて使用する必要があります。プリンシパルに RunInstances アクセス許可もない場合、インスタンスを起動することはできません。詳細についてはインスタンスの使用とインスタンスの起動 (RunInstances)のポリシーを参照してください。

Important

Auto Scaling グループを使用し、すべての新しいインスタンスで IMDSv2 の使用を要求する必要がある場合は、Auto Scaling グループで起動テンプレートを使用する必要があります。Auto Scaling グループが起動テンプレートを使用する場合、新しい Auto Scaling グループが作成されるときに IAM プリンシパルの `ec2:RunInstances` アクセス許可がチェックされます。また、既存の Auto Scaling グループが更新され、新しい起動テンプレートまたは新しいバージョンの起動テンプレートが使用される場合にもチェックされます。`RunInstances` の IAM プリンシパルでの IMDSv1 の使用に関する制限は起動テンプレートを使用している Auto Scaling グループが作成または更新された場合にのみチェックされます。`Latest` または `Default` 起動テンプレートを使用するように設定された Auto Scaling グループでは起動テンプレートの新しいバージョンが作成されたときにアクセス許可はチェックされません。アクセス許可をチェックするには特定のバージョンの起動テンプレートを使用するように Auto Scaling グループを設定する必要があります。

Auto Scaling グループによって起動されるインスタンスで IMDSv2 の使用を強制するには以下の追加ステップが必要です。

1. 作成された新しいプリンシパルのサービスコントロールポリシー (SCP) または IAM アクセス許可の境界を使用して、組織内のすべてのアカウントの起動設定の使用を無効にします。Auto Scaling グループアクセス許可を持つ既存の IAM プリンシパルの場合、関連するポリシーをこの条件キーで更新します。起動設定の使用を無効にするには値が `"autoscaling:LaunchConfigurationName"` として指定された `null` 条件キーを使用して、関連する SCP、アクセス許可の境界、または IAM ポリシーを作成または変更します。
2. 新しい起動テンプレートの場合は起動テンプレートでインスタンスメタデータオプションを設定します。既存の起動テンプレートの場合は新しいバージョンの起動テンプレートを作成し、新しいバージョンでインスタンスメタデータオプションを設定します。
3. 起動テンプレートを使用するアクセス許可を任意のプリンシパルに付与するポリシーで、`$latest` を指定して `$default` と `"autoscaling:LaunchTemplateVersionSpecified": "true"` の関連付けを制限します。使用を特定のバージョンの起動テンプレートに制限することで、インスタンスメタデータオプションが設定されているバージョンを使用して新しいインスタンスを確実に起動できます。詳細についてはAmazon EC2 Auto Scaling API リファレンス (具体的には `Version` パラメータ) の「LaunchTemplateSpecification」を参照してください。

4. 起動設定を使用する Auto Scaling グループの場合、起動設定を起動テンプレートに置き換えます。詳細については「Amazon EC2 Auto Scaling ユーザーガイド」の「Create a launch template for an Auto Scaling group」を参照してください。
5. 起動テンプレートを使用する Auto Scaling グループの場合、インスタンスメタデータオプションが設定された新しい起動テンプレートを使用するか、インスタンスメタデータオプションが設定された現在の起動テンプレートの新しいバージョンを使用します。詳細については、「update-auto-scaling-group」を参照してください。

例

- IMDSv2 の使用を要求する
- IMDSv2 のオプトアウトを拒否する
- ホップ制限の最大値の指定
- インスタンスメタデータオプションを変更できるユーザーの制限
- IMDSv2 からロール認証情報を取得することを要求する

IMDSv2 の使用を要求する

次のポリシーではインスタンスが IMDSv2 の使用を要求するようにオプトインされていない限り (`"ec2:MetadataHttpTokens": "required"` で指定)、RunInstances API を呼び出せないように指定します。インスタンスが IMDSv2 を要求するように指定しないと、RunInstances API を呼び出したときに `UnauthorizedOperation` エラーが発生します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
                {
            "Sid": "RequireImdsV2",
            "Effect": "Deny",
            "Action": "ec2:RunInstances",
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringNotEquals": {
                    "ec2:MetadataHttpTokens": "required"
                }
```

```
            }
        }
    ]
}
```

IMDSv2 のオプトアウトを拒否する

次のポリシーではModifyInstanceMetadataOptions API を呼び出さないように指定し、IMDSv1 または IMDSv2 のオプションを許可します。ModifyInstanceMetadataOptions API を呼び出す場合はHttpTokens 属性を required に設定する必要があります。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Sid": "DenyIMDSv1HttpTokensModification",
        "Effect": "Deny",
        "Action": "ec2:ModifyInstanceMetadataOptions",
        "Resource": "arn:aws:ec2:*:*:instance/*",
        "Condition": {
            "StringNotEquals": {
                "ec2:Attribute/HttpTokens": "required"
            },
            "Null": {
                "ec2:Attribute/HttpTokens": false
            }
        }
    }]
}
```

ホップ制限の最大値の指定

次のポリシーではホップ制限を指定しない限り、RunInstances API を呼び出せないように指定します。また、ホップ制限を 3 以下にするように指定します。これを指定しないと、RunInstances API を呼び出したときに UnauthorizedOperation エラーが発生します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
                {
            "Sid": "MaxImdsHopLimit",
            "Effect": "Deny",
            "Action": "ec2:RunInstances",
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "NumericGreaterThan": {
                    "ec2:MetadataHttpPutResponseHopLimit": "3"
                }
            }
        }
    ]
}
```

インスタンスメタデータオプションを変更できるユーザーの制限

次のポリシーでは一般の管理者がインスタンスメタデータオプションを変更するロール ec2-imds-admins を持つユーザーのみに変更を行うことを許可します。ec2-imds-admins ロール以外のプリンシパルが ModifyInstanceMetadataOptions API を呼び出そうとすると、UnauthorizedOperation エラーが発生します。このステートメントはModifyInstanceMetadataOptions API の使用を制御するために使用できます。現在、ModifyInstanceMetadataOptions API 用の詳細なアクセスコントロール (条件) はありません。

IMDSv2 からロール認証情報を取得することを要求する

次のポリシーではこのポリシーを適用したロールを EC2 サービスが引き受けて、結果の認証情報をリクエストの署名に使用する場合はIMDSv2 から取得した EC2 ロールの認証情報を使用してリクエストに署名する必要があることを指定します。それ以外の場合はすべての API コールで UnauthorizedOperation エラーが発生します。このステートメント/ポリシーはリクエストが EC2 ロールの認証情報によって署名されていない場合は効果がないため、一般的に適用できます。

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
              {
           "Sid": "RequireAllEc2RolesToUseV2",
           "Effect": "Deny",
           "Action": "*",
           "Resource": "*",
           "Condition": {
               "NumericLessThan": {
                   "ec2:RoleDelivery": "2.0"
               }
           }
       }
    ]
}
```

Amazon EBS ボリュームとスナップショットの使用

Amazon EBS ボリュームとスナップショットを使用するポリシーの例については「Amazon EBS のアイデンティティベースのポリシー例」を参照してください。

Amazon EC2 コンソールへのアクセスを制御するポリシーの例

IAM ポリシーを使用して、Amazon EC2 を操作するために必要なアクセス許可をユーザーに付与することができます。詳細な手順については、「IAM ユーザーガイド」の「IAM ポリシーの作成」を参照してください。

コンソールではこの機能を実行するために追加の API アクションを使用するので、これらのポリシーは正常に動作しない可能性があります。例えば、`DescribeVolumes` API アクションのみを使用するアクセス許可を持つユーザーがコンソールでボリュームを表示しようとすると、エラーが発生します。このセクションでは、コンソールの特定の部分をユーザーが操作できるようになるポリシーを説明します。Amazon EC2 向けのポリシー作成の詳細については、以下の AWS セキュリティブログの投稿Granting Users Permission to Work in the Amazon EC2 Consoleを参照してください。

以下の例では、Amazon EC2 を使用するアクセス許可をユーザーに付与するために使用できるポリシーステートメントを示しています。各`###############`を独自の情報に置き換えます。これらのポリシーは、AWS マネジメントコンソールを使用して行われたリクエスト向けに設計されています。Amazon EC2 コンソールは複数の API アクションを呼び出して 1 つのリソースを表示する場合があり、ユーザーがタスクを試行してコンソールにエラーが表示されるまではそれが明確でない場

合があります。詳細については、AWS セキュリティブログの投稿「Granting Users Permission to Work in the Amazon EC2 Console」を参照してください。

例

- 例: 読み取り専用アクセス
- 例: EC2 インスタンス起動ウィザードの使用
- 例: セキュリティグループの操作
- 例: Elastic IP アドレスの操作
- 例: リザーブドインスタンス の操作

コンソールでタスクを実行するために必要な API アクションを探すには、呼び出しをログに記録する AWS CloudTrail などのサービスを使用できます。ポリシーにより特定のリソースを作成または変更するアクセス許可が付与されない場合、コンソールではエンコードされた診断情報のメッセージが表示されます。 の DecodeAuthorizationMessage API アクションAWS STS、または AWS CLI の decode-authorization-message コマンドを使用してメッセージをデコードできます。

例: 読み取り専用アクセス

ユーザーが Amazon EC2 コンソールですべてのリソースを表示できるようにするには、次の例と同じポリシーを使用します: 例: 読み取り専用アクセス。別のステートメントによりユーザーにアクセス許可が与えられない限り、ユーザーはリソースのアクションを実行したり新しいリソースを作成することができません。

インスタンス、AMI、スナップショットを表示する

代わりに、リソースのサブセットへの読み取り専用アクセスを提供できます。これを行うには、`ec2:Describe` API アクションの * (ワイルドカード) を各リソースの固有の `ec2:Describe` アクションに置き換えます。次のポリシーによりユーザーは Amazon EC2 コンソールですべてのインスタンス、AMI、およびスナップショットを表示できます。`ec2:DescribeTags` アクションにより、ユーザーはパブリック AMI を表示できます。コンソールでタグ付け情報にパブリック AMI を表示させる必要がありますが、ユーザーがプライベート AMI だけを表示できるようにするには、このアクションを削除できます。

JSON

```
{
   "Version":"2012-10-17",
```

```
   "Statement": [{
      "Effect": "Allow",
      "Action": [
         "ec2:DescribeInstances",
         "ec2:DescribeImages",
         "ec2:DescribeTags",
         "ec2:DescribeSnapshots"
      ],
      "Resource": "*"
   }
   ]
}
```

> **Note**
>
> Amazon EC2 `ec2:Describe*` API アクションは、リソースレベルのアクセス許可をサポートしていません。そのため、ユーザーがコンソールで表示できる個人のリソースを制御できません。したがって、上記のステートメントの `Resource` エレメントには、* (ワイルドカード) が必要です。どの Amazon EC2 API アクションでどの ARN を使用できるかの詳細については、Amazon EC2 のアクション、リソース、および条件キーを参照してください。

インスタンスと CloudWatch メトリクスを表示する

以下のポリシーは、ユーザーに対して Amazon EC2 コンソールでのインスタンスの表示、[Instances] ページの [Monitoring] タブでの CloudWatch アラームおよびメトリクスの表示を許可します。Amazon EC2 コンソールでは、アラームとメトリクスの表示に CloudWatch API を使用するため、ユーザーに対して `cloudwatch:DescribeAlarms`、`cloudwatch:DescribeAlarmsForMetric`、`cloudwatch:ListMet` および `cloudwatch:GetMetricData` のアクションを使用する許可を付与する必要があります。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
         "ec2:DescribeInstances",
```

```
            "ec2:DescribeInstanceTypes",
            "cloudwatch:DescribeAlarms",
            "cloudwatch:DescribeAlarmsForMetric",
            "cloudwatch:ListMetrics",
            "cloudwatch:GetMetricStatistics",
            "cloudwatch:GetMetricData"
        ],
        "Resource": "*"
    }
    ]
}
```

例: EC2 インスタンス起動ウィザードの使用

Amazon EC2 インスタンス起動ウィザードは、インスタンスを設定し、起動するためのオプションを提供する画面です。ユーザーがウィザードのオプションを操作できるように、API アクションを使用するアクセス許可をポリシーに含める必要があります。ポリシーにそれらのアクションを使用するアクセス許可が含まれない場合、ウィザードの一部の項目は適切にロードされず、ユーザーは起動を完了できません。

基本のインスタンス起動ウィザードのアクセス

起動を正常に完了させるには、ユーザーに ec2:RunInstances API アクションを使用するアクセス許可を付与し、少なくとも以下の API アクションを使用できるようにする必要があります。

- ec2:DescribeImages: AMI を表示して選択します。
- ec2:DescribeInstanceTypes: インスタンスタイプを表示および選択します。
- ec2:DescribeVpcs: 使用できるネットワークオプションを表示します。
- ec2:DescribeSubnets: 選択した VPC のすべての使用可能なサブネットを表示します。
- ec2:DescribeSecurityGroups または ec2:CreateSecurityGroup: 既存のセキュリティグループを表示および選択する、または新しいセキュリティグループを作成します。
- ec2:DescribeKeyPairs または ec2:CreateKeyPair: 既存のキーペアを選択する、または新しいキーペアを作成します。
- ec2:AuthorizeSecurityGroupIngress: インバウンドルールを追加します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeImages",
                "ec2:DescribeInstanceTypes",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeVpcs",
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups",
                "ec2:CreateSecurityGroup",
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:CreateKeyPair"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": "*"
        }
    ]
}
```

ポリシーに次のような API アクションを追加して、ユーザーに追加のオプションを提供できます。

- `ec2:DescribeAvailabilityZones`: 特定のアベイラビリティーゾーンを選択します。
- `ec2:DescribeNetworkInterfaces`: 選択したサブネットの既存のネットワークインターフェイスを表示および選択します。
- VPC セキュリティグループにアウトバウンドルールを追加するには、ユーザーに `ec2:AuthorizeSecurityGroupEgress` API アクションを使用するアクセス許可を付与する必要があります。既存のルールを変更または削除するには、ユーザーに関連する `ec2:RevokeSecurityGroup*` API アクションを使用するアクセス許可を付与する必要があります。

- ec2:CreateTags: により作成されたリソースにタグ付けする場合に使用します。RunInstances詳細については、「Amazon EC2 リソース作成時にタグ付けするアクセス許可の付与」を参照してください。ユーザーにこのアクションを使用する許可がなく、インスタンス起動ウィザードのタグ付けページでてタグを適用しようとした場合、起動に失敗します。

  > **Important**
  >
  > インスタンスの起動中に [Name] (名前) を指定すると、タグが作成され、ec2:CreateTags アクションが必要になります。ユーザーに ec2:CreateTags アクションを使用するアクセス許可を付与すると、aws:ResourceTag 条件キーを使用してユーザーによる他のリソースの使用を制限する能力が制限されるため、注意が必要です。ユーザーに ec2:CreateTags アクションを使用するアクセス許可を付与すると、ユーザーがそれらの制限を回避するためにリソースのタグを変更できます。詳細については、「属性ベースのアクセスを使用するアクセスの制御」を参照してください。

- AMI を選択するときに Systems Manager パラメータを使用するには、ポリシーに ssm:DescribeParameters と ssm:GetParameters を追加する必要があります。ssm:DescribeParameters は、ユーザーに Systems Manager パラメータを表示および選択する許可を付与します。ssm:GetParameters は、ユーザーに Systems Manager パラメータの値を取得する許可を付与します。また、特定の Systems Manager パラメータへのアクセスを制限することもできます。詳細については、このセクションの後半の特定の Systems Manager パラメータへのアクセスの制限を参照してください。

現在、Amazon EC2 Describe* API アクションは、リソースレベルの許可をサポートしていません。そのため、ユーザーがインスタンス起動ウィザードで表示できる個人のリソースを制限することはできません。ただし、ec2:RunInstances API アクションにリソースレベルのアクセス許可を適用して、ユーザーがインスタンスの起動に使用できるリソースを制限できます。ユーザーが使用する権限がないオプションを選択すると、起動は失敗します。

特定のインスタンスタイプ、サブネット、リージョンへのアクセスの制限

次のポリシーにより、ユーザーは Amazon が所有する AMI を使用して t2.micro インスタンスを特定のサブネット (subnet-1a2b3c4d) でのみ起動することができます。ユーザーは指定されたリージョンでのみ起動できます。ユーザーが異なるリージョンを選択するか、インスタンス起動ウィザードで異なるインスタンスタイプ、AMI、サブネットを選択すると、起動は失敗します。

最初のステートメントでは、上記の例で説明したように、インスタンス起動ウィザードでオプションを表示する許可または新しいオプションを作成する許可がユーザーに付与されます。2 番目のステー

トメントでは、ec2:RunInstances アクションでネットワークインターフェイス、ボリューム、キーペア、セキュリティグループ、サブネットリソースを使用するアクセス許可が付与されます。これは、ユーザーが VPC でインスタンスを起動するために必要です。ec2:RunInstances アクションの使用方法の詳細については、インスタンスの起動 (RunInstances)を参照してください。3 番目と 4 番目のステートメントは、それぞれインスタンスと AMI リソースを使用するための許可をユーザーに付与しますが、これは、インスタンスが t2.micro インスタンスであり、AMI が Amazon または特定の信頼できる検証済みのパートナーによって所有されている場合に限られます。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
         "ec2:DescribeInstances",
         "ec2:DescribeImages",
         "ec2:DescribeInstanceTypes",
         "ec2:DescribeKeyPairs",
         "ec2:CreateKeyPair",
         "ec2:DescribeVpcs",
         "ec2:DescribeSubnets", "ec2:DescribeSecurityGroups",
         "ec2:CreateSecurityGroup",
         "ec2:AuthorizeSecurityGroupIngress"
   ],
   "Resource": "*"
   },
   {
      "Effect": "Allow",
      "Action":"ec2:RunInstances",
      "Resource": [
         "arn:aws:ec2:us-east-2:111122223333:network-interface/*",
         "arn:aws:ec2:us-east-2:111122223333:volume/*",
         "arn:aws:ec2:us-east-2:111122223333:key-pair/*",
         "arn:aws:ec2:us-east-2:111122223333:security-group/*",
         "arn:aws:ec2:us-east-2:111122223333:subnet/subnet-1a2b3c4d"
      ]
   },
   {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": [
```

```
            "arn:aws:ec2:us-east-2:111122223333:instance/*"
        ],
        "Condition": {
            "StringEquals": {
                "ec2:InstanceType": "t2.micro"
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": "ec2:RunInstances",
        "Resource": [
              "arn:aws:ec2:us-east-2::image/ami-*"
        ],
        "Condition": {
            "StringEquals": {
                "ec2:Owner": "amazon"
            }
        }
    }
    ]
}
```

特定の Systems Manager パラメータへのアクセスの制限

次のポリシーは、特定の名前の Systems Manager パラメータを使用するアクセスを許可します。

1 つ目のステートメントは、インスタンス起動ウィザードで AMI を選択するときに Systems Manager パラメータを表示する許可をユーザーに付与します。2 つ目のステートメントは、`prod-*` という名前のパラメータのみを使用するアクセス許可をユーザーに付与します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "ssm:DescribeParameters"
        ],
        "Resource": "*"
    },
```

```
   {
      "Effect": "Allow",
      "Action": [
         "ssm:GetParameters"
      ],
     "Resource": "arn:aws:ssm:us-east-2:123456123456:parameter/prod-*"
   }
   ]
}
```

例: セキュリティグループの操作

セキュリティグループを表示し、ルールを追加/削除する

次のポリシーは、Amazon EC2 コンソールでセキュリティグループを表示し、インバウンドおよびアウトバウンドのルールを追加および削除し、タグ Department=Test を含む既存のセキュリティグループのルール説明を変更するアクセス許可をユーザーに付与します。

最初のステートメントの ec2:DescribeTags アクションにより、ユーザーはコンソールでタグを表示できます。これにより、ユーザーは変更できるセキュリティグループをより簡単に識別できます。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
         "ec2:DescribeSecurityGroups",
         "ec2:DescribeSecurityGroupRules",
         "ec2:DescribeTags"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
         "ec2:AuthorizeSecurityGroupIngress",
         "ec2:RevokeSecurityGroupIngress",
         "ec2:AuthorizeSecurityGroupEgress",
```

```
            "ec2:RevokeSecurityGroupEgress",
            "ec2:ModifySecurityGroupRules",
            "ec2:UpdateSecurityGroupRuleDescriptionsIngress",
            "ec2:UpdateSecurityGroupRuleDescriptionsEgress"
        ],
        "Resource": [
            "arn:aws:ec2:us-east-2:111122223333:security-group/*"
        ],
        "Condition": {
            "StringEquals": {
                "aws:ResourceTag/Department": "Test"
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": [
            "ec2:ModifySecurityGroupRules"
        ],
        "Resource": [
            "arn:aws:ec2:us-east-2:111122223333:security-group-rule/*"
        ]
    }
]}
```

[Create Security Group] ダイアログボックスの使用

ユーザーが Amazon EC2 コンソールの [Create Security Group] ダイアログボックスを使用して作業できるようにするポリシーを作成できます。このダイアログボックスを使用するには、ユーザーに少なくとも以下の API アクションを使用するアクセス許可を付与する必要があります。

- `ec2:CreateSecurityGroup`: 新しいセキュリティグループを作成するには
- `ec2:DescribeVpcs`: [VPC] リストに既存の VPC のリストを表示します。

これらのアクセス許可で、ユーザーは新しいセキュリティグループを正常に作成できますが、ルールを追加することはできません。[Create Security Group] ダイアログボックスでルールを操作するには、ポリシーに次の API アクションを追加します。

- `ec2:AuthorizeSecurityGroupIngress`: インバウンドルールを追加します。

- ec2:AuthorizeSecurityGroupEgress: VPC セキュリティグループにアウトバウンドルールを追加します。
- ec2:RevokeSecurityGroupIngress: 既存のインバウンドルールを変更または削除します。これは、ユーザーがコンソールで [Copy to new] 機能を使用できるようにするために役に立ちます。この機能により、[Create Security Group] ダイアログボックスが開き、選択したセキュリティグループと同じルールが追加されます。
- ec2:RevokeSecurityGroupEgress: VPC セキュリティグループのアウトバウンドルールを変更または削除します。これは、すべてのアウトバウンドトラフィックを許可するデフォルトのアウトバウンドルールを変更または削除する場合に役に立ちます。
- ec2:DeleteSecurityGroup: 無効なルールを保存できないときに対応します。コンソールでは、最初にセキュリティグループを作成し、次に指定されたルールを追加します。ルールが無効である場合、アクションは失敗し、コンソールによってセキュリティグループの削除が試行されます。引き続き、[Create Security Group] ダイアログボックスが利用できるため、ユーザーは無効なルールを修正してセキュリティグループを再作成できます。この API アクションは必須ではありませんが、ユーザーにこのアクションを使用するアクセス許可が付与されておらず、無効なルールを持つセキュリティグループを作成しようとすると、ルールのないセキュリティグループが作成され、後でルールを追加することが必要になります。
- ec2:UpdateSecurityGroupRuleDescriptionsIngress: 入力 (受信) セキュリティグループルールの説明を追加または更新するには
- ec2:UpdateSecurityGroupRuleDescriptionsEgress: 出力 (送信) セキュリティグループルールの説明を追加または更新するには
- ec2:ModifySecurityGroupRules: セキュリティグループのルールを変更します。
- ec2:DescribeSecurityGroupRules: セキュリティグループのルールを一覧表示します。

次のポリシーは、[Create Security Group] ダイアログボックスを使用し、特定の VPC (vpc-1a2b3c4d) に関連付けられたセキュリティグループに対してインバウンドおよびアウトバウンドのルールを作成するアクセス許可をユーザーに付与します。ユーザーは VPC のセキュリティグループを作成できますが、ルールを追加することはできません。同様に、ユーザーは VPC vpc-1a2b3c4d に関連付けられていないの既存のセキュリティグループにルールを追加することもできません。ユーザーには、コンソールですべてのセキュリティグループを表示するアクセス許可も付与されます。これにより、ユーザーはインバウンドルールを追加するセキュリティグループをより簡単に識別できるようになります。このポリシーは、ユーザーに VPC vpc-1a2b3c4d に関連付けられたセキュリティグループを削除するアクセス許可も付与します。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeSecurityGroups",
        "ec2:CreateSecurityGroup",
        "ec2:DescribeVpcs"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DeleteSecurityGroup",
        "ec2:AuthorizeSecurityGroupIngress",
        "ec2:AuthorizeSecurityGroupEgress"
      ],
      "Resource": "arn:aws:ec2:us-east-2:111122223333:security-group/*",
      "Condition":{
         "ArnEquals": {
            "ec2:Vpc": "arn:aws:ec2:us-east-2:111122223333:vpc/vpc-1a2b3c4d"
         }
      }
    }
   ]
}
```

例: Elastic IP アドレスの操作

Amazon EC2 コンソールで Elastic IP アドレスを確認することをユーザーに許可するには、`ec2:DescribeAddresses` アクションを使用するためのアクセス許可をユーザーに付与します。

Elastic IP アドレスの使用をユーザーに許可する場合は、ポリシーに次のアクションを追加できます。

- `ec2:AllocateAddress`: Elastic IP アドレスを割り当てます。

- `ec2:ReleaseAddress`: Elastic IP アドレスをリリースするには。
- `ec2:AssociateAddress`: Elastic IP アドレスをインスタンスまたはネットワークインターフェイスに関連付けます。
- `ec2:DescribeNetworkInterfaces` と `ec2:DescribeInstances`: [Associate address] で使用します。この画面には、Elastic IP アドレスを関連付けることができるインスタンスまたはネットワークインターフェイスが表示されます。
- `ec2:DisassociateAddress`: Elastic IP アドレスとインスタンスまたはネットワークインターフェイスの関連付けを解除します。

次のポリシーでは、Elastic IP アドレスの表示、割り当て、インスタンスとの関連付けを行うことができます。ユーザーは Elastic IP アドレスとネットワークインターフェイスの関連付け、Elastic IP アドレスの関連付けの解除、または Elastic IP アドレスのリリースを行うことはできません。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeAddresses",
                "ec2:AllocateAddress",
                "ec2:DescribeInstances",
                "ec2:AssociateAddress"
            ],
            "Resource": "*"
        }
    ]
}
```

例: リザーブドインスタンス の操作

次のポリシーにより、アカウントのリザーブドインスタンスの表示と変更、および AWS マネジメントコンソール での新しいリザーブドインスタンスの購入をすることができます。

このポリシーにより、ユーザーがアカウント内のすべての リザーブドインスタンス と オンデマンドインスタンス を表示できるようになります。個別の リザーブドインスタンス にリソースレベルのアクセス許可を設定することはできません。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
         "ec2:DescribeReservedInstances",
         "ec2:ModifyReservedInstances",
         "ec2:PurchaseReservedInstancesOffering",
         "ec2:DescribeInstances",
         "ec2:DescribeInstanceTypes",
         "ec2:DescribeAvailabilityZones",
         "ec2:DescribeReservedInstancesOfferings"
      ],
      "Resource": "*"
   }
   ]
}
```

`ec2:DescribeAvailabilityZones` アクションは、リザーブドインスタンス を購入できるアベイラビリティーゾーンに関する情報を Amazon EC2 コンソールで表示できるようにするために必要です。`ec2:DescribeInstances` アクションは必須ではありませんが、このアクションにより、ユーザーがアカウントのインスタンスを表示し、正しい仕様に合わせて予約を購入できるようになります。

`ec2:DescribeInstances` を削除するなど、API アクションを調整してユーザーアクセスを制限できます。`ec2:DescribeAvailabilityZones` はユーザーが読み取り専用アクセスを持っていることを意味します。

Amazon EC2 の AWS マネージドポリシー

ユーザー、グループ、ロールにアクセス許可を追加するには自分でポリシーを作成するよりも、AWS マネージドポリシーを使用する方が簡単です。チームに必要な権限のみを提供する IAM カスタマーマネージドポリシーを作成するには時間と専門知識が必要です。すぐに使用を開始するた

めに、AWS マネージドポリシーを使用できます。これらのポリシーは一般的なユースケースを対象範囲に含めており、AWS アカウントで利用できます。AWS マネージドポリシーの詳細については「IAM ユーザーガイド」の「AWS マネージドポリシー」を参照してください。

AWS のサービスはAWS マネージドポリシーを維持および更新します。AWS マネージドポリシーの許可を変更することはできません。サービスでは新しい機能を利用できるようにするために、AWS マネージドポリシーに権限が追加されることがあります。この種類の更新はポリシーがアタッチされている、すべてのアイデンティティ (ユーザー、グループおよびロール) に影響を与えます。新しい機能が立ち上げられた場合や、新しいオペレーションが使用可能になった場合に、各サービスが AWS マネージドポリシーを更新する可能性が最も高くなります。サービスはAWS マネージドポリシーから権限を削除しないため、ポリシーの更新によって既存の権限が破棄されることはありません。

さらに、AWS は複数のサービスにまたがるジョブ機能の特徴に対するマネージドポリシーもサポートしています。例えば、ReadOnlyAccess AWS マネージドポリシーではすべての AWS のサービスおよびリソースへの読み取り専用アクセスを許可します。サービスが新しい機能を起動する場合、AWS は新たなオペレーションとリソース用に、読み取り専用の許可を追加します。ジョブ機能ポリシーの一覧と説明については「IAM ユーザーガイド」の「AWS ジョブ機能のマネージドポリシー」を参照してください。

AWS マネージドポリシー: AmazonEC2FullAccess

`AmazonEC2FullAccess` ポリシーは IAM アイデンティティにアタッチできます。このポリシーではAmazon EC2 への完全なアクセスを可能にする許可を付与します。

このポリシーのアクセス許可を確認するには、「AWS マネージドポリシーリファレンス」の「AmazonEC2FullAccess」を参照してください。

AWS マネージドポリシー: AmazonEC2ReadOnlyAccess

`AmazonEC2ReadOnlyAccess` ポリシーを IAM IDにアタッチできます。このポリシーではAmazon EC2 に対する読み取り専用アクセスを可能にする許可を付与します。

このポリシーのアクセス許可を確認するには、「AWS マネージドポリシーリファレンス」の「AmazonEC2ReadOnlyAccess」を参照してください。

AWS マネージドポリシー: AmazonEC2ImageReferencesAccessPolicy

`AmazonEC2ImageReferencesAccessPolicy` ポリシーを IAM IDにアタッチできます。このポリシーは、EC2 DescribeImageReferences API を使用するために必要なすべてのアクセス許可を付与します (EC2 インスタンス、起動テンプレート、Systems Manager パラメータ、Image Builder レシ

ピを表示するためのアクセス許可が含まれます)。このポリシーは IncludeAllResourceTypes フラグをサポートし、新しいリソースタイプのサポートが AWS で追加されると引き続き機能します。将来におけるポリシー更新は必要ありません。

このポリシーのアクセス許可を確認するには、「AWS マネージドポリシーリファレンス」の「AmazonEC2ImageReferencesAccessPolicy」を参照してください。

AWS マネージドポリシー: AWSEC2CapacityReservationFleetRolePolicy

このポリシーは AWSServiceRoleForEC2CapacityReservationFleet という名前のサービスにリンクされたロールにアタッチされ、サービスがユーザーの代わりにキャパシティ予約を作成、変更、およびキャンセルすることを、キャパシティ予約フリートに許可します。詳細については、「キャパシティ予約フリートでのサービスにリンクされたロールの使用」を参照してください。

このポリシーのアクセス許可を確認するには、「AWS マネージドポリシーリファレンス」の「AWSEC2CapacityReservationFleetRolePolicy」を参照してください。

AWS マネージドポリシー: AWSEC2FleetServiceRolePolicy

このポリシーはAWSServiceRoleForEC2Fleet という名前のサービスにリンクされたロールにアタッチされ、ユーザーの代わりにインスタンスのリクエスト、起動、終了、タグ付けを行うことを、EC2 フリートに許可します。詳細については「EC2 フリート用のサービスにリンクされたロール」を参照してください。

このポリシーのアクセス許可を確認するには、「AWS マネージドポリシーリファレンス」の「AWSEC2FleetServiceRolePolicy」を参照してください。

AWS マネージドポリシー: AWSEC2SpotFleetServiceRolePolicy

このポリシーはAWSServiceRoleForEC2SpotFleet という名前のサービスにリンクされたロールにアタッチされ、ユーザーの代わりにインスタンスの起動および管理を行うことをスポットフリートに許可します。詳細については「スポットフリート用のサービスにリンクされたロール」を参照してください。

このポリシーのアクセス許可を確認するには、「AWS マネージドポリシーリファレンス」の「AWSEC2SpotFleetServiceRolePolicy」を参照してください。

AWS マネージドポリシー: AWSEC2SpotServiceRolePolicy

このポリシーはAWSServiceRoleForEC2Spot という名前のサービスにリンクされたロールにアタッチされ、ユーザーの代わりにスポットインスタンス の起動および管理を行うことを、Amazon EC2

に許可します。詳細については「スポットインスタンスリクエスト向けのサービスにリンクされたロール」を参照してください。

このポリシーのアクセス許可を確認するには、「AWS マネージドポリシーリファレンス」の「AWSEC2SpotServiceRolePolicy」を参照してください。

AWS マネージドポリシー: AWSEC2VssSnapshotPolicy

この管理ポリシーは Amazon EC2 Windows インスタンスに使用する、IAM インスタンスプロファイルロールにアタッチすることができます。このポリシーはAmazon EC2 に、ユーザーに代わって VSS スナップショットを作成し管理することを許可するアクセス許可を付与します。

このポリシーのアクセス許可を確認するには、「AWS マネージドポリシーリファレンス」の「AWSEC2VssSnapshotPolicy」を参照してください。

AWS マネージドポリシー: DeclarativePoliciesEC2Report

このポリシーは、`AWSServiceRoleForDeclarativePoliciesEC2Report` という名前のサービスにリンクされたロールにアタッチされ、宣言ポリシーのアカウントステータスレポートを生成するために必要な読み取り専用 API へのアクセスを提供します。

このポリシーのアクセス許可を確認するには、「AWS マネージドポリシーリファレンス」の「DeclarativePoliciesEC2Report」を参照してください。

AWS マネージドポリシー: EC2FastLaunchFullAccess

`EC2FastLaunchFullAccess` ポリシーはインスタンスプロファイルまたはその他の IAM ロールにアタッチできます。このポリシーはEC2 Fast Launch アクションへのフルアクセスと、次のようにターゲットを絞ったアクセス許可を付与します。

アクセス許可の詳細

- EC2 Fast Launch – 管理アクセスが付与されて、対象ロールが EC2 Fast Launch を有効または無効にしたり、EC2 Fast Launch イメージを記述したりできるようになります。
- Amazon EC2 – Amazon EC2 RunInstances、CreateTags、Describe、Create Launch Template、Modify Launch Template の各オペレーションにアクセス権が付与されます。ネットワークとセキュリティグループのリソースを作成する、イングレスルールを承認する、および EC2 Fast Launch が作成したリソースを削除するためのアクセス権も付与されます。
- IAM – 名前に `ec2fastlaunch` が含まれるインスタンスプロファイルを取得および使用して、EC2FastLaunchServiceRolePolicy のサービスにリンクしたロールを作成するためのアクセス許可が付与されます。

- CloudFormation – EC2 Fast Launch が CloudFormation スタックを記述して作成し、作成したスタックを削除するためのアクセス権が付与されます。

このポリシーのアクセス許可を確認するには、「AWS マネージドポリシーリファレンス」の「EC2FastLaunchFullAccess」を参照してください。

AWS マネージドポリシー: AWSEC2CapacityManagerServiceRolePolicy

このポリシーは、EC2 Capacity Manager がキャパシティリソースを管理し、ユーザーに代わって AWS Organizations と統合できるように、AWSServiceRoleForEC2CapacityManager という名前のサービスリンクロールにアタッチされます。詳細については、「Service-linked roles for EC2 Capacity Manager」を参照してください。

このポリシーのアクセス許可を確認するには、「AWS マネージドポリシーリファレンス」の「AWSEC2CapacityManagerServiceRolePolicy」を参照してください。

AWS マネージドポリシー: EC2FastLaunchServiceRolePolicy

このポリシーはAWSServiceRoleForEC2FastLaunch という名前のサービスにリンクしたロールにアタッチされ、EC2 Fast Launch が有効になっている AMI からのインスタンスの起動にかかる時間を短縮する、事前プロビジョニングされたスナップショットのセットを作成および管理することを、Amazon EC2 に許可します。詳細については「the section called “サービスにリンクされたロール”」を参照してください。

このポリシーのアクセス許可を確認するには、「AWS マネージドポリシーリファレンス」の「EC2FastLaunchServiceRolePolicy」を参照してください。

AWS マネージドポリシー: Ec2InstanceConnect

`Ec2InstanceConnect` ポリシーを IAM アイデンティティにアタッチできます。このポリシーは、お客様が EC2 Instance Connect を呼び出して EC2 インスタンスにエフェメラルキーを公開し、ssh または EC2 Instance Connect CLI 経由で接続するアクセス許可を付与します。

このポリシーのアクセス許可を確認するには、「AWS マネージドポリシーリファレンス」の「Ec2InstanceConnect」を参照してください。

AWS マネージドポリシー: Ec2InstanceConnectEndpoint

このポリシーは AWSServiceRoleForEC2InstanceConnect というサービスリンクロールにアタッチされ、EC2 Instance Connect エンドポイントがユーザーに変わってアクションを実行できるよう

にします。詳細については「EC2 Instance Connect エンドポイントのサービスにリンクされたロール」を参照してください。

このポリシーのアクセス許可を確認するには、「AWS マネージドポリシーリファレンス」の「Ec2InstanceConnectEndpoint」を参照してください。このポリシーの更新の詳細については、「Amazon EC2 での AWS 管理ポリシーに関する更新」を参照してください。

Amazon EC2 での AWS 管理ポリシーに関する更新

Amazon EC2 の AWS 管理ポリシーに対する更新の詳細について、このサービスがこれらの変更の追跡を開始した以降のものを示します。

変更	説明	日付
AWSEC2CapacityManagerServiceRolePolicy - 新しいポリシー	Amazon EC2 は、ユーザーがキャパシティリソースを管理できるようにし、ユーザーに代わって AWS Organizations と統合するために、このポリシーを追加しました。	2025 年 10 月 15 日
AmazonEC2ImageReferencesAccessPolicy - 新しいポリシー	Amazon EC2 は、EC2 DescribeImageReferences API でサポートされているすべてのリソースタイプをスキャンするアクセス許可を提供するために、このポリシーを追加しました。	2025 年 8 月 26 日
Ec2InstanceConnectEndpoint - ポリシーを更新	既存の Instance Connect エンドポイントの変更をサポートするために、Amazon EC2 はこのポリシーを更新し、EC2 Instance Connect エンドポイントによって作成されたネットワークインターフェイスで IPv6 アドレスの割り当てと割り当て解除、およびセ	2025 年 7 月 31 日

変更	説明	日付
	キュリティグループの変更を行うためのアクセス許可を追加しました。また、Amazon EC2 はこのポリシーを更新して、Null 条件演算子を StringLike 条件演算子に置き換えました。	
EC2FastLaunchServiceRolePolicy - ポリシーを更新	孤立したリソースの発生を防ぐため、Amazon EC2 ではこのポリシーが更新され、ボリューム、ボリューム属性、ネットワークインターフェイスを記述し、EC2 Fast Launch によって作成されたボリュームおよびネットワークインターフェイスを削除するアクセス許可が追加されました。	2025 年 7 月 17 日
EC2FastLaunchFullAccess - ポリシーを更新	Amazon EC2 でこのポリシーが更新され、起動テンプレートの作成オペレーションと変更オペレーションが包含されました。これは、ネットワークとセキュリティグループのリソースを作成する、イングレスルールを承認する、および EC2 Fast Launch が作成したリソースを削除するためのものです。また、CloudFormation スタックを記述して作成し、EC2 Fast Launch が作成したスタックを削除することもできます。	2025 年 5 月 14 日

変更	説明	日付
EC2FastLaunchServiceRolePolicy - ポリシーを更新	Amazon EC2 は、このポリシーを更新して EC2 Fast Launch のイベントルールを作成して管理するための Amazon EventBridge アクセスを追加しました。また、EC2 Fast Launch は、CloudFormation スタックの記述、AWS License Manager に関連付けられている AMI からのインスタンスの起動に加えて、作成した AWS KMS グラントの中で廃止できるもののリストの取得や、作成した起動テンプレートの削除も実行できるようになりました。	2025 年 5 月 14 日
AWSEC2CapacityReservationFleetRolePolicy – アクセス許可の更新	Amazon EC2 では、 `ArnLike` 条件演算子の代わりに `StringLike` 条件演算子を使用するように `AWSEC2CapacityReservationFleetRolePolicy` 管理ポリシーを更新しました。	2025 年 3 月 3 日
AmazonEC2ReadOnlyAccess – アクセス許可を追加	Amazon EC2 は `GetSecurityGroupsForVpc` オペレーションを使用してセキュリティグループを取得できるアクセス許可を追加しました。	2024 年 12 月 27 日

変更	説明	日付
EC2FastLaunchFullAccess - 新しいポリシー	Amazon EC2 ではインスタンスから EC2 Fast Launch 機能に関連する API アクションを実行するために、このポリシーが追加されました。このポリシーはEC2 Fast Launch が有効になっている AMI から起動したインスタンスのインスタンスプロファイルにアタッチできます。	2024 年 5 月 14 日
AWSEC2VssSnapshotPolicy - 新しいポリシー	Amazon EC2 で、Amazon マシンイメージ (AMI) および EBS スナップショットにタグを作成および追加するアクセス許可を含む AWSEC2VssSnapshotPolicy ポリシーが追加されました。	2024 年 3 月 28 日
Ec2InstanceConnectEndpoint - 新しいポリシー	Amazon EC2 は Ec2InstanceConnectEndpoint ポリシーを追加しました。このポリシーは、AWSServiceRoleForEC2InstanceConnect サービスにリンクされたロールにアタッチされ、EC2 Instance Connect エンドポイントを作成するときに Amazon EC2 がユーザーに代わってアクションを実行できるようにします。	2023 年 1 月 24 日

変更	説明	日付
EC2FastLaunchServiceRolePolicy - 新しいポリシー	Amazon EC2 で、EC2 Fast Launch 機能が追加され、事前プロビジョニングされたスナップショットのセットを作成することにより、Windows AMI がインスタンスをより速く起動できるようになりました。	2021 年 11 月 26 日
Amazon EC2 が変更の追跡を開始しました。	Amazon EC2 が AWS 管理ポリシーの変更の追跡を開始しました	2021 年 3 月 1 日

Amazon EC2 の IAM ロール

アプリケーションは AWS 認証情報を使用して API リクエストに署名する必要があります。したがって、アプリケーションデベロッパーである場合、EC2 インスタンスで実行するアプリケーションの認証情報を管理する戦略が必要です。例えば、AWS 認証情報をインスタンスに安全に配布することで、他のユーザーから認証情報を保護しながら、それらのインスタンスのアプリケーションで認証情報を使用してリクエストに署名できます。ただし、各インスタンスに認証情報を安全に配布することは難しく、特に AWS がユーザーの代わりに作成するスポットインスタンスや Auto Scaling グループのインスタンスなどではそれが顕著です。また、AWS 認証情報を循環させる場合、各インスタンスの認証情報を更新できる必要もあります。

アプリケーションが使用するセキュリティ認証情報をお客様が管理する必要なく、アプリケーションがインスタンスから API リクエストを安全に作成できるように、IAM ロールをデザインしました。AWS 認証情報を作成および配布する代わりに、以下の方法で、IAM ロールを使用して API リクエストを作成するアクセス許可を委任できます。

1. IAM ロールを作成します。
2. ロールを行うアカウントまたは AWS のサービスを定義する
3. ロールを引き受けた後で、アプリケーションで使用できる API アクションおよびリソースを定義します。
4. インスタンスの起動時にロールを指定するか、既存のインスタンスにロールをアタッチします。

5. アプリケーションで一時的な認証情報のセットを取得して使用します。

例えば、IAM ロールを使用し、Amazon S3 のバケットを使用する必要のあるインスタンスで実行中のアプリケーションに、アクセス許可を与えることができます。JSON 形式のポリシーを作成することにより、IAM ロールのアクセス許可を指定できます。これらのポリシーは ユーザー用に作成するポリシーに類似しています。ロールを変更すると、その変更はすべてのインスタンスに反映されます。

> **Note**
>
> Amazon EC2 IAM ロールの認証情報はロールで設定された最大セッション期間の対象にはなりません。詳細については、「IAM ユーザーガイド」の「ロールを引き受けることができない」を参照してください。

IAM ロールを作成するとき、アプリケーションが必要とする特定の API コールへのアクセスを制限する最小特権の IAM ポリシーを関連付けます。Windows 間の通信では明確に定義されドキュメント化された Windows グループとロールを使用して、Windows インスタンス間のアプリケーションレベルのアクセスを許可します。グループとロールを使用すると、顧客は最小特権のアプリケーションと NTFS フォルダレベルのアクセス許可を定義して、アプリケーション固有の要件へのアクセスを制限できます。

インスタンスにアタッチできる IAM ロールは 1 つだけですが、同じロールを複数のインスタンスにアタッチできます。IAM ロールの作成と使用の詳細については、「IAM ユーザーガイド」の「ロール」を参照してください。

リソースレベルのアクセス許可を IAM ポリシーに適用して、インスタンスの IAM ロールのアタッチ、置換、またはデタッチをユーザーに許可するかどうを制御できます。詳細については、Amazon EC2 API アクションでサポートされるリソースレベルのアクセス許可と例: IAM ロールの使用の例を参照してください。

コンテンツ

- インスタンスプロファイル
- ユースケースに対するアクセス権限
- インスタンスメタデータからのセキュリティ認証情報の取得
- IAM ロールをインスタンスにアタッチするためのアクセス権限を付与する
- インスタンスへの IAM ロールのアタッチ

- Amazon EC2 インスタンスのインスタンスアイデンティティロール

インスタンスプロファイル

Amazon EC2 はIAM ロールのコンテナとしてインスタンスプロファイルを使用します。IAM コンソールを使用して IAM ロールを作成すると、コンソールによりインスタンスプロファイルが自動的に作成され、対応するロールと同じ名前が付けられます。Amazon EC2 コンソールを使用して IAM ロールを持つインスタンスを起動する場合、またはインスタンスに IAM ロールをアタッチする場合はインスタンスプロファイル名のリストに基づいてロールを選択してください。

AWS CLI、API、または AWS SDK を使用してロールを作成する場合、ロールとインスタンスプロファイルを別個のアクションとして作成します。名前は異なる可能性があります。次に AWS CLI、API、または AWS SDK を使用して IAM ロールを持つインスタンスを起動する場合、またはインスタンスに IAM ロールをアタッチする場合はインスタンスプロファイル名を指定します。

インスタンスプロファイルに含めることができる IAM ロールの数は 1 つのみです。IAM ロールを複数のインスタンス プロファイルに含めることができます。

インスタンスのアクセス許可を更新するには、インスタンスプロファイルを置き換えます。この変更が有効になるまでに最大 1 時間の遅延が発生するため、インスタンスプロファイルからロールを削除することはお勧めしません。

詳細については、「IAM ユーザーガイド」の「インスタンスプロファイルの使用」を参照してください。

ユースケースに対するアクセス権限

アプリケーションの IAM ロールを最初に作成するときに、必要な範囲を超えたアクセス権限を付与することがあります。本番環境でアプリケーションを起動する前に、IAM ロールのアクセスアクティビティに基づく IAM ポリシーを生成できます。IAM Access Analyzer は AWS CloudTrail ログを確認し、指定した日付範囲内のロールが使用したアクセス許可を含むポリシーテンプレートを生成します。テンプレートを使用して、きめ細かなアクセス権限で管理ポリシーを作成し、それを IAM ロールにアタッチできます。これにより、特定のユースケースでロールが AWS リソースとインタラクションするために必要なアクセス権限のみを付与します。これは最小特権の付与のベストプラクティスに準拠するのに役立ちます。詳細については、「IAM ユーザーガイド」の「IAM Access Analyzer ポリシーの生成」を参照してください。

インスタンスメタデータからのセキュリティ認証情報の取得

インスタンスのアプリケーションはインスタンスメタデータアイテム `iam/security-credentials/role-name` のロールから提供されたセキュリティ認証情報を取得します。アプリケーションにはロールに関連付けられたセキュリティ認証情報によって、ロールに対して定義したアクションおよびリソースのアクセス許可が付与されます。これらのセキュリティ認証情報は一時的なものであり、私たちが自動的に循環させます。新しい認証情報は古い認証情報が失効する少なくとも 5 分前から有効になるようにします。

インスタンスのメタデータの詳細については[インスタンスメタデータを使用して EC2 インスタンスを管理する]を参照してください。

> **Warning**
>
> IAM ロールでインスタンスメタデータを使用するサービスを使用する場合はサービスで HTTP 呼び出しが行われるときに認証情報を公開しないように注意する必要があります。認証情報を公開できるサービスの種類にはHTTP プロキシ、HTML/CSS 検証サービス、および XML インクルードをサポートする XML プロセッサーが含まれます。

Amazon EC2 ワークロードでは次に説明する方法を使用してセッション認証情報を取得することをお勧めします。これらの認証情報により、インスタンスに既に関連付けられている同じロールを引き受けるために `sts:AssumeRole` を使用する必要なしに、ワークロードが AWS API リクエストを実行できるようにすることができます。属性ベースのアクセス制御 (ABAC) のためにセッションタグを渡す必要がある場合や、ロールの許可をさらに制限するためにセッションポリシーを渡す必要がある場合でない限り、このようなロール引き受け呼び出しは不要です。これは同じ一時的なロールセッション認証情報の新しいセットを作成するためです。

ワークロードがロールを使用してそれ自体を引き受ける場合はその旨を明示的に許可する信頼ポリシーを作成する必要があります。信頼ポリシーを作成しない場合、`AccessDenied` エラーが発生します。詳細については、「IAM ユーザーガイド」の「[ロールの信頼ポリシーの変更]」を参照してください。

IMDSv2

Linux

次のコマンドを Linux インスタンスから実行し、IAM ロールのセキュリティ認証情報を取得します。

```
TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
    && curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/iam/security-credentials/role-name
```

Server

次のコマンドレットを Windows インスタンスから実行し、IAM ロールのセキュリティ認証情報を取得します。

```
[string]$token = Invoke-RestMethod `
    -Headers @{"X-aws-ec2-metadata-token-ttl-seconds" = "21600"} `
    -Method PUT -Uri http://169.254.169.254/latest/api/token
```

```
Invoke-RestMethod `
    -Headers @{"X-aws-ec2-metadata-token" = $token} `
    -Method GET -Uri http://169.254.169.254/latest/meta-data/iam/security-credentials/role-name
```

IMDSv1

Linux

次のコマンドを Linux インスタンスから実行し、IAM ロールのセキュリティ認証情報を取得します。

```
curl http://169.254.169.254/latest/meta-data/iam/security-credentials/role-name
```

Server

次のコマンドレットを Windows インスタンスから実行し、IAM ロールのセキュリティ認証情報を取得します。

```
Invoke-RestMethod -uri http://169.254.169.254/latest/meta-data/iam/security-credentials/role-name
```

以下は出力の例です。セキュリティ認証情報を取得できない場合は「IAM ユーザーガイド」の「EC2 インスタンスにある一時的なセキュリティ認証情報にアクセスできない」を参照してください。

```
{
```

```
  "Code" : "Success",
  "LastUpdated" : "2012-04-26T16:39:16Z",
  "Type" : "AWS-HMAC",
  "AccessKeyId" : "ASIAIOSFODNN7EXAMPLE",
  "SecretAccessKey" : "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
  "Token" : "token",
  "Expiration" : "2017-05-17T15:09:54Z"
}
```

インスタンスで実行されるアプリケーション、AWS CLI、および Tools for Windows PowerShell コマンドのために、一時的なセキュリティ認証情報を明示的に取得する必要はありません。AWS SDK、AWS CLI、および Tools for Windows PowerShell はEC2 インスタンスメタデータサービスから自動的に認証情報を取得し、使用します。一時的なセキュリティ認証情報を使用してインスタンスの外部で呼び出しを行う (IAM ポリシーをテストするなど) にはアクセスキー、秘密キー、およびセッショントークンを提供する必要があります。詳細については、「IAM ユーザーガイド」の「AWS リソースへのアクセスを要求するための一時的なセキュリティ認証情報の使用」を参照してください。

IAM ロールをインスタンスにアタッチするためのアクセス権限を付与する

AWS アカウントの ID (IAM ユーザーなど) にはIAM ロールを使用して Amazon EC2 インスタンスを起動する、IAM ロールをインスタンスにアタッチする、インスタンスの IAM ロールを置き換える、またはインスタンスから IAM ロールをデタッチするための特定のアクセス権限が必要です。必要に応じて、次の API アクションを使用するためのアクセス権限を付与する必要があります。

- `iam:PassRole`
- `ec2:AssociateIamInstanceProfile`
- `ec2:DisassociateIamInstanceProfile`
- `ec2:ReplaceIamInstanceProfileAssociation`

> **Note**
>
> `iam:PassRole` のリソースを `*` として指定すると、任意の IAM ロールをインスタンスに渡すためのアクセス権が付与されます。最小特権のベストプラクティスに従うには以下のポリシー例に示すように、`iam:PassRole` を使用して特定の IAM ロールの ARN を指定します。

プログラムによるアクセス用のポリシー例

次の IAM ポリシーでは AWS CLI または Amazon EC2 API を使用して、IAM ロールを使用してインスタンスを起動したり、IAM ロールをインスタンスにアタッチしたり、インスタンスの IAM ロールを置き換えたりするためのアクセス権限を付与します。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
         "ec2:RunInstances",
         "ec2:AssociateIamInstanceProfile",
         "ec2:DisassociateIamInstanceProfile",
         "ec2:ReplaceIamInstanceProfileAssociation"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::123456789012:role/DevTeam*"
    }
  ]
}
```

コンソールアクセスの追加要件

Amazon EC2 コンソールを使用して同じタスクを完了するためのアクセス権限を付与するには`iam:ListInstanceProfiles` API アクションも含める必要があります。

インスタンスへの IAM ロールのアタッチ

IAM ロールはインスタンスの起動時または起動後に作成してインスタンスにアタッチできます。IAM ロールを置き換えたりデタッチしたりすることもできます。

インスタンスの起動中に IAM ロールを作成してアタッチする (推奨)

1. EC2 インスタンスの起動中に [高度な詳細] を展開します。

2. [IAM インスタンスプロファイル] セクションで、[新しい IAM ロールを作成] を選択します。
3. 以下を実行できるインラインロールの作成フォームが開きます。

   - [ロール名] を指定する (例: `EC2-S3-Access-Role`)
   - AWS マネージドポリシーを選択するか、インスタンス用のカスタムポリシーを作成することで許可を定義する

     例えば、S3 へのアクセス権を付与するには `AmazonS3ReadOnlyAccess` マネージドポリシーを選択します。
   - `ec2.amazonaws.com` によるロールの引き受けを許可する信頼ポリシーを確認する
   - メタデータ用のオプションのタグを追加する
4. [ロールの作成] を選択してください。

   新しく作成されたロールが自動的に選択され、インスタンスの起動時にインスタンスプロファイル経由でインスタンスにアタッチされます。

> **Note**
>
> インスタンスの起動中にコンソールを使用してロールを作成するときは、ロールと同じ名前のインスタンスプロファイルが自動的に作成されます。インスタンスプロファイルは、起動時に IAM ロール情報をインスタンスに渡すコンテナです。

> **Important**
>
> - インスタンスにアタッチできる IAM ロールは 1 つだけですが、同じロールを複数のインスタンスにアタッチできます。
> - アプリケーションが必要とする特定の API コールへのアクセスを制限する最小特権の IAM ポリシーを関連付けてください。

IAM ロールの作成と使用の詳細については、「IAM ユーザーガイド」の「ロール」を参照してください。

インスタンスの起動中に既存の IAM ロールをアタッチする

Amazon EC2 コンソールを使用して起動時に IAM ロールをインスタンスにアタッチするには、[高度な詳細] を展開します。[IAM インスタンスプロファイル] で、ドロップダウンリストから IAM ロールを選択します。

> **Note**
>
> IAM コンソールを使用して IAM ロールを作成した場合はインスタンスプロファイルが自動的に作成され、ロールと同じ名前が付けられています。AWS CLI、API、または AWS SDK を使用して IAM ロールを作成した場合はインスタンスプロファイルにロールと異なる名前を付けた可能性があります。

実行中または停止しているインスタンスに IAM ロールをアタッチできます。インスタンスに既に IAM ロールがアタッチされている場合はそれを新しい IAM ロールに置き換える必要があります。

Console

IAM ロールをインスタンスにアタッチするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択してください。
4. [アクション] メニューで、[セキュリティ]、[IAM ロールの変更] の順に選択してください。
5. [IAM ロール] で IAM インスタンスプロファイルを選択してください。
6. [IAM ロールの更新] を選択してください。

AWS CLI

IAM ロールをインスタンスにアタッチするには

associate-iam-instance-profile コマンドを使用して、IAM ロールをインスタンスにアタッチします。インスタンスプロファイルを指定する際にはインスタンスプロファイルの Amazon リソースネーム (ARN) を使用することも、そのインスタンスプロファイルの名前を使用することもできます。

```
aws ec2 associate-iam-instance-profile \
    --instance-id i-1234567890abcdef0 \
```

```
    --iam-instance-profile Name="TestRole-1"
```

PowerShell

IAM ロールをインスタンスにアタッチするには

Register-EC2IamInstanceProfile コマンドレットを使用します。

```
Register-EC2IamInstanceProfile `
    -InstanceId i-1234567890abcdef0 `
    -IamInstanceProfile_Name TestRole-1
```

既に IAM ロールがアタッチされているインスタンスで IAM ロールを置き換えるには、インスタンスが実行されている必要があります。既存のロールをデタッチしないでインスタンスの IAM ロールを変更する場合に、これを行うことができます。例えば、インスタンスで実行しているアプリケーションが実行する API アクションが中断されないようにするために、これを行うことができます。

Console

インスタンスの IAM ロールを置き換えるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択してください。
4. [アクション] メニューで、[セキュリティ]、[IAM ロールの変更] の順に選択してください。
5. [IAM ロール] で IAM インスタンスプロファイルを選択してください。
6. [IAM ロールの更新] を選択してください。

AWS CLI

インスタンスの IAM ロールを置き換えるには

1. 必要に応じて、describe-iam-instance-profile-associations コマンドを使用して、関連付け ID を取得します。

```
aws ec2 describe-iam-instance-profile-associations \
    --filters Name=instance-id,Values=i-1234567890abcdef0 \
```

```
    --query IamInstanceProfileAssociations.AssociationId
```

2. replace-iam-instance-profile-association コマンドを使用します。既存のインスタンスプロファイルの関連付け ID と、新しいインスタンスプロファイルの ARN または名前を指定します。

```
aws ec2 replace-iam-instance-profile-association \
    --association-id iip-assoc-0044d817db6c0a4ba \
    --iam-instance-profile Name="TestRole-2"
```

PowerShell

インスタンスの IAM ロールを置き換えるには

1. 必要に応じて、Get-EC2IamInstanceProfileAssociation コマンドレットを使用して、関連付け ID を取得します。

```
(Get-EC2IamInstanceProfileAssociation -Filter @{Name="instance-id";
 Values="i-0636508011d8e966a"}).AssociationId
```

2. Set-EC2IamInstanceProfileAssociation コマンドレットを使用します。既存のインスタンスプロファイルの関連付け ID と、新しいインスタンスプロファイルの ARN または名前を指定します。

```
Set-EC2IamInstanceProfileAssociation `
    -AssociationId iip-assoc-0044d817db6c0a4ba `
    -IamInstanceProfile_Name TestRole-2
```

実行中または停止中のインスタンスから IAM ロールをデタッチできます。

Console

インスタンスから IAM ロールをデタッチするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択してください。
4. [アクション] メニューで、[セキュリティ]、[IAM ロールの変更] の順に選択してください。

5. [AM ロール] で、[IAM ロールがありません] を選択してください。
6. [IAM ロールの更新] を選択してください。
7. 確認を求められたら、「Detach」と入力してから、[デタッチ] を選択してください。

AWS CLI

インスタンスから IAM ロールをデタッチするには

1. 必要に応じて、describe-iam-instance-profile-associations を使用して、デタッチする IAM インスタンスプロファイルの関連付け ID を取得します。

```
aws ec2 describe-iam-instance-profile-associations \
    --filters Name=instance-id,Values=i-1234567890abcdef0 \
    --query IamInstanceProfileAssociations.AssociationId
```

2. disassociate-iam-instance-profile コマンドを使用します。

```
aws ec2 disassociate-iam-instance-profile --association-id iip-
assoc-0044d817db6c0a4ba
```

PowerShell

インスタンスから IAM ロールをデタッチするには

1. 必要に応じて、Get-EC2IamInstanceProfileAssociation を使用して、デタッチする IAM インスタンスプロファイルの関連付け ID を取得します。

```
(Get-EC2IamInstanceProfileAssociation -Filter @{Name="instance-id";
 Values="i-0636508011d8e966a"}).AssociationId
```

2. Unregister-EC2IamInstanceProfile コマンドレットを使用します。

```
Unregister-EC2IamInstanceProfile -AssociationId iip-assoc-0044d817db6c0a4ba
```

Amazon EC2 インスタンスのインスタンスアイデンティティロール

起動する各 Amazon EC2 インスタンスにはインスタンスアイデンティティを表すインスタンスアイデンティティロールがあります。インスタンスアイデンティティロールは IAM ロールの一種です。

インスタンス ID ロールを使用するように統合されている AWS サービスと機能はそのロールを使用してサービスのインスタンスを識別できます。

インスタンスアイデンティティロール認証情報は`/identity-credentials/ec2/security-credentials/ec2-instance` のインスタンスメタデータサービス (IMDS) からアクセスできます。認証情報はAWS 一時アクセスキー ID およびセッショントークンで構成されています。これらはインスタンス ID ロールを使用する AWS サービスへの AWS Sigv4 リクエストに署名するために使用されます。認証情報はインスタンスアイデンティティロールを使用するサービスまたは機能がインスタンスで有効になっているかどうかにかかわらず、インスタンスメタデータに存在します。

インスタンス ID ロールはインスタンスの起動時に自動的に作成され、ロール信頼ポリシー文書はなく、ID ポリシーやリソースポリシーの対象にもなりません。

サポートされるサービス

AWS サービスはインスタンス ID ロールを使用します。

- Amazon EC2 – EC2 Instance Connect はインスタンス ID ロールを使用して Linux インスタンスのホストキーを更新します。
- Amazon GuardDuty — ランタイムモニタリングはインスタンスアイデンティティロールを使用して、ランタイムエージェントが GuardDuty VPC エンドポイントにセキュリティテレメトリを送信できるようにします。
- AWS Lambda – Lambda マネージドインスタンスは、ライフサイクルフック、テレメトリ、アーティファクトディストリビューションにインスタンス ID ロールを使用します。
- AWS Security Token Service (AWS STS) - インスタンス ID ロールの認証情報は AWS STS `GetCallerIdentity` アクションで使用できます。
- AWS Systems Manager - デフォルトのホスト管理設定を使用する場合、AWS Systems Manager はインスタンスアイデンティティロールによって提供された ID を使用して EC2 インスタンスを登録します。インスタンスを識別すると、システムマネージャーは `AWSSystemsManagerDefaultEC2InstanceManagementRole` IAM ロールをインスタンスに渡すことができます。

インスタンス ID ロールはインスタンス ID ロールと統合されていないため、他の AWS サービスや機能では使用できません。

インスタンスアイデンティティロール ARN

インスタンスアイデンティティロール ARN は次の形式です。

```
arn:aws-partition:iam::account-number:assumed-role/aws:ec2-instance/instance-id
```

例:

```
arn:aws:iam::0123456789012:assumed-role/aws:ec2-instance/i-1234567890abcdef0
```

ARN の詳細については「IAM ユーザーガイド」の「Amazon リソースネーム (ARN)」を参照してください。

Amazon EC2 Windows インスタンスの更新管理

弊社はEC2 インスタンスのオペレーティングシステムやアプリケーションに対するパッチ適用やそれらの更新およびセキュリティ確保を定期的に行うよう推奨しています。AWS Systems Manager パッチマネージャーを使うと、オペレーティングシステムとアプリケーションの双方に関するセキュリティ関連更新のインストールプロセスを自動化できます。

Auto Scaling グループの EC2 インスタンスの場合、AWS-PatchAsgInstance ランブックを使用すると、パッチ適用中のインスタンスが置き換えられるのを防ぐことができます。代わりに、アプリケーションベンダーが提供している、自動更新サービスまたは推奨更新インストールプロセスを使用することもできます。

リソース

- AL2023 – 「Amazon Linux 2023 ユーザーガイド」の「AL2023 の更新」。
- AL2 – 「Amazon Linux 2 ユーザーガイド」の「Amazon Linux 2 インスタンスでのソフトウェアの管理」。
- Windows インスタンス – the section called “更新管理”。

Windows インスタンスにおけるセキュリティのベストプラクティス

Windows インスタンスにおける以下のセキュリティのベストプラクティスに従うことをお勧めします。

目次

- 高レベルのセキュリティのベストプラクティス
- 更新管理

- 設定管理
- 変更管理
- Amazon EC2 Windows インスタンスでの監査とアカウンタビリティ

高レベルのセキュリティのベストプラクティス

Windows インスタンスにおける以下の高レベルのセキュリティベストプラクティスに従う必要があります。

- 最小アクセス – 想定される信頼できるシステムと場所へのアクセスのみを許可します。これは Active Directory、Microsoft ビジネス生産性サーバーなどのすべての Microsoft 製品、およびリモートデスクトップサービス、リバースプロキシサーバー、IIS ウェブサーバーなどのインフラストラクチャサービスに適用されます。Amazon EC2 インスタンスセキュリティグループ、ネットワークアクセスコントロールリスト (ACL)、Amazon VPC パブリック/プライベートサブネットなど、AWS の機能を使用して、アーキテクチャ内の複数の場所でセキュリティを階層化します。Windows インスタンス内で、顧客は Windows Firewall を使用して、デプロイ内で多層防御戦略をさらに階層化できます。システムが設計どおりに機能するために必要な OS コンポーネントとアプリケーションのみをインストールします。インフラストラクチャ全体のローカルおよびリモートのリソースにアクセスするために、IIS などのインフラストラクチャサービスをサービスアカウントで動作するように設定するか、アプリケーションプール ID などの機能を使用するように設定します。
- 最小特権 – インスタンスとアカウントがそれらの機能を実行するのに必要な権限の最小セットを決定します。サーバーとユーザーにこれらの定義済みのアクセス許可のみが付与されるように制限します。ロールベースのアクセスコントロールなどの手法を使用して、管理アカウントのパブリック面を減らし、タスクを実行するための最も制限されたロールを作成します。NTFS 内の暗号化ファイルシステム (EFS) などの OS 機能を使用して、保管時の機密データを暗号化し、アプリケーションとユーザーのそのデータへのアクセスをコントロールします。
- 設定管理 – ウイルス対策、マルウェア対策、侵入検知/防止、ファイル整合性モニタリングなど、最新のセキュリティパッチとホストベースの保護スイートを組み込んだベースラインサーバー設定を作成します。記録されている最新のベースラインに対して各サーバーを評価し、逸脱を識別して、フラグを付けます。適切なログおよび監査データを生成して安全に保存するように各サーバーが設定されていることを確認します。
- 変更管理 – サーバー設定ベースラインに対する変更を制御するプロセスを作成し、完全に自動化された変更プロセスを目指します。また、Windows PowerShell DSC で Just Enough Administration (JEA) を活用して、管理アクセスを最小限必要な機能に制限します。

- パッチ管理 – EC2 インスタンス上のオペレーティングシステムやアプリケーションに対して定期的にパッチを適用し、更新し、セキュリティを確保するプロセスを実装します。
- 監査ログ – Amazon EC2 インスタンスに対するアクセスとすべての変更を監査して、サーバーの整合性を検証し、承認された変更のみが行われるようにします。IIS の拡張ログ記録などの機能を活用して、デフォルトのログ機能を強化します。VPC Flow Logs や AWS などの AWS CloudTrail 機能もネットワークアクセス (許可/拒否されたリクエストや API コールなど) の監査に利用できます。

更新管理

Amazon EC2 で Windows を実行した場合の最良の結果を得るには以下のベストプラクティスを実践することをお勧めします。

- Configure Windows Update
- Update drivers
- Use the latest Windows AMIs
- Test performance before migration
- Update launch agents
- 更新をインストールした後、Windows インスタンスを再起動します。詳細については「Amazon EC2 インスタンスを再起動する」を参照してください。

Windows インスタンスを Windows Server の新しいバージョンにアップグレードまたは移行する方法については「EC2 Windows インスタンスのより新しいバージョンの Windows Server へのアップグレード」を参照してください。

Windows Update の設定

デフォルトではAWS Windows Server AMI から起動されたインスタンスは Windows Update による更新が適用されません。

Windows ドライバーを更新する

すべての Windows EC2 インスタンスでドライバーを最新の状態に維持し、最新の問題修正とパフォーマンス強化がフリート全体に適用されるようにします。インスタンスタイプによってはAWS PV ドライバー、Amazon ENA ドライバー、AWS NVMe ドライバーを更新する必要があります。

- SNS トピックを使用して、ドライバーの新規リリースの最新情報を受け取ります。

- AWS Systems Manager 自動化ランブック AWSSupport-UpgradeWindowsAWSDrivers を使用して、インスタンス全体で更新を簡単に適用できます。

最新の Windows AMI を使用してインスタンスを起動する

AWS では最新の OS パッチ、ドライバー、起動エージェントを備えた新しい Windows AMI が毎月リリースされています。新しいインスタンスを起動する際、または独自のカスタムイメージを作成する際は最新の AMI を使用してください。

- AWS Windows AMI の各リリースに対する更新を表示するには「AWS Windows AMI バージョン履歴」を参照してください。
- 利用可能な最新の AMI を使用してビルドするには「Systems Manager パラメータストアを使用した最新の Windows AMI のクエリ」を参照してください。
- データベースのインスタンスを起動するのに使用できる特殊な Windows AMI とコンプライアンス強化のユースケースの詳細については「AWS Windows AMI リファレンス」の「専用の Windows AMI」を参照してください。

移行前にシステム/アプリケーションパフォーマンスをテストする

エンタープライズアプリケーションを AWSに移行するには多くの変更や設定が使用になる場合があります。EC2 ソリューションのパフォーマンステストを常に実施して、以下を確実にします。

- インスタンスサイズ、拡張ネットワーク、テナンシー (共有または専有) などのインスタンスタイプが適切に構成されている。
- インスタンストポロジーがワークロードに対し適切であり、必要に応じて高パフォーマンス機能 (専有テナンシー、プレイスメントグループ、インスタンスストアボリューム、ベアメタルなど) を活用します。

EC2Launch v2 の最新バージョンのインストール

最新の機能強化をフリート全体に適用されるように、最新の EC2Launch v2 エージェントにインストールします。詳細については「EC2Launch v2 をインストールする」を参照してください。

フリートが混在している場合、あるいは EC2Launch (Windows Server 2016 または 2019) エージェントまたは EC2 Config (レガシー OS のみ) エージェントを引き続き使用する場合は各エージェントの最新のバージョンに更新します。

自動更新はWindows Server バージョンと起動エージェントの次の組み合わせでサポートされます。[Amazon EC2 起動エージェント] の SSM Quick Setup ホスト管理コンソールで自動更新にオプトインできます。

Windows のバージョン	EC2Launch v1	EC2Launch v2
2016	✓	✓
2019	✓	✓
2022		✓

- EC2Launch v2 への更新の詳細については「the section called “EC2Launch v2 をインストールする”」を参照してください。
- EC2Config を手動で更新する方法については「the section called “EC2Config のインストール”」を参照してください。
- EC2Launch を手動で更新する方法については「the section called “EC2Launch のインストール”」を参照してください。

設定管理

Amazon マシンイメージ (AMI) はAmazon EC2 インスタンスの初期設定を提供します。これにはWindows OS とオプションの顧客固有のカスタマイズ (アプリケーションやセキュリティ管理など) が含まれます。カスタマイズされたセキュリティ設定ベースラインを含む AMI カタログを作成することで、すべての Windows インスタンスが標準のセキュリティコントロールを使用して起動されるようにします。セキュリティベースラインはAMI にベイクしたり、EC2 インスタンスの起動時に動的にブートストラップされるようにしたり、AWS Service Catalog ポートフォリオを介して一様に配布される製品としてパッケージ化したりすることができます。AMI のセキュリティ保護の詳細についてはAMI 構築のベストプラクティスを参照してください。

各 Amazon EC2 インスタンスは組織のセキュリティ標準に準拠する必要があります。不要なWindows のロールや機能をインストールしないでください。また、悪意のあるコードから保護するソフトウェア (ウイルス対策、マルウェア対策、エクスプロイト緩和)、ホストの整合性をモニタリングするソフトウェア、侵入検知を実行するソフトウェアをインストールしてください。OS のセキュリティ設定をモニタリングおよび維持し、重要な OS ファイルの整合性を保護し、セキュリティベースラインからの逸脱について警告するように、セキュリティソフトウェアを設定します。Microsoft

や Center for Internet Security (CIS)、米国国立標準技術研究所 (NIST) が公開している推奨セキュリティ設定ベンチマークの実装を検討してください。特定のアプリケーションサーバーに、他の Microsoft ツール (Best Practice Analyzer for SQL Server など) を使用することを検討してください。

AWS ユーザーは Amazon Inspector 評価を実行して、Amazon EC2 インスタンスにデプロイされたアプリケーションのセキュリティとコンプライアンスを向上させることもできます。Amazon Inspector はアプリケーションの脆弱性やベストプラクティスからの逸脱を自動的に評価します。また、一般的なセキュリティコンプライアンス標準 (PCI DSS など) と脆弱性定義にマッピングされた数百のルールのナレッジベースを含みます。組み込みルールの一例として、リモートルートログインが有効になっているかどうかや、脆弱なソフトウェアバージョンがインストールされているかどうかを確認するものがあります。これらのルールは AWS のセキュリティ調査担当者が定期的に更新しています。

Windows インスタンスを保護する場合はActive Directory ドメインサービスを実装して、分散した場所でスケーラブル、セキュア、管理可能なインフラストラクチャを有効にすることをお勧めします。さらに、Amazon EC2 コンソールまたはなどの Amazon EC2 プロビジョニングツールからインスタンスを起動したら、設定ドリフトが生じた場合に備えて、Microsoft Windows PowerShell DSCAWS CloudFormation などのネイティブ OS 機能を利用して設定状態を維持することをお勧めします。

変更管理

最初のセキュリティベースラインが起動時に Amazon EC2 インスタンスに適用された後、仮想マシンのセキュリティを維持するように、Amazon EC2 の進行中の変更を制御します。AWS リソース (セキュリティグループ、ルートテーブル、ネットワーク ACL など) に対する変更、OS およびアプリケーション設定 (Windows またはアプリケーションパッチ、ソフトウェアアップグレード、設定ファイルなど) に対する変更を承認して組み込むための、変更管理プロセスを確立します。

AWS にはAWS リソース (AWS CloudTrail、AWS Config、CloudFormation、AWS Elastic Beanstalk を含む) の他に、Systems Center Operations Manager および System Center Virtual Machine Manager の管理パックに対する変更を管理できるようにするいくつかのツールが用意されています。Microsoft から毎月第 2 火曜日に (または必要に応じて) Windows パッチがリリースされ、AWS はMicrosoft がパッチをリリースしてから 5 日以内に、AWS で管理されるすべての Windows AMI を更新します。したがって重要になるのはすべてのベースライン AMI に継続的にパッチを適用し、最新の AMI ID で CloudFormation テンプレートと Auto Scaling グループ設定を更新し、インスタンスのパッチ管理の実行を自動化するツールを実装することです。

Microsoft はWindows OS およびアプリケーションの変更を管理するためのいくつかのオプションを提供しています。例えば、SCCM はライフサイクル全体の環境の変更に対応しています。ビジネスの要件に応じて、変更がアプリケーション SLA、容量、セキュリティ、ディザスタリカバリ手順に

与える影響を制御するツールを選択してください。手動の変更を避け、代わりに自動設定管理ソフトウェアやコマンドラインツール (EC2 Run Command や Windows PowerShell など) を活用して、スクリプト化された繰り返し可能な変更プロセスを実装します。この要件を満たすのに役立つように、Windows インスタンスに対するすべての操作の拡張ログ記録が有効になっている踏み台ホストを使用して、すべてのイベントとタスクが自動的に記録されるようにします。

Amazon EC2 Windows インスタンスでの監査とアカウンタビリティ

AWS CloudTrail、AWS Config、AWS Config ルール にはAWS リソースの変更を監査するための監査および変更追跡機能が用意されています。ローカルログファイルを一元ログ管理システムに送信し、セキュリティおよび運用動作分析のためにログデータを保存するように、Windows イベントログを設定します。Microsoft System Center Operations Manager (SCOM) はWindows インスタンスにデプロイされた Microsoft アプリケーションに関する情報を集計し、事前設定されたカスタムルールセットをアプリケーションのロールとサービスに基づいて適用します。System Center Management Pack は SCOM に基づいて構築され、アプリケーション固有のモニタリングと設定のガイダンスを提供します。これらの Management Pack はWindows Server Active Directory、SharePoint Server 2013、Exchange Server 2013、Lync Server 2013、SQL Server 2014 など多くのサーバーとテクノロジーをサポートしています。

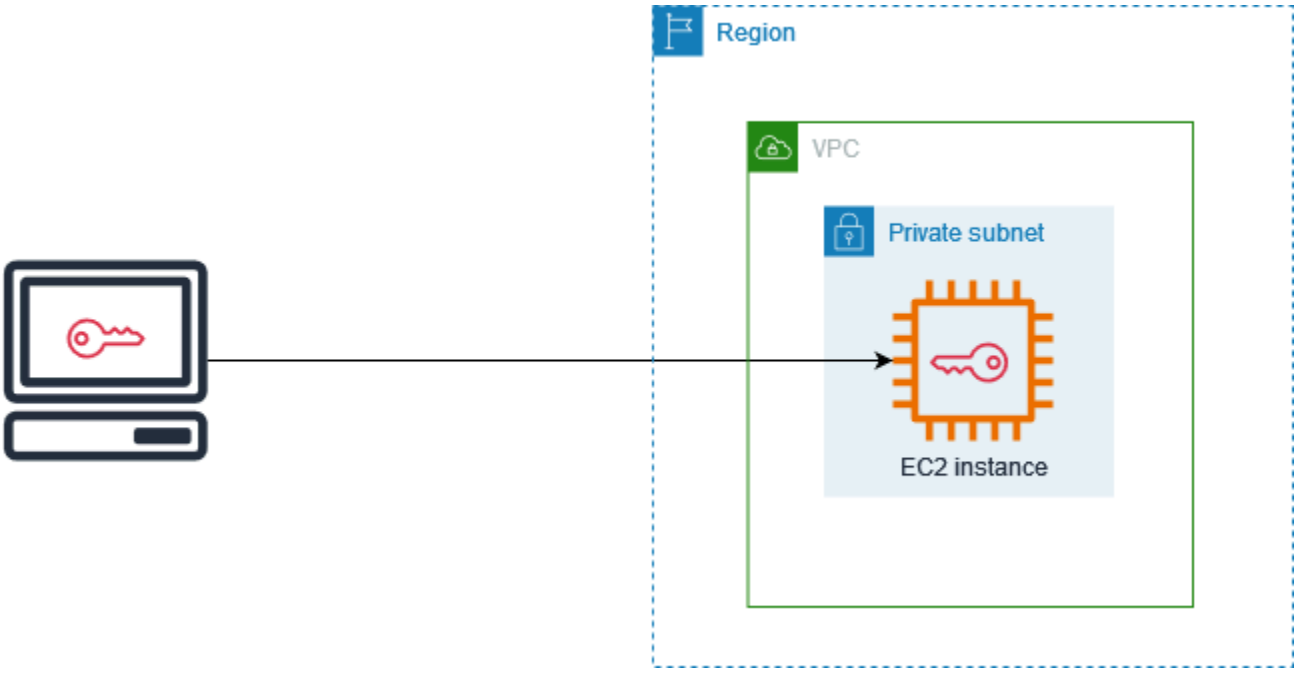
顧客はMicrosoft システム管理ツールに加えて、Amazon CloudWatch を使用して、インスタンスの CPU 使用率、ディスクパフォーマンス、ネットワーク I/O をモニタリングし、ホストおよびインスタンスのステータスチェックを実行できます。EC2Config、EC2Launch、および EC2Launch v2 エージェントではWindows インスタンスのその他の高度な機能にアクセスできます。例えば、Windows システム、セキュリティ、アプリケーション、インターネットインフォメーションサービス (IIS) のログを CloudWatch Logs にエクスポートし、Amazon CloudWatch メトリクスおよびアラームと統合できます。顧客は Windows パフォーマンスカウンターを Amazon CloudWatch カスタムメトリクスにエクスポートするスクリプトを作成することもできます。

Amazon EC2 のキーペアと Amazon EC2 インスタンス

キーペアにはプライベートキーと公開キーを含んでおり、 Amazon EC2 インスタンスへの接続時の身分証明に使用する、セキュリティ認証情報のセットを構成しています。Linux インスタンスの場合、プライベートキーを使用すると、インスタンスに安全に SSH 接続できます。Windows インスタンスの場合、管理者パスワードを復号化するにはプライベートキーが必要です。これを使用してインスタンスに接続します。

次の図に示すように、パブリックキーは Amazon EC2 によりお客様のインスタンス内に保管されます。またプライベートキーはお客様自身が保管します。プライベートキーを所有するすべてのユー

ザーはキーペアを使用するインスタンスに接続できるため、プライベートキーを安全な場所に保存することが重要です。



インスタンスを起動するときに、キーペアを指定することで、キーペアを必要とするメソッドを使用してインスタンスに接続できるようになります。セキュリティの管理方法に応じて、すべてのインスタンスに同じ key pair を指定することも、異なるキーペアを指定することもできます。

Linux インスタンスではインスタンスを初めて起動する際に、起動時に指定したパブリックキーが Linux インスタンス上で `~/.ssh/authorized_keys` 内のエントリに配置されます。SSH を使用して Linux インスタンスに接続する際のログインにはパブリックキーに対応するプライベートキーを指定する必要があります。

EC2 インスタンスへの接続の詳細については「EC2 インスタンスに接続する」を参照してください。

> **Important**
>
> Amazon EC2 ではプライベートキーのコピーが保持されないため、プライベートキーを失った場合、復元することはできません。ただし、プライベートキーを失くしたインスタンスに接続する方法はまだあります。詳細については「プライベートキーを紛失しました。自分のインスタンスに接続するにはどうすればよいですか?」を参照してください。

キーペアの代わりに、インタラクティブなワンクリックのブラウザベースのシェルまたは AWS Command Line Interface (AWS CLI) を使用してインスタンスに接続するために、AWS Systems Manager Session Manager を使用できます。

内容

- Amazon EC2 インスタンスのキーペアを作成する
- キーペアの詳細表示
- キーペアの削除
- Linux インスタンスでパブリックキーを追加または置き換える
- キーペアのフィンガープリントを確認する

Amazon EC2 インスタンスのキーペアを作成する

Amazon EC2 を使用してキーペアを作成できます。またはサードパーティー製のツールを使用して、キーペアを作成してから、Amazon EC2 にインポートすることもできます。

Amazon EC2 はLinux および Windows インスタンスで 2048-bit SSH-2 RSA キーをサポートしています。Amazon EC2 はLinux インスタンスでは ED25519 キーもサポートしています。

キーペアの作成後にインスタンスに接続する方法については「the section called “SSH を使用した Linux インスタンスへの接続”」と「the section called “RDP を使用した Windows インスタンスへの接続”」を参照してください。

内容

- Amazon EC2 を使用してキーペアを作成する
- AWS CloudFormation を使用してキーペアを作成する
- サードパーティー製のツールを使用してキーペアを作成し、Amazon EC2 にパブリックキーをインポートする

Amazon EC2 を使用してキーペアを作成する

Amazon EC2 を使用してキーペアを作成すると、パブリックキーは Amazon EC2 内に保存されます。プライベートキーは自分で保存します。

リージョンごとに最大 5,000 のキーペアを作成できます。増加をリクエストするにはサポートケースを作成します。詳細については、「サポート ユーザーガイド」の「サポートケースの作成」を参照してください。

Console

Amazon EC2 を使用してキーペアを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインの [Network & Security] で、[Key Pairs] を選択してください。
3. [キーペアの作成] を選択してください。
4. [Name (名前)] に、キーペアのわかりやすい名前を入力してください。Amazon EC2 はキー名として指定した名前にパブリックキーを関連付けます。キー名には最大 255 文字の ASCII 文字を含めることができます。先頭または末尾にスペースを含めることはできません。
5. 使用しているオペレーティングシステムに適したキーペアのタイプを選択してください。

   (Linux インスタンス) [キーペアのタイプ]で [RSA] または [ED25519] を選択してください。

   (Windows インスタンス) [キーペアのタイプ] で [RSA] を選択してください。Windows インスタンスでは ED25519 キーはサポートされていません。
6. [Private key ファイル形式] に、プライベートキーを保存する形式を選択してください。OpenSSH で使用できる形式でプライベートキーを保存するには[pem] を選択してください。プライベートキーを PuTTY で使用できる形式で保存するには[ppk] を選択してください。
7. タグを追加するには[タグの追加] ページで [タグの追加] をクリックし、タグのキーと値を入力してください。各タグについて、これを繰り返します。
8. [キーペアの作成] を選択してください。
9. ブラウザによって秘密キーファイルが自動的にダウンロードされます。ベースファイル名はキーペアの名前として指定した名前で、ファイル名拡張子は選択したファイル形式によって決まります。ダウンロードしたプライベートキーのファイルを安全な場所に保存します。

   > **Important**
   > プライベートキーのファイルを保存できるのはこのタイミングだけです。

10. macOS または Linux コンピュータの SSH クライアントを使用して Linux インスタンスに接続する予定がある場合は自分以外のユーザーが読み込むことができないように、次のコマンドを使用してプライベートキーファイルの許可を設定します。

```
chmod 400 key-pair-name.pem
```

これらのアクセス権限を設定しないと、このキーペアを使用してインスタンスに接続できません。詳細については「エラー: Unprotected Private キー ファイル (保護されていないプライベートキーファイル)」を参照してください。

AWS CLI

Amazon EC2 を使用してキーペアを作成するには

1. create-key-pair コマンドを使用し、次のようにキーペアを生成してプライベートキーを .pem ファイルに保存します。--query オプションはプライベートキーのマテリアルを出力に表示します。--output オプションは指定されたファイルにプライベートキーのマテリアルを保存します。拡張子は、キー形式に応じて .ppk または .pem のいずれかである必要があります。プライベートキーには公開キーの名前とは異なる名前を指定できますが、使いやすくするために、同じ名前を使用してください。

```
aws ec2 create-key-pair \
    --key-name my-key-pair \
    --key-type rsa \
    --key-format pem \
    --query "KeyMaterial" \
    --output text > my-key-pair.pem
```

2. macOS または Linux コンピュータの SSH クライアントを使用して Linux インスタンスに接続する予定がある場合は自分以外のユーザーが読み込むことができないように、次のコマンドを使用してプライベートキーファイルの許可を設定します。

```
chmod 400 key-pair-name.pem
```

これらのアクセス権限を設定しないと、このキーペアを使用してインスタンスに接続できません。詳細については「エラー: Unprotected Private キー ファイル (保護されていないプライベートキーファイル)」を参照してください。

PowerShell

Amazon EC2 を使用してキーペアを作成するには

以下の New-EC2KeyPair コマンドレットを使用してキーを生成し、.pem または .ppk ファイルに保存します。Out-File コマンドレットは、プライベートキーのマテリアルを拡張機能とともに

ファイルに保存します。拡張子は、キー形式に応じて .ppk または .pem のいずれかである必要があります。プライベートキーには公開キーの名前とは異なる名前を指定できますが、使いやすくするために、同じ名前を使用してください。

```
(New-EC2KeyPair `
    -KeyName "my-key-pair" `
    -KeyType "rsa" `
    -KeyFormat "pem").KeyMaterial | Out-File -Encoding ascii -FilePath C:\path\my-key-pair.pem
```

AWS CloudFormation を使用してキーペアを作成する

CloudFormation を使用して新しいキーペアを作成すると、プライベートキーは AWS Systems Manager パラメータストアに保存されます。パラメータ名の形式は次のとおりです。

```
/ec2/keypair/key_pair_id
```

詳細については「AWS Systems Manager ユーザーガイド」の「AWS Systems Manager パラメータストア」を参照してください。

CloudFormation を使用してキーペアを作成するには

1. テンプレートに AWS::EC2::KeyPair リソースを指定します。

```
Resources:
  NewKeyPair:
    Type: 'AWS::EC2::KeyPair'
    Properties:
      KeyName: new-key-pair
```

2. キーペアの ID を取得するには次のように describe-key-pairs コマンドを使用します。

```
aws ec2 describe-key-pairs --filters Name=key-name,Values=new-key-pair --query
 KeyPairs[*].KeyPairId --output text
```

以下は出力の例です。

```
key-05abb699beEXAMPLE
```

3. キーのパラメータを取得するために、次のように get-parameter コマンドを使用して、キーマテリアルを .pem ファイルに保存します。

```
aws ssm get-parameter --name /ec2/keypair/key-05abb699beEXAMPLE --with-decryption
 --query Parameter.Value --output text > new-key-pair.pem
```

必要な IAM 許可

CloudFormation がユーザーに代わって Parameter Store パラメータを管理できるようにするには CloudFormation またはユーザーにより引き受けられた IAM ロールは次の許可を持っている必要があります。

- `ssm:PutParameter` – プライベートキーマテリアル用パラメーターの削除を許可します。
- `ssm:DeleteParameter` - プライベートキーマテリアルを保存したパラメータの削除する許可を付与します。この権限はキーペアが CloudFormation によってインポートまたは作成されたかに関係なく必要です。

スタックによって作成またはインポートされたキーペアを CloudFormation が削除する場合、CloudFormation がキーペアをインポートする際ではなく、キーペアを作成する際にのみパラメーターを作成する場合でも権限チェックが実行され、パラメータを削除する権限があるかどうかが判断されます。CloudFormation はアカウント内のどのパラメータとも一致しない偽造パラメータ名を使用して必要なアクセス許可をテストします。そのため、`AccessDeniedException` エラーメッセージに偽造されたパラメータ名が表示されることがあります。

サードパーティー製のツールを使用してキーペアを作成し、Amazon EC2 にパブリックキーをインポートする

Amazon EC2 を使用してキーペアを作成する代わりに、サードパーティー製のツールで RSA または ED25519 のキーペアを作成してから、パブリックキーを Amazon EC2 にインポートすることもできます。

キーペアの要件

- サポートされている型:
  - (Linux と Windows) RSA
  - (Linux のみ) ED25519

> **Note**
>
> Windows インスタンスでは ED25519 キーはサポートされていません。

  - Amazon EC2 は DSA キーを受け付けません。
- サポートされる形式:
  - OpenSSH パブリックキー形式 (Linux の場合、`~/.ssh/authorized_keys` の形式)
  - (Linux のみ) EC2 Instance Connect API の使用中に SSH を使用して接続する場合はSSH2 形式もサポートされます。
  - SSH プライベートキーファイル形式は PEM または PPK である必要があります
  - (RSA のみ)Base64 でエンコードされた DER 形式
  - SSH パブリックキーファイル形式 [RFC4716] で指定
- サポートされている長さ:
  - 1024、2048、および 4096。
  - (Linux のみ) EC2 Instance Connect API の使用中に SSH を使用して接続する場合は長さ 2048 および 4096 がサポートされます。

サードパーティーツールを使用してキーペアを作成するには

1. 選択したサードパーティ製のツールでキーペアを生成します。例えば、ssh-keygen (標準 OpenSSH インストールで提供されるツール) を使用しできます。また、Java、Ruby、Python などのさまざまなプログラミング言語ではキーペアの作成に使用できる標準ライブラリが提供されています。

   > **Important**
   >
   > プライベートキーはPEM または PPK 形式である必要があります。例えば、`ssh-keygen -m PEM` を使用して OpenSSH キーを PEM 形式で生成します。

2. ローカルファイルにパブリックキーを保存します。例えば、`~/.ssh/my-key-pair.pub` (Linux、macOS) または `C:\keys\my-key-pair.pub` (Windows)。このファイル名の拡張子は重要ではありません。
3. `.pem` または `.ppk` 拡張子を持つローカルファイルにプライベートキーを保存します。例えば、`~/.ssh/my-key-pair.pem` または `~/.ssh/my-key-pair.ppk` (Linux、macOS)、ある

いは C:\keys\my-key-pair.pem または C:\keys\my-key-pair.ppk (Windows)。インスタンスへの接続に使用するツールによっては特定のファイル形式が必要になるため、ファイル拡張子は重要です。OpenSSH には .pem ファイルが必要であり、PuTTY には .ppk ファイルが必要です。

> **Important**
>
> プライベートキーファイルを安全な場所に保存します。インスタンスと対応するプライベートキーの起動時には毎回インスタンスに接続するたびに、パブリックキーの名前を入力する必要があります。

キーペアを作成したら、次のいずれかの方法を使用してパブリックキーを Amazon EC2 にインポートします。

Console

パブリックキーを Amazon EC2 にインポートするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[キーペア] を選択してください。
3. [Import Key Pair (キーペアのインポート)] を選択してください。
4. [Name (名前)] に、パブリックキーのわかりやすい名前を入力してください。名前には最大 255 文字の ASCII 文字を含めることができます。先頭または末尾にスペースを含めることはできません。

   > **Note**
   >
   > EC2 コンソールからインスタンスに接続すると、コンソールはプライベートキーファイルの名前としてこの名前が提示します。

5. [Browse (参照)] を選択してパブリックキーに移動して選択するか、パブリックキーのコンテンツを [Public key contents (パブリックキーのコンテンツ)] フィールドに貼り付けます。
6. [Import Key Pair (キーペアのインポート)] を選択してください。
7. インポートしたパブリックキーがキーペアのリストに表示されていることを確認します。

AWS CLI

パブリックキーを Amazon EC2 にインポートするには

import-key-pair コマンドを使用します。

```
aws ec2 import-key-pair \
    --key-name my-key-pair \
    --public-key-material fileb://path/my-key-pair.pub
```

キーペアが正常にインポートされたことを確認するには

describe-key-pairs コマンドを使用します。

```
aws ec2 describe-key-pairs --key-names my-key-pair
```

PowerShell

パブリックキーを Amazon EC2 にインポートするには

Import-EC2KeyPair コマンドレットを使用します。

```
$publickey=[Io.File]::ReadAllText("C:\Users\TestUser\.ssh\id_rsa.pub")
Import-EC2KeyPair `
    -KeyName my-key-pair `
    -PublicKey $publickey
```

キーペアが正常にインポートされたことを確認するには

Get-EC2KeyPair コマンドレットを使用します。

```
Get-EC2KeyPair -KeyName my-key-pair
```

キーペアの詳細表示

Amazon EC2 に保存されているキーペアの詳細情報を表示できます。また、パブリックキーマテリアルを取得し、起動時に指定されたパブリックキーを特定することもできます。

タスク

- キーペアの詳細表示

- パブリックキーマテリアルを取得する
- 起動時に指定されたパブリックキーを特定する

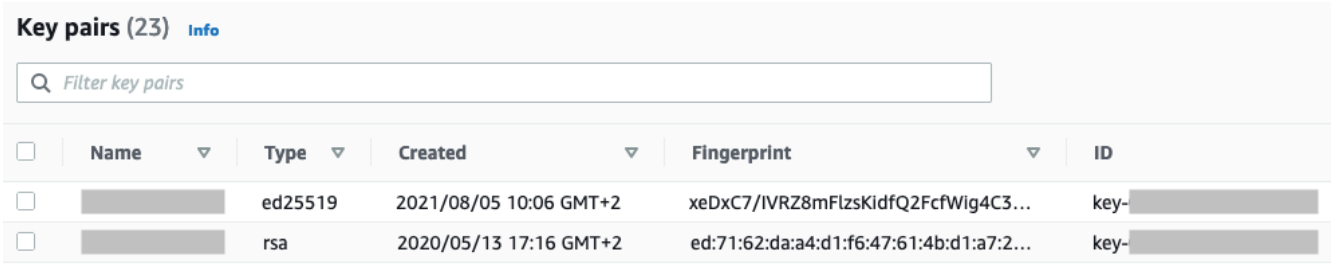
キーペアの詳細表示

Amazon EC2 に保存されているパブリックキーに関する次の情報を表示できます。パブリックキーの名前、ID、キーの種類、フィンガープリント、パブリックキーのマテリアル、Amazon EC2 によるキーの作成日時 (UTC タイムゾーン) (キーがサードパーティのツールで作成された場合はそのキーが Amazon EC2 にインポートされた日時)、およびパブリックキーに関連付けられているすべてのタグ。

Console

キーペアに関する情報を表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左側のナビゲータで、[Key Pairs] (キーペア) を選択してください。
3. [Key pairs] (キーペア) テーブルで各パブリックキーに関する情報を確認できます。



4. パブリックキーのタグを表示するにはキーの横にあるチェックボックスをオンにし、[アクション] および [タグの管理] を選択してください。

AWS CLI

キーペアに関する情報を表示するには

describe-key-pairs コマンドを使用します。

```
aws ec2 describe-key-pairs --key-names key-pair-name
```

PowerShell

キーペアに関する情報を表示するには

Get-EC2KeyPair コマンドレットを使用します。

```
Get-EC2KeyPair -KeyName key-pair-name
```

パブリックキーマテリアルを取得する

キーペアのパブリックキーマテリアルを取得できます。以下はパブリックキーの例です。こちらは、読みやすいように改行が追加されています。

```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQClKsfkNkuSevGj3eYhCe53pcjqP3maAhDFcvBS7O6V
hz2ItxCih+PnDSUaw+WNQn/mZphTk/a/gU8jEzoOWbkM4yxyb/wB96xbiFveSFJuOp/d6RJhJOI0iBXr
lsLnBItntckiJ7FbtxJMXLvvwJryDUilBMTjYtwB+QhYXUMOzce5Pjz5/i8SeJtjnV3iAoG/cQk+0FzZ
qaeJAAHco+CY/5WrUBkrHmFJr6HcXkvJdWPkYQS3xqC0+FmUZofz221CBt5IMucxXPkX4rWi+z7wB3Rb
BQoQzd8v7yeb7OzlPnWOyN0qFU0XA246RA8QFYiCNYwI3f05p6KLxEXAMPLE
```

Private key

ssh-keygen を使用してパブリックキーマテリアルを取得するには (Linux)

ローカルの Linux または macOS コンピュータで、ssh-keygen コマンドを使用します。プライベートキーをダウンロードしたパスを指定します (.pem ファイル)。

```
ssh-keygen -y -f /path_to_key_pair/my-key-pair.pem
```

この ssh-keygen コマンドが失敗した場合は、次の chmod コマンドを実行して、プライベートキーファイルに必要なアクセス許可があることを確認します。

```
chmod 400 key-pair-name.pem
```

PuTTYgen を使用してパブリックキーマテリアルを取得するには (Windows)

ローカルの Windows コンピュータで、PuTTYgen を起動します。[ロード] を選択します。プライベートキーファイル (.ppk または .pem) を選択してください。PuTTYgen の [Public key for pasting into OpenSSH authorized_keys ファイル] にパブリックキーが表示されます。パブリックキーは[パブリックキーの保存] を選択してファイルの名前を指定し、ファイルを保存後、そのファイルを開いて表示することもできます。

AWS CLI

パブリックキーマテリアルを取得するには

次の describe-key-pairs コマンドを使用し、--include-public-key オプションを指定します。

```
aws ec2 describe-key-pairs \
    --key-names key-pair-name \
    --include-public-key \
    --query "KeyPairs[].PublicKey"
```

PowerShell

パブリックキーマテリアルを取得するには

Get-EC2KeyPair コマンドレットを使用します。

```
(Get-EC2KeyPair -KeyName key-pair-name -IncludePublicKey $true).PublicKey
```

IMDSv2

Linux

次のコマンドを Linux インスタンスから実行します。

```
TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-
token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-
data/public-keys/0/openssh-key
```

Windows

次のコマンドレットを Windows インスタンスから実行します。

```
[string]$token = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token-ttl-seconds"
 = "21600"} -Method PUT -Uri http://169.254.169.254/latest/api/token
```

```
Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} -Method GET -Uri
 http://169.254.169.254/latest/meta-data/public-keys/0/openssh-key
```

IMDSv1

Linux

次のコマンドを Linux インスタンスから実行します。

```
curl http://169.254.169.254/latest/meta-data/public-keys/0/openssh-key
```

Windows

次のコマンドレットを Windows インスタンスから実行します。

```
Invoke-RestMethod -uri  http://169.254.169.254/latest/meta-data/public-keys/0/
openssh-key
```

起動時に指定されたパブリックキーを特定する

インスタンスを起動する際にパブリックキーを指定した場合、インスタンスによりパブリックキー名が記録されます。インスタンスのパブリックキーを変更したり、パブリックキーを追加したりしても、インスタンスで報告されたパブリックキー名は変更されません。

Console

インスタンスの起動時に指定されたパブリックキーを特定するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択します。
4. [詳細] タブの [インスタンスの詳細] で、[起動時に割り当てられたキーペア] を見つけます。

AWS CLI

インスタンスの起動時に指定されたパブリックキーを特定するには

次の describe-instances コマンドを使用します。

```
aws ec2 describe-instances \
    --instance-id i-1234567890abcdef0 \
    --query "Reservations[].Instances[].KeyName" \
    --output text
```

以下は出力の例です。

```
key-pair-name
```

PowerShell

インスタンスの起動時に指定されたパブリックキーを特定するには

Get-EC2Instance コマンドレットを使用します。

```
(Get-EC2Instance -InstanceId i-1234567890abcdef0).Instances | Select KeyName
```

以下は出力の例です。

```
KeyName
-------
key-pair-name
```

キーペアの削除

キーペアは削除できます。これにより、Amazon EC2 に保存されているパブリックキーは削除されます。キーペアを削除しても、対応するプライベートキーは削除されません。

以下の方法でパブリックキーを削除すると、キーペアの作成時またはインポート時に Amazon EC2 に保存されたパブリックキーのみが削除されます。パブリックキーをインスタンスに追加していた場合、パブリックキーを削除しても、インスタンスの起動時またはそれ以降に、インスタンスからパブリックキーが削除されることはありません。ローカルコンピュータのプライベートキーも削除されません。Amazon EC2 から削除したパブリックキーを使用してインスタンスを起動していた場合、プライベートキー (.pem) ファイルを保持している限り、引き続きインスタンスに接続できます。

> **Important**
>
> Auto Scaling グループ (Elastic Beanstalk 環境など) を使用している場合、関連する起動テンプレートまたは起動設定に削除するパブリックキーが指定されていないことを確認します。Amazon EC2 Auto Scaling が異常なインスタンスを検出した場合、代替インスタンスを起動します。ただし、パブリックキーが見つからない場合、インスタンスの起動は失敗します。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「起動テンプレート」を参照してください。

Console

Amazon EC2 上のパブリックキーを削除するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[キーペア] を選択してください。
3. 削除するキーペアを選択し、[Actions] (アクション)、[Delete] (削除) の順に選択してください。
4. 確認フィールドで、`Delete` を入力し、[Delete (削除)] を選択してください。

AWS CLI

Amazon EC2 上のパブリックキーを削除するには

delete-key-pair コマンドを使用します。

```
aws ec2 delete-key-pair --key-name my-key-pair
```

PowerShell

Amazon EC2 上のパブリックキーを削除するには

Remove-EC2KeyPair コマンドレットを使用します。

```
Remove-EC2KeyPair -KeyName my-key-pair
```

Linux インスタンスでパブリックキーを追加または置き換える

プライベートキーを紛失した場合、キーペアを使用するインスタンスへのアクセスができなくなります。起動時に指定したキーペアとは異なるキーペアを使用してインスタンスに接続する方法の詳細については「プライベートキーを紛失しました」を参照してください。

インスタンスを起動するとき、キーペアを指定できます。起動時にキーペアを指定した場合、インスタンスを初めて起動すると、パブリックキーマテリアルが Linux インスタンスの `~/.ssh/authorized_keys` 内のエントリに配置されます。SSH を使用して Linux インスタンスに最初に接

続するときは、デフォルトユーザーと、お使いの Linux インスタンスに保存されているパブリックキーに対応するプライベートキーを指定する必要があります。

インスタンスに接続した後、インスタンスのデフォルトシステムアカウントへのアクセスに使用されるキーペアを変更するにはインスタンスに新しいパブリックキーを追加するか、インスタンスでパブリックキー (既存のパブリックキーを削除して新しいパブリックキーを追加します) を置き換えます。インスタンスからすべてのパブリックキーを削除することもできます。

次の場合に、キーペアを追加するか、交換できます。

- 例えば、組織のユーザーが、別のキーペアを使用してシステムユーザーにアクセスする必要がある場合はパブリックキーをインスタンスに追加できます。
- プライベートキー (.pem ファイル) のコピーを持つユーザーがインスタンスに接続するのを防ぐには (例えば、組織を去った場合)、インスタンスのパブリックキーを削除し、新しいものに交換することができます。
- インスタンスから Linux AMI を作成すると、パブリックキーマテリアルがインスタンスから AMI にコピーされます。AMI からインスタンスを起動すると、新しいインスタンスは元のインスタンスからのパブリックキーを含みます。プライベートキーを持つユーザーが新しいインスタンスに接続できないようにするにはAMI を作成する前に、元のインスタンスからパブリックキーを削除します。

次の手順を使用して、デフォルトのユーザー (例: ec2-user) のキーペアを変更します。インスタンスにユーザーを追加する方法についてはインスタンスのオペレーティングシステムのドキュメントを参照してください。

キーペアの追加または交換

1. Amazon EC2 コンソールまたはサードパーティー製のツールで、新しいキーペアを作成します。
2. 新しいキーペアからパブリックキーを取得します。詳細については、「パブリックキーマテリアルを取得する」を参照してください。
3. インスタンスに接続します。
4. インスタンスで、任意のテキストエディタを使用して .ssh/authorized_keys ファイルを開きます。既存のパブリックキー情報の下の新しいキーペアからパブリックキー情報を貼り付け、ファイルを保存します。
5. インスタンスから切断します。新しいキーペアのプライベートキーファイルを使用してインスタンスに接続できることを確認します。

6. 自動スケーリング、EC2 フリート、起動テンプレートのいずれかを使用してインスタンスを起動する場合は、置き換えるキーペアが起動テンプレートまたは起動設定で指定されているかどうかを確認します。指定されていないとインスタンスの起動は失敗します。
7. (オプション) 既存のキーペアを交換している場合はインスタンスに接続し、`.ssh/authorized_keys` ファイルからオリジナルのキーペアのパブリックキー情報を削除します。

インスタンスからパブリックキーを削除するには

1. インスタンスに接続します。
2. 任意のテキストエディタを使用して、インスタンス上にある `.ssh/authorized_keys` ファイルを開きます。パブリックキーの情報を削除し、ファイルを保存します。

> **Warning**
>
> インスタンスからすべてのパブリックキーを削除してインスタンスを切断すると、別のログイン方法を設定していない限り、インスタンスに再度接続することはできません。

キーペアのフィンガープリントを確認する

キーペアのフィンガープリントを確認するにはAmazon EC2 コンソールの [キーペア] ページに表示される、または describe-key-pair コマンドによって返されるフィンガープリントを、ローカルコンピュータのプライベートキーを使用して生成したフィンガープリントと比較します。これらのフィンガープリントは一致するはずです。

Amazon EC2 がフィンガープリントを計算するとき、Amazon EC2 はフィンガープリントに = 文字でパディングを追加することがあります。ssh-keygen などのその他のツールではこのパディングを省略することがあります。

キーペアのフィンガープリントではなく Linux EC2 インスタンスのフィンガープリントを検証しようとしている場合は「インスタンスのフィンガープリントを取得する」を参照してください。

フィンガープリントの計算方法

Amazon EC2 はRSA および ED25519 キーペアのフィンガープリントを計算するために、さまざまなハッシュ関数を使用します。さらに、RSA キーペアの場合、Amazon EC2 はキーペアが Amazon EC2 によって作成されたか、Amazon EC2 にインポートされたかに応じて、異なるハッシュ関数を使用して異なる方法でフィンガープリントを計算します。

次の表はAmazon EC2 で作成され、Amazon EC2 にインポートされた RSA および ED25519 キーペアの、フィンガープリントの計算に使用されるハッシュ関数の一覧です。

(Linux インスタンス) フィンガープリントの計算に使用されるハッシュ関数

キーペアのソース	RSA キーペア (Windows および Linux)	ED25519 キーペア (Linux)
Amazon EC2 によって作成された	SHA-1	SHA-256
Amazon EC2 にインポートされた	MD5[1]	SHA-256

[1] パブリック RSA キーを Amazon EC2 にインポートした場合、フィンガープリントはMD5 ハッシュ関数を使用して計算されます。これはサードパーティーのツールを使用する、Amazon EC2 を使用して作成した既存のプライベートキーから新しいパブリックキーを生成するなど、キーペアの作成方法に関係なく当てはまります。

異なるリージョンで同じキーペアを使用する場合

同じキーペアを使用して、異なる AWS リージョン にあるインスタンスに接続する場合は使用するすべてのリージョンにパブリックキーをインポートする必要があります。Amazon EC2 を使用してキーペアを作成する場合、パブリックキーを他のリージョンにインポートできるよう、パブリックキーマテリアルを取得する することができます。

> **Note**
>
> - Amazon EC2 を使用して RSA キーペアを作成した場合、Amazon EC2 プライベートキーからパブリックキーを生成すると、インポートされたパブリックキーのフィンガープリントは元のパブリックキーとは異なるものになります。これはAmazon EC2 を使用して作成された元の RSA キーのフィンガープリントは SHA-1 ハッシュ関数を使用して計算されるのに対し、インポートされた RSA キーのフィンガープリントは MD5 ハッシュ関数を使用して計算されるためです。
> - ED25519 キーペアでは同じ SHA-256 ハッシュ関数を使用してフィンガープリントが計算されるため、Amazon EC2 で作成されたか、Amazon EC2 にインポートされたかにかかわらず、フィンガープリントは同一になります。

プライベートキーからフィンガープリントを生成する

ローカルマシンのプライベートキーからフィンガープリントを生成するには次のいずれかのコマンドを使用します。

Windows ローカルマシンを使用している場合、Windows Subsystem for Linux (WSL) を使用して、次のコマンドを実行できます。Windows 10 インストールガイドの手順を使用して、WSL と Linux ディストリビューションをインストールします。手順の例ではLinux の Ubuntu ディストリビューションをインストールしますが、任意のディストリビューションをインストールできます。コンピュータを再起動して変更を有効にすることが求められます。

- Amazon EC2 を使用してキーペアを作成した場合

  次の例のように、OpenSSL ツールを使用してフィンガープリントを生成します。

  RSA キーペアの場合:

  ```
  openssl pkcs8 -in path_to_private_key -inform PEM -outform DER -topk8 -nocrypt |
   openssl sha1 -c
  ```

  (Linux インスタンス) ED25519 キーペアの場合:

  ```
  ssh-keygen -l -f path_to_private_key
  ```

- (RSA キーペアのみ) パブリックキーを Amazon EC2 にインポートした場合

  キーペアの作成方法 (サードパーティーのツールを使用する、Amazon EC2 を使用して作成した既存のプライベートキーから新しいパブリックキーを生成するなど) に関係なく、この手順に従うことができます。

  次の例のように、OpenSSL ツールを使用してフィンガープリントを生成します。

  ```
  openssl rsa -in path_to_private_key -pubout -outform DER | openssl md5 -c
  ```

- OpenSSH 7.8 以降を使用して OpenSSH キーペアを作成し、パブリックキーを Amazon EC2 にインポートした場合

  次の例のように、ssh-keygen を使用してフィンガープリントを生成します。

  RSA キーペアの場合:

```
ssh-keygen -ef path_to_private_key -m PEM | openssl rsa -RSAPublicKey_in -outform DER
 | openssl md5 -c
```

(Linux インスタンス) ED25519 キーペアの場合:

```
ssh-keygen -l -f path_to_private_key
```

EC2 インスタンスの Amazon EC2 セキュリティグループ

セキュリティグループは、EC2 インスタンスの仮想ファイアウォールとして機能し、受信トラフィックと送信トラフィックを制御します。インバウンドルールはインスタンスへの受信トラフィックを制御し、アウトバウンドルールはインスタンスからの送信トラフィックをコントロールします。インスタンスの起動時に 1 つ以上のセキュリティグループを指定できます。セキュリティグループを指定しない場合、Amazon EC2 はデフォルトの VPC のセキュリティグループを使用します。インスタンスを起動した後、そのセキュリティグループを変更することができます。

セキュリティは、AWS とお客様の間の共有責任です。詳細については、セキュリティとコンプライアンスの目標を満たすように Amazon EC2 を設定し、Amazon EC2 リソースの保護に役立つ他の サービスの使用方法を学びます。を参照してください。AWS は、インスタンスをセキュリティで保護するためのツールの 1 つとしてセキュリティグループを提供しています。このセキュリティグループをセキュリティニーズに合わせて設定する必要があります。セキュリティグループでは十分に満たせない要件がある場合は、セキュリティグループの使用に加えて、どのインスタンスでも独自のファイアウォールを使用できます。

料金

セキュリティグループは追加料金なしで使用できます。

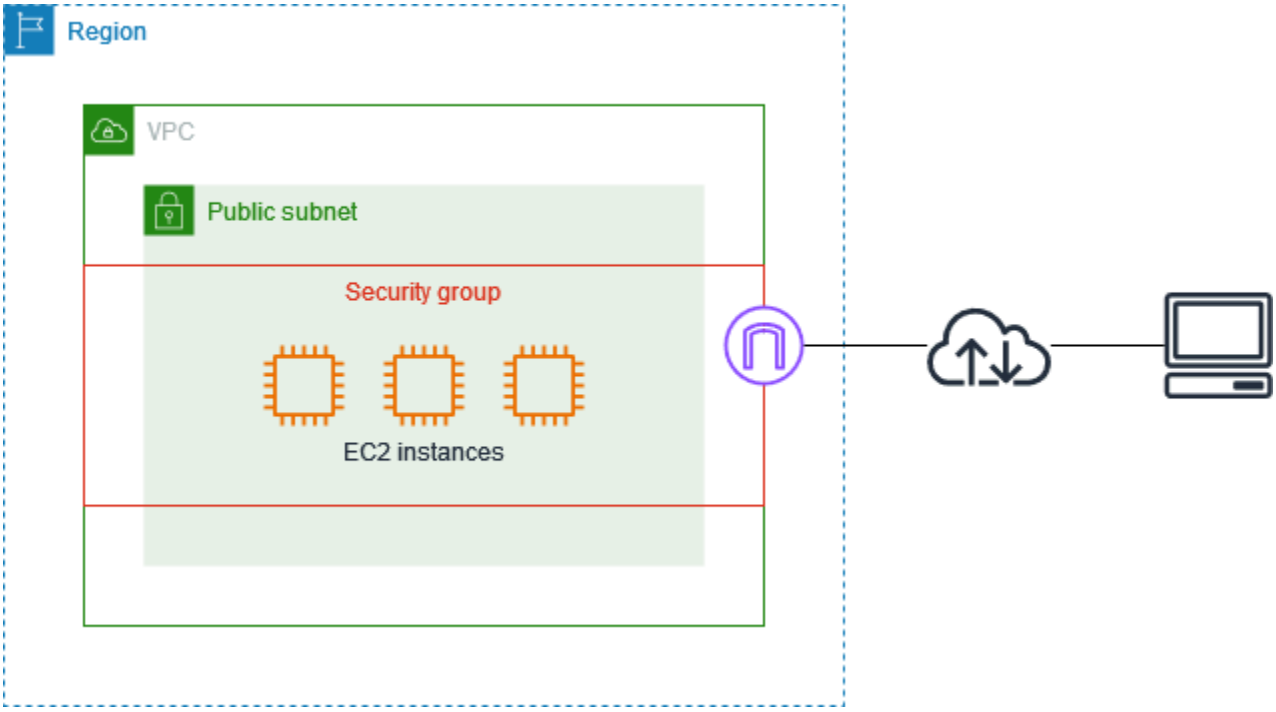
内容

- 概要
- Amazon EC2 インスタンス用のセキュリティグループの作成
- Amazon EC2 インスタンスのセキュリティグループの変更
- Amazon EC2 セキュリティグループの削除
- Amazon EC2 セキュリティグループの接続の追跡
- さまざまなユースケースのセキュリティグループのルール

概要

各インスタンスは複数のセキュリティグループに関連付けることができ、各セキュリティグループは複数のインスタンスに関連付けることができます。各セキュリティグループに対してルールを追加し、関連付けられたインスタンスに対するトラフィックを許可します。セキュリティグループのルールはいつでも変更することができます。新規または変更したルールは、セキュリティグループに関連付けられたすべてのインスタンスに自動的に適用されます。トラフィックがインスタンスに到達することを許可するかどうかを Amazon EC2 が判断するとき、インスタンスに関連付けられているすべてのセキュリティグループのすべてのルールを評価します。詳細については、「Amazon VPC ユーザーガイド」の「セキュリティグループルール」を参照してください。

次の図は、サブネット、インターネットゲートウェイ、セキュリティグループを備えた VPC を示しています。サブネットには EC2 インスタンスが含まれています。セキュリティグループはインスタンスに関連付けられています。インスタンスに到達するトラフィックは、セキュリティグループのルールで許可されているトラフィックだけです。例えば、使用しているネットワークからの SSH トラフィックを許可するルールがセキュリティグループに含まれている場合は、使用しているコンピュータからインスタンスに SSH で接続できます。関連付けられたリソースからのすべてのトラフィックを許可するルールがセキュリティグループに含まれている場合、各インスタンスは他のインスタンスから送信されたすべてのトラフィックを受信できます。



セキュリティグループはステートフルです。インスタンスからリクエストを送信すると、そのリクエストに対するレスポンストラフィックは、セキュリティグループのインバウンドルールにかかわら

ず、流入できます。また、許可されたインバウンドトラフィックに対する応答 (戻りのトラフィック) は、アウトバウンドルールにかかわらずアウト側に対し通過することができます。詳細については「接続追跡」を参照してください。

Amazon EC2 インスタンス用のセキュリティグループの作成

セキュリティグループは、関連付けられたインスタンスのファイアウォールとして動作し、インバウンドトラフィックとアウトバウンドトラフィックの両方をインスタンスレベルでコントロールします。SSH (Linux インスタンス) または RDP (Windows インスタンス) を使用してインスタンスに接続できるようにするルールをセキュリティグループに追加できます。また、ウェブサーバー宛ての HTTP および HTTPS トラフィックなど、クライアントトラフィックを許可するルールを追加することもできます。

インスタンスを起動する際に、インスタンスにセキュリティグループを関連付けることができます。関連付けられたセキュリティグループのルールを追加または削除すると、それらの変更は、そのセキュリティグループを関連付けたすべてのインスタンスに自動的に適用されます。

インスタンスを起動した後、追加のセキュリティグループを関連付けることができます。詳細については「Amazon EC2 インスタンスのセキュリティグループの変更」を参照してください。

インバウンドおよびアウトバウンドのセキュリティグループルールは、セキュリティグループの作成時に追加することも、後で追加することもできます。詳細については「セキュリティグループルールの設定」を参照してください。セキュリティグループに追加できるルールの例については、「さまざまなユースケースのセキュリティグループのルール」を参照してください。

考慮事項

- 新しいセキュリティグループには、すべてのトラフィックがリソースを離れることを許可するアウトバウンドルールのみで開始されます。任意のインバウンドトラフィックを許可するには、またはアウトバウンドトラフィックを制限するには、ルールを追加する必要があります。
- 送信元でインスタンスに対する SSH または RDP アクセスを許可するルールを設定する場合、任意の場所からのアクセスは許可しないでください。これを許可すると、インターネット上のすべての IP アドレスからのインスタンスに対するこのアクセスを許可することになります。この状態は、テスト環境での短時間の使用であれば許容できますが、実稼働環境においては安全ではありません。
- 特定のポートに複数のルールがある場合、Amazon EC2 が最も許容度の大きいルールを適用します。例えば、IP アドレス 203.0.113.1 からの TCP ポート 22 (SSH) に対するアクセスを許可するルールと、任意の場所からの TCP ポート 22 に対するアクセスを許可する別のルールがある場合、全員が TCP ポート 22 にアクセスできます。

- 1 つのインスタンスには複数のセキュリティグループを関連付けることができます。そのため、1 つのインスタンスに数百のルールが適用される場合があります。結果として、インスタンスにアクセスするときに問題が発生する可能性があります。そのため、ルールは可能な限り要約することをお勧めします。
- ルールに送信元または送信先としてセキュリティグループを指定する場合、ルールはセキュリティグループに関連付けられているすべてのインスタンスに影響します。着信トラフィックは、ソースセキュリティグループに関連付けられたインスタンスのプライベート IP アドレスに基づいて許可されます (パブリック IP アドレスまたは Elastic IP アドレスは考慮されません)。IP アドレスについては、Amazon EC2 インスタンスの IP アドレス指定を参照してください。
- デフォルトで、Amazon EC2 はポート 25 のトラフィックをブロックします。詳細については「ポート 25 を使用した E メール送信の制限」を参照してください。

Console

セキュリティグループを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Security Groups] を選択してください。
3. [セキュリティグループの作成] を選択してください。
4. セキュリティグループの分かりやすい名前と簡単な説明を入力します。セキュリティグループの作成後に名前と説明を変更することはできません。
5. [VPC] で、Amazon EC2 インスタンスを実行する VPC を選択します。
6. (オプション) インバウンドルールを追加するには、[インバウンドルール] を選択します。ルールごとに、[ルールの追加] を選択し、プロトコル、ポート、および送信元を指定します。例えば、SSH トラフィックを許可するには、[タイプ] に [SSH] を選択し、[送信元] にコンピュータまたはネットワークのパブリック IPv4 アドレスを指定します。
7. (オプション) アウトバウンドルールを追加するには、[アウトバウンドルール] を選択します。ルールごとに、[ルールの追加] を選択し、プロトコル、ポート、および送信先を指定します。追加しない場合は、デフォルトのルールをそのまま使用でき、すべてのアウトバウンドトラフィックを許可します。
8. (オプション) タグを追加するには、[Add new tag] (新しいタグを追加) を選択し、そのタグのキーと値を入力します。
9. [セキュリティグループの作成] を選択します。

AWS CLI

セキュリティグループを作成するには

次の create-security-group コマンドを使用します。

```
aws ec2 create-security-group \
    --group-name my-security-group \
    --description "my security group" \
    --vpc-id vpc-1234567890abcdef0
```

ルールを追加する例については、「the section called “セキュリティグループルールの設定”」を参照してください。

PowerShell

セキュリティグループを作成するには

New-EC2SecurityGroup コマンドレットを使用します。

```
New-EC2SecurityGroup `
    -GroupName my-security-group `
    -Description "my security group" `
    -VpcId vpc-1234567890abcdef0
```

ルールを追加する例については、「the section called “セキュリティグループルールの設定”」を参照してください。

Amazon EC2 インスタンスのセキュリティグループの変更

Amazon EC2 インスタンスのセキュリティグループは、インスタンスの起動時に指定できます。インスタンスを起動した後、セキュリティグループを追加または削除することができます。また、関連付けられたセキュリティグループのセキュリティグループルールは、いつでも追加、削除、編集できます。

セキュリティグループはネットワークインターフェイスに関連付けられます。セキュリティグループを追加または削除すると、プライマリネットワークインターフェイスに関連付けられているセキュリティグループが変更されます。セカンダリネットワークインターフェイスに関連付けられているセキュリティグループも変更できます。詳細については「ネットワークインターフェイス属性の変更」を参照してください。

タスク

- セキュリティグループの追加または削除
- セキュリティグループルールの設定

セキュリティグループの追加または削除

インスタンスを起動した後、関連付けられているセキュリティグループのリストに対し、セキュリティグループを追加または削除できます。複数のセキュリティグループをインスタンスに関連付けると、各セキュリティグループのルールが効率的に集約され、1 つのルールセットが作成されます。Amazon EC2 はこのルールセットを使用して、トラフィックを許可するかを判断します。

要件

- インスタンスは、`running` または `stopped` 状態である必要があります。

Console

インスタンスのセキュリティグループを変更するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[アクション]、[セキュリティ]、[セキュリティグループの変更] の順に選択します。
4. [関連付けられたセキュリティグループ] で、リストからセキュリティグループを選択し、[セキュリティグループを追加] を選択します。

   すでに関連付けられているセキュリティグループを削除するには、そのセキュリティグループで [削除] を選択します。
5. [保存] を選択します。

AWS CLI

インスタンスのセキュリティグループを変更するには

次の modify-instance-attribute コマンドを使用します。

```
aws ec2 modify-instance-attribute \
```

```
    --instance-id i-1234567890abcdef0 \
    --groups sg-1234567890abcdef0
```

PowerShell

インスタンスのセキュリティグループを変更するには

Edit-EC2InstanceAttribute コマンドレットを使用します。

```
Edit-EC2InstanceAttribute `
    -InstanceId i-1234567890abcdef0 `
    -Group sg-1234567890abcdef0
```

セキュリティグループルールの設定

セキュリティグループを作成したら、そのセキュリティグループルールを追加、更新、削除できます。ルールを追加、更新、または削除すると、その変更は、セキュリティグループに関連付けられているリソースに自動的に適用されます。

セキュリティグループに追加できるルールの例については、さまざまなユースケースのセキュリティグループのルールを参照してください。

必要なアクセス許可

作業を開始する前に、必要なアクセス許可があることを確認してください。詳細については「例: セキュリティグループの操作」を参照してください。

プロトコルとポート

- コンソールでは、事前定義されたタイプを選択すると、プロトコルとポート範囲が指定されます。ポート範囲を入力するには、[カスタム TCP] または [カスタム UDP] のいずれかのカスタムタイプを選択する必要があります。
- AWS CLI では、`--protocol` オプションおよび `--port` オプションを使用して、単一のポートを持つ単一のルールを追加できます。複数のルール、またはポート範囲を持つルールを追加するには、代わりに `--ip-permissions` オプションを使用します。

送信元と送信先

- コンソールでは、インバウンドルールの送信元またはアウトバウンドルールの送信先として、以下を指定できます。

- [カスタム] – IPv4 CIDR ブロック、IPv6 CIDR ブロック、セキュリティグループ、またはプレフィックスリスト。
- [Anywhere-IPv4] – 0.0.0.0/0 IPv4 CIDR ブロック。
- [Anywhere-IPv6] – ::/0 IPv6 CIDR ブロック。
- [マイ IP] – ローカルコンピュータのパブリック IPv4 アドレス。

- AWS CLI では、`--cidr` オプションを使用して IPv4 CIDR ブロックを指定するか、`--source-group` オプションを使用してセキュリティグループを指定できます。プレフィックスリストまたは IPv6 CIDR ブロックを指定するには、`--ip-permissions` オプションを使用します。

> **Warning**
>
> ポート 22 (SSH) または 3389 (RDP) のインバウンドルールを追加する場合は、特定の IP アドレス、またはインスタンスへのアクセスを必要とするアドレスの範囲のみを許可することを強くお勧めします。[Anywhere-IPv4] を選択すると、すべての IPv4 アドレスからのトラフィックが、指定されたプロトコルを使用してインスタンスにアクセスできるようになります。[Anywhere-IPv6] を選択すると、すべての IPv6 アドレスからのトラフィックが、指定されたプロトコルを使用してインスタンスにアクセスできるようになります。

Console

セキュリティグループのルールを設定するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[Security Groups] を選択します。
3. セキュリティグループを選択します。
4. インバウンドルールを編集するには、[アクション] または [インバウンドルール] タブから [インバウンドルールを編集] を選択します。

   a. ルールを追加するには、[ルールを追加] を選択し、ルールのタイプ、プロトコル、ポート、および送信元を入力します。

      タイプが TCP または UDP の場合は、許可するポート範囲を入力する必要があります。カスタムの ICMP の場合は、[プロトコル] から ICMP タイプ名を選択し、該当するものがある場合は [ポート範囲] からコード名を選択します。その他のタイプについては、プロトコルとポート範囲は自動的に設定されます。

b. ルールを更新するには、必要に応じてプロトコル、説明、送信元を変更します。ただし、送信元のタイプを変更することはできません。例えば、送信元が IPv4 CIDR ブロックの場合、IPv6 CIDR ブロック、プレフィックスリスト、またはセキュリティグループを指定することはできません。

c. ルールを削除するには、[削除] ボタンを選択します。

5. アウトバウンドルールを編集するには、[アクション] または [アウトバウンドルール] タブから [アウトバウンドルールを編集] を選択します。

a. ルールを追加するには、[ルールを追加] を選択し、ルールのタイプ、プロトコル、ポート、および送信先を入力します。オプションとして説明を入力することもできます。

タイプが TCP または UDP の場合は、許可するポート範囲を入力する必要があります。カスタムの ICMP の場合は、[プロトコル] から ICMP タイプ名を選択し、該当するものがある場合は [ポート範囲] からコード名を選択します。その他のタイプについては、プロトコルとポート範囲は自動的に設定されます。

b. ルールを更新するには、必要に応じてプロトコル、説明、送信元を変更します。ただし、送信元のタイプを変更することはできません。例えば、送信元が IPv4 CIDR ブロックの場合、IPv6 CIDR ブロック、プレフィックスリスト、またはセキュリティグループを指定することはできません。

c. ルールを削除するには、[削除] ボタンを選択します。

6. [ルールの保存] を選択します。

AWS CLI

セキュリティグループルールを追加するには

authorize-security-group-ingress コマンドを使用して、インバウンドルールを追加します。次の例では、指定されたプレフィックスリストの CIDR ブロックからの、インバウンド SSH トラフィックを許可します。

```
aws ec2 authorize-security-group-ingress \
    --group-id sg-1234567890abcdef0 \
    --ip-permissions
 'IpProtocol=tcp,FromPort=22,ToPort=22,PrefixListIds=[{PrefixListId=pl-
f8a6439156EXAMPLE}]'
```

authorize-security-group-egress コマンドを使用して、アウトバウンドルールを追加します。次の例では、指定されたセキュリティグループを持つインスタンスへの、ポート 80 でのアウトバウンド TCP トラフィックを許可します。

```
aws ec2 authorize-security-group-egress \
    --group-id sg-1234567890abcdef0 \
    --ip-permissions
 'IpProtocol=tcp,FromPort=80,ToPort=80,UserIdGroupPairs=[{GroupId=sg-0aad1c26bb6EXAMPLE}]'
```

セキュリティグループルールを削除するには

インバウンドルールを削除するには、次の revoke-security-group-ingress コマンドを使用します。

```
aws ec2 revoke-security-group-egress \
    --group id sg-1234567890abcdef0 \
    --security-group-rule-ids sgr-09ed298024EXAMPLE
```

アウトバウンドルールを削除するには、次の revoke-security-group-egress コマンドを使用します。

```
aws ec2 revoke-security-group-ingress \
    --group id sg-1234567890abcdef0 \
    --security-group-rule-ids sgr-0352250c1aEXAMPLE
```

セキュリティグループのルールを変更するには

modify-security-group-rules コマンドを使用します。次の例では、指定されたセキュリティグループルールの IPv4 CIDR ブロックを変更します。

```
aws ec2 modify-security-group-rules \
    --group id sg-1234567890abcdef0 \
    --security-group-rules
 'SecurityGroupRuleId=sgr-09ed298024EXAMPLE,SecurityGroupRule={IpProtocol=tcp,FromPort=80,To
```

PowerShell

セキュリティグループルールを追加するには

インバウンドルールを追加するには、Grant-EC2SecurityGroupIngress コマンドレットを使用します。次の例では、指定されたプレフィックスリストの CIDR ブロックからの、インバウンド SSH トラフィックを許可します。

```
$plid = New-Object -TypeName Amazon.EC2.Model.PrefixListId
$plid.Id = "pl-f8a6439156EXAMPLE"
Grant-EC2SecurityGroupIngress `
    -GroupId sg-1234567890abcdef0 `
    -IpPermission @{IpProtocol="tcp"; FromPort=22; ToPort=22; PrefixListIds=$plid}
```

アウトバウンドルールを追加するには、Grant-EC2SecurityGroupEgress コマンドレットを使用します。次の例では、指定されたセキュリティグループを持つインスタンスへの、ポート 80 でのアウトバウンド TCP トラフィックを許可します。

```
$uigp = New-Object -TypeName Amazon.EC2.Model.UserIdGroupPair
$uigp.GroupId = "sg-0aad1c26bb6EXAMPLE"
Grant-EC2SecurityGroupEgress `
    -GroupId sg-1234567890abcdef0 `
    -IpPermission @{IpProtocol="tcp"; FromPort=80; ToPort=80; UserIdGroupPairs=
$uigp}
```

セキュリティグループルールを削除するには

インバウンドルールを削除するには、Revoke-EC2SecurityGroupIngress コマンドレットを使用します。

```
Revoke-EC2SecurityGroupIngress `
    -GroupId sg-1234567890abcdef0 `
    -SecurityGroupRuleId sgr-09ed298024EXAMPLE
```

アウトバウンドルールを削除するには、Revoke-EC2SecurityGroupEgress コマンドレットを使用します。

```
Revoke-EC2SecurityGroupEgress `
    -GroupId sg-1234567890abcdef0 `
    -SecurityGroupRuleId sgr-0352250c1aEXAMPLE
```

セキュリティグループのルールを変更するには

Edit-EC2SecurityGroupRule コマンドレットを使用します。次の例では、指定されたセキュリティグループルールの IPv4 CIDR ブロックを変更します。

```
$sgrr = New-Object -TypeName Amazon.EC2.Model.SecurityGroupRuleRequest
$sgrr.IpProtocol = "tcp"
$sgrr.FromPort = 80
$sgrr.ToPort = 80
$sgrr.CidrIpv4 = "0.0.0.0/0"
$sgr = New-Object -TypeName Amazon.EC2.Model.SecurityGroupRuleUpdate
$sgr.SecurityGroupRuleId = "sgr-09ed298024EXAMPLE"
$sgr.SecurityGroupRule = $sgrr
Edit-EC2SecurityGroupRule  `
    -GroupId sg-1234567890abcdef0 `
    -SecurityGroupRule $sgr
```

Amazon EC2 セキュリティグループの削除

Amazon EC2 インスタンスで使用するために作成したセキュリティグループの使用が済んだら、削除できます。

要件

- セキュリティグループをインスタンスまたはネットワークインターフェイスに関連付けることはできません。
- 別のセキュリティグループのルールでセキュリティグループを参照することはできません。

Console

セキュリティグループを削除するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. (オプション) セキュリティグループがインスタンスに関連付けられていないことを確認するには、以下の操作を実行します。
   a. ナビゲーションペインで、[Security Groups] を選択してください。
   b. 削除するセキュリティグループの ID をコピーします。
   c. ナビゲーションペインで、[インスタンス] を選択してください。
   d. 検索バーで、[セキュリティグループ ID が次と等しい] フィルターを追加し、セキュリティグループの ID を貼り付けます。結果がない場合、セキュリティグループはインスタンスに関連付けられていません。それ以外の場合、セキュリティグループを削除するには、セキュリティグループの関連付けを解除する必要があります。

3. ナビゲーションペインで、[Security Groups] を選択してください。
4. セキュリティグループを選択して、[アクション]、[セキュリティグループを削除] を選択します。
5. 複数のセキュリティグループを選択した場合は、確認を求められます。一部のセキュリティグループを削除できない場合は、削除されるかどうかを示す、各セキュリティグループのステータスが表示されます。削除を確認するには、「Delete」と入力します。
6. [削除] を選択します。

AWS CLI

セキュリティグループを削除するには

次の delete-security-group コマンドを使用します。

```
aws ec2 delete-security-group --group-id sg-1234567890abcdef0
```

PowerShell

セキュリティグループを削除するには

Remove-EC2SecurityGroup コマンドレットを使用します。

```
Remove-EC2SecurityGroup -GroupId sg-1234567890abcdef0
```

Amazon EC2 セキュリティグループの接続の追跡

セキュリティグループは、接続追跡を使用してインスタンスを出入りするトラフィックに関する情報を追跡します。ルールはトラフィックの接続の状態に基づいて適用され、トラフィックを許可するか拒否するかが判断されます。このアプローチでは、セキュリティグループはステートフルです。これは、セキュリティグループのアウトバウンドルールにかかわらず、インバウンドトラフィックに対するレスポンスがインスタンスから送信されることを許可することを意味します。逆も同じです。

例えば、自宅のコンピュータからインスタンスに対し netcat や同様の ICMP コマンドを開始する場合を考えます。この時、インバウンドセキュリティグループは、ICMP トラフィックを許可しているとします。接続に関する情報 (ポート情報を含む) が追跡されます。コマンドに対するインスタンスからのレスポンストラフィックは、新しいリクエストではなく確立済みの接続として追跡されます。また、セキュリティグループのアウトバウンドルールが、アウトバウンドの ICMP トラフィックを制限している場合でも、このトラフィックはインスタンスから外部に出力されることが許されます。

TCP、UDP、または ICMP 以外のプロトコルの場合は、IP アドレスとプロトコル番号のみが追跡されます。インスタンスが別のホストにトラフィックを送信し、そのホストが 600 秒以内に同じタイプのトラフィックをインスタンスに送信した場合、インスタンスのセキュリティグループはインバウンドセキュリティグループルールに関係なく、そのトラフィックを受け入れます。そのトラフィックが元のトラフィックのレスポンストラフィックとみなされるからです。

セキュリティグループルールを変更しても、そのルールで追跡された接続がすぐに中断されることはありません。セキュリティグループは、既存の接続がタイムアウトするまで引き続きパケットを許可します。トラフィックをすぐに中断するか、追跡状態に関係なくすべてのトラフィックをファイアウォールルールの対象にするには、サブネットにネットワーク ACL を使用します。ネットワーク ACL はステートレスであるため、レスポンスのトラフィックを自動的には許可しません。いずれかの方向のトラフィックをブロックするネットワーク ACL を追加すると、既存の接続が切断されます。詳細については、「Amazon VPC ユーザーガイド」の「ネットワーク ACL」を参照してください。

> **Note**
>
> セキュリティグループは、「VPC +2 IP アドレス」(「Amazon Route 53 デベロッパーガイド」の「Amazon Route 53 Resolver とは」を参照)、または「AmazonProvidedDNS」(「Amazon Virtual Private Cloud ユーザーガイド」の「DHCP オプションセットの使用」を参照) と呼ばれることもある Route 53 Resolver から送受信される DNS トラフィックに影響を及ぼすことはありません。Route 53 Resolver で DNS リクエストをフィルタリングしたい場合は、Route 53 Resolver DNS ファイアウォールを有効にできます (「Amazon Route 53 デベロッパーガイド」の「Route 53 Resolver DNS ファイアウォール」を参照)。

追跡されていない接続

すべてのトラフィックフローが追跡されるわけではありません。セキュリティグループのルールがすべてのトラフィック (0.0.0.0/0 または ::/0) の TCP または UDP フローを許可していて、片方の方向には任意のポート (0～65535) のすべての応答トラフィック (0.0.0.0/0 または ::/0) を許可するルールがある場合、そのトラフィックフローは自動的に追跡される接続の一部でない限り追跡されません。追跡されていないフローのレスポンストラフィックは、追跡情報ではなく、レスポンストラフィックを許可するインバウンドルールまたはアウトバウンドルールに基づいて許可されます。

追跡されていないトラフィックフローは、そのフローを有効にするルールが削除または変更されるとすぐに中断されます。例えば、オープン (0.0.0.0/0) のアウトバウンドルールがあり、インスタンスへのすべて(0.0.0.0/0) のインバウンドの SSH (TCP ポート 22) トラフィックを許可するルールを

削除した場合 (または接続を許可しないように変更した場合)、インスタンスへの既存の SSH 接続はすぐに中断されます。接続はそれまで追跡されていないため、この変更によって接続が切断されます。一方、最初に細かく SSH 接続を許可する (つまり、接続を追跡する) インバウンドルールがある場合、現在の SSH クライアントのアドレスからの新しい接続を許可しないようにルールを変更しても、既存の SSH 接続は追跡対象であるため中断されません。

自動的に追跡される接続

セキュリティ グループの構成で追跡が必要ない場合でも、次の方法で行われた接続は自動的に追跡されます。

- Egress-Only インターネットゲートウェイ
- Global Accelerator アクセラレーター
- NAT ゲートウェイ
- Network Firewall ファイアウォールのエンドポイント
- Network Load Balancer
- AWS PrivateLink (インターフェイス VPC エンドポイント)
- AWS Lambda (Hyperplane Elastic Network Interface)
- DynamoDB のゲートウェイエンドポイント – DynamoDB への各接続は 2 つの conntrack エントリを消費します。

追跡できる接続の最大数

Amazon EC2 では、インスタンスごとに追跡できる接続の最大数が定義されています。追跡が最大数に達すると、新しい接続が確立されることはないため、送受信されるパケットはすべてドロップされます。この場合、パケットを送受信するアプリケーションは正しく通信できません。`conntrack_allowance_available` ネットワークパフォーマンスメトリクスを使用して、そのインスタンスタイプでまだ利用可能な接続トラッキングの数を判断します。

インスタンスのネットワークトラフィックが追跡可能な接続の最大数を超えたために、パケットがドロップされたかどうかを判断するには、ネットワークパフォーマンスメトリクス `conntrack_allowance_exceeded` を参照します。詳細については「EC2 インスタンスでの ENA 設定のネットワークパフォーマンスのモニタリング」を参照してください。

Elastic Load Balancing を実行している際にインスタンスごとに追跡できる接続の最大数を超える場合は、ロードバランサーに登録されているインスタンスの数、あるいは登録されているインスタンスのサイズのいずれかをスケールすることをお勧めします。

接続追跡のベストプラクティス

トラフィックが特定のネットワークインターフェースからインスタンスに入り、別のネットワークインターフェースから外に出る、非対称ルーティングでは、フローを追跡した場合に、インスタンスが達成できるピークパフォーマンスが低下する可能性があります。

セキュリティグループで接続追跡が有効になっている場合にピークパフォーマンスを維持して接続管理を最適化するには、次の設定をお勧めします。

- 可能であれば、非対称ルーティングトポロジーは避けてください。
- フィルタリングにセキュリティグループを使用する代わりに、ネットワーク ACL を使用します。
- 接続追跡でセキュリティグループを使用する必要がある場合は、可能な限り短い接続追跡のアイドル状態タイムアウトを設定します。接続追跡のアイドル状態タイムアウトの詳細については、次のセクションを参照してください。
- Nitrov6 インスタンスではデフォルトのタイムアウト時間が短いため、存続期間の長い接続 (データベース接続プール、永続的な HTTP 接続、ストリーミングワークロードなど) を持つアプリケーションは、インスタンス起動時に適切な `TcpEstablishedTimeout` 値を設定する必要があります。
- 存続期間の長い接続の場合は、接続を開いたままにして追跡状態を維持するために TCP キープアライブを 5 分未満の間隔で送信するように設定します。これにより、アイドルタイムアウトによる接続のドロップを防ぎ、接続の再確立のオーバーヘッドを削減できます。

Nitro システムによるパフォーマンスチューニングの詳細については、Nitro System のパフォーマンスチューニングに関する考慮事項を参照してください。

アイドル接続追跡タイムアウト

セキュリティグループは、確立された各接続を追跡し、リターンパケットが期待どおりに配信されることを保証します。インスタンスごとに追跡できる接続の最大数があります。接続がアイドル状態のままになると、接続追跡が使い果たされることで接続が追跡されなくなり、また、パケットがドロップされる原因となります。Elastic Network Interface では、アイドル接続の追跡にタイムアウトを設定できます。

> **Note**
>
> この機能は、Nitro ベースのインスタンスでのみ使用できます。Nitrov6 世代インスタンスでアプリケーションをテストするには、本番環境にデプロイする前に、350 2 番目のデフォルトの接続追跡タイムアウトを短くする必要があります。

設定可能なタイムアウトは 3 つ用意されています。

- TCP 確立タイムアウト: 確立された状態のアイドル TCP 接続のタイムアウト (秒単位)。
  - 最小: 60 秒
  - 最大: 432000 秒
  - デフォルト: Nitro v6 インスタンスタイプ (P6e-GB200 を除く) の場合は 350 秒、その他のすべてのインスタンスタイプ (P6e-GB200 を含む) の場合は 432000 秒。
  - 推奨: 432000 秒未満
- UDP タイムアウト: 単一方向、または 1 つのリクエスト-レスポンストランザクションのみのトラフィックが発生した、アイドル状態にある UDP フローのタイムアウト (秒単位)。
  - 最小: 30 秒
  - 最大: 60 秒
  - デフォルト: 30 秒
- UDP ストリームタイムアウト: 複数のリクエスト-レスポンストランザクションが発生したストリームとして分類される、アイドル状態にある UDP フローのタイムアウト (秒単位)。
  - 最小: 60 秒
  - 最大: 180 秒
  - デフォルト: 180 秒

以下のいずれかに当てはまる場合は、デフォルトのタイムアウトの変更が必要になる場合があります。

- Amazon EC2 のネットワークパフォーマンスメトリクスを使用して追跡接続をモニタリングしている場合は、conntrack_allowance_exceeded および conntrack_allowance_available メトリクスにより、ドロップされたパケットと追跡された接続使用率をモニタリングできるようになります。これにより、スケールアップまたはスケールアウトアクションにより EC2 インスタンスの容量を事前に管理し、パケットのドロップが発生する前にネットワーク接続の需要を満たすことができま

す。EC2 インスタンスで conntrack_allowance_exceeded のドロップが観測された場合は、不適切なクライアントやネットワークのミドルボックスが原因で TCP/UDP セッションの使用期間が長くなりすぎることを考慮して、TCP 確立のタイムアウトを低く設定すると、メリットが得られる場合があります。

- 通常、ロードバランサーまたはファイアウォールの TCP Established アイドルタイムアウトは、60 分～90 分の範囲に設定されています。ネットワークファイアウォールなどのアプライアンスからの、非常に多くの (10万件を超える) 接続を処理することが予想されるワークロードを実行している場合は、EC2 ネットワークインターフェイスでも同様のタイムアウトを設定することをお勧めします。
- 非対称ルーティングトポロジを使用するワークロードを実行している場合は、TCP 確立アイドルタイムアウトを 60 秒に設定することをお勧めします。
- 主に UDP を使用してリクエストを処理するサービス (例えば DNS、SIP、SNMP、Syslog、Radius) など、接続数が多いワークロードを実行している場合、「UDP ストリーム」のタイムアウトを 60 秒に設定すると、既存の容量のスケール対パフォーマンス比が向上し、グレーな障害を防ぐことができます。
- Network Load Balancer を経由する TCP/UDP 接続では、すべての接続が追跡されます。TCP フローのアイドルタイムアウト値は 350 秒、UDP フローのアイドルタイムアウト値は 120 秒で、インターフェイスレベルのタイムアウト値により変化します。ロードバランサーのデフォルトよりもタイムアウトを柔軟にするために、ネットワークインターフェイスレベルでタイムアウトを設定することもできます。

以下の操作を行う際には、接続追跡のタイムアウト設定のオプションが用意されています。

- ネットワークインターフェイスの作成
- ネットワークインターフェイス属性の変更
- EC2 インスタンスの起動
- EC2 インスタンスの起動テンプレートの作成

例

次の例では、セキュリティグループに TCP および ICMP トラフィックを許可するインバウンドルールと、すべてのアウトバウンドトラフィックを許可するアウントバウンドルールがあります。

インバウンド

プロトコルのタイプ	ポート番号	ソース
TCP	22 (SSH)	203.0.113.1/32
TCP	80 (HTTP)	0.0.0.0/0
TCP	80 (HTTP)	::/0
ICMP	すべて	0.0.0.0/0

アウトバウンド

プロトコルのタイプ	ポート番号	送信先
すべて	すべて	0.0.0.0/0
すべて	すべて	::/0

インスタンスまたはネットワークインターフェイスに対して直接ネットワーク接続を確立した場合、追跡動作は次のようになります。

- インバウンドルールでは 203.0.113.1/32 からのトラフィックのみ許可されるため、ポート 22 のインバウンドおよびアウトバウンド TCP トラフィック (SSH) は追跡されますが、必ずしもすべての IP アドレス (0.0.0.0/0) が追跡されるとは限りません。
- インバウンドルールとアウトバウンドルールですべての IP アドレスからのトラフィックが許可されるため、ポート 80 (HTTP) のインバウンドおよびアウトバウンド TCP トラフィックは追跡されません。
- ICMP トラフィックは常に追跡されます。

IPv4 トラフィックのアウトバウンドルールを削除すると、ポート 80 (HTTP) のトラフィックを含めすべてのインバウンドおよびアウトバウンド IPv4 トラフィックが追跡されます。IPv6 トラフィックのアウトバウンドルールを削除すると、IPv6 トラフィックでも同じことが起きます。

さまざまなユースケースのセキュリティグループのルール

セキュリティグループを作成し、そのセキュリティグループに関連付けられたインスタンスのロールを反映したルールを追加できます。例えば、ウェブサーバーとして構成されているインスタンスには、インバウンドの HTTP および HTTPS アクセスを許可するセキュリティグループルールが必要です。同様に、データベースのインスタンスには、データベースのタイプに対するアクセス (MySQL のポート 3306 でのアクセスなど) を許可するルールが必要です。

以下は、特定の種類のアクセスのセキュリティグループに追加できるルールの種類の例です。

例

- ウェブサーバールール
- データベースサーバールール
- コンピュータからのインスタンスへの接続ルール
- 同じセキュリティグループを持つインスタンスからインスタンスに接続するためのルール
- Ping/ICMP のルール
- DNS サーバールール
- Amazon EFS ルール
- Elastic Load Balancing ルール

手順については、「セキュリティグループを作成する」および「the section called “セキュリティグループルールの設定”」を参照してください。

ウェブサーバールール

次のインバウンドルールでは、任意の IP アドレスからの HTTP および HTTPS アクセスを許可します。VPC が IPv6 に対して有効になっている場合、IPv6 アドレスからインバウンド HTTP および HTTPS トラフィックを制御するルールを追加できます。

プロトコルのタイプ	プロトコル番号	ポート	送信元 IP	コメント
TCP	6	80 (HTTP)	0.0.0.0/0	任意の IPv4 アドレスからのインバウンド HTTP アクセスを許可します

プロトコルのタイプ	プロトコル番号	ポート	送信元 IP	コメント
TCP	6	443 (HTTPS)	0.0.0.0/0	任意の IPv4 アドレスからのインバウンド HTTPS アクセスを許可します
TCP	6	80 (HTTP)	::/0	任意の IPv6 アドレスからのインバウンド HTTP アクセスを許可します
TCP	6	443 (HTTPS)	::/0	任意の IPv6 アドレスからのインバウンド HTTPS アクセスを許可します

データベースサーバールール

次のインバウンドルールは、インスタンスで実行中のデータベースのタイプに応じて、データベースアクセス用に追加するルールの例です。Amazon RDS インスタンスの詳細については、Amazon RDS ユーザーガイドを参照してください。

ソース IP には、次のいずれかを指定します。

- ローカルネットワークの特定の IP アドレスまたは IP アドレス範囲 (CIDR ブロック表記)
- データベースにアクセスするインスタンスのグループのセキュリティグループ ID

プロトコルのタイプ	プロトコル番号	ポート	コメント
TCP	6	1433 (MS SQL)	Amazon RDS インスタンス上など、Microsoft SQL Server データベースにアクセスするデフォルトのポート
TCP	6	3306 (MYSQL/Aurora)	Amazon RDS インスタンス上など、MySQL または Aurora データベー

プロトコルのタイプ	プロトコル番号	ポート	コメント
			スにアクセスするデフォルトのポート
TCP	6	5439 (Redshift)	Amazon Redshift クラスターデータベースにアクセスするデフォルトのポート。
TCP	6	5432 (PostgreSQL)	Amazon RDS インスタンス上など、PostgreSQL データベースにアクセスするデフォルトのポート
TCP	6	1521 (Oracle)	Amazon RDS インスタンス上など、Oracle データベースにアクセスするデフォルトのポート

必要に応じて、データベースサーバーからのアウトバウンドトラフィックを制限できます。例えば、ソフトウェアの更新ではインターネットへのアクセスを許可し、その他のトラフィックはすべて制限することができます。最初に、すべてのアウトバンドトラフィックを許可するデフォルトのアウトバウンドルールを削除する必要があります。

プロトコルのタイプ	プロトコル番号	ポート	送信先 IP	コメント
TCP	6	80 (HTTP)	0.0.0.0/0	任意の IPv4 アドレスへのアウトバウンド HTTP アクセスを許可します
TCP	6	443 (HTTPS)	0.0.0.0/0	任意の IPv4 アドレスへのアウトバウンド HTTPS アクセスを許可します
TCP	6	80 (HTTP)	::/0	(IPv6 が有効な VPC のみ) 任意の IPv6 アドレスへのアウト

プロトコルのタイプ	プロトコル番号	ポート	送信先 IP	コメント
				バウンド HTTP アクセスを許可します
TCP	6	443 (HTTPS)	::/0	(IPv6 が有効な VPC のみ)、任意の IPv6 アドレスへのアウトバウンド HTTPS アクセスを許可します

コンピュータからのインスタンスへの接続ルール

インスタンスに接続するには、セキュリティグループに SSH アクセス (Linux インスタンスの場合) または RDP アクセス (Windows インスタンスの場合) を許可するインバウンドルールが必要です。

プロトコルのタイプ	プロトコル番号	ポート	送信元 IP
TCP	6	22 (SSH)	ローカルコンピュータのパブリック IPv4 アドレス、またはローカルネットワークの IP アドレスの範囲。VPC が IPv6 に対して有効で、インスタンスに IPv6 アドレスがある場合、IPv6 アドレスまたは範囲を入力できます。
TCP	6	3389 (RDP)	ローカルコンピュータのパブリック IPv4 アドレス、またはローカルネットワークの IP アドレスの範囲。VPC が IPv6 に対して有効で、インスタンスに IPv6 アドレスがある場合、IPv6 アドレスまたは範囲を入力できます。

同じセキュリティグループを持つインスタンスからインスタンスに接続するためのルール

同じセキュリティグループに関連付けられたインスタンスが相互に通信できるようにするには、そのためのルールを明示的に追加する必要があります。

> **Note**
>
> ミドルボックスアプライアンスを介して異なるサブネット内の 2 つのインスタンス間のトラフィックを転送するようにルートを設定するには、両方のインスタンスのセキュリティグループでインスタンス間のトラフィックがフローできるようにする必要があります。各インスタンスのセキュリティグループは、他のインスタンスのプライベート IP アドレス、または他のインスタンスを含むサブネットの CIDR 範囲を送信元として参照する必要があります。他のインスタンスのセキュリティグループを送信元として参照する場合、インスタンス間のトラフィックは許可されません。

次の表は、関連付けられたインスタンスの相互通信を可能にするセキュリティグループのインバウンドルールを示します。このルールでは、すべてのタイプのトラフィックが許可されます。

プロトコルのタイプ	プロトコル番号	ポート	送信元 IP
-1 (すべて)	-1 (すべて)	-1 (すべて)	セキュリティグループの ID、または他のインスタンスを含むサブネットの CIDR 範囲 (注を参照)。

Ping/ICMP のルール

ping コマンドは、ICMP トラフィックの一種です。インスタンスで Ping を実行するには、次のインバウンド ICMP ルールのうち 1 つを追加する必要があります。

タイプ	プロトコル	ソース		
カスタム ICMP - IPv4	エコーリクエスト	お使いのコンピュータのパブ		

タイプ	プロトコル	ソース		
		リック IPv4 アドレス、特定の IPv4 アドレス、または任意の場所の IPv4 あるいは IPv6 アドレス。		
すべての ICMP - IPv4	IPv4 ICMP (1)	お使いのコンピュータのパブリック IPv4 アドレス、特定の IPv4 アドレス、または任意の場所の IPv4 あるいは IPv6 アドレス。		

ping6 コマンドを使用してインスタンスの IPv6 アドレスに ping を実行するには、次のインバウンド ICMPv6 ルールを追加する必要があります。

タイプ	プロトコル	ソース		
すべての ICMP - IPv6	IPv6 ICMP (58)	お使いのコンピュータの IPv6 アドレス、特定の IPv4 アドレス、または任意の場所の IPv4 あるいは IPv6 アドレス。		

DNS サーバールール

DNS サーバーとして EC2 インスタンスをセットアップした場合、TCP および UDP のトラフィックがポート 53 経由で DNS サーバーに到達できるようにする必要があります。

ソース IP には、次のいずれかを指定します。

- ネットワークの IP アドレスまたは IP アドレス範囲 (CIDR ブロック表記)
- ネットワークで、DNS サーバーにアクセスする必要がある一連のインスタンスのセキュリティグループの ID

プロトコルのタイプ	プロトコル番号	ポート
TCP	6	53
UDP	17	53

Amazon EFS ルール

Amazon EC2 インスタンスで Amazon EFS ファイルシステムを使用している場合、Amazon EFS マウントターゲットに関連付けるセキュリティグループは、NFS プロトコル経由のトラフィックを許可する必要があります。

プロトコルのタイプ	プロトコル番号	ポート	送信元 IP	コメント
TCP	6	2049 (NFS)	セキュリティグループの ID	このセキュリティグループに関連付けられたリソース (マウントターゲットを含む) からのインバウンド NFS アクセスを許可します。

Amazon EC2 インスタンスに Amazon EFS ファイルシステムをマウントするには、インスタンスに接続する必要があります。したがって、インスタンスに関連付けられているセキュリティグループに

は、ローカルコンピュータまたはローカルネットワークからのインバウンド SSH を許可するルールが必要です。

プロトコルのタイプ	プロトコル番号	ポート	送信元 IP	コメント
TCP	6	22 (SSH)	ローカルコンピュータの IP アドレス範囲、またはネットワークの IP アドレス範囲 (CIDR ブロック表記)。	ローカルコンピュータからのインバウンド SSH アクセスを許可します。

Elastic Load Balancing ルール

EC2 インスタンスをロードバランサーに登録する場合、ロードバランサーに関連付けられたセキュリティグループは、インスタンスとの通信を許可する必要があります。詳細については、「Elastic Load Balancing ドキュメント」の次のトピックを参照してください。

- Application Load Balancer のセキュリティグループ
- Network Load Balancer のセキュリティグループ
- Classic Load Balancer のセキュリティグループの設定

Amazon EC2 インスタンスの NitroTPM

Nitro Trusted Platform Module (NitroTPM) は AWS Nitro System によって提供され TPM 2.0 仕様に準拠した仮想デバイスです。インスタンスの認証に使用されるアーティファクト (パスワード、証明書、暗号化キーなど) を安全に保存します。NitroTPM はキーを生成し、暗号化機能 (ハッシング、署名、暗号化、復号化など) に使用できます。

NitroTPM は測定されたブートを提供します。これはブートローダーとオペレーティングシステムがすべてのブートバイナリの暗号化ハッシュを作成し、それらを NitroTPM 内部プラットフォーム構成レジスタ (PCR) の以前の値と組み合わせるプロセスです。測定されたブートを使用することで、NitroTPM から署名された PCR 値を取得し、それらを使用してインスタンスのブートソフトウェアの整合性をリモートエンティティに証明することができます。これはリモート認証と呼ばれます。

NitroTPM を使用することで、キーおよびシークレットに特定の PCR 値をタグ付けできるため、PCR の値、つまりインスタンスの整合性が変更された場合にそれらにアクセスすることはできません。この特別な形式の条件付きアクセスは封印および開封と呼ばれます。BitLocker などのオペレーティングシステムのテクノロジーはNitroTPM を使用してドライブの復号化キーを封印し、オペレーティングシステムが正しく起動し、正常な状態である場合にのみドライブを復号化できます。

NitroTPM を使用するにはNitroTPM サポート用に設定された Amazon マシンイメージ (AMI) を選択してから、その AMI を使用して Nitro ベースのインスタンスを起動する必要があります。Amazon のビルド済みの AMI を選択できるほか、自分で作成することもできます。

料金

NitroTPM を使用しても追加コストはかかりません。お客様は使用した基本リソースに対してのみ、料金を支払います。

内容

- Amazon EC2 インスタンスで NitroTPM を使用するための要件
- NitroTPM 用の Linux AMI の有効化
- AMI が NitroTPM に対して有効になっていることを確認する
- Amazon EC2 インスタンスでの NitroTPM の使用を有効化または停止する
- Amazon EC2 インスタンスが NitroTPM に対して有効になっていることを検証する
- EC2 インスタンスの公開承認キーを取得する

Amazon EC2 インスタンスで NitroTPM を使用するための要件

NitroTPM を有効にしたインスタンスを起動するには以下の要件を満たしている必要があります。

トピック

- AMI
- インスタンスのタイプ
- 考慮事項

AMI

AMI では NitroTPM が有効になっている必要があります。

Linux AMI

事前設定済みの AMI はありません。独自の AMI を設定する必要があります。詳細については「NitroTPM 用の Linux AMI の有効化」を参照してください。

Windows AMI

Microsoft キーを使用して NitroTPM および UEFI Secure Boot 用に事前設定された AWS Windows AMIを見つけるには、「AWSWindows AMI リファレンス」の「NitroTPM と UEFI Secure Boot で設定された Windows Server AMI を見つける」をご覧ください。

> **Note**
>
> オペレーティングシステム - AMI にはTPM 2.0 Command Response Buffer (CRB) ドライバーを搭載したオペレーティングシステムが含まれている必要があります。現在のほとんどのオペレーティングシステムにはTPM 2.0 CRB ドライバーが含まれています。
> UEFI ブートモード — AMI が UEFI ブートモード用に設定されている必要があります。詳細については「Amazon EC2 インスタンスの UEFI 安全ブート」を参照してください。

インスタンスのタイプ

次の仮想化インスタンスタイプのいずれかを使用する必要があります。

- 汎用:
  M5、M5a、M5ad、M5d、M5dn、M5n、M5zn、M6a、M6g、M6gd、M6i、M6id、M6idn、M6in、M7a、M
  flex、M8a、M8azn、M8g、M8gb、M8gd、M8gn、M8i、M8id、M8i-
  flex、M8in、M8idn、M8ine、M8ib、M8idb、M9g、M9gd、T3、T3a、T4g
- コンピューティング最適化:
  C5、C5a、C5ad、C5d、C5n、C6a、C6g、C6gd、C6gn、C6i、C6id、C6in、C7a、C7g、C7gd、C7gn、
  flex、C8a、C8g、C8gb、C8gd、C8gn、C8i、C8id、C8i-flex、C8in、C8ine、C8ib、C9g、C9gd
- メモリ最適化:
  R5、R5a、R5ad、R5b、R5d、R5dn、R5n、R6a、R6g、R6gd、R6i、R6id、R6idn、R6in、R7a、R7g、
  flex、R8in、R8idn、R8ib、R8idb、U7i-6tb、U7i-8tb、U7i-12tb、U7in-16tb、U7in-24tb、U7in-32tb、X2id
- ストレージ最適化: D3、D3en、I3en、I4i、I7i、I7ie、I8g、I8ge、Im4gn
- 高速コンピューティング:
  F2、G4dn、G5、G6、G6e、G6f、Gr6、Gr6f、G7、G7e、Inf1、Inf2、P5、P5e、P5en、P6-
  B200、P6-B300、P6e-GB200、Trn2、Trn2u

- ハイパフォーマンスコンピューティング: Hpc6a、Hpc6id、Hpc8a

考慮事項

以下の考慮事項はNitroTPM を使用する場合に適用されます。

- NitroTPM が有効な AMI を使用してインスタンスを起動した後、インスタンスタイプを変更する場合は選択した新しいインスタンスタイプも NitroTPM をサポートしている必要があります。
- NitrotPM ベースキーで暗号化された BitLocker ボリュームはオリジナルインスタンスでのみ使用できます。
- NitroTPM 状態は Amazon EC2 コンソールには表示されません。
- NitroTPM 状態は Amazon EBS スナップショットには含まれていません。
- NitroTPM 状態は VM Import/Export イメージには含まれていません。
- NitroTPM はAWS Outposts、Local Zones、または Wavelength Zone ではサポートされていません。

NitroTPM 用の Linux AMI の有効化

インスタンスの NitroTPM を有効にするにはNitroTPM が有効な AMI を使用してインスタンスを起動する必要があります。Linux AMI を登録するときに、Linux AMI に NitroTPM サポートを設定する必要があります。NitroTPM サポートを後で設定することはできません。

NitroTPM サポートのために事前設定されている Windows AMI のリストについては「Amazon EC2 インスタンスで NitroTPM を使用するための要件」を参照してください。

RegisterImage API を使用して NitroTPM を設定して AMI を作成する必要があります。Amazon EC2 コンソールまたは VM Import/Export を使用することはできません。

NitroTPM 用の Linux AMI を有効にするには

1. 必要な Linux AMI を使用して一時インスタンスを起動します。ルートボリュームの ID を書き留めます。この ID はコンソールのインスタンスの [ストレージ] タブにあります。
2. インスタンスが `running` 状態になったら、インスタンスのルートボリュームのスナップショットを作成します。詳細については、「EBS ボリュームのスナップショットを作成する」を参照してください。
3. 作成したスナップショットを AMI として登録します。ブロックデバイスマッピングで、ルートボリューム用に作成したスナップショットを指定します。

register-image コマンドの例を次に示します。--tpm-support で、v2.0 を指定します。--boot-mode で、uefi を指定します。

```
aws ec2 register-image \
    --name my-image \
    --boot-mode uefi \
    --architecture x86_64 \
    --root-device-name /dev/xvda \
    --block-device-mappings DeviceName=/dev/
xvda,Ebs={SnapshotId=snap-0abcdef1234567890} \
    --tpm-support v2.0
```

Register-EC2Image コマンドレットの例を次に示します。

```
$block = @{SnapshotId=snap-0abcdef1234567890}
Register-EC2Image `
    -Name my-image `
    -Architecture "x86_64" `
    -RootDeviceName /dev/xvda `
    -BlockDeviceMapping @{DeviceName="/dev/xvda";Ebs=$block} `
    -BootMode Uefi `
    -TpmSupport V20
```

4. ステップ 1 で起動した一時インスタンスを終了します。

AMI が NitroTPM に対して有効になっていることを確認する

インスタンスの NitroTPM を有効にするにはNitroTPM が有効な AMI を使用してインスタンスを起動する必要があります。イメージを記述して、NitroTPM で有効になっていることを確認できます。AMI 所有者の場合は、tpmSupport イメージ属性を記述できます。

Amazon EC2 コンソールには、TpmSupport は表示されません。

AWS CLI

NitroTPM が有効になっていることを確認するには

describe-images コマンドを使用します。

```
aws ec2 describe-images \
```

```
    --image-ids ami-0abcdef1234567890 \
    --query Images[*].TpmSupport
```

NitroTPM が AMI に対して有効になっている場合、出力は次のとおりです。TPM が有効になっていない場合、出力は空です。

```
[
    "v2.0"
]
```

また、AMI 所有者の場合は、describe-image-attribute コマンドと tpmSupport 属性を使用できます。

```
aws ec2 describe-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --attribute tpmSupport
```

以下は出力の例です。

```
{
    "ImageId": "ami-0abcdef1234567890",
    "TpmSupport": {
        "Value": "v2.0"
    }
}
```

NitroTPM が有効になっている AMI を検索するには

次の例では、NitroTPM が有効になっている AMI の ID が一覧表示されています。

```
aws ec2 describe-images \
    --owners self \
    --filters Name=tpm-support,Values=v2.0 \
    --query Images[].ImageId
```

PowerShell

NitroTPM が有効になっていることを確認するには

Get-EC2Image コマンドレットを使用します。

```
Get-EC2Image `
    -ImageId ami-0abcdef1234567890 | Select TpmSupport
```

NitroTPM が AMI に対して有効になっている場合、出力は次のとおりです。TPM が有効になっていない場合、出力は空です。

```
TpmSupport
----------
v2.0
```

または、AMI 所有者の場合は、Get-EC2ImageAttribute コマンドレットと tpmSupport 属性を使用できます。

```
Get-EC2ImageAttribute `
    -ImageId ami-0abcdef1234567890 `
    -Attribute tpmSupport
```

NitroTPM が有効になっている AMI を検索するには

次の例では、NitroTPM が有効になっている AMI の ID が一覧表示されています。

```
Get-EC2Image `
    -Owner self `
    -Filter @{Name="tpm-support; Values="v2.0"} | Select ImageId
```

Amazon EC2 インスタンスでの NitroTPM の使用を有効化または停止する

NitroTPM に対して Amazon EC2 インスタンスを有効にできるのは起動時のみです。NitroTPM に対してインスタンスを有効にすると、無効にすることはできません。NitroTPM を使用する必要がなくなった場合はオペレーティングシステムを設定してその使用を停止する必要があります。

トピック

- NitroTPM を有効にしてインスタンスを起動する
- インスタンスでの NitroTPM の使用を停止する

NitroTPM を有効にしてインスタンスを起動する

前提条件を使用してインスタンスを起動すると、インスタンスで NitroTPM が自動的に有効になります。インスタンスで NitroTPM を有効にできるのは起動時のみです。インスタンスの起動方法についての詳細はAmazon EC2 インスタンスの起動 を参照してください。

インスタンスでの NitroTPM の使用を停止する

NitroTPM を有効にしてインスタンスを起動した後はそのインスタンスに対する NitroTPM を無効にすることはできません。ただし、次のツールを使用して、インスタンスで TPM 2.0 デバイスドライバーを無効にすることで、NitroTPM の使用をオペレーティングシステムで停止するよう設定できます。

- Linux インスタンスの場合はtpm-tools を使用します。
- Windows インスタンスの場合は TPM 管理コンソールの tpm.msc) を使用します。

デバイスドライバーを無効化する方法の詳細についてはオペレーティングシステムのドキュメントを参照してください。

Amazon EC2 インスタンスが NitroTPM に対して有効になっていることを検証する

Amazon EC2 インスタンスが NitroTPM に対して有効であるかどうかを確認できます。インスタンスで NitroTPM サポートが有効になっている場合、コマンドは `"v2.0"` を返します。それ以外の場合、`TpmSupport` フィールドは出力に表示されません。

Amazon EC2 コンソールには `TpmSupport` フィールドは表示されません。

AWS CLI

インスタンスが NitroTPM に対して有効になっているかどうかを検証するには

describe-instances コマンドを使用します。

```
aws ec2 describe-instances \
    --instance-ids i-1234567890abcdef0 \
    --query Reservations[].Instances[].TpmSupport
```

PowerShell

インスタンスが NitroTPM に対して有効になっているかどうかを検証するには

Get-EC2Instance コマンドレットを使用します。

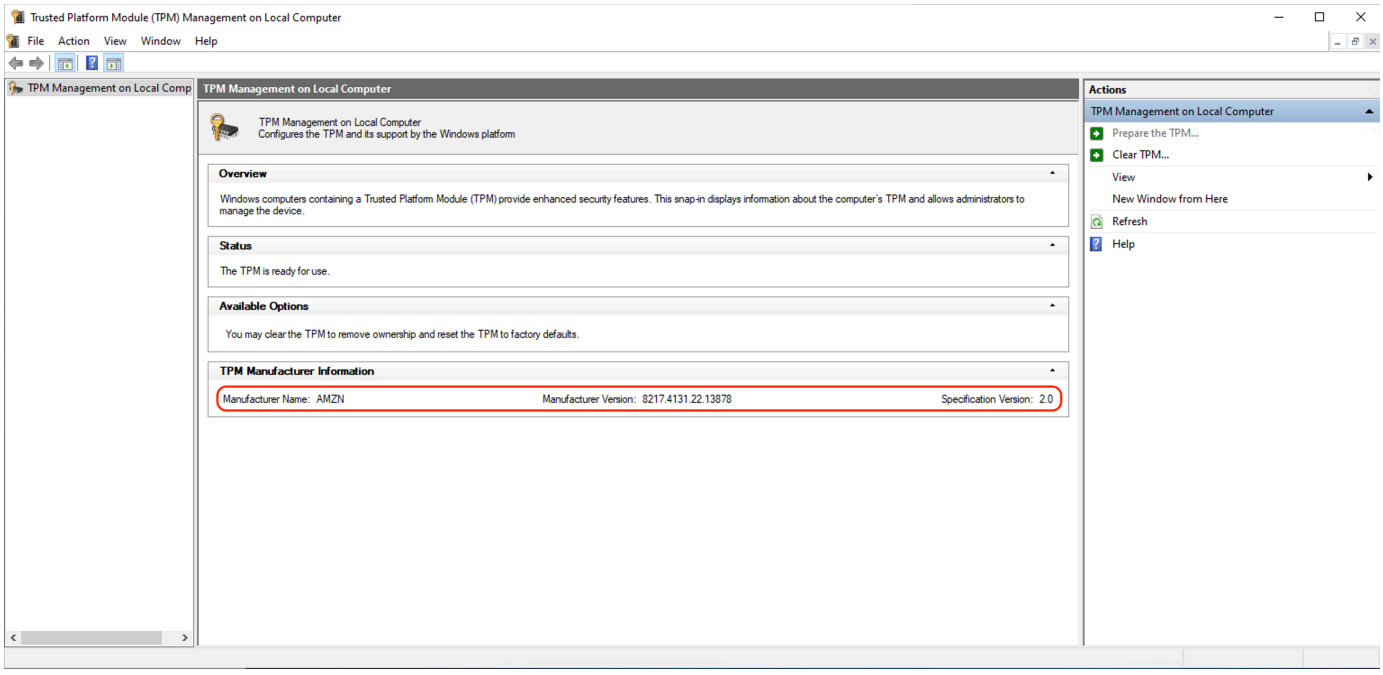
```
(Get-EC2Instance `
    -InstanceId i-1234567890abcdef0).Instances.TpmSupport
```

Windows インスタンスでの NitroTPM アクセスの確認

(Windows インスタンスのみ) Windows が NitroTPM にアクセスできるかどうかを検証するには

1. EC2 Windows インスタンスに接続します。
2. インスタンスで、tpm.msc プログラムを実行します。

   [TPM Management on Local Computer] (ローカルコンピュータでの TPM 管理) ウィンドウが開きます。
3. [TPM Manufacturer Information] (TPM 製造元の情報) フィールドをチェックします。フィールドには製造元の名前とインスタンスの NitroTPM バージョンが含まれます。



EC2 インスタンスの公開承認キーを取得する

インスタンスの公開承認キーは、いつでも安全に取得できます。

AWS CLI

インスタンスの公開承認キーを取得する方法

get-instance-tpm-ek-pub コマンドを使用します。

例 1

以下の例では、指定したインスタンスの rsa-2048 公開承認鍵を tpmt 形式で取得しています。

```
aws ec2 get-instance-tpm-ek-pub \
    --instance-id i-1234567890abcdef0 \
    --key-format tpmt \
    --key-type rsa-2048
```

以下は出力例です。

```
{
    "InstanceId": "i-01234567890abcdef",
    "KeyFormat": "tpmt",
    "KeyType": "rsa-2048",
    "KeyValue": "AAEACwADALIAIINxl2dEhLEXAMPLEUal1yT9UtduBlILZPKh2hszFGmqAAYAgABDA
    EXAMPLEAAABAOiRd7WmgtdGNoV1h/AxmW+CXExblG8pEUfNm0LOLiYnEXAMPLERqApiFa/UhvEYqN4
    Z7jKMD/usbhsQaAB1gKA5RmzuhSazHQkax7EXAMPLEzDthlS7HNGuYn5eG7qnJndRcakS+iNxT8Hvf
    0S1ZtNuItMs+Yp4SO6aU28MT/JZkOKsXIdMerY3GdWbNQz9AvYbMEXAMPLEPyHfzgVO0QTTJVGdDxh
    vxtXCOu9GYf0crbjEXAMPLEd4YTbWdDdgOKWF9fjzDytJSDhrLAOUctNzHPCd/92l5zEXAMPLEOIFA
    Ss50C0/802c17W2pMSVHvCCa9lYCiAfxH/vYKovAAE="
}
```

例 2

以下の例では、指定したインスタンスの rsa-2048 公開承認鍵を der 形式で取得しています。

```
aws ec2 get-instance-tpm-ek-pub \
    --instance-id i-1234567890abcdef0 \
    --key-format der \
    --key-type rsa-2048
```

以下は出力例です。

```
{
    "InstanceId": "i-1234567890abcdef0",
    "KeyFormat": "der",
    "KeyType": "rsa-2048",
    "KeyValue": "MIIBIjANBgEXAMPLEw0BAQEFAAOCAQ8AMIIBCgKCAQEA6JF3taEXAMPLEXWH8DGZb4
    JcTFuUbykRR82bQs4uJifaKSOv5NGoEXAMPLEG8Rio3hnuMowP+6xuGxBoAHWAoDlGbO6FJrMdEXAMP
    LEnYUHvMO2GVLsc0a5ifl4buqcmd1FxqRL6I3FPwe9/REXAMPLE0yz5inhI7ppTbwxP8lmQ4qxch0x6
    tjcZ1Zs1DP0EXAMPLERUYLQ/Id/OBU7RBNMlUZ0PGG/G1cI670Zh/RytuOdx9iEXAMPLEtZ0N2A4pYX
    1+PMPK0lIOGssA5Ry03Mc8J3/3aXnOD2/ASRQ4gUBKznQLT/zTZEXAMPLEJUe8IJr2VgKIB/Ef+9gqi
    8AAQIDAQAB"
}
```

PowerShell

インスタンスの公開承認キーを取得する方法

Get-EC2InstanceTpmEkPub コマンドレットを使用します。

例 1

以下の例では、指定したインスタンスの rsa-2048 公開承認鍵を tpmt 形式で取得しています。

```
Get-EC2InstanceTpmEkPub `
    -InstanceId i-1234567890abcdef0 `
    -KeyFormat tpmt `
    -KeyType rsa-2048
```

例 2

以下の例では、指定したインスタンスの rsa-2048 公開承認鍵を der 形式で取得しています。

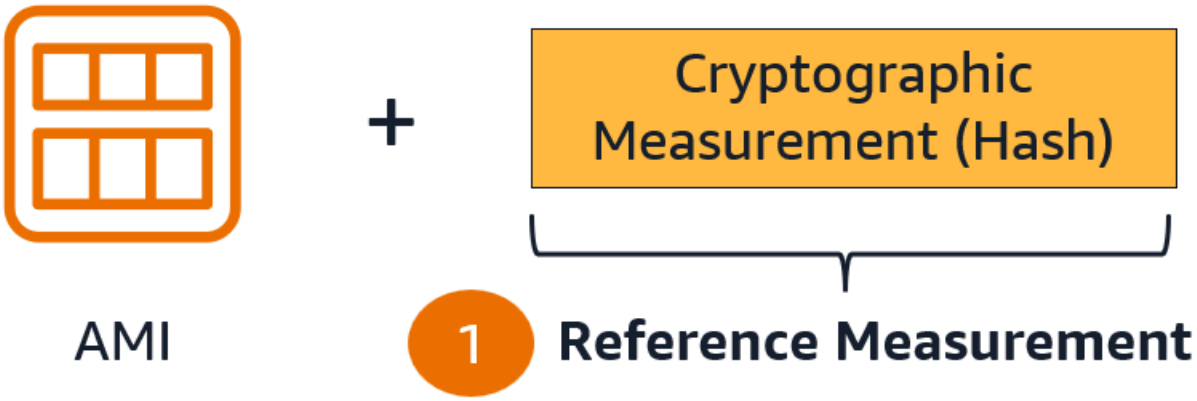
```
Get-EC2InstanceTpmEkPub `
    -InstanceId i-1234567890abcdef0 `
    -KeyFormat der `
    -KeyType rsa-2048
```

Amazon EC2 インスタンス構成証明

アテステーションは、信頼できるソフトウェア、ドライバー、ブートプロセスのみが Amazon EC2 インスタンスで実行されていることを任意の当事者に暗号的に証明できるプロセスです。Amazon

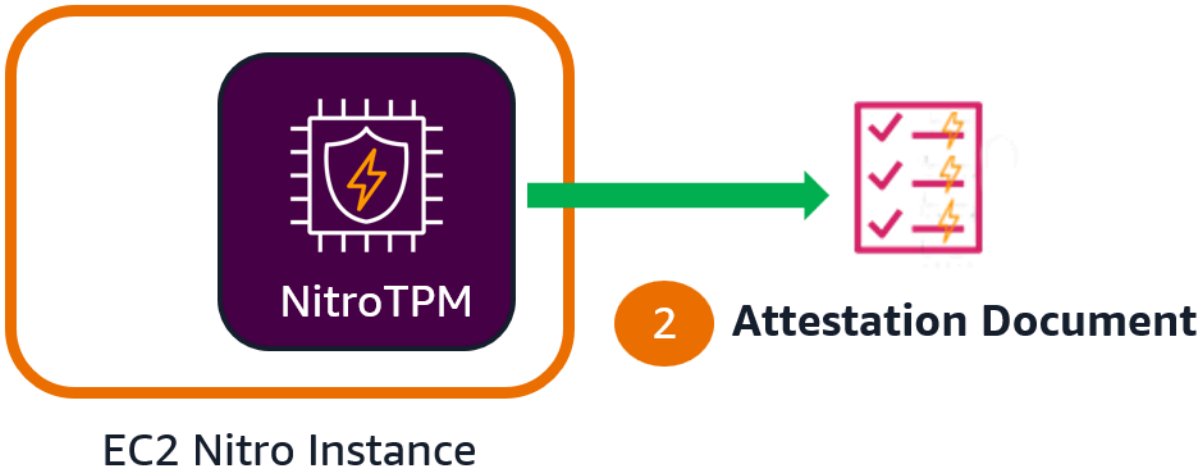
EC2 インスタンス構成証明は、Nitro Trusted Platform Module (NitroTPM) と 構成証明可能 AMI によって実現されます。

アテステーションの最初のステップは、構成証明可能 AMI を構築し、その AMI のリファレンス測定値を決定することです。構成証明可能 AMI は、アテステーションのためにゼロから構築された AMI です。リファレンス測定値は、AMI に含めたすべてのソフトウェアと設定の測定値です。リファレンス測定値を取得する方法の詳細については、「サンプルイメージの説明を作成する」を参照してください。



次のステップでは、構成証明可能 AMI を使用して Nitro-TPM 対応 EC2 インスタンスを起動します。インスタンスを起動したら、NitroTPM ツールを使用してアテステーションドキュメントを生成できます。次に、アテステーションドキュメントからの EC2 インスタンスの実際の測定値とリファレンス測定値を比較して、インスタンスに信頼できるソフトウェアと設定があるかどうかをチェックできます。

構成証明可能 AMI の作成プロセス中に生成されたリファレンス測定値をインスタンスの構成証明ドキュメントに含まれる測定値と比較することで、信頼できるソフトウェアとコードのみがインスタンスで実行されていることを検証できます。



AWS KMS との統合

測定値の比較プロセスを容易にするために、アテステーションドキュメントの検証ツールとして AWS Key Management Service (AWS KMS) を使用できます。AWS KMS では、リファレンス測定値に一致する測定値をアテステーションドキュメントに提供する場合にのみ、KMS キーを使用した特定のオペレーションを許可するアテステーションベースの KMS キーポリシーを作成できます。これを行うには、リファレンス測定値を条件キー値として使用する特定の条件キーを KMS キーポリシーに追加し、条件キーが満たされた場合に許可される KMS オペレーションを指定します。

KMS キーを使用して KMS オペレーションを実行する場合、アテステーションドキュメントを KMS リクエストにアタッチする必要があります。そうすると、AWS KMS は、アテステーションドキュメントからの測定値を KMS キーポリシーのリファレンス測定値と照合し、測定値が一致する場合にのみキーアクセスを許可します。

さらに、インスタンスのアテステーションドキュメントを生成するときは、所有するキーペアのパブリックキーを指定する必要があります。指定されたパブリックキーは、アテステーションドキュメントに含まれています。AWS KMS がアテステーションドキュメントを検証し、復号オペレーションを許可すると、返される前にアテステーションドキュメントに含まれるパブリックキーを使用してレスポンスが自動的に暗号化されます。これにより、レスポンスは、アテステーションドキュメントに含まれるパブリックキーに一致するプライベートキーでのみ復号し使用できます。

これにより、信頼できるソフトウェアとコードを実行しているインスタンスのみが KMS キーを使用して暗号オペレーションを実行できます。

分離されたコンピューティング環境のアテステーション

一般に、EC2 インスタンスを分離されたコンピューティング環境として構築および設定できます。これにより、管理者やユーザーが EC2 インスタンスで処理されているデータにアクセスするためのインタラクティブなアクセスやメカニズムが提供されることはありません。EC2 インスタンスアテステーションを使用すると、インスタンスが分離されたコンピューティング環境として実行されていることをサードパーティーまたはサービスに証明できます。詳細については、「自社のオペレーターからデータを分離する」を参照してください。

例については、分離されたコンピューティング環境を作成する Amazon Linux 2023 のサンプルイメージの説明を参照してください。このサンプルイメージの説明を出発点にして、要件を満たすようにカスタマイズできます。

AWS 責任共有モデル

NitroTPM と構成証明可能 AMI は、EC2 インスタンスでアテステーションを設定および構成するのに役立つ構成要素です。お客様は、それぞれのユースケースに合わせて AMI を設定する責任があります。詳細については、「AWS 責任共有モデル」を参照してください。

トピック

- 構成証明可能 AMI
- アテステーション用に AWS KMS を準備する
- NitroTPM アテステーションドキュメントを取得する
- AWS KMS との統合
- 自社のオペレーターからデータを分離する

構成証明可能 AMI

構成証明可能 AMI は、すべてのコンテンツを表す対応する暗号化ハッシュを持つ Amazon マシンイメージ (AMI) です。ハッシュは AMI の作成プロセス中に生成され、アプリケーション、コード、起動プロセスなど、その AMI のコンテンツ全体に基づいて計算されます。

Attestable State の維持

インスタンスの測定値は、初期起動状態に基づいています。起動後にインスタンスでソフトウェアまたはコードが変更され、再起動後も保持されていると、再起動後にインスタンスの測定値が変更され

ます。測定値が変更されると、構成証明可能 AMI のリファレンス測定値から逸脱し、インスタンスの再起動後にインスタンスは AWS KMS に対して正常にアテステーションができなくなります。したがって、構成証明可能 AMI が役立つように、インスタンスは再起動後に元のブート状態に戻る必要があります。

常に元のブート状態に戻るようにすると、インスタンスは再起動後に正常にアテステーションができます。次のユーティリティを使用して、再起動後もインスタンスのアテステーションを可能にできます。

- `erofs` – 拡張読み取り専用ファイルシステム。このユーティリティにより、ルートファイルシステムが読み取り専用になります。このユーティリティを使用すると、`/etc`、`/run`、`/var` などのファイルシステムへの書き込みはメモリに保存され、インスタンスの再起動時に失われるため、ルートファイルシステムは元の起動状態のままになります。詳細については、erofs のドキュメントを参照してください。
- `dm-verity` – 読み取り専用ルートファイルシステムの整合性を保護します。このユーティリティはファイルシステムブロックのハッシュを計算し、カーネルコマンドラインに保存します。これにより、カーネルは起動時にファイルシステムの整合性を検証できます。詳細については、dm-verity のドキュメントを参照してください。

構成証明可能 AMI を作成するための要件

構成証明可能 AMI には以下の要件があります。

- ベースとなるオペレーティングシステム – Amazon Linux 2023 および NixOS
- アーキテクチャ – `x86_64` または `arm64` アーキテクチャ
- TPM サポート – NitroTPM を有効にする必要があります。詳細については、「Amazon EC2 インスタンスで NitroTPM を使用するための要件」を参照してください。
- ブートモード – UEFI ブートモードを有効にする必要があります。

トピック

- 構成証明可能 AMI の作成
- Amazon Linux 2023 のサンプルイメージの説明を作成する
- Amazon Linux 2023 のサンプルイメージの説明
- ワークロードに合わせてAmazon Linux 2023 のサンプルイメージの説明をカスタマイズする
- カスタム AMI の PCR 測定値を計算する

構成証明可能 AMI の作成

構成証明可能 AMI を作成するには、KIWI Next Generation (KIWI NG) で Amazon Linux 2023 を使用する必要があります。Amazon Linux 2023 は、KIWI NG を使用して構成証明可能 AMI を構築するために必要なすべてのソフトウェアとユーティリティを提供します。

KIWI NG は、事前設定された Linux ベースのイメージを構築するためのオープンソースツールです。KIWI NG は、イメージの内容を定義する XML によるイメージの説明を使用します。イメージの説明では、特定のユースケース向けにすぐに使用できる AMI を構築するために実行するベースオペレーティングシステム、ソフトウェア、カーネル設定、スクリプトを指定します。

AMI のビルド時は、`nitro-tpm-pcr-compute` ユーティリティを使用して、KIWI NG によって生成された Unified Kernel Image (UKI) に基づいてリファレンス測定値を生成する必要があります。`nitro-tpm-pcr-compute` ユーティリティの使用に関する詳細については、「カスタム AMI の PCR 測定値を計算する」を参照してください。

AWS は、分離されたコンピューティング環境で EC2 インスタンスを設定するために必要なすべての設定を含む Amazon Linux 2023 のサンプルイメージの説明を提供します。詳細については、「Amazon Linux 2023 のサンプルイメージの説明を作成する」を参照してください。

Amazon Linux 2023 のサンプルイメージの説明を作成する

AWS は、ワークロード用に独自のカスタム構成証明可能 AMI を作成するための出発点として使用できる Amazon Linux 2023 のサンプルイメージの説明を提供します。サンプルイメージの説明には、ベースオペレーティングシステムとしての Amazon Linux 2023、ファイルシステムのイミュータビリティ用の `dm-verity` と `erofs` の設定が含まれ、分離されたコンピューティング環境を作成するためにすべてのインタラクティブアクセス (SSH、EC2 インスタンス接続、シリアルコンソールなど) が削除されます。サンプルイメージの説明の詳細については、Github リポジトリを参照してください。

サンプルイメージの説明では、NitroTPM ツール (`nitro-tpm-pcr-compute` および `nitro-tpm-attest`) が `/usr/bin/` ディレクトリのビルド済みイメージに自動的にインストールされます。これにより、AMI から起動されたインスタンスにツールがプリインストールされます。

サンプルイメージの説明には、リファレンス測定値の生成に必要なコマンドを含むスクリプトである `edit_boot_install.sh` が含まれています。スクリプトは、KIWI NG によって作成された raw ディスクイメージファイル (`.raw`) をループバックデバイスにマウントし、UKI (`.efi` ファイル拡張子を持つ) を見つけ、`nitro-tpm-pcr-compute` ユーティリティを実行して AMI のリファレンス測定値を生成します。スクリプトは、ビルド時に KIWI NG によって自動的に実行されます。

このチュートリアルでは、サンプルイメージの説明を作成して構成証明可能 AMI を作成する方法を示します。

独自のイメージの説明の作成に関する詳細については、次の KIWI NG ドキュメントを参照してください。

- クイックスタート
- イメージの説明
- Amazon Linux 2023 のサンプルイメージの説明

前提条件

このチュートリアルを完了するには、IAM アイデンティティに次の許可が必要です。

- `arn:aws:ec2:*::snapshot/*` に対する `ebs:CompleteSnapshot`、`ebs:StartSnapshot`、`ebs:PutSnapshotBlock`
- すべてのリソースの `ec2:RegisterImage`

KIWI NG を使用して Amazon Linux 2023 のサンプルイメージの説明を構築するには

1. 最新の AL2023 AMI を使用して Amazon EC2 インスタンスを起動する インスタンスに AMI を構築するのに十分なストレージスペースを確保するために、少なくとも 12 GB のストレージをプロビジョニングしてください。
2. 必要な依存ファイルをインストールします。次のコマンドは、次のユーティリティをインストールします。

   - `kiwi-cli`
   - `veritysetup`
   - `erofs-utils`
   - `aws-nitro-tpm-tools`

   ```
   sudo dnf install -y kiwi-cli python3-kiwi kiwi-systemdeps-core python3-poetry-core
    qemu-img veritysetup erofs-utils git cargo aws-nitro-tpm-tools
   ```
3. `coldsnap` ユーティリティをインストールします。このユーティリティを使用すると、raw イメージデータから Amazon EBS スナップショットを作成できます。このユーティリティを使用

して、KIWI NG によって作成された raw ディスクイメージファイルから EBS スナップショットを作成します。

```
git clone https://github.com/awslabs/coldsnap.git
cd coldsnap
cargo install --locked coldsnap
cd ..
```

4. サンプルイメージの説明ファイルを取得します。

```
sudo dnf install kiwi-image-descriptions-examples
```

サンプルイメージの説明ファイルは、/usr/share/kiwi-image-descriptions-examples/al2023/attestable-image-example のディレクトリにダウンロードされます。

5. KIWI NG system build コマンドを使用してサンプルイメージの説明を作成します。次のコマンドは、./image ディレクトリに raw ディスクイメージファイルを作成します。

```
sudo kiwi-ng \
--color-output \
--loglevel 0 \
system build \
--description /usr/share/kiwi-image-descriptions-examples/al2023/attestable-image-example \
--target-dir ./image
```

詳細については、kiwi-ng system build のドキュメントを参照してください。

6. AMI のリファレンス測定値を取得します。測定値は、前のステップのイメージのビルド時に nitro-tpm-pcr-compute ユーティリティによって生成されます。リファレンス測定値は、./image/pcr_measurements.json というファイルにあります。

測定値は、次の JSON 形式で提供されます。

```
{
  "Measurements": {
    "HashAlgorithm": "SHA384 { ... }",
    "PCR4": "PCR4_measurement",
    "PCR7": "PCR7_measurement",
    "PCR12": "PCR12_measurement"
  }
```

```
}
```

7. coldsnap ユーティリティを使用して、KIWI NG によって作成された raw ディスクイメージを EBS スナップショットにアップロードします。コマンドはスナップショット ID を返します。次のステップで必要になるため、ID を書きとめておきます。

```
SNAPSHOT=$(.cargo/bin/coldsnap upload ./image/al2023*.raw)
echo "Created snapshot: $SNAPSHOT"
```

   coldsnap ユーティリティの詳細については、コールドスナップ GitHub リポジトリを参照してください。

8. 前のステップのスナップショットを使用して、TPM 2.0 対応 AMI を UEFI ブートモードで登録します。--architecture には、Intel の場合は x86_64、Graviton の場合は arm64 を指定します。

```
aws ec2 register-image \
--name "attestable_isolated_al2023_ami" \
--virtualization-type hvm \
--boot-mode uefi \
--architecture x86_64|arm64 \
--root-device-name /dev/xvda \
--block-device-mappings DeviceName=/dev/xvda,Ebs={SnapshotId=${SNAPSHOT}} \
--tpm-support v2.0 \
--ena-support
```

Amazon Linux 2023 のサンプルイメージの説明

Amazon Linux 2023 のサンプルイメージの説明には、次の特性があります。

1. Unified Kernel Image (UKI) ブート – カーネル、initrd、およびブートパラメータを 1 つのイミュータブルイメージに結合する単一の署名付きバイナリを使用してブートします。
2. 読み取り専用ルートファイルシステム – dm-verity 保護で拡張読み取り専用ファイルシステム (erofs) を使用して、ルートファイルシステムを変更できなくし、暗号整合性の検証を維持します。
3. 一時オーバーレイファイルシステム – /etc、/run、/var などのディレクトリへの一時的な書き込みを許可する一時オーバーレイファイルシステムを作成します。このオーバーレイファイルシステムはメモリにのみ存在するため、インスタンスの再起動時にすべての変更が自動的に失われるため、システムは元の信頼された状態に戻ります。

4. 無効なリモートアクセス方法 – リモートアクセスを防ぐために、次のリモートアクセスメカニズムを削除します。

アクセス方法	説明	イメージの説明の実装
SSH	OpenSSH サーバーを除外します。インスタンスが本質的に SSH トラフィックを処理できないようにします。	`openssh-server` パッケージを無視する *
[User Data]	Cloud-init を削除します。オペレーターがインスタンスにユーザーデータを提供し、ブートタイムスクリプトを実行する機能を除去します。	`cloud-init` および `cloud-init-cfg-ec2` パッケージを無視する *
Chrony	chrony コマンドポートを無効にします。実行中のインスタンスでオペレーターが chrony コマンドを実行できないようにします。	`amazon-chrony-config` パッケージを無視する *
MOTD	MOTD パッケージを削除します。実行中のインスタンスでオペレーターがメッセージや機能を変更できないようにします。	`update-motd` パッケージを無視する *
AWS SSM	AWS SSM エージェントを削除します。AWS SSM を使用して実行中のインスタンスへのリモートアクセスができないようにします。	`amazon-ssm-agent` パッケージを無視する *
EC2 Instance Connect	EC2 Instance Connect パッケージを削除します。このツールを使用して SSH アクセスを無効にします。	`ec2-instance-connect` パッケージを無視する *

アクセス方法	説明	イメージの説明の実装
シリアルコンソール	シリアルコンソールを無効にします。実行中のインスタンスでコンソールアクセスが使用できないことを確認し、オペレーターがシリアルコンソールにログインする機能を削除します。	カーネルコマンドラインパラメータで無効化

* 詳細については、「イメージ説明の要素」を参照してください。

ワークロードに合わせてAmazon Linux 2023 のサンプルイメージの説明をカスタマイズする

Amazon Linux 2023 のサンプルイメージの説明をカスタマイズし、特定のワークロードに必要なソフトウェアパッケージ、スクリプト、ファイルを含めることができます。カスタマイズは、KIWI NG イメージの説明のさまざまな要素に追加または変更することで実現されます。

トピック

- リポジトリ管理
- パッケージ管理
- ファイルとディレクトリの追加
- カスタムスクリプトの追加

リポジトリ管理

デフォルトでは、サンプルイメージの説明には、Amazon Linux 2023 コアリポジトリのミラーエンドポイントを指す単一の `<repository>` 要素が含まれています。必要に応じて、必要なソフトウェアをインストールする他のリポジトリへの参照を追加できます。

サンプルイメージの説明では、`<packagemanager>` 要素で定義されている `dnf` パッケージマネージャーを使用します。

リポジトリの追加の詳細については、「リポジトリのセットアップ」を参照してください。

パッケージ管理

デフォルトでは、サンプルイメージの説明には、erofs 読み取り専用ファイルシステムを持つ分離されたコンピューティング環境用の Amazon Linux 2023 構成証明可能 AMI を作成するために必要なすべてのパッケージが含まれています。

イメージの説明の <packages> 要素に追加することで、イメージの説明に追加のソフトウェアパッケージを含めることができます。<packages> 要素は、AMI にインストールする必要があるすべてのソフトウェアを定義します。

<packages> 要素を使用して、特定のソフトウェアパッケージをアンインストールまたは削除することもできます。

イメージの説明でパッケージを追加または削除する方法の詳細については、「パッケージの追加と削除」を参照してください。

ファイルとディレクトリの追加

サンプルイメージの説明には、オーバーレイツリーディレクトリ (/root/) が含まれています。オーバーレイツリーディレクトリは、イメージビルドプロセス中にイメージにコピーされるファイルとディレクトリを含むディレクトリです。オーバーレイツリーディレクトリに配置するファイルとディレクトリは、イメージビルドプロセス中にイメージのルートファイルシステムに直接コピーされます。

すべてのパッケージがインストールされると、オーバーレイツリーディレクトリがイメージにコピーされます。新しいファイルが追加され、既存のファイルが上書きされます。

カスタムスクリプトの追加

サンプルイメージの説明には、単一のカスタムスクリプト edit_boot_install.sh が含まれています。このスクリプトには、イメージの内容に基づいてリファレンス測定値を生成する nitro-tpm-pcr-compute ユーティリティの実行に必要なコマンドが含まれています。このスクリプトは、ブートローダーのインストール直後に呼び出されます。

必要に応じて、イメージの説明に独自のカスタムスクリプトを含めて、イメージビルドプロセス中またはイメージの最初の起動時にタスクまたは設定を実行できます。スクリプトを使用すると、イメージの説明だけでは実現できない方法でイメージをカスタマイズできます。

イメージの説明にカスタムスクリプトを含めるには、スクリプトのタイプに基づいてスクリプトに正しく名前を付け、appliance.kiwi ファイルと同じディレクトリに追加する必要があります。KIWI NG は、スクリプトの名前が正しくて正しい場所に配置されていれば、スクリプトを自動的に検出して実行します。イメージの説明ファイルでスクリプトを明示的に参照する必要はありません。

KIWI NG でサポートされているスクリプトの詳細については、「ユーザー定義スクリプト」を参照してください。

カスタム AMI の PCR 測定値を計算する

`nitro-tpm-pcr-compute` ユーティリティを使用すると、Unified Kernel Image (UKI) に基づいて、ビルド時に構成証明可能 AMI のリファレンス測定値を生成できます。

Amazon Linux 2023 のサンプルイメージの説明では、`/usr/bin/` ディレクトリのビルド済みイメージにユーティリティが自動的にインストールされます。サンプルイメージの説明には、イメージのビルド時にユーティリティを実行してリファレンス測定値を生成するために必要なコマンドを含むスクリプトも含まれています。サンプルイメージの説明を使用している場合は、ユーティリティをインストールしたり、手動で実行したりする必要はありません。詳細については、「Amazon Linux 2023 のサンプルイメージの説明を作成する」を参照してください。

`nitro-tpm-pcr-compute` ユーティリティをインストールする

Amazon Linux 2023 を使用している場合は、次のように Amazon Linux リポジトリから `nitro-tpm-pcr-compute` ユーティリティをインストールできます。

```
sudo yum install aws-nitro-tpm-tools
```

ツールは `/usr/bin` ディレクトリにインストールされます。

`nitro-tpm-pcr-compute` ユーティリティを使用する

ユーティリティは、リファレンス測定値を生成するための単一のコマンド `nitro-tpm-pcr-compute` を提供します。

コマンドの実行時に以下を指定する必要がありますます。

- Unified Kernel Image (`UKI.efi`) – 標準ブートと UEFI に必要です。

構成証明可能 AMI のリファレンス測定値を生成するには:

次のコマンドとパラメータを使用します。

```
/usr/bin/nitro-tpm-pcr-compute \
--image UKI.efi
```

ユーティリティは、リファレンス測定値を次の JSON 形式で返します。

```
{
  "Measurements": {
    "HashAlgorithm": "SHA384 { ... }",
    "PCR4": "PCR4_measurement",
    "PCR7": "PCR7_measurement",
    "PCR12": "PCR12_measurement"
  }
}
```

`nitro-tpm-pcr-compute` ユーティリティの使用方法の実例については、Amazon Linux 2023 のサンプルイメージの説明に含まれている `edit_boot_install.sh` スクリプトを参照してください。

アテステーション用に AWS KMS を準備する

> **Note**
>
> サードパーティーのサービスに対してアテステーションを行う場合は、アテステーションドキュメントを受信、解析、検証するための独自のカスタムメカニズムを構築する必要があります。詳細については、「NitroTPM アテステーションドキュメントを検証する」を参照してください。

構成証明可能 AMI を作成したら、Amazon EC2 インスタンスからのリクエストを検証するために使用できるリファレンス測定値が必要です。AWS KMS には、NitroTPM によるアテステーションのサポートが組み込まれています。

シークレットデータの暗号化に使用した AWS KMS キーでは、構成証明可能 AMI の作成プロセス中に生成したリファレンス測定値に一致する測定値を使用するアテステーションドキュメントが API リクエストに含まれている場合にのみ、キーアクセスを許可するキーポリシーを追加します。標準ブートには PCR4 および PCR12 測定値を使用し、安全ブートには PCR7 測定値を使用します。これにより、構成証明可能 AMI を使用して起動されたインスタンスからのリクエストのみが、AWS KMS キーを使用して暗号オペレーションを実行できます。

AWS KMS には、NitroTPM KMS キーポリシーのアテステーションベースの条件を作成できる `kms:RecipientAttestation:NitroTPMPCR4`、`kms:RecipientAttestation:NitroTPMPCR7`、`kms` 条件キーが用意されています。詳細については、「Condition keys for NitroTPM」を参照してください。

例えば、次の AWS KMS キーポリシーでは、MyEC2InstanceRole インスタンスプロファイルがアタッチされたインスタンスからリクエストが発生し、特定の PCR 4 値および PCR 12 値を含むアテステーションドキュメントがリクエストに含まれている場合にのみ、キーアクセスを許可します。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Allow requests from instances with attested AMI only",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/MyEC2InstanceRole"
      },
      "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey",
        "kms:GenerateRandom"
      ],
      "Resource": "*",
      "Condition": {
        "StringEqualsIgnoreCase": {

 "kms:RecipientAttestation:NitroTPMPCR4":"EXAMPLE6b9b3d89a53b13f5dfd14a1049ec0b80a9ae4b159adde4

 "kms:RecipientAttestation:NitroTPMPCR12":"000000000000000000000000000000000000000000000000000000
        }
      }
    }
  ]
}
```

NitroTPM アテステーションドキュメントを取得する

アテステーションドキュメントは、NitroTPM アテステーションプロセスの主要なコンポーネントです。これには、インスタンスのアイデンティティを検証し、信頼できるソフトウェアのみを実行していることを証明するために使用できる一連の暗号化測定値が含まれています。AWS KMS を持つアテステーションドキュメントを使用して、NitroTPM アテステーションの組み込みサポートを利用したり、独自の暗号化アテステーションメカニズムを構築したりできます。

nitro-tpm-attest ユーティリティを使用すると、実行時に Amazon EC2 インスタンスの署名付き NitroTPM アテステーションドキュメントを取得できます。

Amazon Linux 2023 のサンプルイメージの説明では、/usr/bin/ ディレクトリのビルド済みイメージにユーティリティが自動的にインストールされます。これにより、AMI を使用して起動されたインスタンスにユーティリティがプリインストールされます。ユーティリティを手動でインストールする必要はありません。詳細については、「Amazon Linux 2023 のサンプルイメージの説明を作成する」を参照してください。

トピック

- nitro-tpm-attest ユーティリティをインストールする
- nitro-tpm-attest ユーティリティを使用する
- NitroTPM アテステーションドキュメントの内容
- NitroTPM アテステーションドキュメントを検証する

nitro-tpm-attest ユーティリティをインストールする

Amazon Linux 2023 を使用している場合は、次のように Amazon Linux リポジトリから nitro-tpm-attest ユーティリティをインストールできます。

```
sudo yum install aws-nitro-tpm-tools
```

nitro-tpm-attest ユーティリティを使用する

ユーティリティには、アテステーションドキュメントを取得するための単一のコマンド nitro-tpm-attest が用意されています。コマンドからは、Concise Binary Object Representation (CBOR) でエンコードされ、CBOR Object Signing and Encryption (COSE) を使用して署名されたアテステーションドキュメントが返されます。

コマンドの実行時に以下のパラメータをオプションで指定できます。

- public-key – レスポンスデータが返される前に暗号化するために AWS KMS または外部サービスが使用できるパブリックキー。これにより、プライベートキーを所有している本来の受信者のみがデータを復号できます。例えば、AWS KMS でアテステーションを行う場合、サービスはアテステーションドキュメントのパブリックキーを使用してプレーンテキストデータを暗号化し、結果の暗号文をレスポンスの CiphertextForRecipient フィールドに返します。RSA キーのみがサポートされています。
- user-data – ユーザーデータを使用して、追加の署名付きデータを外部サービスに配信できます。このユーザーデータは、リクエスト元のインスタンスと外部サービスとの間で合意されたプロトコルを完了するために使用できます。AWS KMS でのアテステーションには使用されません。

- nonce – ノンスを使用してインスタンスと外部サービス間のチャレンジレスポンス認証を設定し、なりすまし攻撃を防ぐことができます。ノンスを使用すると、外部サービスは、インスタンスとライブでやり取りしていて、古いアテステーションドキュメントを再利用している偽装者とのやり取りではないことを確認できます。AWS KMS でのアテステーションには使用されません。

アテステーションドキュメントを取得するには

以下のコマンドとオプションのパラメータを使用します。

```
/usr/bin/nitro-tpm-attest \
--public-key rsa_public_key \
--user-data user_data \
--nonce nonce
```

RSA キーペアを生成する方法と、パブリックキーによるアテステーションをリクエストする方法を示す完全な例については、nitro-tpm-attest GitHub リポジトリを参照してください。

NitroTPM アテステーションドキュメントの内容

アテステーションドキュメントは NitroTPM によって生成され、Nitro Hypervisor によって署名されます。これには、Amazon EC2 インスタンスに関連する一連のプラットフォーム設定レジスタ (PCR) 値が含まれます。アテステーションドキュメントには、次の PCR が含まれています。

> **Important**
>
> PCR0 と PCR1 は通常、AWS によって制御される初期ブートコードを測定するために使用されます。初期ブートコードを安全に更新できるように、これらの PCR の値は常に一定です。

- PCR0 – コアシステムファームウェア実行可能コード
- PCR1 – コアシステムファームウェアデータ
- PCR2 – 拡張またはプラグ可能な実行可能コード
- PCR3 – 拡張またはプラグ可能なファームウェアデータ
- PCR4 – ブートマネージャーコード
- PCR5 – ブートマネージャーコード設定とデータおよび GPT パーティションテーブル
- PCR6 – ホストプラットフォームの製造元の詳細

- PCR7 – セキュアブートポリシー
- PCR8 - 15 – 静的オペレーティングシステムで使用するために定義
- PCR16 – デバッグ
- PCR23 – アプリケーションのサポート

PCR4、PCR7、PCR12 は、構成証明可能 AMI を使用してインスタンスが起動されたことを検証するために特別に使用されます。PCR4 および PCR12 は標準ブートで検証するために使用でき、PCR7 は安全ブートで検証するために使用できます。

- PCR4 (ブートマネージャーコード) – インスタンスが起動すると、NitroTPM は UEFI 環境によって実行されるすべてのバイナリの暗号化ハッシュを作成します。構成証明可能 AMI では、これらのブートバイナリはハッシュを埋め込み、一致するハッシュを持たないバイナリの今後のロードを防ぎます。このようにして、単一のブートバイナリハッシュは、インスタンスが今後実行するコードを正確に記述できます。
- PCR7 (セキュアブートポリシー) – UEFI ブートバイナリは、UEFI セキュアブート署名キーで署名できます。UEFI セキュアブートが有効になっている場合、UEFI は設定されたポリシーと一致しない UEFI ブートバイナリの実行を禁止します。PCR7 には、インスタンスの UEFI セキュアブートポリシーのハッシュが含まれています。

  インスタンスの更新全体で保持される単一の KMS ポリシーを維持する必要がある場合は、PCR7 に対して検証して UEFI セキュアブート証明書を検証するポリシーを作成できます。そうすると、構成証明可能 AMI の作成時に、証明書を使用してブートバイナリに署名し、AMI の UEFI データで許可された唯一の証明書としてインストールできます。このモデルでは、古い (信頼済でない) AMI から起動されたインスタンスが KMS ポリシーを通過しないようにするには、新しい証明書を生成し、ポリシーにインストールして AMI を更新する必要があることに注意してください。
- PCR12 — UEFI ブートバイナリに渡されるコマンドラインのハッシュを含みます。コマンドラインが変更されていないことを検証するために、標準ブートにおいて PCR4 と組み合わせることが必要です。

NitroTPM アテステーションドキュメントを検証する

> **Note**
>
> このトピックは、サードパーティーのキー管理サービスを使用していて、独自のアテステーションドキュメント検証メカニズムを構築する必要があるユーザーを対象としています。

このトピックでは、NitroTPM アテステーションフロー全体の概要を詳しく説明します。また、アテステーションドキュメントがリクエストされたときに AWS Nitro システムによって生成される内容と、キー管理サービスがアテステーションドキュメントを処理する方法について説明します。

トピック

- アテステーションドキュメント
- アテステーションドキュメントの検証

アテステーションの目的は、実行中のコードと設定に基づいて、インスタンスが信頼できるエンティティであることを証明することです。インスタンスの信頼の根源は、アテステーションドキュメントを提供する AWS Nitro システム内にあります。

アテステーションドキュメントは AWS Nitro Attestation Public Key Infrastructure (PKI) によって署名されます。PKI には、任意のサービスに組み込むことができる公開認証機関が含まれます。

アテステーションドキュメント

アテステーションドキュメントは Concise Binary Object Representation (CBOR) でエンコードされ、CBOR Object Signing and Encryption (COSE) を使用して署名されます。

CBOR の詳細については、「RFC 8949: Concise Binary Object Representation (CBOR)」を参照してください。

アテステーションドキュメントの仕様

以下は、アテステーションドキュメントの構造を示しています。

```
AttestationDocument = {
    module_id: text,                    ; issuing Nitro hypervisor module ID
    timestamp: uint .size 8,            ; UTC time when document was created, in
                                        ; milliseconds since UNIX epoch
    digest: digest,                     ; the digest function used for calculating the
                                        ; register values
    nitrotpm_pcrs: { + index => pcr },  ; map of PCRs at the moment the Attestation
 Document was generated
    certificate: cert,                  ; the public key certificate for the public
 key
                                        ; that was used to sign the Attestation
 Document
    cabundle: [* cert],                 ; issuing CA bundle for infrastructure
 certificate
```

```
    ? public_key: user_data,             ; an optional DER-encoded key the attestation
                                         ; consumer can use to encrypt data with
    ? user_data: user_data,              ; additional signed user data, defined by
 protocol
    ? nonce: user_data,                  ; an optional cryptographic nonce provided by
 the
                                         ; attestation consumer as a proof of
 authenticity
}

cert = bytes .size (1..1024)       ; DER encoded certificate
user_data = bytes .size (0..1024)
pcr = bytes .size (32/48/64)       ; PCR content
index = 0..31
digest = "SHA384"
```

アテステーションドキュメントのオプションパラメータ (public_key、user_data、および nonce) を使用して、アテステーションを行うインスタンスと外部サービスの間にカスタム検証プロトコルを確立できます。

アテステーションドキュメントの検証

Nitro Hypervisor からアテステーションドキュメントをリクエストすると、署名されたアテステーションドキュメントを含むバイナリ BLOB を受け取ります。署名付きアテステーションドキュメントは、CBOR エンコードされ、 COSE 署名 (COSE_Sign1 署名構造を使用) されたオブジェクトです。全体的な検証プロセスには、次のステップが含まれます。

1. CBOR オブジェクトをデコードし、COSE_Sign1 構造にマッピングします。
2. COSE_Sign1 構造からアテステーションドキュメントを抽出します。
3. 証明書のチェーンを確認します。
4. アテステーションドキュメントが正しく署名されていることを確認します。

アテステーションドキュメントは、商用 AWS パーティションのルート証明書を含む AWS Nitro Attestation PKI によって署名されます。ルート証明書は https://aws-nitro-enclaves.amazonaws.com/AWS_NitroEnclaves_Root-G1.zip からダウンロードでき、次のフィンガープリントを使用して検証できます。

```
64:1A:03:21:A3:E2:44:EF:E4:56:46:31:95:D6:06:31:7E:D7:CD:CC:3C:17:56:E0:98:93:F3:C6:8F:79:BB:5B
```

ルート証明書は AWS Certificate Manager Private Certificate Authority (AWS Private CA) プライベートキーに基づいており、有効期間は 30 年です。PCA のサブジェクトの形式は次のとおりです。

```
CN=aws.nitro-enclaves, C=US, O=Amazon, OU=AWS
```

トピック

- COSE と CBOR
- セマンティックの有効性
- 証明書の有効性
- 証明書チェーンの有効性

COSE と CBOR

通常、COSE_Sign1 署名構造は、1 つの署名のみがメッセージに配置される場合に使用されます。コンテンツと署名を処理するパラメータは、COSE_Sign を分離するのではなく、保護されたヘッダーに配置されます。構造は、使用するコンテキストに応じて、タグ付けまたはタグ無しのいずれかでエンコードできます。タグ付きの COSE_Sign1 構造は、CBOR タグ 18 によって識別されます。

本文、署名、および本文と署名に関する情報を保持する CBOR オブジェクトは、COSE_Sign1 構造と呼ばれます。COSE_Sign1 構造は CBOR 配列です。配列には、次のフィールドが含まれます。

```
[
  protected:   Header,
  unprotected: Header,
  payload:     This field contains the serialized content to be signed,
  signature:   This field contains the computed signature value.
]
```

アテステーションドキュメントのコンテキストでは、配列には以下が含まれます。

```
18(/* COSE_Sign1 CBOR tag is 18 */
    {1: -35}, /* This is equivalent with {algorithm: ECDS 384} */
    {}, /* We have nothing in unprotected */
    $ATTESTATION_DOCUMENT_CONTENT /* Attestation Document */,
    signature /* This is the signature */
)
```

CBOR の詳細については、「RFC 8949: Concise Binary Object Representation (CBOR)」を参照してください。

セマンティックの有効性

アテステーションドキュメントには、常に次の順序で CA バンドルがあります。

```
[ ROOT_CERT - INTERM_1 - INTERM_2 .... - INTERM_N]
      0          1          2             N - 1
```

この順序は、「Java PKI プログラマーズガイド」の Java の CertPath など、既存のツールによっては順序が異なる必要がある場合があることに注意してください。

証明書を検証するには、アテステーションドキュメント CA バンドルから開始し、必要なチェーンを生成します。ここで、TARGET_CERT はアテステーションドキュメントの証明書です。

```
[TARGET_CERT, INTERM_N, ..... , INTERM_2, INTERM_1, ROOT_CERT]
```

証明書の有効性

チェーン内のすべての証明書について、現在の日付が証明書で指定された有効期間内であることを確認する必要があります。

証明書チェーンの有効性

一般的に、ある CA によって署名されたパブリックキー所有者の証明書と、他の CA によって署名された CA の追加の証明書 (無くてもよく、複数でもよい) で構成される、複数の証明書のチェーンが必要になる場合があります。パブリックキーユーザーは限られた数の保証された CA パブリックキーでのみ初期化されるため、認証パスと呼ばれるこのようなチェーンが必要です。インターネット PKI の認証パスの検証手順は、X.509 で提供されているアルゴリズムに基づいています。認証パス処理は、サブジェクト識別名および/またはサブジェクト代替名と、サブジェクトパブリックキーとの間のバインドを検証します。バインドは、そのパスを構成する証明書で指定された制約と、証明書利用者によって指定された入力によって制限されます。基本的な制約とポリシー制約の拡張により、認証パス処理ロジックで意思決定プロセスを自動化できます。

> **Note**
>
> 検証を実行するときは CRL を無効にする必要があります。

ルートパスと生成された証明書チェーンから Java の使用を開始する場合、チェーンの検証は次のようになります。

```
validateCertsPath(certChain, rootCertficate) {
    /* The trust anchor is the root CA to trust */
    trustAnchors.add(rootCertificate);

    /* We need PKIX parameters to specify the trust anchors
     * and disable the CRL validation
     */
    validationParameters = new PKIXParameters(trustAnchors);
    certPathValidator = CertPathValidator.getInstance(PKIX);
    validationParameters.setRevocationEnabled(false);

    /* We are ensuring that certificates are chained correctly */
    certPathValidator.validate(certPath, validationParameters);
}
```

AWS KMS との統合

インスタンスには、NitroTPM から取得したアテステーションドキュメントを使用して AWS KMS API リクエストを行うことができるアプリケーションが必要です。アテステーションドキュメントを使用してリクエストを行うと、AWS KMS は提供されたアテステーションドキュメントの測定値を KMS キーポリシーのリファレンス測定値と照合します。リクエストは、アテステーションドキュメントの測定値が KMS キーポリシーのリファレンス測定値と一致する場合にのみ許可されます。

アテステーションドキュメントを使用して Decrypt、DeriveSharedSecret、GenerateDataKey、GenerateDataKeyPair、または GenerateRandom の API オペレーションを呼び出す場合、これらの API は、アテステーションドキュメント由来のパブリックキーのレスポンス内にあるプレーンテキストを暗号化し、プレーンテキストの代わりに暗号文を返します。この暗号文は、インスタンス内に生成された対応するプライベートキーを使用してのみ復号できます。

詳細については、「AWS Key Management Service デベロッパーガイド」の「NitroTPM の暗号化アテステーション」を参照してください。

> **Note**
>
> サードパーティーのサービスに対してアテステーションを行う場合は、アテステーションドキュメントを受信、解析、検証するための独自のカスタムメカニズムを構築する必要があります。詳細については、「NitroTPM アテステーションドキュメントを検証する」を参照してください。

自社のオペレーターからデータを分離する

AWS Nitro System には、ゼロオペレーターがアクセスがあります。AWS システムまたはユーザーが Amazon EC2 Nitro ホストにログインしたり、EC2 インスタンスのメモリにアクセスしたり、ローカルの暗号化されたインスタンスストレージまたはリモートの暗号化された Amazon EBS ボリュームに保存されている顧客データにアクセスしたりするためのメカニズムはありません。

高度の機密データを処理するときは、自社のオペレーターでも EC2 インスタンスにアクセスできないようにして、そのデータへのアクセスを制限することを検討してください。

分離されたコンピューティング環境を提供するように設定されたカスタム構成証明可能 AMI を作成できます。AMI の設定は、ワークロードとアプリケーションの要件によって異なります。AMI を構築して分離されたコンピューティング環境を作成するときは、以下のベストプラクティスを考慮してください。

- オペレーターまたはユーザーがインスタンスにアクセスできないように、すべてのインタラクティブアクセスを削除します。
- 信頼できるソフトウェアとコードのみが AMI に含まれているようにします。
- アクセスをブロックするようにインスタンス内のネットワークファイアウォールを設定します。
- すべてのストレージおよびファイルシステムを読み取り専用かつイミュータブルな状態にします。
- 認証され、認可され、ログに記録された API コールにインスタンスへのアクセスを制限します。

インタラクティブアクセスを持たない構成証明可能 AMI の更新

分離されたコンピューティング環境 AMI を使用してインスタンスを起動すると、ユーザーやオペレーターがインスタンスに接続できなくなります。つまり、起動後にインスタンスにソフトウェアをインストールまたは更新する方法はありません。

新しいソフトウェアまたはソフトウェア更新が必要な場合は、必要なソフトウェアまたはソフトウェア更新を含む新しい構成証明可能 AMI を作成する必要があります。次に、その AMI を使用して新しいインスタンスを起動するか、元のインスタンスでルートボリュームの置換を実行します。AMI にソフトウェアの変更が適用されると、新しいハッシュが生成されます。

次のアクションを実行すると、NitroTPM アテステーションドキュメントのリファレンス測定値が変更されます。

- 構成証明可能 AMI で起動されたインスタンスの停止と起動
- 別の AMI でルートボリュームの置換を実行する

これらのアクションを実行する場合は、新しいリファレンス測定値でアテステーションサービスを更新する必要があります。例えば、アテステーションに AWS KMS を使用している場合は、KMS キーポリシーを新しいリファレンス測定値に更新する必要があります。

インスタンスは、インスタンスライフサイクル全体で NitroTPM キーマテリアルを保持し、停止/起動およびルートボリューム置換オペレーションを通じて保持します。

Windows インスタンスの クレデンシャルガード

AWS Nitro System は Amazon Elastic Compute Cloud (Amazon EC2) Windows インスタンスの クレデンシャルガード をサポートしています。クレデンシャルガード は Windows 仮想化ベースのセキュリティ (VBS) 機能です。これにより、Windows カーネルの保護だけでなく、Windows ユーザー認証情報やコードインテグリティの適用などのセキュリティ資産を保護するための隔離環境を構築できます。EC2 Windows インスタンスを実行すると、クレデンシャルガード は AWS Nitro System を使用して、Windows ログイン認証情報がオペレーティングシステムのメモリから抽出されないように保護します。

内容

- 前提条件
- サポートされているインスタンスを起動する
- メモリインテグリティの無効化
- クレデンシャルガード を有効にする
- クレデンシャルガード が実行されていることを確認する

前提条件

クレデンシャルガード を使用するにはWindows インスタンスが 次の前提条件を満たす必要があります。

Amazon マシンイメージ (AMI)

NitroTPM と UEFI 安全ブート を有効にするにはAMI を事前に設定しておく必要があります。サポートされている AMI の詳細についてはthe section called “要件”を参照してください。

メモリインテグリティ

ハイパーバイザー保護コードインテグリティ (HVCI) またはハイパーバイザー強制コードインテグリティとも呼ばれるメモリーインテグリティはサポートされていません。認証情報ガードを有

効にする前に、この機能が無効になっていることを確認する必要があります。詳細についてはメモリインテグリティの無効化を参照してください。

インスタンスタイプ

次のインスタンスタイプは特段の記載がない限り、すべてのサイズで クレデンシャルガード をサポートしています: C5、C5d、C5n、C6i、C6id、C6in、C7i、C7i-flex、M5、M5d、M5dn、M5n、M5zn、M6i、M6id、M6idn、M6in、M7i、M7i-flex、R5、R5b、R5d、R5dn、R5n、R6i、R6id、R6idn、R6in、R7i、R7iz、T3。

> **Note**
>
> - NitroTPM には共通して必須インスタンスタイプがいくつかありますが、クレデンシャルガード をサポートするにはインスタンスタイプが上記のインスタンスタイプのいずれかである必要があります。
> - クレデンシャルガード は以下ではサポートされていません。
>   - ベアメタルインスタンス。
>   - 次のインスタンスタイプ: C7i.48xlarge、M7i.48xlarge、R7i.48xlarge。

インスタンスタイプの詳細については、「Amazon EC2 インスタンスタイプガイド」を参照してください。

サポートされているインスタンスを起動する

Amazon EC2 コンソールまたは AWS Command Line Interface (AWS CLI) を使用して、クレデンシャルガード をサポートしているインスタンスを起動できます。インスタンスを起動するにはAWS リージョン ごとに固有の互換性のある AMI ID が必要です。

> **Tip**
>
> 以下のリンクを使用すると、Amazon EC2 コンソールで Amazon が提供する AMI と互換性のあるインスタンスを検出して起動できます。
> https://console.aws.amazon.com/ec2/v2/home?#Images:visibility=public-images;v=3;search=:TPM-Windows_Server;ownerAlias=amazon

Console

インスタンスを起動するには

サポートされているインスタンスタイプと事前設定された Windows AMI を指定し、ステップに従ってインスタンスを起動します。

AWS CLI

インスタンスを起動するには

run-instances コマンドを使用して、サポートされているインスタンスタイプと事前設定された Windows AMI を使用してインスタンスを起動します。

```
aws ec2 run-instances \
    --image-id resolve:ssm:/aws/service/ami-windows-latest/TPM-Windows_Server-2022-English-Full-Base \
    --instance-type c6i.large \
    --region us-east-1 \
    --subnet-id subnet-0abcdef1234567890
    --key-name key-name
```

PowerShell

インスタンスを起動するには

New-EC2Instance コマンドを使用して、サポートされているインスタンスタイプと事前設定された Windows AMI を使用してインスタンスを起動します。

```
New-EC2Instance `
    -ImageId resolve:ssm:/aws/service/ami-windows-latest/TPM-Windows_Server-2022-English-Full-Base `
    -InstanceType c6i.large `
    -Region us-east-1 `
    -SubnetId subnet-0abcdef1234567890 `
    -KeyName key-name
```

メモリインテグリティの無効化

サポートされているシナリオではローカルグループポリシーエディタを使用してメモリインテグリティを無効にできます。次のガイダンスは[仮想化ベースのコードインテグリティ保護] で各構成設定に適用できます。

- ロックなしで有効 — メモリインテグリティを無効にするには設定を [無効] に変更します。
- UEFI ロックで有効 — メモリインテグリティは UEFI ロックで有効になっています。UEFI ロックで一度有効にすると、メモリの整合性を無効にすることはできません。メモリインテグリティを無効にして新しいインスタンスを作成し、サポートされていないインスタンスが使用されていない場合は終了することをお勧めします。

ローカルグループポリシーエディタでメモリの整合性を無効にするには

1. リモートデスクトッププロトコル (RDP) を使用して、管理者権限を持つユーザーアカウントとしてインスタンスに接続します。詳細についてはthe section called “RDP クライアントを使用して接続する”を参照してください。
2. スタートメニューを開き、**cmd** を検索してコマンドプロンプトを起動します。
3. 次のコマンドを実行して、ローカルグループポリシーエディタ `gpedit.msc` を開きます。
4. ローカルグループポリシーエディタで、[コンピュータの構成]、[管理用テンプレート]、[システム]、[デバイスガード] の順に選択してください。
5. [仮想化ベースのセキュリティを有効にする] を選択してから、[ポリシー設定の編集] を選択してください。
6. [仮想化ベースのコードインテグリティ保護] の設定ドロップダウンを開き、[無効] を選択し、[適用] を選択してください。
7. インスタンスを再起動して、変更を適用します。

クレデンシャルガード を有効にする

サポートされているインスタンスタイプと互換性のある AMI を使用して Windows インスタンスを起動し、メモリの整合性が無効になっていることを確認すると、認証情報ガードを有効にできます。

> **Important**
>
> 次の手順を実行して クレデンシャルガード を有効にするには管理者権限が必要です。

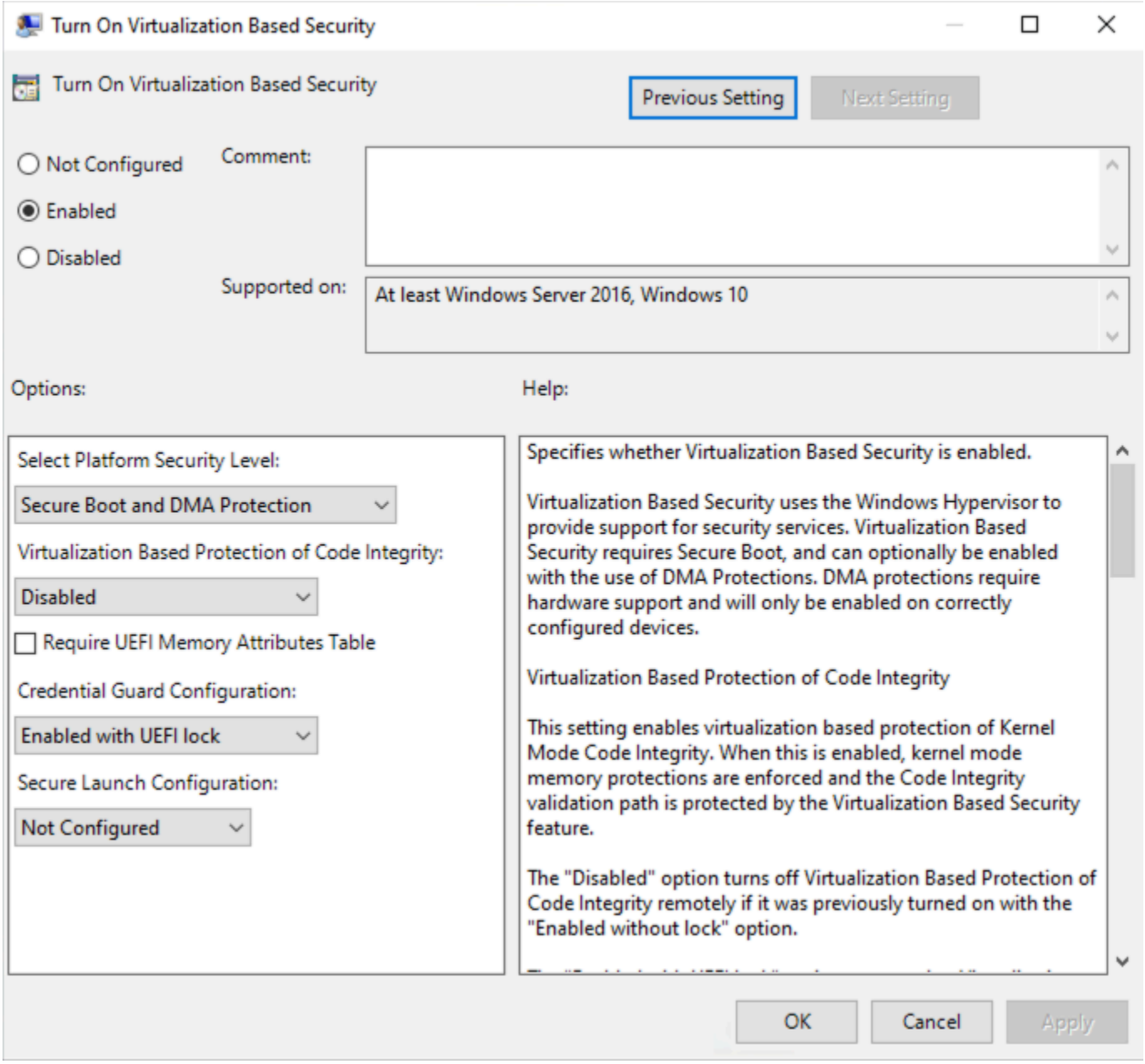
クレデンシャルガード を有効にするには

1. リモートデスクトッププロトコル (RDP) を使用して、管理者権限を持つユーザーアカウントとしてインスタンスに接続します。詳細についてはthe section called “RDP クライアントを使用して接続する”を参照してください。

2. スタートメニューを開き、**cmd** を検索してコマンドプロンプトを起動します。
3. 次のコマンドを実行して、ローカルグループポリシーエディタ `gpedit.msc` を開きます。
4. ローカルグループポリシーエディタで、[コンピュータの構成]、[管理用テンプレート]、[システム]、[デバイスガード] の順に選択してください。
5. [仮想化ベースのセキュリティを有効にする] を選択してから、[ポリシー設定の編集] を選択してください。
6. [仮想化ベースのセキュリティを有効にする] メニューで [有効] を選択してください。
7. [プラットフォームセキュリティレベルの選択] では[安全ブート と DMA 保護] を選択してください。
8. [クレデンシャルガード の設定] では[UEFI ロックで有効] を選択してください。

> **Note**
>
> 残りのポリシー設定は クレデンシャルガード を有効にするために必要ないため、[未構成] のままにしておくことができます。

以下の画像は前述のように構成された VBS 設定を示しています。



9. インスタンスを再起動して、設定を適用します。

クレデンシャルガード が実行されていることを確認する

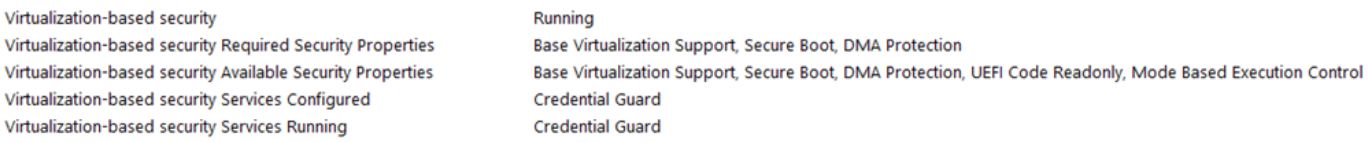
Microsoft システム情報 (`Msinfo32.exe`) ツールを使用して、クレデンシャルガード が実行されていることを確認できます。

> **Important**
>
> クレデンシャルガード を有効にするために必要なポリシー設定の適用を完了するにはまずインスタンスを再起動する必要があります。

クレデンシャルガード が実行されていることの確認するには

1. リモートデスクトッププロトコル (RDP) を使用してインスタンスに接続します。詳細についてはthe section called “RDP クライアントを使用して接続する”を参照してください。
2. インスタンス用の RDP セッションで、スタートメニューを開いて、**cmd** を検索して、コマンドプロンプトを開始します。
3. 次のコマンドを実行して、システム情報を開きます: `msinfo32.exe`
4. Microsoft システム情報ツールにはVBS 設定の詳細が一覧表示されます。仮想化ベースのセキュリティサービスの横に、[クレデンシャルガード] が [実行中] と表示されていることを確認します。

   次の画像はVBS が前述のように実行されていることを示しています。



インターフェイス VPC エンドポイントを使用して Amazon EC2 にアクセスします。

VPC 内のリソースと Amazon EC2 API の間にプライベート接続を作成することで、VPC のセキュリティ体制を向上させることができます。インターネットゲートウェイ、NAT デバイス、VPN 接続、または Direct Connect 接続を使用せずに、VPC 内にあるかのように Amazon EC2 API にアクセスできます。VPC 内の EC2 インスタンスは、Amazon EC2 API にアクセスするためにパブリック IP アドレスを必要としません。

詳細については「AWS PrivateLink ガイド」の「Access AWS のサービス through AWS PrivateLink」を参照してください。

目次

- インターフェイス VPC エンドポイントを作成する

- エンドポイントポリシーを作成する

インターフェイス VPC エンドポイントを作成する

以下のサービス名を使用して、Amazon EC2 のインターフェイス エンドポイントを作成します。

- com.amazonaws.*region*.ec2 — Amazon EC2 API アクションのエンドポイントを作成します。

詳細については、「AWS PrivateLink ガイド」の「インターフェイス VPC エンドポイントを使用して AWS のサービス にアクセスする」を参照してください。

エンドポイントポリシーを作成する

エンドポイントポリシーはインターフェイスエンドポイントにアタッチできる IAM リソースです。デフォルトのエンドポイントポリシーではインターフェイスエンドポイント経由での Amazon EC2 API へのフルアクセスが許可されています。VPC から Amazon EC2 API へのアクセス許可をコントロールするにはカスタムエンドポイントポリシーをインターフェイスエンドポイントにアタッチします。

エンドポイントポリシーは以下の情報を指定します。

- アクションを実行できるプリンシパル。
- 実行可能なアクション。
- このアクションを実行できるリソース。

> **Important**
>
> デフォルト以外のポリシーが Amazon EC2 のインターフェイス VPC エンドポイントに適用されると、`RequestLimitExceeded` からの失敗したリクエストなど、特定の失敗した API リクエストが AWS CloudTrail または Amazon CloudWatch に ログ記録されない場合があります。

詳細については、「AWS PrivateLink ガイド」の「Control access to services using endpoint policies」を参照してください。

次の例は暗号化されていないボリュームを作成したり、暗号化されていないボリュームでインスタンスを起動したりするアクセス許可を拒否する VPC エンドポイントポリシーを示しています。このポリシー例では他のすべての Amazon EC2 のアクションを実行するアクセス許可も付与しています。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
    {
        "Action": "ec2:*",
        "Effect": "Allow",
        "Resource": "*",
        "Principal": "*"
    },
    {
        "Action": [
            "ec2:CreateVolume"
        ],
        "Effect": "Deny",
        "Resource": "*",
        "Principal": "*",
        "Condition": {
            "Bool": {
                "ec2:Encrypted": "false"
            }
        }
    },
    {
        "Action": [
            "ec2:RunInstances"
        ],
        "Effect": "Deny",
        "Resource": "*",
        "Principal": "*",
        "Condition": {
            "Bool": {
                "ec2:Encrypted": "false"
            }
        }
    }]
}
```

Amazon EC2 インスタンスのストレージオプション

Amazon EC2 にはインスタンスを格納するための、柔軟で使いやすく、コスト効率の良いデータストレージオプションが用意されています。各オプションは独自のパフォーマンスと耐久性を備えています。これらのストレージオプションは、要件に応じて個別に使用することも、組み合わせて使用することもできます。

ブロックストレージ

- Amazon EBS: Amazon EBS は、インスタンスにアタッチまたはデタッチできる、耐久性の高いブロックレベルストレージボリュームを提供します。複数の EBS ボリュームを 1 つのインスタンスにアタッチできます。EBS ボリュームは、関連するインスタンスの有効期間とは無関係に存続します。EBS ボリュームは暗号化できます。データのバックアップコピーを保持するには、EBS ボリュームからスナップショットを作成します。スナップショットは Amazon S3 に保存されます。スナップショットから EBS ボリュームを作成できます。
- EC2 インスタンス用のインスタンスストアの一時ブロックストレージ: インスタンスストアは、インスタンス用のブロックレベルの一時ストレージを提供します。インスタンスストアボリュームの数、サイズ、タイプは、インスタンスタイプとインスタンスサイズによって決まります。インスタンスストアボリュームのデータは、関連するインスタンスの存続中にのみ保持されます。インスタンスを停止、休止、または終了すると、インスタンスストアボリュームのすべてのデータが失われます。

オブジェクトストレージ

- Amazon S3: Amazon S3 により、低コストで信頼性に優れたデータストレージインフラストラクチャが実現します。ウェブスケールのコンピューティングをさらに簡単に行えるように設計されており、Amazon EC2 内から、またはウェブ上のどこからでも、いつでも必要な量だけデータを格納および取得できます。例えば、Amazon S3 を使用して、データとアプリケーションのバックアップコピーを保存することができます。Amazon EC2 は、Amazon S3 を使用して EBS スナップショットと Amazon S3-backed AMI を保存します。

ファイルストレージ

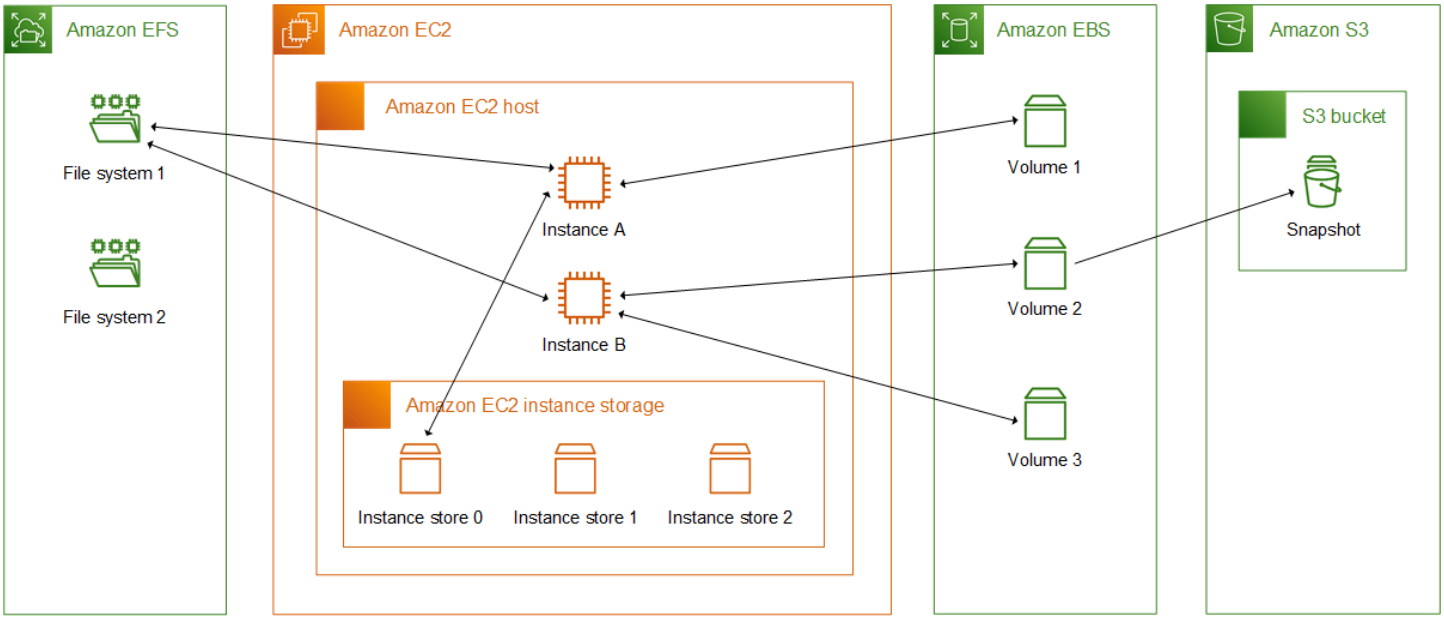
- Amazon EC2 EFS (Linux インスタンスのみ): Amazon EFS は、Amazon EC2 と併用できるスケーラブルなファイルストレージを提供します。EFS ファイルシステムを作成し、ファイルシステムをマウントするためにインスタンスを設定できます。複数のインスタンスで実行している作業負荷やアプリケーションの一般的なデータソースとしてEFSファイルシステムを使用できます。
- Amazon FSx: Amazon FSx を使用すると、多機能で高性能なファイルシステムをクラウドで起動、実行、およびスケールできます。Amazon FSx は、広範なワークロードをサポートするフ

ルマネージド型サービスです。Lustre、NetApp ONTAP、OpenZFS、Windows File Server など、広く使用されているファイルシステムから選択できます。

ファイルキャッシュ

- Amazon EC2 インスタンスでの Amazon File Cache の使用: Amazon File Cache は、AWS 上にファイルデータを処理するための一時的な高性能キャッシュを提供します。キャッシュは、ミリ秒未満のレイテンシー、最大数百 GB/秒のスループット、最大数百万の IOPS で、Amazon EC2 でのコンピューティングワークロードへの読み取りおよび書き込みデータアクセスを提供します。

各ストレージオプションとインスタンスの関係を下の図に示します。



AWS ストレージ料金表

[AWS の料金] を開き、[AWS 製品の料金] までスクロールし、[ストレージ] を選択します。ストレージ製品を選択して、料金ページを開きます。

Amazon EC2 インスタンス用の Amazon EBS の永続的なブロックストレージ

Amazon Elastic Block Store (Amazon EBS) はAmazon EC2 インスタンスで使用できるスケーラブルかつ高性能なブロックストレージリソースを提供します。Amazon EBS では次のブロックストレージリソースを作成および管理できます。

- Amazon EBS ボリューム - Amazon EC2 インスタンスにアタッチするストレージボリュームです。ボリュームをインスタンスにアタッチした後はブロックストレージを使用するのと同じ方法で使用できます。インスタンスはローカルドライブと同じようにボリュームとやり取りできます。
- Amazon EBS スナップショット - ボリューム自体とは独立して保持される Amazon EBS ボリュームのポイントインタイムバックアップです。スナップショットを作成して、Amazon EBS ボリュームのデータをバックアップできます。その後、それらのスナップショットからいつでも新しいボリュームを復元できます。

起動時に EBS ボリュームを作成してインスタンスにアタッチできます。また、起動後もいつでも EBS ボリュームを作成してインスタンスにアタッチできます。ボリュームをデタッチしたり、インスタンスを再起動したりする必要なく、EBS ボリュームのサイズやパフォーマンスを向上させることもできます。

EBS ボリュームの作成後はいつでもその EBS ボリュームから EBS スナップショットを作成できます。EBS スナップショットを使用して、ボリュームに保存されているデータをバックアップできます。その後、スナップショットを使用して、ボリュームを瞬時に復元したり、AWS アカウント、AWS リージョン、またはアベイラビリティーゾーンの間でデータを移行したりできます。Amazon Data Lifecycle Manager または AWS Backup を使用して、EBS スナップショットの作成、保持、削除を自動化できます。

マネージド EBS ボリュームは、Amazon EKS Auto Mode などのサービスプロバイダーによって管理されます。管理された EBS ボリュームの設定を直接変更することはできません。管理された EBS ボリュームは 管理されたフィールドの 正 値によって識別されます。詳細については「Amazon EC2 マネージドインスタンス」を参照してください。

ボリュームとスナップショットの使用方法の詳細については「Amazon EBS ユーザーガイド」を参照してください。

Amazon EC2 インスタンスの Amazon EBS ボリューム制限

インスタンスにアタッチできる Amazon EBS ボリュームの最大数はインスタンスのタイプとサイズによって異なります。インスタンスのボリュームアタッチメント制限を超えると、アタッチメントリクエストは `AttachmentLimitExceeded` エラーで失敗します。

インスタンスに追加するボリューム数を検討する際はI/O 帯域幅とストレージ容量のどちらを増加するのかを、考慮する必要があります。

帯域幅と容量

整合性がとれており予測可能な帯域幅のユースケースではAmazon EBS 最適化インスタンスと、汎用 SSD ボリュームまたはプロビジョンド IOPS SSD ボリュームを使用します。パフォーマンスを最大化するためにはボリュームに対してプロビジョニングした IOPS を、使用しているインスタンスで利用可能な帯域幅と一致させます。

RAID 構成ではI/O オーバーヘッドの増加が原因となり、8 つのボリュームよりも大きなアレイがパフォーマンスを低下させることがあります。個々のアプリケーションのパフォーマンスをテストし、必要に応じて調整してください。

目次

- Nitro システム上に構築されたインスタンスにおけるボリューム制限
  - 専用 EBS ボリュームの制限
  - 共有 EBS ボリュームの制限
- Xen ベースのインスタンスのボリューム制限
  - Linux インスタンス
  - Windows インスタンス

Nitro システム上に構築されたインスタンスにおけるボリューム制限

Nitro System 上に構築されたインスタンスのボリュームの制限はインスタンスタイプによって異なります。一部の Nitro インスタンスタイプには専用 EBS ボリュームの制限があり、ほとんどのインスタンスタイプには共有ボリュームの制限があります。各インスタンスタイプのボリュームアタッチメント制限を表示するには、以下を参照してください。

- Amazon EBS の仕様 — 汎用
- Amazon EBS の仕様 — コンピューティング最適化
- Amazon EBS の仕様 — メモリ最適化
- Amazon EBS の仕様 — ストレージ最適化
- Amazon EBS の仕様 — アクセラレーテッドコンピューティング
- Amazon EBS の仕様 – ハイパフォーマンスコンピューティング
- Amazon EBS の仕様 — 旧世代

専用 EBS ボリュームの制限

以下の Nitro インスタンスタイプにはインスタンスサイズに応じて異なる専用 EBS ボリュームの制限があります。この制限は他のデバイスアタッチメントとは共有されません。つまり、NVMe インスタンスストアボリュームやネットワークインターフェイスなど、アタッチされているデバイスの数に関係なく、ボリュームアタッチメントの上限まで任意の数の EBS ボリュームをアタッチできます。

- 汎用: M7a | M7i | M7i-flex | M8a | M8azn | M8g | M8gb | M8gd | M8gn | M8i | M8id | M8i-flex | M8in | M8idn | M8ine | M8ib | M8idb | M9g | M9gd
- コンピューティング最適化: C7a | C7i | C7i-flex | C8a | C8g | C8gb | C8gd | C8gn | C8i | C8id | C8i-flex | C8in | C8ine | C8ib | C9g | C9gd
- メモリ最適化: R7a | R7i | R7iz | R8a | R8g | R8gb | R8gd | R8gn | R8i | R8id | R8i-flex | R8in |R8idn | R8ib | R8idb |U7i-6tb | U7i-8tb | U7i-12tb | U7in-16tb | U7in-24tb | U7in-32tb | U7inh-32tb | X8g | X8aedz | X8i
- ストレージの最適化: I7i | I7ie | I8g | I8ge
- 高速コンピューティング: F2 | G6 | G6e | G6f | Gr6 | Gr6f | G7 | G7e | P4d | P4de | P5 | P5e | P5en | P6-B200 | P6-B300 | P6e-GB200 | Trn2 | Trn2u
- ハイパフォーマンスコンピューティング: Hpc7a | Hpc8a

共有 EBS ボリュームの制限

その他のすべての Nitro インスタンスタイプ (専用 EBS ボリュームの制限 に記載されていないもの) には Amazon EBS ボリューム、ネットワークインターフェイス、および NVMe インスタンスストアボリューム間で共有されるボリュームのアタッチ数に制限があります。その制限数から、アタッチ済みのネットワークインターフェイスと NVMe インスタンスストアボリュームの数を差し引いた数を上限として、Amazon EBS ボリュームをいくつでもアタッチできます。すべてのインスタンスには少なくとも 1 つのネットワークインターフェースが必要であり、NVMe インスタンスストアボリュームは起動時に自動的にアタッチされることに注意してください。

ほとんどの Nitro インスタンスでは最大 28 のアタッチメントをサポートしています。次の例はアタッチできる EBS ボリューム数を計算する方法を示しています。

例

- プライマリネットワークインターフェイスのみを持つ `m5.xlarge` インスタンスでは27 個の EBS ボリュームをアタッチできます。

28 (ボリューム) - 1 (ネットワークインターフェイス) = 27

- 2 つの追加ネットワークインターフェイスを持つ `m5.xlarge` インスタンスでは25 個の EBS ボリュームをアタッチできます。

  28 (ボリューム) - 3 (ネットワークインターフェイス) = 25

- 2 つの追加ネットワークインターフェイスを持つ `m5d.xlarge` インスタンスでは24 個の EBS ボリュームをアタッチできます。

  28 (ボリューム) - 3 (ネットワークインターフェイス) - 1 (NVMe インスタンスストアボリューム) = 24

Xen ベースのインスタンスのボリューム制限

Xen ベースのインスタンス (T2) のボリュームの制限はオペレーティングシステムによって異なります。

詳細については「Xen-based instances」を参照してください。

Linux インスタンス

Xen ベースの Linux インスタンスに 40 個を超えるボリュームをアタッチすると、ブートに失敗する可能性があります。この数にはルートボリュームに加え、アタッチされたインスタンスストアボリュームと Amazon EBS ボリュームも含まれます。

多数のボリュームがアタッチされているインスタンスで起動の問題が発生した場合はインスタンスを停止し、起動プロセスに不可欠ではないボリュームをデタッチした上で再起動し、インスタンスの実行後にそれらのボリュームを再度アタッチします。

> **Important**
>
> Xen ベースの Linux インスタンスに 40 より多くのボリュームをアタッチすることはベストエフォートベースでのみサポートされており、動作は保証されていません。

Windows インスタンス

次の表はXen ベース Windows インスタンスでの、使用中のドライバーに基づいたボリュームの制限を示しています。これらの数値にはルートボリュームに加え、任意のアタッチされたインスタンスストアボリュームや Amazon EBS ボリュームが含まれます。

ドライバー	ボリュームの制限
AWS PV	26
Citrix PV	26
Red Hat PV	17

パフォーマンスに問題が発生する可能性が高いため、AWS PV または Citrix PV ドライバーを使用している Xen ベースの Windows インスタンスに対し、26 個を超えるボリュームをアタッチすることはお勧めしません。インスタンスで使用している PV ドライバーの種類を確認する、または Windows インスタンスで Red Hat PV ドライバーから Citrix PV ドライバーにアップグレードするには「the section called “PV ドライバーのアップグレード”」を参照してください。

> **Important**
>
> Xen ベースの Windows インスタンスに次の数よりも多くのボリュームをアタッチすることはベストエフォートベースでのみサポートされており、動作は保証されていません。

デバイス名とボリュームの関係の詳細については「Amazon EC2 Windows インスタンスへのボリュームのアタッチとマッピング方法」を参照してください。

EBS カード

ほとんどのインスタンスタイプは 1 つのネットワークカードをサポートします。複数の EBS カードをサポートするインスタンスタイプは、EBS スループットと IOPS の両方で EBS 最適化パフォーマンスを向上させます。Amazon EC2 インスタンスの最大パフォーマンスは、各 EBS カードに均等に分散されます。例えば、2 つの EBS カードで合計 1,000,000 の IOPS をサポートする EC2 インスタンスでは、各 EBS カードは最大 500,000 IOPS をサポートできます。Amazon EC2 インスタンスでサポートされている Amazon EBS パフォーマンスの詳細については、「Amazon EBS 最適化インスタンスタイプ」を参照してください。

Amazon EBS ボリュームを複数の EBS カードをサポートするインスタンスにアタッチする場合、EBS カードインデックスを指定することによってボリュームの EBS カードを選択できます。ルートボリュームは EBS カードインデックス 0 にアタッチする必要があります。

複数の EBS カードを持つインスタンスタイプ

次のインスタンスタイプは複数の EBS カードをサポートします。インスタンスタイプがサポートする Amazon EBS の数については、「Amazon EC2 インスタンスの Amazon EBS ボリューム制限」を参照してください。

インスタンスタイプ	EBS カードの数
汎用	
m8gb.48xlarge	2
m8gb.metal-48xl	2
m8gn.48xlarge	2
m8gn.metal-48xl	2
m8in.96xlarge	2
m8in.metal-96xl	2
m8idn.96xlarge	2
m8idn.metal-96xl	2
m8ib.96xlarge	2
m8ib.metal-96xl	2
m8idb.96xlarge	2
m8idb.metal-96xl	2
コンピューティング最適化	
c8gb.48xlarge	2
c8gb.metal-48xl	2
c8gn.48xlarge	2

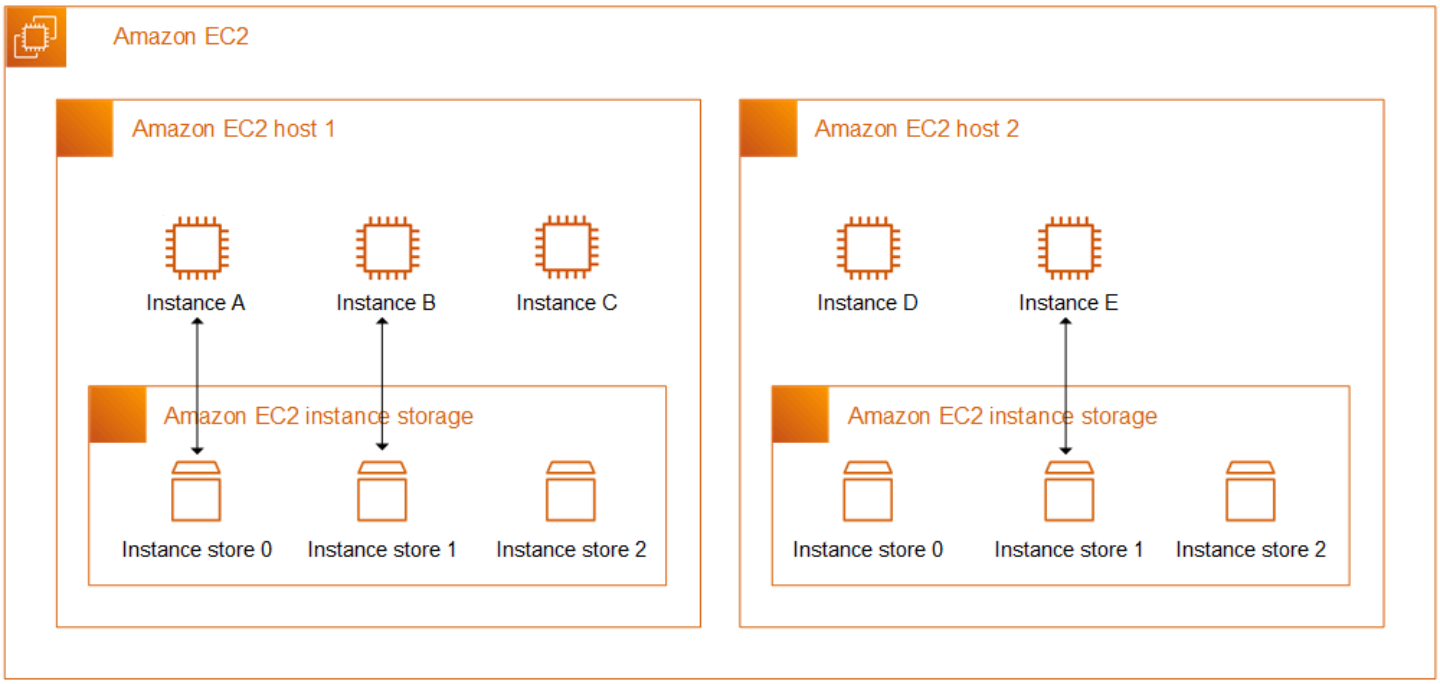
インスタンスタイプ	EBS カードの数
c8gn.metal-48xl	2
c8in.96xlarge	2
c8in.metal-96xl	2
c8ib.96xlarge	2
c8ib.metal-96xl	2
メモリを最適化	
r8gb.48xlarge	2
r8gb.metal-48xl	2
r8gn.48xlarge	2
r8gn.metal-48xl	2
r8in.96xlarge	2
r8in.metal-96xl	2
r8idn.96xlarge	2
8idn.metal-96xl	2
r8ib.96xlarge	2
r8ib.metal-96xl	2
r8idb.96xlarge	2
8idb.metal-96xl	2
高速コンピューティング	
g7.48xlarge	2

EC2 インスタンス用のインスタンスストアの一時ブロックストレージ

インスタンスストアはEC2 インスタンス用に一時的なブロックレベルストレージを提供します。このストレージはホストコンピュータに物理的にアタッチされたディスクによって提供されます。インスタンスストアはバッファ、キャッシュ、スクラッチデータ、その他の一時的データのように頻繁に変化する情報の一時的なストレージに最適です。また、負荷分散されたウェブサーバーのプールなど、インスタンスのフリート全体で複製する一時データを保存するためにも使用できます。

インスタンスストアはブロックデバイスとして表示される 1 つ以上のインスタンスストアボリュームで構成されます。インスタンスストアのサイズと、利用可能なデバイスの数はインスタンスタイプおよびインスタンスサイズによって異なります。例えば、すべてのインスタンスタイプでインスタンスストアボリュームが提供されるわけではありません。詳細については「EC2 インスタンスのインスタンスストアボリュームの制限」を参照してください。

インスタンスストアボリュームの仮想デバイスにはephemeral0 から ephemeral23 の順序で仮想デバイス名が付けられます。例えば、1 つのインスタンスストアボリュームをサポートするインスタンスタイプではその 1 つのボリュームの仮想デバイス名は ephemeral0 になります。4 つのインスタンスストアボリュームをサポートするインスタンスタイプではその 4 つのボリュームの仮想デバイス名は ephemeral0、ephemeral1、ephemeral2、および ephemeral3 になります。



インスタンスストアの価格

インスタンス用に提供されたインスタンスストアボリュームの使用には追加料金はかかりません。インスタンスストアボリュームはインスタンスの使用料の一部として含まれます。

内容

- Amazon EC2 インスタンスストアボリュームのデータの永続性
- EC2 インスタンスのインスタンスストアボリュームの制限
- EC2 インスタンスの SSD インスタンスストアボリューム
- EC2 インスタンスへのインスタンスストアボリュームの追加
- M1 および C1 EC2 インスタンスのインスタンスストアスワップボリュームの有効化
- EC2 インスタンスでのインスタンスストアボリュームの初期化
- Amazon EC2 インスタンスストアボリュームの詳細なパフォーマンス統計

Amazon EC2 インスタンスストアボリュームのデータの永続性

インスタンスストアボリュームはインスタンスの起動時にのみアタッチされます。起動後にインスタンスストアボリュームをアタッチすることはできません。1 つのインスタンスからインスタンスストアをデタッチして別のインスタンスにアタッチすることはできません。

インスタンスストアボリュームはアタッチされているインスタンスのライフタイム中にのみ存在します。インスタンスストアボリュームが、関連付けられたインスタンスのライフタイムを超えて維持されるように設定することはできません。

インスタンスストアボリューム上のデータはインスタンスが再起動しても保持されます。ただし、インスタンスが停止、休止、終了するとデータは消滅します。インスタンスが停止、休止、終了した場合、インスタンスストアボリュームのすべてのブロックが暗号で消去されます。

このため、長期的に使用する重要なデータがある場合はインスタンスストアに頼りすぎないようにしてください。インスタンスストアボリュームに保存されているデータを、インスタンスのライフタイムを超えて保持する必要がある場合はそのデータを Amazon EBS ボリューム、Amazon S3 バケット、Amazon EFS ファイルシステムなどのより永続的なストレージに手動でコピーする必要があります。

イベントによってはインスタンスのライフタイムを通じてデータが保持されなくなる場合があります。次の表は仮想インスタンスとベアメタルインスタンスの両方について、特定のイベント中にインスタンスストアボリュームのデータが保持されるかどうかを示しています。

イベント	データはどうなりますか？
ユーザー主導のインスタンスライフサイクルイベント	
インスタンスが再起動されます。	The data persists
インスタンスが停止しました。	The data does not persist
インスタンスが休止しました。	The data does not persist
インスタンスが終了しました。	The data does not persist
インスタンスタイプが変更されます。	The data does not persist *
EBS-backed AMI はインスタンスから作成されます。	The data does not persist in the created AMI **
Amazon S3-backed AMI はインスタンスから作成されます。 (Linux instances)	The data persists in the AMI bundle uploaded to Amazon S3 ***
ユーザー主導の OS イベント	
A shutdown is initiated	The data does not persist †
A restart is initiated	The data persists
AWS で予定されているイベント	
インスタンスの停止	The data does not persist
インスタンスの再起動	The data persists
システムの再起動	The data persists
インスタンスのリタイア	The data does not persist
想定外のイベント	
簡易自動復旧	The data does not persist
CloudWatch アクションに基づく復旧	The data does not persist

イベント	データはどうなりますか？
The underlying disk fails	The data on the failed disk does not persist
Power failure	The data persists upon reboot

* 新しいインスタンスタイプがインスタンスストアをサポートしている場合、インスタンスは新しいインスタンスタイプがサポートしているインスタンスストアボリュームの数を取得しますが、データは新しいインスタンスに転送されません。新しいインスタンスタイプがインスタンスストアをサポートしていない場合、インスタンスはインスタンスストアボリュームを取得しません。

** データは EBS-backed AMI には含まれず、その AMI から起動されたインスタンスにアタッチされたインスタンスストアボリュームにも含まれません。

*** データは Amazon S3 にアップロードされる AMI バンドルに含まれます。その AMI からインスタンスを起動すると、インスタンスはAMI の作成時に含まれていたデータとともに AMI にバンドルされたインスタンスストアボリュームを取得します。

† 終了保護と停止保護はインスタンスのオペレーティングシステムを通じて開始したシャットダウンの結果、インスタンスが停止または終了することに対してインスタンスを保護しません。インスタンスストアボリュームに保存されたデータはインスタンスの停止イベントと終了イベントの両方で保持されません。

EC2 インスタンスのインスタンスストアボリュームの制限

インスタンスストアボリュームの数、サイズ、タイプはインスタンスタイプによって決まります。C8i、M8i、R8i などの一部のインスタンスタイプはインスタンスストアボリュームをサポートしていませんが、C8id、M8id、R8id などのその他のインスタンスタイプはインスタンスストアボリュームをサポートしています。1 つのインスタンスに、そのインスタンスタイプでサポートされる量を超えるインスタンスストアボリュームをアタッチすることはできません。インスタンスストアボリュームをサポートするインスタンスタイプの場合、インスタンスストアボリュームの数とサイズはインスタンスサイズによって異なります。例えば、`r8id.large` は 1 x 118 GB のインスタンスストアボリュームをサポートし、`r8id.32xlarge` は 2 x 3800 GB のインスタンスストアボリュームをサポートします。

NVMe インスタンスストアボリュームを使用するインスタンスタイプではサポートされているすべてのインスタンスストアボリュームが、起動時に自動的にインスタンスにアタッチされます。C1、C3、M1、M2、M3、R3、D2、H1、I2、X1、X1e など、NVMe 以外のインスタンスストア

ボリュームのインスタンスタイプでは起動時にアタッチするインスタンスストアボリュームのブロックデバイスマッピングを手動で指定する必要があります。次に、インスタンスが起動したら、アタッチされたインスタンスストアボリュームを使用する前に、フォーマットしてマウントする必要があります。インスタンスの起動後にインスタンスストアボリュームをアタッチすることはできません。

インスタンスタイプにはNVMe または SATA ベースのソリッドステートドライブ (SSD) を使用するものと、SATA ベースのハードディスクドライブ (HDD) を使用するものがあります。SSD は極めて低いレイテンシーで高いランダム I/O パフォーマンスを提供しますが、インスタンスの終了時にデータを保持する必要はなく、フォールトトレラントアーキテクチャを活用できます。詳細については「EC2 インスタンスの SSD インスタンスストアボリューム」を参照してください。

NVMe インスタンスストアボリューム、および一部の HDD インスタンスストアボリュームにあるデータはその保存時に暗号化されます。詳細については「Amazon EC2 でのデータ保護」を参照してください。

使用可能なインスタンスストアボリューム

「Amazon EC2 Instance Types ガイド」にはサポートされている各インスタンスタイプで使用できる、インスタンスストアボリュームの数量、サイズ、タイプ、パフォーマンス最適化を記載しています。詳細については次を参照してください:

- インスタンスストア仕様 — 汎用
- インスタンスストア仕様 — コンピューティング最適化
- インスタンスストア仕様 — メモリ最適化
- インスタンスストア仕様 — ストレージ最適化
- インスタンスストア仕様 — 高速コンピューティング
- インスタンスストア仕様 — ハイパフォーマンスコンピューティング
- インスタンスストア仕様 — 旧世代

Console

インスタンスストアボリュームの情報を取得するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンスタイプ]を選択します。
3. Local instance storage = true フィルターを追加します。[ストレージ] 列には、インスタンスタイプの、インスタンスストレージの合計容量が記載されています。

4. (オプション) [設定] アイコンをクリックして、[ストレージディスク数] をオンにします。この列には、インスタンスストアボリュームの数が記載されています。
5. (オプション) フィルターを追加して、対象の、特定のインスタンスタイプをさらに絞り込みます。

AWS CLI

インスタンスストアボリュームの情報を取得するには

describe-instance-types コマンドを使用します。次の例では、インスタンスストアボリュームを持つの R8i インスタンスファミリーの、各インスタンスタイプのインスタンスストレージの合計サイズが表示されています。

```
aws ec2 describe-instance-types \
    --filters "Name=instance-type,Values=r8i*" "Name=instance-storage-
supported,Values=true" \
    --query 'sort_by(InstanceTypes, &InstanceStorageInfo.TotalSizeInGB)[].
{InstanceType:InstanceType,TotalSizeInGB:InstanceStorageInfo.TotalSizeInGB}' \
    --output table
```

以下は出力の例です。

```
-------------------------------------
|        DescribeInstanceTypes       |
+------------------+-----------------+
|   InstanceType   |  TotalSizeInGB  |
+------------------+-----------------+
|  r8id.large      |  118            |
|  r8id.xlarge     |  237            |
|  r8id.2xlarge    |  474            |
|  r8id.4xlarge    |  950            |
|  r8id.8xlarge    |  1900           |
|  r8id.12xlarge   |  2850           |
|  r8id.16xlarge   |  3800           |
|  r8id.24xlarge   |  5700           |
|  r8id.32xlarge   |  7600           |
|  r8id.48xlarge   |  11400          |
|  r8id.metal-48xl |  11400          |
|  r8id.96xlarge   |  22800          |
|  r8id.metal-96xl |  22800          |
+------------------+-----------------+
```

インスタンスタイプのインスタンスストレージの詳細を入手するには

describe-instance-types コマンドを使用します。

```
aws ec2 describe-instance-types \
    --filters "Name=instance-type,Values=r8id.32xlarge" \
    --query 'InstanceTypes[0].InstanceStorageInfo' \
    --output json
```

こちらの出力例には、このインスタンスタイプに 3800 GB の NVMe SSD ボリュームが 2 つあり、合計 7600 GB のインスタンスストレージがあることが示されています。

```
{
    "TotalSizeInGB": 7600,
    "Disks": [
        {
            "SizeInGB": 3800,
            "Count": 2,
            "Type": "ssd"
        }
    ],
    "NvmeSupport": "required",
    "EncryptionSupport": "required"
}
```

PowerShell

インスタンスストアボリュームの情報を取得するには

Get-EC2InstanceType コマンドレットを使用します。次の例では、インスタンスストアボリュームを持つの R8i インスタンスファミリーの、各インスタンスタイプのインスタンスストレージの合計サイズが表示されています。

```
(Get-EC2InstanceType -Filter `
    @{Name="instance-type"; Values="r8i*"},
    @{Name="instance-storage-supported"; Values="true"}) |
    Sort-Object {$_.InstanceStorageInfo.TotalSizeInGB} |
    Format-Table InstanceType,

 @{Name="Disks.SizeInGB";Expression={$_.InstanceStorageInfo.Disks[0].SizeInGB}},
        @{Name="Disks.Count";Expression={$_.InstanceStorageInfo.Disks[0].Count}},
```

```
        @{Name="TotalSizeInGB";Expression={$_.InstanceStorageInfo.TotalSizeInGB}}
```

以下は出力の例です。

```
InstanceType    Disks.SizeInGB Disks.Count TotalSizeInGB
------------    -------------- ----------- -------------
r8id.large                 118           1           118
r8id.xlarge                237           1           237
r8id.2xlarge               474           1           474
r8id.4xlarge               950           1           950
r8id.8xlarge              1900           1          1900
r8id.12xlarge             2850           1          2850
r8id.16xlarge             3800           1          3800
r8id.24xlarge             2850           2          5700
r8id.32xlarge             3800           2          7600
r8id.48xlarge             3800           3         11400
r8id.metal-48xl           3800           3         11400
r8id.96xlarge             3800           6         22800
r8id.metal-96xl           3800           6         22800
```

インスタンスタイプのインスタンスストレージの詳細を入手するには

Get-EC2InstanceType コマンドレットを使用します。

```
(Get-EC2InstanceType `
    -Filter @{Name="instance-type"; Values="r8id.32xlarge"}).InstanceStorageInfo |
    Format-List *,
        @{Name="Disks.Count";Expression={$_.Disks[0].Count}},
        @{Name="Disks.SizeInGB";Expression={$_.Disks[0].SizeInGB}},
        @{Name="Disks.Type";Expression={$_.Disks[0].Type.Value}}
```

こちらの出力例には、このインスタンスタイプに 3800 GB の NVMe SSD ボリュームが 2 つあり、合計 7600 GB のインスタンスストレージがあることが示されています。

```
Disks             : {Amazon.EC2.Model.DiskInfo}
EncryptionSupport : required
NvmeSupport       : required
TotalSizeInGB     : 7600
Disks.Count       : 2
Disks.SizeInGB    : 3800
Disks.Type        : ssd
```

EC2 インスタンスの SSD インスタンスストアボリューム

他のインスタンスストアボリュームと同様に、インスタンスの SSD インスタンスストアボリュームを起動するときにマップする必要があります。SSD インスタンスボリューム上のデータは関連するインスタンスの存続期間中のみ維持されます。詳細については「EC2 インスタンスへのインスタンスストアボリュームの追加」を参照してください。

NVMe SSD ボリューム

インスタンスによってはNon-Volatile Memory Express (NVMe) ソリッドステートドライブ (SSD) インスタンスストアボリュームを提供するものもあります。各インスタンスタイプによりサポートされるインスタンスストアボリュームのタイプの詳細についてはEC2 インスタンスのインスタンスストアボリュームの制限を参照してください。

NVMe インスタンスストレージのデータはインスタンスのハードウェアモジュールに実装されている XTS-AES-256 ブロック暗号を使用して暗号化されます。暗号化キーはハードウェアモジュールで作成され、NVMe インスタンスストレージデバイスごとに固有です。すべての暗号化キーはインスタンスが停止または終了して復元できないときに破棄されます。この暗号化を無効にしたり、独自の暗号キーを指定したりすることはできません。

Linux インスタンス

NVMe ボリュームにアクセスするにはNVMe ドライバーをインストールする必要があります。以下の AMI はこの要件を満たしています。

- AL2023
- Amazon Linux 2
- Amazon Linux AMI 2018.03 以降
- `linux-aws` カーネルを搭載した Ubuntu 14.04 以降

> **Note**
>
> AWS Graviton ベースのインスタンスタイプには`linux-aws` カーネル搭載の Ubuntu 18.04 以降が必要です

- Red Hat Enterprise Linux 7.4 以降
- SUSE Linux Enterprise Server 12 SP2 以降
- CentOS 7.4.1708 以降

- FreeBSD 11.1 以降
- Debian GNU/Linux 9 以降

- Bottlerocket

インスタンスに接続したら、lspci コマンドを使用して NVMe デバイスをリストできます。次に示すのは4 つの NVMe デバイスをサポートする `i3.8xlarge` インスタンスの出力例です。

```
[ec2-user ~]$ lspci
00:00.0 Host bridge: Intel Corporation 440FX - 82441FX PMC [Natoma] (rev 02)
00:01.0 ISA bridge: Intel Corporation 82371SB PIIX3 ISA [Natoma/Triton II]
00:01.1 IDE interface: Intel Corporation 82371SB PIIX3 IDE [Natoma/Triton II]
00:01.3 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 01)
00:02.0 VGA compatible controller: Cirrus Logic GD 5446
00:03.0 Ethernet controller: Device 1d0f:ec20
00:17.0 Non-Volatile memory controller: Device 1d0f:cd01
00:18.0 Non-Volatile memory controller: Device 1d0f:cd01
00:19.0 Non-Volatile memory controller: Device 1d0f:cd01
00:1a.0 Non-Volatile memory controller: Device 1d0f:cd01
00:1f.0 Unassigned class [ff80]: XenSource, Inc. Xen Platform Device (rev 01)
```

サポートされているオペレーティングシステムを使用しているが、NVMe デバイスが表示されない場合は次のコマンドを使用して NVMe モジュールが読み込まれていることを確認します。

- Amazon Linux、Amazon Linux 2、Ubuntu 14/16、Red Hat Enterprise Linux、SUSE Linux Enterprise Server、CentOS 7

  ```
  $ lsmod | grep nvme
  nvme           48813  0
  ```

- Ubuntu 18

  ```
  $ cat /lib/modules/$(uname -r)/modules.builtin | grep nvme
  s/nvme/host/nvme-core.ko
  kernel/drivers/nvme/host/nvme.ko
  kernel/drivers/nvmem/nvmem_core.ko
  ```

NVMe ボリュームは NVMe 1.0e 仕様に準拠しています。NVMe コマンドは NVMe ボリュームで使用できます。Amazon Linux では`nvme-cli` コマンドを使用して repo から yum install パッケージをイ

ンストールできます。サポートされているバージョンの Linux ではイメージで利用可能でない場合は `nvme-cli` パッケージをダウンロードできます。

Windows インスタンス

以下のオペレーティングシステムの最新の AWS Windows AMI にはAWS NVMe ドライバーが含まれています。これらはパフォーマンス向上のために NVMe ブロックデバイスとして公開される SSD インスタンスストアボリュームを操作するために使用します。

- Windows Server 2025
- Windows Server 2022
- Windows Server 2019
- Windows Server 2016
- Windows Server 2012 R2

インスタンスに接続したら、ディスクマネージャーで NVMe ボリュームが表示されることを確認できます。タスクバーで、Windows ロゴのコンテキスト (右クリック) メニューを開き、[Disk Management] を選択してください。

Amazon が提供する AWS Windows AMI にはAWS NVMe ドライバーが含まれています。最新の AWS Windows AMI を使用していない場合は現行の AWS NVMe ドライバーをインストールできます。

非 NVMe SSD ボリューム

C3、I2、M3、R3、X1 の各インスタンスは非 NVMe SSD を使用するインスタンスストアボリュームをサポートすることで、高いランダム I/O パフォーマンスを実現しています。各インスタンスタイプによりサポートされるインスタンスストアボリュームの詳細についてはEC2 インスタンスのインスタンスストアボリュームの制限を参照してください。

SSD ベースのインスタンスストアボリュームの I/O パフォーマンス

インスタンスに SSD ベースのインスタンスストアボリュームを使用するほど、アーカイブできる書き込み IOPS の数は減少します。これはSSD コントローラーが実行する必要がある追加の作業が原因です。SSD コントローラーは利用可能な領域を見つけ、既存のデータを再書き込みし、未使用の領域を消去して、再書き込みができるようにします。このガベージコレクションというプロセスにより、SSD への内部的な書き込み増幅が発生し、ユーザーの書き込み操作に対する SSD 書き込み操作

の割合として表示されます。書き込み操作が 4,096 バイトの倍数でないか、4,096 バイトの境界に整合していない場合、パフォーマンスの低下はさらに大きくなります。少量のバイト数または整合していないバイト数で書き込む場合、SSD コントローラーは周辺のデータを読み取り、その結果を新しい場所に保存する必要があります。このパターンにより、書き込み増幅が大幅に増え、レイテンシーが増加し、I/O パフォーマンスが大きく低下します。

SSD コントローラーは複数の方法を利用すると、書き込み増幅の影響を減らすことができます。このような方法の 1 つにはSSD インスタンスストレージに領域を予約し、コントローラーが書き込み操作に利用できる領域をより効率的に管理できるようにすることです。これをオーバープロビジョニングと呼びます。インスタンスに提供された SSD ベースのインスタンスストアボリュームにはオーバープロビジョニングに対して予約された領域がありません。書き込み増幅を減らすにはボリュームの 10% を未使用の状態のままにし、SSD コントローラーがこれをオーバープロビジョニングに使用できるようにすることをお勧めします。これにより、使用できるストレージは減りますが、ディスクが総容量に近づいた場合でもパフォーマンスを向上させることができます。

TRIM をサポートするインスタンスストアボリュームの場合、TRIM コマンドを使用して、書き込んだデータが不要になったときはいつでも SSD コントローラーに通知することができます。これにより、より多くの空き領域がコントローラーに与えられ、その結果書き込み増幅が減り、パフォーマンスが向上します。詳細については「インスタンスストアボリュームの TRIM のサポート」を参照してください。

インスタンスストアボリュームの TRIM のサポート

一部のインスタンスタイプではTRIM を持つ SSD ボリュームがサポートされます。詳細については「EC2 インスタンスのインスタンスストアボリュームの制限」を参照してください。

> **Note**
>
> (Windows インスタンスのみ) Windows Server 2012 R2 を実行するインスタンスはAWS PV ドライバーバージョン 7.3.0 以降の TRIM をサポートしています。以前のバージョンの Windows Server を実行しているインスタンスは TRIM をサポートしていません。

TRIM をサポートしているインスタンスストアボリュームはインスタンスに割り当てられる前に完全に TRIM が実行されます。これらのボリュームはインスタンスの起動時にファイルシステムを使用してフォーマットされないため、マウントして使用する前にボリュームをフォーマットする必要があります。これらのボリュームを迅速に使用できるようにするにはボリュームをフォーマットするときに、TRIM 操作をスキップします。

(Windows インスタンス) 初回フォーマット中に TRIM のサポートを一時的に無効化するには`fsutil behavior set DisableDeleteNotify 1` コマンドを使用します。フォーマットが完了したら、`fsutil behavior set DisableDeleteNotify 0` を使用して TRIM のサポートを再度有効にします。

TRIM をサポートするインスタンスストアボリュームではTRIM コマンドを使用して、書き込んだデータが不要になったときに SSD コントローラーに通知することができます。これにより、より多くの空き領域がコントローラーに与えられ、その結果書き込み増幅が減り、パフォーマンスが向上します。Linux インスタンスでは`fstrim` コマンドを使用して定期的な TRIM を有効にします。Windows インスタンスでは`fsutil behavior set DisableDeleteNotify 0` コマンドを使用して、通常のオペレーション中に TRIM サポートが有効になるようにします。

EC2 インスタンスへのインスタンスストアボリュームの追加

NVMe インスタンスストアボリュームを使用するインスタンスタイプではサポートされているすべてのインスタンスストアボリュームが、起動時に自動的にインスタンスにアタッチされます。NVMe インスタンスストアボリュームはインスタンスの起動時に自動的に列挙され、デバイス名が割り当てられます。

C1、C3、M1、M2、M3、R3、D2、H1、I2、X1、X1e など、NVMe 以外のインスタンスストアボリュームのインスタンスタイプでは起動時にアタッチするインスタンスストアボリュームのブロックデバイスマッピングを手動で指定する必要があります。ブロックデバイスマッピングはインスタンス起動リクエストで、またはインスタンスの起動に使用される AMI で指定できます。ブロックデバイスマッピングにはデバイス名とそれがマッピングされたボリュームが含まれます。詳細については「Amazon EC2 インスタンスでのボリュームのブロックデバイスマッピング」を参照してください。

> **Important**
>
> インスタンスを起動する場合にのみ、インスタンスストアボリュームをインスタンスにアタッチできます。また、起動後のインスタンスにインスタンスストアボリュームをアタッチすることはできません。

インスタンスを起動したら、使用する前に、インスタンスのインスタンスストアボリュームがフォーマットされ、マウントされていることを確認する必要があります。インスタンスストアのルートボリュームは自動的にマウントされます。

ルートボリュームに関する考慮事項

ブロックデバイスマッピングでは常にインスタンスのルートボリュームを指定します。ルートボリュームは常に自動的にマウントされます。

Linux インスタンス – ルートボリュームはAmazon EBS ボリュームまたはインスタンスストアボリュームのいずれかです。ルートボリュームのインスタンスストアボリュームを持つインスタンスの場合、このボリュームのサイズは AMI によって異なりますが、最大サイズは 10 GB です。詳細については「ルートボリュームタイプ」を参照してください。

Windows インスタンス – ルートボリュームは Amazon EBS ボリュームである必要があります。インスタンスストアはルートボリュームではサポートされていません。

内容

- Amazon EC2 AMI にインスタンスストアボリュームを追加する
- 起動時に EC2 インスタンスにインスタンスストアボリュームを追加する
- インスタンスストアボリュームを EC2 インスタンスで使用できるようにする

Amazon EC2 AMI にインスタンスストアボリュームを追加する

インスタンスストアボリュームが含まれる、ブロックデバイスマッピングを持つ AMI を作成できます。

インスタンスストアボリュームブロックデバイスマッピングを指定する AMI を使用して、非 NVMe のインスタンスストアボリュームをサポートするインスタンスを起動すると、インスタンスにインスタンスストアボリュームが含まれます。AMI のインスタンスストアボリュームブロックデバイスマッピングの数がインスタンスに利用できるインスタンスストアボリュームの数を超えた場合、追加のインスタンスストアボリュームブロックデバイスマッピングは無視されます。

インスタンスストアボリュームブロックデバイスマッピングを指定する AMI を使用して、NVMe インスタンスストアボリュームをサポートするインスタンスを起動した場合、インスタンスストアボリュームブロックデバイスマッピングは無視されます。NVMe インスタンスストアボリュームをサポートするインスタンスはインスタンス起動リクエストと AMI で指定されたブロックデバイスマッピングに関らず、サポートされているすべてのインスタンスストアボリュームを取得します。これらのボリュームのデバイスマッピングは、オペレーティングシステムがこれらのボリュームを列挙する順序に応じて決まります。

考慮事項

- 使用できるインスタンスストアボリュームの数はインスタンスタイプによって異なります。詳細については「the section called “使用可能なインスタンスストアボリューム”」を参照してください。

- デバイス名をブロックデバイスごとに指定する必要があります。詳細については「Amazon EC2 インスタンス上のボリュームのデバイス名」を参照してください。
- インスタンスを起動する際に、AMI ブロックデバイスマッピングで指定された 非 NVMe インスタンスストアボリュームを省略したり、インスタンスストアボリュームを追加したりできます。
- M3 インスタンスの場合はAMI ではなく、インスタンスのブロックデバイスマッピングにあるインスタンスストアボリュームを指定します。Amazon EC2 はAMI のインスタンスストアボリュームブロックデバイスマッピングを無視することがあります。

Console

インスタンスストアボリュームを Amazon EBS-backed AMI に追加するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Instances] を選択し、インスタンスを選択してください。
3. [アクション]、[Image and templates (イメージとテンプレート)]、[イメージの作成] の順に選択してください。
4. [イメージの作成] ページで、イメージの意味のある名前と説明を追加します。
5. 追加する各インスタンスストアボリュームについて、[ボリュームの追加] を選択し、[ボリュームタイプ] からインスタンスストアボリュームを選択して、[デバイス] からデバイス名を選択してください。
6. [イメージを作成] を選択してください。

AWS CLI

インスタンスストアボリュームを AMI に追加するには

EBS-Backed AMI のブロックデバイスマッピングを指定するには、create-image コマンドを `--block-device-mappings` オプションと共に使用します。Amazon S3-backed AMI のブロックデバイスマッピングを指定するには、register-image コマンドを `--block-device-mappings` オプションと共に使用します。

```
--block-device-mappings file://mapping.json
```

次のブロックデバイスマッピングでは、2 つのインスタンスストアボリュームを追加します。

```
[
    {
```

```
        "DeviceName": "/dev/sdc",
        "VirtualName": "ephemeral0"
    },
    {
        "DeviceName": "/dev/sdd",
        "VirtualName": "ephemeral1"
    }
]
```

PowerShell

インスタンスストアボリュームを AMI に追加するには

New-EC2Image コマンドレットを -BlockDeviceMapping パラメータと共に使用して、EBS-Backed AMI のブロックデバイスマッピングを指定します。Register-EC2Image コマンドレットを -BlockDeviceMapping パラメータと共に使用して、Amazon S3-backed AMI のブロックデバイスマッピングを指定します。

```
-BlockDeviceMapping $bdm
```

次のブロックデバイスマッピングでは、2 つのインスタンスストアボリュームを追加します。

```
$bdm = @()

$sdc = New-Object -TypeName Amazon.EC2.Model.BlockDeviceMapping
$sdc.DeviceName = "/dev/sdc"
$sdc.VirtualName = "ephemeral0"
$bdm += $sdc

$sdd = New-Object -TypeName Amazon.EC2.Model.BlockDeviceMapping
$sdd.DeviceName = "/dev/sdd"
$sdd.VirtualName = "ephemeral1"
$bdm += $sdd
```

起動時に EC2 インスタンスにインスタンスストアボリュームを追加する

C1、C3、M1、M2、M3、R3、D2、H1、I2、X1、X1e など、非 NVMe インスタンスストアボリュームを持つインスタンスタイプを起動する場合は起動時にアタッチするインスタンスストアボリュームのブロックデバイスマッピングを指定する必要があります。ブロックデバイスマッピングはインスタンス起動リクエストで、またはインスタンスの起動に使用される AMI で指定する必要があります。

AMI にインスタンスストアボリュームのブロックデバイスマッピングが含まれている場合、AMI に含まれるよりも多くのインスタンスストアボリュームが必要でない限り、インスタンス起動リクエストでブロックデバイスマッピングを指定する必要はありません。

AMI にインスタンスストアボリュームのブロックデバイスマッピングが含まれていない場合はインスタンス起動リクエストでブロックデバイスマッピングを指定する必要があります。

NVMe インスタンスストアボリュームを使用するインスタンスタイプではサポートされているすべてのインスタンスストアボリュームが、起動時に自動的にインスタンスにアタッチされます。

考慮事項

- 使用できるインスタンスストアボリュームの数はインスタンスタイプによって異なります。詳細については「the section called “使用可能なインスタンスストアボリューム”」を参照してください。
- デバイス名をブロックデバイスごとに指定する必要があります。詳細については「Amazon EC2 インスタンス上のボリュームのデバイス名」を参照してください。
- M3 インスタンスの場合はインスタンスのブロックデバイスマッピングで指定しなくても、インスタンスストアボリュームを受け取る可能性があります。

Console

インスタンス起動リクエストでブロックデバイスマッピングを指定するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ダッシュボードから、[インスタンスの作成] を選択してください。
3. [Application and OS Images] (アプリケーションと OS イメージ) セクションで、使用する AMI を選択してください。
4. [ストレージの設定] セクションの [インスタンスストアボリューム] セクションにはインスタンスにアタッチできるインスタンスストアボリュームが一覧表示されます。
5. アタッチする各インスタンスストアボリュームの [デバイス名] で、使用するデバイス名を選択してください。
6. 必要に応じて残りのインスタンスの設定を設定し、[インスタンスの起動] を選択してください。

AWS CLI

インスタンス起動リクエストでブロックデバイスマッピングを指定するには

--block-device-mappings オプションで run-instances コマンドを使用します。

```
--block-device-mappings file://mapping.json
```

次のブロックデバイスマッピングでは、2 つのインスタンスストアボリュームを追加します。

```
[
    {
        "DeviceName": "/dev/sdc",
        "VirtualName": "ephemeral0"
    },
    {
        "DeviceName": "/dev/sdd",
        "VirtualName": "ephemeral1"
    }
]
```

PowerShell

インスタンス起動リクエストでブロックデバイスマッピングを指定するには

-BlockDeviceMapping オプションで New-EC2Instance コマンドレットを使用します。

```
-BlockDeviceMapping $bdm
```

次のブロックデバイスマッピングでは、2 つのインスタンスストアボリュームを追加します。

```
$bdm = @()

$sdc = New-Object -TypeName Amazon.EC2.Model.BlockDeviceMapping
$sdc.DeviceName = "/dev/sdc"
$sdc.VirtualName = "ephemeral0"
$bdm += $sdc

$sdd = New-Object -TypeName Amazon.EC2.Model.BlockDeviceMapping
$sdd.DeviceName = "/dev/sdd"
$sdd.VirtualName = "ephemeral1"
$bdm += $sdd
```

インスタンスストアボリュームを EC2 インスタンスで使用できるようにする

インスタンスストアボリュームが接続しているインスタンスを起動したら、アクセスする前にボリュームをマウントする必要があります。

Linux インスタンス

インスタンスを起動してから、選択したファイルシステムでボリュームをフォーマットすることもできます。

Linux でインスタンスストアボリュームを使用できるようにするには

1. SSH クライアントを使用してインスタンスに接続します。詳細については「SSH を使用した Linux インスタンスへの接続」を参照してください。
2. df -h コマンドを使用して、フォーマットおよびマウントされたボリュームを表示します。

```
$ df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        3.8G   72K  3.8G   1% /dev
tmpfs           3.8G     0  3.8G   0% /dev/shm
/dev/nvme0n1p1  7.9G  1.2G  6.6G  15% /
```

3. lsblk を使用して、起動時にマッピングされたが、フォーマットおよびマウントされていないボリュームを表示します。

```
$ lsblk
NAME          MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
nvme0n1       259:1    0    8G  0 disk
##nvme0n1p1   259:2    0    8G  0 part /
##nvme0n1p128 259:3    0    1M  0 part
nvme1n1       259:0    0 69.9G  0 disk
```

4. マッピングされたのみのインスタンスストアボリュームをフォーマットしてマウントするには以下の作業を行います。

   a. mkfs コマンドを使用してデバイスでファイルシステムを作成します。

   ```
   $ sudo mkfs -t xfs /dev/nvme1n1
   ```

   b. mkdir コマンドを使用してデバイスをマウントするディレクトリを作成します。

```
$ sudo mkdir /data
```

c. mount コマンドを使用して、新しく作成されたディレクトリにデバイスをマウントします。

```
$ sudo mount /dev/nvme1n1 /data
```

Windows インスタンス

Windows インスタンスではNTFS ファイルシステムでインスタンスストアボリュームをフォーマットします。

Windows Disk Management を使用してインスタンスストアボリュームを表示できます。詳細については「非 NVMe ディスクを一覧表示する」を参照してください。

インスタンスストアボリュームを手動でマウントするには

1. [スタート] ボタンをクリックし、コンピュータの管理と入力して、Enterキーを押します。
2. 左側のパネルで、[ディスクの管理] を選択してください。
3. ボリュームを初期化するかどうかを確認するメッセージが表示されたら、初期化するボリュームを選択し、ユースケースに応じて必要なパーティションタイプを選択して、[OK] を選択してください。
4. ボリュームの一覧で、マウントするボリュームを右クリックし、[新しいシンプルボリューム] を選択してください。
5. ウィザードで、[次へ] を選択してください。
6. [ボリュームサイズの指定] 画面で、[次へ] を選択して最大ボリュームサイズを使用します。または最小ディスク容量と最大ディスク容量の間のボリュームサイズを選択してください。
7. [ドライブ文字またはパスの割り当て] 画面で、次のいずれかの操作を行い、[次へ] を選択してください。

   - ドライブ文字を使用してボリュームをマウントするには[次のドライブ文字を割り当てる] を選択し、使用するドライブ文字を選択してください。
   - ボリュームをフォルダとしてマウントするには[次の空の NTFS フォルダーにマウントする] を選択し、[参照] を選択して、使用するフォルダを作成または選択してください。
   - ドライブ文字またはパスを使用せずにボリュームをマウントするには[ドライブ文字またはドライブパスを割り当てない] を選択してください。

8. [パーティションのフォーマット] 画面で、ボリュームをフォーマットするかどうかを指定します。ボリュームをフォーマットする場合は必要なファイルシステムとユニットサイズを選択し、ボリュームラベルを指定します。
9. [次へ]、[完了] の順に選択してください。

M1 および C1 EC2 インスタンスのインスタンスストアスワップボリュームの有効化

> **Note**
>
> このトピックは`c1.medium` および `m1.small` Linux インスタンスにのみ当てはまります。

`c1.medium` および `m1.small` インスタンスタイプは物理メモリの量が制限されています。そのため、起動時に Linux システムの仮想メモリとして機能する 900 MiB のスワップボリューム (またはスワップ空間) が与えられます。Linux のスワップ空間はシステムで物理的に割り当てられたよりも多くのメモリを必要とする場合に使用できます。スワップ空間を有効にすると、Linux システムは頻繁に使用されないメモリページを物理メモリからスワップ空間 (既存のファイルシステムの専用パーティションまたはスワップファイル) にスワップし、高速なアクセスを必要とするメモリページのためにその空間を解放します。

> **Note**
>
> - スワップ空間をメモリページングに使用しても、RAM 使用時ほど高速でも効率的でもありません。スワップ空間に定期的にメモリをページングするワークロードの場合はRAM が多くサイズの大きいインスタンスタイプに移行することを検討してください。詳細については「Amazon EC2 インスタンスタイプの変更」を参照してください。
> - Linux カーネルはこのスワップ空間をルートボリューム上のパーティションとして認識しますが、ルートボリュームのタイプに関係なく、実際には別のインスタンスストアボリュームです。

Amazon Linux は自動的にこのスワップ空間を有効にして使用しますが、AMI ではこのスワップ空間を認識して使用するために、追加のステップが必要になる場合があります。インスタンスがスワップ空間を使用しているかどうか確認するにはswapon -s コマンドを使用できます。

```
[ec2-user ~]$ swapon -s
Filename                                Type            Size    Used    Priority
/dev/xvda3                              partition       917500  0       -1
```

上記のインスタンスには900 MiB のスワップボリュームがアタッチされ、有効になっています。このコマンドでスワップボリュームが表示されない場合はそのデバイスに対してスワップ空間を有効しなければならない可能性があります。利用可能なディスクはlsblk コマンドを使用して確認します。

```
[ec2-user ~]$ lsblk
NAME  MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
xvda1 202:1    0    8G  0 disk /
xvda3 202:3    0  896M  0 disk
```

ここで、スワップボリューム xvda3 はインスタンスで利用できますが、有効になっていません (MOUNTPOINT フィールドが空です)。スワップボリュームは swapon コマンドを使って有効にできます。

> **Note**
>
> /dev/ でリストされるデバイス名の先頭に lsblk を付加する必要があります。デバイスはsda3、sde3、xvde3 など、異なる名前になる場合があります。システムのデバイス名は次のコマンドで使用します。

```
[ec2-user ~]$ sudo swapon /dev/xvda3
```

これで、スワップ空間が lsblk および swapon -s 出力に表示されます。

```
[ec2-user ~]$ lsblk
NAME  MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
xvda1 202:1    0    8G  0 disk /
xvda3 202:3    0  896M  0 disk [SWAP]
[ec2-user ~]$ swapon -s
Filename                                Type            Size    Used    Priority
/dev/xvda3                              partition       917500  0       -1
```

また、システムを起動する度にこのスワップ空間が自動的に有効になるように、/etc/fstab ファイルを編集する必要もあります。

```
[ec2-user ~]$ sudo vim /etc/fstab
```

(システムの swap デバイス名を使用して) 次の行を /etc/fstab ファイルに追加します。

```
/dev/xvda3       none    swap    sw  0       0
```

インスタンスストアボリュームをスワップ空間として使用するには

どのインスタンスストアボリュームもスワップ空間として使用できます。例えば、m3.medium インスタンスタイプはスワップ空間に適した 4 GB の SSD インスタンスストアボリュームを含みます。インスタンスストアボリュームがはるかに大きい場合 (例えば、350 GB)、ボリュームに 4~8 GB の小さいスワップパーティションを作成し、残りをデータボリュームにすることもできます。

> **Note**
>
> この手順はインスタンスストレージをサポートするインスタンスタイプのみに適用されます。サポートされているインスタンスタイプについてはEC2 インスタンスのインスタンスストアボリュームの制限を参照してください。

1. インスタンスにアタッチされたブロックデバイスの一覧を表示して、インスタンスストアボリュームのデバイス名を取得します。

   ```
   [ec2-user ~]$ lsblk -p
   NAME       MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
   /dev/xvdb  202:16   0   4G  0 disk /media/ephemeral0
   /dev/xvda1 202:1    0   8G  0 disk /
   ```

   この例ではインスタンスストアボリュームは /dev/xvdb です。これは Amazon Linux インスタンスであるため、インスタンスストアボリュームはフォーマットされ、/media/ephemeral0 にマウントされます。すべての Linux オペレーティングシステムでこれが自動的に実行されるわけではありません。
2. (省略可能) インスタンスストアボリュームがマウントされている場合 (lsblk コマンドの出力に MOUNTPOINT が表示されます)、次のコマンドを使ってアンマウントする必要があります。

   ```
   [ec2-user ~]$ sudo umount /dev/xvdb
   ```
3. mkswap コマンドを使って、デバイスに Linux スワップ領域をセットアップします。

```
[ec2-user ~]$ sudo mkswap /dev/xvdb
mkswap: /dev/xvdb: warning: wiping old ext3 signature.
Setting up swapspace version 1, size = 4188668 KiB
no label, UUID=b4f63d28-67ed-46f0-b5e5-6928319e620b
```

4. 新しいスワップ空間を有効にします。

```
[ec2-user ~]$ sudo swapon /dev/xvdb
```

5. 新しいスワップ空間が使用されていることを確認します。

```
[ec2-user ~]$ swapon -s
Filename    Type  Size Used Priority
/dev/xvdb                               partition 4188668 0 -1
```

6. システムを起動する度にこのスワップ空間が自動的に有効になるように、/etc/fstab ファイルを編集します。

```
[ec2-user ~]$ sudo vim /etc/fstab
```

/etc/fstab ファイルに /dev/xvdb (または /dev/sdb) 用の項目がある場合はそれを以下の行に合わせて変更します。このデバイス用の項目がない場合は/etc/fstab ファイルに以下の行を追加します (システムのスワップデバイス名を使用します)。

```
/dev/xvdb       none    swap    sw  0       0
```

> **Important**
>
> インスタンスが停止または休止すると、インスタンスストアボリュームデータが失われます。これにはStep 3で作成したインスタンスストアスワップ領域のフォーマットが含まれます。インスタンスストアのスワップ領域を使用するように設定されたインスタンスを停止および再起動した場合、新しいインスタンスストアボリュームでStep 1からStep 5を繰り返す必要があります。

EC2 インスタンスでのインスタンスストアボリュームの初期化

Amazon EC2 でのディスクの仮想化方法が原因となり、一部のインスタンスストアボリュームに対する最初の書き込みは書き込みの場所にかかわらず、それ以降の書き込みより速度が遅くなります。ほとんどのアプリケーションではインスタンスの存続期間全体でこのコストを負担することは許容範囲内です。ただし、高いディスクパフォーマンスを必要とする場合は本稼働環境での使用の前に、ドライブのすべての場所に一度書き込みを行うことで初期化することをお勧めします。

> **Note**
>
> 直接アタッチされた Solid State Drive (SSD) および TRIM サポートを使用するインスタンスタイプは初期化を行わずに、起動時に最大限のパフォーマンスを発揮します。各インスタンスタイプのインスタンスストアについてはEC2 インスタンスのインスタンスストアボリュームの制限を参照してください。

レイテンシーやスループットに関してさらに柔軟性が必要な場合は Amazon EBS を使用することをお勧めします。

インスタンスストアボリュームを初期化するには初期化するストア (例: dd または /dev/sdb) に応じて、次の /dev/nvme1n1 コマンドを使用します。

> **Note**
>
> 必ずドライブをアンマウントしてから、このコマンドを実行してください。
> 初期化には長い時間がかかる場合があります (エクストララージのインスタンスで約 8 時間)。

インスタンスストアボリュームを初期化するには`m1.large`、`m1.xlarge`、`c1.xlarge`、`m2.xlarge`、`m2.2xlarge`、`m2.4xlarge` インスタンスタイプで次のコマンドを使用します。

```
dd if=/dev/zero of=/dev/sdb bs=1M
dd if=/dev/zero of=/dev/sdc bs=1M
dd if=/dev/zero of=/dev/sdd bs=1M
dd if=/dev/zero of=/dev/sde bs=1M
```

すべてのインスタンスストアボリュームに対して同時に初期化を実行するには次のコマンドを使用します。

```
dd if=/dev/zero bs=1M|tee /dev/sdb|tee /dev/sdc|tee /dev/sde > /dev/sdd
```

ドライブを RAID 用に構成すると、ドライブのすべての場所に書き込みを行うことで、ドライブが初期化されます。ソフトウェアベースの RAID を構成するときは再構築の最低速度を必ず変更してください。

```
echo $((30*1024)) > /proc/sys/dev/raid/speed_limit_min
```

Amazon EC2 インスタンスストアボリュームの詳細なパフォーマンス統計

Amazon EC2 は、Nitro ベースの Amazon EC2 インスタンスにアタッチされた NVMe インスタンスストアボリュームに対するリアルタイムかつ高解像度のパフォーマンス統計を提供します。これらの統計は集計カウンターとして表示され、インスタンスの存続期間にわたって保持されます。この統計は、オペレーションの累積数、送受信されたバイト数、読み取り/書き込み I/O オペレーションに費やされた時間、読み取り/書き込み I/O オペレーションのヒストグラムに関する詳細を提供します。これらの統計では Amazon EBS の詳細なパフォーマンス統計との一貫性が維持されていますが、I/O サイズごとに分類された詳細なレイテンシーヒストグラムも含まれるため、ストレージのパフォーマンスパターンをより詳細に把握することができます。それにより可視性が向上し、レイテンシーの問題が発生している特定の I/O サイズを正確に特定できることから、アプリケーションのパフォーマンスの最適化と、問題の効果的なトラブルシューティングが可能になります。

これらの統計は最大 1 秒間隔で詳細に収集できます。リクエストが 1 秒間隔よりも頻繁に行われると、NVMe ドライバーが、後で処理されるリクエストを他の管理コマンドとともにキューに入れることがあります。

考慮事項

- この統計は、Nitro ベースのインスタンスにアタッチされた NVMe インスタンスストアボリュームのみでサポートされています。
- カウンターは、インスタンスの停止および再起動をまたいで保持されません。
- この統計は追加料金なしでご利用いただけます。

統計

NVMe ブロックデバイスは次の統計を提供します。

統計名	フルネーム	型	説明
total_rea d_ops	読み取りオペレーションの合計回数	Counter	完了した読み取りオペレーションの合計回数。
total_wri te_ops	書き込みオペレーションの合計数	Counter	完了した書き込みオペレーションの合計回数。
total_rea d_bytes	読み取りバイトの合計数	Counter	転送された読み取りバイトの合計数。
total_wri te_bytes	書き込みバイトの合計数	Counter	転送された書き込みバイトの合計数。
total_rea d_time	読み取り時間の合計数	Counter	完了したすべての読み取りオペレーションで費やされた合計時間 (マイクロ秒単位)。
total_wri te_time	書き込み時間の合計数	Counter	完了したすべての書き込みオペレーションで費やされた合計時間 (マイクロ秒単位)。
instance_ store_vol ume_perfo rmance_ex ceeded_io ps	需要がボリュームの最大 IOPS を超えた時間の合計数	Counter	IOPS リクエストがボリュームの最大 IOPS を超えた時間の合計数 (単位: マイクロ秒)。0 を超える値は、ワークロードが、ボリュームが提供できる IOPS よりも多くの IOPS を要求したことを示しています。このメトリクスの2 つのスナップショットの時点間における増分カウントは、最小限に抑えるのが理想的です。
instance_ store_vol ume_perfo rmance_ex ceeded_tp	リクエストがボリュームの最大スループットを越えた時間の合計数	Counter	スループットリクエストがボリュームの最大スループットを超えた時間の合計数 (単位: マイクロ秒)。0 を超える値は、ワークロードが、ボリュームが提供できるスループットよりも多くの IOPS を要求したことを示しています。このメトリクスの2 つのスナップショットの時点間における増分カウントは、最小限に抑えるのが理想的です。

統計名	フルネーム	型	説明
volume_queue_length	ボリュームキューの長さ	ポイントインタイム	完了を待機している読み取りおよび書き込みオペレーションの数。
read_io_latency_histogram	I/O ヒストグラムの読み取り	ヒストグラム*	各レイテンシービン内で完了した読み取りオペレーションの数 (単位: マイクロ秒)。
write_io_latency_histogram	I/O ヒストグラムの書き込み	ヒストグラム*	各レイテンシービン内で完了した書き込みオペレーションの数 (単位: マイクロ秒)。

> **Note**
>
> * ヒストグラム統計は正常に完了した I/O オペレーションのみを表します。停止している、あるいは正常でない I/O オペレーションは含まれませんが、ポイントインタイム統計として volume_queue_length 統計に表示されます。

統計へのアクセス

統計には、インスタンスストアボリュームがアタッチされているインスタンスから、直接アクセスする必要があります。統計には、次のいずれかの方法を使用してアクセスできます。

Linux インスタンス

Amazon CloudWatch

インスタンスから統計を収集し Amazon CloudWatch のカスタムメトリクスとして利用できるように、Amazon CloudWatch エージェントを設定できます。その後、CloudWatch のメトリクスを使用して、I/O パターンの分析、パフォーマンスの傾向の追跡、カスタムダッシュボードの作成、パフォーマンスのしきい値に基づく自動アラームの設定を行うことができます。

CloudWatch エージェントの設定の詳細については「Collect Amazon EC2 instance store volume metrics」を参照してください。

nvme-cli tool

統計にアクセスするには

1. ボリュームがアタッチされているインスタンスに接続します。
2. 2025 年 9 月 15 日以降にリリースされた Amazon Linux 2023 AMI には、nvme-cli ツールの最新バージョンが含まれています。以前の Amazon Linux AMI を使用している場合は nvme-cli ツールを更新します。

```
sudo yum install nvme-cli
```

3. 次のコマンドを実行し、ボリュームのデバイス名を指定します。

```
sudo nvme amzn stats /dev/nvme0n1
```

この統計は、I/O サイズ別に分類された詳細なレイテンシーヒストグラムも提供します。I/O サイズ別に分類された統計を表示するには --details オプションを含めます。例えば、次のようになります。

```
sudo nvme amzn stats --details /dev/nvme0n1
```

--help オプションを指定することで、ツールの使用方法に関する詳細情報を入手できます。例えば、次のようになります。

```
sudo nvme amzn stats --help
```

Windows インスタンス

nvme_amzn.exe tool

統計にアクセスするには

1. ボリュームがアタッチされているインスタンスに接続します。
2. AWSNVMe ドライバーのバージョン 1.7.0 以降を使用していることを確認します。AWSNVMe ドライバーの更新の詳細については、「AWS NVMe ドライバー」を参照してください。

3. ボリュームのディスク番号を取得します。詳細は、「Amazon EC2 Windows インスタンスの NVMe ディスクをボリュームにマッピングする」を参照してください。
4. 次のコマンドを管理者として実行し、ボリュームのディスク番号を指定します。

```
.\nvme_amzn.exe stats disk_number
```

この統計は、I/O サイズ別に分類された詳細なレイテンシーヒストグラムも提供します。I/O サイズ別に分類された統計を表示するには `--details` オプションを含めます。例：

```
.\nvme_amzn.exe stats --details disk_number
```

Amazon EC2 インスタンスのルートボリューム

インスタンスを起動すると、インスタンスのルートボリュームが作成されます。ルートボリュームには、インスタンスの起動に使用されるイメージが含まれています。各インスタンスには 1 つのルートボリュームがあります。ストレージボリュームは、起動中または起動後にインスタンスに追加できます。

インスタンスの起動に使用する AMI によってルートボリュームのタイプが決まります。インスタンスは、Amazon EBS-backed AMI (Linux インスタンス および Windows インスタンス) または Amazon S3-backed AMI (Linux インスタンスのみ) から起動できます。各タイプの AMI を使用して実行できることは大きく異なります。違いについての詳細は ルートボリュームタイプ を参照してください。

Amazon EBS-backed AMI を使用することをお勧めします。これらのインスタンスは起動速度が速く、永続的ストレージを使用しているからです。

特定のデバイス名がルートボリューム用に予約されています。詳細については「Amazon EC2 インスタンス上のボリュームのデバイス名」を参照してください。

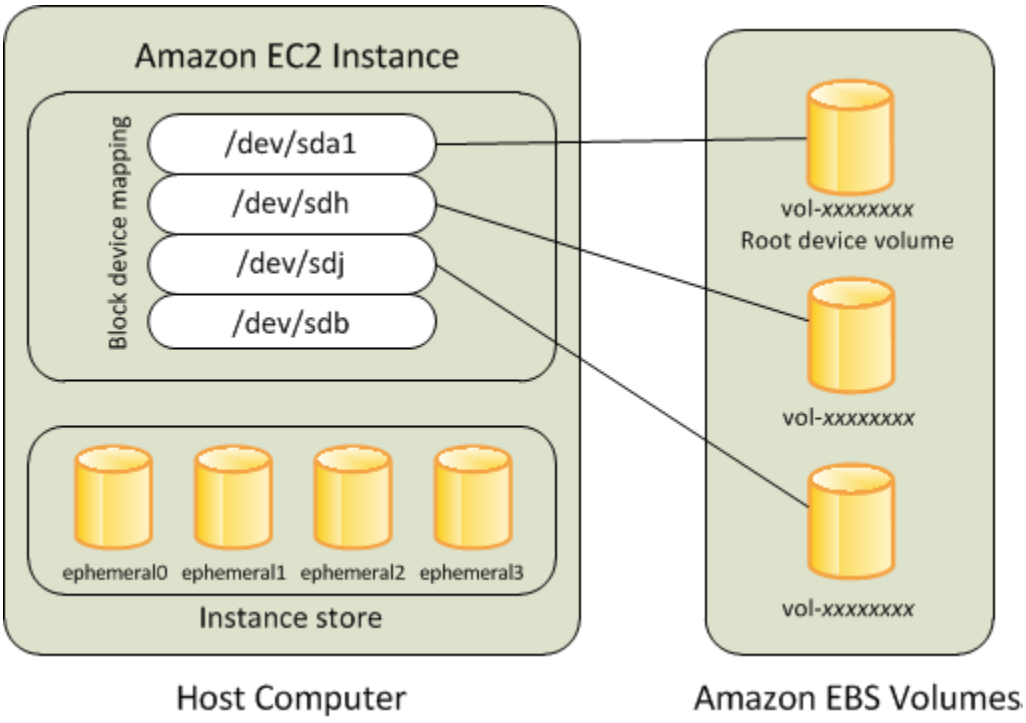
内容

- Amazon EBS ルートボリュームを持つインスタンス
- インスタンスストアのルートボリュームを持つインスタンス (Linux インスタンスのみ)
- Amazon EC2 インスタンスの終了後も Amazon EBS ルートボリュームを保持する
- Amazon EC2 インスタンスを停止させることなく、そのルートボリュームを置換する

Amazon EBS ルートボリュームを持つインスタンス

Amazon EBS をルートボリュームに使用するインスタンスには、Amazon EBS ボリュームが自動的にアタッチされます。Amazon EBS-AMI を持つインスタンスを起動するときに、AMI で参照されている Amazon EBS スナップショットごとに 1 つの Amazon EBS ボリュームが作成されます。インスタンスタイプによっては、Amazon EBS ボリュームまたはインスタンスストアボリュームをオプションで使用できます。

EBS ルートボリュームを持つインスタンスは、停止後に再起動できます。アタッチされているボリュームに格納されているデータに影響を及ぼすこともありません。EBS ルートボリュームを持つインスタンスが停止状態にあるときは、インスタンスおよびボリューム関連のさまざまなタスクを実行できます。例えば、インスタンスのプロパティの変更、そのサイズの変更、あるいは使用しているカーネルの更新を行えます。また、デバッグなどの目的で別の実行中インスタンスにそのルートボリュームをアタッチすることもできます。詳細については、「Amazon EBS ボリューム」を参照してください。



制限

st1 または sc1 EBS ボリュームをルートボリュームとして使用することはできません。

インスタンスの障害

EBS ルートボリュームを持つインスタンスに障害が発生した場合は、以下のいずれかの方法によってセッションを復元できます。

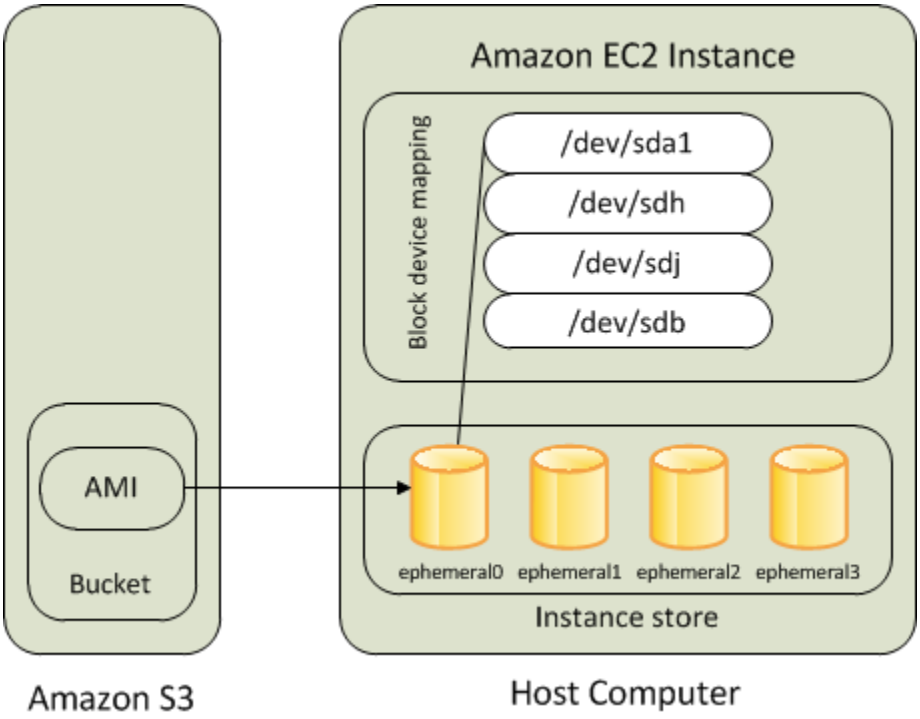
- 停止して再起動します (最初にこの方法を試してください)。
- 関連するすべてのボリュームのスナップショットを自動的に作成し、新しい AMI を作成します。詳細については「Amazon EBS-backed AMI を作成する」を参照してください。
- 以下の手順に従って、ボリュームを新しいインスタンスにアタッチします。
  1. ルートボリュームのスナップショットを作成します。
  2. 作成したスナップショットを使用して新しい AMI を登録します。
  3. 新しい AMI から新しいインスタンスを起動します。
  4. 残りの Amazon EBS ボリュームを古いインスタンスからデタッチします。
  5. Amazon EBS ボリュームを新しいインスタンスに再アタッチします。

インスタンスストアのルートボリュームを持つインスタンス (Linux インスタンスのみ)

> **Note**
>
> Windows インスタンスは、インスタンスストアのルートボリュームをサポートしていません。

インスタンスストアをルートボリュームに使用するインスタンスでは、自動的に 1 つまたは複数のインスタンスストアボリュームを利用できるようになり、そのボリュームの 1 つがルートボリュームとして機能します。インスタンスが Amazon S3-backed AMI から起動されると、AMI がルートボリュームにコピーされます。インスタンスタイプによっては、オプションで追加のインスタンスストアボリュームを使用できることに注意してください。

インスタンスストアボリュームのデータはインスタンスが実行している間は維持されますが、インスタンスが終了すると (インスタンスストアのルートボリュームを持つインスタンスは [停止] アクションをサポートしていません)、またはインスタンスが失敗すると (基盤となるドライブに問題がある場合など)、このデータは削除されます。詳細については「EC2 インスタンス用のインスタンスストアの一時ブロックストレージ」を参照してください。



サポートされるインスタンスタイプ

インスタンスストアボリュームをルートボリュームとしてサポートするインスタンスタイプは、C1、C3、D2、I2、M1、M2、M3、R3、X1 のみです。

インスタンスの障害

インスタンスストアのルートボリュームを持つインスタンスが失敗または終了した場合、そのインスタンスは復元できません。インスタンスストアのルートボリュームを持つインスタンスの使用を予定している場合は、インスタンスストアボリュームのデータを複数のアベイラビリティーゾーンにまたがって分散させることを強くお勧めします。また、インスタンスストアボリュームからの重要データは永続的ストレージに定期的にバックアップする必要があります。

Amazon EC2 インスタンスの終了後も Amazon EBS ルートボリュームを保持する

デフォルトでは、インスタンスの Amazon EBS ルートボリュームは、インスタンスを終了すると削除されます。インスタンスの終了後も Amazon EBS ルートボリュームが永続化するように、デフォルトの動作を変更できます。デフォルトの動作を変更するには、`DeleteOnTermination` 属性を `false` に設定します。これは、インスタンスの起動時または後で実行できます。

タスク

- インスタンスの起動時に永続化するためのルートボリュームの設定
- 既存のインスタンスで永続化するためのルートボリュームの設定
- ルートボリュームが永続化するように設定されていることの確認

インスタンスの起動時に永続化するためのルートボリュームの設定

インスタンスの起動時に永続化するようにルートボリュームを設定できます。

Console

インスタンスの起動時に、永続化するようにルートボリュームを設定するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [インスタンス]、[インスタンスの作成] の順に選択します。
3. Amazon マシンイメージ (AMI) を選択して、インスタンスタイプ、キーペアの順に選択し、ネットワークを設定します。
4. [ストレージを設定] には [アドバンスト] を選択します。
5. ルートボリュームを拡張します。
6. [終了時に削除] には、[いいえ] を選択します。
7. インスタンスの設定が終了したら、[インスタンスを起動] をクリックします。

AWS CLI

インスタンスの起動時に、永続化するようにルートボリュームを設定するには

run-instances コマンドを使用して、次のオプションを含めます。

```
--block-device-mappings file://mapping.json
```

mapping.json で、DeleteOnTermination 属性を false に設定するブロックデバイスマッピングを指定します。

```
[
    {
        "DeviceName": "/dev/sda1",
        "Ebs": {
            "DeleteOnTermination": false
```

```
        }
    }
]
```

PowerShell

インスタンスの起動時に、永続化するようにルートボリュームを設定するには

New-EC2Instance コマンドレットを使用して、次のパラメータを含めます。

```
-BlockDeviceMapping $bdm
```

DeleteOnTermination 属性を $false に設定するブロックデバイスマッピングを作成します。

```
$ebs = New-Object Amazon.EC2.Model.EbsBlockDevice
$ebs.DeleteOnTermination = $false
$bdm = New-Object Amazon.EC2.Model.BlockDeviceMapping
$bdm.DeviceName = "dev/xvda"
$bdm.Ebs = $ebs
```

既存のインスタンスで永続化するためのルートボリュームの設定

実行中のインスタンスで永続化するようにルートボリュームを設定できます。Amazon EC2 コンソールを使用してこのタスクを完了することはできないことに留意してください。

AWS CLI

既存のインスタンスで永続化するようにルートボリュームを設定するには

DeleteOnTermination 属性を false に設定するブロックデバイスマッピングを指定して modify-instance-attribute コマンドを使用します。

```
aws ec2 modify-instance-attribute \
    --instance-id i-1234567890abcdef0 \
    --block-device-mappings file://mapping.json
```

mapping.json で、以下を指定します。

```
[
```

```
    {
        "DeviceName": "/dev/xvda",
        "Ebs": {
            "DeleteOnTermination": false
        }
    }
]
```

PowerShell

既存のインスタンスで永続化するようにルートボリュームを設定するには

ブロックデバイスマッピングで DeleteOnTermination 属性を $false に設定して、Edit-EC2InstanceAttribute コマンドレットを使用します。

```
$ebs = New-Object Amazon.EC2.Model.EbsInstanceBlockDeviceSpecification
$ebs.DeleteOnTermination = $false
$bdm = New-Object Amazon.EC2.Model.InstanceBlockDeviceMappingSpecification
$bdm.DeviceName = "/dev/xvda"
$bdm.Ebs = $ebs
Edit-EC2InstanceAttribute `
    -InstanceId i-1234567890abcdef0 `
    -BlockDeviceMapping $bdm
```

ルートボリュームが永続化するように設定されていることの確認

ルートボリュームが永続化するように設定されていることを確認できます。

Console

ルートボリュームが永続化するように設定されているかを確認するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [インスタンス] を選択してから、インスタンスを選択します。
3. [ストレージ] タブの [ブロックデバイス] で、ルートボリュームのエントリを見つけます。[合わせて削除] が No の場合、ボリュームは永続化するように設定されます。

AWS CLI

ルートボリュームが永続化するように設定されているかを確認するには

describe-instances コマンドを使用して、DeleteOnTermination 属性が false に設定されていることを確認します。

```
aws ec2 describe-instances \
    --instance-id i-1234567890abcdef0 \
    --query "Reservations[].Instances[].BlockDeviceMappings"
```

以下は出力の例です。

```
[
    [
        {
            "DeviceName": "/dev/xvda",
            "Ebs": {
                "AttachTime": "2024-07-12T04:05:33.000Z",
                "DeleteOnTermination": false,
                "Status": "attached",
                "VolumeId": "vol-1234567890abcdef0"

        }
    ]
]
```

PowerShell

ルートボリュームが永続化するように設定されているかを確認するには

Get-EC2Instance コマンドレットを使用して、DeleteOnTermination 属性が False に設定されていることを確認します。

```
(Get-EC2Instance -InstanceId i-
i-1234567890abcdef0).Instances.BlockDeviceMappings.Ebs
```

以下は出力の例です。

```
AssociatedResource  :
AttachTime          : 7/12/2024 4:05:33 AM
DeleteOnTermination : False
Operator            :
Status              : attached
VolumeId            : vol-1234567890abcdef0
```

Amazon EC2 インスタンスを停止させることなく、そのルートボリュームを置換する

Amazon EC2 では、実行中のインスタンスのルート EBS ボリュームを置き換えることができます。

- インスタンスストアボリュームで復元されたデータ – インスタンスストアボリュームは、ルートボリュームが復元された後も、インスタンスにアタッチされたままになります。
- データ (非ルート) Amazon EBS ボリュームに保存されたデータ - 非ルート Amazon EBS ボリュームは、ルートボリュームが復元された後もインスタンスに接続されたままです。
- ネットワーク設定 — すべてのネットワークインターフェイスはインスタンスにアタッチされたままとなり、IP アドレス、識別子、およびアタッチメント ID を保持します。インスタンスが利用可能になると、保留中のネットワークトラフィックがすべてフラッシュされます。さらに、インスタンスは同じ物理ホスト上に残るため、パブリック IP アドレス、プライベート IP アドレス、および DNS 名が保持されます。
- IAM ポリシー — インスタンスに関連付けられた IAM プロファイルとポリシー (タグベースのポリシーなど) は保持され、適用されます。

内容

- ルートボリューム置換タスクの仕組み
- 考慮事項
- ルートボリュームを置き換える

ルートボリューム置換タスクの仕組み

インスタンスのルートボリュームを復元すると、ルートボリュームの置換タスクが作成されます。元のルートボリュームがインスタンスからデタッチされ、その代わりに新しいルートボリュームがインスタンスにアタッチされます。インスタンスのブロックデバイスマッピングは、置換用ルートボリュームの ID を反映するように更新されます。

インスタンスのルートボリュームを置き換える場合は、新しいボリュームのソースを指定する必要があります。可能なオプションは次のとおりです。

ルートボリュームを元の状態に復元する

このオプションは、現在のルートボリュームを、その作成に使用されたスナップショットに基づくボリュームに置き換えます。

起動状態の使用に関する考慮事項

置換用ルートボリュームには、元のルートボリュームと同じタイプ、サイズ、終了時の削除属性が割り当てられます。

スナップショットを使用してボリュームを置き換える

このオプションは、現在のルートボリュームを、指定したスナップショットに基づく置換ボリュームに置き換えます。このルートボリュームから以前に作成した特定のスナップショットはその一例です。これは、ルートボリュームの破損やゲストオペレーティングシステムのネットワーク設定エラーによって発生した問題から復旧する必要がある場合に役立ちます。

置換用ルートボリュームには、元のルートボリュームと同じタイプ、サイズ、終了時の削除属性が割り当てられます。

スナップショット使用時の考慮事項

- 使用できるのは、インスタンスの現在または以前のルートボリュームから直接作成されたスナップショットのみです。
- ルートボリュームから作成したスナップショットにより作成されたスナップショットのコピーは使用できません。
- ルートボリュームの置き換えに成功した後も、元のルートボリュームから取得したスナップショットを使用して、新しい (置換) ルートボリュームに置き換えることができます。

AMI を使用したボリュームを置き換える

このオプションは、指定した AMI を使用して現在のルートボリュームを置き換えます。これは、オペレーティングシステム、およびアプリケーションのパッチ適用やアップグレードを実行する必要がある場合に役立ちます。AMI には、インスタンスと同じ製品コード、請求情報、アーキテクチャタイプ、仮想化タイプが必要です。

インスタンスで ENA または sriov-net が有効になっている場合は、これらの機能をサポートする AMI を使用する必要があります。インスタンスで ENA または sriov-net が有効になっていない場合は、それらの機能をサポートしていない AMI を選択できますが、ENA または sriov-net をサポートする AMI を選択すると、自動的にサポートを追加できます。

インスタンスで NitroTPM が有効になっている場合は、NitroTPM が有効になっている AMI を使用する必要があります。NitroTPM サポートは、選択した AMI に関係なく、インスタンスが NitroTPM サポート用に設定されていない場合は有効になりません。

インスタンスが AMI のブートモードをサポートしている場合は、インスタンスとは異なるブートモードの AMI を選択できます。インスタンスがブートモードをサポートしていない場合、リクエストは失敗します。インスタンスがブートモードをサポートしている場合、新しいブートモードがインスタンスに伝達され、それに応じてその UEFI データが更新されます。ブート順序を手動で変更したり、プライベートカーネルモジュールをロードするためにプライベート UEFI セキュアブートキーを追加したりした場合、ルートボリュームの置換時に変更内容が失われます。

置換用ルートボリュームには、元のルートボリュームと同じボリュームタイプ、削除時の削除属性が割り当てられます。置換ルートボリュームのサイズは、AMI ルートボリュームブロックデバイスマッピングのサイズと現在のルートボリュームのサイズのいずれか大きい方です。

ルートボリュームの置換タスクの完了後、コンソール、AWS CLI または AWS SDK を使用してインスタンスを記述すると、次の新しい情報および更新された情報が反映されます。

- 新しい AMI ID
- ルートボリュームの新しいボリューム ID
- 更新されたブートモード設定 (AMI によって変更された場合)
- 更新された NitroTPM 設定 (AMI によって有効になっている場合)
- 更新された ENA 設定 (AMI によって有効になっている場合)
- 更新された sriov-net 設定 (AMI によって有効になっている場合)

新しい AMI ID はインスタンスメタデータにも反映されます。

AMI を使用ためのする考慮事項

- ブロックデバイスマッピングが複数ある AMI を使用する場合、AMI のルートボリュームのみが使用されます。他の (非ルート) ボリュームは無視されます。
- この機能を使用できるのは、AMI とそれに関連するルートボリュームスナップショットに対するアクセス許可を持っている場合のみです。この機能は AWS Marketplace AMI では使用できません。
- インスタンスに製品コードがない場合にのみ、製品コードなしの AMI を使用できます。
- 置換ルートボリュームのサイズは、AMI ルートボリュームブロックデバイスマッピングのサイズと現在のルートボリュームのサイズのいずれか大きい方です。
- インスタンスのインスタンス ID ドキュメントは自動的に更新されます。
- インスタンスが NitroTPM に対応している場合、インスタンスの NitroTPM データがリセットされ、新しいキーが生成されます。

既存の Amazon EBS ボリュームを使用してルートボリュームを置き換える

新しいボリュームを作成するスナップショットおよび AMI オプションとは異なり、このオプションは現在のルートボリュームを、あらかじめ設定した既存の Amazon EBS ボリュームに置き換えます。これは、データベースなどのステートフルワークロードで、ルートボリュームをインスタンスにアタッチする前にメタデータまたはソフトウェアを準備する場合に便利です。

このオプションを使用する前に、置換用の Amazon EBS ボリュームを作成して準備しておく必要があります。データボリュームとしてインスタンスにアタッチし、必要なデータをコピーしてからデタッチすることで置換用ボリュームを準備できます。ボリュームはインスタンスと同じアベイラビリティーゾーンに存在している必要があります。

置換用ルートボリュームは、作成および準備時に設定したサイズ、タイプ、およびその他の属性を保持します。置換用ルートボリュームには、元のルートボリュームの削除時の削除属性が引き継がれることはありません。

既存の Amazon EBS ボリュームを使用する際の考慮事項

- 置換用ボリュームは、インスタンスと同じアベイラビリティーゾーンに存在している必要があります。
- 置換用ボリュームは、リクエスト時にインスタンスにアタッチしないでください。
- 置換用ボリュームは、`available` 状態である必要があります。
- 置換用ボリュームがこれらの要件を満たしていない場合、ルートボリュームの置換リクエストは失敗します。
- このオプションは AWS Marketplace インスタンスで使用できますが、置換用ボリュームのマーケットプレイス製品コードと請求コードの両方がインスタンス上のものと一致する必要があります。
- 元のルートボリュームが暗号化されている場合、置換用ボリュームも暗号化する必要があります。暗号化されていないボリュームから暗号化されたボリュームに移動することはできますが、暗号化されたボリュームから暗号化されていないボリュームに移動することはできません。置換用ボリュームが暗号化されている場合、IAM プリンシパルには元のボリュームをデタッチするための AWS KMS アクセス許可が必要になります。また、置換用ボリュームをアタッチし復号するためのアクセス許可も必要です。これらは、暗号化されたボリュームをアタッチするのに必要なアクセス許可と同じです。

ルートボリュームの置換プロセスが完了した後も、元のルートボリュームを維持するかどうかを選択できます。置換プロセスの完了後に元のルートボリュームを削除すると、元のルートボリュームは自

動的に削除され、回復できなくなります。処理が完了した後も元のルートボリュームを維持することを選択した場合、ボリュームはアカウントにプロビジョニングされたままになるため、不要になったらそのボリュームを手動で削除する必要があります。

ルートボリュームの置き換えタスクは、次の状態に移行します:

- `pending` — 置換ボリュームが作成されています。
- `in-progress` — 元のボリュームがデタッチされ、置換ボリュームがアタッチされています。
- `succeeded` — 置換ボリュームはインスタンスに正常にアタッチされ、インスタンスは利用可能です。
- `failing` — 置換タスクが失敗を処理しています。
- `failed` — 置換タスクは失敗しましたが、ルートボリュームはまだアタッチされています。
- `failing-detached` — 置換タスクが正常に処理されていません。インスタンスにルートボリュームがアタッチされていない可能性があります。
- `failed-detached` — 置換タスクが失敗し、インスタンスにルートボリュームがアタッチされていません。

ルートボリュームの置き換えタスクが失敗した場合、インスタンスは再起動され、元のルートボリュームはインスタンスにアタッチされたままになります。

考慮事項

開始する前に、以下を考慮してください。

要件

- インスタンスは `running` の状態である必要があります。
- インスタンスは、プロセス中に自動的に再起動されます。メモリ (RAM) の内容は、再起動中に消去されます。手動で再起動する必要はありません。
- インスタンスストアボリュームの場合、ルートボリュームを置き換えることはできません。Amazon EBS ルートボリュームを持つインスタンスのみに対応しています。
- ルートボリュームを置き換えることができるのは、すべての仮想化インスタンスタイプと EC2 Mac ベアメタルインスタンスのみです。他のいかなるベアメタルインスタンスタイプもサポートされていません。
- スナップショットを使用する場合、使用できるのは、インスタンスの現在または以前のルートボリュームから直接作成されたスナップショットのみです。

- 現在のリージョンで、アカウントのAmazon EBS の暗号化がデフォルトで有効になっていると、指定したスナップショットや指定した AMI のルートボリュームにおける暗号化ステータスに関係なく、ルートボリューム置換タスクによって作成された置換用ルートボリュームは常に暗号化されます。

暗号化の結果

次の表は、考えられる暗号化の結果をまとめたものです。

	元のルートボリューム	指定されたスナップショット、AMI、またはボリューム	デフォルトでの暗号化	置換用ルートボリューム	置換用ルートボリュームに使用される暗号化キー
置換用ルートボリュームを起動状態に復元する	暗号化された	該当しない	考慮しない	暗号化された	元のルートボリュームと同じ KMS キー
	暗号化されていない	該当しない	Disabled	暗号化されていない	該当しない
	暗号化されていない	該当しない	有効	暗号化された	アカウントの Amazon EBS 暗号化用のデフォルト KMS キー
スナップショットまたは AMI から置換用ルートボリュームを復元する	暗号化された	暗号化されていない	考慮しない	暗号化された	元のルートボリュームと同じ KMS キー
	暗号化された	暗号化された	考慮しない	暗号化された	元のルートボリュームと同じ KMS キー
	暗号化されていない	暗号化されていない	Disabled	暗号化されていない	該当しない

	元のルートボリューム	指定されたスナップショット、AMI、またはボリューム	デフォルトでの暗号化	置換用ルートボリューム	置換用ルートボリュームに使用される暗号化キー
	暗号化されていない	暗号化されていない	有効	暗号化された	アカウントの Amazon EBS 暗号化用のデフォルト KMS キー

	元のルートボリューム	指定されたスナップショット、AMI、またはボリューム	デフォルトでの暗号化	置換用ルートボリューム	置換用ルートボリュームに使用される暗号化キー
	暗号化されていない	暗号化された	考慮しない	暗号化された	AMI またはスナップショットがアカウントで所有されている場合、置換用ボリュームは、その AMI またはスナップショットの KMS キーで暗号化されます。アカウントが AMI またはスナップショットを共有している場合、置換用ボリュームは、そのアカウントの デフォルトの Amazon EBS 暗号化の KMS キーで暗号化されます。

	元のルートボリューム	指定されたスナップショット、AMI、またはボリューム	デフォルトでの暗号化	置換用ルートボリューム	置換用ルートボリュームに使用される暗号化キー
既存の Amazon EBS ボリュームから置換用ルートボリュームを復元する	暗号化された	暗号化された	該当しない	暗号化された	指定されたボリュームの KMS キー
	暗号化された	暗号化されていない	該当しない	許可されていません。暗号化されたルートボリュームを暗号化されていないボリュームに置き換えることはできません。Amazon EC2 は置き換えタスクを拒否します。	該当しない
	暗号化されていない	暗号化されていない	該当しない	暗号化されていない	該当しない
	暗号化されていない	暗号化された	該当しない	暗号化された	指定されたボリュームの KMS キー

ルートボリュームを置き換える

インスタンスのルートボリュームを復元すると、ルートボリュームの置換タスクが作成されます。ルートボリュームの置換タスクを使用して、置換プロセスの進行状況と結果をモニタリングできます。

Console

ルートボリュームを置き換えるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. ルートボリュームを置き換えるインスタンスを選択し、[Actions] (アクション)、[Monitor and troubleshoot] (監視とトラブルシューティング)、[Replace root volume] (ルートボリュームを置換) の順に選択します。

   > **Note**
   >
   > 選択したインスタンスが `running` 状態でない場合、[ルートボリュームを置き換える] アクションは無効です。

4. [復元] には、以下のいずれかのオプションを選択します。

   - [起動状態] - 現在のルートボリュームの作成に使用されたスナップショットから置換ルートボリュームを復元します。
   - [スナップショット] - 置換ルートボリュームを、指定したスナップショットに復元します。[スナップショット] で、使用するスナップショットを選択します。
   - [イメージ] – 指定した AMI を使用して、置換ルートボリュームを復元します。[イメージ] で、使用する AMI を選択します。

5. (オプション) [ボリューム初期化レート] には、ボリューム初期化のための Amazon EBS プロビジョンドレート (ボリューム初期化レート) を MiB/秒で指定でき、スナップショットブロックはこのレートで Amazon S3 からボリュームにダウンロードされます。詳細については、「ボリュームの初期化に Amazon EBS プロビジョンドレートを使用する」を参照してください。デフォルトの初期化レート、または高速スナップショット復元 (選択したスナップショットで有効化されている場合) を使用する場合は、レートを指定しないでください。
6. (オプション) 置き換えようとしているルートボリュームを削除するには、[置き換えられるルートボリュームを削除] を選択します。
7. [置換タスクを作成] を選択します。
8. 置換タスクをモニタリングするには、インスタンスの [ストレージ] タブを選択し、[最近のルートボリューム置換タスク] を展開します。

AWS CLI

置換用ルートボリュームを起動状態に復元するには

create-replace-root-volume-task コマンドを使用します。--instance-id には、ルートボリュームを置き換えるインスタンスの ID を指定します。--snapshot-id および --image-id のパラメータは省略します。置換後に元のルートボリュームを削除するには、--delete-replaced-root-volume を含めて、true を指定してください。スナップショットブロックが Amazon S3 からボリュームにダウンロードされるボリューム初期化レートを指定するには、100 から 300 MiB/秒 までの値を --volume-initialization-rate に指定します。

```
aws ec2 create-replace-root-volume-task \
--instance-id i-1234567890abcdef0 \
--delete-replaced-root-volume \
--volume-initialization-rate 150
```

置換用ルートボリュームを特定のスナップショットに復元するには

create-replace-root-volume-task コマンドを使用します。--instance-id には、ルートボリュームを置き換えるインスタンスの ID を指定します。--snapshot-id には、使用するスナップショットの ID を指定します。置換後に元のルートボリュームを削除するには、--delete-replaced-root-volume を含めて、true を指定してください。スナップショットブロックが Amazon S3 からボリュームにダウンロードされるボリューム初期化レートを指定するには、100 から 300 MiB/秒 までの値を --volume-initialization-rate に指定します。

```
aws ec2 create-replace-root-volume-task \
--instance-id i-1234567890abcdef0 \
--snapshot-id snap-9876543210abcdef0 \
--delete-replaced-root-volume \
--volume-initialization-rate 150
```

AMI を使用して置換用ルートボリュームを復元するには

create-replace-root-volume-task コマンドを使用します。--instance-id には、ルートボリュームを置き換えるインスタンスの ID を指定します。--image-id に使用する AMI の ID を指定します。置換後に元のルートボリュームを削除するには、--delete-replaced-root-volume を含めて、true を指定してください。スナップショットブロックが Amazon S3 からボリュームにダウンロードされるボリューム初期化レートを指定するには、100 から 300 MiB/秒 までの値を --volume-initialization-rate に指定します。

```
aws ec2 create-replace-root-volume-task \
--instance-id i-1234567890abcdef0 \
--image-id ami-09876543210abcdef \
--delete-replaced-root-volume \
--volume-initialization-rate 150
```

既存の Amazon EBS ボリュームを使用してルートボリュームを置き換えるには

create-replace-root-volume-task コマンドを使用します。--instance-id には、ルートボリュームを置き換えるインスタンスの ID (例えば i-1234567890abcdef0) を指定します。--volume-id には、新しいルートボリュームとして使用する Amazon EBS ボリュームの ID を指定します。置換完了後に元のルートボリュームを削除するには、--delete-replaced-root-volume を含めます。Amazon EC2 は --volume-initialization-rate パラメータを受け入れますが、このオプションではスナップショットデータをダウンロードしないため、パラメータは無視されます。

```
aws ec2 create-replace-root-volume-task \
--instance-id i-1234567890abcdef0 \
--volume-id vol-0123456789abcdef0 \
--delete-replaced-root-volume
```

ルートボリューム置換タスクのステータスを表示するには

describe-replace-root-volume-task コマンドを使用して、表示するルートボリューム置換タスクの ID を指定します。

```
aws ec2 describe-replace-root-volume-tasks \
    --replace-root-volume-task-ids replacevol-1234567890abcdef0 \
    --query ReplaceRootVolumeTasks[].TaskState
```

以下は出力の例です。

```
[
    "succeeded"
]
```

または、instance-id フィルターを指定して、結果をインスタンス別にフィルタリングします。

```
$ aws ec2 describe-replace-root-volume-tasks \
```

```
    --filters Name=instance-id,Values=i-1234567890abcdef0
```

PowerShell

置換用ルートボリュームを起動状態に復元するには

New-EC2ReplaceRootVolumeTask コマンドを使用します。-InstanceId には、ルートボリュームを置き換えるインスタンスの ID を指定します。-SnapshotId および -ImageId のパラメータは省略します。置換後に元のルートボリュームを削除するには、-DeleteReplacedRootVolume を含めて、$true を指定してください。スナップショットブロックが Amazon S3 からボリュームにダウンロードされるボリューム初期化レートを指定するには、100 から 300 MiB/秒 までの値を -VolumeInitializationRate に指定します。

```
New-EC2ReplaceRootVolumeTask `
    -InstanceId i-1234567890abcdef0 `
    -VolumeInitializationRate 150 `
    -DeleteReplacedRootVolume $true
```

置換用ルートボリュームを特定のスナップショットに復元するには

New-EC2ReplaceRootVolumeTask コマンドを使用します。--InstanceId には、ルートボリュームを置き換えるインスタンスの ID を指定します。-SnapshotId には、使用するスナップショットの ID を指定します。置換後に元のルートボリュームを削除するには、-DeleteReplacedRootVolume を含めて、$true を指定してください。スナップショットブロックが Amazon S3 からボリュームにダウンロードされるボリューム初期化レートを指定するには、100 から 300 MiB/秒 までの値を -VolumeInitializationRate に指定します。

```
New-EC2ReplaceRootVolumeTask `
    -InstanceId i-1234567890abcdef0 `
    -SnapshotId snap-9876543210abcdef0 `
    -VolumeInitializationRate 150 `
    -DeleteReplacedRootVolume $true
```

AMI を使用して置換用ルートボリュームを復元するには

New-EC2ReplaceRootVolumeTask コマンドを使用します。-InstanceId には、ルートボリュームを置き換えるインスタンスの ID を指定します。-ImageId に使用する AMI の ID を指定します。置換後に元のルートボリュームを削除するには、-DeleteReplacedRootVolume を含めて、$true を指定してください。スナップショットブロックが Amazon S3 からボリューム

にダウンロードされるボリューム初期化レートを指定するには、100 から 300 MiB/秒 までの値を -VolumeInitializationRate に指定します。

```
New-EC2ReplaceRootVolumeTask `
    -InstanceId i-1234567890abcdef0 `
    -ImageId ami-0abcdef1234567890 `
    -VolumeInitializationRate 150 `
    -DeleteReplacedRootVolume $true
```

既存の Amazon EBS ボリュームを使用してルートボリュームを置き換えるには

New-EC2ReplaceRootVolumeTask コマンドを使用します。-InstanceId には、ルートボリュームを置き換えるインスタンスの ID (例えば i-1234567890abcdef0) を指定します。-VolumeId には、新しいルートボリュームとして使用する Amazon EBS ボリュームの ID を指定します。置換完了後に元のルートボリュームを削除するには、-DeleteReplacedRootVolume を含めるとともに、$true を指定してください。Amazon EC2 は -VolumeInitializationRate パラメータを受け入れますが、このオプションではスナップショットデータをダウンロードしないため、パラメータは無視されます。

```
New-EC2ReplaceRootVolumeTask `
    -InstanceId i-1234567890abcdef0 `
    -VolumeId vol-0123456789abcdef0 `
    -DeleteReplacedRootVolume $true
```

ルートボリューム置換タスクのステータスを表示するには

Get-EC2ReplaceRootVolumeTask コマンドを使用して、表示するルートボリューム置換タスクの ID を指定します。

```
(Get-EC2ReplaceRootVolumeTask `
    -ReplaceRootVolumeTaskIds replacevol-1234567890abcdef0).TaskState.Value
```

以下は出力の例です。

```
Succeeded
```

または、instance-id フィルターを指定して、結果をインスタンス別にフィルタリングします。

```
PS C:\> Get-EC2ReplaceRootVolumeTask -Filters @{Name = 'instance-id'; Values =
 'i-1234567890abcdef0'} | Format-Table
```

Amazon EC2 インスタンス上のボリュームのデバイス名

インスタンスにボリュームをアタッチする場合はボリュームのデバイス名を含めます。このデバイス名は Amazon EC2 によって使用されます。インスタンスのブロックデバイスドライバーはボリュームのマウント時に実際のボリューム名を割り当てますが、この割り当てられた名前はAmazon EC2 が使用する名前とは異なる可能性があります。

インスタンスがサポートできるボリュームの数はオペレーティングシステムによって決まります。詳細についてはAmazon EC2 インスタンスの Amazon EBS ボリューム制限を参照してください。

内容

- 使用できるデバイス名
- デバイス名に関する考慮事項

使用できるデバイス名

Linux インスタンス

Linux インスタンスでは準仮想化 (PV) とハードウェア仮想マシン (HVM) の 2 種類の仮想化を使用できます。インスタンスの仮想化タイプはインスタンスの起動に使用される AMI によって決まります。すべてのインスタンスタイプが HVM AMI をサポートしています。一部の旧世代のインスタンスタイプは PV AMI をサポートしています。使用できる推奨のデバイス名はインスタンスの仮想化タイプによって異なるため、必ず AMI の仮想化タイプを確認してください。詳細については「仮想化タイプ」を参照してください。

次の表に、ブロックデバイスマッピングまたは EBS ボリュームの接続時に指定できる使用可能なデバイス名を示します。

仮想化タイプ	利用可能	ルートボリューム用に予約済み	EBS データボリュームとして推奨	インスタンスストアボリューム
準仮想化	/dev/sd[a-z]	/dev/sda1	/dev/sd[f-p]	/dev/sd[b-e]

仮想化タイプ	利用可能	ルートボリューム用に予約済み	EBS データボリュームとして推奨	インスタンスストアボリューム
	/dev/sd[a-z][1-15] /dev/hd[a-z] /dev/hd[a-z][1-15]		/dev/sd[f-p][1-6]	
HVM	/dev/sd[a-z] /dev/xvd[a-c][a-z] /dev/xvdd[a-x]	AMI による違い /dev/sda1 or /dev/xvda	/dev/sd[b-z] /dev/xvdb[b-z] *	/dev/sd[b-e] /dev/sd[b-h] (h1.16xlarge) /dev/sd[b-y] (d2.8xlarge) /dev/sd[b-i] (i2.8xlarge) **

* ブロックデバイスマッピングの NVMe EBS ボリュームで指定したデバイス名はNVMe デバイス名 (`/dev/nvme[0-26]n1`) を使用して名称変更されます。ブロックデバイスドライバーはブロックデバイスマッピングのボリュームに指定した順序とは異なる順序で NVMe デバイス名を割り当てることができます。

** NVMe インスタンスストアボリュームは自動的に列挙され、NVMe デバイス名が割り当てられます。

Windows インスタンス

AWS Windows AMI は、仮想化ハードウェアへのアクセスを許可するために、次のいずれかのドライバーセットを使用します。

- AWS PV: Windows インスタンス用 Paravirtual ドライバー
- AWS NVMe: NVMe ドライバー

Nitro ベースのインスタンスのデバイス名

次の表に、ブロックデバイスマッピングで、または EBS ボリュームを Nitro ベースのインスタンスに接続する際に指定できる使用可能なデバイス名を示します。

ドライバータイプ	利用可能	ルートボリューム用に予約済み	EBS ボリュームとして推奨	インスタンスストアボリューム
AWS NVMe	xvd[a-z] xvd[a-c][a-z] xvdd[a-x] /dev/sda1	/dev/sda1	xvd[b-z] xvdb[b-z]	*

* NVMe インスタンスストアボリュームは自動的に列挙され、Windows のドライブ文字が割り当てられます。

Xen ベースのインスタンスのデバイス名

次の表に、ブロックデバイスマッピングで、または EBS ボリュームを Xen ベースのインスタンスに接続する際に指定できる使用可能なデバイス名を示します。

ドライバータイプ	利用可能	ルートボリューム用に予約済み	EBS ボリュームとして推奨	インスタンスストアボリューム
AWS PV	xvd[b-z] xvd[b-c][a-z] /dev/sda1 /dev/sd[b-e]	/dev/sda1	xvd[f-z]	xvdc[a-x] xvd[a-e]
Citrix PV (サポート終了)	xvd[b-z] xvd[b-c][a-z] /dev/sda1	/dev/sda1	xvd[f-z]	xvdc[a-x] xvd[a-e]

ドライバータイプ	利用可能	ルートボリューム用に予約済み	EBS ボリュームとして推奨	インスタンスストアボリューム
	/dev/sd[b-e]			
Red Hat PV (サポート終了)	xvd[a-z] xvd[b-c][a-z] /dev/sda1 /dev/sd[b-e]	/dev/sda1	xvd[f-p]	xvdc[a-x] xvd[a-e]

インスタンスストアボリュームの詳細については[EC2 インスタンス用のインスタンスストアの一時ブロックストレージ] を参照してください。EBS デバイスの識別方法を含む、NVMe EBS ボリューム (Nitro ベースのインスタンス) の詳細については「Amazon EBS ユーザーガイド」の「[Amazon EBS および NVMe]」を参照してください。

デバイス名に関する考慮事項

デバイス名を選択するときは以下の点を常に考慮する必要があります。

- 使用するデバイス名の末尾の部分はインスタンスの起動時に問題が発生する可能性があるため、重複しないようにする必要があります。例えば、同じインスタンスにアタッチされたボリュームには/dev/xvdf と xvdf のような組み合わせを使用しないでください。
- インスタンスストアボリュームをアタッチするために使用されたデバイス名を使用して EBS ボリュームをアタッチすることはできますが、動作を予測できない場合があるため、この方法は使用しないことを強くお勧めします。
- インスタンスの NVMe インスタンスストアボリュームの数はインスタンスのサイズによって異なります。NVMe インスタンスストアボリュームは自動的に列挙され、NVMe デバイス名 (Linux インスタンス) または Windows ドライブ文字 (Windows インスタンス) に割り当てられます。
- (Windows インスタンス) AWS Windows AMI には初めて起動したときにインスタンスを準備する追加のソフトウェアが付属しています。これは EC2Config サービス (Windows Server 2016 以前の Windows AMI) または EC2Launch (Windows Server 2016 以降) です。デバイスがドライブにマッピングされると、初期化され、マウントされます。ルートドライブは C:\ として初期化およびマウントされます。デフォルトではEBS ボリュームが Windows インスタンスにアタッチされている場合、インスタンスではいずれかのドライブ文字で表示されます。設定を変更して、独自の指定によってボリュームのドライブ文字を設定できます。インスタンスストアボリュームの場合、デ

フォルトはドライバーにより異なります。AWSPV ドライバーと Citrix PV ドライバーはインスタンスストアボリュームに Z: から A: の順番でドライブ文字を割り当てます。Red Hat ドライバーはインスタンスストアボリュームに D: から Z: の順番でドライブ文字を割り当てます。詳細については「Amazon EC2 Windows インスタンス上の Windows 起動エージェント」および「Amazon EC2 Windows インスタンスへのボリュームのアタッチとマッピング方法」を参照してください。

- (Linux インスタンス) カーネルのブロックデバイスドライバーによっては指定したものとは異なる名前でデバイスがアタッチされる可能性があります。例えば、デバイス名 (/dev/sdh) を指定した場合、デバイスの名前が /dev/xvdh や /dev/hdh に変更される場合があります。ほとんどの場合、末尾の文字は変更されません。Red Hat Enterprise Linux (および CentOS などのバリアント) の一部バージョンでは末尾の文字が変更されることがあります (/dev/sda が /dev/xvde になることがあります)。このような場合、各デバイス名の末尾の文字は同じ規則で変更されます。例えば、/dev/sdb の名前が /dev/xvdf に変更されると、/dev/sdc の名前は /dev/xvdg に変更されます。Amazon Linux は名前が変更されたデバイスに対して指定した名前のシンボリックデバイスを作成します。他のオペレーティングシステムの動作は異なる場合があります。
- (Linux インスタンス) HVM AMI では/dev/sda1 (ルートボリューム用に予約) と /dev/sda2 を除き、デバイス名の末尾に数字を使用することをサポートしていません。/dev/sda2 は使用できますが、HVM インスタンスでこのデバイスマッピングを使用することは推奨していません。
- (Linux インスタンス) PV AMI を使用する場合は末尾の数字の有無にかかわらず、同じデバイス文字を共有するボリュームをアタッチすることはできません。例えば、あるボリュームを /dev/sdc としてアタッチし、別のボリュームを /dev/sdc1 としてアタッチした場合、インスタンスは /dev/sdc のみを認識します。デバイス名の末尾に数字を使用するには同じベース文字を共有するすべてのデバイス名の末尾に数字を使用する必要があります (例:/dev/sdc1 、/dev/sdc2、/dev/sdc3)。
- (Linux インスタンス) 一部のカスタムカーネルでは使用できるものが /dev/sd[f-p] や /dev/sd[f-p][1-6] に制限されている場合があります。/dev/sd[q-z] または /dev/sd[q-z][1-6] の使用に関して問題がある場合は/dev/sd[f-p] または /dev/sd[f-p][1-6] に切り替えてみてください。

選択したデバイス名を指定するときは事前に、そのデバイス名が使用可能であることを確認します。確認しないと、そのデバイス名はすでに使用されている、とのエラーが表示されます。ディスクデバイスとそのマウントポイントを表示するにはlsblk コマンド (Linux インスタンス)、ディスク管理ユーティリティまたは diskpart コマンド (Windows インスタンス) を使用します。

Amazon EC2 インスタンスでのボリュームのブロックデバイスマッピング

起動する各インスタンスにはAmazon EBS ボリューム、またはインスタンスストアボリュームのどちらかのルートボリュームが関連付けられています。ブロックデバイスマッピングを使用すると、インスタンスの起動時にそのインスタンスにアタッチする追加の EBS ボリュームまたはインスタンスストアボリュームを指定できます。追加する EBS ボリュームは実行中のインスタンスにアタッチすることもできます。ただし、インスタンスストアボリュームをインスタンスにアタッチする方法は、インスタンスの起動時にブロックデバイスマッピングを使用してボリュームをアタッチする以外方法はありません。

内容

- ブロックデバイスマッピングの概念
- AMI へのブロックデバイスマッピングの追加
- Amazon EC2 インスタンスへのブロックデバイスマッピングの追加

ブロックデバイスマッピングの概念

ブロックデバイスは一連のバイトまたはビット (ブロック) でデータを移動するストレージデバイスです。これらのデバイスはランダムアクセスをサポートし、バッファ付き I/O を使用します。例にはハードディスク、CD-ROM ドライブ、フラッシュドライブが含まれます。ブロックデバイスは物理的にコンピュータにアタッチできます。また、コンピュータに物理的にアタッチされているかのように、リモートでアクセスすることもできます。

Amazon EC2 は2 種類のブロックデバイスをサポートしています。

- インスタンスストアボリューム (基盤となるハードウェアがインスタンスのホストコンピュータに物理的にアタッチされている仮想デバイス)
- EBS ボリューム (リモートストレージデバイス)

ブロックデバイスマッピングではインスタンスにアタッチするブロックデバイス (インスタンスストアボリュームと EBS ボリューム) を定義します。ブロックデバイスマッピングはAMI 作成プロセスの一環として、AMI から起動されるすべてのインスタンスによって使用されるように指定できます。また、インスタンスの起動時にブロックデバイスマッピングを指定することもできます。起動したインスタンスの AMI ですでに指定されているマッピングはこのマッピングによって上書きされます。インスタンスタイプによってサポートされるすべての NVMe インスタンスストアボリュームが

自動的に列挙され、インスタンスの起動時にデバイス名が割り当てられることに注意してください。それらをブロックデバイスマッピングに含めます。含めないとインスタンスは効果がありません。

コンテンツ

- ブロックデバイスマッピングのエントリ
- ブロックデバイスマッピングのインスタンスストアの注意事項
- ブロックデバイスマッピングの例
- オペレーティングシステムでデバイスを使用できるようにする方法

ブロックデバイスマッピングのエントリ

ブロックデバイスマッピングを作成するとき、インスタンスにアタッチする必要があるブロックデバイスごとに以下の情報を指定します。

- Amazon EC2 内で使用されるデバイス名。インスタンスのブロックデバイスドライバーはボリュームをマウントするときに実際のボリューム名を割り当てます。この割り当てられた名前は Amazon EC2 が推奨する名前とは異なる可能性があります。詳細については Amazon EC2 インスタンス上のボリュームのデバイス名を参照してください。

インスタンスストアボリュームの場合は次の情報も指定します。

- 仮想デバイスの名前: `ephemeral[0-23]`。インスタンスで使用できるインスタンスストアボリュームの数とサイズはインスタンスタイプによって異なります。

NVMe インスタンスストアボリュームの場合は次の情報も適用されます。

- これらのボリュームが自動的に列挙され、デバイス名が割り当てられます。それらをブロックデバイスマッピングに含めます。含めないとインスタンスは効果がありません。

EBS ボリュームの場合は次の情報も指定します。

- ブロックデバイスを作成するときに使用するスナップショットの ID (snap-xxxxxxxx)。ボリュームサイズを指定する場合、この値はオプションです。アーカイブされたスナップショットの ID は指定できません。
- ボリュームのサイズ (GiB 単位) 。指定されたサイズは指定されたスナップショットのサイズ以上である必要があります。

- インスタンス終了時にボリュームを削除するかどうか (`true` または `false`) デフォルト値はルートボリュームでは `true`、アタッチされたボリュームでは `false` です。AMI を作成するときはそのブロックデバイスマッピングがインスタンスからこの設定を継承します。インスタンスを起動するときに、AMI からこの設定を継承します。
- ボリュームタイプは汎用 SSD の場合は `gp2` および `gp3`、プロビジョンド IOPS SSD の場合は `io1` および `io2`、スループット最適化 HDD の場合は `st1`、Cold HDD の場合は `sc1`、磁気の場合は `standard` です。
- ボリュームがサポートする 1 秒あたりの入力/出力オペレーションの数 (IOPS) (`io1` および `io2` ボリュームでのみ使用)
- 一部のインスタンスタイプは、複数の EBS カードをサポートしています。EBS カードインデックスを指定することで、アタッチするボリュームの EBS カードを選択できます。詳細については、「EBS カード」を参照してください。

ブロックデバイスマッピングのインスタンスストアの注意事項

ブロックデバイスマッピングでインスタンスストアボリュームがある場合はインスタンスを AMI から起動すると、いくつかの警告が表示されます。

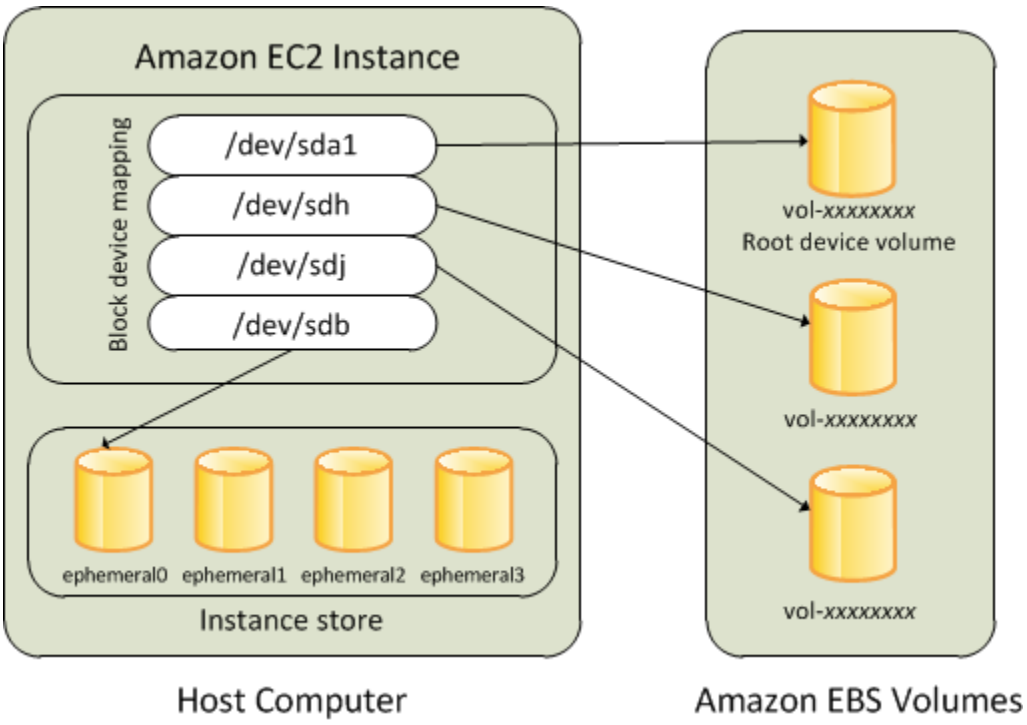
- インスタンスタイプによって中に含まれるインスタンスストアボリューム数が異なり、インスタンスストアボリュームがまったく含まれないインスタンスタイプもあります。単一インスタンスストアボリュームのみをサポートするインスタンスタイプで、AMI が 2 つのインスタンスストアボリュームにマッピングされている場合、インスタンスは単一のインスタンスストアボリュームのみで起動します。
- インスタンスストアボリュームをマッピングできるのは起動時のみに限られます。インスタンスストアボリュームのないインスタンスを停止することはできません (`t2.micro` など)。インスタンスストアボリュームをサポートするインスタンスに変更し、インスタンスストアボリュームを含めて再起動します。ただし、AMI をインスタンスから作成し、インスタンスストアボリュームをサポートするインスタンスタイプで起動して、インスタンスストアボリュームをインスタンスにマッピングすることは可能です。
- インスタンスストアボリュームをマッピングしたインスタンスを起動し、インスタンスを停止して、インスタンスストアボリュームの少ないインスタンスタイプに変更して再開すれば、最初の起動からマッピングしたインスタンスストアボリュームもインスタンスのメタデータに表示されます。ただし、インスタンスに使用できるのはそのインスタンスタイプでサポートされているインスタンスストアボリュームの最大数までです。

> **Note**
>
> インスタンスが停止されると、インスタンスストアボリュームのデータはすべて失われます。

- 起動時のインスタンスストア容量によってはM3 インスタンスが AMI インスタンスストアブロックデバイスのマッピングを (起動時に指定されていない限り) 無視します。インスタンスの起動時にインスタンスストアボリュームを使用するには起動する AMI ボリュームに AMI でインスタンスストアボリュームがマッピングされていたとしても、起動時にインスタンスストアブロックデバイスのマッピングを指定する必要があります。

ブロックデバイスマッピングの例

この図は EBS-backed インスタンスのブロックデバイスマッピングの例を示しています。この例では/dev/sdb を ephemeral0 にマッピングし、2 つの EBS ボリュームを 1 つは /dev/sdh に、もう 1 つは /dev/sdj にマッピングします。また、ルートボリュームである EBS ボリューム、/dev/sda1 も示しています。



このブロックデバイスマッピングの例はこのトピックのコマンドおよび API の例で使用されています。ブロックデバイスマッピングを作成するコマンドおよび API の例については[AMI 用のブロック

デバイスマッピングの指定およびインスタンス起動時のブロックデバイスマッピングの更新を参照してください。

オペレーティングシステムでデバイスを使用できるようにする方法

Amazon EC2 ではブロックデバイスの記述に、/dev/sdh や xvdh などのデバイス名が使われます。また、Amazon EC2 ではEC2 インスタンスにアタッチするブロックデバイスを、ブロックデバイスマッピングで指定します。ストレージデバイスにアクセスするにはインスタンスにアタッチしたブロックデバイスが、オペレーティングシステムによって事前にマウントされていなければなりません。ブロックデバイスがインスタンスからデタッチされると、そのデバイスはオペレーティングシステムによってアンマウントされ、ストレージデバイスにアクセスできなくなります。

Linux インスタンス – ブロックデバイスマッピングで指定されたデバイス名はインスタンスの初回起動時に対応するブロックデバイスにマッピングされます。デフォルトでフォーマットおよびマウントされるインスタンスストアボリュームはインスタンスタイプによって決まります。インスタンスタイプで使用できるインスタンスストアボリューム数を超えていない場合は起動時に追加のインスタンスストアボリュームをマウントできます。詳細についてはEC2 インスタンス用のインスタンスストアの一時ブロックストレージを参照してください。ボリュームがフォーマットおよびマウントされるときに使用されるデバイスはインスタンスのブロックデバイスドライバーによって決まります。

Windows インスタンス – ブロックデバイスマッピングで指定されたデバイス名はインスタンスの初回起動時に対応するブロックデバイスにマッピングされ、Ec2Config サービスによってドライブが初期化されマウントされます。ルートボリュームは C:\ としてマウントされます。インスタンスストアボリュームはZ:\、Y:\ などとしてマウントされます。EBS ボリュームについては使用可能な任意のドライブ文字を使用してマウントできます。ただし、ドライブ文字を EBS ボリュームに割り当てる方法を設定することができます。詳細については「the section called “Windows 起動エージェント”」を参照してください。

AMI へのブロックデバイスマッピングの追加

各 AMI にブロックデバイスマッピングがあります。このブロックデバイスマッピングはAMI からのインスタンスの起動時にそのインスタンスにアタッチするブロックデバイスを指定します。追加のブロックデバイスを AMI に追加するには独自の AMI を作成する必要があります。

コンテンツ

- AMI 用のブロックデバイスマッピングの指定
- AMI ブロックデバイスマッピングの EBS ボリュームの表示

AMI 用のブロックデバイスマッピングの指定

AMI を作成する場合に、ルートボリュームに加えて、ボリュームを指定するには2 つの方法があります。インスタンスから AMI を作成する前に、実行中のインスタンスにすでにボリュームをアタッチしている場合、AMI のブロックデバイスマッピングにそれらの同じボリュームが含まれます。EBS ボリュームの場合、既存のデータが新しいスナップショットに保存され、それがブロックデバイスマッピングで指定される新しいスナップショットになります。インスタンスストアボリュームの場合、データは維持されません。

EBS-backed AMI の場合、ブロックデバイスマッピングを使用して、EBS ボリュームとインスタンスストアボリュームを追加できます。Amazon S3-backed AMI の場合、イメージの登録時にイメージマニフェストファイルでブロックデバイスマッピングエントリを変更して、インスタンスストアボリュームのみを追加できます。

> **Note**
>
> M3 インスタンスの場合、インスタンスのブロックデバイスマッピングで起動時にインスタンスストアボリュームを指定する必要があります。AMI のブロックデバイスマッピングで指定したインスタンスストアボリュームはインスタンスブロックデバイスマッピングの一部として指定されていない場合、M3 インスタンスを起動した際に無視される可能性があります。

Console

AMI にボリュームを追加するには

1. Amazon EC2 コンソールを開きます。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択し、[アクション]、[イメージとテンプレート]、[イメージの作成] の順に選択してください。
4. イメージの名前と説明を入力してください。
5. インスタンスボリュームが [インスタンスボリューム] の下に表示されます。別のボリュームを追加するには[ボリュームの追加] を選択してください。
6. [Volume type] で、ボリュームタイプを選択してください。[Device] で、デバイス名を選択してください。EBS ボリュームではスナップショット、ボリュームサイズ、ボリュームタイプ、IOPS、暗号化状態などの追加の詳細を指定できます。

7. [イメージを作成] を選択してください。

AWS CLI

AMI にボリュームを追加するには

EBS-Backed AMI のブロックデバイスマッピングを指定するには create-image マンドを使用します。Amazon S3-backed AMI のブロックデバイスマッピングを指定するには register-image コマンドを使用します。

--block-device-mappings パラメータを使用してブロックデバイスマッピングを指定します。次に示すように、JSON でエンコードされた引数をコマンドラインで直接指定することも、JSON ファイルを参照して指定することもできます。

```
--block-device-mappings file://mapping.json
```

インスタンスストアボリュームを追加するには次のマッピングを使用します。NVMe インスタンスストアボリュームは自動的に追加されることに注意してください。

```
{
    "DeviceName": "device_name",
    "VirtualName": "ephemeral0"
}
```

空の 100 GiB ボリュームを追加するには次のマッピングを使用します。

```
{
    "DeviceName": "device_name",
    "Ebs": {
      "VolumeSize": 100
    }
}
```

スナップショットに基づいた EBS ボリュームを追加するには次のマッピングを使用します。

```
{
    "DeviceName": "device_name",
    "Ebs": {
      "SnapshotId": "snap-1234567890abcdef0"
```

```
    }
}
```

デバイスのマッピングを省略するには次のマッピングを使用します。

```
{
    "DeviceName": "device_name",
    "NoDevice": ""
}
```

PowerShell

New-EC2Image コマンドレットを使用して、EBS-Backed AMI のブロックデバイスマッピングを指定します。Register-EC2Image コマンドレットを使用して、Amazon S3-backed AMI のブロック デバイスマッピングを指定します。

-BlockDeviceMapping オプションを追加し、 bdm で更新を指定します。

```
-BlockDeviceMapping $bdm
```

次のマッピングは、スナップショットに基づいてボリュームを追加します。

```
$ebd = New-Object -TypeName Amazon.EC2.Model.EbsBlockDevice
$ebd.SnapshotId = "snap-1234567890abcdef0"
$bdm = New-Object -TypeName Amazon.EC2.Model.BlockDeviceMapping
$bdm.DeviceName = "device_name"
$bdm.Ebs = $ebd
```

次のマッピングは、空の 100 GB ボリュームを追加します。

```
$ebd = New-Object -TypeName Amazon.EC2.Model.EbsBlockDevice
$ebd.VolumeSize = 100
$bdm = New-Object -TypeName Amazon.EC2.Model.BlockDeviceMapping
$bdm.DeviceName = "device_name"
$bdm.Ebs = $ebd
```

AMI ブロックデバイスマッピングの EBS ボリュームの表示

AMI のブロックデバイスマッピングの EBS ボリュームを簡単に列挙できます。

Console

AMI の EBS ボリュームを表示するには

1. Amazon EC2 コンソールを開きます。
2. ナビゲーションペインで [AMI] を選択してください。
3. [Filter] リストから [EBS images] を選択して、EBS-Backed AMI のリストを取得します。
4. ご希望の AMI を選択し、[Details] タブを確認します。少なくとも、ルートボリューム (ルートデバイスという用語がルートボリュームと同等である場合) には、次の情報を使用できます。

   - ルートデバイスタイプ (ebs)
   - [ルートデバイス名] (例: /dev/sda1)
   - [Block Devices] (例: /dev/sda1=snap-1234567890abcdef0:8:true)

   AMI がブロックデバイスマッピングを使用して追加の EBS ボリュームで作成された場合、[Block Devices] フィールドにはその追加の EBS ボリュームのマッピングも表示されます。(この画面にはインスタンスストアボリュームは表示されません。)

AWS CLI

AMI の EBS ボリュームを表示するには

describe-images コマンドを使用します。

```
aws ec2 describe-images \
    --image-ids ami-0abcdef1234567890 \
    --query Image[0].BlockDeviceMappings
```

PowerShell

AMI の EBS ボリュームを表示するには

Get-EC2Image コマンドレットを使用します。

```
(Get-EC2Image -ImageId ami-0abcdef1234567890).BlockDeviceMappings
```

Amazon EC2 インスタンスへのブロックデバイスマッピングの追加

デフォルトでは起動するインスタンスにはそのインスタンスを起動した AMI のブロックデバイスマッピングで指定されたストレージデバイスが含まれます。インスタンスを起動するときに、インスタンスのブロックデバイスマッピングへの変更を指定できます。この変更は AMI のブロックデバイスマッピングを上書きするか、このブロックデバイスマッピングに統合されます。

制限事項

- ルートボリュームの場合、変更できるのはボリュームサイズ、ボリュームタイプ、および [合わせて削除] フラグのみです。
- EBS ボリュームを変更する場合、そのサイズを小さくすることはできません。そのため、指定するスナップショットのサイズはAMI のブロックデバイスマッピングで指定されたスナップショットのサイズ以上であることが必要です。

タスク

- インスタンス起動時のブロックデバイスマッピングの更新
- 実行中のインスタンスのブロックデバイスマッピングの更新
- インスタンスブロックデバイスマッピングの EBS ボリュームの表示
- インスタンスストアボリュームのインスタンスブロックデバイスマッピングの表示

インスタンス起動時のブロックデバイスマッピングの更新

インスタンスの起動時に、EBS ボリュームとインスタンスストアボリュームをインスタンスに追加できます。インスタンスのブロックデバイスマッピングを更新しても、そのインスタンスが起動された AMI のブロックデバイスマッピングは完全には変更されないことに注意してください。

Console

起動時にインスタンスのボリュームを更新するには

1. 手順に沿ってインスタンスを起動しますが、次のステップが完了しボリュームが更新されるまで、インスタンスを起動しないでください。
2. (オプション) ボリュームを追加するには、[ストレージを設定]、[新しいボリュームの追加] を選択します。ボリュームサイズとボリュームタイプを選択します。
3. (オプション) AMI のブロックデバイスマッピングで指定されたボリュームを抑制するには、[ストレージを設定]、[削除] を選択します。

4. (オプション) EBS ボリュームの設定を変更するには、[ストレージを設定] ペインで [アドバンスト] を選択します。ボリュームの情報を展開し、必要な変更を加えます。
5. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択してください。詳細については、「コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する」を参照してください。

AWS CLI

起動時にインスタンスのボリュームを更新するには

`--block-device-mappings` オプションで run-instances コマンドを使用します。

```
--block-device-mappings file://mapping.json
```

例えば、AMI ブロックデバイスマッピングが以下のように指定しているとします。

- `/dev/xvda` - EBS ルートボリューム
- `/dev/sdh` - `snap-1234567890abcdef0` から作成された EBS ボリューム
- `/dev/sdj` - サイズが `100` の空の EBS ボリューム
- `/dev/sdb` - インスタンスストアボリューム `ephemeral0`

こちらが、`mapping.json` にあるインスタンスのブロックデバイスマッピングだとします。

```
[
    {
        "DeviceName": "/dev/xvda",
        "Ebs": {
            "VolumeSize": 100
        }
    },
    {
        "DeviceName": "/dev/sdj",
        "NoDevice": ""
    },
    {
        "DeviceName": "/dev/sdh",
        "Ebs": {
            "VolumeSize": 300
        }
```

```
    },
    {
        "DeviceName": "/dev/sdc",
        "VirtualName": "ephemeral1"
    }
]
```

このインスタンスブロックデバイスマッピングは、次の操作を行います。

- ルートボリューム /dev/xvda のサイズを上書きして 100 GiB まで増やす。
- /dev/sdj がインスタンスにアタッチされることを防ぐ。
- /dev/sdh のサイズを上書きし、300 GiB に増やす。スナップショットの ID は、再度指定する必要はありません。
- エフェメラルボリューム、/dev/sdc を追加する。インスタンスタイプが複数のインスタンスストアボリュームをサポートしていない場合、これには効果がありません。インスタンスタイプが NVMe インスタンスストアボリュームをサポートしている場合、それらは自動で列挙され、インスタンスブロックデバイスマッピングに追加されて、上書きできなくなります。

PowerShell

起動時にインスタンスのボリュームを更新するには

-BlockDeviceMapping パラメータと、New-EC2Instance コマンドレットと -BlockDeviceMapping パラメータを使用します。

```
-BlockDeviceMapping $bdm
```

こちらが、$bdm にあるインスタンスのブロックデバイスマッピングだとします。

```
$bdm = @()

$root = New-Object -TypeName Amazon.EC2.Model.BlockDeviceMapping
$root.DeviceName = "/dev/xvda"
$ebs1 = New-Object -TypeName Amazon.EC2.Model.EbsBlockDevice
$ebs1.VolumeSize = 100
$root.Ebs = $ebs1
$bdm += $root

$sdj = New-Object -TypeName Amazon.EC2.Model.BlockDeviceMapping
```

```
$sdj.DeviceName = "/dev/sdj"
$sdj.NoDevice = ""
$bdm += $sdj

$sdh = New-Object -TypeName Amazon.EC2.Model.BlockDeviceMapping
$sdh.DeviceName = "/dev/sdh"
$ebs2 = New-Object -TypeName Amazon.EC2.Model.EbsBlockDevice
$ebs2.VolumeSize = 300
$sdh.Ebs = $ebs2
$bdm += $sdh

$sdc = New-Object -TypeName Amazon.EC2.Model.BlockDeviceMapping
$sdc.DeviceName = "/dev/sdc"
$sdc.VirtualName = "ephemeral1"
$bdm += $sdc
```

このインスタンスブロックデバイスマッピングは、次の操作を行います。

- ルートボリューム /dev/xvda のサイズを上書きして 100 GiB まで増やす。
- /dev/sdj がインスタンスにアタッチされることを防ぐ。
- /dev/sdh のサイズを上書きし、300 GiB に増やす。スナップショットの ID は、再度指定する必要はありません。
- エフェメラルボリューム、/dev/sdc を追加する。インスタンスタイプが複数のインスタンスストアボリュームをサポートしていない場合、これには効果がありません。インスタンスタイプが NVMe インスタンスストアボリュームをサポートしている場合、それらは自動で列挙され、インスタンスブロックデバイスマッピングに追加されて、上書きできなくなります。

実行中のインスタンスのブロックデバイスマッピングの更新

この属性を変更する前に、インスタンスを停止する必要はありません。

AWS CLI

実行中のインスタンスのブロックデバイスマッピングを更新するには

modify-instance-attribute コマンドを使用します。

--block-device-mappings オプションを追加します。

```
--block-device-mappings file://mapping.json
```

mapping.json で、更新を指定します。例えば、次の更新では、ルートボリュームが存続するように変更されます。

```
[
  {
    "DeviceName": "/dev/sda1",
    "Ebs": {
      "DeleteOnTermination": false
    }
  }
]
```

PowerShell

実行中のインスタンスのブロックデバイスマッピングを更新するには

Edit-EC2InstanceAttribute コマンドレットを使用します。

-BlockDeviceMapping オプションを追加します。

```
-BlockDeviceMapping $bdm
```

bdm で、更新を指定します。例えば、次の更新では、ルートボリュームが存続するように変更されます。

```
$ebd = New-Object -TypeName Amazon.EC2.Model.EbsBlockDevice
$ebd.DeleteOnTermination = false
$bdm = New-Object -TypeName Amazon.EC2.Model.BlockDeviceMapping
$bdm.DeviceName = "/dev/sda1"
$bdm.Ebs = $ebd
```

インスタンスブロックデバイスマッピングの EBS ボリュームの表示

インスタンスにマッピングされた EBS ボリュームを簡単に列挙できます。

Console

インスタンスの EBS ボリュームを表示するには

1. Amazon EC2 コンソールを開きます。

2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択し、[ストレージ] タブに表示された詳細を確認します。少なくとも、ルートボリューム (ルートデバイスという用語がルートボリュームと同等である場合) には、次の情報を使用できます。

   - ルートデバイスタイプ (例: EBS)
   - [ルートデバイス名] (例: /dev/xvda)
   - [ブロックデバイス] (例: /dev/xvda、/dev/sdf、/dev/sdj)

   インスタンスがブロックデバイスマッピングを使用して追加の EBS ボリュームで起動した場合は[ブロックデバイス] の下に表示されます。このタブにはインスタンスストアボリュームは表示されません。
4. EBS ボリュームに関する追加情報を表示するにはそのボリューム ID を選択して [ボリューム] ページに移動します。

AWS CLI

インスタンスの EBS ボリュームを表示するには

describe-instances コマンドを使用します。

```
aws ec2 describe-instances \
    --instance-ids i-1234567890abcdef0 \
    --query Reservations[*].Instances[0].BlockDeviceMappings
```

PowerShell

インスタンスの EBS ボリュームを表示するには

Get-EC2Instance コマンドレットを使用します。

```
(Get-EC2Instance -InstanceId i-0bac57d7472c89bac).Instances.BlockDeviceMappings
```

インスタンスストアボリュームのインスタンスブロックデバイスマッピングの表示

インスタンスタイプはインスタンスに利用できるインスタンスストアボリュームの数とタイプを決定します。ブロックデバイスマッピングのインスタンスストアボリュームの数が、インスタンスに利用

できるインスタンスストアボリュームの数を超える場合は追加のボリュームは無視されます。インスタンスのインスタンスストアボリュームを表示するにはlsblk コマンド (Linux インスタンス) を実行するか、Windows ディスク管理 (Windows インスタンス を開いてください。各インスタンスタイプでサポートされるインスタンスストアボリュームの数については「Amazon EC2 インスタンスタイプの仕様」を参照してください。

インスタンスのブロックデバイスマッピングを表示した場合、EBS ボリュームのみが表示され、インスタンスストアボリュームは表示されません。インスタンスのインスタンスストアボリュームを表示する方法はボリュームタイプによって異なります。

NVMe インスタンスストアボリューム

Linux インスタンス

ブロックデバイスマッピング内の NVMe インスタンスストアボリュームをクエリするにはNVMe コマンドラインパッケージ (nvme-cli) を使用します。パッケージをダウンロードし、インスタンスにインストールした上で、次のコマンドを実行します。

```
[ec2-user ~]$ sudo nvme list
```

インスタンスに関する出力例を次に示します。Model 列のテキストはこのボリュームが EBS ボリュームであるか、インスタンスストアボリュームであるかを示します。この例では/dev/nvme1n1 および /dev/nvme2n1 がインスタンスストアボリュームです。

```
Node             SN                   Model
 Namespace
---------------- -------------------- ----------------------------------------
 ---------
/dev/nvme0n1     vol06afc3f8715b7a597 Amazon Elastic Block Store               1

/dev/nvme1n1     AWS2C1436F5159EB6614 Amazon EC2 NVMe Instance Storage         1

/dev/nvme2n1     AWSB1F4FF0C0A6C281EA Amazon EC2 NVMe Instance Storage         1
  ...
```

Windows インスタンス

Disk Management または PowerShell を使用して、EBS とインスタンスストアの NVMe ボリュームの両方を一覧表示できます。詳細については「the section called “NVMe ディスクをボリュームにマッピングする”」を参照してください。

HDD もしくは SSD のインスタンスストアボリューム

ブロックデバイスマッピングで HDD もしくは SSD のインスタンスストアボリュームをクエリするにはインスタンスメタデータを使用します。NVMe インスタンスストアボリュームは含まれていません。

インスタンスメタデータのすべてのリクエストの基本 URI は `http://169.254.169.254/latest/` です。詳細については「インスタンスメタデータを使用して EC2 インスタンスを管理する」を参照してください。

Linux インスタンス

まず、実行中にインスタンスに接続します。インスタンスからこのクエリを使用して、そのブロックデバイスマッピングを取得します。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/block-device-mapping/
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/block-device-mapping/
```

レスポンスにはインスタンスのブロックデバイスの名前が含まれます。例えば、インスタントストアバック `m1.small` インスタンスの出力は次のようになります。

```
ami
ephemeral0
root
swap
```

`ami` デバイスはインスタンスによって判断されるルートデボリュームです。インスタンスストアボリュームの名前は `ephemeral[0-23]` です。`swap` デバイスはページファイル用です。EBS ボリュームもマップした場合、そのボリュームは`ebs1`、`ebs2` のように表示されます。

ブロックデバイスマッピングの個別のブロックデバイスの詳細を確認するにはここで示すように、前のクエリにブロックデバイスの名前を追加します。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/block-device-mapping/ephemeral0
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/block-device-mapping/ephemeral0
```

Windows インスタンス

まず、実行中にインスタンスに接続します。インスタンスからこのクエリを使用して、そのブロックデバイスマッピングを取得します。

```
PS C:\> Invoke-RestMethod -uri http://169.254.169.254/latest/meta-data/block-device-mapping/
```

レスポンスにはインスタンスのブロックデバイスの名前が含まれます。例えば、インスタントストアバック m1.small インスタンスの出力は次のようになります。

```
ami
ephemeral0
root
swap
```

ami デバイスはインスタンスによって判断されるルートデボリュームです。インスタンスストアボリュームの名前は ephemeral[0-23] です。swap デバイスはページファイル用です。EBS ボリュームもマップした場合、そのボリュームはebs1、ebs2 のように表示されます。

ブロックデバイスマッピングの個別のブロックデバイスの詳細を確認するにはここで示すように、前のクエリにブロックデバイスの名前を追加します。

```
PS C:\> Invoke-RestMethod -uri http://169.254.169.254/latest/meta-data/block-device-mapping/ephemeral0
```

Amazon EC2 Windows インスタンスへのボリュームのアタッチとマッピング方法

> **Note**
>
> このトピックは Windows インスタンスにのみ当てはまります。

EBS ボリュームを持つ Windows インスタンスは、ルートボリュームとして機能します。Windows インスタンスが AWS PV または Citrix PV ドライバーを使用している場合、オプションで最大 25 個のボリュームを追加し、合計 26 個のボリュームを作成できます。詳細については、Amazon EC2 インスタンスの Amazon EBS ボリューム制限を参照してください。

インスタンスのインスタンスタイプによって、0 から 24 のインスタンスストアボリュームをインスタンスに使用できます。インスタンスで使用できる任意のインスタンスストアボリュームを使用するには、AMI の作成時またはインスタンスの起動時にそれらを指定する必要があります。インスタンスが実行中は、AMI の作成時、またはインスタンスの起動時に EBS ボリュームを追加するか、アタッチすることもできます。

インスタンスにボリュームを追加するときに、Amazon EC2 が使用するデバイス名を指定します。詳細については、「Amazon EC2 インスタンス上のボリュームのデバイス名AWS」を参照してください。Windows Amazon マシンイメージ (AMI) には、Amazon EC2 でインスタンスストアおよび EBS ボリュームを Windows ディスクおよびドライブ文字にマップするのに使用するドライバー一式が含まれています。

ディスクを EBS ボリュームにマッピングする方法

- Amazon EC2 Windows インスタンスの NVMe ディスクをボリュームにマッピングする
- Amazon EC2 Windows インスタンスの非 NVMe ディスクをボリュームにマッピングする

Amazon EC2 Windows インスタンスの NVMe ディスクをボリュームにマッピングする

Nitro ベースのインスタンスでは、EBS ボリュームは NVMe デバイスとして公開されます。このトピックでは、インスタンスの Windows オペレーティングシステムで使用できる NVMe ディスクを表示する方法について説明します。また、これらの NVMe ディスクを基礎となる Amazon EBS ボ

リューム、および Amazon EC2 で使用されるブロックデバイスマッピングに指定されたデバイス名にマッピングする方法も示します。

トピック

- NVMe ディスクを一覧表示する
- NVMe ディスクをボリュームにマッピングする

NVMe ディスクを一覧表示する

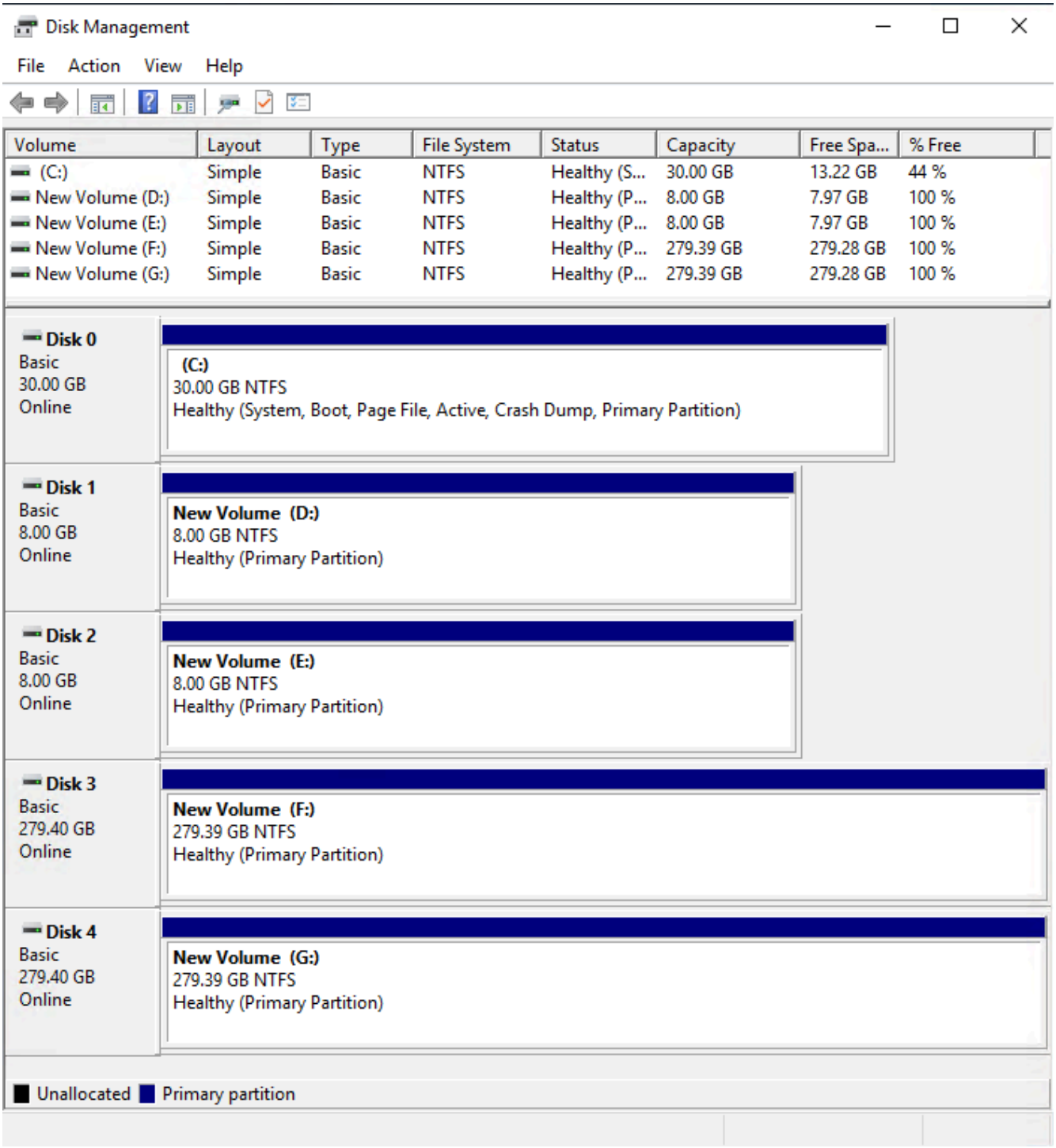
Disk Management または Powershell を使用して Windows インスタンス上のディスクを検索できます。

Disk Management

Windows インスタンス上のディスクを見つけるには

1. リモートデスクトップを使用して Windows インスタンスにログインします。詳細については、RDP を使用した Windows インスタンスへの接続を参照してください。
2. [Disk Management] ユーティリティを起動します。
3. ディスクを確認します。ルートボリュームは、`C:\` としてマウントされた EBS ボリュームです。他に表示されているディスクがない場合は、AMI を作成したとき、またはインスタンスを起動したときに追加のボリュームを指定しませんでした。

   以下は、2 つの追加の EBS ボリュームで、`r5d.4xlarge` インスタンスを起動した場合に使用可能なディスクの例です。



PowerShell

次の PowerShell スクリプトでは、各ディスクと対応するデバイス名およびボリュームを一覧表示できます。NVMe EBS とインスタンスストアボリュームを使用する Nitro ベースのインスタンスで使用することを目的としています。

Windows インスタンスに接続し、次のコマンドを実行して PowerShell スクリプトの実行を有効にします。

```
Set-ExecutionPolicy RemoteSigned
```

次のスクリプトをコピーし、Windows インスタンスに mapping.ps1 として保存します。

```
# List the disks for NVMe volumes

function Get-EC2InstanceMetadata {
    param([string]$Path)
    (Invoke-WebRequest -Uri "http://169.254.169.254/latest/$Path").Content
}

function GetEBSVolumeId {
    param($Path)
    $SerialNumber = (Get-Disk -Path $Path).SerialNumber
    if($SerialNumber -clike 'vol*'){
        $EbsVolumeId = $SerialNumber.Substring(0,20).Replace("vol","vol-")
    }
    else {
       $EbsVolumeId = $SerialNumber.Substring(0,20).Replace("AWS","AWS-")
    }
    return $EbsVolumeId
}

function GetDeviceName{
    param($EbsVolumeId)
    if($EbsVolumeId -clike 'vol*'){

        $Device  = ((Get-EC2Volume -VolumeId $EbsVolumeId ).Attachment).Device
        $VolumeName = ""
    }
     else {
        $Device = "Ephemeral"
        $VolumeName = "Temporary Storage"
    }
    Return $Device,$VolumeName
}

function GetDriveLetter{
    param($Path)
    $DiskNumber =  (Get-Disk -Path $Path).Number
```

```
    if($DiskNumber -eq 0){
        $VirtualDevice = "root"
        $DriveLetter = "C"
        $PartitionNumber = (Get-Partition -DriveLetter C).PartitionNumber
    }
    else
    {
        $VirtualDevice = "N/A"
        $DriveLetter = (Get-Partition -DiskNumber $DiskNumber).DriveLetter
        if(!$DriveLetter)
        {
            $DriveLetter = ((Get-Partition -DiskId $Path).AccessPaths).Split(",")[0]
        }
        $PartitionNumber = (Get-Partition -DiskId $Path).PartitionNumber
    }

    return $DriveLetter,$VirtualDevice,$PartitionNumber

}

$Report = @()
foreach($Path in (Get-Disk).Path)
{
    $Disk_ID = ( Get-Partition -DiskId $Path).DiskId
    $Disk = ( Get-Disk -Path $Path).Number
    $EbsVolumeId  = GetEBSVolumeId($Path)
    $Size =(Get-Disk -Path $Path).Size
    $DriveLetter,$VirtualDevice, $Partition = (GetDriveLetter($Path))
    $Device,$VolumeName = GetDeviceName($EbsVolumeId)
    $Disk = New-Object PSObject -Property @{
      Disk          = $Disk
      Partitions    = $Partition
      DriveLetter   = $DriveLetter
      EbsVolumeId   = $EbsVolumeId
      Device        = $Device
      VirtualDevice = $VirtualDevice
      VolumeName= $VolumeName
    }
 $Report += $Disk
}

$Report | Sort-Object Disk | Format-Table -AutoSize -Property Disk, Partitions,
 DriveLetter, EbsVolumeId, Device, VirtualDevice, VolumeName
```

スクリプトを次のように実行します。

```
PS C:\> .\mapping.ps1
```

次に、ルートボリューム、2 つの EBS ボリューム、および 2 つのインスタンスストアボリュームを持つインスタンスの出力例を示します。

```
Disk Partitions DriveLetter EbsVolumeId           Device    VirtualDevice VolumeName
---- ---------- ----------- -----------           ------    ------------- ----------
   0          1 C           vol-03683f1d861744bc7 /dev/sda1 root
   1          1 D           vol-082b07051043174b9 xvdb      N/A
   2          1 E           vol-0a4064b39e5f534a2 xvdc      N/A
   3          1 F           AWS-6AAD8C2AEEE1193F0 Ephemeral N/A           Temporary
 Storage
   4          1 G           AWS-13E7299C2BD031A28 Ephemeral N/A           Temporary
 Storage
```

Windows インスタンスで Tools for Windows PowerShell の認証情報を設定しなかった場合、スクリプトは EBS ボリューム ID を取得できず、`EbsVolumeId` 列に N/A が使用されます。

NVMe ディスクをボリュームにマッピングする

Get-Disk コマンドを使用することで、Windows ディスク番号を Amazon EBS ボリュームと Amazon EC2 インスタンスストアボリュームにマッピングできます。

```
PS C:\> Get-Disk
Number Friendly Name Serial Number                    HealthStatus
 OperationalStatus      Total Size Partition

                      Style
------ ------------- -------------                    ------------
 -----------------      ---------- ----------
3      NVMe Amazo... AWS6AAD8C2AEEE1193F0_00000001.   Healthy              Online
              279.4 GB MBR
4      NVMe Amazo... AWS13E7299C2BD031A28_00000001.   Healthy              Online
              279.4 GB MBR
2      NVMe Amazo... vol0a4064b39e5f534a2_00000001.   Healthy              Online
                8 GB MBR
0      NVMe Amazo... vol03683f1d861744bc7_00000001.   Healthy              Online
                30 GB MBR
```

```
1      NVMe Amazo... vol082b07051043174b9_00000001.   Healthy              Online
                  8 GB MBR
```

ebsnvme-id コマンドを実行して、NVMe ディスク番号を EBS ボリューム ID およびデバイス名にマッピングすることもできます。

```
PS C:\> C:\PROGRAMDATA\Amazon\Tools\ebsnvme-id.exe
Disk Number: 0
Volume ID: vol-03683f1d861744bc7
Device Name: sda1

Disk Number: 1
Volume ID: vol-082b07051043174b9
Device Name: xvdb

Disk Number: 2
Volume ID: vol-0a4064b39e5f534a2
Device Name: xvdc
```

Amazon EC2 Windows インスタンスの非 NVMe ディスクをボリュームにマッピングする

AWS PV または Citrix PV ドライバーを使用する Windows AMI から起動されたインスタンスの場合、このページ記載された関係を使用して、Windows ディスクをインスタンスストアおよび EBS ボリュームにマッピングできます。このトピックでは、インスタンスの Windows オペレーティングシステムで使用できる非 NVMe ディスクを表示する方法について説明します。また、これらの非 NVMe ディスクを基礎となる Amazon EBS ボリューム、および Amazon EC2 で使用されるブロックデバイスマッピングに指定されたデバイス名にマッピングする方法も示します。

> **Note**
>
> インスタンスを起動する場合 Windows AMI で Red Hat PV ドライバーを使用している場合、Citrix ドライバーを使用するようにインスタンスを更新できます。詳細については、「the section called “PV ドライバーのアップグレード”」を参照してください。

トピック

- 非 NVMe ディスクを一覧表示する

• 非 NVMe ディスクをボリュームにマッピングする

非 NVMe ディスクを一覧表示する

Disk Management または PowerShell を使用して、Windows インスタンス上のディスクを確認できます。

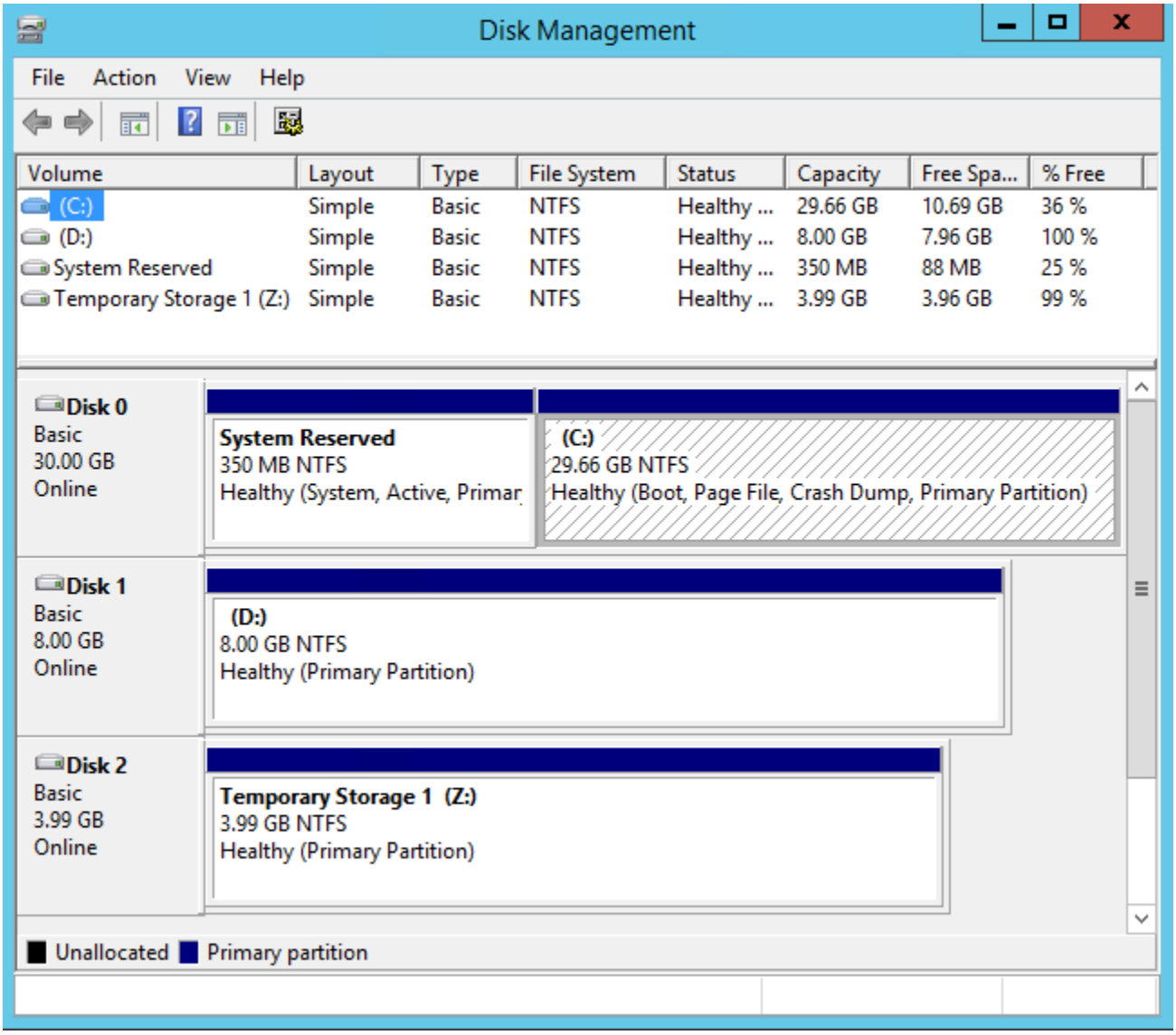
Disk Management

Windows インスタンス上のディスクを見つけるには

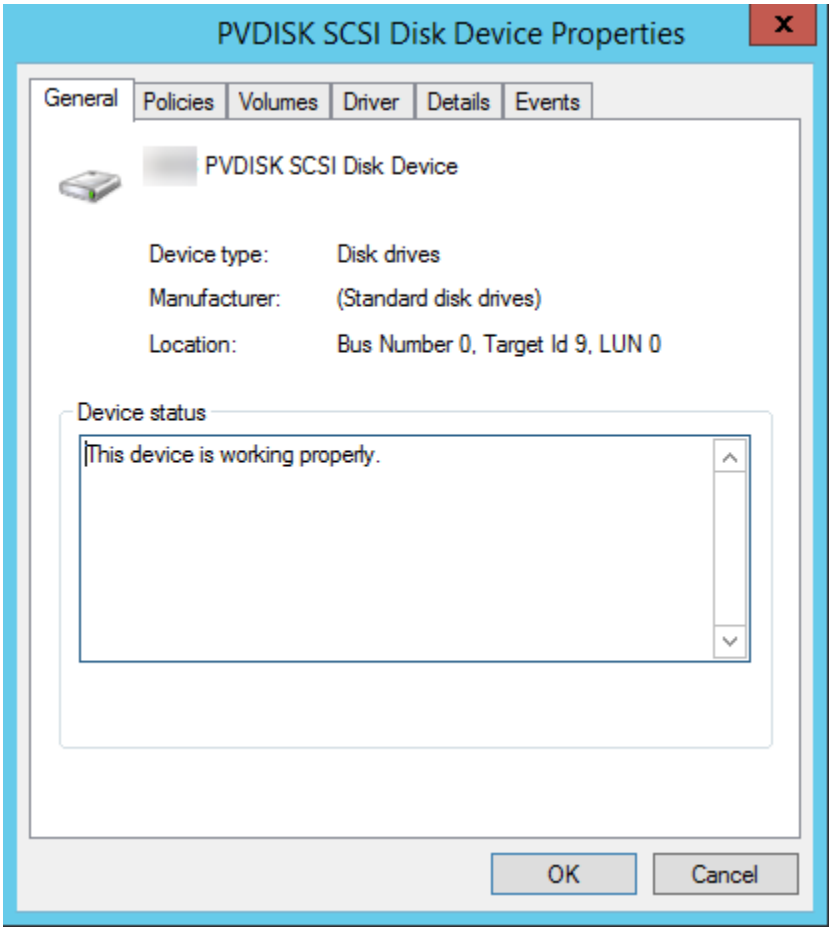
1. リモートデスクトップを使用して Windows インスタンスにログインします。詳細については、RDP を使用した Windows インスタンスへの接続を参照してください。
2. [Disk Management] ユーティリティを起動します。

   タスクバーで Windows ロゴを右クリックし、[ディスクの管理] を選択します。
3. ディスクを確認します。ルートボリュームは、`C:\` としてマウントされた EBS ボリュームです。他に表示されているディスクがない場合は、AMI を作成したとき、またはインスタンスを起動したときに追加のボリュームを指定しませんでした。

   以下は、インスタンスストアボリューム (ディスク 2) と追加の EBS ボリューム (ディスク 1) で、`m3.medium` インスタンスを起動した場合に使用可能なディスクの例です。



4. [Disk 1] というラベルが付けられた灰色のペインを右クリックし、[Properties] を選択します。[Location] の値を書き留め、非 NVMe ディスクをボリュームにマッピングする のテーブルで調べます。例えば、次のディスクに Bus Number 0, Target Id 9, LUN 0 という場所があるとします。EBS のテーブルから、この場所のデバイス名は `xvdj` であることがわかります。



PowerShell

次の PowerShell スクリプトでは、各ディスクと対応するデバイス名およびボリュームを一覧表示できます。

要件と制限

- Windows Server 2012 以降が必要です。
- EBS ボリューム ID を取得するには認証情報が必要です。Tools for PowerShell を使用してプロファイルを設定するか、インスタンスに IAM ロールをアタッチできます。
- NVMe ボリュームをサポートしません。
- ダイナミックディスクはサポートしません。

Windows インスタンスに接続し、次のコマンドを実行して PowerShell スクリプトの実行を有効にします。

```
Set-ExecutionPolicy RemoteSigned
```

次のスクリプトをコピーし、Windows インスタンスに mapping.ps1 として保存します。

```
# List the disks
function Convert-SCSITargetIdToDeviceName {
  param([int]$SCSITargetId)
  If ($SCSITargetId -eq 0) {
    return "sda1"
  }
  $deviceName = "xvd"
  If ($SCSITargetId -gt 25) {
    $deviceName += [char](0x60 + [int]($SCSITargetId / 26))
  }
  $deviceName += [char](0x61 + $SCSITargetId % 26)
  return $deviceName
}

[string[]]$array1 = @()
[string[]]$array2 = @()
[string[]]$array3 = @()
[string[]]$array4 = @()

Get-WmiObject Win32_Volume | Select-Object Name, DeviceID | ForEach-Object {
  $array1 += $_.Name
  $array2 += $_.DeviceID
}

$i = 0
While ($i -ne ($array2.Count)) {
  $array3 += ((Get-Volume -Path $array2[$i] | Get-Partition | Get-
Disk).SerialNumber) -replace "_[^ ]*$" -replace "vol", "vol-"
  $array4 += ((Get-Volume -Path $array2[$i] | Get-Partition | Get-
Disk).FriendlyName)
  $i ++
}

[array[]]$array = $array1, $array2, $array3, $array4

Try {
  $InstanceId = Get-EC2InstanceMetadata -Category "InstanceId"
  $Region = Get-EC2InstanceMetadata -Category "Region" | Select-Object -
ExpandProperty SystemName
```

```
}
Catch {
  Write-Host "Could not access the instance Metadata using AWS Get-
EC2InstanceMetadata CMDLet.
Verify you have AWSPowershell SDK version '3.1.73.0' or greater installed and
 Metadata is enabled for this instance." -ForegroundColor Yellow
}
Try {
  $BlockDeviceMappings = (Get-EC2Instance -Region $Region -Instance
 $InstanceId).Instances.BlockDeviceMappings
  $VirtualDeviceMap = (Get-EC2InstanceMetadata -Category
 "BlockDeviceMapping").GetEnumerator() | Where-Object { $_.Key -ne "ami" }
}
Catch {
  Write-Host "Could not access the AWS API, therefore, VolumeId is not available.
Verify that you provided your access keys or assigned an IAM role with adequate
 permissions." -ForegroundColor Yellow
}

Get-disk | ForEach-Object {
  $DriveLetter = $null
  $VolumeName = $null
  $VirtualDevice = $null
  $DeviceName = $_.FriendlyName

  $DiskDrive = $_
  $Disk = $_.Number
  $Partitions = $_.NumberOfPartitions
  $EbsVolumeID = $_.SerialNumber -replace "_[^ ]*$" -replace "vol", "vol-"
  if ($Partitions -ge 1) {
    $PartitionsData = Get-Partition -DiskId $_.Path
    $DriveLetter = $PartitionsData.DriveLetter | Where-object { $_ -notin @("",
 $null) }
    $VolumeName = (Get-PSDrive | Where-Object { $_.Name -in
 @($DriveLetter) }).Description | Where-object { $_ -notin @("", $null) }
  }
  If ($DiskDrive.path -like "*PROD_PVDISK*") {
    $BlockDeviceName = Convert-SCSITargetIdToDeviceName((Get-WmiObject -Class
 Win32_Diskdrive | Where-Object { $_.DeviceID -eq ("\\.\PHYSICALDRIVE" +
 $DiskDrive.Number) }).SCSITargetId)
    $BlockDeviceName = "/dev/" + $BlockDeviceName
    $BlockDevice = $BlockDeviceMappings | Where-Object { $BlockDeviceName -like "*"
 + $_.DeviceName + "*" }
    $EbsVolumeID = $BlockDevice.Ebs.VolumeId
```

```
    $VirtualDevice = ($VirtualDeviceMap.GetEnumerator() | Where-Object { $_.Value -
eq $BlockDeviceName }).Key | Select-Object -First 1
  }
  ElseIf ($DiskDrive.path -like "*PROD_AMAZON_EC2_NVME*") {
    $BlockDeviceName = (Get-EC2InstanceMetadata -Category
 "BlockDeviceMapping")."ephemeral$((Get-WmiObject -Class Win32_Diskdrive | Where-
Object { $_.DeviceID -eq ("\\.\PHYSICALDRIVE" + $DiskDrive.Number) }).SCSIPort - 2)"
    $BlockDevice = $null
    $VirtualDevice = ($VirtualDeviceMap.GetEnumerator() | Where-Object { $_.Value -
eq $BlockDeviceName }).Key | Select-Object -First 1
  }
  ElseIf ($DiskDrive.path -like "*PROD_AMAZON*") {
    if ($DriveLetter -match '[^a-zA-Z0-9]') {
      $i = 0
      While ($i -ne ($array3.Count)) {
        if ($array[2][$i] -eq $EbsVolumeID) {
          $DriveLetter = $array[0][$i]
          $DeviceName = $array[3][$i]
        }
        $i ++
      }
    }
    $BlockDevice = ""
    $BlockDeviceName = ($BlockDeviceMappings | Where-Object { $_.ebs.VolumeId -eq
 $EbsVolumeID }).DeviceName
  }
  ElseIf ($DiskDrive.path -like "*NETAPP*") {
    if ($DriveLetter -match '[^a-zA-Z0-9]') {
      $i = 0
      While ($i -ne ($array3.Count)) {
        if ($array[2][$i] -eq $EbsVolumeID) {
          $DriveLetter = $array[0][$i]
          $DeviceName = $array[3][$i]
        }
        $i ++
      }
    }
    $EbsVolumeID = "FSxN Volume"
    $BlockDevice = ""
    $BlockDeviceName = ($BlockDeviceMappings | Where-Object { $_.ebs.VolumeId -eq
 $EbsVolumeID }).DeviceName
  }
  Else {
    $BlockDeviceName = $null
```

```
    $BlockDevice = $null
  }
  New-Object PSObject -Property @{
    Disk          = $Disk;
    Partitions    = $Partitions;
    DriveLetter   = If ($DriveLetter -eq $null) { "N/A" } Else { $DriveLetter };
    EbsVolumeId   = If ($EbsVolumeID -eq $null) { "N/A" } Else { $EbsVolumeID };
    Device        = If ($BlockDeviceName -eq $null) { "N/A" } Else
 { $BlockDeviceName };
    VirtualDevice = If ($VirtualDevice -eq $null) { "N/A" } Else { $VirtualDevice };
    VolumeName    = If ($VolumeName -eq $null) { "N/A" } Else { $VolumeName };
    DeviceName    = If ($DeviceName -eq $null) { "N/A" } Else { $DeviceName };
  }
} | Sort-Object Disk | Format-Table -AutoSize -Property Disk, Partitions,
 DriveLetter, EbsVolumeId, Device, VirtualDevice, DeviceName, VolumeName
```

スクリプトを次のように実行します。

```
PS C:\> .\mapping.ps1
```

出力例を次に示します。

```
Disk  Partitions  DriveLetter   EbsVolumeId             Device      VirtualDevice
 DeviceName              VolumeName
----  ----------  -----------   -----------             ------      -------------
 ----------              ----------
   0           1             C   vol-0561f1783298efedd   /dev/sda1   N/A
 NVMe Amazon Elastic B   N/A
   1           1             D   vol-002a9488504c5e35a   xvdb        N/A
 NVMe Amazon Elastic B   N/A
   2           1             E   vol-0de9d46fcc907925d   xvdc        N/A
 NVMe Amazon Elastic B   N/A
```

Windows インスタンスで認証情報を指定しなかった場合、スクリプトは EBS ボリューム ID を取得できず、EbsVolumeId 列で N/A を使用します。

非 NVMe ディスクをボリュームにマッピングする

インスタンスのブロックデバイスドライバーは、ボリュームをマウントするときに実際のボリューム名を割り当てます。

Mappings

- インスタンスストアボリューム
- EBS ボリューム

インスタンスストアボリューム

次の表は、Citrix PV および AWS PV ドライバーが non-NVMe インスタンスストアボリュームを Windows ボリュームにどのようにマップするかを示しています。使用できるインスタンスストアボリュームの数は、インスタンスタイプによって決まります。詳細については、EC2 インスタンスのインスタンスストアボリュームの制限を参照してください。

場所	デバイス名
Bus Number 0, Target ID 78, LUN 0	xvdca
Bus Number 0, Target ID 79, LUN 0	xvdcb
Bus Number 0, Target ID 80, LUN 0	xvdcc
Bus Number 0, Target ID 81, LUN 0	xvdcd
Bus Number 0, Target ID 82, LUN 0	xvdce
Bus Number 0, Target ID 83, LUN 0	xvdcf
Bus Number 0, Target ID 84, LUN 0	xvdcg
Bus Number 0, Target ID 85, LUN 0	xvdch
Bus Number 0, Target ID 86, LUN 0	xvdci
Bus Number 0, Target ID 87, LUN 0	xvdcj
Bus Number 0, Target ID 88, LUN 0	xvdck
Bus Number 0, Target ID 89, LUN 0	xvdcl

EBS ボリューム

次の表は、Citrix PV および AWS PV ドライバーが非 NVME EBS ボリュームを Windows ボリュームにどのようにマップするかを示しています。

場所	デバイス名
Bus Number 0, Target ID 0, LUN 0	/dev/sda1
Bus Number 0, Target ID 1, LUN 0	xvdb
Bus Number 0, Target ID 2, LUN 0	xvdc
Bus Number 0, Target ID 3, LUN 0	xvdd
Bus Number 0, Target ID 4, LUN 0	xvde
Bus Number 0, Target ID 5, LUN 0	xvdf
Bus Number 0, Target ID 6, LUN 0	xvdg
Bus Number 0, Target ID 7, LUN 0	xvdh
Bus Number 0, Target ID 8, LUN 0	xvdi
Bus Number 0, Target ID 9, LUN 0	xvdj
Bus Number 0, Target ID 10, LUN 0	xvdk
Bus Number 0, Target ID 11, LUN 0	xvdl
Bus Number 0, Target ID 12, LUN 0	xvdm
Bus Number 0, Target ID 13, LUN 0	xvdn
Bus Number 0, Target ID 14, LUN 0	xvdo
Bus Number 0, Target ID 15, LUN 0	xvdp
Bus Number 0, Target ID 16, LUN 0	xvdq
Bus Number 0, Target ID 17, LUN 0	xvdr

場所	デバイス名
Bus Number 0, Target ID 18, LUN 0	xvds
Bus Number 0, Target ID 19, LUN 0	xvdt
Bus Number 0, Target ID 20, LUN 0	xvdu
Bus Number 0, Target ID 21, LUN 0	xvdv
Bus Number 0, Target ID 22, LUN 0	xvdw
Bus Number 0, Target ID 23, LUN 0	xvdx
Bus Number 0, Target ID 24, LUN 0	xvdy
Bus Number 0, Target ID 25, LUN 0	xvdz

Amazon EC2 Linux インスタンスでの Torn Write Prevention

> **Note**
>
> Torn Write Prevention は Linux インスタンスでのみサポートされています。

Torn Write Prevention はデータの回復力に悪影響を及ぼすことなく、I/O 集約型リレーショナルデータベースのワークロードのパフォーマンスを向上させ、待ち時間を短縮するために AWS によって設計されたブロックストレージ機能です。MySQL や MariaDB など、データベースエンジンとして InnoDB や XtraDB を使用するリレーショナルデータベースはTorn Write Prevention が有効です。

通常、ストレージデバイスの電源障害発生量を超えるページを使用するリレーショナルデータベースではデータロギングメカニズムを使用して書き込み操作が中断されるのを防ぎます。MariaDB と MySQL はデータテーブルに書き込む前に二重書き込みバッファファイルを使用してデータを口グに記録します。書き込み処理中にオペレーティングシステムがクラッシュしたり、停電したりして、書き込みが不完全になる、または中断された場合、データベースは二重書き込みバッファからデータを回復できます。二重書き込みバッファへの書き込みに伴い追加で発生する I/O オーバーヘッドはデータベースのパフォーマンスとアプリケーションのレイテンシーに影響を与え、1 秒あたりに処理

できるトランザクション数が減少します。二重書き込みバッファの詳細については「MariaDB」と「MySQL」のドキュメントを参照してください。

Torn Write Prevention 機能ではデータがオールオアナッシングの書き込みトランザクションでストレージに書き込まれるため、二重書き込みバッファを使用する必要がなくなります。これにより、オペレーティングシステムがクラッシュしたり、書き込みトランザクション中に停電したりした場合に、データの一部または破損したデータがストレージに書き込まれるのを防ぎます。ワークロードの回復力を損なうことなく、1 秒あたりに処理されるトランザクション数を最大 30% 増やし、書き込みレイテンシーを最大 50% 削減できます。

料金

Torn Write Prevention 機能の使用に、追加料金はかかりません。

内容

- Amazon EC2 での Torn Write Prevention のブロックサイズ
- Amazon EC2 で Torn Write Prevention を使用するための要件
- Amazon EC2 インスタンスでの Torn Write Prevention のサポートを確認する
- Amazon EC2 でのワークロードを Torn Write Prevention 用に設定する

Amazon EC2 での Torn Write Prevention のブロックサイズ

Torn Write Prevention は4 KiB、8 KiB、16 KiB のデータブロックの書き込み操作をサポートします。データブロック開始論理ブロックアドレス (LBA) は4 KiB、8 KiB、または 16 KiB のそれぞれのブロック境界サイズに合わせる必要があります。例えば、16 KiB の書き込み操作ではデータブロック開始 LBA を 16 KiB のブロック境界サイズに合わせる必要があります。

次の表はストレージとインスタンスタイプによるサポートを示しています。

	4 KiB ブロック	8 KiB ブロック	16 KiB ブロック
インスタンスストアボリューム	現行世代の I-family インスタンスにアタッチされたすべての NVMe インスタンスストアボリューム。	I4i、Im4gn、Is4gen、I7i、I7ie、I8g、および I8ge インスタンスが AWS Nitro SSD でサポートされています。	

	4 KiB ブロック	8 KiB ブロック	16 KiB ブロック
Amazon EBS ボリューム	Nitro ベースのインスタンスにアタッチされているすべての Amazon EBS ボリューム。		

インスタンスとボリュームが Torn Write Prevention をサポートしているかどうかを確認するにはクエリを実行して、インスタンスが Torn Write Prevention をサポートしているかどうか、およびサポートされているブロックサイズや境界サイズなどのその他の詳細を確認します。詳細については「Amazon EC2 インスタンスでの Torn Write Prevention のサポートを確認する」を参照してください。

Amazon EC2 で Torn Write Prevention を使用するための要件

Torn Write Prevention が正しく機能するにはNTWPU、NTWGU、NTWBU フィールドで指定されているサイズ、配置、および境界の要件を I/O 操作が満たしている必要があります。デバイスに送信する前に、特定のストレージサブシステム (ファイルシステム、LVM、RAID など) がストレージスタックの I/O プロパティ (ブロックマージ、分割、ブロックアドレスの再配置など) を変更しないように、オペレーティングシステムを設定する必要があります。

Torn Write Prevention は次の設定でテスト済みです。

- 必要なブロックサイズをサポートするインスタンスタイプとストレージタイプ。
- カーネルバージョン 5.10 以降の Amazon Linux 2。
- `bigalloc` が有効になっていて、クラスターサイズが 16 KiB で、最新の ext4 ユーティリティ (e2fsprogs 1.46.5 以降) を使用している ext4。
- Linux カーネルバッファキャッシュをバイパスする `O_DIRECT` ファイルアクセスモード。

> **Note**
>
> MySQL と MariaDB のワークロードの I/O マージを無効にする必要はありません。

Amazon EC2 インスタンスでの Torn Write Prevention のサポートを確認する

インスタンスとボリュームが Torn Write Prevention をサポートしているかどうかを確認し、Torn Write Prevention に関する情報を含む NVMe 名前空間のベンダー固有のデータを表示するには次のコマンドを使用します。

```
$ sudo nvme id-ns -v device_name
```

> **Note**
>
> このコマンドはベンダー固有の情報を ASCII 解釈の 16 進数で返します。出力の読み取りと解析ができる ebsnvme-id と似たツールをアプリケーションに組み込む必要がある場合があります。

例えば、次のコマンドは /dev/nvme1n1 の Torn Write Prevention に関する情報を含む NVMe 名前空間のベンダー固有のデータを返します。

```
$ sudo nvme id-ns -v /dev/nvme1n1
```

インスタンスとボリュームが Torn Write Prevention をサポートしている場合、NVMe 名前空間のベンダー固有データに次の AWS Torn Write Prevention 情報が返されます。

> **Note**
>
> 次の表のバイトは NVMe 名前空間のベンダー固有データの先頭からのオフセットをバイト単位で表しています。

バイト	説明
0:31	例えば /dev/xvda 、デバイス接続マウントポイントの名前。ボリュームアタッチメントリクエスト時にこれを指定すると、Amazon EC2 インスタンスが NVMe ブロックデバイス (nvmeXn1) へのシンボリックリンクを作成するために使用できます。

バイト	説明
32:63	ボリューム ID。例えば、vol01234567890abcdef　。このフィールドを使用して、NVMe デバイスを接続されたボリュームにマッピングできます。
64:255	将来の利用のために予約されています。
256:257	Torn Write Prevention 名前空間ユニットサイズ (NTWPU)。このフィールドは停電またはエラー状態時に NVM にアトミックに書き込まれることが保証されている書き込み操作の名前空間固有のサイズを示します。このフィールドは0 ベース値で表される論理ブロックで指定されます。
258:259	Torn Write Prevention 名前空間粒度サイズ (NTWPG)。このフィールドは停電またはエラー状態時に NVM にアトミックに書き込まれることが保証されている書き込み操作の NTWPU 以下の名前空間固有のサイズ増分を示します。つまり、サイズは NTWPG * n <= NTWPU で、n は正の整数でなければなりません。書き込み操作の LBA オフセットもこのフィールドに合わせる必要があります。このフィールドは0 ベース値で表される論理ブロックで指定されます。
260:263	Torn Write Prevention 名前空間境界サイズ (NTWPB)。このフィールドはこの名前空間の NTWPU 値のアトミック境界サイズを示します。この名前空間への書き込みがアトミック境界を越える場合、停電やエラー時に NVM への書き込みがアトミックに行われることは保証されません。0h の値が、停電やエラー時にアトミックな境界がないことを示します。他のすべての値はNTWPU フィールドと同じエンコーディングを使用して論理ブロック単位でサイズを指定します。

Amazon EC2 でのワークロードを Torn Write Prevention 用に設定する

Torn Write Prevention はサポートされているボリュームを持つサポートされているインスタンスタイプではデフォルトで有効になっています。Torn Write Prevention を有効にするために、追加の設定を行う必要はありません。

> **Note**
>
> Torn Write Prevention に対応していないワークロードについては、パフォーマンスへの影響はありません。これらのワークロードについては、変更を加える必要はありません。
> Torn Write Prevention はサポートしているが、使用するように設定されていないワークロードは引き続き二重書き込みバッファを使用するため、パフォーマンス上のメリットはありません。

MySQL または MariaDB ソフトウェアスタックを設定して二重書き込みバッファを無効にし、Torn Write Prevention を使用するには次の手順を実行します。

1. BigAlloc オプションで ext4 ファイルシステムを使用するようにボリュームを設定し、クラスターサイズを 4 KiB、8 KiB、または 16 KiB に設定します。クラスターサイズが 4 KiB、8 KiB、または 16 KiB の BigAlloc を使用すると、ファイルシステムがそれぞれの境界に合わせてファイルを割り当てることができます。

   ```
   $  mkfs.ext4 -O bigalloc -C 4096|8192|16384 device_name
   ```

   > **Note**
   >
   > MySQL と MariaDB の場合はデータベースのページサイズに合わせて `-C 16384` を使用する必要があります。割り当ての粒度をページサイズの倍数以外の値に設定すると、ストレージデバイスの Torn Write Prevention の境界と割り当てが一致しなくなる可能性があります。

   例:

   ```
   $  mkfs.ext4 -O bigalloc -C 16384 /dev/nvme1n1
   ```

2. `O_DIRECT` フラッシュ方式を使用するように InnoDB を設定し、InnoDB の二重書き込みをオフにします。任意のテキストエディタを使用して `/etc/my.cnf` を開き、以下のように `innodb_flush_method` および `innodb_doublewrite` パラメータを更新します。

   ```
   innodb_flush_method=O_DIRECT
   innodb_doublewrite=0
   ```

> **Important**
>
> Logical Volume Manager (LVM) またはその他のストレージ仮想化レイヤーを使用している場合はボリュームの開始オフセットが 16 KiB の倍数になるように調整してください。これはストレージ仮想化レイヤーで使用されるメタデータヘッダーとスーパーブロックを考慮して、基盤となる NVMe ストレージを基準としています。LVM 物理ボリュームにオフセットを追加すると、ファイルシステムの割り当てと NVMe デバイスのオフセットがずれて、Torn Write Prevention が無効になる可能性があります。詳細については「Linux 手動ページ」の「`--dataalignmentoffset`」を参照してください。

アプリケーションと整合性のある、Windows VSS ベースの Amazon EBS スナップショット

Amazon EC2 インスタンスで Windows にアタッチされたすべての Amazon EBS ボリュームのアプリケーションコンシステントスナップショットを取得するには、AWS Systems Manager Run Command を使用します。スナップショットプロセスでは、Windows Volume Shadow Copy Service (VSS) を使用して、VSS 対応アプリケーションの EBS ボリュームレベルバックアップを取得します。スナップショットには、これらのアプリケーションとディスクとの間で保留されているトランザクションのデータが含まれます。すべてのアタッチされたボリュームをバックアップする際に、インスタンスをシャットダウンまたは切断する必要はありません。

VSS ベースの EBS スナップショットは追加コストなしで使用できます。バックアッププロセスにより作成される EBS スナップショットの料金のみです。詳細については、「Amazon EBS スナップショットの請求方法」を参照してください。

> **Note**
>
> アプリケーション整合性のある Windows VSS ベースのスナップショットは、Windows インスタンスでのみサポートされています。

内容

- VSS とは
- VSS ベースの Amazon EBS スナップショットソリューションの仕組み
- Windows VSS ベースの EBS スナップショットを作成するための前提条件

- EC2 Windows インスタンスの VSS ベースの EBS スナップショットを作成する
- Windows VSS ベースの EBS スナップショットのトラブルシューティング
- AWS VSS ソリューションを使用してインスタンスのデータを復元する
- AWS VSS ソリューションのバージョン履歴

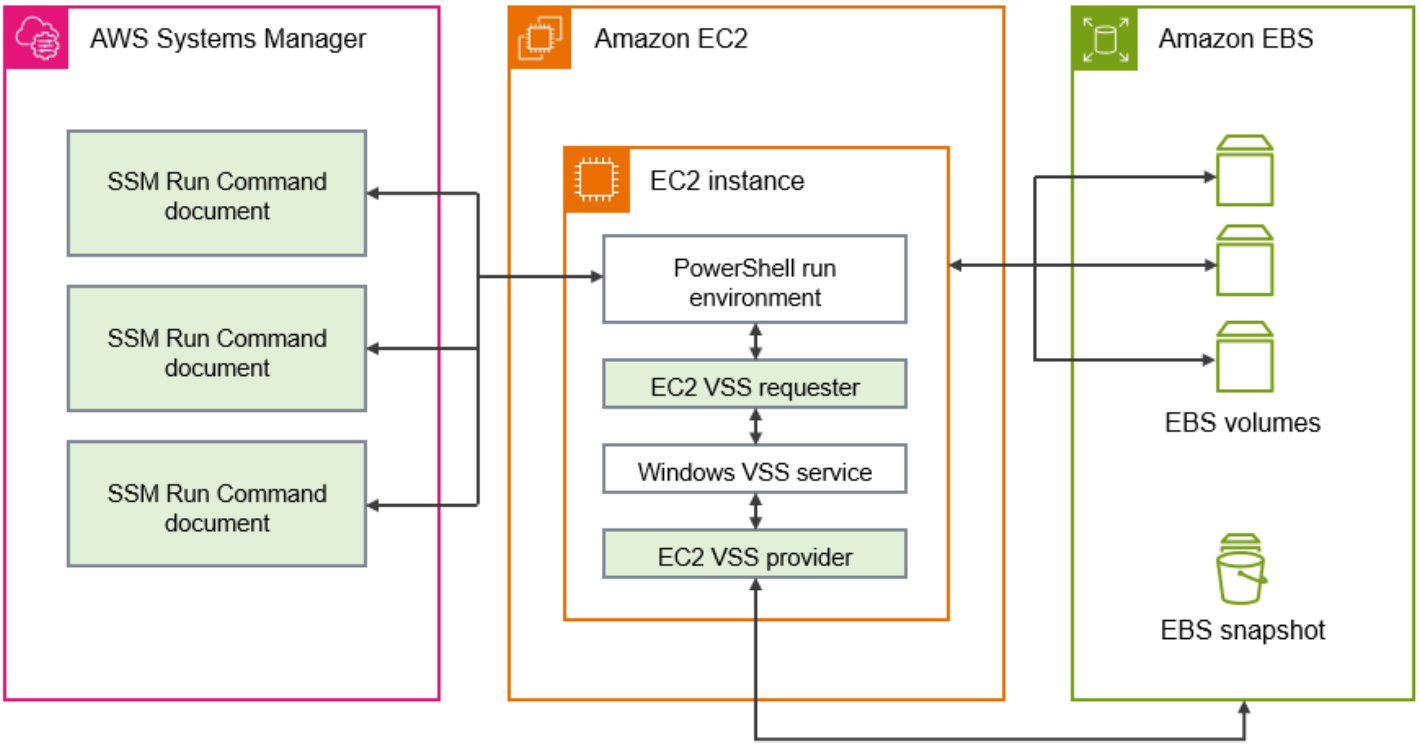
VSS とは

ボリュームシャドウコピーサービス (VSS) は、Microsoft Windows に含まれるバックアップおよび復元テクノロジーです。使用中のコンピューターファイルまたはボリュームのバックアップコピーまたはスナップショットを作成できます。詳細については、「ボリュームシャドウコピーサービス」を参照してください。

アプリケーションと整合性のあるスナップショットを作成するには、以下のソフトウェアコンポーネントが必要です。

- VSS サービス – Windows オペレーティングシステムの一部
- VSS リクエスタ – シャドウコピーの作成を要求するソフトウェア
- VSS ライター – 通常、SQL Server などのアプリケーションの一部として提供され、バックアップするデータセットの整合性を保ちます
- VSS プロバイダー – 基盤となるボリュームのシャドウコピーを作成するコンポーネント

Windows VSS ベースの Amazon EBS スナップショットソリューションは、バックアップ作成を促進する、複数の Systems Manager (SSM) Run Command ドキュメントおよび、`AwsVssComponents` と呼ばれる、EC2 VSS リクエスターとEC2 VSS プロバイダーを含む システムマネージャー ディストリビューター パッケージで構成されます。EBS ボリュームのアプリケーションと整合性のあるスナップショットを取得するには、`AwsVssComponents` パッケージを EC2 Windows インスタンスにインストールする必要があります。次の図は、これらのソフトウェアコンポーネント間の関係を示しています。



VSS ベースの Amazon EBS スナップショットソリューションの仕組み

アプリケーション整合性のある、VSS ベースの EBS スナップショットスクリプトを作成するプロセスは次の手順で構成されます。

1. Windows VSS ベースの EBS スナップショットを作成するための前提条件 を完了します。
2. AWSEC2-VssInstallAndSnapshot SSM ドキュメントのパラメータを入力し、Run Command を使用してこのドキュメントを実行します。詳細については「AWSEC2-VssInstallAndSnapshot コマンドドキュメント (推奨) を実行します。」を参照してください。
3. インスタンスの Windows VSS サービスが、実行中のアプリケーションで進行中のすべての I/O オペレーションを調整します。
4. システムがすべての I/O バッファをフラッシュし、一時的にすべての I/O オペレーションを一時停止します。一時停止の持続時間は最長でも 10 秒です。
5. 一時停止中に、インスタンスにアタッチされたすべてのボリュームのスナップショットがシステムによって作成されます。
6. 一時停止が解除され、I/O がオペレーションを再開します。
7. システムは、EBS スナップショットのリストに、新規作成されたすべてのスナップショットを追加します。システムは、このプロセスによって正常に作成されたすべての VSS ベースの EBS スナップショットに、AppConsistent:true とタグ付けします。

8. スナップショットから復元する必要がある場合は、スナップショットからボリュームを作成する EBS の標準プロセスを使用するか、AWS VSS ソリューションを使用してインスタンスのデータを復元する で説明したサンプルスクリプトを使用してすべてのボリュームをインスタンスに復元することができます。

Windows VSS ベースの EBS スナップショットを作成するための前提条件

VSS ベースの EBS スナップショットは、Systems Manager Run Command、AWS Backup または Amazon Data Lifecycle Manager を使用して作成できます。次の前提条件はすべてのソリューションに適用されます。

システム要件

EC2 Windows インスタンスが、Windows オペレーティングシステム、.NET Framework、PowerShell、AWS Tools for Windows PowerShell、および AWS Systems Manager エージェントのサポートされているバージョンなど、VSS ベースのスナップショットを作成するためのすべてのシステム要件を満たしていることを確認します。

IAM アクセス許可

Amazon EC2 Windows インスタンスにアタッチされた IAM ロールには、VSS を使用してアプリケーション整合性のあるスナップショットを作成するためのアクセス許可が必要です。必要なアクセス許可を付与するには、`AWSEC2VssSnapshotPolicy` 管理されたポリシーをインスタンスプロファイルにアタッチします。

VSS のコンポーネント

Windows オペレーティングシステムでアプリケーションと整合性のあるスナップショットを作成するには、`AwsVssComponents` パッケージをインスタンスにインストールする必要があります。このパッケージには、EBS ボリュームの VSS リクエスタおよび EC2 VSS プロバイダーとして機能するインスタンス上の EC2 VSS エージェントが含まれています。

システム要件

Systems Manager Agent のインストール

VSS は、PowerShell を使用する Systems Manager エージェントによってオーケストレーションされます。SSM Agent のバージョン `3.0.502.0` 以降が EC2 インスタンスにインストールされていることを確認します。SSM Agent の旧バージョンを使用している場合は、Run Command を使用してバージョンを更新します。詳細については、「AWS Systems Manager ユーザーガ

イド」の「Amazon EC2 インスタンス用 System Manager のセットアップ」および「Windows Server 用 EC2 インスタンスで SSM Agent を使用する」を参照してください。

Amazon EC2 Windows インスタンスの要件

VSS ベースの EBS スナップショットは、Windows Server 2016 以降を実行するインスタンスでサポートされています。

.NET Framework のバージョン

`AwsVssComponents` パッケージには、.NET Framework バージョン `4.6` 以降が必要です。Windows Server 2016 より前のバージョンの Windows オペレーティングシステムでは、デフォルトで以前のバージョンの .NET Framework が使用されます。インスタンスで以前のバージョンの .NET Framework を使用している場合は、Windows Update を使用してバージョン `4.6` 以降をインストールする必要があります。

AWS Tools for Windows PowerShell バージョン

インスタンスが AWS Tools for Windows PowerShell のバージョン `3.3.48.0` 以降を実行中であることを確認します。お使いのバージョンを確認するには、インスタンスの PowerShell ターミナルで次のコマンドを実行します。

```
C:\> Get-AWSPowerShellVersion
```

インスタンスの AWS Tools for Windows PowerShell を更新する必要がある場合は、「AWS Tools for PowerShell ユーザーガイド」の「AWS Tools for Windows PowerShell のインストール」を参照してください。

Windows PowerShell バージョン

インスタンスが Windows PowerShell のメジャーバージョン `3`、`4`、`5` のいずれかを実行中であることを確認します。お使いのバージョンを確認するには、インスタンスの PowerShell ターミナルで次のコマンドを実行します。

```
C:\> $PSVersionTable.PSVersion
```

PowerShell 言語モード

インスタンスの PowerShell 言語モードが `FullLanguage` に設定されていることを確認します。詳細については、「Microsoft ドキュメント」の「about_Language_Modes」を参照してください。

IAM 管理ポリシーを使用して VSS ベースのスナップショットに対するアクセス許可を付与する

AWSEC2VssSnapshotPolicy 管理ポリシーを使用すると、Systems Manager が Windows インスタンスに対して以下のアクションを実行できるようになります。

- EBS スナップショットを作成してタグ付けする
- Amazon マシンイメージ (AMI) を作成してタグ付けする
- VSS が作成するデフォルトのスナップショットタグに、デバイス ID などのメタデータをアタッチする

このトピックでは、VSS 管理ポリシーのアクセス許可の詳細と、それを EC2 インスタンスプロファイルの IAM ロールにアタッチする方法について説明します。

内容

- AWSEC2VssSnapshotPolicy 管理ポリシーの詳細
- VSS スナップショット管理ポリシーをインスタンスプロファイルロールにアタッチする

AWSEC2VssSnapshotPolicy 管理ポリシーの詳細

AWS マネージドポリシーは、Amazon が AWS に提供しているスタンドアロンのポリシーです。AWS マネージドポリシーは、一般的なユースケースでアクセス許可を付与できるように設計されています。AWS マネージドポリシーで定義したアクセス許可は変更できません。ただし、ポリシーをコピーして、ユースケースに固有のカスタマー管理ポリシーのベースラインとして使用することは可能です。

AWS マネージドポリシーの詳細については「IAM ユーザーガイド」の「AWS マネージドポリシー」を参照してください。

AWSEC2VssSnapshotPolicy 管理ポリシーを使用するには、そのポリシーを、EC2 Windows インスタンスにアタッチされている IAM ロールにアタッチします。このポリシーにより、EC2 VSS ソリューションは、タグを作成してこれを Amazon マシンイメージ (AMI) と EBS スナップショットに追加できるようになります。ポリシーのアタッチ方法については「VSS スナップショット管理ポリシーをインスタンスプロファイルロールにアタッチする」を参照してください。

AWSEC2VssSnapshotPolicy によって付与されるアクセス許可

AWSEC2VssSnapshotPolicy ポリシーには、Amazon EC2 がユーザーに代わって VSS スナップショットを作成および管理できるようにするために、以下の Amazon EC2 アクセス許可が含まれます。この管理ポリシーを、EC2 Windows インスタンスに使用する IAM インスタンスプロファイルロールにアタッチできます。

- ec2:CreateTags — リソースの識別と分類に役立つタグを EBS スナップショットと AMI に追加します。
- ec2:DescribeInstanceAttribute — ターゲットインスタンスにアタッチされている EBS ボリュームと対応するブロックデバイスマッピングを取得します。
- ec2:CreateSnapshots — EBS ボリュームのスナップショットを作成します。
- ec2:CreateImage — 実行中の EC2 インスタンスから AMI を作成します。
- ec2:DescribeImages — EC2 AMI とスナップショットの情報を取得します。
- ec2:DescribeSnapshots — スナップショットの作成時刻とステータスを確認し、アプリケーションの一貫性を検証します。

> **Note**
>
> このポリシーのアクセス許可の詳細を確認するには、「AWS 管理ポリシーリファレンス」の「AWSEC2VssSnapshotPolicy」を参照してください。

特定のユースケース用にアクセス許可を合理化する - 上級

`AWSEC2VssSnapshotPolicy` 管理ポリシーには、VSS ベースのスナップショットを作成できるすべての方法に対応したアクセス許可が含まれています。必要なアクセス許可のみを含むカスタムポリシーを作成できます。

ユースケース: AMI の作成、ユースケース: AWS Backup サービスの使用

`CreateAmi` オプションを排他的に使用する場合、または AWS Backup サービスを通じてのみ VSS ベースのスナップショットを作成する場合は、次のようにポリシーステートメントを合理化できます。

- 次のステートメント ID (SID) で識別されるポリシーステートメントは省略します。
  - `CreateSnapshotsWithTag`

  - CreateSnapshotsAccessInstance
  - CreateSnapshotsAccessVolume
- CreateTagsOnResourceCreation ステートメントを以下のように調整します。
  - リソースから arn:aws:ec2:*:*:snapshot/* を削除します。
  - CreateSnapshots を ec2:CreateAction 条件から削除します。
- CreateTagsAfterResourceCreation ステートメントを調整し、リソースから arn:aws:ec2:*:*:snapshot/* を削除します。
- DescribeImagesAndSnapshots ステートメントを調整し、ステートメントアクションから ec2:DescribeSnapshots を削除します。

ユースケース: スナップショットのみ

CreateAmi オプションを使用しない場合は、ポリシーステートメントを次のように合理化できます。

- 次のステートメント ID (SID) で識別されるポリシーステートメントは省略します。
  - CreateImageAccessInstance
  - CreateImageWithTag
- CreateTagsOnResourceCreation ステートメントを以下のように調整します。
  - リソースから arn:aws:ec2:*:*:image/* を削除します。
  - CreateImage を ec2:CreateAction 条件から削除します。
- CreateTagsAfterResourceCreation ステートメントを調整し、リソースから arn:aws:ec2:*:*:image/* を削除します。
- DescribeImagesAndSnapshots ステートメントを調整し、ステートメントアクションから ec2:DescribeImages を削除します。

> **Note**
>
> カスタマイズしたポリシーが想定どおりに機能するようにするため、定期的に管理ポリシーを検証して更新を組み込むことが推奨されます。

VSS スナップショット管理ポリシーをインスタンスプロファイルロールにアタッチする

EC2 Windows インスタンスの VSS ベースのスナップショットに対するアクセス許可を付与するには、次のように AWSEC2VssSnapshotPolicy 管理ポリシーをインスタンスプロファイルロールにアタッチします。インスタンスがすべての システム要件 を満たしていることを、確認することが重要です。

> **Note**
>
> 管理ポリシーを使用するには、インスタンスに `AwsVssComponents` パッケージバージョン `2.3.1` 以降がインストールされている必要があります。バージョン履歴については、「AwsVssComponents パッケージのバージョン」を参照してください。

1. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
2. ナビゲーションペインで [ロール] を選択すると、アクセスできる IAM ロールのリストが表示されます。
3. インスタンスにアタッチされているロールの [ロール名] リンクをクリックします。そのロールの詳細ページが開きます。
4. 管理ポリシーをアタッチするには、リストパネルの右上にある [許可を追加] を選択します。ドロップダウンリストから [ポリシーをアタッチ] を選択します。
5. 検索を効率化するには、検索バー (`AWSEC2VssSnapshotPolicy`) にポリシー名を入力します。
6. アタッチするポリシーの名前の横にあるチェックボックスをオンにし、[アクセス許可を追加] を選択します。

Windows VSS ベースの EBS スナップショット用の VSS コンポーネントパッケージを管理する

VSS ベースの EBS スナップショットを作成する前に、Windows インスタンスに最新バージョンの VSS コンポーネントパッケージがインストールされていることを確認してください。`AwsVssComponents` パッケージを既存のインスタンスにインストールするにはいくつかの方法があります。以下に例を示します。

- (推奨) AWSEC2-VssInstallAndSnapshot コマンドドキュメント (推奨) を実行します。。これにより、実行するたびに必要に応じて自動的にインストールまたはアップデートされます。

- VSS コンポーネントを EC2 Windows インスタンスに手動でインストールする.
- EC2 Windows インスタンスで VSS コンポーネントパッケージを更新する.

aws-vss-components-windows 管理されたコンポーネントを使用してイメージの AwsVssComponents パッケージをインストールする EC2 イメージビルダー で AMI を作成することもできます。管理されたコンポーネントは AWS Systems Manager Distributor を使用してパッケージをインストールします。イメージビルダー がイメージを作成すると、関連する AMI から起動するすべてのインスタンスに VSS パッケージがインストールされます。VSS パッケージがインストールされた AMI を作成する方法についてはEC2 イメージビルダー ユーザーガイド の「Windows 用ディストリビューターパッケージ管理コンポーネント」を参照してください。

内容

- VSS コンポーネントを EC2 Windows インスタンスに手動でインストールする
- EC2 Windows インスタンスで VSS コンポーネントパッケージを更新する

VSS コンポーネントを EC2 Windows インスタンスに手動でインストールする

システムマネージャー でアプリケーションと整合性のあるスナップショットを作成する前に、EC2 Windows インスタンスに VSS コンポーネントをインストールする必要があります。アプリケーションと整合性のあるスナップショットを作成するたびにパッケージを自動的にインストールまたは更新する AWSEC2-VssInstallAndSnapshot コマンドドキュメントを実行しない場合はパッケージを手動でインストールする必要があります。

また、次のいずれかの方法を使用して EC2 インスタンスからアプリケーションと整合性のあるスナップショットを作成する場合も、手動でインストールする必要があります。

- AWS Backup を使用して VSS スナップショットを作成する
- Amazon Data Lifecycle Manager を使用して VSS スナップショットを作成する

手動インストールが必要な場合はEC2 Windows インスタンス上でアプリケーションと整合性のあるスナップショットの信頼性とパフォーマンスを向上させるために、最新の AWS VSS コンポーネントパッケージを使用することが推奨されています。

> **Note**
>
> アプリケーションと整合性のあるスナップショットを作成するたびに AwsVssComponents パッケージを自動的にインストールまたは更新するにはシステムマネージャー を使用して

AWSEC2-VssInstallAndSnapshot ドキュメントを実行することをお勧めします。詳細についてはAWSEC2-VssInstallAndSnapshot コマンドドキュメント (推奨) を実行します。を参照してください。

Amazon EC2 Windows インスタンスに VSS コンポーネントをインストールするには希望する環境の手順に従います。

Console

SSM ディストリビューターを使用して VSS コンポーネントをインストールするには

1. AWS Systems Manager コンソールの https://console.aws.amazon.com/systems-manager/ を開いてください。
2. ナビゲーションペインで [Run Command] を選択してください。
3. [Run command] を選択してください。
4. [コマンドのドキュメント] で、[AWS-ConfigureAWSPackage] の横にあるボタンを選択してください。
5. [コマンドのパラメータ] で、以下の作業を行います。
   a. [アクション] が [インストール] に設定されていることを確認します。
   b. [名前] にAwsVssComponentsと入力してください。
   c. [バージョン] にバージョンを入力するか、フィールドを空のままにします。すると、システムマネージャー が最新バージョンをインストールします。
6. [Targets] (ターゲット) で、手動でインスタンスを指定または選択して、このオペレーションを実行するインスタンスを指定します。

   > **Note**
   > インスタンスを手動で選択することにしたが、そのインスタンスがリストに表示されない場合はAWS Systems Manager ユーザーガイドのインスタンスの場所でトラブルシューティングのヒントを参照してください。

7. [その他のパラメータ] で、以下の操作を行います。
   - (オプション) [コメント] に、このコマンドに関する情報を入力してください。

- [タイムアウト (秒)] に、コマンドの実行全体が失敗するまでにシステムが待機する秒数を指定します。

8. (オプション) [レートの制御] で、以下の操作を行います。

   - [同時実行] で、コマンドを同時に実行するインスタンスの数または割合 (%) を指定します。

     > **Note**
     >
     > Amazon EC2 タグを選択してターゲットを選択し、選択したタグを使用するインスタンスの数が不明な場合は同時に割合 (%) を指定してドキュメントを実行できるインスタンスの数を制限します。

   - [エラーのしきい値] で、インスタンスの数または割合 (%) で失敗した後で他のインスタンスでのコマンドの実行をいつ停止するか指定します。例えば、3 つのエラーを指定した場合、4 番目のエラーが受信されると、システムマネージャー はコマンドの送信を停止します。コマンドを処理しているインスタンスもエラーを送信する可能性があります。

9. (オプション) [出力オプション] セクションで、コマンド出力をファイルに保存する場合は [S3 バケットへの書き込みの有効化] の横にあるチェックボックスをオンにします。バケットと、(オプションで) プリフィックス (フォルダ) 名を指定します。

   > **Note**
   >
   > S3 バケットにデータを書き込む機能を許可する S3 アクセス許可はこのタスクを実行する ユーザーのものではなく、インスタンスに割り当てられたインスタンスプロファイルのものです。詳細については、AWS Systems Managerユーザーガイドの「EC2 インスタンスの権限の設定」を参照してください。

10. (オプション) [SNS 通知] のオプションを指定します。

    Run Command の Amazon SNS 通知の設定については[AWS Systems Manager に Amazon SNS 通知を設定する]を参照してください。

11. [実行] を選択してください。

AWS CLI

次の手順に従い、AWS CLI の Run Command を使用して、AwsVssComponents パッケージをダウンロードしてインスタンスにインストールします。パッケージによって、VSS リクエスタおよび VSS プロバイダという 2 つのコンポーネントがインストールされます。システムはインスタンス上のディレクトリにこれらのコンポーネントをコピーしてから、プロバイダ DLL を VSS プロバイダとして登録します。

VSS パッケージをインストールするには

次のコマンドを実行して、システムマネージャー に必要な VSS コンポーネントをダウンロードしてインストールします。

```
aws ssm send-command \
    --document-name "AWS-ConfigureAWSPackage" \
    --instance-ids "i-1234567890abcdef0" \
    --parameters '{"action":["Install"],"name":["AwsVssComponents"]}'
```

PowerShell

次の手順を使用して、Tools for Windows PowerShell から Run Command を使用して、インスタンスで AwsVssComponents パッケージをダウンロードし、インストールします。パッケージによって、VSS リクエスタおよび VSS プロバイダという 2 つのコンポーネントがインストールされます。システムはインスタンス上のディレクトリにこれらのコンポーネントをコピーしてから、プロバイダ DLL を VSS プロバイダとして登録します。

VSS パッケージをインストールするには

次のコマンドを実行して、システムマネージャー に必要な VSS コンポーネントをダウンロードしてインストールします。

```
Send-SSMCommand `
    -DocumentName "AWS-ConfigureAWSPackage" `
    -InstanceId "i-1234567890abcdef0" `
    -Parameter @{'action'='Install';'name'='AwsVssComponents'}
```

AWS VSS コンポーネントの署名を確認

次の手順に従い、AwsVssComponents パッケージの署名をパッケージの署名を確認します。

1. Windows インスタンスに接続します。詳細については[RDP を使用した Windows インスタンスへの接続]を参照してください。
2. C:\Program ファイルs\Amazon\AwsVssComponents に移動します。
3. `ec2-vss-agent.exe` のコンテキスト メニュー (右クリック) を開き、[Properties] を選択してください。
4. 「デジタル署名」タブに移動し、署名者の名前が Amazon Web Services 株式会社であることを確認します。
5. 前述の手順を使用して、`Ec2VssInstaller` およびの署名を確認します `Ec2VssProvider.dll`。

EC2 Windows インスタンスで VSS コンポーネントパッケージを更新する

VSS コンポーネントは最新の推奨バージョンに更新しておくことをお勧めします。`AwsVssComponents` パッケージの新しいバージョンがリリースされたときに、コンポーネントを更新するにはいくつかの方法があります。

更新方式

- AWS VSS コンポーネントの新しいバージョンがリリースされたときに、[VSS コンポーネントを EC2 Windows インスタンスに手動でインストールする] で説明する手順を繰り返すことができます。
- パッケージが使用可能になったときに新しい `AwsVssComponents` または更新済みの VSS コンポーネントを自動的にダウンロードしインストールするように、System Manager ステートマネージャーの関連付けを設定します。
- アプリケーションと整合性のあるスナップショットを作成したり、システムマネージャーを使用して `AWSEC2-VssInstallAndSnapshot` ドキュメントを実行したりするたびに、`AwsVssComponents` パッケージを自動的にインストールまたは更新できます。

> **Note**
>
> システムマネージャー を使用して `AWSEC2-VssInstallAndSnapshot` コマンドドキュメントを実行することをお勧めします。こうすると、アプリケーションと整合性のあるスナップショットが作成される前に、`AwsVssComponents` パッケージが自動的にインストールまたは更新されます。詳細については[AWSEC2-VssInstallAndSnapshot コマンドドキュメント (推奨) を実行します。]を参照してください。

システムマネージャーのステートマネージャー の関連付けを作成するには希望する環境に応じた手順を実行します。

Console

Systems Manager State Manager の関連付けを作成する場合、AwsVssComponents パッケージを更新するには、次のように 2 つのオプションがあります。

アンインストールと再インストール

この方法では、追加の前提条件なしでパッケージをダウンロードし、インストールします。

インプレース更新

これにより、パッケージのインプレース更新を実行するもので、次の前提条件があります。

- インスタンスにインストールされている SSM エージェントのバージョンは、 3.3.808.0 以降である必要があります。詳細については、「AWS Systems Manager ユーザーガイド」の「Windows Server の EC2 インスタンスで SSM Agent を使用する」を参照してください。
- 指定する場合、AwsVssComponents パッケージのバージョンはバージョン 2.5.0 以降である必要があります。以前のバージョンでは、インプレース更新はサポートされていません。

> **Note**
>
> インスタンスがこれらの前提条件を満たさないと、インプレース更新は失敗します。代わりに Uninstall and reinstall オプションを使用します。

ステートマネージャーの関連付けを作成するには

1. AWS Systems Manager コンソール (https://console.aws.amazon.com/systems-manager/) を開きます。
2. ナビゲーションペインで、[ステートマネージャー] を選択してください。

   システムマネージャー のホームページが最初に開く場合はナビゲーションペインを開き、[ステートマネージャー] を選択してください。
3. [アソシエーションを作成する] を選択してください。
4. [名前] フィールドに、わかりやすい名前を入力してください。

5. [ドキュメント] リストで、[AWS-ConfigureAWSPackage] を選択してください。
6. [パラメータ] セクションで、[アクション] リストから [インストール] を選択してください。
7. [インストールタイプ] で、[アンインストールと再インストール]、または [インプレース更新] を選択します。
8. [名前] フィールドに `AwsVssComponents` を入力してください。[バージョン] () フィールドと [追加の引数] フィールドは空のままにしておくことができます。
9. [ターゲット] セクションで、オプションを選択してください。

> **Note**
>
> タグを使用してインスタンスを対象にし、Linux インスタンスにマッピングされるタグを指定する場合、関連付けは Windows インスタンスでは成功しますが、Linux インスタンスでは失敗します。関連付けの全体的なステータスは Failed と表示されます。

10. [スケジュール指定] セクションで、オプションを選択してください。
11. [詳細オプション] セクションの [コンプライアンスの重要度] で、関連付けの重要度レベルを選択してください。詳細については「ステージ管理関連付けのコンプライアンスについて」を参照してください。[カレンダー変更] ではあらかじめ設定されている変更カレンダーを選択してください。詳細については AWS Systems Manager「カレンダー変更」を参照してください。
12. [レートコントロール] では次の操作を行います。
    - [同時実行数] の場合、コマンドを同時に実行する管理されたノードの数または割合を指定します。
    - [エラーのしきい値] で、ノードの数または割合のいずれかで失敗した後、他の管理されたノードでのコマンドの実行をいつ停止するか指定します。
13. (オプション) [出力オプション] でコマンド出力をファイルに保存するには[S3 への出力の書き込みを有効にする] を選択してください。ボックスにバケット名とプレフィックス (フォルダ) 名を入力してください。
14. [関連付けを作成する] を選択してから、[閉じる] を選択してください。システムはインスタンスで関連付けを作成し、状態を即時に適用します。

> **Note**
>
> Windows サーバー の EC2 インスタンスのステータスに [失敗] と表示されている場合はインスタンスで SSM エージェントが実行されていることと、そのインスタンスが システムマネージャー の AWS Identity and Access Management (IAM) ロールで設定されていることを確認します。詳細についてはAWS Systems Manager「のセットアップ」を参照してください。

AWS CLI

create-association コマンドを使用して、関連付けられたアプリケーションをオフラインにすることなく、スケジュールに従って Distributor パッケージを更新します。パッケージ内の新規または更新されたファイルのみが置き換えられます。

ステートマネージャーの関連付けを作成するには

次のコマンドを実行して、関連付けを作成します。--name の値、ドキュメント名は常に AWS-ConfigureAWSPackage です。次のコマンドではキー InstanceIds を使用してターゲットインスタンスを指定します。

```
aws ssm create-association \
    --name "AWS-ConfigureAWSPackage" \
    --parameters action=Install,installationType="Uninstall and
 reinstall",name=AwsVssComponents \
    --targets Key=InstanceIds,Values=i-1234567890abcdef0,i-000011112222abcde
```

PowerShell

ステートマネージャーの関連付けを作成するには

New-SSMAssociation コマンドレットを使用します。

```
New-SSMAssociation `
    -Name "AWS-ConfigureAWSPackage" `
    -Parameter  @{
        "action" = "Install"
        "installationType" = "Uninstall and reinstall"
        "name" = "AwsVssComponents"
    } `
```

```
    -Target @{
        "Key" = "InstanceIds"
        "Values" = @("i-1234567890abcdef0", "i-000011112222abcde")
    }
```

EC2 Windows インスタンスの VSS ベースの EBS スナップショットを作成する

「Windows VSS ベースの EBS スナップショットを作成するための前提条件」のすべての前提条件を満たしたら、次のいずれかの方法を使用して、EC2 インスタンスから VSS ベースのスナップショットを作成できます。

AWS Systems Manager コマンドドキュメント

「Systems Manager コマンドドキュメントを使用する」を実行して、VSS ベースのスナップショットを作成します。

バックアップを自動化するには、AWSEC2-VssInstallAndSnapshot コマンドドキュメントを使用する AWS Systems Manager メンテナンスウィンドウタスクを作成します。詳細については、AWS Systems Manager ユーザーガイドのメンテナンスウィンドウの使用 (コンソール)を参照してください。

AWS Backup

コンソールまたは CLI で VSS を有効にすると、AWS Backup 使用時に VSS バックアップを作成できます。詳しくは、「AWS Backup デベロッパーガイド」の「Creating Windows VSS backups」を参照してください。

> **Note**
>
> AWS Backup は AwsVssComponents パッケージをインスタンスに自動的にインストールしません。インスタンスには手動でインストールする必要があります。詳細については、「VSS コンポーネントを EC2 Windows インスタンスに手動でインストールする」を参照してください。

Amazon Data Lifecycle Manager

Amazon Data Lifecycle Manager を使用して VSS スナップショットを作成するには、スナップショットライフサイクルポリシーで事前スクリプトと事後スクリプトを有効にします。詳細につ

いては、「Amazon EBS ユーザーガイド」の「アプリケーション整合性のあるスナップショットの自動化」を参照してください。

> **Note**
>
> Amazon Data Lifecycle Manager は、`AwsVssComponents` パッケージをインスタンスに自動的にインストールしません。インスタンスには手動でインストールする必要があります。詳細については、「VSS コンポーネントを EC2 Windows インスタンスに手動でインストールする」を参照してください。

Systems Manager コマンドドキュメントを使用して VSS ベースのスナップショットを作成する

AWS Systems Manager コマンドドキュメントを使用して、VSS ベースのスナップショットを作成できます。以下のコンテンツでは、使用可能なコマンドドキュメントと、そのドキュメントがスナップショットの作成に使用するランタイムパラメータを紹介します。

Systems Manager のコマンドドキュメントを使用する前に、すべてのWindows VSS ベースの EBS スナップショットを作成するための前提条件を満たしていることを確認してください。

トピック

- Systems Manager VSS スナップショットドキュメントのパラメータ
- Systems Manager VSS スナップショットコマンドドキュメントを実行する

Systems Manager VSS スナップショットドキュメントのパラメータ

VSS スナップショットを作成する Systems Manager ドキュメントはすべて、特に明記されていない限り、以下のパラメータを使用します。

AmiName (文字列、オプション)

CreateAmi オプションが `True` に設定されている場合は、バックアップが作成する AMI の名前を指定します。

description (文字列、オプション)

このプロセスで作成されるスナップショットまたはイメージの説明を指定します。

CollectDiagnosticLogs (文字列、オプション)

スナップショットと AMI の作成ステップでより多くの情報を収集するには、このパラメータを `True` に設定します。このパラメータのデフォルト値は "`False`" です。統合診断ログは、インスタンス上の次の場所に `.zip` 形式のアーカイブとして保存されます。

`C:\ProgramData\Amazon\AwsVss\Logs\`*`timestamp`*`.zip`

CopyOnly (文字列、オプション)

AWS VSS に加えてネイティブ SQL Server バックアップを使用する場合は、コピーのみのバックアップを実行すると、AWS VSS によるネイティブ差分バックアップチェーンの中断を防ぐことができます。コピーのみのバックアップ操作を実行するには、このパラメータを `True` に設定します。

このパラメータのデフォルト値は `False` であり、AWS VSS のフルバックアップ操作実行の原因となります。

CreateAmi (文字列、オプション)

VSS ベースの Amazon マシンイメージ (AMI) を作成してインスタンスをバックアップするには、このパラメータを `True` に設定します。このパラメータのデフォルト値は `False` であり、代わりに EBS スナップショットを使用してインスタンスをバックアップします。

インスタンスから AMI を作成する方法の詳細については、「Amazon EBS-backed AMI を作成する」と「」を参照してください。

executionTimeout (文字列、オプション)

インスタンスでスナップショット作成プロセスを実行する最大時間、またはインスタンスから AMI を作成する最大時間を秒単位で指定します。このタイムアウトを長くすることで、VSS がフリーズを開始し、作成するリソースのタグ付けが完了するまで、コマンドがより長く待機できるようになります。このタイムアウトは、スナップショットまたは AMI の作成ステップにのみ適用されます。`AwsVssComponents` パッケージをインストールまたは更新する最初のステップは、タイムアウトには含まれません。

ExcludeBootVolume (文字列、オプション)

この設定では、スナップショットを作成する場合に、バックアッププロセスからブートボリュームが除外されます。スナップショットからブートボリュームを除外するには、ExcludeBootVolume を `True` に、CreateAmi を `False` に設定します。

バックアップ用に AMI を作成する場合、このパラメータを `False` に設定する必要があります。このパラメータのデフォルト値は `False` です。

NoWriters (文字列、オプション)

アプリケーション VSS ライターをスナップショットプロセスから除外するには、このパラメータを `True` に設定します。アプリケーション VSS ライターを除外すると、サードパーティの VSS バックアップコンポーネントとの競合を解決しやすくなります。このパラメータのデフォルト値は `False` です。

`SaveVssMetadata` が `True` の場合、このパラメータを `False` に設定する必要があります。

SaveVssMetadata (文字列、オプション)

スナップショットを作成するたびに VSS メタデータファイルを保存するには、このパラメータを `True` に設定します。デフォルト値は `False` です。これらのファイルにより、バックアップ操作に含まれていたコンポーネントやライター、および各スナップショットの関連ファイルとボリューム関するインサイトが得られます。メタデータファイルは、VSS 復元ソリューションを使用して SQL データベースを復元するときに使用されます。VSS スナップショットから SQL データベースを復元する方法の詳細については、「Use an automation runbook to restore your database from AWS VSS solution snapshots」を参照してください。

メタデータファイルの名前にスナップショット設定 ID が関連付けられています。メタデータファイルはインスタンスの次の場所にあります。

```
C:\ProgramData\Amazon\AwsVss\VssMetadata\
```

> **Warning**
>
> - VSS メタデータファイルを保存するには、バージョン 2.4.0 以降の `AwsVssComponents` パッケージが必要です。インスタンスの旧バージョンがインストールされている場合、`SaveVssMetadata` を `True` に設定するとスナップショットの作成が失敗します。
> - `NoWriters` パラメータと `SaveVssMetadata` パラメータは相互に排他的です。両方とも `True` に設定すると、スナップショットの作成が失敗します。

tags (文字列、オプション)

スナップショットとイメージにタグを付けて、リソースを見つけて管理できるようにすること (スナップショットのリストからボリュームを復元するなど) をお勧めします。システムは、空白の値で `Name` キーを追加するので、出力スナップショットまたはイメージに適用する名前を指定できます。

追加のタグを指定する場合は、セミコロンを使用してタグを区切ります。例えば、`Key=Environment,Value=Test;Key=User,Value=TestUser1`。

> **Note**
>
> タグのキーと値には、英数字と次の特殊文字のみを含める必要があります: `( ) . / \ - " ' @ _ + : = { }`。

デフォルトでは、システムは VSS ベースのスナップショットおよびイメージ用に、次の予約済みタグを追加します。

- Device – VSS ベースのスナップショットの場合、これはスナップショットがキャプチャする EBS ボリュームのデバイス名です。
- AppConsistent – このタグは、VSS 対応スナップショットまたは AMI が正常に作成されたことを示します。
- AwsVssConfig – VSS を有効にして作成されたスナップショットと AMI を識別します。タグには、`AwsVssComponents` のバージョンとスナップショット設定 ID などのメタ情報が含まれています。

> **Warning**
>
> パラメータリストでこれらの予約済みタグのいずれかを指定すると、エラーが発生します。

VssVersion (文字列、オプション)

`AWSEC2-VssInstallAndSnapshot` ドキュメントに限り、`VssVersion` パラメータを指定して、`AwsVssComponents` パッケージの特定のバージョンをインスタンスにインストールすることができます。推奨されているデフォルトバージョンをインストールするには、このパラメータを空白のままにします。

指定したバージョンの `AwsVssComponents` パッケージがすでにインストールされている場合、スクリプトはインストール手順をスキップしてバックアップ手順に進みます。`AwsVssComponents` パッケージバージョンと運用サポートの一覧については、「AWS VSS ソリューションのバージョン履歴」を参照してください。

Systems Manager VSS スナップショットコマンドドキュメントを実行する

次のように AWS Systems Manager コマンドドキュメントを使用して、VSS ベースの EBS スナップショットを作成できます。

AWSEC2-VssInstallAndSnapshot コマンドドキュメント (推奨) を実行します。

AWS Systems Manager を使用して `AWSEC2-VssInstallAndSnapshot` ドキュメントを実行すると、スクリプトは次のステップを実行します。

1. このスクリプトは、インスタンスが既にインストールされているかどうかに応じて、最初にインスタンスに `AwsVssComponents` パッケージをインストールまたは更新します。
2. このスクリプトは、最初のステップが完了した後に、アプリケーションと整合性のあるスナップショットを作成します。

`AWSEC2-VssInstallAndSnapshot` ドキュメントを実行するには、ご希望の環境に応じた手順に従ってください。

Console

コンソールから VSS ベースの EBS スナップショットを作成する

1. AWS Systems Manager コンソール (https://console.aws.amazon.com/systems-manager/) を開きます。
2. ナビゲーションペインから [コマンドの実行] をクリックします。該当する場合、アカウントで現在実行されているコマンドのリストが表示されます。
3. [Run command] を選択してください。これにより、アクセスできるコマンドドキュメントのリストが開きます。
4. コマンドドキュメントのリストから `AWSEC2-VssInstallAndSnapshot` を選択します。結果を効率化するために、ドキュメント名の全体または一部を入力できます。所有者、プラットフォームタイプ、またはタグでフィルタリングすることもできます。

   コマンドドキュメントを選択すると、詳細がリストの下に表示されます。
5. [ドキュメントバージョン] リストから `Default version at runtime` を選択します。
6. [コマンドパラメータ] を設定して、`AWSEC2-VssInstallAndSnapshot` による `AwsVssComponents` パッケージのインストール方法と VSS スナップショットまたは AMI によるバックアップ方法を定義します。パラメータの詳細については、「Systems Manager VSS スナップショットドキュメントのパラメータ」を参照してください。

7. [ターゲットの選択] で、タグを指定するか、手動でインスタンスを選択して、この操作を実行するインスタンスを特定します。

   > **Note**
   >
   > インスタンスを手動で選択することにしたが、そのインスタンスがリストに表示されない場合は、「インスタンスの場所」でトラブルシューティングのヒントを参照してください。

8. [レート制御] などの Systems Manager Run Command の動作を定義する追加パラメータについては、「コンソールからのコマンドの実行」の説明に従って値を入力します。
9. [実行] を選択します。

   成功すると、コマンドによって EBS スナップショットのリストに新しいスナップショットが入力されます。指定したタグまたは `AppConsistent` を検索することで、EBS スナップショットのリスト内でこれらのスナップショットを見つけることができます。コマンドの実行が失敗した場合、Systems Manager コマンド出力で、実行が失敗した理由の詳細を確認してください。コマンドは正常に完了したが特定のボリュームのバックアップが失敗した場合、EBS ボリュームのリストで失敗をトラブルシューティングできます。

AWS CLI

AWS CLI で次のコマンドを実行して、VSS ベースの EBS スナップショットを作成し、スナップショット作成のステータスを取得できます。

VSS ベースの EBS スナップショットを作成する

次のコマンドを実行して、VSS ベースの EBS スナップショットを作成します。スナップショットを作成するには、`--instance-ids` パラメータを使用してインスタンスを識別する必要があります。使用可能なその他のパラメータについては、「Systems Manager VSS スナップショットドキュメントのパラメータ」を参照してください。

```
aws ssm send-command \
 --document-name "AWSEC2-VssInstallAndSnapshot" \
 --instance-ids "i-01234567890abcdef" \
 --parameters '{"ExcludeBootVolume":["False"],"description":["Description"],"tags":
["Key=key_name,Value=tag_value"],"VssVersion":[""]}'
```

成功すると、コマンドドキュメントによって EBS スナップショットのリストに新しいスナップショットが入力されます。指定したタグまたは AppConsistent を検索することで、EBS スナップショットのリスト内でこれらのスナップショットを見つけることができます。コマンドの実行が失敗した場合、 コマンド出力で、実行が失敗した理由の詳細を確認してください。

コマンドステータスの取得

スナップショットの現在のステータスを取得するには、send-command から返されたコマンド ID を使用して次のコマンドを実行します。

```
aws ssm get-command-invocation
 --instance-ids "i-01234567890abcdef" \
 --command-id "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111" \
 --plugin-name "CreateVssSnapshot"
```

PowerShell

AWS Tools for Windows PowerShell を使用して次のコマンドを実行して、VSS ベースの EBS スナップショットを作成し、出力作成の現在のランタイムステータスを取得します。前のリストで説明されたパラメータを指定して、スナップショットプロセスの動作を変更します。

Windows 用ツール PowerShell を使用して VSS ベースの EBS スナップショットを作成する

次のコマンドを実行して、VSS ベースの EBS スナップショットまたは AMI を作成します。

```
Send-SSMCommand -DocumentName "AWSEC2-VssInstallAndSnapshot" -InstanceId
 "i-01234567890abcdef" -Parameter
 @{'ExcludeBootVolume'='False';'description'='a_description'
 ;'tags'='Key=key_name,Value=tag_value';'VssVersion'=''}
```

コマンドステータスの取得

スナップショットの現在のステータスを取得するには、Send-SSMCommand から返されたコマンド ID を使用して次のコマンドを実行します。

```
Get-SSMCommandInvocationDetail -InstanceId "i-01234567890abcdef" -CommandId
 "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111" -PluginName "CreateVssSnapshot"
```

成功すると、コマンドによって EBS スナップショットのリストに新しいスナップショットが入力されます。指定したタグまたは AppConsistent を検索することで、EBS スナップショット

のリスト内でこれらのスナップショットを見つけることができます。コマンドの実行が失敗した場合、 コマンド出力で、実行が失敗した理由の詳細を確認してください。

AWSEC2-CreateVssSnapshot コマンドドキュメントを実行する

AWSEC2-CreateVssSnapshot ドキュメントを実行するには、ご希望の環境に応じた手順に従ってください。

Console

コンソールから VSS ベースの EBS スナップショットを作成する

1. AWS Systems Manager コンソール (https://console.aws.amazon.com/systems-manager/) を開きます。
2. ナビゲーションペインから [コマンドの実行] をクリックします。該当する場合、アカウントで現在実行されているコマンドのリストが表示されます。
3. [Run command] を選択してください。これにより、アクセスできるコマンドドキュメントのリストが開きます。
4. コマンドドキュメントのリストから AWSEC2-CreateVssSnapshot を選択します。結果を効率化するために、ドキュメント名の全体または一部を入力できます。所有者、プラットフォームタイプ、またはタグでフィルタリングすることもできます。

   コマンドドキュメントを選択すると、詳細がリストの下に表示されます。
5. [ドキュメントバージョン] リストから Default version at runtime を選択します。
6. [コマンドのパラメータ] を設定して、AWSEC2-CreateVssSnapshot が VSS スナップショットまたは AMI を使用してバックアップする方法を定義します。パラメータの詳細については、「Systems Manager VSS スナップショットドキュメントのパラメータ」を参照してください。
7. [ターゲットの選択] で、タグを指定するか、手動でインスタンスを選択して、この操作を実行するインスタンスを特定します。

> **Note**
>
> インスタンスを手動で選択することにしたが、そのインスタンスがリストに表示されない場合は、「インスタンスの場所」でトラブルシューティングのヒントを参照してください。

8. [レート制御] などの Systems Manager Run Command の動作を定義する追加パラメータについては、「コンソールからのコマンドの実行」の説明に従って値を入力します。
9. [実行] を選択します。

   成功すると、コマンドによって EBS スナップショットのリストに新しいスナップショットが入力されます。指定したタグまたは AppConsistent を検索することで、EBS スナップショットのリスト内でこれらのスナップショットを見つけることができます。コマンドの実行が失敗した場合、Systems Manager コマンド出力で、実行が失敗した理由の詳細を確認してください。コマンドは正常に完了したが特定のボリュームのバックアップが失敗した場合、EBS ボリュームのリストで失敗をトラブルシューティングできます。

AWS CLI

AWS CLI で次のコマンドを実行して、VSS ベースの EBS スナップショットを作成できます。

VSS ベースの EBS スナップショットを作成する

次のコマンドを実行して、VSS ベースの EBS スナップショットを作成します。スナップショットを作成するには、--instance-ids パラメータを使用してインスタンスを識別する必要があります。使用可能なその他のパラメータについては、「Systems Manager VSS スナップショットドキュメントのパラメータ」を参照してください。

```
aws ssm send-command \
 --document-name "AWSEC2-CreateVssSnapshot" \
 --instance-ids "i-01234567890abcdef" \
 --parameters '{"ExcludeBootVolume":["False"],"description":["Description"],"tags":
["Key=key_name,Value=tag_value"]}'
```

成功すると、コマンドドキュメントによって EBS スナップショットのリストに新しいスナップショットが入力されます。指定したタグまたは AppConsistent を検索することで、EBS スナップショットのリスト内でこれらのスナップショットを見つけることができます。コマンドの実行が失敗した場合、 コマンド出力で、実行が失敗した理由の詳細を確認してください。

PowerShell

AWS Tools for Windows PowerShell で次のコマンドを実行して、VSS ベースの EBS スナップショットを作成します。

Windows 用ツール PowerShell を使用して VSS ベースの EBS スナップショットを作成する

次のコマンドを実行して、VSS ベースの EBS スナップショットを作成します。スナップショットを作成するには、InstanceId パラメータを使用してインスタンスを識別する必要があります。複数のインスタンスを指定してスナップショットを作成できます。使用可能なその他のパラメータについては、「Systems Manager VSS スナップショットドキュメントのパラメータ」を参照してください。

```
Send-SSMCommand -DocumentName AWSEC2-CreateVssSnapshot -InstanceId
 "i-01234567890abcdef" -Parameter
 @{'ExcludeBootVolume'='False';'description'='a_description'
 ;'tags'='Key=key_name,Value=tag_value'}
```

成功すると、コマンドによって EBS スナップショットのリストに新しいスナップショットが入力されます。指定したタグまたは AppConsistent を検索することで、EBS スナップショットのリスト内でこれらのスナップショットを見つけることができます。コマンドの実行が失敗した場合、 コマンド出力で、実行が失敗した理由の詳細を確認してください。コマンドは正常に完了したが特定のボリュームのバックアップが失敗した場合、EBS スナップショットのリストで失敗をトラブルシューティングできます。

共有 EBS ストレージを持つ Windows フェイルオーバークラスターのコマンドドキュメントを実行する

前のセクションで説明したいずれかのコマンドライン手順を使用して、VSS ベースのスナップショットを作成できます。コマンドドキュメント (AWSEC2-VssInstallAndSnapshot または AWSEC2-CreateVssSnapshot) は、クラスターのプライマリノードで実行する必要があります。セカンダリノードは共有ディスクにアクセスできないため、ドキュメントの実行は失敗します。プライマリノードとセカンダリノードが動的に変化する場合は、AWS Systems Manager Run Command ドキュメントを複数のノードで同時に実行できますが、プライマリノードではコマンドが成功し、セカンダリノードでは失敗します。

> **Note**
>
> バックアップを自動化するには、AWSEC2-VssInstallAndSnapshot ドキュメントを使用する AWS Systems Manager メンテナンスウィンドウタスクを作成できます。詳細については、AWS Systems Manager ユーザーガイドのメンテナンスウィンドウの使用 (コンソール)を参照してください。

Windows VSS ベースの EBS スナップショットのトラブルシューティング

他のトラブルシューティングのステップを試す前に、次の詳細を確認することをお勧めします。

- すべてのWindows VSS ベースの EBS スナップショットを作成するための前提条件を満たしていることを確認してください。
- 使用しているオペレーティングシステムに対応した、AwsVssComponents パッケージの最新の Windows OS バージョンのサポートを使用していることを確認してください。見つかった問題は、新しいバージョンで対処されている可能性があります。

トピック

- ログファイルのチェック
- 追加の診断ログを収集する
- プロキシが設定されたインスタンスでの VSS の使用
- エラー: パイプ接続の解凍がタイムアウトしました、解凍時のエラー、VSS フリーズ待機のタイムアウト、またはその他のタイムアウトエラー
- エラー: メソッドを呼び出せません。メソッド呼び出しは、この言語モードのコアタイプでのみサポートされます。

ログファイルのチェック

VSS ベースの EBS スナップショットの作成時に問題が発生したり、エラーメッセージが表示されたりした場合は、Systems Manager コンソールでコマンドの出力を表示できます。

VSS スナップショットを作成する Systems Manager ドキュメントの場合、実行時に CollectDiagnosticLogs パラメータを「True」に設定します。CollectDiagnosticLogs パラメータが「True」に設定されると、VSS はデバッグに役立つ追加のログを収集します。詳細については「追加の診断ログを収集する」を参照してください。

診断ログを収集する場合、Systems Manager ドキュメントはそれらをインスタンスの次の場所に保存します: C:\ProgramData\Amazon\AwsVss\Logs*timestamp*.zip。CollectDiagnosticLogs パラメータのデフォルト値は "False" です。

> **Note**
>
> デバッグに関する追加のヘルプが必要な場合は、.zip ファイルを サポート に送付してください。

診断ログを収集するかどうかにかかわらず、次の追加ログを使用できます。

- `%ProgramData%\Amazon\SSM\InstanceData\`*`InstanceID`*`\document\orchestration\`*`SSMCommandID`*`\awsrunPowerShellScript\runPowerShellScript\stdout`
- `%ProgramData%\Amazon\SSM\InstanceData\`*`InstanceID`*`\document\orchestration\`*`SSMCommandID`*`\awsrunPowerShellScript\runPowerShellScript\stderr`

イベントビューアー Windows アプリケーションを開き、[Windows Logs]、[Application] の順に選択して、追加のログを表示することもできます。EC2 Windows VSS プロバイダーおよび Volume Shadow Copy Service からのイベントを具体的に表示するには、**`Ec2VssSoftwareProvider`** および **`VSS`** の条件で [Source] (送信元) でフィルタリングします。

VPC エンドポイントで Systems Manager を使用していて、Systems Manager の SendCommand API アクション (コンソールの Run Command) が失敗した場合は、以下のエンドポイントが正しく設定されていることを確認してください: com.amazonaws.***`region`***.ec2

Amazon EC2 エンドポイントが定義されていない場合、アタッチした EBS ボリュームを列挙する呼び出しは失敗し、Systems Manager コマンドが失敗します。Systems Manager による VPC エンドポイントの設定の詳細については、AWS Systems Manager ユーザーガイドのVirtual Private Cloud のエンドポイントを作成するを参照してください。

追加の診断ログを収集する

Systems Manager send コマンドを使用して VSS スナップショットドキュメントを実行するときに追加の診断ログを収集するには、実行時に入力パラメータ `CollectDiagnosticLogs` を「`True`」に設定します。トラブルシューティング時に、このパラメータは「`True`」に設定することを推奨します。

コマンドラインの例を確認するには、次のいずれかのタブを選択してください。

AWS CLI

次の例では、AWS CLI で AWSEC2-CreateVssSnapshot Systems Manager ドキュメントを実行します。

```
aws ssm send-command \
    --document-name "AWSEC2-CreateVssSnapshot" \
    --instance-ids "i-1234567890abcdef0" \
    --parameters '{"description":["Example - create diagnostic logs at
 runtime."],"tags":["Key=tag_name,Value=tag_value"],"CollectDiagnosticLogs":
["True"]}'
```

PowerShell

次の例では、PowerShell で AWSEC2-CreateVssSnapshot Systems Manager ドキュメントを実行します。

```
Send-SSMCommand `
    -DocumentName "AWSEC2-CreateVssSnapshot" `
    -InstanceId "i-1234567890abcdef0" `
    -Parameter @{'description'='Example - create diagnostic logs at
 runtime.';'tags'='Key=tag_name,Value=tag_value';'CollectDiagnosticLogs'='True'}
```

プロキシが設定されたインスタンスでの VSS の使用

プロキシを使用して EC2 エンドポイントにアクセスするインスタンスで VSS ベースの EBS スナップショットを作成する際に問題が発生した場合は、インスタンスで次の設定を確認します。

- SYSTEM として実行している AWS Tools for Windows PowerShell がインスタンスのリージョンおよび IMDS の EC2 サービスエンドポイントにアクセスできるようにプロキシが設定されていることを確認します。
- システムによって設定された WinHTTP プロキシの使用をサポートするため、インスタンスに AwsVssComponents の最新バージョンがインストールされていることを確認します。WinHTTP プロキシ設定の詳細については、Microsoft ウェブサイトの「Netsh Commands for Windows Hypertext Transfer Protocol (WINHTTP)」を参照してください。

エラー: パイプ接続の解凍がタイムアウトしました、解凍時のエラー、VSS フリーズ待機のタイムアウト、またはその他のタイムアウトエラー

EC2 Windows VSS プロバイダーは、インスタンスでのアクティビティまたはサービスにより、VSS ベースのスナップショットが適切なタイミングで進行できないためにタイムアウトすることがあります。Windows VSSフレームワークでは、ファイルシステムとの通信が一時停止される、設定不可の 10 秒のウィンドウがもたらされます。この間、AWSEC2-CreateVssSnapshot はボリュームのスナップショットを作成します。

以下の問題により、スナップショットの作成中に EC2 Windows VSS Provider が時間制限に達する可能性があります。

- ボリュームへの過剰な I/O
- インスタンスでの EC2 API の応答が遅い
- 断片化されたボリューム
- 一部のウイルス対策ソフトウェアとの非互換性
- VSSアプリケーションライターの問題
- 多数の PowerShell モジュールに対してモジュールログ記録が有効になっている場合、PowerShell スクリプトの実行速度が遅くなる可能性があります

AWSEC2-CreateVssSnapshot コマンドドキュメントを実行するときに発生するタイムアウトの問題のほとんどは、バックアップ時にインスタンスのワークロードが高すぎることが原因です。次のアクションは、スナップショットを正常に作成するのに役立ちます。

- AWSEC2-CreateVssSnapshot コマンドを再試行して、スナップショットの試行が成功するかどうかを確認します。一部で再試行が成功した場合、インスタンスの負荷を減らすことで、スナップショットがより成功する可能性があります。
- インスタンスのワークロードが減少するまでしばらく待ってから、AWSEC2-CreateVssSnapshot コマンドを再試行します。または、インスタンスが低ストレスであることがわかっている場合に、スナップショットの作成を試みることもできます。
- システムのウイルス対策ソフトウェアがオフになっているときに、VSS スナップショットの作成を試みます。これで問題が解決する場合は、ウイルス対策ソフトウェアの指示を参照し、VSS スナップショットの作成を許可するように設定します。
- スナップショットを実行しているリージョンと同じリージョン内のアカウントで大量の Amazon EC2 API コールがある場合、API スロットリングによりスナップショットの操作が遅れる可能性が

あります。スロットリングの影響を減らすには、最新の AwsVssComponents パッケージを使用します。このパッケージは EC2 CreateSnapshots API アクションを利用して、ボリュームごとのスナップショットの作成やタグ付けなど、変異アクションの数を減らします。

- 複数の AWSEC2-CreateVssSnapshot コマンドスクリプトを同時に実行している場合は、次のステップを実行して同時実行の問題を減らすことができます。
  - API アクティビティが少ない時間帯にスナップショットをスケジュールすることを検討してください。
  - Systems Manager コンソールで Run Command を使用 (または API で SendCommand を使用) してコマンドスクリプトを実行する場合は、Systems Manager のレート制御を使用して同時実行を減らすことができます。

    Systems Manager のレート制御を使用すると、Systems Manager を使用してコマンドスクリプトを実行する AWS Backup のようなサービスの同時実行を減らすこともできます。
- シェルで vssadmin list writers コマンドを実行し、システム上のいずれかのライターの [Last error] フィールドにエラーが報告されるかどうかを確認します。いずれかのライターがタイムアウトエラーを報告する場合は、インスタンスの負荷が少ないときにスナップショットの作成の再試行を検討してください。
- *t2 | t3 | t3a*.nano や *t2 | t3 | t3a*.micro のような小さいインスタンスタイプを使用すると、メモリと CPU の制約によりタイムアウトが発生する可能性があります。タイムアウトの問題を減らすには、以下の対処が有効な場合があります。
  - スナップショットを撮る前に、メモリや CPU を大量に消費するアプリケーションを閉じる。
  - インスタンスのアクティビティが少ない時間帯にスナップショットを撮る。

エラー: メソッドを呼び出せません。メソッド呼び出しは、この言語モードのコアタイプでのみサポートされます。

このエラーは、PowerShell 言語モードが FullLanguage に設定されていない場合に発生します。AWSEC2-CreateVssSnapshot SSM ドキュメントでは、PowerShell を FullLanguage モードに設定する必要があります。

言語モードを確認するには、PowerShell コンソールでインスタンスに次のコマンドを実行します。

```
$ExecutionContext.SessionState.LanguageMode
```

言語モードの詳細については、Microsoft ドキュメントの「Shell」を参照してください。

AWS VSS ソリューションを使用してインスタンスのデータを復元する

AWS VSS ソリューションによって作成された VSS ベースのスナップショットから Windows インスタンスに EBS ボリュームを復元できます。AWS VSS ソリューションスナップショットに Microsoft SQL Server データベースのバックアップが含まれている場合は、`AWSEC2-RestoreSqlServerDatabaseWithVss` AWS Systems Managerオートメーションランブックを使用してデータベースを復元できます。

データベース復元ランブックは、スナップショットからのボリュームの作成やインスタンスへのアタッチなど、復元プロセス全体を自動化します。自動化は VSS テクノロジーを活用してデータベースを復元するため、SQL Server アプリケーションを停止したり、アクティブな接続を切断したりすることなく復元できます。

Microsoft SQL Server データベースランブックの使用方法の詳細については、「Microsoft SQL Server on Amazon EC2 User Guide」の「Restore from VSS based snapshots」を参照してください。

AWS VSS ソリューションスナップショットから EBS ボリュームを復元するスクリプトをカスタマイズする

`RestoreVssSnapshotSampleScript.ps1` スクリプトをモデルとして使用して、AWS VSS ソリューションスナップショットから EBS ボリュームを復元する独自のカスタムスクリプトを作成できます。サンプルスクリプトは以下のタスクを実行します。

- インスタンスを停止する
- インスタンスからすべての既存のドライブを削除する (ルートボリュームが除外されている場合はそれを除外)
- スナップショットから新しいボリュームを作成する
- スナップショットのデバイス ID タグを使用してインスタンスにボリュームをアタッチする
- インスタンスを再起動する

> **Important**
>
> 以下のスクリプトは、インスタンスにアタッチされたすべてのボリュームをデタッチし、スナップショットから新しいボリュームを作成します。インスタンスを適切にバックアップしていることを確認してください。古いボリュームは削除されません。必要に応じて、古いボリュームを削除するようにスクリプトを編集できます。

VSS ベースの EBS スナップショットからボリュームを復元するには

1. RestoreVssSnapshotSampleScript.zip ファイルをダウンロードして、ファイルの内容を展開します。
2. `RestoreVssSnapshotSampleScript.ps1` をテキストエディタで開き、スクリプトの最後にあるサンプル呼び出しを編集して有効な EC2 インスタンス ID と EBS スナップショット ID を反映し、このスクリプトを PowerShell から実行します。

AWS VSS ソリューションのバージョン履歴

このページには、AWS VSS コンポーネントパッケージのバージョン別のリリースノートと、サポートされている各バージョンの Windows Server のコンポーネントおよびスクリプトのバージョン要件が含まれています。

トピック

- AwsVssComponents パッケージのバージョン
- Windows OS バージョンのサポート

AwsVssComponents パッケージのバージョン

以下の表に、AWS VSS コンポーネントパッケージのリリース済みバージョンを示します。

バージョン	詳細	リリース日	ダウンロード可能
2.5.1	ターゲットデータベースパラメータを指定したときに SQL データベースの復元が失敗する可能性があるケースを修正しました。	2025 年 3 月 13 日	はい
2.5.0	• VSS メタデータファイルを読み取ってインスタンス上の Microsoft SQL Server データベースを復元する機能を追加しました。詳細については、「Microsoft SQL Server on Amazon EC2 User Guide」の「Restore from VSS based snapshots」を参照してください。	2025 年 1 月 17 日	はい

バージョン	詳細	リリース日	ダウンロード可能
	• `AwsVssComponents` パッケージのインストールまたはアップグレード時のインプレース更新オプションのサポートが追加されました。		
2.4.0	スナップショット作成時に VSS メタデータファイルを保存する機能を追加しました。この機能を有効にするには、「Systems Manager VSS スナップショットドキュメントのパラメータ」の SaveVssMetadata を参照してください。	2024 年 10 月 7 日	はい
2.3.3	スナップショットの作成時に `Ec2VssProvider` が使用されるように、VSS エージェントを更新しました。	2024 年 6 月 25 日	はい
2.3.2	アンインストール時に VSS プロバイダー登録が削除されないケースを修正しました。	2024 年 5 月 9 日	はい
2.3.1	AWS VSS によって作成されたスナップショットと AMI を識別するための新しいデフォルトタグ `AwsVssConfig` を追加しました。	2024 年 3 月 7 日	はい

バージョン	詳細	リリース日	ダウンロード可能
2.2.1	• `DescribeInstanceAttribute` API の使用のサポートを追加しました。 • バグ修正と信頼性の向上。 • Windows Server 2012 および 2012 R2 のサポートを終了しました。AWSWindows Server 2012 および 2012 R2 での VSS コンポーネントバージョン 2.2.1 のインストールは失敗します。AWSVSS コンポーネントバージョン 2.1.0 は、Windows Server 2012 および 2012 R2 をサポートする最後のバージョンです。	2024 年 1 月 18 日	はい
2.1.0	`CreateSnapshots` API の使用のサポートを追加しました。	2023 年 11 月 6 日	はい
2.0.1	WinHTTP プロキシ設定を使用するためのサポートが追加されました。	2023 年 10 月 26 日	いいえ
2.0.0	AWS VSS コンポーネントにスナップショットと AMI を作成する機能が追加されました。これにより、PowerShell モジュールロギング、スクリプトブロックロギング、およびトランスクリプション機能との互換性が可能になりました。	2023 年 4 月 28 日	いいえ
1.3.2.0	インストールの失敗が正しく報告されないケースを修正しました。	2022 年 5 月 10 日	いいえ

バージョン	詳細	リリース日	ダウンロード可能
1.3.1.0	• NTDS VSS ライターのログ記録エラーに関連してドメインコントローラーで発生するスナップショットのエラーを修正しました。 • バージョン 1.0 VSS プロバイダーのアンインストール時に発生する VSS エージェントのエラーを修正しました。	2020 年 2 月 6 日	はい
1.3.00	• 不要な冗長性を減らすことで、ログ記録を改良しました。 • インストール中のリージョン化の問題を修正しました。 • 一部のプロバイダー登録エラー条件のリターンコードを修正しました。 • インストールのさまざまな問題を修正しました。	2019 年 3 月 19 日	いいえ
1.2.00	• コマンドラインパラメータ -nw (ライターなし) および -copy (コピーのみ) をエージェントに追加しました。 • 不適切なメモリ割り当て呼び出しによって発生する EventLog エラーを修正しました。	2018 年 11 月 15 日	いいえ
1.1	デフォルトの Windows バックアップおよび復元プロバイダとして誤って使用されていた AWS VSS コンポーネントを修正しました。	2017 年 12 月 12 日	いいえ

バージョン	詳細	リリース日	ダウンロード可能
1.0	初回リリース。	2017 年 11 月 20 日	いいえ

Windows OS バージョンのサポート

以下のテーブルでは、Amazon EC2 で Windows Server の各バージョンで実行する必要のある AWS VSS ソリューションのバージョンを示しています。

Windows サーバーバージョン	AwsVssComponents バージョン	AWSEC2-VssInstallAndSnapshot のバージョン名	AWSEC2-CreateVssSnapshot バージョン名
Windows Server 2025	デフォルト	デフォルト	デフォルト
Windows Server 2022	デフォルト	デフォルト	デフォルト
Windows Server 2019	デフォルト	デフォルト	デフォルト
Windows Server 2016	デフォルト	デフォルト	デフォルト
Windows Server 2012 R2	2.1.0	サポート外	2012R2

Windows サーバーバージョン	AwsVssComponents バージョン	AWSEC2-VssInstallAndSnapshot のバージョン名	AWSEC2-CreateVssSnapshot バージョン名
Windows Server 2012	2.1.0	サポート外	2012R2
Windows サーバー 2008 R2	1.3.1.0	サポート外	2008R2

Amazon EC2 でのオブジェクトストレージ、ファイルストレージ、およびファイルキャッシュ

クラウドファイルストレージは、クラウド上にデータを保存する方法で、サーバーとアプリケーションは共有ファイルシステムを通してデータにアクセスできます。クラウドファイルストレージが持つこの互換性は共有ファイルシステムに依存するワークロードに最適で、コードを変更することなく統合を容易に行えます。

スケーラビリティを持たない、またはデータを保護するための冗長性がほとんどないブロックストレージを基盤として使用する、コンピューティングインスタンス上の単一ノードのファイルサーバーから、独自のクラスター化されたソリューション、完全マネージド型のソリューションまで、さまざまなファイルストレージソリューションがあります。以下のコンテンツでは、Amazon EC2 インスタンスで使用するために AWS で提供されているストレージサービスの一部を紹介します。

内容

- Amazon EC2 インスタンスでの Amazon S3 の使用
- Amazon EC2 EC2 Linux インスタンスでの Amazon EC2 EFS の使用
- Amazon EC2 インスタンスでの Amazon FSx の使用
- Amazon EC2 インスタンスでの Amazon File Cache の使用

Amazon EC2 インスタンスでの Amazon S3 の使用

Amazon Simple Storage Service (Amazon S3) は業界をリードするスケーラビリティ、データ可用性、セキュリティ、およびパフォーマンスを提供するオブジェクトストレージサービスです。Amazon S3 を使用して、データレイク、ウェブサイト、バックアップ、ビッグデータ分析など、さまざまなユースケースの任意の量のデータを Amazon EC2 インスタンスから、またはインターネット経由でどこからでも保存および取得できます。詳細については「Amazon S3 とは」を参照してください。

Amazon EC2 インスタンスから Amazon S3 データにアクセスする方法は 2 つあります:

- ファイルアクセス — Amazon S3 ファイルを使用して、インスタンスでの S3 バケットを高性能ファイルシステムとしてマウントします。
- オブジェクトアクセス – Amazon S3 API、AWS CLI、AWS SDK、または wget などのツールを使用して、S3 との間でオブジェクトをコピーします。

Amazon S3 ファイルによるファイルアクセス

Amazon S3 Files は、S3 汎用バケットをコンピューティングインスタンスでの高性能ファイルシステムとしてマウントできるサーバーレスファイルシステムです。S3 ファイルでは、ローカルマウントパスでの読み取りと書き込みなどの標準ファイルシステムオペレーションを使用して、S3 オブジェクトにファイルとしてアクセスできます。

S3 ファイルシステムは、起動時、または実行中のインスタンスに対して後から、EC2 インスタンスにマウントできます。

前提条件

EC2 インスタンスで S3 ファイルをセットアップする前に、以下があることを確認してください:

- 使用可能な状態の S3 ファイルシステムと少なくとも 1 つのマウントターゲット。S3 ファイルシステムの作成の詳細については、「Amazon S3 ユーザーガイド」の「Amazon S3 ファイルの使用」を参照してください。
- アタッチされたインスタンスプロファイルがある Amazon EC2 インスタンス。ファイルシステムをマウントするために必要なアクセス許可の詳細については、「Amazon S3 ユーザーガイド」の「IAM ロールとポリシー」を参照してください。
- インスタンスとファイルシステムのマウントターゲット間の NFS トラフィック (ポート 2049) を許可するセキュリティグループ。必要なセキュリティグループ設定の詳細については、「Amazon S3 ユーザーガイド」の「セキュリティグループ」を参照してください。

EC2 コンソールを使用して起動時に EC2 インスタンスにファイルシステムをマウントするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [インスタンスを起動] を選択してください。
3. [Network settings] (ネットワーク設定) で、次の操作を行います :
   a. [編集] を選択します。
   b. [Subnet (サブネット)] で、サブネットを選択してください。
   c. デフォルトのセキュリティグループを選択して、EC2 インスタンスが S3 ファイルシステムにアクセスできることを確認します。このセキュリティグループを使用して、Secure Shell (SSH) で EC2 インスタンスにアクセスすることはできません。SSH によるアクセスについては、後でデフォルトのセキュリティグループを編集し、SSH を許可するルールを追加するか、SSH を許可する新しいセキュリティグループを追加できます。以下の設定を使用できます。
      i. タイプ: SSH
      ii. [Protocol]: TCP
      iii. ポート範囲: 22
      iv. 出典: Anywhere 0.0.0.0/0
4. [ストレージの設定] で、次の作業を行います:
   a. [ファイルシステム] で、[S3 Files] を選択します。
   b. [Add shared file system] (共有ファイルシステムの追加) を選択します。
   c. [S3 ファイルシステム] では、ファイルシステムは、ネットワーク設定で選択したサブネットに基づいて、アベイラビリティーゾーンに表示されます。マウントする S3 ファイルシステムを選択します。ファイルシステムがない場合は、[新しいファイルシステムを作成する] を選択して新しいファイルシステムを作成します。
   d. ファイルシステムをマウントする EC2 インスタンスのローカルマウントパスを入力します (例: `/mnt/s3files`)。
   e. ファイルシステムをマウントし、fstab に追加するコマンドが生成されます。このコマンドは、[高度な詳細] の [ユーザーデータ] フィールドに追加できます。その後、EC2 インスタンスは、起動時および再起動されるたびに S3 ファイルシステムをマウントするよう設定されます。これらのコマンドは、起動後に EC2 インスタンスで実行することもできます。
5. [高度な詳細] で、インスタンスプロファイルをインスタンスにアタッチします。IAM ロールには、ファイルシステムをマウントし、S3 バケットにアクセスするためのアクセス許可がある

必要があります。必要なアクセス許可の詳細については、「Amazon S3 ユーザーガイド」の「IAM ロールとポリシー」を参照してください。

6. [インスタンスを起動] を選択してください。

   インスタンスの起動後、必要なソフトウェアユーティリティがインストールされ、ファイルシステムがマウントされます。ローカルマウントパスに移動することで、ファイルシステムを表示できます。

起動後に EC2 インスタンスにファイルシステムをマウントするには

1. Secure Shell (SSH) を介して、または EC2 コンソールで EC2 Instance Connect を使用して EC2 インスタンスに接続します。
2. S3 ファイルシステムをマウントするには、マウントヘルパーユーティリティ `amazon-efs-utils` を使用します。Linux ディストリビューションに応じて、次のいずれかのコマンドを使用して `amazon-efs-utils` パッケージをインストールします:

   a. Amazon Linux を使用している場合は、次のコマンドを実行して Amazon のリポジトリから efs-utils をインストールします:

   ```
   sudo yum -y install amazon-efs-utils
   ```

   b. その他のサポートされている Linux ディストリビューションを使用している場合は、次のコマンドを実行します:

   ```
   curl https://amazon-efs-utils.aws.com/efs-utils-installer.sh | sudo sh -s -- --install
   ```

   c. その他の Linux ディストリビューションについては、GitHub の efs-utils リポジトリを参照してください。
3. 次のコマンドを使用して、ファイルシステムのマウントポイントのディレクトリを作成します:

   ```
   sudo mkdir {path/to/mount}
   ```

4. S3 ファイルシステムをマウントします:

   ```
   FS="{YOUR_FILE_SYSTEM_ID}"
   sudo mount -t s3files $FS:/ {path/to/mount}
   ```

5. ファイルシステムがマウントされていることを確認します:

```
df -h {path/to/mount}
```

S3 バケット内のオブジェクトをファイルとして表示するには

前述の手順を完了したので、標準のファイルシステムオペレーションを使用して、S3 オブジェクトをローカルマウントパス上のファイルとして読み書きできるようになりました。S3 バケットにオブジェクトがある場合は、次のコマンドを使用して、オブジェクトをファイルとして表示できます:

```
ls {path/to/mount}
```

オブジェクトベースのアクセス

S3 API、AWS CLI、AWS SDK、または標準 HTTP ツールを使用して、Amazon S3 との間でファイルをコピーできます。必要なアクセス許可がある場合は以下の方法を使用して、Amazon S3 とインスタンスとの間でファイルをコピーできます。

wget

> **Note**
>
> この手法はパブリックなオブジェクトに対してのみ有効です。オブジェクトがパブリックでない場合は`ERROR 403: Forbidden`メッセージが出力されます。このエラーを受け取った場合はAmazon S3 コンソール、AWS CLI、AWS API、AWS SDK、または AWS Tools for Windows PowerShell を使用する必要があります。この際は適切なアクセス許可が必要です。詳細については、Amazon S3 ユーザーガイドのAmazon S3 での Identity and Access Managementおよびオブジェクトのダウンロードを参照してください。

wget ユーティリティは、Amazon S3 からパブリックオブジェクトをダウンロードできる HTTP および FTP のクライアントです。これは Amazon Linux やその他のほとんどのディストリビューションにデフォルトでインストールされ、Windows ではダウンロード可能です。Amazon S3 オブジェクトをダウンロードするには次のコマンドを入力し、ダウンロードするオブジェクトの URL に置き換えます。

```
[ec2-user ~]$ wget https://amzn-s3-demo-bucket.s3.amazonaws.com/path-to-file
```

PowerShell

AWS Tools for Windows PowerShell を使用して、Amazon S3 との間でオブジェクトを移動できます。

次のように、Copy-S3Object コマンドレットを使用して、Amazon S3 オブジェクトを Windows インスタンスにコピーします。

```
Copy-S3Object `
    -BucketName amzn-s3-demo-bucket `
    -Key path-to-file `
    -LocalFile my_copied_file.ext
```

あるいは、Windows インスタンスでウェブブラウザを使用して Amazon S3 コンソールを開くこともできます。

AWS CLI

AWS Command Line Interface (AWS CLI) を使用して Amazon S3 から制限された項目をダウンロードし、項目をアップロードできます。ツールのインストールおよび設定方法などの詳細についてはAWS Command Line Interface の詳細ページを参照してください。

aws s3 cp コマンドは Unix cp コマンドと似ています。ファイルを Amazon S3 からインスタンスにコピーしたり、ファイルをインスタンスから Amazon S3 にコピーしたりできるほか、ファイルを Amazon S3 の 1 つの場所から別の場所にコピーすることもできます。

オブジェクトを Amazon S3 からインスタンスにコピーするには次のコマンドを使用します:

```
aws s3 cp s3://amzn-s3-demo-bucket/my_folder/my_file.ext my_copied_file.ext
```

オブジェクトをインスタンスから Amazon S3 にコピーして戻すには次のコマンドを使用します:

```
aws s3 cp my_copied_file.ext s3://amzn-s3-demo-bucket/my_folder/my_file.ext
```

aws s3 sync コマンドを使用すると、Amazon S3 バケット全体をローカルディレクトリの場所に同期できます。この機能はデータセットをダウンロードし、リモートセットでローカルコピーを最新の状態に保つ際に役立ちます。Amazon S3 バケットに対して適切なアクセス許可がある場合

はコマンドで送信元と送信先の場所を入れ替えることで、終了時にローカルディレクトリバックアップをクラウドにプッシュできます。

Amazon S3 バケット全体をインスタンスのローカルディレクトリにダウンロードするには次のコマンドを使用します:

```
aws s3 sync s3://amzn-s3-demo-source-bucket local_directory
```

Amazon S3 API

API を使用して Amazon S3 のデータにアクセスできます。この API はアプリケーションの開発および、他の API および SDK との統合に役立てることができます。詳細については「Amazon Simple Storage Service API リファレンス」の「 SDK を使用した Amazon S3 のコード例 AWSSDKs」を参照してください。

Amazon EC2 EC2 Linux インスタンスでの Amazon EC2 EFS の使用

> **Note**
>
> Amazon EC2 EFS は Windows インスタンスではサポートされていません。

Amazon EC2 EFS はAmazon EC2 EC2 と併用できるスケーラブルなファイルストレージを提供します。複数のインスタンスで実行している作業負荷やアプリケーションの一般的なデータソースとしてEFSファイルシステムを使用できます。詳細についてはAmazon EC2 Elastic File System 製品ページを参照してください。

このチュートリアルではインスタンスの起動時に、Amazon EC2 EFS クイック作成ウィザードを使用して Amazon EC2 EFS ファイルシステムを作成し、アタッチする方法について解説します。Amazon EC2 EFS コンソールを使用してファイルシステムを作成する方法に関するチュートリアルについては「Amazon EC2 Elastic File System User Guide」(Amazon EC2 Elastic File System ユーザーガイド) の「Amazon EC2 Elastic File System の使用開始」を参照してください。

> **Note**
>
> EFS クイック作成を使用して EFS ファイルシステムを作成すると、次のサービス推奨設定でファイルシステムが作成されます。
>
> - 自動バックアップを有効化しました。

- 選択した VPC のマウントターゲットを管理します。
- 汎用パフォーマンスモード。
- バーストスループットモード。
- Amazon EC2 EFS (aws/elasticfilesystem) のデフォルトのキーを使用して、保管中のデータの暗号化を有効にしました。
- 30日間のポリシーで Amazon EC2 EFS ライフサイクル管理を有効にしました。

タスク

- Amazon EC2 EFS クイック作成を使用した EFS ファイルシステムの作成
- EFS ファイルシステムをテストする
- EFS ファイルシステムを削除する

Amazon EC2 EFS クイック作成を使用した EFS ファイルシステムの作成

Amazon EC2 EC2 インスタンス起動ウィザードの Amazon EC2 EFS クイック作成機能を使用してインスタンスを起動するときに、EFS ファイルシステムを作成してインスタンスにマウントできます。

Amazon EC2 EFS クイック作成を使用して EFS ファイルシステムを作成するには

1. Amazon EC2 EC2 コンソール のhttps://console.aws.amazon.com/ec2/ を開いてください。
2. [インスタンスを起動] を選択してください。
3. (オプション) [Names and tags] (名前とタグ) における [Name] (名前) ではインスタンスを識別するための名前を入力してください。
4. [Application and OS Images (Amazon EC2 マシンイメージ)] (アプリケーションおよび OS イメージ (Amazon EC2 マシンイメージ)) で Linux オペレーティングシステムを選択し、[Amazon EC2 マシンイメージ (AMI)] (Amazon EC2 マシンイメージ (AMI)) で Linux AMI を選択してください。
5. [Instance type] (インスタンスタイプ) の [Instance type](インスタンスタイプ) で、インスタンスタイプを選択するか、デフォルトのままにします。
6. [Key pair (login)] (キーペア (ログイン)) の [Key pair name] (キーペア名) で、既存のキーペアを選択するか、新しいキーペアを作成します。

7. [Network settings] (ネットワーク設定) で [Edit] (編集) (右側) を選択し、[Subnet] (サブネット) でサブネットを選択してください。

> **Note**
>
> EFS ファイルシステムを追加する前に、サブネットを選択する必要があります。

8. [Configure storage] (ストレージを設定) で、[Edit](編集) (右下) を選択し、次の操作を実行します。

   a. [ファイルシステム] で EFS が選択されていることを確認し、[新しい共有ファイルシステムを作成] を選択してください。

   b. [ファイルシステム名] に Amazon EC2 EFS ファイルシステムの名前を入力し、[ファイルシステムの作成] を選択してください。

   c. [マウントポイント] で、カスタムのマウントポイントを指定するか、デフォルトのままにします。

   d. ファイルシステムへのアクセスを有効にするには[Automatically create and attach security groups] (セキュリティグループを自動的に作成してアタッチ) を選択してください。このチェックボックスをオンにすると、次のセキュリティグループが自動的に作成され、ファイルシステムのインスタンスおよびマウントターゲットにアタッチされます。

      - インスタンスセキュリティグループ – NFS 2049 ポート経由のトラフィックを許可するアウトバウンドルールは含まれますが、インバウンドルールは含まれません。
      - ファイルシステムマウントターゲットセキュリティグループ — インスタンスセキュリティグループ (上述) からの NFS 2049 ポート経由のトラフィックを許可するインバウンドルールと、NFS 2049 ポート経由のトラフィックを許可するアウトバウンドルールが含まれます。

      > **Note**
      >
      > 代わりに、セキュリティグループを手動で作成してアタッチすることができます。セキュリティグループを手動で作成してアタッチする場合は[Automatically create and attach the required security groups] (必要なセキュリティグループを自動的に作成してアタッチ) をオフにします。

   e. インスタンスの起動時に共有ファイルシステムを自動的にマウントするには[Automatically mount shared file system by attaching required user data script] (必要なユーザーデータス

クリプトをアタッチして共有ファイルシステムを自動的にマウント) を選択してください。自動的に生成されたユーザーデータを表示するには[Advanced details] (高度な詳細) を展開し、[User data] (ユーザー データ) まで下方向にスクロールします。

> **Note**
>
> このチェックボックスをオンにする前にユーザーデータを追加した場合、自動的に生成されたユーザーデータによって元のユーザーデータが上書きされます。

9. 必要に応じて、その他のインスタンスの設定を行います。
10. [合計mary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択してください。詳細については「コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する」を参照してください。

EFS ファイルシステムをテストする

インスタンスに接続し、ファイルシステムが指定したディレクトリにマウントされていることを確認します (例えば、/mnt/efs)。

ファイルシステムがマウントされていることを確認するには

1. インスタンスに接続します。詳細については「SSH を使用した Linux インスタンスへの接続」を参照してください。
2. インスタンスのターミナルウィンドウから、df -T コマンドを実行して、EFS ファイルシステムがマウントされていることを確認します。

```
$ df -T
Filesystem     Type              1K-blocks    Used          Available Use% Mounted
 on
/dev/xvda1     ext4                8123812 1949800            6073764  25% /
devtmpfs       devtmpfs            4078468      56            4078412   1% /dev
tmpfs          tmpfs               4089312       0            4089312   0% /dev/shm
efs-dns        nfs4       9007199254740992       0   9007199254740992   0% /mnt/efs
```

なお、ファイルシステムの名前 (サンプル出力では *efs-dns* として表示) は次の形式になります。

```
file-system-id.efs.aws-region.amazonaws.com:/
```

3. (オプション) インスタンスからファイルシステムでファイルを作成し、別のインスタンスからファイルを表示できることを確認します。

   a. インスタンスから、次のコマンドを実行してファイルを作成します。

   ```
   $ sudo touch /mnt/efs/test-file.txt
   ```

   b. 他のインスタンスから、次のコマンドを実行してファイルを表示します。

   ```
   $ ls /mnt/efs
   test-file.txt
   ```

EFS ファイルシステムを削除する

ファイルシステムが不要になった場合にはそれを削除することができます。

ファイルシステムを削除するには

1. Amazon EC2 Elastic File System コンソール のhttps://console.aws.amazon.com/efs/ を開いてください。
2. 削除するファイル システムを選択してください。
3. [Actions]、[Delete file system] の順に選択してください。
4. 確認を求められたら、ファイルシステム ID を入力し、[Delete file system (ファイルシステムの削除)] を選択してください。

Amazon EC2 インスタンスでの Amazon FSx の使用

Amazon FSx ファミリーのサービスにより、人気のある市販のファイルシステムやオープンソースのファイルシステムで動作する共有ストレージを簡単に起動、実行、スケーリングできます。新しいインスタンス起動ウィザードを使用すると、起動時に Amazon EC2 インスタンスに次のタイプの Amazon FSx ファイルシステムを自動的にアタッチできます。

- Amazon FSx for NetApp ONTAP により、AWS クラウドのフルマネージド共有ストレージで NetApp ONTAP の人気のあるデータアクセス機能と管理機能を利用できます。
- Amazon FSx for OpenZFS により、人気のある OpenZFS ファイルシステムでフルマネージドの費用対効果の高い共有ストレージを利用できます。

> **Note**
>
> - この機能は、新しいインスタンス起動ウィザードでのみ使用できます。詳細については、「コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する」を参照してください
> - Amazon FSx for Windows File Server と Amazon FSx for Lustre ファイルシステムを起動時にマウントすることはできません。これらのファイルシステムは、起動後に手動でマウントする必要があります。

以前に作成した既存のファイルシステムをマウントすることも、起動時にインスタンスにマウントする新しいファイルシステムを作成することもできます。

トピック

- セキュリティグループとユーザーデータスクリプト
- 起動時に Amazon FSx ファイルシステムをマウントする

セキュリティグループとユーザーデータスクリプト

インスタンス起動ウィザードを使用して Amazon FSx ファイルシステムをインスタンスにマウントする場合、ファイルシステムへのアクセスを有効にするために必要なセキュリティグループを自動的に作成してアタッチするかどうかを選択できます。また、ファイルシステムをマウントして使用可能にするために必要なユーザーデータスクリプトを自動的に含めるかどうかを選択できます。

トピック

- セキュリティグループ
- ユーザーデータスクリプト

セキュリティグループ

ファイルシステムへのアクセスを有効にするために必要なセキュリティグループを自動的に作成する場合は、インスタンス起動ウィザードが 2 つのセキュリティグループを作成してアタッチします。1 つはインスタンスにアタッチされ、もう 1 つはファイルシステムにアタッチされます。セキュリティグループの要件の詳細については、「FSx for ONTAP file system access control with Amazon VPC」(Amazon VPC での FSx for ONTAP ファイルシステムアクセスコントロール) と「FSx for

OpenZFS file system access control with Amazon VPC」(Amazon VPC での FSx for OpenZFS ファイルシステムアクセスコントロール) を参照してください。

作成されてインスタンスにアタッチされたセキュリティグループにタグ `Name=instance-sg-1` を追加します。インスタンス起動ウィザードが Amazon FSx ファイルシステム用に新しいセキュリティグループを作成するたびに、 タグの値が自動的に増分されます。

セキュリティグループには次の出力ルールが含まれていますが、インバウンドルールは含まれていません。

アウトバウンドルール

プロトコルのタイプ	ポート番号	デスティネーション
UDP	111	##################
UDP	20001 ~ 20003	##################
UDP	4049	##################
UDP	2049	##################
UDP	635	##################
UDP	4045 ~ 4046	##################
TCP	4049	##################
TCP	635	##################
TCP	2049	##################
TCP	111	##################
TCP	4045 ~ 4046	##################
TCP	20001 ~ 20003	##################
すべて	すべて	##################

作成されてファイルシステムにアタッチされたセキュリティグループには、Name=fsx-sg-*1* というタグが付けられます。インスタンス起動ウィザードが Amazon FSx ファイルシステム用に新しいセキュリティグループを作成するたびに、 タグの値が自動的に増分されます。

セキュリティグループには次のルールが含まれます。

インバウンドルール

プロトコルのタイプ	ポート番号	ソース
UDP	2049	################
UDP	20001 ~ 20003	################
UDP	4049	################
UDP	111	################
UDP	635	################
UDP	4045 ~ 4046	################
TCP	4045 ~ 4046	################
TCP	635	################
TCP	2049	################
TCP	4049	################
TCP	20001 ~ 20003	################
TCP	111	################

アウトバウンドルール

プロトコルのタイプ	ポート番号	デスティネーション
すべて	すべて	0.0.0.0/0

ユーザーデータスクリプト

ユーザーデータスクリプトを自動的にアタッチする場合は、インスタンス起動ウィザードが次のユーザーデータをインスタンスに追加します。このスクリプトは、必要なパッケージをインストールし、ファイルシステムをマウントします。また、インスタンスが再起動されるたびにファイルシステムが自動的に再マウントされるようにインスタンスの設定を更新します。

```
#cloud-config
package_update: true
package_upgrade: true
runcmd:
- yum install -y nfs-utils
- apt-get -y install nfs-common
- svm_id_1=svm_id
- file_system_id_1=file_system_id
- vol_path_1=/vol1
- fsx_mount_point_1=/mnt/fsx/fs1
- mkdir -p "${fsx_mount_point_1}"
- if [ -z "$svm_id_1" ]; then printf "\n${file_system_id_1}.fsx.eu-
north-1.amazonaws.com:/${vol_path_1} ${fsx_mount_point_1} nfs4
 nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport,_netdev
 0 0\n" >> /etc/fstab; else printf "\n${svm_id_1}.${file_system_id_1}.fsx.eu-
north-1.amazonaws.com:/${vol_path_1} ${fsx_mount_point_1} nfs4
 nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport,_netdev 0
 0\n" >> /etc/fstab; fi
- retryCnt=15; waitTime=30; while true; do mount -a -t nfs4 defaults; if [ $? = 0 ] ||
 [ $retryCnt -lt 1 ]; then echo File system mounted successfully; break; fi; echo File
 system not available, retrying to mount.; ((retryCnt--)); sleep $waitTime; done;
```

起動時に Amazon FSx ファイルシステムをマウントする

起動時に新規または既存の Amazon FSx ファイルシステムをマウントするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択し、[Launch instance] (インスタンスの起動) を選択して、インスタンス起動ウィザードを開きます。
3. [Application and OS Images] (アプリケーションと OS イメージ) セクションで、使用する AMI を選択します。
4. [Instance type] (インスタンスタイプ) セクションで、インスタンスタイプを選択します。

5. [Key pair] (キーペア) セクションで、既存のキーペアを選択するか、新規にキーペアを作成します。
6. [Network settings] (ネットワーク設定) セクションで、以下の操作を行います。
   a. [編集] を選択します。
   b. 既存のファイルシステムをマウントする場合は、[Subnet] (サブネット) でファイルシステムの優先サブネットを選択します。パフォーマンスを最適化するために、ファイルシステムの優先サブネットと同じアベイラビリティーゾーンでインスタンスを起動することをお勧めします。

      インスタンスにマウントするファイルシステムを新規に作成する場合は、[Subnet] (サブネット) で、インスタンスを起動するサブネットを選択します。

      > **Important**
      >
      > 新しいインスタンス起動ウィザードで Amazon FSx の機能を有効にするサブネットを選択する必要があります。サブネットを選択しないと、既存のファイルシステムをマウントすることも、新規にファイルシステムを作成することもできなくなります。

7. [Storage] (ストレージ) セクションで、以下の操作を行います。
   a. 必要に応じてボリュームを設定します。
   b. [File systems] (ファイルシステム) セクションを展開し、[FSx] を選択します。
   c. [Add shared file system] (共有ファイルシステムの追加) を選択します。
   d. [File system] (ファイルシステム) で、マウントするファイルシステムを選択します。

      > **Note**
      >
      > 選択したリージョンのアカウントにあるすべての Amazon FSx for NetApp ONTAP ファイルシステムと Amazon FSx for OpenZFS ファイルシステムがリストに表示されます。

   e. ファイルシステムへのアクセスを有効にするために必要なセキュリティグループを自動的に作成してアタッチするには、[Automatically create and attach security groups] (セキュリティグループを自動的に作成してアタッチする) を選択します。セキュリティグループを手

動で作成する場合、チェックボックスをオフにします。詳細については、「セキュリティグループ」を参照してください。

f. ファイルシステムをマウントするために必要なユーザーデータスクリプトを自動的にアタッチするには、[Automatically mount shared file system by attaching required user data script] (必要なユーザーデータスクリプトをアタッチして、共有ファイルシステムを自動的にマウントする) を選択します。ユーザーデータスクリプトを手動で指定する場合、チェックボックスをオフにします。詳細については、「ユーザーデータスクリプト」を参照してください。

8. [Advanced] (詳細) セクションで、必要に応じて追加でインスタンスの設定を行います。
9. [Launch] (起動する) を選択します。

Amazon EC2 インスタンスでの Amazon File Cache の使用

Amazon File Cache は、AWS でフルマネージド型の高速キャッシュを提供します。これにより、データの保存場所に関係なく、ファイルデータを簡単に処理できるようになります。Amazon File Cache は、オンプレミスのファイルシステム、AWS ファイルシステム、Amazon Simple Storage Service (Amazon S3) バケットに保存されているデータの一時的な高性能ストレージの場所として機能します。この機能を使用すると、分散したデータセットを AWS 上のファイルベースのアプリケーションで統合ビューを使用して高速 (ミリ秒未満のレイテンシーと高いスループット) で利用できるようになります。詳細については、「Amazon File Cache ユーザーガイド」を参照してください。

Amazon File Cache は、最も一般的な Linux AMI で動作し、x86 ベースのインスタンスタイプおよび Graviton インスタンスタイプと互換性があります。オープンソースの Lustre クライアントを使用して、Amazon EC2 インスタンスからキャッシュにアクセスできます。キャッシュをマウントすることで、標準の Linux コマンドを使用してキャッシュ内のファイルやディレクトリを操作できます。Amazon EC2 インスタンスは、同じ仮想プライベートクラウド (VPC) 内の他のアベイラビリティーゾーンからキャッシュにアクセスできます。ただし、ネットワーク設定によって VPC 内のサブネット間でのアクセスが許可されている場合に限ります。共有 VPC にキャッシュを作成することもできます。

開始するには、「Amazon File Cache ユーザーガイド」の「Getting started with Amazon File Cache」を参照してください。

Amazon EC2 リソースの管理

リソースとは、ユーザーが操作できるエンティティです。リソースは、サービスの機能を使用する際に Amazon EC2 により作成されます。例えば、Amazon EC2 リソースには、イメージ、インスタンス、フリート、キーペア、およびセキュリティグループが含まれます。すべての Amazon EC2 リソースタイプには、リソースについて記述する属性が含まれます。例えば、名前、説明、リソース識別子、Amazon リソースネーム (ARN) です。

Amazon EC2 リソースは、そのリソースが存在する AWS リージョンまたはゾーンに固有のものです。例えば、Amazon マシンイメージ (AMI) は AWS リージョンに固有ですが、AMI から起動したインスタンスは起動元のゾーンに固有です。ARN を使用すると、アクセス許可ポリシーで Amazon EC2 リソースを指定できます。

AWS アカウントには、Amazon EC2 のデフォルトのクォータがあります。これらのクォータは、作成できるリソースの最大数を定義します。例えば、実行中のインスタンス全体での vCPU の最大数に対するクォータがあります。インスタンスを起動したり、停止しているインスタンスを開始したりすることでクォータを超えた場合、その操作は失敗します。

リソース ID またはタグを使用して、リージョンごとに AWS アカウントの特定のリソースを検索できます。複数のリージョン全体で特定のリソースまたはリソースタイプを検索するには、Amazon EC2 グローバルビューを使用します。

内容

- Amazon EC2 リソースのリージョンを選択する
- Amazon EC2 リソースの検索
- AWS Global View を使用して複数のリージョンにわたるリソースを表示する
- Amazon EC2 リソースのタグ付け
- Amazon EC2 の Service Quotas

Amazon EC2 リソースのリージョンを選択する

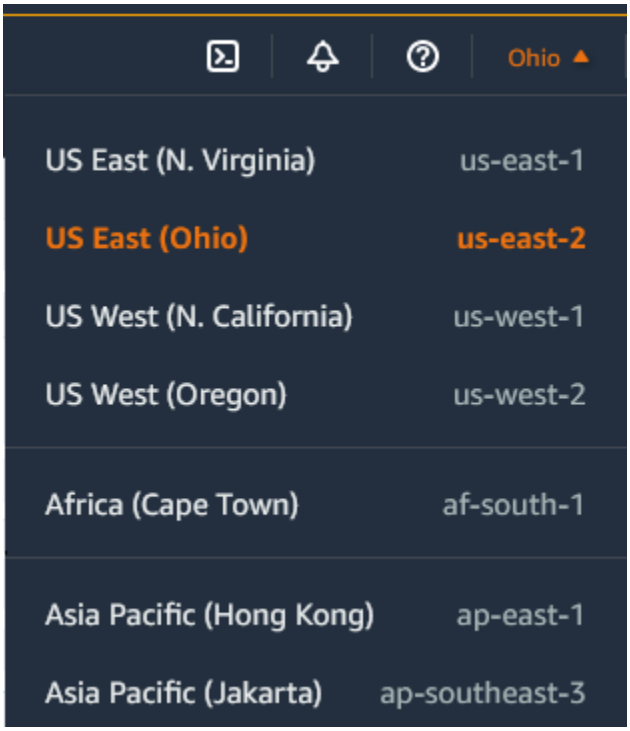
Amazon EC2 リソースは、そのリソースが存在する AWS リージョンまたはゾーンに固有のものです。Amazon EC2 リソースを作成する際に、リソースのリージョンを選択します。

考慮事項

- 一部の AWS リソースは、一部のリージョンで利用できない場合があります。リージョンでリソースの作成を開始する前に、リージョンで必要なすべての AWS リソースを作成できることを確認してください。
- 一部のリージョンはデフォルトで無効になっています。これらのリージョンを使用するには、有効にする必要があります。詳細については、「AWS リージョン」を参照してください。

コンソールを使用してリソースのリージョンを選択するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションバーで、[Regions] (リージョン) セレクタを選択し、リージョンを選択します。



3. リージョンセレクタには、AWS アカウントで使用できるすべてのリソースが含まれます。リストの下部付近にある下線付きのテキストを選択すると、アカウントで有効になっていないリージョンが表示されます。

AWS CLI を使用してリソースのリージョンを選択するには

デフォルトのリージョンを使用するように AWS CLI を設定できます。コマンドでリージョンを指定しない場合は、デフォルトのリージョンが使用されます。特定のコマンドに別のリージョンを使用するには、次のオプションを追加します。

```
--region us-east-1
```

Tools for PowerShell を使用してリソースのリージョンを選択するには

Tools for Windows PowerShell を、デフォルトのリージョンを使用するように設定できます。コマンドでリージョンを指定していない場合は、デフォルトのリージョンが使用されます。特定のコマンドに別のリージョンを使用するには、次のパラメータを追加します。

```
-Region us-east-1
```

Amazon EC2 リソースの検索

Amazon EC2 コンソールを使用して、一部のタイプのリソースのリストを取得できます。対応するコマンドまたは API アクションを使用して、各タイプのリソースのリストを取得できます。リソースが多い場合は所定の条件に一致するリソースのみを表示、または非表示にするように結果をフィルタリングすることができます。

内容

- コンソールを使用したリソースの一覧表示およびフィルタリング
- コマンドラインを使用した一覧表示およびフィルタリング
- AWS Global View を使用して複数のリージョンにわたるリソースを表示する

コンソールを使用したリソースの一覧表示およびフィルタリング

目次

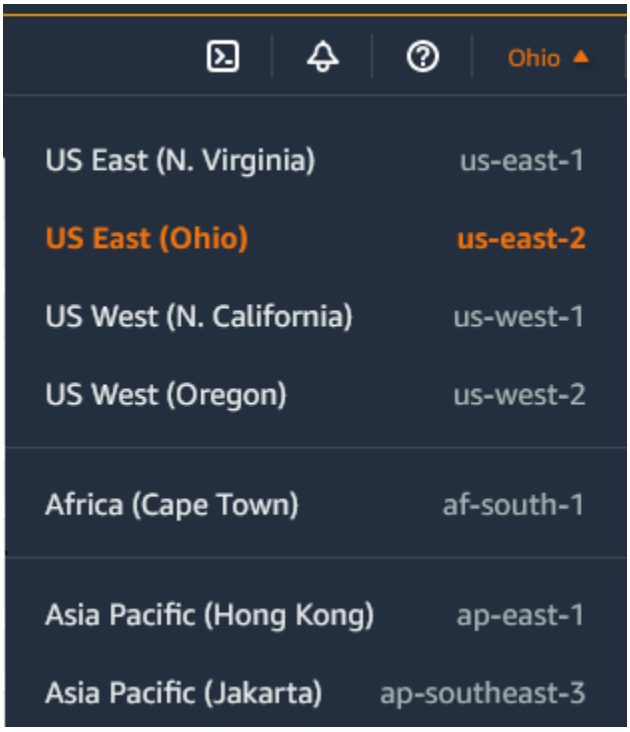
- コンソールを使用したリソースの一覧表示
- コンソールを使用したリソースのフィルタリング
  - サポートされているフィルタ
- コンソールを使用してフィルターセットの保存
  - 主な特徴
  - フィルターセットを作成
  - フィルターセットを変更
  - フィルターセットを削除

コンソールを使用したリソースの一覧表示

コンソールを使用して、最も一般的に使用される Amazon EC2 リソースタイプを表示できます。その他のリソースを表示するにはコマンドラインインターフェイスまたは API アクションを使用します。

コンソールを使用して EC2 リソースをリスト表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. Amazon EC2 リソースは AWS リージョンに固有です。ナビゲーションバーで、[リージョン] セレクタからリージョンを選択してください。



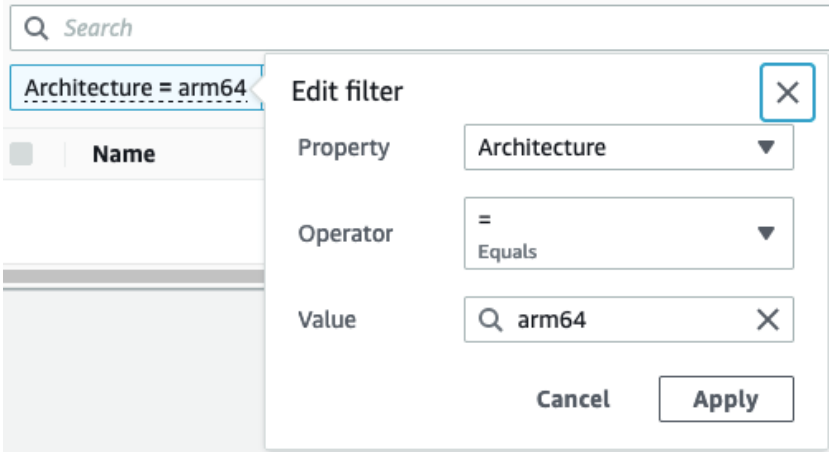
3. ナビゲーションペインで、リソースタイプに対応するオプションを選択してください。例えば、すべてのインスタンスを一覧表示するには[インスタンス] を選択してください。

コンソールを使用したリソースのフィルタリング

リソースリストをフィルタリングするには

1. ナビゲーションペインで、リソースタイプを選択します ([Instances] など)。
2. 検索フィールドを選択してください。
3. リストからフィルターを選択してください。

4. = (等しい))、などの演算子を選択してください。一部の属性では他の演算子を選択することもできます。すべての画面で演算子が選択できるわけではないことに注意してください。
5. フィルター値を選択してください。
6. 選択したフィルタを編集するにはフィルタートークン (青いボックス) を選択し、必要な編集を行った上で [Apply] (適用) をクリックしてください。すべての画面で、選択したフィルターの編集がサポートされているわけではないことに注意してください。



7. 完了したら、フィルターを削除します。

サポートされているフィルタ

Amazon EC2 コンソールでは2 種類のフィルタリングがサポートされています。

- API フィルタリングはサーバー側で行われます。このフィルタリングは API コールに適用され、サーバーから返されるリソースの数が減少します。これにより、大量のリソースにわたる迅速なフィルタリングが可能になり、サーバーとブラウザ間のデータ転送時間とコストを削減できます。API に関するフィルタリングでは[=] (等しい) および [:] (含む) の演算子が使用できます。また、常に大文字と小文字が区別されます。
- クライアントのフィルタリング はクライアント側で行われます。これにより、ブラウザで既に使用可能なデータ (つまり、APIによって既に返されたデータ) をフィルタリングできます。クライアントフィルタリングはブラウザ内の小さなデータセットまでフィルタリングするために、API フィルタと併用してうまく機能します。クライアントフィルタリングは、= (等しい) および : (含む) の演算子に加えて、>= (以上) のような範囲演算子や、!= (等しくない) などの否定 (反転) 演算子もサポートできます。

Amazon EC2 コンソールでは次のタイプの検索がサポートされます。

キーワードによる検索

キーワードによる検索は検索する属性やタグキーを指定せずに、リソースの属性とタグの全体から値を検索できる、フリーテキスト検索です。

> **Note**
>
> すべてのキーワード検索ではクライアントフィルタリングが使用されます。

キーワードで検索するには検索フィールドに検索したいキーワードを入力するか貼り付けて、Enter を選択してください。例えば 123 を検索すると、IP アドレス、インスタンス ID、VPC ID、AMI ID などの属性の中、あるいは Name などのタグの中に 123 が含まれる、すべてのインスタンスが一致します。フリーテキスト検索で予期しない一致が返された場合は追加のフィルタを適用します。

属性による検索

属性による検索ではすべてのリソースで特定の属性を検索できます。

> **Note**
>
> 属性検索では選択した属性に応じて、API フィルタリングまたはクライアントフィルタリングが使用されます。属性検索を実行すると、属性はそれに応じてグループ化されます。

例えば、すべてのインスタンスの [インスタンスの状態] 属性を検索して、`stopped` 状態にあるインスタンスのみを取得することができます。目的:

1. インスタンス画面の検索フィールドで、「`Instance state`」の入力を開始します。文字を入力すると、[Instance state (インスタンスの状態)] には[API filters (API フィルター)] と [Client filters (クライアントフィルター)] の 2 種類のフィルターが表示されます 。
2. サーバー側で検索するには[API filters (API フィルター)] で [Instance state (インスタンスの状態)] を選択してください。クライアント側で検索するには[Client filters (クライアントフィルター)] で [Instance state (client) (インスタンスの状態 (クライアント))] を選択してください。

   選択した属性に使用可能な演算子のリストが表示されます。
3. [=] (等しい) 演算子をクリックしてください。

   選択された属性と演算子に適合する可能性のある、値のリストが表示されます。

4. リストから [停止] を選択してください。

タグによる検索

タグによる検索では現在表示されているテーブル内のリソースを、タグキーまたはタグ値でフィルタリングできます。

タグ検索では[Preferences] (設定) ウィンドウの設定に応じて、API フィルタリングまたはクライアントフィルタリングのどちらかが使用されます。

タグに対し API フィルタリングが使用されるようにするには

1. [Preferences] (設定) ウインドウを開きます。
2. [正規表現一致を使用] チェックボックスをオフにします。このチェックボックスがオンの場合、クライアントのフィルタリングが実行されます。
3. [大文字と小文字を区別する一致を使用] チェックボックスをオンにします。このチェックボックスがオフの場合、クライアントのフィルタリングが実行されます。
4. [確認] を選択してください。

タグに関する検索では以下の値を使用できます。

- [(empty)] ((空)) – 指定したタグキーを持ち、かつタグ値を持たないすべてのリソースを検索します。
- [All values] (すべての値) – 指定したタグキーと任意のタグ値を持つすべてのリソースを検索します。
- [Not tagged] (タグ付けなし) – 指定したタグキーを持たないすべてのリソースを検索します。
- [The displayed value] (表示された値) – 指定したタグキーと指定したタグ値を持つすべてのリソースを検索します。

次のテクニックを使用して、検索の精度を高めたり、絞り込んだりできます。

逆順検索

逆検索では指定した値に一致しないリソースを検索できます。[インスタンス] 画面および [AMI] 画面での逆検索は、!= (等しくない) または !: (含まない) 演算子を選択してから、値を選択することで実行されます。他の場面では検索キーワードのプレフィックスに感嘆符 (!) を付けることによって逆検索が実行されます。

> **Note**
>
> 逆検索はクライアントフィルタのキーワード検索および属性検索でのみサポートされます。API フィルタの属性検索ではサポートされていません。

例えば、すべてのインスタンスの [インスタンスの状態] 属性を検索して、`terminated` 状態にあるインスタンスをすべて除外することができます。目的:

1. インスタンス画面の検索フィールドで、「`Instance state`」の入力を開始します。文字を入力すると、[Instance state (インスタンスの状態)] には[API filters (API フィルター)] と [Client filters (クライアントフィルター)] の 2 種類のフィルターが表示されます 。
2. [Client filters] (クライアントフィルター) で、[Instance state (client)] (インスタンスの状態 (クライアント)) を選択してください。逆検索はクライアントフィルタでのみサポートされます。

   選択した属性に使用可能な演算子のリストが表示されます。
3. [! =] (等しくない) を選択した上で、[terminated] (終了) をクリックしてください。

インスタンス状態の属性に基づいてインスタンスをフィルタリングするには[Instance state (インスタンスの状態)] 列の検索アイコン ( ) を使用することもできます。プラス記号 (+) が付いた検索アイコンはその属性に一致するすべてのインスタンスを表示します。マイナス記号 (-) が付いた検索アイコンはその属性に一致するすべてのインスタンスを除外します。

もう 1 つ逆検索の例を挙げます。`launch-wizard-1` という名前のセキュリティグループが割り当てられていないすべてのインスタンスを一覧表示するには[Client filters] (クライアントフィルター) で、[!=] を選択した上で検索バーに `launch-wizard-1` と入力し、[Security group name] (セキュリティグループ名) 属性による検索を行います。

部分検索

部分検索では部分文字列値を検索できます。部分検索を実行するには検索するキーワードの一部だけを入力してください。[Instances] (インスタンス) 画面、および [AMI] 画面では[:] (含む) 演算子を使用する場合のみ部分検索ができます。他の画面ではクライアントフィルター属性を選択して、キーワードの一部だけを直接入力して検索できます。例えば、[Instance type] (インスタンスタイプ) 画面で `t2.micro`、`t2.small`、`t2.medium` のすべてのインスタンスを検索するにはキーワードに `t2` を入力し、[Instance Type] (インスタンスタイプ) 属性で検索します。

正規表現検索

正規表現検索を使用するには[設定] ウィンドウの [正規表現一致を使用] チェックボックスをオンにする必要があります。

フィールド内の値を特定のパターンと一致させる場合、正規表現が役立ちます。例えば、s で始まる値を検索するには^s を検索します。xyz で終わる値を検索するにはxyz$ を検索します。または1 つ以上の文字が続く数字で始まる値を検索するには[0-9]+.* を検索します。

> **Note**
>
> 正規表現検索はクライアントフィルタでのキーワード検索および属性検索でのみサポートされます。API フィルタの属性検索ではサポートされていません。

大文字と小文字を区別する検索

大文字と小文字を区別する検索を使用するには[設定] ウィンドウの [大文字と小文字を区別する一致を使用] チェックボックスをオンにする必要があります。大文字と小文字を区別する設定はクライアントフィルタとタグフィルタにのみ適用されます。

> **Note**
>
> API フィルターでは常に大文字と小文字が区別されます。

ワイルドカード検索

0 文字以上の文字に一致させるには* ワイルドカードを使用します。0 文字または 1 文字に一致させるには? ワイルドカードを使用します。例えば、prod、prods、および production の値を含むデータセットの場合、prod* での検索はすべての値と一致しますが、prod? での検索は prod と prods にのみ一致します。リテラル値を使用するにはバックスラッシュ (\) でエスケープします。例えば、「prod*」 は「prod*」と一致します。

> **Note**
>
> ワイルドカード検索はAPI フィルタの属性およびタグ検索でのみサポートされます。これはキーワード検索、ならびに、クライアントフィルタでの属性とタグによる検索では使用できません。

検索を組み合わせる

一般に、同じ属性を持つ複数のフィルタはOR で自動的に結合されます。例えば、Instance State : Running および Instance State : Stopped を検索すると、実行中または停止中のすべてのインスタンスが返されます。AND で検索を結合するにはさまざまな属性を検索します。例えば、Instance State : Running および Instance Type : c4.large を検索すると、タイプが c4.large で、かつ実行状態のインスタンスだけが返されます。

コンソールを使用してフィルターセットの保存

保存されたフィルターセットはAmazon EC2 リソースを効率的に表示するために作成して再利用できるカスタマイズされたフィルターのグループです。この機能によってワークフローが合理化され、特定のリソースビューにすばやくアクセスできるようになります。

保存されたフィルターセットは Amazon EC2 コンソールでのみサポートされており、現在は [ボリューム] ページにのみ利用できます。

主な特徴

- カスタマイズ: ニーズに合わせてフィルターセットを作成します。例えば、フィルターセットを作成し、指定した日付以降に作成された gp3 ボリュームのみを表示できます。
- デフォルトフィルター: ページにデフォルトのフィルターセットを設定すると、ページに移動するとデフォルトフィルターが自動的に適用されます。デフォルトが設定されていない場合、フィルターは適用されません。
- 適用が簡単: 保存したフィルターセットを選択し、すぐに適用します。次に Amazon EC2 は関連するリソースを表示し、アクティブなフィルターは青いトークンで示されます。
- 柔軟性: 必要に応じてフィルターセットを作成、変更、削除します。

フィルターセットを作成

新規のフィルターセットを作成する方法

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインの [ボリューム] を選択します。

> **Note**
>
> 現在、保存されたフィルターセットは [ボリューム] でのみ利用可能です。

3. 検索フィールドでフィルターセットのフィルターを選択してください。
4. [フィルターをクリア] ボタンの横にある矢印を選択し、[新規のフィルターセットを保存] を選択してください。
5. [フィルターセットの保存] ウィンドウで次の操作を行います。
   a. [フィルターセット名] にはフィルターセットの名前を入力してください。
   b. (オプション) [フィルターセットの説明] にはフィルターセットの説明を入力してください。
   c. (オプション) フィルターセットをデフォルトフィルターとして設定するには[デフォルトとして設定] チェックボックスをオンにします。

      > **Note**
      >
      > コンソールページを開くたびに、デフォルトフィルターが自動的に適用されます。

   d. [Save] を選択してください。

フィルターセットを変更

フィルターを変更する方法

1. [保存されたフィルターセット] リストから変更するフィルターを選択してください。
2. フィルターを追加するには検索フィールドでフィルターセットに追加するフィルターを選択してください。セットのフィルターを削除するにはフィルタートークンの [X] を選択してください。
3. [フィルターをクリア] ボタンの横にある矢印を選択し、[フィルターセットの変更] を選択してください。
4. [フィルターセットの変更] ウィンドウで次の操作を行います。
   a. (オプション) フィルターセットをデフォルトフィルターとして設定するには[デフォルトとして設定] チェックボックスをオンにします。

> **Note**
>
> コンソールページを開くたびに、デフォルトフィルターが自動的に適用されます。

b. Modify を選択してください。

フィルターセットを削除

フィルターを削除する方法

1. [保存されたフィルターセット] リストから削除するフィルターを選択してください。
2. [フィルターをクリア] ボタンの横にある矢印を選択し、[フィルターセットの削除] を選択してください。
3. [フィルターセットの削除] ウィンドウでフィルターを確認して削除するフィルターを確定したら、[削除] を選択してください。

コマンドラインを使用した一覧表示およびフィルタリング

各リソースタイプにはそれぞれ、そのタイプのリソースの記述、一覧表示、取得に使用される、対応する API アクションがあります。結果として得られるリソースのリストは長くなる場合があるため、特定の条件に一致するリソースのみを含めるように結果をフィルタリングする方がより速く、より便利です。

フィルタリングの考慮事項

- 1 回のリクエストで、フィルターごとに最大 50 種類のフィルターと最大 200 個の値を指定できます。
- フィルターでは最大 255 文字を使用できます。
- フィルタの値にはワイルドカードを使用することもできます。アスタリスク (*) は 0 個以上の文字、クエスチョンマーク (?) は 0 文字または 1 文字にマッチングします。
- フィルタの値は大文字と小文字が区別されます。
- 検索にはワイルドカード文字のリテラル値を含めることができます。ただ、文字の前にバックスラッシュを使用してエスケープする必要があります。例えば、*amazon\?\\ という値ではリテラル文字列 *amazon?\ が検索されます。

- フィルター値として を指定することはできませんnull。代わりに、クライアント側のフィルタリングを使用します。例えば、次のコマンドは --queryオプションを使用して、キーペアなしで起動されたインスタンスの IDs を返します。

```
aws ec2 describe-instances \
    --query 'Reservations[*].Instances[?!not_null(KeyName)].InstanceId' \
    --output text
```

AWS CLI

Example例: 単一のフィルタを指定する

describe-instances を使用して Amazon EC2 インスタンスを一覧表示できます。フィルタを使用しないと、レスポンスにはすべてのリソースに関する情報が含まれます。次のコマンドを使用することで、実行中のインスタンスのみを出力に含めることができます。

```
--filters Name=instance-state-name,Values=running
```

実行中のインスタンスのインスタンス ID のみを一覧表示するには次のように `--query` オプションを追加します。

```
--query "Reservations[*].Instances[*].InstanceId"
```

Example例: 複数のフィルタまたはフィルタ値の指定

複数のフィルタまたは複数のフィルタ値を指定する場合は、リソースが、出力に追加されるすべてのフィルタと一致しなくてはなりません。

次のオプションを使用すると、タイプが `m5.large` または `m5d.large` のいずれかであるすべてのインスタンスを一覧表示できます。

```
--filters Name=instance-type,Values=m5.large,m5d.large
```

次のオプションを使用して、タイプが `t2.micro` であるすべての停止したインスタンスを一覧表示できます。

```
--filters Name=instance-state-name,Values=stopped Name=instance-type,Values=t2.micro
```

Example例: フィルタ値でのワイルドカードの使用

次のオプションを describe-snapshots と共に使用すると、説明が「データベース」であるスナップショットのみを返すことができます。

```
--filters Name=description,Values=database
```

ワイルドカード * はゼロ文字以上と一致します。次のオプションを使用すると、説明に database という単語が含まれるスナップショットのみを返すことができます。

```
--filters Name=description,Values=*database*
```

ワイルドカード ? は厳密に 1 文字に一致します。次のオプションを使用すると、説明が「database」または「database」とその後に 1 文字が続くスナップショットのみを返すことができます。

```
--filters Name=description,Values=database?
```

次のオプションを使用すると、説明が「データベース」とその後に最大 4 文字が続くスナップショットのみを返すことができます。「database」の後に 5 文字以上続く記述は除外されます。

```
--filters Name=description,Values=database????
```

Example例: 日付に基づくフィルタリング

AWS CLI では JMESPath を使用して、式を使用した結果をフィルタリングできます。例えば、次の describe-snapshots コマンドを実行すると、指定した AWS アカウントが指定した日より前に作成したすべてのスナップショットの ID が表示されます。所有者を指定しない場合、結果にはすべてのパブリックスナップショットが含まれます。

```
aws ec2 describe-snapshots \
    --filters Name=owner-id,Values=123456789012 \
    --query "Snapshots[?(StartTime<='2024-03-31')].[SnapshotId]" \
    --output text
```

次の例は、指定した日付の範囲内に作成された、すべてのスナップショットの ID を表示します。

```
aws ec2 describe-snapshots \
```

```
    --filters Name=owner-id,Values=123456789012 \
    --query "Snapshots[?(StartTime>='2024-01-01') && (StartTime<='2024-12-31')].
[SnapshotId]" \
    --output text
```

例: タグに基づくフィルタリング

タグに従ってリソースのリストをフィルタリングする方法の例については「タグによる Amazon EC2 リソースのフィルタリング」を参照してください。

PowerShell

Example例: 単一のフィルタを指定する

Get-EC2Instance を使用して Amazon EC2 インスタンスを一覧表示できます。フィルタを使用しないと、レスポンスにはすべてのリソースに関する情報が含まれます。次のパラメータを使用すると、出力に実行中のインスタンスのみを含めることができます。

```
-Filter @{Name="instance-state-name"; Values="running"}
```

次の例は、実行中のインスタンスのインスタンス ID のみを一覧表示します。

```
(Get-EC2Instance -Filter @{Name="instance-state-name"; Values="stopped"}).Instances
 | Select InstanceId
```

Example例: 複数のフィルタまたはフィルタ値の指定

複数のフィルタまたは複数のフィルタ値を指定する場合、リソースはすべてのフィルタに一致して結果に含める必要があります。

次のオプションを使用すると、タイプが m5.large または m5d.large のいずれかであるすべてのインスタンスを一覧表示できます。

```
-Filter @{Name="instance-type"; Values="m5.large", "m5d.large"}
```

次のオプションを使用して、タイプが t2.micro であるすべての停止したインスタンスを一覧表示できます。

```
-Filter @{Name="instance-state-name"; Values="stopped"}, @{Name="instance-type";
 Values="t2.micro"}
```

AWS Global View を使用して複数のリージョンにわたるリソースを表示する

AWS Global View では、単一の AWS リージョンまたは単一のコンソールで同時に複数のリージョンでリソースを表示して検索できます。詳細については、「AWS Global View を使用して複数のリージョンにわたるリソースを表示する」を参照してください。

AWS Global View を使用して複数のリージョンにわたるリソースを表示する

AWS Global View では、単一の AWS リージョンまたは単一のコンソールにおける複数のリージョンで一部のリソースを表示できます。AWSGlobal View では、グローバル検索機能を使用して、特定のリソースまたは特定のリソースタイプを複数のリージョンにまたがって同時に検索できます。

AWS Global View では、いかなる方法でもリソースを変更することはできません。

サポートされているリソース

AWS Global View を使用すると、AWS アカウント が有効なすべてのリージョンの以下のリソースのグローバルサマリーを表示できます。

- 「Auto Scaling グループ」
- アベイラビリティーゾーン
- キャパシティ予約とキャパシティブロック
- DB クラスター
- DB インスタンス
- DHCP オプションセット
- Amazon Elastic Container Service (Amazon ECS) クラスター
- ECS サービス
- ECS タスク
- Egress-Only インターネットゲートウェイ
- Elastic IP
- エンドポイントサービス
- インスタンス

- インターネットゲートウェイ
- マネージドプレフィックスリスト
- NAT ゲートウェイ
- ネットワーク ACL
- ネットワークインターフェイス
- Outposts
- ルートテーブル
- S3 バケット
- セキュリティグループ
- サブネット
- ボリューム
- VPC
- VPC エンドポイント
- VPC ピアリング接続

必要なアクセス許可

AWS Global View を使用するユーザーには、次の許可が必要です。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "autoscaling:DescribeAutoScalingGroups",
      "ec2:DescribeRegions",
      "ec2:DescribeCapacityReservations",
      "ec2:DescribeDhcpOptions",
      "ec2:DescribeEgressOnlyInternetGateways",
      "ec2:DescribeAddresses",
      "ec2:DescribeVpcEndpointServices",
      "ec2:DescribeInstances",
      "ec2:DescribeInternetGateways",
```

```
        "ec2:DescribePrefixLists",
        "ec2:DescribeNatGateways",
        "ec2:DescribeNetworkAcls",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DescribeRouteTables",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSubnets",
        "ec2:DescribeVolumes",
        "ec2:DescribeVpcs",
        "ec2:DescribeVpcEndpoints",
        "ec2:DescribeVpcPeeringConnections",
        "ec2:DescribeAvailabilityZones",
        "ec2:DescribeVpcEndpointServiceConfigurations",
        "ec2:DescribeManagedPrefixLists",
        "ecs:ListClusters",
        "ecs:DescribeServices",
        "ecs:ListTasks",
        "outposts:ListOutposts",
        "rds:DescribeDBInstances",
        "rds:DescribeDBClusters",
        "s3:ListAllMyBuckets"
    ],
    "Resource": "*"
    }]
}
```

AWS Global View を使用するには

AWS Global View コンソールにサインインします。

> **Important**
>
> Firefox のプライベートウィンドウを使用して AWS Global View にアクセスすることはできません。

このコンソールは以下のもので構成されています。

- [リージョンエクスプローラー] – このページには、次のセクションがあります。
  - [概要] – すべてのリージョンにわたるリソースの高レベルの概要を提供します。

[Show all resource summary] を展開すると、AWS アカウント が有効になっているリージョンの数が表示されます。残りのフィールドは、それらのリージョンに現在あるリソースの数を示します。いずれかのリンクを選択すると、すべてのリージョンでそのタイプのリソースを表示します。例えば、インスタンスラベルの下のリンクが10リージョンで29の場合は、現在リージョン29間のインスタンス10が存在しているということを示しています。リンクを選択して、29すべてのインスタンスのリストを表示します。

- [リージョンエクスプローラー] – すべての AWS リージョン (アカウントが有効になっていないリージョンを含む) が一覧表示され、各リージョンのリソースタイプの合計数を確認できます。

  リージョン名を選択すると、特定のリージョンのすべてのタイプのリソースを表示します。例えば、アフリカ (ケープタウン) af-south-1 を選択すると、そのリージョン内の VPC、サブネット、インスタンス、セキュリティグループ、ボリューム、および Auto Scaling グループをすべて表示します。または、リージョンを選択し、選択したリージョンのリソースの表示を選びます。

  特定のリージョン内の特定のリソースタイプの値を選択して、そのリージョン内のタイプのリソースのみを表示します。例えば、アフリカ (ケープタウン) af-south-1のInstancesの値を選択して、そのリージョンのインスタンスのみを表示します。

- グローバル検索 – このページでは、1 つのリージョンまたは複数のリージョンにわたって特定のリソースまたは特定のリソースタイプを検索できます。また、特定のリソースの詳細を表示できます。

  リソースを検索するには、グリッドの前のフィールドに検索条件を入力します。リージョン、リソースタイプ、およびリソースに割り当てられたタグにより検索できます。

  特定のリソースの詳細を表示するには、グリッドでそのリソースを選択します。またリソースのリソース ID を選択して、それぞれのコンソールで開くこともできます。例えば、インスタンス ID を選択して Amazon EC2 コンソールでインスタンスを開く、またはサブネット ID を選択して Amazon VPC コンソールでサブネットを開きます。

- リージョンおよびゾーン – このページでは、使用可能なすべてのリージョン、アベイラビリティーゾーン、ローカルゾーン、および Wavelength Zone を表示および管理できます。

  [リージョン] タブから、すべての AWS リージョンを表示できます。[ステータス] 列には、AWS アカウントで有効になっているリージョンが表示されます。このページでは、リージョンを選択して以下の操作を行うことができます。

  - リージョンの詳細 (リージョンコード、地域、各タイプのゾーンの数など) を表示する。

アベイラビリティーゾーン、ローカルゾーン、Wavelength Zone のリストと、リージョン内のリソースのリストを表示することもできます。

- リージョンを有効化または無効化する。

各ゾーンタブから、そのゾーンタイプのリストを表示できます。[ローカルゾーン] タブから、ローカルゾーンにオプトインできます。

> **Tip**
>
> 特定のリージョンまたはリソースタイプのみを使用する場合は、AWS Global View をカスタマイズして、それらのリージョンとリソースタイプのみを表示できます。表示されるリージョンとリソースタイプをカスタマイズするには、ナビゲーションパネルで [設定] を選択してから、[リソース] タブと [リージョン] タブで、[AWS グローバルビュー] に表示しないリージョンとリソースタイプを選択します。

Amazon EC2 リソースのタグ付け

インスタンスやイメージなどの Amazon EC2 リソースを管理しやすくするために、独自のメタデータをタグとして各リソースに割り当てることができます。タグを使用すると、AWS リソースを用途、所有者、環境などのさまざまな方法で分類できます。これは同じタイプのリソースが多数ある場合に役立ちます。割り当てたタグに基づいて、特定のリソースをすばやく識別できます。ここではタグとその作成方法について説明します。

> **Warning**
>
> タグのキーと値は多くの異なる API コールから返されます。`DescribeTags` へのアクセスを拒否しても、他の API から返されるタグへのアクセスは自動的に拒否されません。ベストプラクティスとして、機密データをタグに含めないようお勧めします。

内容

- タグの基本
- リソースのタグ付け
- タグの制限

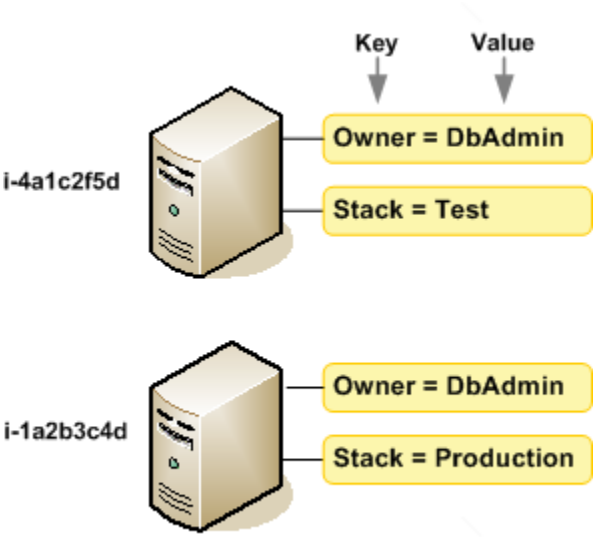
- タグとアクセス管理
- 請求用のリソースのタグ付け
- Amazon EC2 リソース作成時にタグ付けするアクセス許可の付与
- Amazon EC2 リソースのタグの追加と削除
- タグによる Amazon EC2 リソースのフィルタリング
- インスタンスメタデータを使用して EC2 インスタンスのタグを表示する

タグの基本

タグとはAWS リソースに割り当てるラベルです。タグはそれぞれ、1 つのキーとオプションの 1 つの値で構成されており、どちらもお客様側が定義します。

タグを使用すると、AWS リソースを用途、所有者、環境などのさまざまな方法で分類できます。例えば、アカウントの各インスタンスの所有者とスタックレベルを追跡しやすくするため、Amazon EC2 インスタンスに対して一連のタグを定義できます。

次の図はタグの機能を示しています。図の中ではインスタンスのそれぞれに 2 つのタグを割り当てています。1 つは `Owner` のキーを使用、もう 1 つは `Stack` キーを使用します。各タグには値も関連付けられています。



各リソースタイプのニーズを満たす一連のタグキーを考案することをお勧めします。一貫性のある一連のタグキーを使用することで、リソースの管理が容易になります。追加したタグに基づいてリソースを検索およびフィルタリングできます。効果的なリソースのタグ付け戦略を実装する方法の詳細についてはAWS ホワイトペーパーの「タグ付けのベストプラクティス」を参照してください。

タグにはAmazon EC2 に関連する意味はなく、完全に文字列として解釈されます。また、タグは自動的にリソースに割り当てられます。タグのキーと値は編集でき、タグはリソースからいつでも削除できます。タグの値を空の文字列に設定することはできますが、タグの値を null に設定することはできません。特定のリソースについて既存のタグと同じキーを持つタグを追加した場合、以前の値は新しい値によって上書きされます。リソースを削除すると、リソースのタグも削除されます。

> **Note**
>
> リソースを削除した後も、そのタグがコンソール、API、および CLI の出力にしばらく表示されたままになる場合があります。これらのタグは徐々にリソースから切り離され、完全に削除されます。

リソースのタグ付け

Amazon EC2 コンソールを使用しているときは関連するリソース画面で [タグ] タブを使用してリソースにタグを適用するか、または AWS Resource Groups コンソールで [タグエディタ] を使用できます。一部のリソースの画面ではリソースの作成時にリソースのタグを指定できます。例えば、`Name` のキーと指定した値をタグ付けします。ほとんどの場合、リソースの作成後すぐに (リソースの作成時ではなく) コンソールによりタグが適用されます。コンソールは `Name` タグに従ってリソースを整理する場合がありますが、このタグには Amazon EC2 サービスに対する意味論的意味はありません。

Amazon EC2 API、AWS CLI、または AWS SDK を使用している場合、`CreateTags` EC2 API アクションを使用してタグを既存のリソースに適用できます。さらに、リソース作成アクションによってはリソースの作成時にリソースのタグを指定できます。リソースの作成時にタグを適用できない場合はリソース作成プロセスがロールバックされます。これにより、リソースがタグ付きで作成されるか、まったく作成されないようになるため、タグ付けされていないリソースが存在することがなくなります。作成時にリソースにタグ付けすることで、リソース作成後にカスタムタグ付けスクリプティングを実行する必要がなくなります。作成時にユーザーがリソースにタグ付けできるようにする方法については「Amazon EC2 リソース作成時にタグ付けするアクセス許可の付与」を参照してください。

IAM ポリシーでタグベースのリソースレベルアクセス許可を、作成時のタグ付けをサポートする Amazon EC2 API アクションに適用し、作成時にリソースにタグ付けできるユーザーとグループを細かく制御できます。リソースは作成時から適切に保護されます。タグはリソースに即座に適用されるため、リソースの使用を制御するタグベースのリソースレベルアクセス権限がただちに有効になり

ます。リソースはより正確に追跡および報告されます。新しいリソースにタグ付けの使用を適用し、リソースで設定されるタグキーと値を制御できます。

さらに、リソースレベルのアクセス許可を IAM ポリシーの `CreateTags` および `DeleteTags` Amazon EC2 API アクションに適用し、既存のリソースで設定されるタグキーと値を制御することもできます。詳細については「例: リソースのタグ付け」を参照してください。

請求用リソースへのタグ付けの詳細についてはAWS Billing ユーザーガイドの「コスト配分タグの使用」を参照してください。

タグの制限

タグには以下のような基本制限があります。

- リソースあたりのタグの最大数 - 50 件
- タグキーはリソースごとにそれぞれ一意である必要があります。また、各タグキーに設定できる値は 1 つのみです。
- キーの最大長 - UTF-8 の 128 Unicode 文字
- 値の最大長 - UTF-8 の 256 Unicode 文字
- 使用できる文字
  - EC2 ではタグ内に任意の文字を使用できますが、他の AWS のサービスでは制限があります。すべての AWS のサービスで使用できる文字はUTF-8 で表現できる英字 (a-z、A-Z)、数字 (0-9)、スペース、および `+ - = . _ : / @` です。
  - インスタンスメタデータでインスタンスタグを有効にすると、インスタンスタグキーは文字 (a-z、A-Z)、数字 (0-9)、および次の文字のみを使用できます: `+ - = . , _ : @`。インスタンスタグキーはスペースまたは `/` を含めることはできず、`.` (1 つのピリオド)、`..` (2 つのピリオド)、または `_index` のみを含めることはできません。詳細については「インスタンスメタデータを使用して EC2 インスタンスのタグを表示する」を参照してください。
- タグのキーと値は大文字と小文字が区別されます。
- `aws:` プレフィックスは AWS 用に限定されています。タグにこのプレフィックスが付いたタグキーがある場合、タグのキーまたは値を編集、削除することはできません。`aws:` プレフィックスを持つタグはリソースあたりのタグ数の制限時には計算されません。

タグのみに基づいてリソースを終了、停止、終了することはできません。リソース識別子を指定する必要があります。例えば、`DeleteMe` というタグキーを使用してタグ付けしたスナップショッ

トを削除するにはDeleteSnapshots のようなスナップショットのリソース識別子を指定して snap-1234567890abcdef0 アクションを使用する必要があります。

パブリックリソースまたは共有リソースにタグを付けると、割り当てたタグはタグ付けを行った AWS アカウントだけが使用できます。他の AWS アカウントはそれらのタグにアクセスできません。共有リソースへのタグベースのアクセス制御ではリソースへのアクセスを制御するために、各 AWS アカウントに独自のタグセットを割り当てる必要があります。

タグとアクセス管理

AWS Identity and Access Management (IAM) を使用している場合はタグを作成、編集、削除する許可を持つ AWS アカウントのユーザーを制御できます。詳細については「Amazon EC2 リソース作成時にタグ付けするアクセス許可の付与」を参照してください。

リソースタグを使用して、属性ベースの制御 (ABAC) を実装することもできます。リソースのタグに基づいてオペレーションを許可する IAM ポリシーを作成できます。詳細については「属性ベースのアクセスを使用するアクセスの制御」を参照してください。

請求用のリソースのタグ付け

タグを使用して AWS 請求書を整理し、自分のコスト構造を反映できます。そのためにはサインアップして、タグキー値が含まれた AWS アカウントの請求書を取得する必要があります。タグによるコスト配分レポートの設定の詳細については『AWS Billing ユーザーガイド』の「コスト配分月次レポート」を参照してください。リソースを組み合わせたコストを確認するには同じタグキー値を持つリソースに基づいて、請求情報を整理します。例えば、複数のリソースに特定のアプリケーション名のタグを付け、請求情報を整理することで、複数のサービスを利用しているアプリケーションの合計コストを確認することができます。詳細についてはAWS Billing ユーザーガイド の「コスト配分タグの使用」を参照してください。

> **Note**
>
> レポートを有効にすると、約 24 時間後に、今月のデータを表示できるようになります。

コスト割り当てタグはどのリソースがコストに貢献しているかを示すことができますが、リソースを削除または非アクティブ化にしてもコストは必ずしも削減されるわけではありません。例えば、元のデータを含むスナップショットが削除された場合でも、別のスナップショットによって参照されるスナップショットデータは維持されます。詳細についてはAWS Billing ユーザーガイドの「Amazon Elastic Block Store のボリュームとスナップショット」を参照してください。

> **Note**
>
> タグされている Elastic IP アドレスはコスト配分レポートには表示されません。

Amazon EC2 リソース作成時にタグ付けするアクセス許可の付与

一部のリソース作成 Amazon EC2 API アクションではリソースの作成時にタグを指定できます。リソースタグを使用して、属性ベースの制御 (ABAC) を実装できます。詳細については[リソースのタグ付け]および[属性ベースのアクセスを使用するアクセスの制御]を参照してください。

ユーザーがリソースの作成時にタグを付けるにはリソースを作成するアクション (`ec2:RunInstances` や `ec2:CreateVolume` など) を使用するためのアクセス許可が必要です。タグがリソース作成アクションで指定されている場合、Amazon は `ec2:CreateTags` アクションで追加の認可を実行してユーザーがタグを作成するアクセス許可を持っているかどうかを確認します。そのため、ユーザーには`ec2:CreateTags` アクションを使用する明示的なアクセス権限が必要です。

`ec2:CreateTags` アクションの IAM ポリシー定義で、`Condition` 要素と `ec2:CreateAction` 条件キーを使用して、リソースを作成するアクションにタグ付けのアクセス許可を付与します。

次のポリシー例ではインスタンスを起動し、起動時にインスタンスとボリュームにタグを適用することをユーザーに許可します。ユーザーには既存のリソースへのタグ付けが許可されません (`ec2:CreateTags` アクションを直接呼び出すことはできません)。

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
         "ec2:RunInstances"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
         "ec2:CreateTags"
      ],
      "Resource": "arn:aws:ec2:us-east-1:111122223333:*/*",
      "Condition": {
```

```
          "StringEquals": {
              "ec2:CreateAction" : "RunInstances"
           }
        }
    }
  ]
}
```

同様に、次のポリシーではユーザーがボリュームを作成し、ボリューム作成時にボリュームにタグを適用することができます。ユーザーには既存のリソースへのタグ付けが許可されません (ec2:CreateTags アクションを直接呼び出すことはできません)。

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
         "ec2:CreateVolume"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
         "ec2:CreateTags"
      ],
      "Resource": "arn:aws:ec2:us-east-1:111122223333:*/*",
      "Condition": {
         "StringEquals": {
             "ec2:CreateAction" : "CreateVolume"
          }
       }
    }
  ]
}
```

ec2:CreateTags アクションはタグがリソース作成アクション時に適用された場合のみ評価されます。したがって、リクエストでタグが指定されていない場合、リソースを作成するアクセス許可を持っているユーザー (タグ付け条件がないと仮定) にはec2:CreateTags アクションを実行するアクセス許可は必要ありません。ただし、ユーザーがタグを使用してリソースを作成しようとした場合、ユーザーが ec2:CreateTags アクションを使用するアクセス権限を持っていない場合はリクエストに失敗します。

ec2:CreateTags アクションはタグが起動テンプレートに指定されている場合にも評価されます。ポリシーの例については起動テンプレートのタグを参照してください。

特定のタグへのアクセスの制御

IAM ポリシーの Condition 要素で追加の条件を使用して、リソースに適用できるタグキーとタグ値を制御できます。

次の条件キーは前のセクションの例で使用できます。

- aws:RequestTag: 特定のタグキーまたはタグキーと値がリクエストに存在している必要があることを指定する場合に使用します。リクエストでは他のタグも指定できます。
  - StringEquals 条件演算子とともに使用して、特定のタグキーと値の組み合わせを適用します。例えば、タグ cost-center=cc123: を適用します。

    ```
    "StringEquals": { "aws:RequestTag/cost-center": "cc123" }
    ```

  - StringLike 条件演算子とともに使用して、リクエストで特定のタグキーを適用します。例えば、タグキー purpose を適用します。

    ```
    "StringLike": { "aws:RequestTag/purpose": "*" }
    ```

- aws:TagKeys: リクエストで使用されるタグキーを適用する場合に使用します。
  - リクエストにタグが指定されている場合はForAllValues 修飾子を使用して特定のタグキーのみを適用します (リクエストにタグが指定されている場合、特定のタグキーのみが許可されます。他のタグは許可されません)。例えば、タグキー environment または cost-center が適用されます:

    ```
    "ForAllValues:StringEquals": { "aws:TagKeys": ["environment","cost-center"] }
    ```

  - ForAnyValue 修飾子とともに使用して、指定されたタグキーの少なくとも 1 つがリクエストに存在することを要求します。例えば、タグキー environment または webserver のうち少なくとも 1 つがリクエストに存在している必要があります。

    ```
    "ForAnyValue:StringEquals": { "aws:TagKeys": ["environment","webserver"] }
    ```

これらの条件キータグ付けをサポートするリソース作成アクションと、ec2:CreateTags および ec2:DeleteTags アクションに適用できます。Amazon EC2 API アクションがタグ付けをサポート

しているかどうかについてはAmazon EC2 のアクション、リソース、および条件キーを参照してください

リソースの作成時にタグを指定するようにユーザーに強制するにはリソース作成アクションで aws:RequestTag 修飾子とともに aws:TagKeys 条件キーまたは ForAnyValue 条件キーを使用する必要があります。ユーザーがリソース作成アクションのタグを指定しない場合、ec2:CreateTags アクションは評価されません。

条件においては条件キーでは大文字と小文字が区別されず、条件値では大文字と小文字が区別されます。したがって、タグキーの大文字と小文字を区別するには条件の値としてタグキーが指定される aws:TagKeys 条件キーを使用します。

IAM ポリシーの例はAmazon EC2 API へのアクセスを制御するポリシーの例を参照してください。詳細については「IAM ユーザーガイド」の「複数のキーまたは値を含む IAM 条件」参照してください。

Amazon EC2 リソースのタグの追加と削除

Amazon EC2 インスタンスなどの Amazon EC2 リソースを作成するときはリソースに追加するタグを指定できます。Amazon EC2 コンソールを使用して、特定の Amazon EC2 リソースのタグを表示することもできます。また、既存の Amazon EC2 リソースからタグを追加または削除することもできます。

AWS Resource Groups コンソールのタグエディタを使用して、すべてのリージョンのすべての AWS リソースでタグを表示、追加、または削除できます。複数のタイプのリソースに対して一度にタグを適用または削除できます。詳細については「AWS リソースのタグ付けのユーザーガイド」を参照してください。

タスク

- コンソールを使用してタグを追加する
- AWS CLI を使用してタグを追加する
- PowerShell を使用してタグを追加する
- CloudFormation を使用してタグを追加する

コンソールを使用してタグを追加する

タグを、リソースのページから既存のリソースに直接追加することができます。

既存のリソースにタグを追加するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションバーから、リソースがあるリージョンを選択してください。
3. ナビゲーションペインで、リソースタイプを選択します ([Instances] など)。
4. リストからリソースを選択してください。
5. [タグ] タブから、[タグを管理] を選択してください。
6. [新しいタグを追加] を選択し、タグのキーと値を入力します。
7. [保存] を選択します。

AWS CLI を使用してタグを追加する

リソースの作成時にタグを追加したり、既存のリソースにタグを追加したりできます。

リソースの作成時にタグを追加するには

リソースの作成時に、リソースにタグを付けるには、`-tag-specifications` オプションを使用します。タグ仕様には、タグを付けるリソースのタイプ、タグキー、タグの値が必要です。次の例では、タグを作成して、それをタグ仕様に追加します。

```
--tag-specifications 'ResourceType=instance,Tags=[{Key=stack,Value=production}]'
```

既存のリソースにタグを追加するには

次の例では、create-tags コマンドを使用して既存のリソースにタグを追加する方法を紹介します。

Example例: リソースにタグを追加する

次のコマンドではタグ (**Stack=production**) を指定されたイメージに追加するか、タグキーが stack の AMI 用に既存のタグを上書きします。コマンドが成功した場合、出力は返りません。

```
aws ec2 create-tags \
    --resources ami-0abcdef1234567890 \
    --tags Key=stack,Value=production
```

Example例: タグを複数のリソースに追加する

この例では2 つのタグを AMI とインスタンス用に追加 (または上書き) します。一方のタグにはキー (webserver) のみ含まれており、値は設定されていません (値を空の文字列に設定)。もう 1 つのタグはキー (stack) と値 (**Production**) で構成されます。コマンドが成功した場合、出力は返りません。

```
aws ec2 create-tags \
    --resources ami-0abcdef1234567890 i-1234567890abcdef0 \
    --tags Key=webserver,Value=  Key=stack,Value=Production
```

Example例: 特殊文字のタグを追加する

この例ではタグ ([Group]=test) をインスタンスに追加します。角括弧 ([および]) は特殊文字であり、エスケープする必要があります。

Linux または OS X を使用している場合、特殊文字を エスケープするには特殊文字を含む要素を二重引用符 (") で囲んでから、キーと値の構造全体を一重引用符 (') で囲みます。

```
aws ec2 create-tags \
    --resources i-1234567890abcdef0 \
    --tags 'Key="[Group]",Value=test'
```

Windows を使用している場合、特殊文字を エスケープするには特殊文字を含む要素を二重引用符 (") で囲み、各二重引用符の前にバックスラッシュ (****) を付けます。

```
aws ec2 create-tags ^
    --resources i-1234567890abcdef0 ^
    --tags Key=\"[Group]\",Value=test
```

Windows PowerShell を使用している場合、特殊文字を エスケープするには次のように特殊文字を含む値を二重引用符 (**"**) で囲み、各二重引用符の前にバックスラッシュ (****) を付けてから、キーと値の構造全体を一重引用符 (**'**) で囲みます。

```
aws ec2 create-tags `
    --resources i-1234567890abcdef0 `
    --tags 'Key=\"[Group]\",Value=test'
```

PowerShell を使用してタグを追加する

リソースの作成時にタグを追加したり、既存のリソースにタグを追加したりできます。

リソースの作成時にタグを追加するには

リソースの作成時に、リソースにタグを付けるには、-TagSpecification パラメータを使用します。タグ仕様には、タグを付けるリソースのタイプ、タグキー、タグの値が必要です。次の例では、タグを作成して、それをタグ仕様に追加します。

```
$tag = @{Key="stack"; Value="production"}
$tagspec = new-object Amazon.EC2.Model.TagSpecification
$tagspec.ResourceType = "instance"
$tagspec.Tags.Add($tag)
```

次の例では、このタグを -TagSpecification パラメータの中で指定します。

```
-TagSpecification $tagspec
```

既存のリソースにタグを追加するには

New-EC2Tag コマンドレットを使用します。リソース、タグキー、タグ値を指定する必要があります。

```
New-EC2Tag `
    -Resource i-1234567890abcdef0 `
    -Tag @{Key="purpose"; Value="production"}
```

CloudFormation を使用してタグを追加する

Amazon EC2 リソースタイプではTags または TagSpecifications プロパティを使用してタグを指定します。

次の例では**Stack=Production** プロパティを使用して AWS::EC2::Instance にタグ Tags を追加します。

Example例: YAML のタグ

```
Tags:
  - Key: "Stack"
    Value: "Production"
```

Example例: JSON のタグ

```
"Tags": [
```

```
    {
        "Key": "Stack",
        "Value": "Production"
    }
]
```

次の例では**Stack=Production** プロパティを使用して AWS::EC2::LaunchTemplate LaunchTemplateData にタグ TagSpecifications を追加します。

Example例: YAML のタグ仕様

```
TagSpecifications:
  - ResourceType: "instance"
    Tags:
    - Key: "Stack"
      Value: "Production"
```

Example例: JSON のタグ仕様

```
"TagSpecifications": [
    {
        "ResourceType": "instance",
        "Tags": [
            {
                "Key": "Stack",
                "Value": "Production"
            }
        ]
    }
]
```

タグによる Amazon EC2 リソースのフィルタリング

タグを追加すると、タグキーとタグ値に基づいて Amazon EC2 リソースをフィルタリングできます。

Console

タグでリソースをフィルタリングするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

2. ナビゲーションペインで、リソースタイプを選択します ([Instances] など)。
3. 検索フィールドを選択してください。
4. リストの [タグ] で、タグキーを選択してください。
5. リストから対応するタグ値を選択してください。
6. 完了したら、フィルターを削除します。

Amazon EC2 コンソールでのフィルターの使用の詳細については「Amazon EC2 リソースの検索」を参照してください。

AWS CLI

指定されたタグキーを使用して単一タイプのリソースを記述するには

次のフィルターを describe コマンドに追加して、タグの値に関係なく、Stack タグを使用してそのタイプのリソースを記述します。

```
--filters Name=tag-key,Values=Stack
```

指定されたタグを使用して単一タイプのリソースを記述するには

次のフィルターを describe コマンドに追加して、タグ Stack=production を使用してそのタイプのリソースを記述します。

```
--filters Name=tag:Stack,Values=production
```

指定されたタグ値を使用して単一タイプのリソースを記述するには

次のフィルターを describe コマンドに追加して、タグキーに関係なく、値が production であるタグを使用してそのタイプのリソースを記述します。

```
--filters Name=tag-value,Values=production
```

指定されたタグを使用してすべての EC2 リソースを記述するには

次のフィルターを describe-tags コマンドに追加して、タグ Stack=test を使用してすべての EC2 リソースを記述します。

```
--filters Name=key,Values=Stack Name=value,Values=test
```

PowerShell

タグキーで単一タイプのリソースをフィルタリングするには

次のフィルターを Get コマンドレットに追加して、タグの値に関係なく、Stack タグを使用してそのタイプのリソースを記述します。

```
-Filter @{Name="tag-key"; Values="Stack"}
```

タグで単一タイプのリソースをフィルタリングするには

次のフィルターを Get コマンドレットに追加して、タグ Stack=production を使用してそのタイプのリソースを記述します。

```
-Filter @{Name="tag:Stack"; Values="production"}
```

タグ値で単一タイプのリソースをフィルタリングするには

次のフィルターを Get コマンドレットに追加して、タグキーの値に関係なく、値が production であるタグを使用してそのタイプのリソースを記述します。

```
-Filter @{Name="tag-value"; Values="production"}
```

タグですべての EC2 リソースをフィルタリングするには

次のフィルターを Get-EC2Tag コマンドレットに追加し、タグ Stack=test を使用してすべての EC2 リソースを記述します。

```
-Filter @{Name="tag:Stack"; Values="test"}
```

インスタンスメタデータを使用して EC2 インスタンスのタグを表示する

インスタンスのメタデータからインスタンスのタグにアクセスできます。インスタンスメタデータからタグにアクセスすると、DescribeInstances または DescribeTags API コールを使用してタグ情報を取得する必要がなくなります。これにより、1 秒あたりの API トランザクションが削減され、制御するインスタンスの数に応じてタグ取得をスケールできるようになります。さらに、インスタンスで実行されているローカルプロセスはインスタンスのメタデータからインスタンスのタグ情報を直接表示できます。

デフォルトではインスタンスメタデータからタグは使用できません。アクセスを明示的に許可する必要があります。インスタンスの起動時、または実行中または停止したインスタンスでの起動後にアクセスを許可できます。起動テンプレートでこれを指定することで、タグへのアクセスを許可することもできます。テンプレートを使用してインスタンスが起動されると、インスタンスメタデータ内のタグへのアクセスが許可されます。

インスタンスタグを追加または削除すると、インスタンスの実行中にインスタンスのメタデータが更新されます。インスタンスを停止して起動したりする必要はありません。

タスク

- インスタンスメタデータ内のタグへのアクセスを有効にする
- インスタンスメタデータからタグを取得する
- インスタンスメタデータ内のタグへのアクセスを無効にする

インスタンスメタデータ内のタグへのアクセスを有効にする

デフォルトではインスタンスメタデータ内のインスタンスタグへのアクセス権はありません。各インスタンスで、明示的にアクセスを有効にする必要があります。

> **Note**
>
> インスタンスメタデータのタグへのアクセスを許可する場合、インスタンスタグキーが具体的な制限の対象になります。コンプライアンスに違反すると、新しいインスタンスの起動が失敗したり、既存のインスタンスのエラーが発生します。制限は次のとおりです。
>
> - 英字 (a-z、A-Z)、数字 (0-9)、および + - = . , _ : @ の各文字のみを含めることができます。
> - スペースと / を含めることはできません。
> - . (1 つのピリオド)、.. (2 つのピリオド)、または _index だけで構成することはできません。
>
> 詳細については「タグの制限」を参照してください。

Console

インスタンスの起動時に、インスタンスメタデータ内のタグへのアクセスを許可するには

1. インスタンスを起動するための手順に従います。
2. [詳細] を展開し、[メタデータのタグを許可する] で、[有効にする] を選択してください。
3. [合計mary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択してください。詳細については「コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する」を参照してください。

インスタンスの起動後に、インスタンスメタデータ内のタグへのアクセスを許可するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択し、[アクション]、[インスタンス設定]、[インスタンスメタデータ内のタグを許可する] の順に選択してください。
4. インスタンスメタデータのタグへのアクセスを許可するには[許可] チェックボックスをオンにします。
5. [保存] を選択します。

AWS CLI

インスタンスの起動時に、インスタンスメタデータ内のタグへのアクセスを許可するには

run-instances コマンドを使用して、次の `--metadata-options` オプションを追加します。

```
--metadata-options "InstanceMetadataTags=enabled"
```

インスタンスの起動後に、インスタンスメタデータ内のタグへのアクセスを許可するには

次の modify-instance-metadata-options コマンドを使用します。

```
aws ec2 modify-instance-metadata-options \
    --instance-id i-1234567890abcdef0 \
    --instance-metadata-tags enabled
```

インスタンスメタデータ内のタグへのアクセスが許可されていることを確認するには

describe-instances コマンドを使用して InstanceMetadataTags の値を確認します。

```
aws ec2 describe-instances \
    --instance-ids i-1234567890abcdef0 \
    --query "Reservations[*].Instances[].MetadataOptions[].InstanceMetadataTags"
```

以下は出力の例です。この値は enabled または disabled のどちらかです。

```
[
    "enabled"
]
```

PowerShell

インスタンスの起動時に、インスタンスメタデータ内のタグへのアクセスを許可するには

New-EC2Instance コマンドレットを使用して、次の -MetadataOptions_InstanceMetadataTags パラメータを追加します。

```
-MetadataOptions_InstanceMetadataTags enabled
```

インスタンスの起動後に、インスタンスメタデータ内のタグへのアクセスを許可するには

Edit-EC2InstanceMetadataOption コマンドレットを使用します。

```
Edit-EC2InstanceMetadataOption `
    -InstanceId i-1234567890abcdef0 `
    -InstanceMetadataTags enabled
```

インスタンスメタデータ内のタグへのアクセスが許可されていることを確認するには

Get-EC2Instance コマンドレットを使用して、InstanceMetadataTags の値をチェックします。

```
(Get-EC2Instance `
    -
InstanceId i-1234567890abcdef0).Instances.MetadataOptions.InstanceMetadataTags.Value
```

以下は出力の例です。この値は enabled または disabled のどちらかです。

```
enabled
```

インスタンスメタデータからタグを取得する

インスタンスメタデータでインスタンスタグへのアクセスを許可すると、インスタンスのメタデータから tags/instance カテゴリにアクセスできます。詳細については「EC2 インスタンスのインスタンスメタデータにアクセスする」を参照してください。

IMDSv2

Linux

次のコマンドを Linux インスタンスから実行し、インスタンスのすべてのタグキーを一覧表示します。

```
TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
    && curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/tags/instance
```

この例では、前回の例で取得したキーの値を取得します。IMDSv2 リクエストは、前の例のコマンドで作成された保管されているトークンを使用します。このトークンは有効期限が切れていない必要があります。

```
curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/tags/instance/tag-key
```

Windows

次のコマンドレットを Windows インスタンスから実行し、インスタンスのすべてのタグキーを一覧表示します。

```
$token = Invoke-RestMethod `
    -Headers @{"X-aws-ec2-metadata-token-ttl-seconds" = "21600"} `
    -Method PUT -Uri http://169.254.169.254/latest/api/token
```

```
Invoke-RestMethod `
    -Headers @{"X-aws-ec2-metadata-token" = $token} `
```

```
    -Method GET -Uri http://169.254.169.254/latest/meta-data/tags/instance
```

この例では、前回の例で取得したキーの値を取得します。IMDSv2 リクエストは、前の例のコマンドで作成された保管されているトークンを使用します。このトークンは有効期限が切れていない必要があります。

```
Invoke-RestMethod `
    -Headers @{"X-aws-ec2-metadata-token" = $token} `
    -Method GET -Uri http://169.254.169.254/latest/meta-data/tags/instance/tag-key
```

IMDSv1

Linux

次のコマンドを Linux インスタンスから実行し、インスタンスのすべてのタグキーを一覧表示します。

```
curl http://169.254.169.254/latest/meta-data/tags/instance
```

この例では、前回の例で取得したキーの値を取得します。

```
curl http://169.254.169.254/latest/meta-data/tags/instance/tag-key
```

Windows

次のコマンドレットを Windows インスタンスから実行し、インスタンスのすべてのタグキーを一覧表示します。

```
Invoke-RestMethod -Uri http://169.254.169.254/latest/meta-data/tags/instance
```

この例では、前回の例で取得したキーの値を取得します。

```
Invoke-RestMethod -Uri http://169.254.169.254/latest/meta-data/tags/instance/tag-key
```

インスタンスメタデータ内のタグへのアクセスを無効にする

インスタンスメタデータ内の、インスタンスタグへのアクセスを無効にすることができます。インスタンスメタデータ内のインスタンスタグへのアクセスは、デフォルトで無効になっているため、起動時にこれを無効にする必要はありません。

Console

インスタンスメタデータ内のタグへのアクセスを無効にするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択し、[Actions] (アクション)、[Instance settings] (インスタンス設定)、[Allow tags in instance metadata] (インスタンスメタデータ内のタグを許可する) の順に選択してください。
4. インスタンスメタデータのタグへのアクセスを無効にするには[許可] チェックボックスをオフにします。
5. [保存] を選択します。

AWS CLI

インスタンスメタデータ内のタグへのアクセスを無効にするには

次の modify-instance-metadata-options コマンドを使用します。

```
aws ec2 modify-instance-metadata-options \
    --instance-id i-1234567890abcdef0 \
    --instance-metadata-tags disabled
```

PowerShell

インスタンスメタデータ内のタグへのアクセスを無効にするには

Edit-EC2InstanceMetadataOption コマンドレットを使用します。

```
Edit-EC2InstanceMetadataOption `
    -InstanceId i-1234567890abcdef0 `
    -InstanceMetadataTag disabled
```

Amazon EC2 の Service Quotas

AWS アカウントを作成するときに、リージョンごとに AWS リソースに対してデフォルトのクォータ (制限とも呼ばれます) が設定されます。リソースのクォータの制限を超える処理を実行しようと

すると、リクエストは失敗します。例えば、リージョン内のオンデマンドインスタンスにプロビジョニングできる Amazon EC2 vCPU の最大数があります。リージョンでインスタンスを起動しようとしたとき、このリクエストにより使用量がこのクォータを超える場合、リクエストは失敗します。この場合、リソースの使用量を減らすか、クォータの引き上げをリクエストすることができます。

Service Quotas コンソールは、AWS サービスのクォータを表示および管理したり、使用する多くのリソースのクォータの引き上げをリクエストしたりできる一元的な場所です。提供されるクォータとに関する情報を利用して、AWS インフラストラクチャを管理します。クォータの引き上げに対するリクエストは、クォータの引き上げが必要となる前に計画してください。

関連するクォータドキュメント

- Amazon EC2 エンドポイントとクォータ
- Amazon EC2 インスタンスタイプクォータ
- Amazon EBS のクォータ
- Amazon VPC クォータ

現在の制限を表示するには

の Service Quotas コンソールを使用して、各リージョンのクォータを表示できます。

Service Quotas コンソールを使用して現在のクォータを表示するには

1. https://console.aws.amazon.com/servicequotas/home/services/ec2/quotas/ で Service Quotasのコンソールを開きます。
2. 画面上部のナビゲーションバーで、リージョンを選択します。
3. [クォータの管理] ペインで、Amazon Elastic Compute Cloud (Amazon EC2) などのサービスを選択します。[クォータの表示] をクリックします。
4. リソースネームごとにリストを絞り込むには、フィルターフィールドを使用します。例えば、**On-Demand** と入力してオンデマンドインスタンスのクォータを確認してください。
5. 詳細情報を表示するには、クォータ名を選択してクォータの詳細ページを開きます。

引き上げのリクエスト

各リージョンに対して、クォータの増加をリクエストすることができます。

増加をリクエストするには、Service Quotas コンソールを使用してください。

1. https://console.aws.amazon.com/servicequotas/home/services/ec2/quotas/ で Service Quotasのコンソールを開きます。
2. 画面上部のナビゲーションバーで、リージョンを選択します。
3. リソースネームごとにリストを絞り込むには、フィルターフィールドを使用します。例えば、**`On-Demand`** と入力してオンデマンドインスタンスのクォータを確認してください。
4. クォータが調整可能な場合は、クォータを選択し、[クォータの引き上げをリクエスト] を選択します。
5. [クォータ値を変更] に、新しいクォータ値を入力します。
6. [リクエスト] を選択します。
7. コンソールで保留中または最近解決された要求を表示するには、ナビゲーションペインから [ダッシュボード] を選択します。保留中のリクエストの場合は、リクエストのステータスを選択してリクエストの受信をオープンします。リクエストの初期ステータスは [Pending] (保留中) です。ステータスが [要求されたクォータ] に変わると、サポート とケース番号が表示されます。リクエストのチケットを開くには、ケース番号を選択します。

AWS CLI や SDK を使用してクォータの増加をリクエストする方法などの詳細については、「Service Quotas ユーザーガイド」の「クォータ増加のリクエスト」を参照してください。

ポート 25 を使用した E メール送信の制限

デフォルトでは、Amazon EC2 はポート 25 経由のアウトバウンドトラフィックを、プライベート IPv4 アドレスにのみ許可します。ポート 25 経由のトラフィックは、パブリック IPv4 アドレスと IPv6 アドレスに対してはブロックされます。

この制限の解除をリクエストできます。詳細については、「Amazon EC2 インスタンスまたは Lambda 関数のポート 25 の制限を解除するにはどうすればよいですか?」を参照してください。

この制限は、ポート 25 経由で次の宛先に送信されるアウトバウンドトラフィックには適用されません。

- 発信元のネットワークインターフェイスが存在する VPC のプライマリ CIDR ブロック内の IP アドレス。
- RFC 1918、RFC 6598、および RFC 4193 で定義されている CIDR 内の IP アドレス。

Amazon EC2 リソースのモニタリング

モニタリングはAmazon EC2 インスタンスおよび AWS ソリューションの信頼性、可用性、およびパフォーマンスを維持するうえで重要な部分です。マルチポイント障害が発生した場合はその障害をより簡単にデバッグできるように、AWS ソリューションのすべての部分からモニタリングデータを収集する必要があります。

AWS はAmazon EC2 のモニタリングに使用できるさまざまなツールを提供します。Amazon EC2 および CloudWatch のコンソールダッシュボードにはAmazon EC2 環境の状態が一目でわかるビューが表示されます。さらに、以下が提供されています。

- システムステータスのチェック - インスタンスを使用する際に必要な AWS システムをモニタリングして、正常に実行されていることを確認します。これらのチェックでは修復には AWS の関与が必要なインスタンスの根本的な問題が検出されます。システムステータスチェックが失敗した場合、AWS によって問題が修正されるのを待つか、自分自身で (例えば、インスタンスを停止、再起動、終了、置換するなどによって) 問題を解決できます。システムステータスチェックの失敗の原因となる問題には次のようなものがあります。
  - ネットワーク接続の喪失
  - システム電源の喪失
  - 物理ホストのソフトウェアの問題
  - ネットワーク到達可能性に影響する、物理ホスト上のハードウェアの問題

  詳細についてはAmazon EC2 インスタンスのステータスチェックを参照してください。
- インスタンスステータスのチェック - 個々のインスタンスのソフトウェアとネットワークの設定をモニタリングします。これらのチェックではユーザーが関与して修復する必要のある問題が検出されます。インスタンスステータスチェックが失敗した場合、通常はお客様ご自身で (インスタンスの再起動、オペレーティングシステムの修正など) 問題を修復する必要があります。インスタンスステータスチェックの失敗の原因となる問題には次のようなものがあります。
  - 失敗したシステムステータスチェック
  - 誤って設定されたネットワークまたは起動設定
  - メモリの枯渇
  - 破損したファイルシステム
  - 互換性のないカーネル

  詳細についてはAmazon EC2 インスタンスのステータスチェックを参照してください。

- Amazon CloudWatch アラーム - 指定した期間にわたって単一のメトリクスをモニタリングし、複数の期間にわたる特定のしきい値に対するメトリクスの値に基づいて 1 つ以上のアクションを実行します。アクションは Amazon Simple Notification Service (Amazon SNS) のトピックまたは Amazon EC2 Auto Scaling のポリシーに送信される通知です。アラームは持続している状態変化に対してのみアクションを呼び出します。CloudWatch アラームはメトリクスが特定の状態になっただけではアクションを呼び出しません。アクションを呼び出すには状態が変化して、指定した期間継続している必要があります。詳細については「CloudWatch を使用したインスタンスのモニタリング」を参照してください。
- Amazon EventBridge イベント - AWS サービスを自動化し、システムイベントに自動的に応答します。AWS サービスからのイベントはほぼリアルタイムに EventBridge に提供され、イベントが記述したルールと一致したときに実行する自動アクションを指定できます。詳細については「the section called “EventBridge を使用して自動化する”」を参照してください。
- AWS CloudTrail ログ - Amazon EC2 API に対して行われた呼び出しに関する詳細情報を取得し、Amazon S3 でログファイルとして保存します。CloudTrail ログを使用して、行われた呼び出し、呼び出しのソース IP アドレス、呼び出し元、および呼び出し時間を判断できます。詳細については「the section called “CloudTrail を使用して API 呼び出しを口グに記録する”」を参照してください。
- CloudWatch エージェント - EC2 インスタンスとオンプレミスサーバー上のホストとゲストの両方からログとシステムレベルのメトリクスを収集します。詳細についてはAmazon CloudWatch ユーザーガイドの「CloudWatch エージェントを使用して Amazon EC2 インスタンスとオンプレミスサーバーからメトリクスと口グを収集する」を参照してください。

Amazon EC2 インスタンスのステータスのモニタリング

インスタンスのステータスをモニタリングして、インスタンスのステータスチェックや、インスタンスにスケジュールされたイベントを表示できます。

ステータスチェックでは Amazon EC2 によって実行される自動化されたチェックからの情報が提供されます。これらの自動化されたチェックは特定の問題がインスタンスに影響を与えているかどうかを検出します。ステータスチェックの情報と、Amazon CloudWatch で提供されるデータによって、各インスタンスの詳細な動作状況を把握できます。

インスタンスに予定されている特定イベントのステータスを確認することもできます。イベントのステータスは再起動やリタイアなど、インスタンスに対して予定されている今後のアクティビティに関する情報を提供します。また、各イベントの予定開始予定時刻および終了時刻も提供されています。

内容

- Amazon EC2 インスタンスのステータスチェック
- Amazon EC2 インスタンスの状態変更イベント
- Amazon EC2 インスタンスの予定されているイベント

Amazon EC2 インスタンスのステータスチェック

インスタンスのステータスのモニタリングではインスタンスによるアプリケーションの実行を妨げる可能性のある問題を Amazon EC2 が検出したかどうかをすばやく判断できます。Amazon EC2 は稼働中のすべての EC2 インスタンスに対して自動チェックを実行して、ハードウェアおよびソフトウェアの問題を特定します。これらのステータスチェックの結果を表示して、具体的で検出可能な問題を識別できます。このイベントステータスデータは各インスタンス (`pending`、`running`、`stopping`) の状態に関して Amazon EC2 が既に提供している情報と、Amazon CloudWatch が監視している使用状況メトリクス (CPU 使用率、ネットワークトラフィック、ディスクアクティビティ) を補足するものです。

ステータスチェックは 1 分ごとに実行され、それぞれ成功または失敗のステータスが返ります。すべてのチェックが成功すると、インスタンス全体のステータスが OK になります。1つ以上のチェックが失敗すると、全体のステータスが impaired になります。ステータスチェックは Amazon EC2 に組み込まれています。そのため、無効にしたり、削除したりすることはできません。

ステータスチェックに失敗すると、ステータスチェックの対応する CloudWatch メトリクスは増加します。詳細についてはステータスチェックメトリクスを参照してください。このようなメトリクスを使用して、ステータスチェックの結果に基づいてトリガーされる CloudWatch アラームを作成することができます。例えば、特定のインスタンスでステータスチェックが失敗したときに警告するアラームを作成できます。詳細についてはステータスチェックに失敗した Amazon EC2 インスタンスの CloudWatch アラームの作成を参照してください。

また、Amazon EC2 インスタンスをモニタリングし、基になる問題によりインスタンスが正常に機能しなくなった場合に、自動的にインスタンスを復旧する Amazon CloudWatch アラームを作成できます。詳細についてはインスタンスの自動復旧を参照してください。

コンテンツ

- ステータスチェックのタイプ
- Amazon EC2 インスタンスのステータスチェックの表示
- ステータスチェックに失敗した Amazon EC2 インスタンスの CloudWatch アラームの作成

ステータスチェックのタイプ

ステータスチェックには 3 種類あります。

- システムステータスのチェック
- インスタンスステータスのチェック
- アタッチ済みの EBS ステータスチェック

システムステータスのチェック

システムステータスチェックはインスタンスが実行されている AWS システムをモニタリングします。これらのチェックでは修復には AWS の関与が必要なインスタンスの基盤の問題が検出されます。システムステータスチェックが失敗した場合、AWS が問題を解決するのを待つか、自分で解決できるかを選択できます。Amazon EBS でバックアップされたインスタンスの場合はインスタンスを自分で停止および起動することができます。通常、インスタンスは新しいホストに移行されます。インスタンスストアによってサポートされているインスタンスの場合 (Linux インスタンスのみをサポート)、インスタンスを終了して置き換えることができます。インスタンスストアボリュームは短期のものであり、インスタンスが停止するとすべてのデータが失われることに注意してください。

システムステータスチェックの失敗の原因となる問題の例を次に示します。

- ネットワーク接続の喪失
- システム電源の喪失
- 物理ホストのソフトウェアの問題
- ネットワーク到達可能性に影響する、物理ホスト上のハードウェアの問題

システムステータスチェックが失敗した場合、StatusCheckFailed_System メトリクスをインクリメントします。

ベアメタルインスタンス

ベアメタルインスタンスで再起動を実行すると、システムステータスチェックが一時的に失敗ステータスを返すことがあります。インスタンスが使用可能になると、システムステータスチェックからは成功ステータスが返されます。

インスタンスステータスのチェック

[インスタンスステータスのチェック] 個々のインスタンスのソフトウェアとネットワークの設定をモニタリングします。Amazon EC2 はネットワークインターフェイス (NIC) にアドレス解決プロトコ

ル (ARP) リクエストを送信することでインスタンスのヘルスをチェックします。これらのチェックではユーザーが関与して修復する必要のある問題が検出されます。インスタンスステータスチェックが失敗した場合は通常、自分自身で (例えば、インスタンスを再起動する、インスタンス設定を変更するなどによって) 問題に対処する必要があります。

> **Note**
>
> ネットワーク設定に `systemd-networkd` を使用する最近の Linux ディストリビューションではヘルスチェックに関するレポートが以前のディストリビューションとは異なる場合があります。起動プロセス中、このタイプのネットワークはインスタンスのヘルスにも影響する可能性のある他のスタートアップタスクよりも早く起動し、また早く終了する可能性もあります。ネットワークの可用性に依存するステータスチェックでは他のタスクが完了する前に正常なステータスをレポートできます。

インスタンスステータスチェックの失敗の原因となる問題の例を次に示します。

- 失敗したシステムステータスチェック
- 正しくないネットワークまたは起動設定
- メモリの枯渇
- 破損したファイルシステム
- 互換性のないカーネル
- 再起動中、インスタンスのステータスチェックはインスタンスが再び使用可能になるまで失敗を報告します。

インスタンスのステータスチェックが失敗した場合、StatusCheckFailed_Instance メトリクスをインクリメントします。

ベアメタルインスタンス

ベアメタルインスタンスで再起動を実行すると、インスタンスのステータスチェックが一時的に失敗ステータスを返すことがあります。インスタンスが使用可能になると、インスタンスステータスチェックからは成功ステータスが返されます。

アタッチ済みの EBS ステータスチェック

アタッチ済みの EBS ステータスチェックはインスタンスにアタッチされている Amazon EBS ボリュームが到達可能かどうか、および I/O 操作を完了できるかどうかをモニタリングしま

す。StatusCheckFailed_AttachedEBS メトリクスはインスタンスにアタッチされている 1 つ以上の EBS ボリュームが I/O 操作を完了できない場合に障害が発生することを示すバイナリ値です。これらのステータスチェックはコンピューティングまたは Amazon EBS インフラストラクチャの根本的な問題を検出します。アタッチ済みの EBS ステータスチェックメトリクスが失敗した場合は AWS を待って問題を解決するか、影響を受けたボリュームの置き換えやインスタンスの停止および再起動などのアクションを取ることができます。

アタッチ済みの EBS ステータスチェックが失敗する原因となる問題の例を次に示します。

- EBS ボリュームの基盤となるストレージサブシステムのハードウェアまたはソフトウェアの問題
- EBS ボリュームの到達可能性に影響する、物理ホスト上のハードウェアの問題
- インスタンスと EBS ボリューム間の接続に関する問題

StatusCheckFailed_AttachedEBS メトリクスを使うことで、ワークロードの耐障害性を向上できます。このメトリクスを使用して、ステータスチェックの結果に基づいてトリガーされる Amazon CloudWatch アラームを作成することができます。例えば、長期にわたる影響を検出した場合はセカンダリインスタンスまたはアベイラビリティーゾーンにフェイルオーバーできます。またはEBS CloudWatch メトリクスを使用してアタッチされた各ボリュームの I/O パフォーマンスをモニタリングし、障害のあるボリュームを検出して置き換えることもできます。EBS ステータスチェックで障害が検出され、アタッチされている EBS ボリュームにワークロードが I/O を発生させていない場合は、インスタンスを停止および開始してインスタンスを新しいホストに移行してください。これにより、EBS ボリュームの到達可能性に影響を与えている、基盤となるホストの問題を解決できます。詳細については「Amazon EBS の Amazon CloudWatch メトリクス」を参照してください。

アタッチ済みの EBS ステータスチェックの失敗を検出し、影響を受けたインスタンスを新しいインスタンスに置き換えるように Amazon EC2 Auto Scaling グループを設定することもできます。詳細については「Amazon EC2 Auto Scaling ユーザーガイド」の「障害のあるAmazon EBS ボリュームを持つ Auto Scaling インスタンスの監視と置き換え」を参照してください。

> **Note**
>
> アタッチ済みの EBS ステータスチェックメトリクスはNitro インスタンスでのみ使用できます。

Amazon EC2 インスタンスのステータスチェックの表示

インスタンスに失敗したステータスチェックがある場合、通常は自分自身で (例えば、インスタンスを再起動する、インスタンス設定を変更するなどによって) 問題に対処する必要があります。ご自分でシステムまたはインスタンスのステータスチェック失敗のトラブルシューティングを行う場合はステータスチェックに失敗した Amazon EC2 Linux インスタンスをトラブルシューティングするを参照してください。

Console

ステータスチェックを表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. [インスタンス] ページで、[Status check (ステータスチェック)] 列には各インスタンスの動作状況が表示されます。
4. 特定のインスタンスのステータスを表示するにはインスタンスを選択して、[ステータスとアラーム] タブを選択してください。
5. ステータスチェックで CloudWatch メトリクスを確認するには[ステータスとアラーム] タブで [メトリクス] を展開し、以下のメトリクスのグラフを表示します。

   - [システムのステータスチェックの失敗]
   - [インスタンスのステータスチェックの失敗]
   - [アタッチ済みの EBS ステータスチェックに失敗しました]

   詳細については、「the section called “ステータスチェックメトリクス”」を参照してください。

AWS CLI

ステータスチェックを表示するには

describe-instance-status コマンドを使用します。

例: 実行中のすべてのインスタンスのステータスを取得する

```
aws ec2 describe-instance-status
```

例: すべてのインスタンスのステータスを取得する

```
aws ec2 describe-instance-status --include-all-instances
```

例: 単一の実行中のインスタンスのステータスを取得する

```
aws ec2 describe-instance-status --instance-ids i-1234567890abcdef0
```

例: ステータスが impaired のすべてのインスタンスを取得する

```
aws ec2 describe-instance-status \
--filters Name=instance-status.status,Values=impaired
```

PowerShell

ステータスチェックを表示するには

Get-EC2VolumeStatus コマンドを使用します。

例: 実行中のすべてのインスタンスのステータスを取得する

```
Get-EC2InstanceStatus
```

例: すべてのインスタンスのステータスを取得する

```
Get-EC2InstanceStatus -IncludeAllInstance $true
```

例: 単一の実行中のインスタンスのステータスを取得する

```
Get-EC2InstanceStatus -InstanceId i-1234567890abcdef0
```

例: ステータスが impaired のすべてのインスタンスを取得する

```
Get-EC2InstanceStatus \
-Filter @{Name="instance-status.status"; Values="impaired"}
```

ステータスチェックに失敗した Amazon EC2 インスタンスの CloudWatch アラームの作成

ステータスチェックメトリクスを使用して、インスタンスのステータスチェックに失敗したときに通知されるように CloudWatch アラームを作成することができます。

メトリクスデータポイントが欠落している場合、ステータスチェックとステータスチェックアラームは一時的にデータ不足という状態に陥ることがあります。これはまれですが、インスタンスが正常であっても、メトリクスレポートシステムに中断がある場合に発生する場合があります。この状態を、ステータスチェックの失敗やアラーム違反ではなく、欠落データとして扱うことをお勧めします。これは、応答としてインスタンス上で停止、終了、再起動、または復旧アクションを行うときに特に重要です。

Console

この例では、インスタンスがステータスチェックに失敗したときに通知を送信するアラームを設定します。オプションで、インスタンスを停止、終了、または復旧できます。

ステータスチェックアラームを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択して、[Status Checks (ステータスチェック)] タブを選択し、[アクション]、[Create status check alarm (ステータスチェックアラームの作成)] を選択してください。
4. [CloudWatch アラームの管理] ページの [アラームの追加または編集] で、[Create an alarm (新しいアラームの作成)] を選択してください。
5. [Alarm notification (アラーム通知)] で、トグルをオンにして Amazon Simple Notification Service (Amazon SNS) 通知を設定します。既存の Amazon SNS トピックを選択するか、名前を入力して新しいトピックを作成します。

   受信者のリストに E メールアドレスを追加したか、トピックを新規作成した場合、Amazon SNS は新しいアドレスごとに確認の E メールを送信します。各受信者は、E メール内の確認リンクを選択する必要があります。確認されたアドレスのみがアラート通知を受け取ります。
6. [Alarm action (アラームアクション)] で、トグルをオンにして、アラームがトリガーされたときに実行するアクションを指定します。アクションを選択してください。
7. [Alarm thresholds (アラームのしきい値)] で、アラームのメトリクスと条件を指定します。

[サンプルをグループ化] (平均) と [サンプリングするデータのタイプ] (ステータスチェックも失敗) をデフォルト設定のままにするか、または必要に応じて変更することもできます。

[Consecutive period] (連続期間) の場合、評価する期間数を設定し、[Period] (期間) で、アラームをトリガーして E メールを送信するまでの評価の間隔を入力してください。

8. (オプション) [Sample metric data] (サンプルメトリクスデータ) の場合、[Add to dashboard] (ダッシュボードに追加) を選択してください。
9. [作成] を選択します。

インスタンスステータスのアラームを変更する必要がある場合、そのアラームを編集できます。

ステータスチェックアラームを編集するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択し、[Actions (アクション)]、[Monitoring (モニタリング)]、[CloudWatch アラームの管理] の順に選択してください。
4. [CloudWatch アラームの管理] ページの [アラームの追加または編集] で、[Edit an alarm (新しいアラームの編集)] を選択してください。
5. [Search for alarm (アラームの検索)] で、アラームを選択してください。
6. 変更が完了したら、[Update (更新)] を選択してください。

AWS CLI

次の例では、インスタンスが少なくとも 2 つの連続する期間内にインスタンスチェックまたはシステムステータスチェックに失敗した場合、アラームが SNS トピックに通知を発行します。使用する CloudWatch メトリクスは `StatusCheckFailed` です。

ステータスチェックアラームを作成するには

1. 既存の SNS トピックを選択するか、新しいキーペアを作成することができます。詳細については、AWS Command Line InterfaceAmazon Athena ユーザーガイド の Athena へのアクセスAWS CLIを参照してください。
2. Amazon EC2 の使用可能な Amazon CloudWatch メトリクスを表示するには list-metrics コマンドを使用します。

```
aws cloudwatch list-metrics --namespace AWS/EC2
```

3. アラームを作成するには次の put-metric-alarm コマンドを使用します。

```
aws cloudwatch put-metric-alarm \
    --alarm-name StatusCheckFailed-Alarm-for-i-1234567890abcdef0 \
    --metric-name StatusCheckFailed \
    --namespace AWS/EC2 \
    --statistic Maximum \
    --dimensions Name=InstanceId,Value=i-1234567890abcdef0 \
    --unit Count \
    --period 300 \
    --evaluation-periods 2 \
    --threshold 1 \
    --comparison-operator GreaterThanOrEqualToThreshold \
    --alarm-actions arn:aws:sns:us-west-2:111122223333:my-sns-topic
```

期間は Amazon CloudWatch メトリクスが収集される期間 (秒) です。この例では60 秒に 5 分を乗算した 300 を使用します。評価期間はメトリクスの値がしきい値と比較されなければならない連続した期間の数です。この例では 2 を使用します。アラームアクションはこのアラームがトリガーされたときに実行するアクションです。

PowerShell

ステータスチェックアラームを作成するには

Write-CWMetricAlarm コマンドレットを次のように使用して、インスタンスが少なくとも 2 つの連続する期間内にステータスチェックに失敗した場合に SNS トピックに通知を発行します。

```
Write-CWMetricAlarm `
    -AlarmName "StatusCheckFailed-Alarm-for-i-1234567890abcdef0" `
    -MetricName "StatusCheckFailed" `
    -Namespace "AWS/EC2" `
    -Statistic "Maximum" `
    -Dimension @{Name="InstanceId"; Values="i-1234567890abcdef0"} `
    -Unit "Count" `
    -Period 300 `
    -EvaluationPeriod 2 `
    -Threshold 1 `
    -ComparisonOperator "GreaterThanOrEqualToThreshold" `
```

```
    -AlarmAction "arn:aws:sns:us-west-2:111122223333:my-sns-topic"
```

期間は Amazon CloudWatch メトリクスが収集される期間 (秒) です。この例では60 秒に 5 分を乗算した 300 を使用します。評価期間はメトリクスの値がしきい値と比較されなければならない連続した期間の数です。この例では 2 を使用します。アラームアクションはこのアラームがトリガーされたときに実行するアクションです。

Amazon EC2 インスタンスの状態変更イベント

インスタンスの状態が変化すると、Amazon EC2 は Amazon EventBridge に `EC2 Instance State-change Notification` イベントを送信します。

以下はこのイベントのサンプルデータです。この例ではインスタンスは `pending` 状態に入りました。

```
{
   "id":"7bf73129-1428-4cd3-a780-95db273d1602",
   "detail-type":"EC2 Instance State-change Notification",
   "source":"aws.ec2",
   "account":"123456789012",
   "time":"2021-11-11T21:29:54Z",
   "region":"us-east-1",
   "resources":[
      "arn:aws:ec2:us-east-1:123456789012:instance/i-1234567890abcdef0"
   ],
   "detail":{
      "instance-id":"i-1234567890abcdef0",
      "state":"pending"
   }
}
```

`state` に指定できる値は次のとおりです。

- `pending`
- `running`
- `stopping`
- `stopped`
- `shutting-down`

- terminated

インスタンスを起動または開始した場合、インスタンスは pending 状態に移行してから、running 状態になります。インスタンスを停止した場合、インスタンスは stopping 状態に移行してから、stopped 状態になります。インスタンスを終了した場合、インスタンスは shutting-down 状態に移行してから、terminated 状態になります。詳細については、「Amazon EC2 インスタンスの状態変更」を参照してください。

Amazon EC2 インスタンスの状態が変更されたときに E メールを送信するアラームの作成

インスタンスの状態が変化したときに E メール通知を受け取るには、Amazon SNS トピックを作成してから、EC2 Instance State-change Notification イベントの EventBridge ルールを作成します。

SNS トピックを作成するには

1. Amazon SNS コンソールの https://console.aws.amazon.com/sns/v3/home を開いてください。
2. ナビゲーションペインで、[トピック] を選択してください。
3. [トピックの作成] を選択してください。
4. [Type (タイプ)] で、[Standard (標準)] を選択してください。
5. [Name] (名前) で、トピックの名前を入力してください。
6. [トピックの作成] を選択してください。
7. [Create subscription] を選択してください。
8. [Protocol (プロトコル)] として [Email (E メール)] を選択してください。
9. [Endpoint] (エンドポイント) で、通知を受信するメールアドレスを入力してください。
10. [Create subscription] を選択してください。
11. 次の件名の E メールメッセージが届きます: AWS Notification - Subscription Confirmation。指示に沿って操作し、登録を確認します。

EventBridge ルールを作成するには

1. Amazon EventBridge コンソールの https://console.aws.amazon.com/events/ を開いてください。
2. [Create rule] を選択してください。

3. [Name] (名前) に、ルールの名前を入力してください。
4. [ルールタイプ] で、[イベントパターンを持つルール] を選択してください。
5. [Next] を選択してください。
6. [Event pattern] (イベントパターン) の場合は次のいずれかを実行します。

   a. イベントソース で AWS のサービス を選択してください。
   b. [AWS のサービス] で、[EC2] を選択してください。
   c. [イベントタイプ] に、[EC2 インスタンスの状態変更通知] を選択してください。
   d. デフォルトではすべてのインスタンスの状態変更に関する通知が送信されます。必要に応じて、特定の状態またはインスタンスを選択できます。
7. [Next] を選択してください。
8. 次のようにターゲットを指定します。

   a. [Target types] (ターゲットタイプ) には[AWS のサービス] を選択してください。
   b. [Select a target] (ターゲットの選択) には[SNS topic] (SNS トピック) を選択してください。
   c. [Topic] (トピック) で、前の手順で作成した SNS トピックを選択してください。
9. [Next] を選択してください。
10. (オプション) ルールにタグを追加します。
11. [Next] を選択してください。
12. ルールの作成を選択してください。
13. ルールをテストするには状態変更を開始します。例えば、停止されたインスタンスを開始したり、実行中のインスタンスを停止したり、インスタンスを起動したりします。次の件名の E メールメッセージが届きます: AWS Notification Message。E メールの本文にはイベントデータが含まれます。

Amazon EC2 インスタンスの予定されているイベント

インフラストラクチャの信頼性とパフォーマンスを維持するために、 AWS はインスタンスの再起動、停止、終了イベントをスケジュールすることがあります。これらのイベントは頻繁には発生しません。

予定されたイベントの影響を受けるインスタンスがある場合、AWS は AWS アカウントに関連付けられている E メールアドレスを使用して、事前に E メールで通知します。この E メールには、開始日や終了日などのイベントに関する詳細が記載されています。イベントタイプによっては、イベ

ントのタイミングを制御するアクションを実行できる場合があります。AWS は AWS Health イベントも送信し、Amazon EventBridge を使用してモニタリングおよび管理できます。詳細については「Amazon EventBridge を使用した AWS Health でのイベントのモニタリング」を参照してください。

予定されたイベントは AWS によって管理されます。インスタンスのイベントをスケジュールすることはできません。ただし、以下はできます。

- インスタンスに予定されたイベントを表示します。
- 予定されたイベント通知をカスタマイズして、E メール通知にタグを追加または削除します。
- 特定の予定されたイベントを再スケジュールします。
- 予定されているイベントのカスタムイベントウィンドウを作成します。
- インスタンスの再起動、停止、またはリタイアが予定されているときにアクションを実行します。

スケジュールされたイベントの通知を確実に受信するには、[アカウント] ページで連絡先情報を確認します。

> **Note**
>
> インスタンスがスケジュールされたイベントの影響を受け、それが Auto Scaling グループの一部である場合、Amazon EC2 Auto Scaling はヘルスチェックの一部として最終的にそのインスタンスを置き換えるので、追加のアクションは必要ありません。Amazon EC2 Auto Scaling によって実行されるヘルスチェックの詳細については「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling グループのインスタンスのヘルスチェック」を参照してください。

予定されたイベントのタイプ

Amazon EC2 は、インスタンスに対して、スケジュールされた時間に発生する以下の種類のスケジュールされたイベントを作成できます。

イベントタイプ	イベントコード	イベントアクション
インスタンスの停止	`instance-stop`	スケジュールされた時刻になると、インスタンスは停止します。再度起動すると

イベントタイプ	イベントコード	イベントアクション
		、新しいホストに移行されます。Amazon EBS ルートボリュームを持つインスタンスに対してのみ適用されます。
インスタンスのリタイア	`instance-retirement`	スケジュールされた時刻に、インスタンスは Amazon EBS ルートボリュームがある場合は停止され、インスタンスストアルートボリュームがある場合は終了されます。
インスタンスの再起動	`instance-reboot`	スケジュールされた時刻になると、インスタンスは再起動されます。インスタンスはホストに留まり、再起動中にホストのメンテナンスが実行されます。これはインプレース再起動と呼ばれます。
システムの再起動	`system-reboot`	スケジュールされた時刻に、インスタンスが再起動され、新しいホストに移行されます。これは再起動移行と呼ばれます。
[System maintenance]	`system-maintenance`	スケジュールされた時刻になると、インスタンスはネットワークメンテナンスまたは電源のメンテナンスの影響を一時的に受ける場合があります。

イベントタイプを決定する

インスタンスでスケジュールされているイベントのタイプを確認できます。

Console

イベントタイプを確認するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインの [Events] を選択してください。
3. テーブルの [イベントタイプ] 列にイベントコードが表示されます。
4. テーブルをフィルタリングしてインスタンスのイベントのみを表示するには、検索フィールドでフィルターリストから[リソースタイプ: インスタンス] を選択します。

AWS CLI

インスタンスのイベントタイプを確認するには

describe-instance-status コマンドを使用します。インスタンスにスケジュールされたイベントが関連付けられている場合、出力にはスケジュールされたイベントに関する情報が提供されます。

```
aws ec2 describe-instance-status \
    --instance-ids i-1234567890abcdef0 \
    --query InstanceStatuses[].Events
```

以下は出力の例です。スケジュールされたイベントコードは system-reboot です。

```
[
    "Events": [
        {
            "InstanceEventId": "instance-event-0d59937288b749b32",
            "Code": "system-reboot",
            "Description": "The instance is scheduled for a reboot",
            "NotAfter": "2020-03-14T22:00:00.000Z",
            "NotBefore": "2020-03-14T20:00:00.000Z",
            "NotBeforeDeadline": "2020-04-05T11:00:00.000Z"
        }
    ]
]
```

PowerShell

インスタンスのイベントタイプを確認するには

Get-EC2InstanceStatus コマンドレットを使用します。インスタンスにスケジュールされたイベントが関連付けられている場合、出力にはスケジュールされたイベントに関する情報が提供されます。

```
(Get-EC2InstanceStatus `
    -InstanceId i-1234567890abcdef0).Events
```

以下は出力の例です。スケジュールされたイベントコードは system-reboot です。

```
Code              : system-reboot
Description       : The instance is scheduled for a reboot
InstanceEventId   : instance-event-0d59937288b749b32
NotAfter          : 2020-03-14T22:00:00.000Z
NotBefore         : 2020-03-14T20:00:00.000Z
NotBeforeDeadline : 2020-04-05T11:00:00.000Z
```

内容

- 停止またはリタイアがスケジュールされている Amazon EC2 インスタンスを管理する
- 再起動が予定されている Amazon EC2 インスタンスを管理する
- メンテナンスがスケジュールされている Amazon EC2 インスタンスを管理する
- Amazon EC2 インスタンスに影響する予定されているイベントの表示
- EC2 インスタンスのスケジュールされたイベント通知をカスタマイズする
- EC2 インスタンスのスケジュールされたイベントを再スケジュールする
- Amazon EC2 インスタンスに影響する予定されているイベントのカスタムイベントウィンドウの作成

停止またはリタイアがスケジュールされている Amazon EC2 インスタンスを管理する

AWS はインスタンスの基盤となるホストの回復不能な障害を検出すると、インスタンスのルートボリュームタイプに応じて、インスタンスを停止または終了するようにスケジュールします。

- インスタンスに Amazon EBS ルートボリュームがある場合、インスタンスは停止するようにスケジュールされます。
- インスタンスにインスタンスストアのルート ボリュームがある場合、インスタンスは終了するようにスケジュールされます。

詳細については、「インスタンスのリタイア」を参照してください。

> **Important**
>
> インスタンスストアボリュームに格納されているデータはいずれも、インスタンスが停止、休止、または終了されると失われます。これにはEBS ルートボリュームを備えたインスタンスにアタッチされたインスタンスストアボリュームも含まれます。インスタンスが停止、休止、または終了される前に、後で必要となるインスタンスストアボリュームからデータを必ず保存しておきます。

実行できるアクション

EBS ルートボリュームを持つインスタンスに対して実行できるアクション

スケジュールされた `instance-stop` イベント通知を受け取った場合、次のいずれかのアクションを実行できます。

- スケジュールされた停止の待機: 予定されたメンテナンスウィンドウ中にインスタンスが停止するのを待機できます。
- 手動による停止と起動の実行: 都合の良いときにインスタンスを自分で停止および起動して、新しいホストに移行できます。これは、インスタンスの再起動と同じではありません。詳細については、「Amazon EC2 インスタンスの停止と開始」を参照してください。
- 停止と開始の自動化: スケジュールされた `instance-stop` イベントに対応した、即時の停止と開始を自動化できます。詳細については「AWS Health ユーザーガイド」の「AWS Health のイベントに応答して EC2 インスタンスでオペレーションを自動的に実行」を参照してください。

インスタンスストアのルートボリュームを持つインスタンスに対して実行できるアクション

スケジュールされた`system-retirement`イベント通知を受信し、データを保持したい場合は、次の操作を実行できます。

1. 最新の AMI から置換インスタンスを起動します。
2. インスタンスの終了がスケジュールされる前に、必要なすべてのデータを代替インスタンスに移行します。
3. 元のインスタンスを終了するか、予定どおりに終了するまで待機します。

実行できるアクションの詳細については、「インスタンスのリタイア」を参照してください。

再起動が予定されている Amazon EC2 インスタンスを管理する

AWS は更新のインストールや基盤となるホストのメンテナンスなどのタスクを実行する必要があるとき、インスタンスの再起動をスケジュールできます。スケジュールされた再起動中、インスタンスは次のようにイベントに応じて、同じホストに留まるか、別のホストに移行します。

- `instance-reboot` イベント
  - 再起動中、インスタンスはホスト上に留まります。これはインプレース再起動と呼ばれます。
  - 現在のホストはメンテナンス中です。
  - 通常、秒単位で完了します。
- `system-reboot` イベント
  - 再起動中、インスタンスは新しいホストに移行されます。これは再起動移行と呼ばれます。
  - 通常、数分で完了します。

インスタンスでスケジュールされているイベントのタイプを確認するには、「イベントタイプを決定する」を参照してください。

実行できるアクション

スケジュールされた `instance-reboot` または `system-reboot` イベント通知を受け取った場合、次のいずれかのアクションを実行できます。

- スケジュールされた再起動の待機: スケジュールされたメンテナンスウィンドウ内にインスタンスの再起動が行われるまで待機できます。
- 再起動のスケジュールの変更: インスタンスの再起動を、都合のよい日時に合わせて再スケジュールできます。
- ユーザー操作による再起動の実行: ユーザーは好きなタイミングで手動でインスタンスを再起動できます。ただし、再起動結果はイベントによって異なります。
  - `instance-reboot` イベント – インスタンスは現在のハードウェア上に留まり (インプレース再起動)、ホストのメンテナンスは行われず、イベントは開いたままになります。
  - `system-reboot` イベント
    - インスタンスで再起動移行が有効になっている状態でユーザー操作による再起動を行うと、システムはインスタンスを新しいハードウェアに移行しようとします。成功すると、イベントはクリアされます。失敗するとインプレース再起動が発生し、イベントはスケジュールされたままになります。

- インスタンスで再起動移行が無効になっている状態では、ユーザー操作による再起動を行ってもインスタンスは同じハードウェアに保持され (インプレース再起動)、ホストのメンテナンスは行われず、イベントはスケジュールされたままになります。その後スケジュールされたイベントが予定通りに発生すると、AWS はインスタンスを新しいハードウェアに移動します (再起動移行)。

AWS がインスタンスを再起動した後

AWS がインスタンスを再起動した後、以下が適用されます。

- スケジュールされたイベントはクリアされます。
- イベントの説明が更新されました。
- `instance-reboot` イベントの場合:
  - 基盤となるホストのメンテナンスが完了しました。
- `system-reboot` イベントの場合:
  - インスタンスは新しいホストに移動します。
  - インスタンスは IP アドレスと DNS 名を保持します。
  - ローカルインスタンスストアボリュームのデータはすべて保持されます。
- インスタンスは、完全に起動したら使用できます。

代替オプション

再起動イベントを再スケジュールできない、またはユーザー操作による再起動に対して再起動移行を有効にできないが、スケジュールされているメンテナンスウィンドウの予定時間には通常の操作を維持したい場合は、次の操作を行ってください。

- EBS ルートボリュームを持つインスタンスの場合
  - インスタンスを手動で停止して起動し、新しいホストに移行します。これは、インスタンスを同じホストに留まっているインスタンスを手動で再起動する場合とは異なります。
  - オプションで、スケジュールされた再起動イベントに対応した、インスタンスの即時の停止と開始を自動化することができます。詳細については「AWS Health ユーザーガイド」の「AWS Health のイベントに応答して EC2 インスタンスでオペレーションを自動的に実行」を参照してください。

> **Important**
>
> インスタンスストアボリューム上のデータは、インスタンスの停止に伴って失われます。詳細については、「Amazon EC2 インスタンスの停止と開始」を参照してください。

- インスタンスストアのルートボリュームを持つインスタンスの場合
  1. 最新の AMI から置換インスタンスを起動します。
  2. スケジュールされたメンテナンスウィンドウの前に、必要なすべてのデータを代替インスタンスに移行します。
  3. 元のインスタンスを終了します。

再起動移行を有効または無効にする

インスタンスが `system-reboot` イベントにスケジュールされている場合も、イベント予定日時の前にインスタンスを手動で再起動できます。ユーザー操作による再起動の結果は、そのインスタンスの再起動移行設定によって異なります。

- 有効 – ユーザー操作による再起動を行うと、システムはインスタンスを新しいハードウェアに移行しようとします (再起動移行)。成功すると、イベントはクリアされます。失敗するとインプレース再起動が発生し、イベントはスケジュールされたままになります。なお、再起動移行を有効に設定していても、再起動移行はそのインスタンスが再起動移行要件を満たしている場合にのみ実行可能である点に留意してください。
- 無効 – ユーザー操作による再起動を行ってもインスタンスは同じハードウェアに保持され (インプレース再起動)、ホストメンテナンスは行われず、イベントはスケジュールされたままになります。その後スケジュールされたイベントが予定通りに発生すると、AWS はインスタンスを新しいハードウェアに移動します (再起動移行)。

移行を伴う再起動は、インプレース再起動よりも時間がかかります。

- インプレース再起動: 約 30 秒
- 移行を伴う再起動: 数分

> **Note**
>
> `system-reboot` イベント通知を受信するインスタンスでは、デフォルトでユーザー操作による再起動移行が有効になっています。

再起動移行を有効にするための要件

再起動移行は、次の条件を満たすインスタンスで有効にできます。

インスタンスのタイプ

すべてのインスタンスタイプが再起動移行の有効化をサポートしているわけではありません。再起動移行の有効化をサポートするインスタンスタイプを確認することができます。

Console

再起動移行の有効化をサポートするインスタンスタイプを確認するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左のナビゲーションペインで、[Instance Types] (インスタンスタイプ) を選択してください。
3. フィルターバーに、Reboot Migration support: supported と入力します。あるいはこの文字列を入力していくと該当するフィルター名が表示されるので、そのフィルターを選択できます。

   [インスタンスタイプ] テーブルに、再起動移行の有効化をサポートするすべてのインスタンスタイプが表示されます。

AWS CLI

再起動移行の有効化をサポートするインスタンスタイプを確認するには

describe-instance-types コマンドを `reboot-migration-support` フィルターと共に使用します。

```
aws ec2 describe-instance-types \
    --filters Name=reboot-migration-support,Values=supported \
    --query "InstanceTypes[*].[InstanceType]" \
    --output text | sort
```

PowerShell

再起動移行の有効化をサポートするインスタンスタイプを確認するには

Get-EC2InstanceType コマンドレットを reboot-migration-support フィルタと共に使用します。

```
Get-EC2InstanceType `
    -Filter @{Name="reboot-migration-support";Values="true"} | `
    Select InstanceType | Sort-Object InstanceType
```

テナンシー

- Shared
- ハードウェア専有インスタンス

詳細については「Amazon EC2 ハードウェア専有インスタンス」を参照してください。

制限事項

再起動移行は、次の特性を持つインスタンスではサポートされていません。

- プラットフォーム: Xen ハイパーバイザーでネイティブに実行されているインスタンス
- インスタンスサイズ: metalインスタンス
- テナンシー: 専用ホスト。専用ホスト の場合は代わりに Dedicated Host Auto Recovery を使用します。
- ストレージ: インスタンスストアボリュームを持つインスタンス
- ネットワーク: エラスティックファブリックアダプタ を使用するインスタンス
- Auto Scaling: Auto Scaling グループの一部であるインスタンス

再起動移行を有効化または無効化する手順

インスタンスが system-reboot イベントを受信すると、デフォルトで再起動移行が有効になります。再起動移行が無効に設定されている場合、ユーザー操作による再起動を行ってもインスタンスは同じハードウェア上に保持されます (インプレース再起動)。

default 設定では、サポートされていないインスタンスの再起動移行を有効にすることはできません。詳細については、「再起動移行を有効にするための要件」を参照してください。

実行中または停止中のインスタンスで再起動移行を無効または有効にできます。

AWS CLI

再起動移行を無効にするには

modify-instance-maintenance-options コマンドを使用して、--reboot-migration パラメータを disabled に設定します。

```
aws ec2 modify-instance-maintenance-options \

    --instance-id i-0abcdef1234567890 \

    --reboot-migration disabled
```

再起動移行を有効にするには

modify-instance-maintenance-options コマンドを使用して、--reboot-migration パラメータを default に設定します。

```
aws ec2 modify-instance-maintenance-options \

    --instance-id i-0abcdef1234567890 \

    --reboot-migration default
```

PowerShell

再起動移行を無効にするには

Edit-EC2InstanceMaintenanceOption コマンドレットを使用します。

```
Edit-EC2InstanceMaintenanceOption `
    -InstanceId  `
    -RebootMigration Disabled
```

再起動移行を有効にするには

Edit-EC2InstanceMaintenanceOption コマンドレットを使用します。

```
Edit-EC2InstanceMaintenanceOption `
    -InstanceId i-1234567890abcdef0 `
    -RebootMigration Enabled
```

メンテナンスがスケジュールされている Amazon EC2 インスタンスを管理する

AWS はインスタンスの基盤となるホストをメンテナンスする必要があるときに、インスタンスのメンテナンスを予定します。2 種類のメンテナンスイベントがあります。1 つはネットワークメンテナンスで、もう 1 つは電源のメンテナンスです。

- ネットワークメンテナンス中は短い期間、予定されたインスタンスのネットワーク接続が切断されます。メンテナンスが終了すると、インスタンスとの通常のネットワーク接続が回復します。
- 電源のメンテナンス中は短い期間、予定されたインスタンスはオフラインになり、その後再起動されます。再起動されると、インスタンスの設定内容はすべて維持されます。

インスタンスが再起動したら (通常、数分かかります)、アプリケーションが正常に動作していることを確認します。この時点で、インスタンスにスケジュールされたイベントは残っていません。残っている場合はスケジュールされたイベントの説明の先頭に [Completed] と表示されます。インスタンスのステータス説明が更新するのに、最大で 1 時間ほどかかる場合があります。完了したメンテナンスイベントは最長で 1 週間、Amazon EC2 コンソールのダッシュボードに表示されます。

実行できるアクション

EBS ルートボリュームを持つインスタンスに対して実行できるアクション

`system-maintenance` イベント通知を受け取った場合、次のいずれかのアクションを実行できます。

- スケジュールされたメンテナンスの待機: メンテナンスがスケジュールどおりに実行されるのを待機できます。
- 手動で停止と停止の実行: インスタンスの停止と起動を行うと、インスタンスは別のホストに移行されます。これは、インスタンスの再起動と同じではありません。詳細については、「Amazon EC2 インスタンスの停止と開始」を参照してください。
- 停止と開始の自動化: スケジュールされたメンテナンスイベントに対応した、即時の停止と開始を自動化することができます。詳細については「AWS Health ユーザーガイド」の「AWS Health のイベントに応答して EC2 インスタンスでオペレーションを自動的に実行」を参照してください。

インスタンスストアのルートボリュームを持つインスタンスに対して実行できるアクション

`system-maintenance` イベント通知を受け取った場合、次のいずれかのアクションを実行できます。

- スケジュールされたメンテナンスの待機: メンテナンスがスケジュールどおりに実行されるのを待機できます。
- 代替インスタンスの起動: スケジュールされたメンテナンスウィンドウ中に通常の操作を維持する場合:
  1. 最新の AMI から置換インスタンスを起動します。
  2. スケジュールされたメンテナンスウィンドウの前に、必要なすべてのデータを代替インスタンスに移行します。
  3. 元のインスタンスを終了します。

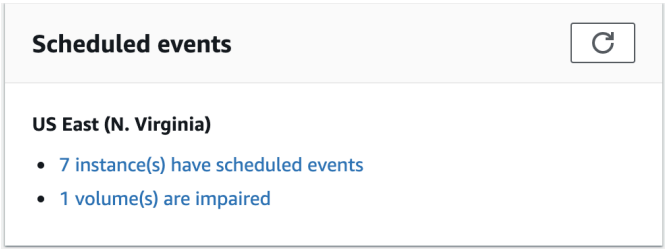
Amazon EC2 インスタンスに影響する予定されているイベントの表示

スケジュールされたイベント通知を E メールで受信することに加え、スケジュールされたイベントを確認できます。

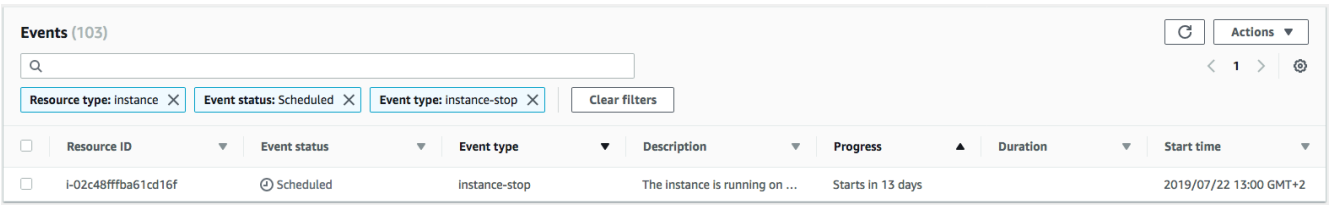
Console

インスタンスに予定されたイベントを表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ダッシュボードには[Scheduled events] の下に関連付けられているイベントを持つすべてのリソースが表示されます。



3. 詳細についてはナビゲーションペインで [イベント] を選択してください。イベントに関連付けられたリソースがすべて表示されます。イベントタイプ、リソースタイプ、アベイラビリティーゾーンなどの特性でフィルタリングできます。



AWS CLI

インスタンスに予定されたイベントを表示するには

describe-instance-status コマンドを使用します。

```
aws ec2 describe-instance-status \
    --instance-ids i-1234567890abcdef0 \
    --query "InstanceStatuses[].Events"
```

以下の出力例は再起動イベントを示しています。

```
[
    "Events": [
        {
            "InstanceEventId": "instance-event-0d59937288b749b32",
            "Code": "system-reboot",
            "Description": "The instance is scheduled for a reboot",
            "NotAfter": "2019-03-15T22:00:00.000Z",
            "NotBefore": "2019-03-14T20:00:00.000Z",
            "NotBeforeDeadline": "2019-04-05T11:00:00.000Z"
         }

    ]
]
```

インスタンスのリタイアイベントを示す出力例を次に示します。

```
[
    "Events": [
        {
            "InstanceEventId": "instance-event-0e439355b779n26",

            "Code": "instance-stop",
            "Description": "The instance is running on degraded hardware",
            "NotBefore": "2015-05-23T00:00:00.000Z"
        }
    ]
]
```

PowerShell

インスタンスに予定されたイベントを表示するには

次の Get-EC2InstanceStatus コマンドを使用します。

```
(Get-EC2InstanceStatus -InstanceId i-1234567890abcdef0).Events
```

インスタンスのリタイアイベントを示す出力例を次に示します。

```
Code         : instance-stop
Description  : The instance is running on degraded hardware
NotBefore    : 5/23/2015 12:00:00 AM
```

Instance metadata

インスタンスメタデータを使用してインスタンスに予定されたイベントを表示するには

インスタンスのアクティブなメンテナンスイベントに関する情報はインスタンスメタデータサービスバージョン 2 または インスタンスメタデータサービスバージョン 1 を使用してインスタンスメタデータから取得できます。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/events/maintenance/scheduled
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/events/maintenance/scheduled
```

以下は予定されたシステムの再起動イベントに関する情報を JSON 形式で出力した例です。

```
[
  {
    "NotBefore" : "21 Jan 2019 09:00:43 GMT",
    "Code" : "system-reboot",
    "Description" : "scheduled reboot",
    "EventId" : "instance-event-0d59937288b749b32",
    "NotAfter" : "21 Jan 2019 09:17:23 GMT",
```

```
    "State" : "active"
  }
]
```

インスタンスメタデータを使用して、インスタンスの完了またはキャンセルされたイベントのイベント履歴を表示するには

インスタンスの完了済みまたはキャンセル済みイベントに関する情報はインスタンスメタデータサービスバージョン 2 または インスタンスメタデータサービスバージョン 1 を使用してインスタンスメタデータから取得できます。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/events/maintenance/history
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/events/maintenance/history
```

以下は取り消されたシステム再起動イベントおよび完了したシステム再起動イベントに関する情報を JSON 形式で出力した例です。

```
[
  {
    "NotBefore" : "21 Jan 2019 09:00:43 GMT",
    "Code" : "system-reboot",
    "Description" : "[Canceled] scheduled reboot",
    "EventId" : "instance-event-0d59937288b749b32",
    "NotAfter" : "21 Jan 2019 09:17:23 GMT",
    "State" : "canceled"
  },
  {
    "NotBefore" : "29 Jan 2019 09:00:43 GMT",
    "Code" : "system-reboot",
    "Description" : "[Completed] scheduled reboot",
    "EventId" : "instance-event-0d59937288b749b32",
    "NotAfter" : "29 Jan 2019 09:17:23 GMT",
```

```
    "State" : "completed"
  }
]
```

AWS Health

AWS Health Dashboard を使用して、インスタンスに影響を与える可能性があるイベントについて確認できます。Health Dashboard では未解決の問題、予定された変更、その他の通知という 3 つのグループに問題が分類されます。予定された変更には進行中または予定されている変更が含まれます。

詳細についてはAWS Health ユーザーガイドの「AWS Health Dashboard の開始方法」を参照してください。

EC2 インスタンスのスケジュールされたイベント通知をカスタマイズする

スケジュールされたイベント通知をカスタマイズして、メール通知にタグを含めることができます。これにより、影響を受けるリソース (インスタンスまたは 専用ホスト) を特定して、その後のイベントに対するアクションに優先順位を付けやすくなります。

タグを含めるようにイベント通知をカスタマイズする場合、次のいずれかを含めることができます。

- 影響を受けるリソースに関連付けられているすべてのタグ
- 影響を受けるリソースに関連付けられている特定のタグのみ

例えば、`application`、`costcenter`、`project`、`owner` タグをすべてのインスタンスに割り当てるとします。イベント通知にはこれらのすべてのタグを含めることができます。また、イベント通知に `owner` タグと `project` タグのみを表示したい場合はそれらのタグのみを含めることもできます。

含めるタグを選択すると、イベント通知には影響を受けるリソースに関連付けられているリソース ID (インスタンス ID または Dedicated Host ID) とタグのキーと値のペアが含まれます。

タスク

- イベント通知にタグを含める
- イベント通知からのタグの削除
- イベント通知に含めるタグの表示

イベント通知にタグを含める

含めるように選択したタグは選択したリージョンのすべてのリソース (インスタンスと 専用ホスト) に適用されます。他のリージョンのイベント通知をカスタマイズするにはまず必要なリージョンを選択してから、次の手順を実行します。

Console

イベント通知にタグを含めるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインの [Events] を選択してください。
3. [アクション]、[Manage event notifications (イベント通知の管理)] の順に選択してください。
4. [イベント通知にタグを含める]をオンにします。
5. イベント通知に含めるタグに応じて、次のいずれかの操作を行います。
   - 影響を受けるインスタンスまたは専有ホストに関連付けられている全タグを含めるには[全タグを含める] を選択してください。
   - 含めるタグを選択するには [含めるタグを選択] を選択し、タグキーを選択または入力してください。
6. [Save] を選択してください。

AWS CLI

イベント通知にすべてのタグを含めるには

`IncludeAllTagsOfInstance` コマンドの register-instance-event-notification-attributes を使用して、`true` パラメータをに設定します。

```
aws ec2 register-instance-event-notification-attributes \
    --instance-tag-attribute "IncludeAllTagsOfInstance=true"
```

イベント通知に特定のタグを含めるには

コマンドの register-instance-event-notification-attributes を使用して、パラメー`InstanceTagKeys` タを使用して含めるタグを指定します。

```
aws ec2 register-instance-event-notification-attributes \
```

```
    --instance-tag-attribute 'InstanceTagKeys=["tag_key_1", "tag_key_2",
 "tag_key_3"]'
```

PowerShell

イベント通知にすべてのタグを含めるには

Register-EC2InstanceEventNotificationAttribute コマンドレットを使用します。

```
Register-EC2InstanceEventNotificationAttribute `
    -InstanceTagAttribute_IncludeAllTagsOfInstance $true
```

イベント通知に特定のタグを含めるには

Register-EC2InstanceEventNotificationAttribute コマンドレットを使用します。

```
Register-EC2InstanceEventNotificationAttribute `
    -InstanceTagAttribute_InstanceTagKey tag_key_1, tag_key_2, tag_key_3
```

イベント通知からのタグの削除

イベント通知からタグを削除できます。

Console

イベント通知からタグを削除するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインの [Events] を選択してください。
3. [アクション]、[Manage event notifications (イベント通知の管理)] の順に選択してください。
4. イベント通知からすべてのタグを削除するには[イベント通知にタグを含める] をオフにします。
5. イベント通知から特定のタグを削除するには対応するタグキーの [X] を選択してください。
6. [Save] を選択してください。

AWS CLI

イベント通知からすべてのタグを削除するには

IncludeAllTagsOfInstance コマンドの deregister-instance-event-notification-attributes を使用して、false パラメータをに設定します。

```
aws ec2 deregister-instance-event-notification-attributes \
    --instance-tag-attribute "IncludeAllTagsOfInstance=false"
```

イベント通知からタグを削除するには

コマンドの deregister-instance-event-notification-attributes を使用して、パラメInstanceTagKeysタを使用して削除するタグを指定します。

```
aws ec2 deregister-instance-event-notification-attributes \
    --instance-tag-attribute 'InstanceTagKeys=["tag_key_3"]'
```

PowerShell

イベント通知からすべてのタグを削除するには

Unregister-EC2InstanceEventNotificationAttribute コマンドレットを使用します。

```
Unregister-EC2InstanceEventNotificationAttribute `
    -InstanceTagAttribute_IncludeAllTagsOfInstance $false
```

イベント通知からタグを削除するには

Unregister-EC2InstanceEventNotificationAttribute コマンドレットを使用します。

```
Unregister-EC2InstanceEventNotificationAttribute `
    -InstanceTagAttribute_InstanceTagKey tag_key_3
```

イベント通知に含めるタグの表示

イベント通知に含めるタグを表示できます。

Console

イベント通知に含めるタグを表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

2. ナビゲーションペインの [Events] を選択してください。
3. [アクション]、[Manage event notifications (イベント通知の管理)] の順に選択してください。

AWS CLI

イベント通知に含めるタグを表示するには

コマンドの describe-instance-event-notification-attributes を使用します。

```
aws ec2 describe-instance-event-notification-attributes
```

PowerShell

イベント通知に含めるタグを表示するには

Get-EC2InstanceEventNotificationAttribute コマンドレットを使用します。

```
Get-EC2InstanceEventNotificationAttribute
```

EC2 インスタンスのスケジュールされたイベントを再スケジュールする

都合の良い日時にイベントが発生するように、予定を再スケジュールできます。イベントを再スケジュールすると、更新された日付が表示されるまでに 1～2 分かかる場合があります。

制限事項

- イベント期限があるイベントのみ再スケジュールできます。イベントはイベント期限日まで再スケジュールできます。イベントの期限は、[期限] 列 (コンソール) と `NotBeforeDeadline` フィールド (AWS CLI) に表示されます。
- まだ開始していないイベントのみ再スケジュールできます。開始時刻は、[開始時刻] 列 (コンソール) と `NotBefore` フィールド (AWS CLI) に表示されます。あと 5 分で開始するようにスケジュールされているイベントは再スケジュールできません。
- 新しいイベント開始時刻は現在の時刻から少なくとも 60 分後にする必要があります。
- コンソールを使用して複数のイベントを再スケジュールすると、イベント期限は最も早い期限日のイベントによって決定されます。

Console

イベントを再スケジュールするには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインの [Events] を選択してください。
3. フィルターリストから [リソースタイプ: インスタンス] を選択してください。
4. 1 つ以上のインスタンスを選択し、[アクション]、[Schedule event] の順に選択してください。

   [期限] でイベント期限を設定したイベントのみを再スケジュールできます。選択したイベントのいずれかに期限がない場合、[アクション]、[Schedule event] は無効になります。
5. [New start time] に、イベントの新しい日時を入力してください。新しい日時は[Event deadline] より前に設定する必要があります。
6. [Save] を選択してください。

   更新されたイベント開始時刻がコンソールに反映されるまで、1～2 分かかることがあります。

AWS CLI

イベントを再スケジュールするには

modify-instance-event-start-time コマンドを使用します。

```
aws ec2 modify-instance-event-start-time \
    --instance-id i-1234567890abcdef0 \
    --instance-event-id instance-event-0d59937288b749b32 \
    --not-before 2020-03-25T10:00:00.000
```

PowerShell

イベントを再スケジュールするには

Edit-EC2InstanceEventStartTime コマンドレットを使用します。

```
Edit-EC2InstanceEventStartTime `
    -InstanceId i-1234567890abcdef0 `
    -InstanceEventId instance-event-0d59937288b749b32 `
    -NotBefore 2020-03-25T10:00:00.000
```

Amazon EC2 インスタンスに影響する予定されているイベントのカスタムイベントウィンドウの作成

スケジュールされたイベントに対して、週ごとに繰り返されるカスタムのイベントウィンドウを定義して、Amazon EC2 インスタンスを再起動、停止、終了させることができます。イベントウィンドウには1 つ以上のインスタンスを関連付けることができます。これらのインスタンスにスケジュールされたイベントが設定されている場合、AWS は関連するイベントウィンドウ内でイベントをスケジュールします。

ワークロードのオフピーク期間にイベントウィンドウを指定することで、ワークロードの可用性を最大化できます。また、内部的な保守スケジュールにイベントウィンドウを合わせることもできます。

イベントウィンドウを定義するには一連の時間範囲を指定します。最小期間は 2 時間です。全体を合計した時間範囲は最小で 4 時間必要です。

インスタンス ID またはインスタンスタグを使用して、1 つ以上のインスタンスをイベントウィンドウに関連付けることができます。また、ホスト ID を使用して、専用ホスト をイベントウィンドウに関連付けることもできます。

> **Warning**
>
> イベントウィンドウはインスタンスを停止、再起動、または終了する、スケジュールされたイベントにのみ適用されます。
> イベントウィンドウは以下には適用されません。
>
> - 繰り上げられた、スケジュールされたイベントとネットワーク保守イベント。
> - 自動インスタンスリカバリや予期しない再起動などのスケジュール外の保守作業。

イベントウィンドウの使用

- 考慮事項
- イベントウィンドウを作成する
- イベントウィンドウとターゲットの関連付け
- イベントウィンドウからターゲットの関連付けを解除する
- イベントウィンドウを変更する
- イベントウィンドウを削除する

考慮事項

- イベントウィンドウの時刻はすべて UTC で表示されます。
- イベントウィンドウには、複数の時間範囲を含めることができます。各範囲は少なくとも 2 時間である必要がありますが、すべての範囲の合計期間は少なくとも 4 時間である必要があります。
- イベントウィンドウにはターゲットタイプ (インスタンスID、Dedicated Host ID、またはインスタンスタグ) を 1 つだけ関連付けることができます。
- 1 つのターゲット (インスタンス ID、Dedicated Host ID、またはインスタンスタグ) は1 つのイベントウィンドウにのみ関連付けることが可能です。
- 1 つのイベントウィンドウには最大 100 個のインスタンス ID、または 50 個の Dedicated Host ID、または 50 個のインスタンスタグを関連付けることができます。インスタンスタグは任意の数のインスタンスに関連付けることができます。
- 個々の AWS リージョンで、最大 200 個までのイベントウィンドウを作成できます。
- 複数のインスタンスがイベントウィンドウに関連付けられている場合、スケジュールされたイベントが同時に発生する可能性があります。
- 既に AWS によりスケジュールされたイベントが存在する場合、イベントウィンドウを変更しても、スケジュールされたイベントの時間は変更されません。イベントに締め切り日がある場合はイベントの再スケジュールが行えます。
- スケジュールされたイベントの前にインスタンスを停止および開始できます。これにより、インスタンスが新しいホストに移行され、イベントがクリアされます。

イベントウィンドウを作成する

イベントウィンドウは複数作成できます。イベントウィンドウごとに、1 つ以上の時間ブロックを指定します。例えば、毎日の午前 4 時に発生し 2 時間継続する時間ブロックを持つイベントウィンドウを作成できます。あるいは日曜日の午前 2 時から午前 4 時、および水曜日の午前 3 時から午前 5 時に発生する時間ブロックを持つイベントウィンドウを作成することもできます。

イベントウィンドウは削除されない限り毎週繰り返されます。

Console

イベントウィンドウを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインの [Events] を選択してください。

3. [Actions] (アクション)、[Manage event windows] (イベントウィンドウの管理) を選択してください。
4. [インスタンスのイベントウィンドウを作成] をクリックしてください。
5. [イベントウィンドウ名] に、イベントウィンドウのわかりやすい名前を入力してください。
6. [イベントウィンドウのスケジュール] で、イベントウィンドウ内の時間ブロックを指定するために、Cron スケジュールビルダーを使用するか、あるいは時間範囲で指定するかを選択してください。

   - [Cron スケジュールビルダー] を選択した場合は以下を指定します。
     1. [曜日 (UTC)] で、イベントウィンドウを発生させる曜日を指定します。
     2. [開始時刻 (UTC)] で、イベントウィンドウが開始する時刻を指定します。
     3. [期間] で、イベントウィンドウ内の時間ブロックの継続時間を指定します。各時間ブロックに設定できる最小期間は 2 時間です。イベントウィンドウの最小期間は合計で 4 時間以上にする必要があります。すべての時刻は協定世界時 (UTC) です。
   - [時間範囲] を選択した場合は[新しい時間範囲の追加] をクリックし、開始する日時ならびに終了する日時を指定します。各時間範囲に対して、これを繰り返します。各時間範囲に設定できる最小期間は 2 時間です。時間範囲の最小期間は全体を合計して 4 時間以上にする必要があります。
7. (オプション) [ターゲットの詳細] で、1 つ以上のインスタンスをイベントウィンドウに関連付けます。インスタンスを関連付けるには、インスタンス ID またはインスタンス タグを使用します。ホスト ID を使用して専有ホストを関連付けます。これらのターゲットのメンテナンスがスケジュールされている場合、イベントはこのイベントウィンドウ中に発生します。

   イベントウィンドウの作成時、そのウィンドウとターゲットの関連付けは必須ではありません。作成後、ウィンドウを変更して、1 つ以上のターゲットを関連付けることができます。
8. (オプション) [イベントウィンドウのタグ] で、[タグを追加] をクリックし、タグのキーおよび値を入力してください。各タグについて、これを繰り返します。
9. [イベントウィンドウの作成] をクリックしてください。

AWS CLI

時間範囲を設定したイベントウィンドウを作成するには

create-instance-event-window コマンドを使用します。

```
aws ec2 create-instance-event-window \
    --time-range StartWeekDay=monday,StartHour=2,EndWeekDay=wednesday,EndHour=8 \
    --tag-specifications "ResourceType=instance-event-
window,Tags=[{Key=K1,Value=V1}]" \
    --name myEventWindowName
```

以下は出力の例です。

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "TimeRanges": [
            {
                "StartWeekDay": "monday",
                "StartHour": 2,
                "EndWeekDay": "wednesday",
                "EndHour": 8
            }
        ],
        "Name": "myEventWindowName",
        "State": "creating",
        "Tags": [
            {
                "Key": "K1",
                "Value": "V1"
            }
        ]
    }
}
```

cron 式を指定したイベントウィンドウを作成するには

create-instance-event-window コマンドを使用します。

```
aws ec2 create-instance-event-window \
    --cron-expression "* 21-23 * * 2,3" \
    --tag-specifications "ResourceType=instance-event-
window,Tags=[{Key=K1,Value=V1}]" \
    --name myEventWindowName
```

以下は出力の例です。

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "State": "creating",
        "Tags": [
            {
                "Key": "K1",
                "Value": "V1"
            }
        ]
    }
}
```

PowerShell

時間範囲を設定したイベントウィンドウを作成するには

New-EC2InstanceEventWindow コマンドレットを使用します。

```
$timeRange = New-Object Amazon.EC2.Model.InstanceEventWindowTimeRangeRequest
$timeRange.StartWeekDay = "monday"
$timeRange.EndWeekDay = "wednesday"
$timeRange.StartHour = 2
$timeRange.EndHour = 8
$tag = @{Key="key1"; Value="value1"}
$tagspec = New-Object Amazon.EC2.Model.TagSpecification
$tagspec.ResourceType = "instance-event-window"
$tagspec.Tags.Add($tag)
New-EC2InstanceEventWindow `
    -Name my-event-window `
    -TagSpecification $tagspec `
    -TimeRange @($timeRange)
```

以下は出力の例です。

```
AssociationTarget     :
CronExpression        :
InstanceEventWindowId : iew-0abcdef1234567890
Name                  : my-event-window
State                 : creating
```

```
Tags                    : {key1}
TimeRanges              : {Amazon.EC2.Model.InstanceEventWindowTimeRange}
```

cron 式を指定したイベントウィンドウを作成するには

New-EC2InstanceEventWindow コマンドレットを使用します。

```
$tag = @{Key="key1"; Value="value1"}
$tagspec = New-Object Amazon.EC2.Model.TagSpecification
$tagspec.ResourceType = "instance-event-window"
$tagspec.Tags.Add($tag)
New-EC2InstanceEventWindow `
    -Name my-event-window `
    -TagSpecification $tagspec`
    -CronExpression "* 21-23 * * 2,3"
```

以下は出力の例です。

```
AssociationTarget     :
CronExpression        : * 21-23 * * 2,3
InstanceEventWindowId : iew-0abcdef1234567890
Name                  : my-event-window
State                 : creating
Tags                  : {key1}
TimeRanges            : {}
```

イベントウィンドウとターゲットの関連付け

イベントウィンドウを作成した後に、ターゲットをイベントウィンドウに関連付けることができます。イベントウィンドウに関連付けることができるターゲットのタイプは 1 つだけです。インスタンス ID、専有ホスト ID、またはインスタンスタグを指定できます。

Console

イベントウィンドウにターゲットを関連付けるには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインの [Events] を選択してください。
3. 変更するイベントウィンドウを選択します。
4. [アクション]、[インスタンスイベントウィンドウを変更] の順に選択します。

AWS CLI

イベントウィンドウとインスタンスタグを関連付けるには

associate-instance-event-window コマンドを使用します。

```
aws ec2 associate-instance-event-window \
    --instance-event-window-id iew-0abcdef1234567890 \
    --association-target "InstanceTags=[{Key=k2,Value=v2},{Key=k1,Value=v1}]"
```

以下は出力の例です。

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "AssociationTarget": {
            "InstanceIds": [],
            "Tags": [
                {
                    "Key": "k2",
                    "Value": "v2"
                },
                {
                    "Key": "k1",
                    "Value": "v1"
                }
            ],
            "DedicatedHostIds": []
        },
        "State": "creating"
    }
}
```

イベントウィンドウとインスタンス ID を関連付けるには

associate-instance-event-window コマンドを使用します。

```
aws ec2 associate-instance-event-window \
    --instance-event-window-id iew-0abcdef1234567890 \
    --association-target "InstanceIds=i-1234567890abcdef0,i-0598c7d356eba48d7"
```

以下は出力の例です。

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "AssociationTarget": {
            "InstanceIds": [
                "i-1234567890abcdef0",
                "i-0598c7d356eba48d7"
            ],
            "Tags": [],
            "DedicatedHostIds": []
        },
        "State": "creating"
    }
}
```

専有ホストとイベントウィンドウを関連付けるには

associate-instance-event-window コマンドを使用します。

```
aws ec2 associate-instance-event-window \
    --instance-event-window-id iew-0abcdef1234567890 \
    --association-target "DedicatedHostIds=h-029fa35a02b99801d"
```

以下は出力の例です。

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "AssociationTarget": {
            "InstanceIds": [],
            "Tags": [],
            "DedicatedHostIds": [
                "h-029fa35a02b99801d"
            ]
        },
        "State": "creating"
```

```
    }
}
```

PowerShell

イベントウィンドウとインスタンスタグを関連付けるには

Register-EC2InstanceEventWindow コマンドレットを使用します。

```
$tag1 = @{Key="key1"; Value="value1"}
$tag2 = @{Key="key2"; Value="value2"}
Register-EC2InstanceEventWindow `
    -InstanceEventWindowId iew-0abcdef1234567890 `
    -AssociationTarget_InstanceTag @($tag1,$tag2)
```

イベントウィンドウとインスタンス ID を関連付けるには

Register-EC2InstanceEventWindow コマンドレットを使用します。

```
Register-EC2InstanceEventWindow `
    -InstanceEventWindowId iew-0abcdef1234567890 `
    -AssociationTarget_InstanceId i-1234567890abcdef0, i-0598c7d356eba48d7
```

専有ホストとイベントウィンドウを関連付けるには

Register-EC2InstanceEventWindow コマンドレットを使用します。

```
Register-EC2InstanceEventWindow `
    -InstanceEventWindowId iew-0abcdef1234567890 `
    -AssociationTarget_DedicatedHostId h-029fa35a02b99801d
```

イベントウィンドウからターゲットの関連付けを解除する

Console

イベントウィンドウとターゲットの関連付けを解除するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインの [Events] を選択してください。

3. 変更するイベントウィンドウを選択します。
4. [アクション]、[インスタンスイベントウィンドウを変更] の順に選択します。

AWS CLI

イベントウィンドウからインスタンスタグの関連付けを解除するには

disassociate-instance-event-window コマンドを使用します。

```
aws ec2 disassociate-instance-event-window \
    --instance-event-window-id iew-0abcdef1234567890 \
    --association-target "InstanceTags=[{Key=k2,Value=v2},{Key=k1,Value=v1}]"
```

以下は出力の例です。

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "AssociationTarget": {
            "InstanceIds": [],
            "Tags": [],
            "DedicatedHostIds": []
        },
        "State": "creating"
    }
}
```

イベントウィンドウからインスタンス ID の関連付けを解除するには

disassociate-instance-event-window コマンドを使用します。

```
aws ec2 disassociate-instance-event-window \
    --instance-event-window-id iew-0abcdef1234567890 \
    --association-target "InstanceIds=i-1234567890abcdef0,i-0598c7d356eba48d7"
```

以下は出力の例です。

```
{
```

```
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "AssociationTarget": {
            "InstanceIds": [],
            "Tags": [],
            "DedicatedHostIds": []
        },
        "State": "creating"
    }
}
```

専有ホストとイベントウィンドウとの関連付けを解除するには

disassociate-instance-event-window コマンドを使用します。

```
aws ec2 disassociate-instance-event-window \
    --instance-event-window-id iew-0abcdef1234567890 \
    --association-target DedicatedHostIds=h-029fa35a02b99801d
```

以下は出力の例です。

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "AssociationTarget": {
            "InstanceIds": [],
            "Tags": [],
            "DedicatedHostIds": []
        },
        "State": "creating"
    }
}
```

PowerShell

イベントウィンドウからインスタンスタグの関連付けを解除するには

Unregister-EC2InstanceEventWindow コマンドレットを使用します。

```
$tag1 = @{Key="key1"; Value="value1"}
$tag2 = @{Key="key2"; Value="value2"}
Unregister-EC2InstanceEventWindow `
    -InstanceEventWindowId iew-0abcdef1234567890 `
    -AssociationTarget_InstanceTag @($tag1, $tag2)
```

イベントウィンドウからインスタンス ID の関連付けを解除するには

Unregister-EC2InstanceEventWindow コマンドレットを使用します。

```
Unregister-EC2InstanceEventWindow `
    -InstanceEventWindowId iew-0abcdef1234567890 `
    -AssociationTarget_InstanceId i-1234567890abcdef0, i-0598c7d356eba48d7
```

専有ホストとイベントウィンドウとの関連付けを解除するには

Unregister-EC2InstanceEventWindow コマンドレットを使用します。

```
Unregister-EC2InstanceEventWindow `
    -InstanceEventWindowId iew-0abcdef1234567890 `
    -AssociationTarget_DedicatedHostId h-029fa35a02b99801d
```

イベントウィンドウを変更する

イベントウィンドウに関してはその ID 以外のすべてのフィールドを変更できます。例えば、夏時間の開始時に、イベントウィンドウのスケジュールを変更できます。既存のイベントウィンドウに対してはターゲットの追加または削除が必要になることもあります。

イベントウィンドウでは時間範囲または cron 式のいずれかの変更が可能ですが、両方を変更することはできません。

Console

イベントウィンドウを変更するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインの [Events] を選択してください。
3. [Actions] (アクション)、[Manage event windows] (イベントウィンドウの管理) を選択してください。

4. 変更するイベントウィンドウを選択し、そして[Actions] (アクション)、[Modify instance event window] (インスタンスイベントウィンドウの変更) を選択してください。
5. イベントウィンドウの各フィールドを変更した後、[イベントウィンドウの変更] をクリックしてください。

AWS CLI

イベントウィンドウの時間範囲を変更するには

modify-instance-event-window コマンドを使用します。

```
aws ec2 modify-instance-event-window
    --instance-event-window-id iew-0abcdef1234567890 \
    --time-range StartWeekDay=monday,StartHour=2,EndWeekDay=wednesday,EndHour=8
```

以下は出力の例です。

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "TimeRanges": [
            {
                "StartWeekDay": "monday",
                "StartHour": 2,
                "EndWeekDay": "wednesday",
                "EndHour": 8
            }
        ],
        "Name": "myEventWindowName",
        "AssociationTarget": {
            "InstanceIds": [
                "i-0abcdef1234567890",
                "i-0be35f9acb8ba01f0"
            ],
            "Tags": [],
            "DedicatedHostIds": []
        },
        "State": "creating",
        "Tags": [
            {
                "Key": "K1",
```

```
                "Value": "V1"
            }
        ]
    }
}
```

イベントウィンドウの時間範囲のセットを変更するには

modify-instance-event-window コマンドを使用します。

```
aws ec2 modify-instance-event-window
    --instance-event-window-id iew-0abcdef1234567890 \
    --time-range '[{"StartWeekDay": "monday", "StartHour": 2, "EndWeekDay":
 "wednesday", "EndHour": 8},
   {"StartWeekDay": "thursday", "StartHour": 2, "EndWeekDay": "friday",
 "EndHour": 8}]'
```

以下は出力の例です。

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "TimeRanges": [
            {
                "StartWeekDay": "monday",
                "StartHour": 2,
                "EndWeekDay": "wednesday",
                "EndHour": 8
            },
            {
                "StartWeekDay": "thursday",
                "StartHour": 2,
                "EndWeekDay": "friday",
                "EndHour": 8
            }
        ],
        "Name": "myEventWindowName",
        "AssociationTarget": {
            "InstanceIds": [
                "i-0abcdef1234567890",
                "i-0be35f9acb8ba01f0"
            ],
            "Tags": [],
```

```
            "DedicatedHostIds": []
        },
        "State": "creating",
        "Tags": [
            {
                "Key": "K1",
                "Value": "V1"
            }
        ]
    }
}
```

イベントウィンドウの cron 式を変更するには

modify-instance-event-window コマンドを使用します。

```
aws ec2 modify-instance-event-window
    --instance-event-window-id iew-0abcdef1234567890 \
    --cron-expression "* 21-23 * * 2,3"
```

以下は出力の例です。

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "AssociationTarget": {
            "InstanceIds": [
                "i-0abcdef1234567890",
                "i-0be35f9acb8ba01f0"
            ],
            "Tags": [],
            "DedicatedHostIds": []
        },
        "State": "creating",
        "Tags": [
            {
                "Key": "K1",
                "Value": "V1"
            }
        ]
    }
```

```
}
```

PowerShell

イベントウィンドウの時間範囲を変更するには

Edit-EC2InstanceEventWindow コマンドレットを使用します。

```
$timeRange1 = New-Object Amazon.EC2.Model.InstanceEventWindowTimeRangeRequest
$timeRange1.StartWeekDay = "monday"
$timeRange1.EndWeekDay = "wednesday"
$timeRange1.StartHour = 2
$timeRange1.EndHour = 8
$timeRange2 = New-Object Amazon.EC2.Model.InstanceEventWindowTimeRangeRequest
$timeRange2.StartWeekDay = "thursday"
$timeRange2.EndWeekDay = "friday"
$timeRange2.StartHour = 1
$timeRange2.EndHour = 6
Edit-EC2InstanceEventWindow `
    -InstanceEventWindowId iew-0abcdef1234567890 `
    -TimeRange @($timeRange1, $timeRange2)
```

イベントウィンドウの cron 式を変更するには

Edit-EC2InstanceEventWindow コマンドレットを使用します。

```
Edit-EC2InstanceEventWindow `
    -InstanceEventWindowId iew-0abcdef1234567890 `
    -CronExpression "* 21-23 * * 2,3"
```

イベントウィンドウを削除する

一度に 1 つのイベントウィンドウを削除できます。

Console

イベントウィンドウを削除するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインの [Events] を選択してください。
3. [アクション]、[イベントウィンドウの管理] の順にクリックしてください。

4. 削除するイベントウィンドウを選択し、[アクション]、[インスタンスのイベントウィンドウの削除] の順にクリックしてください。
5. 確認を求めるメッセージが表示されたら、**delete**と入力し、[削除] を選択してください。

AWS CLI

イベントウィンドウを削除するには

削除するイベントウィンドウを指定しながら、delete-instance-event-window コマンドを実行します。

```
aws ec2 delete-instance-event-window \
    --instance-event-window-id iew-0abcdef1234567890
```

イベントウィンドウを強制的に削除するには

現在、イベントウィンドウがターゲットに関連付けられている場合には--force-delete パラメータを使用します。

```
aws ec2 delete-instance-event-window \
    --instance-event-window-id iew-0abcdef1234567890 \
    --force-delete
```

PowerShell

イベントウィンドウを削除するには

Remove-EC2InstanceEventWindow コマンドレットを使用します。

```
Remove-EC2InstanceEventWindow `
    -InstanceEventWindowId iew-0abcdef1234567890
```

イベントウィンドウを強制的に削除するには

Remove-EC2InstanceEventWindow コマンドレットを使用します。

```
Remove-EC2InstanceEventWindow `
    -InstanceEventWindowId iew-0abcdef1234567890 `
    -ForceDelete $true
```

CloudWatch を使用したインスタンスのモニタリング

Amazon CloudWatch を使用してインスタンスをモニタリングすることで、Amazon EC2 から未加工データを収集し、リアルタイムに近い読み取り可能なメトリクスに加工することができます。これらの統計は 15 か月間記録されるため、履歴情報にアクセスしてウェブアプリケーションやサービスの動作をより的確に把握できます。

デフォルトでは Amazon EC2 は 5 分ごとにメトリクスデータを CloudWatch に送信します。1 分ごとにインスタンスのメトリクスデータを CloudWatch に送信するにはインスタンスで詳細モニタリングを有効にできます。詳細については「EC2 インスタンスの詳細モニタリングを管理する」を参照してください。

Amazon EC2 コンソールには、Amazon CloudWatch の未加工データに基づいて一連のグラフが表示されます。必要に応じて、コンソールのグラフではなく Amazon CloudWatch からインスタンスのデータを取得することもできます。

Amazon CloudWatch の請求とコストに関する情報については、「Amazon CloudWatch ユーザーガイド」の「CloudWatch の請求とコスト」を参照してください。

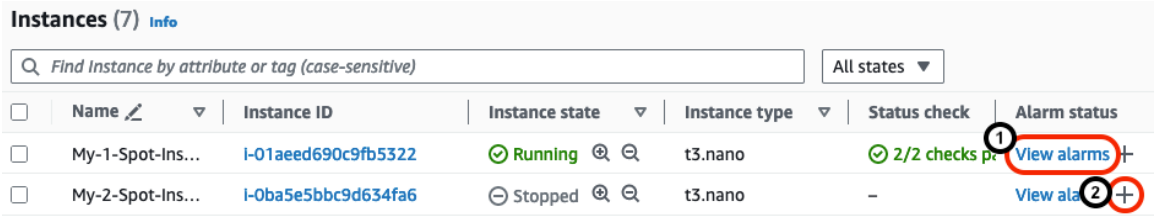
内容

- Amazon EC2 コンソールで EC2 インスタンスの CloudWatch アラームを管理する
- EC2 インスタンスの詳細モニタリングを管理する
- インスタンスに対して利用可能な CloudWatch メトリクス
- Amazon EC2 コンソールを使用して CloudWatch エージェントをインストールおよび設定し、メトリクスを追加する
- インスタンスに対する CloudWatch メトリクスの統計
- インスタンスのモニタリンググラフを表示する
- インスタンスの CloudWatch アラームを作成する
- インスタンスを停止、終了、再起動、または復旧するアラームを作成する

Amazon EC2 コンソールで EC2 インスタンスの CloudWatch アラームを管理する

Amazon EC2 コンソールの [インスタンス] 画面から、インスタンスのための Amazon CloudWatch アラームを管理できます。[インスタンス] テーブルの [アラームステータス] 列にはアラームを表示

するためのコントロールと、アラームを作成または編集するためのコントロールの 2 つのコンソールコントロールがあります。次のスクリーンショットは、[1] ([アラームの表示]) と [2] (アラームを作成または編集するための [+] 記号) の番号が付されたこれらのコンソールコントロールを示しています。



インスタンス画面からアラームを表示する

[インスタンス] 画面から各インスタンスのアラームを表示できます。

インスタンス画面からインスタンスのアラームを表示する方法

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. [インスタンス] テーブルで、選択したインスタンスの [アラームの表示] (前のスクリーンショットでの [1] 番) を選択してください。
4. [***i-1234567890abcdef0*** のアラーム詳細] ウィンドウで、CloudWatch コンソールにアラームを表示させるアラームの名前を選択します。

インスタンス画面からアラームを作成する

[インスタンス]画面からインスタンスごとにアラームを作成できます。

インスタンス画面からインスタンスにアラームを作成する方法

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. [インスタンス] テーブルで、選択したインスタンスに対してプラス記号 (前のスクリーンショットでの [2] 番) を選択してください。
4. [CloudWatch アラームの管理] 画面で、アラームを作成します。詳細については「インスタンスの CloudWatch アラームを作成する」を参照してください。

インスタンス画面からアラームを編集する

[インスタンス]画面からインスタンスのためのアラームを編集できます。

インスタンス画面からインスタンスにアラームを編集するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. [インスタンス] テーブルで、選択したインスタンスに対してプラス記号 (前のスクリーンショットでの [2] 番) を選択してください。
4. [CloudWatch アラームの管理] 画面で、アラームを編集します。詳細については、「Amazon CloudWatch ユーザーガイド」の「CloudWatch アラームの編集または削除」を参照してください。

EC2 インスタンスの詳細モニタリングを管理する

Amazon CloudWatch には基本モニタリングと詳細モニタリングの 2 つのカテゴリのモニタリングが用意されています。デフォルトではインスタンスは基本モニタリングを使用するように設定されています。オプションで詳細モニタリングを有効にすることで、運用上の問題をより迅速に特定して対処できるようにすることができます。インスタンスの起動時、実行中、または停止中に、詳細モニタリングを有効にしたり無効にしたりできます。

インスタンスで詳細モニタリングを有効にしても、そのインスタンスにアタッチされている EBS ボリュームのモニタリングには影響しません。詳細については「Amazon EBS の Amazon CloudWatch メトリクス」を参照してください。

次のテーブルはインスタンスの基本モニタリングと詳細モニタリングの違いを示しています。

モニタリングタイプ	説明	料金
基本モニターリング	ステータスチェックメトリクスは 1 分間隔で利用できます。その他のメトリクスはすべて 5 分間隔で利用できます。	料金は発生しません。

モニタリングタイプ	説明	料金
詳細モニターリング	インスタンスの詳細モニタリングが有効化されている場合は、1 分間隔でメトリクスを取得できます。 詳細モニタリングを有効にすると、同様のインスタンスグループ全体のデータを集計できます。	料金はAmazon EC2 が CloudWatch に送信したメトリクスごとに発生します。データストレージに対しては料金が発生しません。詳細については、「Amazon CloudWatch 料金表」ページの「有料利用枠」を参照してください。

内容

- 必要なアクセス許可
- 起動時に詳細モニタリングを有効にする
- 詳細モニタリングを管理する

必要なアクセス許可

インスタンスの詳細モニタリングを有効にするにはユーザーに MonitorInstances API アクションを使用するための許可が必要です。インスタンスの詳細モニタリングをオフにするにはユーザーに UnmonitorInstances API アクションを使用するための許可が必要です。

起動時に詳細モニタリングを有効にする

起動時に詳細モニタリングを有効にするには次の手順を使用します。デフォルトではインスタンスでは基本モニタリングが使用されます。

Console

インスタンスの起動時に詳細モニタリングを有効にするには

Amazon EC2 コンソールを使用してインスタンスを起動するとき、[高度な詳細] で、[CloudWatch モニタリングの詳細] チェックボックスを選択してください。

AWS CLI

インスタンスの起動時に詳細モニタリングを有効にするには

`--monitoring` オプションで run-instances コマンドを使用します。

```
--monitoring Enabled=true
```

PowerShell

インスタンスの起動時に詳細モニタリングを有効にするには

-Monitoring パラメータで New-EC2Instance コマンドレットを使用します。

```
-Monitoring $true
```

詳細モニタリングを管理する

実行中または停止中のインスタンスの詳細モニタリングを管理するには次の手順を使用します。

Console

詳細モニタリングを管理するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択してください。
4. [アクション]、[モニタリングとトラブルシューティング]、[詳細モニタリングの管理] の順に選択してください。
5. [詳細モニタリング] ページの [詳細モニタリング] で、次のいずれかを実行します。
   - 詳細モニタリング – [有効] を選択してください。
   - 基本モニタリング — [有効] をクリアします。
6. [確認] を選択してください。

AWS CLI

詳細モニタリングを有効にするには

次の monitor-instances コマンドを使用します。

```
aws ec2 monitor-instances --instance-ids i-1234567890abcdef0
```

詳細モニタリングを無効にするには

unmonitor-instances コマンドを使用します。

```
aws ec2 unmonitor-instances --instance-ids i-1234567890abcdef0
```

PowerShell

詳細モニタリングを有効にするには

Start-EC2InstanceMonitoring コマンドレットを使用します。

```
Start-EC2InstanceMonitoring -InstanceId i-1234567890abcdef0
```

詳細モニタリングを無効にするには

Stop-EC2InstanceMonitoring コマンドレットを使用します。

```
Stop-EC2InstanceMonitoring -InstanceId i-1234567890abcdef0
```

インスタンスに対して利用可能な CloudWatch メトリクス

Amazon EC2 はメトリクスを Amazon CloudWatch に送信します。AWS マネジメントコンソール、AWS CLI、または API を使用して、Amazon EC2 が CloudWatch に送信するメトリクスを一覧表示できます。デフォルトで、各データポイントではインスタンスのアクティビティの開始後 5 分間が対象となります。詳細モニタリングを有効にした場合、各データポイントは開始後 1 分間のアクティビティを対象とします。注意事項[最小]、[最大]、[平均] の統計ではEC2 が提供するメトリックスの最小粒度は 1 分であることに注意します。

AWS マネジメントコンソール または AWS CLI を使用して利用可能なメトリクスを表示する方法の詳細については,「Amazon CloudWatch ユーザーガイド」の「利用可能なメトリクスを表示する」を参照してください。

これらのメトリクスの統計の取得については インスタンスに対する CloudWatch メトリクスの統計を参照してください。

内容

- インスタンスメトリクス
- アクセラレーターメトリクス
- CPU クレジットメトリクス
- 専用ホストメトリクス

- Nitro ベースのインスタンスの Amazon EBS メトリクス
- ステータスチェックメトリクス
- トラフィックミラーリングのメトリクス
- Auto Scaling グループメトリクス
- Amazon EC2 メトリクスディメンション
- Amazon EC2 使用状況メトリクス

インスタンスメトリクス

AWS/EC2 名前空間には次のインスタンスメトリクスが含まれます。

メトリクス	説明	Unit	有意義な統計
CPUUtilization	Amazon EC2 が EC2 インスタンスを実行するために使用する物理 CPU 時間の割合。これにはユーザーコードと Amazon EC2 コードの両方を実行するために費やされた時間が含まれます。 非常に高いレベルではCPUUtilization はゲスト CPUUtilization とハイパーバイザー CPUUtilization の合計です。 オペレーティングシステムのツールは CloudWatch と異なる割合を表示することがあります。これはレガシーデバイスのシミュレーション、レガシーではないデバイスの設定、中断の多いワークロード、ライブ移行、ライブアップデートなどが原因です。	Percent	• Average • Minimum • Maximum
DiskReadOps	指定された期間にインスタンスで利用できるすべてのインスタンスストアボリュームでの、完了した読み取り操作。 その期間の 1 秒あたりの I/O 操作回数 (IOPS) の平均を算出するにはその期間の操作回数の合計をその期間の秒数で割ります。	Count	• Sum • Average • Minimum • Maximum

メトリクス	説明	Unit	有意義な統計
	インスタンスストアボリュームがない場合は、値が 0 であるか、メトリクスがレポートされません。		
DiskWriteOps	指定された期間にインスタンスで利用できるすべてのインスタンスストアボリュームへの、完了した書き込み操作。 その期間の 1 秒あたりの I/O 操作回数 (IOPS) の平均を算出するにはその期間の操作回数の合計をその期間の秒数で割ります。 インスタンスストアボリュームがない場合は、値が 0 であるか、メトリクスがレポートされません。	Count	• Sum • Average • Minimum • Maximum

メトリクス	説明	Unit	有意義な統計
DiskReadBytes	インスタンスで利用できるすべてのインスタンスストアボリュームから読み取られたバイト数。 このメトリクスを使用すると、このインスタンスのハードディスクからアプリケーションが読み取るデータの量がわかります。これを利用すると、アプリケーションの速度がわかります。 報告された数は期間中に受信されたバイト数です。基本 (5 分) モニタリングを使用している場合、この数を 300 で除算してバイト/秒を求めることができます。詳細 (1 分) モニタリングを使用している場合はこの数を 60 で除算します。CloudWatch メトリクスの計算関数 DIFF_TIME を使用して、1 秒あたりのバイト数を求めることもできます。例えば、CloudWatch で DiskReadBytes のグラフを m1 として作成した場合、メトリクスの数式 m1/(DIFF_TIME(m1)) はメトリクスをバイト/秒単位で返します。DIFF_TIME およびメトリクス計算関数の詳細については、「Amazon CloudWatch ユーザーガイド」の「メトリクス数式の使用」を参照してください。 インスタンスストアボリュームがない場合は値が 0 であるか、メトリクスがレポートされません。	Bytes	• Sum • Average • Minimum • Maximum

メトリクス	説明	Unit	有意義な統計
DiskWriteBytes	インスタンスで利用できるすべてのインスタンスストアボリュームに書き込まれたバイト数。 このメトリクスを使用すると、このインスタンスのハードディスクにアプリケーションが書き込むデータの量がわかります。これを利用すると、アプリケーションの速度がわかります。 報告された数は期間中に受信されたバイト数です。基本 (5 分) モニタリングを使用している場合、この数を 300 で除算してバイト/秒を求めることができます。詳細 (1 分) モニタリングを使用している場合はこの数を 60 で除算します。CloudWatch メトリクスの計算関数 DIFF_TIME を使用して、1 秒あたりのバイト数を求めることもできます。例えば、CloudWatch で DiskWriteBytes のグラフを m1 として作成した場合、メトリクスの数式 m1/(DIFF_TIME(m1)) はメトリクスをバイト/秒単位で返します。DIFF_TIME およびメトリクス計算関数の詳細については、「Amazon CloudWatch ユーザーガイド」の「メトリクス数式の使用」を参照してください。 インスタンスストアボリュームがない場合は値が 0 であるか、メトリクスがレポートされません。	Bytes	• Sum • Average • Minimum • Maximum

メトリクス	説明	Unit	有意義な統計
MetadataNoToken	トークンを使用しないメソッドを使用してインスタンスメタデータサービス (IMDS) に正常にアクセスした回数。 このメトリクスにより、トークンを使用しないインスタンスメタデータサービスバージョン 1 (IMDSv1) を使用してインスタンスメタデータにアクセスするプロセスがあるかどうかがわかります。すべてのリクエストがトークン支援のセッション (インスタンスメタデータサービスバージョン 2 (IMDSv2)) を使用している場合、値は 0 になります。詳細については「インスタンスメタデータサービスバージョン 2 の使用への移行」を参照してください。	Count	• Sum • Percentiles
MetadataNoTokenRejected	IMDSv1 が無効になった後に IMDSv1 呼び出しが試行された回数。 このメトリクスが表示された場合はIMDSv1 呼び出しが試行され、拒否されたことを示します。IMDSv1 を再度有効にするか、すべての呼び出しで IMDSv2 が使用されていることを確認します。詳細については「インスタンスメタデータサービスバージョン 2 の使用への移行」を参照してください。	Count	• Sum • Percentiles

メトリクス	説明	Unit	有意義な統計
NetworkIn	すべてのネットワークインターフェイスを通じ、このインスタンスによって受信されたバイトの数。このメトリクスは 1 つのインスタンスへの受信ネットワークトラフィックの量を表しています。 報告された数は期間中に受信されたバイト数です。基本 (5 分) のモニタリングで統計情報に 合計 使用している場合であれば、この数を 300 で除算してバイト/秒の値を求めることができます。詳細 (1 分) のモニタリングで統計情報に 合計 使用している場合合はこの数を 60 で除算します。	Bytes	• Sum • Average • Minimum • Maximum
NetworkOut	すべてのネットワークインターフェイスを通じ、このインスタンスから送信されたバイトの数。このメトリクスは 1 つのインスタンスからの送信ネットワークトラフィックの量を表しています。 報告された数は期間中に送信されたバイト数です。基本 (5 分) のモニタリングで統計情報に 合計 使用している場合であれば、この数を 300 で除算してバイト/秒の値を求めることができます。詳細 (1 分) のモニタリングで統計情報に 合計 使用している場合合はこの数を 60 で除算します。	Bytes	• Sum • Average • Minimum • Maximum

メトリクス	説明	Unit	有意義な統計
NetworkPacketsIn	すべてのネットワークインターフェイスを通じ、このインスタンスによって受信されたパケットの数。このメトリクスは受信トラフィックのボリュームを単一インスタンスでのパケット数として識別します。 このメトリクスは基本モニタリング (5分間) でのみ使用が可能です。5 分間でインスタンスが受信した 1 秒あたりのパケット数 (PPS) を計算するには、Sum の統計値を 300 で割ることで算出されます。	Count	• Sum • Average • Minimum • Maximum
NetworkPacketsOut	すべてのネットワークインターフェイスを通じ、このインスタンスから送信されたパケットの数。このメトリクスは送信トラフィックのボリュームを単一インスタンスでのパケット数として識別します。 このメトリクスは基本モニタリング (5分間) でのみ使用が可能です。5 分間にインスタンスが受信した 1 秒あたりのパケット数 (PPS) を計算するには合計 統計値を 300 で割ります。	Count	• Sum • Average • Minimum • Maximum

アクセラレーターメトリクス

AWS/EC2 名前空間には、高速コンピューティングインスタンス向けの以下のアクセラレーターメトリクスが含まれています。このメトリクスは、高速コンピューティングインスタンスタイプのサブセットのみでサポートされます。

メトリクス	説明	Unit	有意義な統計
GPUPowerUtilization	最大有効電力の割合 (%) としての有効電力使用量。	割合 (%)	• Average • Minimum • Maximum

CPU クレジットメトリクス

AWS/EC2 名前空間はバーストパフォーマンスインスタンスの以下の CPU クレジットメトリクスを含みます。

メトリクス	説明	Unit	有意義な統計
CPUCreditUsage	CPU 使用率に関してインスタンスで消費される CPU クレジットの数。1 つの CPU クレジットは1 個の vCPU が 100% の使用率で 1 分間実行されること、またはvCPU、使用率、時間の同等の組み合わせ (例えば、1 個の vCPU が 50% の使用率で 2 分間実行されるか、2 個の vCPU が 25% の使用率で 2 分間実行される) に相当します。 CPU クレジットメトリクスは5 分間隔でのみ利用可能です。5 分を超える期間を指定する場合はAverage 統計の代わりに Sum 統計を使用します。	クレジット (vCPU 分)	• Sum • Average • Minimum • Maximum
CPUCreditBalance	インスタンスが起動または開始後に蓄積した獲得 CPU クレジットの数。T2 スタンダードの場合、CPUCreditBalance には蓄積された起動クレジットの数も含まれます。 クレジットは獲得後にクレジット残高に蓄積され、消費されるとクレジット残高から削除されます。クレジット残高にはインスタンスサイズによって決まる上限があります。制限に到達すると、獲得された新しいクレジットはすべて破棄されます。T2 スタンダードの場合、起動クレジットは制限に対してカウントされません。 CPUCreditBalance のクレジットはインスタンスがそのベースライン CPU 使用率を超えてバーストするために消費できます。	クレジット (vCPU 分)	• Sum • Average • Minimum • Maximum

メトリクス	説明	Unit	有意義な統計
	インスタンスが実行中の場合、CPUCredit Balance のクレジットは期限切れになりません。T3 または T3a インスタンスが停止すると、CPUCreditBalance 値は 7 日間保持されます。その後、蓄積されたすべてのクレジットが失われます。T2 インスタンスが停止すると、CPUCreditBalance 値は保持されず、蓄積されたすべてのクレジットが失われます。 CPU クレジットメトリクスは5 分間隔でのみ利用可能です。		
CPUSurplusCreditBalance	CPUCreditBalance 値がゼロの場合に unlimited インスタンスによって消費された余剰クレジットの数。 CPUSurplusCreditBalance 値は獲得した CPU クレジットによって支払われます。余剰クレジットの数が、24 時間にインスタンスが獲得できるクレジットの最大数を超えている場合、最大数を超えて消費された余剰クレジットに対しては料金が発生します。 CPU クレジットメトリクスは5 分間隔でのみ利用可能です。	クレジット (vCPU 分)	• Sum • Average • Minimum • Maximum

メトリクス	説明	Unit	有意義な統計
CPUSurplusCreditsCharged	獲得 CPU クレジットにより支払われないために追加料金が発生した、消費された余剰クレジットの数。 消費された余剰クレジットは以下のいずれかの状況に当てはまると料金が発生します。 • 消費された余剰クレジットが、インスタンスが 24 時間に獲得できる最大クレジット数を超えている。最大数を越えて消費された余剰クレジットは時間の最後に課金されます。 • インスタンスが停止または終了した。 • インスタンスは unlimited から standard に切り替わります。 CPU クレジットメトリクスは5 分間隔でのみ利用可能です。	クレジット (vCPU 分)	• Sum • Average • Minimum • Maximum

専用ホストメトリクス

AWS/EC2 名前空間にはT3 専用ホスト のための以下のメトリクスが含まれます。

メトリクス	説明	Unit	有意義な統計
DedicatedHostCPUUtilization	Dedicated Host で実行されているインスタンスによって現在使用されている割り当て済みコンピューティング容量の割合。	Percent	• Sum • Average • Minimum • Maximum

Nitro ベースのインスタンスの Amazon EBS メトリクス

AWS/EC2 名前空間にはベアメタルインスタンスではない、Nitro ベースのインスタンスにアタッチされているボリュームに関する、追加の Amazon EBS メトリクスが含まれます。

メトリクス	説明	Unit	有意義な統計
InstanceEBSIOPSExceededCheck	過去 1 分以内にアプリケーションがインスタンスの最大 EBS IOPS 制限を超える IOPS を駆動しようとしたかどうかをレポートします。このメトリクスは、0 (IOPS 制限を超えていない) または 1 (IOPS 制限を超えている) のいずれの値をとります。	なし	• 合計 • 平均 • 最小値 • 最大値
InstanceEBSThroughputExceededCheck	過去 1 分以内にアプリケーションがインスタンスの最大 EBS スループット制限を超えるスループットを駆動しようとしたかどうかをレポートします。このメトリクスは、0 (スループット制限を超えていない) または 1 (スループット制限を超えている) のいずれかの値をとります。	なし	• 合計 • 平均 • 最小値 • 最大値
EBSReadOps	指定された期間にインスタンスに接続されたすべての Amazon EBS ボリュームからの、完了した読み込みオペレーション。 その期間の 1 秒あたりの読み込み I/O 操作回数 (読み込み IOPS) の平均を算出するにはその期間の操作回数の合計をその期間の秒数で割ります。基本 (5 分) モニタリングを使用している場合、この数を 300 で除算して読み込み IOPS を計算することができます。詳細 (1 分) モニタリングを使用している場合はこの数を 60 で除算します。CloudWatch メトリクスの計算関数 DIFF_TIME を使用して、1 秒あたりのオペレーション数を求めることもできます。例えば、CloudWatch で EBSReadOps のグラフを m1 として作成した場合、メトリクスの計算関数 m1/(DIFF_TIME(m1)) はメトリクスをオペレーション/秒単位で返します。DIFF_TIME およびその他のメトリクス計算関数の詳細については、「Amazon	カウント	• 合計 • 平均 • 最小値 • 最大値

メトリクス	説明	Unit	有意義な統計
	CloudWatch ユーザーガイド」の「CloudWatch メトリクスでの数式の使用」を参照してください。		
EBSWriteOps	指定された期間にインスタンスに接続されたすべての EBS ボリュームからの、完了した書き込み操作。 その期間の 1 秒あたりの書き込み I/O 操作回数 (書き込み IOPS) の平均を算出するにはその期間の操作回数の合計をその期間の秒数で割ります。基本 (5 分) モニタリングを使用している場合、この数を 300 で除算して書き込み IOPS を計算することができます。詳細 (1 分) モニタリングを使用している場合はこの数を 60 で除算します。CloudWatch メトリクスの計算関数 DIFF_TIME を使用して、1 秒あたりのオペレーション数を求めることもできます。例えば、CloudWatch で EBSWriteOps のグラフを m1 として作成した場合、メトリクスの計算関数 m1/(DIFF_TIME(m1)) はメトリクスをオペレーション/秒単位で返します。DIFF_TIME およびその他のメトリクス計算関数の詳細については、「Amazon CloudWatch ユーザーガイド」の「CloudWatch メトリクスでの数式の使用」を参照してください。	カウント	• 合計 • 平均 • 最小値 • 最大値

メトリクス	説明	Unit	有意義な統計
EBSReadBytes	指定した期間内にインスタンスに接続されたすべての EBS ボリュームから読み取られたバイト数。 報告された数は期間中に読み取られたバイト数です。基本 (5 分) モニタリングを使用している場合、この数を 300 で除算して読み込みバイト/秒を求めることができます。詳細 (1 分) モニタリングを使用している場合はこの数を 60 で除算します。CloudWatch メトリクスの計算関数 DIFF_TIME を使用して、1 秒あたりのバイト数を求めることもできます。例えば、CloudWatch で EBSReadBytes のグラフを m1 として作成した場合、メトリクスの数式 m1/(DIFF_TIME(m1)) はメトリクスをバイト/秒単位で返します。DIFF_TIME およびメトリクス計算関数の詳細については、「Amazon CloudWatch ユーザーガイド」の「メトリクス数式の使用」を参照してください。	バイト	• 合計 • 平均 • 最小値 • 最大値

メトリクス	説明	Unit	有意義な統計
EBSWriteBytes	指定した期間内にインスタンスに接続されたすべての EBS ボリュームに書き込まれたバイト数。 報告された数は期間中に書き込まれたバイト数です。基本 (5 分) モニタリングを使用している場合、この数を 300 で除算して書き込みバイト/秒を求めることができます。詳細 (1 分) モニタリングを使用している場合はこの数を 60 で除算します。CloudWatch メトリクスの計算関数 DIFF_TIME を使用して、1 秒あたりのバイト数を求めることもできます。例えば、CloudWatch で EBSWriteBytes のグラフを m1 として作成した場合、メトリクスの数式 m1/(DIFF_TIME(m1)) はメトリクスをバイト/秒単位で返します。DIFF_TIME およびメトリクス計算関数の詳細については、「Amazon CloudWatch ユーザーガイド」の「メトリクス数式の使用」を参照してください。	バイト	• 合計 • 平均 • 最小値 • 最大値
EBSIOBalance%	バーストバケットの I/O 残りクレジットの割合に関する情報を提供します。このメトリクスは基本モニタリング専用です。 このメトリクスは少なくとも 24 時間に 1 回、30 分間だけ最大パフォーマンスにバーストする一部の *.4xlarge インスタンスサイズ以下でのみ使用できます。 Sum 統計はこのメトリクスに該当しません。	割合 (%)	• 最小値 • 最大値

メトリクス	説明	Unit	有意義な統計
EBSByteBalance%	バーストバケットのスループット残りクレジットの割合に関する情報を提供します。このメトリクスは基本モニタリング専用です。 このメトリクスは少なくとも 24 時間に 1 回、30 分間だけ最大パフォーマンスにバーストする一部の *.4xlarge インスタンスサイズ以下でのみ使用できます。 Sum 統計はこのメトリクスに該当しません。	割合 (%)	• 最小値 • 最大値

EBS ボリューム用のメトリクスの詳細については「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームのメトリクス」を参照してください。EC2 フリートおよびスポットフリートに提供されるメトリクスの詳細については「CloudWatch を使用して EC2 フリートまたはスポットフリートをモニタリングする」を参照してください。

ステータスチェックメトリクス

デフォルトではステータスチェックメトリクスは無料で 1 分の頻度で利用できます。新しく起動したインスタンスの場合、ステータスチェックメトリクスデータはインスタンスが初期化状態を完了するまで使用できません (インスタンスが running の状態になってから数分以内)。EC2 ステータスチェックの詳細についてはAmazon EC2 インスタンスのステータスチェックを参照してください。

AWS/EC2 名前空間には次のステータスチェックメトリクスが含まれます。

メトリクス	説明	Unit	有意義な統計
StatusCheckFailed	最近 1 分間にインスタンスがすべてのステータスチェックに成功したかどうかを報告します。 このメトリクスは 0 (合格) または 1 (失敗) となります。 デフォルトではこのメトリクスは無料で 1 分の頻度で利用できます。	Count	• Average • Minimum • Maximum

メトリクス	説明	Unit	有意義な統計
StatusCheckFailed_Instance	最近 1 分間にインスタンスがインスタンスステータスチェックに成功したかどうかを報告します。 このメトリクスは 0 (合格) または 1 (失敗) となります。 デフォルトではこのメトリクスは無料で 1 分の頻度で利用できます。	Count	• Average • Minimum • Maximum
StatusCheckFailed_System	最近 1 分間にインスタンスが システムステータスチェックに成功したかどうかを報告します。 このメトリクスは 0 (合格) または 1 (失敗) となります。 デフォルトではこのメトリクスは無料で 1 分の頻度で利用できます。	Count	• Average • Minimum • Maximum
StatusCheckFailed_AttachedEBS	直近 1 分間でインスタンスがアタッチ済みの EBS ステータスチェックに成功したかどうかを報告します。 このメトリクスは 0 (合格) または 1 (失敗) となります。 デフォルトではこのメトリクスは無料で 1 分の頻度で利用できます。	Count	• Average • Minimum • Maximum

AWS/EBS 名前空間には次のステータスチェックメトリクスが含まれます。

メトリクス	説明	Unit	有意義な統計
VolumeStalledIOCheck	注: Nitro インスタンスのみが対象です。Amazon ECS と AWS Fargate タスクにアタッチされたボリュームについては公開されていません。 ボリュームが過去 1 分間のストールした IO のチェックに合格したか失敗したかを報告します。このメトリクスは 0 (合格) または 1 (失敗) となります。	なし	• Average • Minimum • Maximum

トラフィックミラーリングのメトリクス

AWS/EC2 名前空間にはミラートラフィックのメトリクスが含まれます。詳細については「Amazon VPC トラフィックミラーリングガイド」の「Monitor mirrored traffic using Amazon CloudWatch」(Amazon CloudWatch によるミラーリングされたトラフィックのモニタリング) を参照してください。

Auto Scaling グループメトリクス

AWS/AutoScaling 名前空間にはAuto Scaling グループのメトリクスが含まれます。詳細については「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling グループとインスタンスの CloudWatch メトリクスのモニタリング」ご参照してください。

Amazon EC2 メトリクスディメンション

以下のディメンションを使用して、前の表に示したメトリクスを絞り込むことができます。

ディメンション	説明
AutoScalingGroupName	このディメンションを指定すると、リクエストしたデータがフィルタリングされて、指定したキャパシティーグループ内のインスタンスのものだけになります。Auto Scaling グループは Auto Scaling を使用する場合に定義するインスタンスのコレクションです。このディメンションを Amazon EC2 のメトリクスに対して使用できるのはインスタンスが Auto Scaling グループ

ディメンション	説明
	内にあるときに限られます。詳細モニタリングまたは基本モニタリングが有効になっているインスタンスに対して使用できます。
ImageId	このディメンションを指定すると、リクエストしたデータがフィルタリングされて、この Amazon EC2 Amazon マシンイメージ (AMI) を実行しているインスタンスのものだけになります。詳細モニタリングが有効になっているインスタンスに対して使用できます。
InstanceId	このディメンションを指定すると、リクエストしたデータがフィルタリングされて、指定のインスタンスのものだけになります。これを利用すると、どのインスタンスからのデータをモニタリングするかを指定できます。
InstanceType	このディメンションを指定すると、リクエストしたデータがフィルタリングされて、指定のインスタンスタイプで実行されているインスタンスのものだけになります。これを利用すると、実行されているインスタンスのタイプでデータを分類することができます。例えば、m1.small インスタンスと m1.large インスタンスのデータを比較して、アプリケーションに対するビジネス価値はどちらが上かを判断します。詳細モニタリングが有効になっているインスタンスに対して使用できます。

Amazon EC2 使用状況メトリクス

CloudWatch 使用状況メトリクスを使用して、アカウントのリソースの使用状況を把握できます。これらのメトリクスを使用して、CloudWatch グラフやダッシュボードで現在のサービスの使用状況を可視化できます。

Amazon EC2 使用状況メトリクスはAWS のサービスクォータに対応しています。使用量がサービスクォータに近づいたときに警告するアラームを設定することもできます。CloudWatch とサービスクオータの統合については、「Amazon CloudWatch ユーザーガイド」の「AWS 使用状況メトリクス」を参照してください。

Amazon EC2 はAWS/Usage 名前空間に以下のメトリクスを公開します。

メトリクス	説明
ResourceCount	アカウントで実行されている指定されたリソースの数。リソースはメトリクスに関連付けられたディメンションによって定義されます。 このメトリクスで最も役に立つ統計は MAXIMUM です。これは1分間の期間中に使用されるリソースの最大数を表します。

次のディメンションはAmazon EC2 によって発行される使用状況メトリクスを絞り込むために使用されます。

ディメンション	説明
Service	リソースを含む AWS のサービスの名前。Amazon EC2 使用状況メトリクスの場合、このディメンションの値は EC2 です。
Type	レポートされるエンティティのタイプ。現在、Amazon EC2 使用状況メトリクスの有効な値は Resource のみです。
Resource	実行中のリソースのタイプ。現在、Amazon EC2 使用状況メトリクスの有効な値は vCPU のみです。これは実行中のインスタンスに関する情報を返します。
Class	追跡されるリソースのクラス。vCPU ディメンションの値として Resource を使用する Amazon EC2 使用状況メトリクスの場合、有効な値はStandard/OnDemand 、F/OnDemand 、G/OnDemand 、Inf/OnDemand 、P/OnDemand 、および X/OnDemand です。 このディメンションの値はメトリクスによって報告されるインスタンスタイプの最初の文字を定義します。例えば、Standard/OnDemand はA、C、D、H、I、M、R、T、Z で始まるタイプのすべての実行中のインスタンスに関する情報を返し、G/OnDemand は G で始まるタイプのすべてのインスタンスに関する情報を返します。

Amazon EC2 コンソールを使用して CloudWatch エージェントをインストールおよび設定し、メトリクスを追加する

Amazon EC2 コンソールを使用した CloudWatch エージェントのインストールおよび設定は Amazon EC2 ではベータ版であり、今後変更される場合があります。

デフォルトで、Amazon CloudWatch は Amazon EC2 インスタンスをモニタリングするための基本的なメトリクス (`CPUUtilization`、`NetworkIn` など) を提供します。追加のメトリクスを収集するにはEC2 インスタンスに CloudWatch エージェントをインストールします。その後、選択したメトリクスを出力するようにエージェントを設定します。すべての EC2 インスタンスに CloudWatch エージェントを手動でインストールして設定するのではなく、Amazon EC2 コンソールを使用してインストールおよび設定を行えます。

Amazon EC2 コンソールを使用して、インスタンスに CloudWatch エージェントをインストールし、選択したメトリクスを出力するようにエージェントを設定できます。

このプロセスを手動で実行するときは、「Amazon CloudWatch ユーザーガイド」の「CloudWatch エージェントのインストール」を参照してください。CloudWatch エージェントの詳細については「CloudWatch エージェントを使用してメトリクス、ログ、トレースを収集する」を参照してください。

内容

- 前提条件
- 仕組み
- コスト
- CloudWatch エージェントをインストールして設定する

前提条件

Amazon EC2 を使用して CloudWatch エージェントをインストールおよび設定するにはこのセクションで説明するユーザーとインスタンスの前提条件を満たす必要があります。

> **Tip**
>
> この機能はすべての AWS リージョン で利用できるわけではありません。このページのインストール手順で説明されているメニュー項目が Amazon EC2 コンソールに存在せず、インス

タンスの実行場所について柔軟に対応できる場合は、別のリージョンをお試しください。それ以外の場合は、「Amazon CloudWatch ユーザーガイド」の手動の指示に従ってエージェントをインストールおよび設定することができます。

ユーザーの前提条件

この機能を使用するにはIAM コンソールユーザーまたはロールはAmazon EC2 を使用するために必要なアクセス許可および、次の IAM アクセス許可を持っている必要があります。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameter",
                "ssm:PutParameter"
            ],
            "Resource": "arn:aws:ssm:*:*:parameter/EC2-Custom-Metrics-*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:SendCommand",
                "ssm:ListCommandInvocations",
                "ssm:DescribeInstanceInformation"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetInstanceProfile",
                "iam:SimulatePrincipalPolicy"
            ],
            "Resource": "*"
        }
```

```
    ]
}
```

インスタンスの前提条件

- インスタンスの状態: `running`
- サポートされるオペレーティングシステム：Linux
- AWS Systems Manager Agent (SSM Agent): インストール済み。SSM エージェントに関する注意事項は 2 つあります。
  - SSM Agent はAWS および信頼できるサードパーティが提供している一部の Amazon マシンイメージ (AMI) にプリインストールされています。サポートされている AMI と SSM Agent のインストール手順については「AWS Systems Manager ユーザーガイド」の「SSMエージェントをプリインストール付 Amazon マシンイメージ (AMI) 」を参照してください。
  - SSM Agent で問題が発生した場合は「AWS Systems Manager ユーザーガイド」の「SSMエージェントのトラブルシューティング」を参照してください。
- インスタンスの IAM アクセス許可: インスタンスにアタッチされている IAM ロールには以下の AWS 管理ポリシーを追加する必要があります。
  - AmazonSSMManagedInstanceCore — インスタンスが Systems Manager を使用して CloudWatch エージェントをインストールおよび設定できるようにします。
  - CloudWatchAgentServerPolicy — インスタンスが CloudWatch エージェントを使用して CloudWatch にデータを書き込めるようにします。

  インスタンスに IAM アクセス許可を追加する方法については「IAM ユーザーガイド」の「インスタンスプロファイルの使用」を参照してください。

仕組み

Amazon EC2 コンソールを使用して CloudWatch エージェントをインストールおよび設定する前に、IAM ユーザーまたはロール、およびメトリクスを追加するインスタンスが特定の前提条件を満たしていることを確認する必要があります。その後 Amazon EC2 コンソールを使用して、選択したインスタンスで CloudWatch エージェントをインストールおよび設定できます。

まずは前提条件を満たします

- IAM アクセス許可が必要です — 開始する前に、この機能を使用するのに必要な IAM アクセス許可がコンソールのユーザーまたはロールにあることを確認してください。

- インスタンス — この機能を使用するにはEC2 インスタンスが Linux インスタンスで、SSM エージェントがインストールされており、インスタンスに必要な IAM アクセス許可がある必要があります。また、インスタンスが実行中である必要があります。

その後、機能を使用できるようになります

1. インスタンスを選択する — Amazon EC2 コンソールで、CloudWatch エージェントをインストールして設定するインスタンスを選択してください。次に、[CloudWatch エージェントの設定] を選択してプロセスを開始します。
2. SSM Agent を検証する — Amazon EC2 はSSM Agent が各インスタンスにインストールされ起動していることをチェックします。このチェックに合格しないインスタンスはプロセスから除外されます。SSM Agent はこのプロセス中にインスタンスに対してアクションを実行するために使用されます。
3. IAM アクセス許可を検証する — Amazon EC2 はこのプロセスに必要な IAM アクセス許可が各インスタンスにあることをチェックします。このチェックに合格しないインスタンスはプロセスから除外されます。IAM アクセス許可により CloudWatch エージェントでインスタンスからメトリクスを収集し、AWS Systems Manager と統合して SSM エージェントを使用できます。
4. CloudWatch エージェントを検証する — Amazon EC2 はCloudWatch エージェントが各インスタンスにインストールされ、実行されていることをチェックします。このチェックに合格しないインスタンスがある場合、Amazon EC2 がユーザーに代わって CloudWatch エージェントをインストールおよび起動します。このプロセスが完了すると、CloudWatch エージェントは各インスタンスで選択したメトリクスを収集します。
5. メトリクス設定を選択する — CloudWatch エージェントがインスタンスから出力するメトリクスを選択してください。選択すると、Amazon EC2 は設定ファイルをパラメータストアに保存します。設定ファイルはプロセスが完了するまでそこに残ります。Amazon EC2 ではプロセスが中断されない限り、パラメータストアから設定ファイルが削除されます。以前にインスタンスに追加したメトリクスを選択しない場合、このプロセスの完了後にインスタンスから削除されることに注意してください。
6. CloudWatch エージェント設定を更新する — Amazon EC2 は CloudWatch エージェントにメトリクス設定を送信します。これがプロセスの最後のステップです。成功すると、インスタンスは選択したメトリクスのデータを出力でき、Amazon EC2 はパラメータストアから設定ファイルを削除します。

コスト

このプロセス中に追加したメトリクスはカスタムメトリクスとして課金されます。CloudWatch メトリクスの料金の詳細については「Amazon CloudWatch の料金」を参照してください。

CloudWatch エージェントをインストールして設定する

Amazon EC2 コンソールを使用して CloudWatch エージェントをインストールおよび設定し、メトリクスを追加できます。

> **Note**
>
> この手順を実行するたびに、既存の CloudWatch エージェント設定を上書きします。以前に選択したメトリクスを選択しない場合、そのメトリクスはインスタンスから削除されます。

Amazon EC2 コンソールを使用して CloudWatch エージェントをインストールおよび設定するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. CloudWatch エージェントをインストールおよび設定するインスタンスを選択してください。
4. [アクション]、[モニタリングとトラブルシューティング]、[CloudWatch エージェントの設定] を選択してください。
5. プロセスの各ステップで、コンソールのテキストを読み、[次へ] を選択してください。
6. プロセスを完了するには最後のステップで [完了] を選択してください。

Amazon EC2 コンソールが作成したエージェントの設定を更新するには

EC2 コンソールが作成した設定は手動でカスタマイズできます。詳細については、「Amazon CloudWatch ユーザーガイド」の「CloudWatch エージェント設定ファイルを手動で作成または編集する」を参照してください。

インスタンスに対する CloudWatch メトリクスの統計

インスタンスの CloudWatch メトリクスの統計情報を取得できます。統計とはメトリクスデータを指定した期間で集計したものです。CloudWatch はカスタムデータまたは AWS の他のサービスから CloudWatch に提供された、メトリクスデータポイントに基づく統計を提供します。集約は指定した

期間内に、名前空間、メトリクス名、ディメンション、データポイントの測定単位を用いて行われます。次の表は利用可能な統計を説明しています。

統計	説明
Minimum	指定された期間に認められた最小値です。この値を用いて、アプリケーションの低ボリュームのアクティビティを判断できます。
Maximum	指定された期間に認められた最大値です。この値を用いて、アプリケーションの高ボリュームのアクティビティを判断できます。
Sum	該当するメトリクスで加算されたすべての合計値です。この統計はメトリクスの合計ボリュームを判断するのに役立ちます。
Average	指定した期間の Sum/SampleCount の値です。この統計を Minimum および Maximum と比較することで、メトリクスの全容、および平均使用量がどれくらい Minimum と Maximum に近いかを判断できます。この比較は必要に応じていつリソースを増減させるべきかを知るのに役立ちます。
SampleCount	統計計算で使用するデータポイントのカウント (数) です。
pNN.NN	指定されたパーセンタイルの値。小数点以下最大 2 桁を使用して、任意のパーセンタイルを指定できます (p95.45 など)。

内容

- 特定のインスタンスの統計を取得する
- インスタンス全体の統計の集約
- Auto Scaling グループ別に統計を集計する
- AMI 別に統計を集計する

特定のインスタンスの統計を取得する

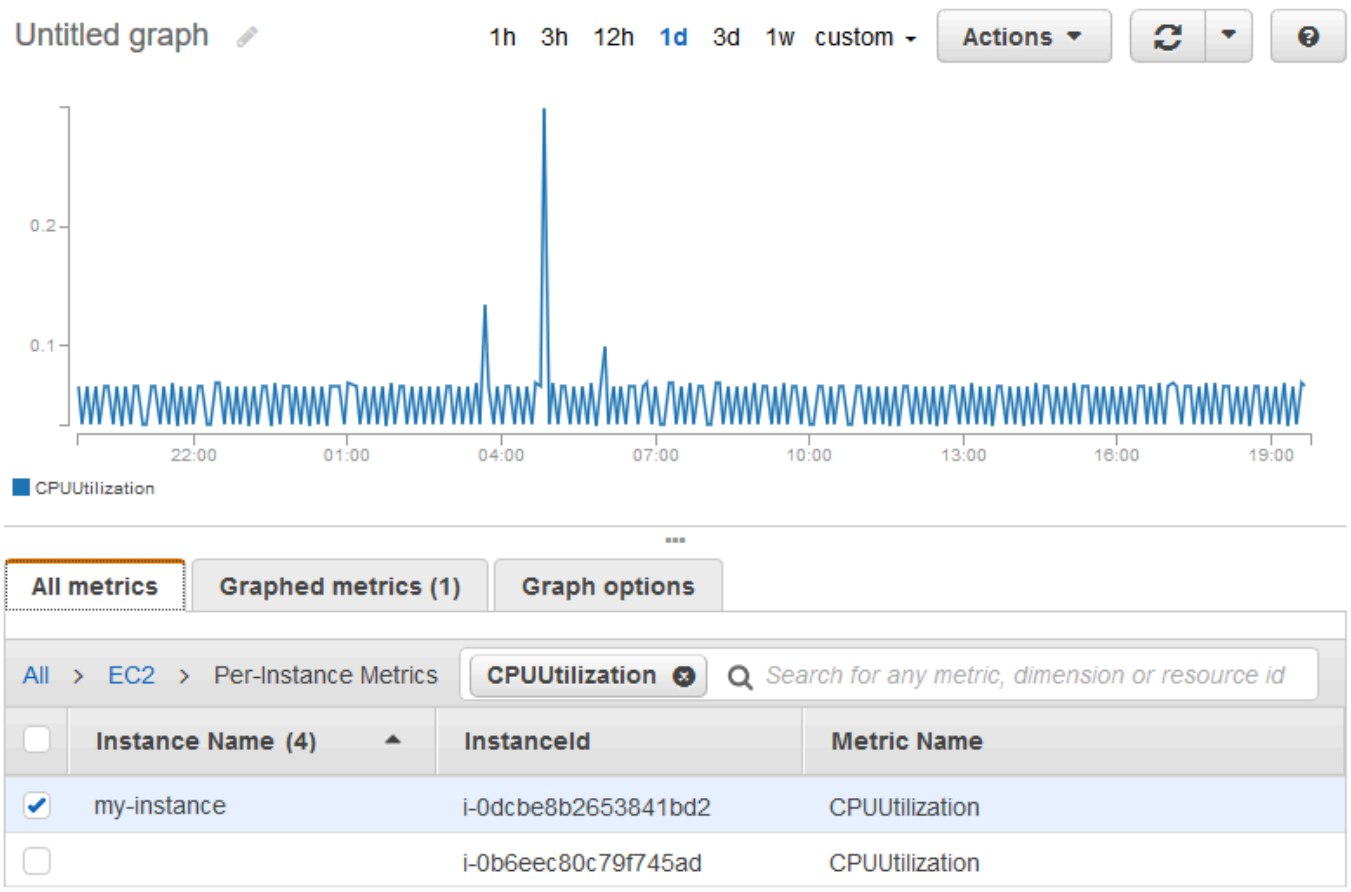
AWS マネジメントコンソールまたは AWS CLI を使用して、特定のインスタンスの統計を取得できます。次の例ではAWS マネジメントコンソール または AWS CLI を使用して、特定の EC2 インスタンスの最大 CPU 使用率を決定することができます。

要件

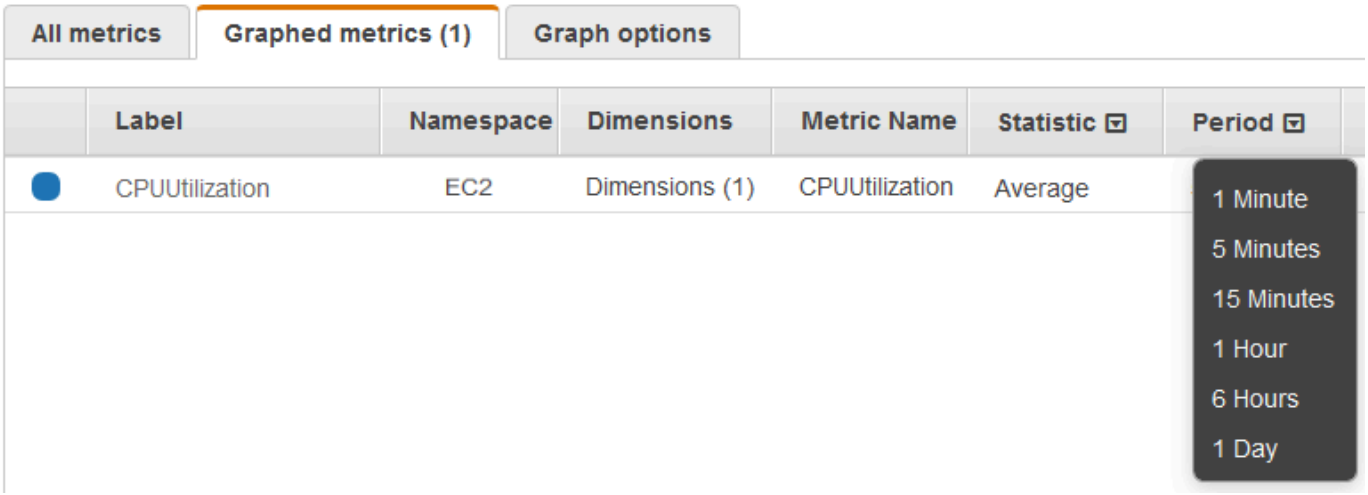
- インスタンスの ID が必要です。インスタンス ID はAWS マネジメントコンソール コンソールまたは describe-instances コマンドを使って取得します。
- デフォルトでは基本モニタリングが有効化されていますが、詳細モニタリングを有効化することもできます。詳細については「EC2 インスタンスの詳細モニタリングを管理する」を参照してください。

特定のインスタンスの CPU 使用率を表示するには (コンソール)

1. CloudWatch コンソールの https://console.aws.amazon.com/cloudwatch/ を開いてください。
2. ナビゲーションペインで [Metrics (メトリクス)] を選択してください。
3. EC2 のメトリクスの名前空間を選択してください。
4. インスタンス別メトリクスのディメンションを選択してください。
5. 検索フィールドに **`CPUUtilization`** と入力して Enter キーを押します。特定のインスタンスの行を選択してください。すると、そのインスタンスの [CPUUtilization] メトリクスのグラフが表示されます。グラフに名前を付けるには鉛筆アイコンを選択してください。時間範囲を変更するには事前定義済みの値を選択するか、[custom] を選択してください。



6. メトリクスの統計または期間を変更するには[Graphed metrics] タブを選択してください。列見出しまたは個々の値を選択し、次に異なる値を選択してください。



特定のインスタンスの CPU 使用率を取得するには (AWS CLI)

次の get-metric-statistics コマンドを使用すると、期間と時間間隔を指定して、特定のインスタンスの [CPUUtilization] メトリクスを取得できます。

```
aws cloudwatch get-metric-statistics --namespace AWS/EC2 --metric-name CPUUtilization
 --period 3600 \
--statistics Maximum --dimensions Name=InstanceId,Value=i-1234567890abcdef0 \
--start-time 2022-10-18T23:18:00 --end-time 2022-10-19T23:18:00
```

以下は出力の例です。それぞれの値は単一の EC2 インスタンスの最大 CPU 使用率を表しています。

```
{
    "Datapoints": [
        {
            "Timestamp": "2022-10-19T00:18:00Z",
            "Maximum": 0.33000000000000002,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2022-10-19T03:18:00Z",
            "Maximum": 99.670000000000002,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2022-10-19T07:18:00Z",
            "Maximum": 0.34000000000000002,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2022-10-19T12:18:00Z",
            "Maximum": 0.34000000000000002,
            "Unit": "Percent"
        }
    ],
    "Label": "CPUUtilization"
}
```

インスタンス全体の統計の集約

集約された統計は詳細モニタリングが有効になっているインスタンスで利用が可能です。基本モニタリングを使用するインスタンスは集約されません。インスタンス全体から集約された統計情報を取得

するには詳細モニタリングを事前に有効化し、1 分間隔でデータが提供されるようにしておく必要があります (追加料金がかかります)。

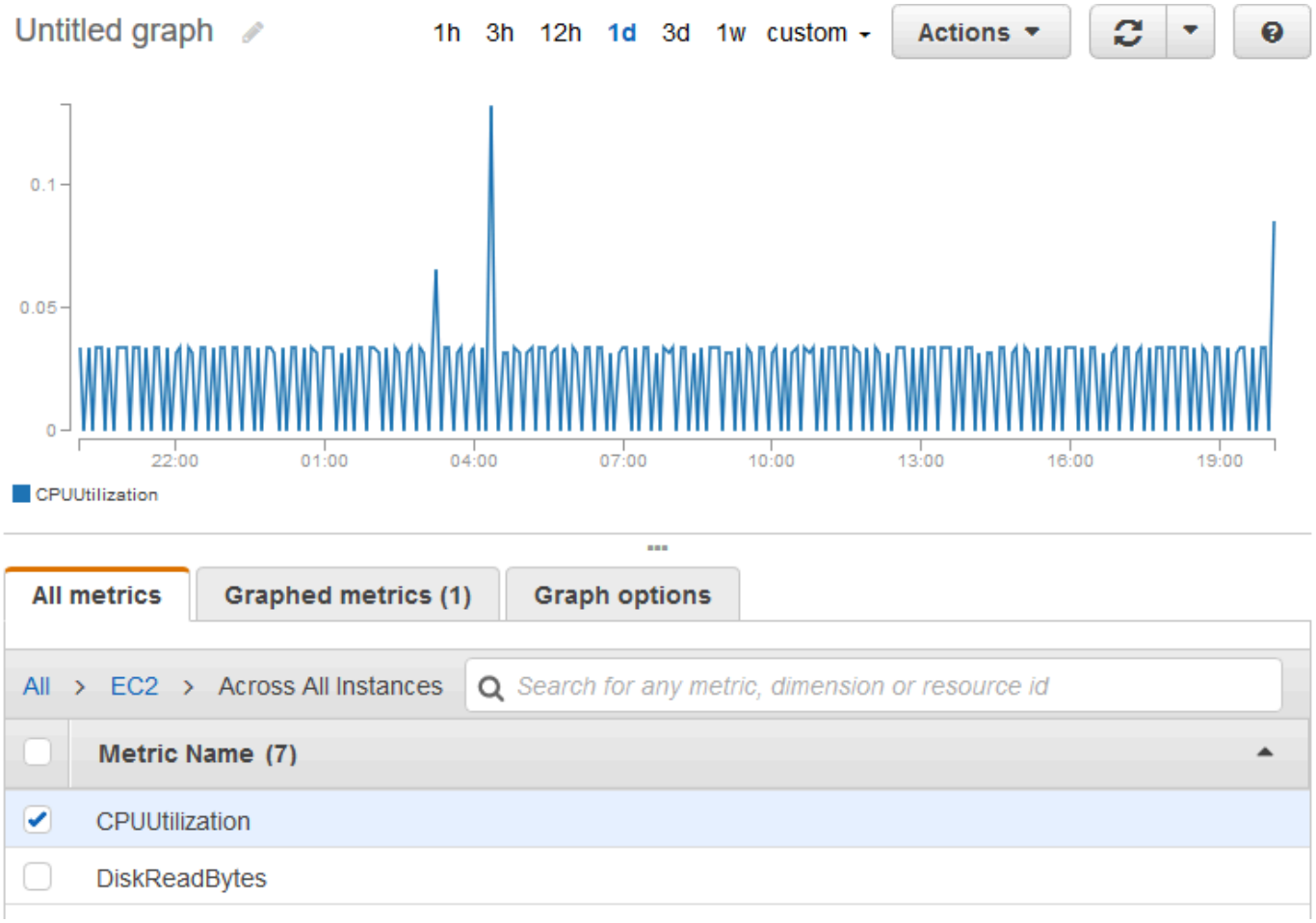
Amazon CloudWatch は AWS リージョンをまたがってデータを集約することはできません。メトリクスはリージョン間で完全に独立しています。

この例ではEC2 インスタンスの平均 CPU 使用率を取得するために詳細モニタリングを使用する方法について示します。ディメンションを指定していないため、CloudWatch はAWS/EC2 名前空間にある全ディメンションの統計を返します。

> **Important**
>
> AWS 名前空間にあるすべてのディメンションを取得するこの手法は、Amazon CloudWatch に発行するカスタム名前空間では機能しません。カスタム名前空間の場合、データポイントを含む統計を取得するにはそのデータポイントに関連付けられたディメンション一式をすべて指定する必要があります。

インスタンスの平均 CPU 使用率を表示するには (コンソール)

1. CloudWatch コンソールの https://console.aws.amazon.com/cloudwatch/ を開いてください。
2. ナビゲーションペインで [Metrics (メトリクス)] を選択してください。
3. [EC2] 名前空間を選択して、[Across All Instances] を選択してください。
4. [CPUUtilization] を含む行を選択してください。すべての EC2 インスタンスのメトリクスがグラフとして表示されます。グラフに名前を付けるには鉛筆アイコンを選択してください。時間範囲を変更するには事前定義済みの値を選択するか、[custom] を選択してください。



5. メトリクスの統計または期間を変更するには[Graphed metrics] タブを選択してください。列見出しまたは個々の値を選択し、次に異なる値を選択してください。

複数のインスタンスの平均 CPU 使用率を取得するには (AWS CLI)

次のように get-metric-statistics コマンドを使用し、インスタンス全体の平均 [CPUUtilization] メトリクスを取得します。

```
aws cloudwatch get-metric-statistics \
    --namespace AWS/EC2 \
    --metric-name CPUUtilization \
    --period 3600  --statistics "Average" "SampleCount" \
    --start-time 2022-10-11T23:18:00 \
    --end-time 2022-10-12T23:18:00
```

出力例を次に示します。

```
{
    "Datapoints": [
        {
            "SampleCount": 238.0,
            "Timestamp": "2022-10-12T07:18:00Z",
            "Average": 0.038235294117647062,
            "Unit": "Percent"
        },
        {
            "SampleCount": 240.0,
            "Timestamp": "2022-10-12T09:18:00Z",
            "Average": 0.16670833333333332,
            "Unit": "Percent"
        },
        {
            "SampleCount": 238.0,
            "Timestamp": "2022-10-11T23:18:00Z",
            "Average": 0.041596638655462197,
            "Unit": "Percent"
        }
    ],
    "Label": "CPUUtilization"
}
```


Auto Scaling グループ別に統計を集計する

Auto Scaling グループ内で EC2 インスタンスの統計を集計することができます。Amazon CloudWatch は AWS リージョンをまたがってデータを集約することはできません。メトリクスはリージョン間で完全に独立しています。

この例では1 つの Auto Scaling グループについて、ディスクに書き込まれた総バイト数を取得する方法を説明します。合計は指定された Auto Scaling グループのすべての EC2 インスタンスで、24 時間おきに 1 分間に対して算出されます。

Auto Scaling グループ内のインスタンスの DiskWriteBytes を表示するには (コンソール)

1. CloudWatch コンソールの https://console.aws.amazon.com/cloudwatch/ を開いてください。
2. ナビゲーションペインで [Metrics (メトリクス)] を選択してください。
3. [EC2] 名前空間を選択し、次に [By Auto Scaling Group] を選択してください。
4. [DiskWriteBytes] メトリクスの行と特定の Auto Scaling グループを選択してください。すると、その Auto Scaling グループ内にあるインスタンスのメトリクスがグラフとして表示されます。

グラフに名前を付けるには鉛筆アイコンを選択してください。時間範囲を変更するには事前定義済みの値を選択するか、[custom] を選択してください。

5. メトリクスの統計または期間を変更するには[Graphed metrics] タブを選択してください。列見出しまたは個々の値を選択し、次に異なる値を選択してください。

Auto Scaling グループのインスタンスの DiskWriteBytes を表示するには (AWS CLI)

以下のように get-metric-statistics コマンドを使用します。

```
aws cloudwatch get-metric-statistics --namespace AWS/EC2 --metric-name DiskWriteBytes
 --period 360 \
--statistics "Sum" "SampleCount" --dimensions Name=AutoScalingGroupName,Value=my-asg --
start-time 2022-10-16T23:18:00 --end-time 2022-10-18T23:18:00
```

出力例を次に示します。

```
{
    "Datapoints": [
        {
            "SampleCount": 18.0,
            "Timestamp": "2022-10-19T21:36:00Z",
            "Sum": 0.0,
            "Unit": "Bytes"
        },
        {
            "SampleCount": 5.0,
            "Timestamp": "2022-10-19T21:42:00Z",
            "Sum": 0.0,
            "Unit": "Bytes"
        }
    ],
    "Label": "DiskWriteBytes"
}
```

AMI 別に統計を集計する

詳細モニタリングが有効になっているインスタンスに対して、AMI を使用して統計を集計できます。基本モニタリングを使用するインスタンスは集約されません。インスタンス全体から集約された統計情報を取得するには詳細モニタリングを事前に有効化し、1 分間隔でデータが提供されるようにしておく必要があります (追加料金がかかります)。

Amazon CloudWatch は AWS リージョンをまたがってデータを集約することはできません。メトリクスはリージョン間で完全に独立しています。

この例では特定の Amazon マシンイメージ (AMI) を使用するすべてのインスタンスの平均 CPU 使用率を特定する方法を説明します。平均値は1 日間、60 秒間隔の平均値です。

AMI 別の平均 CPU 使用率を表示するには (コンソール)

1. CloudWatch コンソールの https://console.aws.amazon.com/cloudwatch/ を開いてください。
2. ナビゲーションペインで [Metrics (メトリクス)] を選択してください。
3. [EC2] 名前空間を選択し、次に [By Image (AMI) Id] を選択してください。
4. [CPUUtilization] メトリクスの行と特定の AMI を選択してください。すると、その AMI のメトリクスがグラフとして表示されます。グラフに名前を付けるには鉛筆アイコンを選択してください。時間範囲を変更するには事前定義済みの値を選択するか、[custom] を選択してください。
5. メトリクスの統計または期間を変更するには[Graphed metrics] タブを選択してください。列見出しまたは個々の値を選択し、次に異なる値を選択してください。

特定のイメージ ID の平均 CPU 使用率を取得するには (AWS CLI)

以下のように get-metric-statistics コマンドを使用します。

```
aws cloudwatch get-metric-statistics --namespace AWS/EC2 --metric-name CPUUtilization
 --period 3600 \
--statistics Average --dimensions Name=ImageId,Value=ami-3c47a355 --start-
time 2022-10-10T00:00:00 --end-time 2022-10-11T00:00:00
```

以下は出力の例です。それぞれの値は指定した AMI を実行する EC2 インスタンスの平均 CPU 使用率 (%) を表します。

```
{
    "Datapoints": [
        {
            "Timestamp": "2022-10-10T07:00:00Z",
            "Average": 0.041000000000000009,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2022-10-10T14:00:00Z",
            "Average": 0.079579831932773085,
```

```
            "Unit": "Percent"
        },
        {
            "Timestamp": "2022-10-10T06:00:00Z",
            "Average": 0.036000000000000011,
            "Unit": "Percent"
        }
    ],
    "Label": "CPUUtilization"
}
```

インスタンスのモニタリンググラフを表示する

インスタンスを起動した後はAmazon EC2 コンソールの [モニタリング] タブを開いて、インスタンスのモニタリンググラフを表示できます。各グラフは利用可能な Amazon EC2 メトリクスのいずれかに基づいています。

以下のグラフが利用可能です。

- 平均 CPU 使用率 (パーセント)
- 平均ディスク読み込み (バイト)
- 平均ディスク書き込み (バイト)
- 最大ネットワーク受信 (バイト)
- 最大ネットワーク送信 (Bytes)
- 要約ディスク読み取り操作 (カウント)
- 要約ディスク書き込み操作 (カウント)
- 要約ステータス (任意)
- 要約ステータスインスタンス (カウント)
- 要約ステータスシステム (カウント)

グラフに表示されるメトリクスおよびデータの詳細については[インスタンスに対して利用可能な CloudWatch メトリクス]を参照してください。

CloudWatch コンソールを使用したメトリクスのグラフ化

CloudWatch コンソールを使用して、Amazon EC2 や他の AWS のサービスによって生成されたメトリクスデータをグラフ化できます。詳細については「Amazon CloudWatch ユーザーガイド」の「[メトリクスのグラフ化]」を参照してください。

インスタンスの CloudWatch アラームを作成する

インスタンスの 1 つのCloudWatch メトリクスをモニタリングする、CloudWatch アラームを作成できます。CloudWatch はメトリクスが指定したしきい値に到達すると、自動的に通知を送信します。CloudWatch アラームは Amazon EC2 コンソールを使用するか、CloudWatch コンソールに用意されている高度なオプションを使用して作成できます。

CloudWatch コンソールを使用してアラームを作成するには

例については、「Amazon CloudWatch ユーザーガイド」の「Amazon CloudWatch アラームの作成」を参照してください。

Amazon EC2 コンソールを使用してアラームを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択し、[アクション]、[モニタリングとトラブルシューティング]、[CloudWatch アラームの管理] の順に選択してください。
4. [CloudWatch アラームの管理] 詳細ページの [アラームの追加または編集] で、[アラームの作成] を選択してください。
5. [アラーム通知] で、Amazon Simple Notification Service (Amazon SNS) 通知を設定するかどうかを選択してください。既存の Amazon SNS トピックを入力するか、名前を入力して新しいトピックを作成します。
6. [アラームアクション] で、アラームがトリガーされた際に実行するアクションを指定するかどうかを選択してください。リストからアクションを選択してください。
7. [アラームのしきい値] で、アラームのメトリクスと条件を選択してください。例えば、CPU 使用率が 80% に達した状態が 5 分間継続した場合にトリガーされるアラームを作成するには次の操作を行います。

   a. [サンプルのグループ化基準] ([平均]) と [サンプリングするデータのタイプ] ([CPU 使用率]) の設定はデフォルトのままにしてきます。

   b. [アラームの条件] で [>=] を選択し、[割合] に「**0.80**」と入力してください。

   c. [連続した期間] に「**1**」と入力し、[期間] で [5 分間] を選択してください。
8. (オプション) [Sample metric data] (サンプルメトリクスデータ) の場合、[Add to dashboard] (ダッシュボードに追加) を選択してください。
9. [Create] (作成) を選択してください。

CloudWatch アラーム設定は Amazon EC2 コンソールまたは CloudWatch コンソールから編集できます。アラームを削除する場合はCloudWatch コンソールから行えます。詳細については「Amazon CloudWatch ユーザーガイド」の「CloudWatch アラームの編集または削除」を参照してください。

インスタンスを停止、終了、再起動、または復旧するアラームを作成する

Amazon CloudWatch アラームアクションを使用して、 インスタンスを自動的に停止、終了、再起動、または復旧するアラームを作成できます。停止または終了アクションを使用すると、今後インスタンスを実行する必要がなくなったときにコストを節約できます。再起動アクションを使用すると、これらのインスタンスを自動的に再起動でき、復旧アクションを使用すると、システムで障害が発生した場合に新しいハードウェアで復旧できます。

> **Note**
>
> Amazon CloudWatch アラームの請求および料金に関する情報については、「Amazon CloudWatch ユーザーガイド」の「CloudWatch の請求とコスト」を参照してください。

サービスにリンクされたロール `AWSServiceRoleForCloudWatchEvents` を使用すると、AWS がお客様に代わってアラームアクションを実行できます。AWS マネジメントコンソール、AWS CLI、または IAM API で初めてアラームを作成する場合、CloudWatch はサービスにリンクされたロールを作成します。

自動的にインスタンスを停止または終了するシナリオはいくつもあります。例えば、バッチ給与計算処理ジョブまたは科学計算タスクを専用に行うインスタンスを使用している場合が挙げられます。これらのインスタンスは一定期間動作して仕事を完了します。このようなインスタンスは、アイドル状態 (課金されている状態) にせずに、停止または終了するとコスト削減につながります。停止アラームアクションと終了アラームアクションの主な違いとして、停止したインスタンスは、後で再実行が必要な場合に起動しやすいことと、同じインスタンス ID およびルートボリュームを維持できることがあります。しかし、終了したインスタンスを起動することはできません。代わりに新しいインスタンスを開始する必要があります。インスタンスストアボリュームのデータは、インスタンスの停止または終了に伴って失われます。

停止、終了、再起動、復旧の各アクションは、Amazon EC2 インスタンスメトリクスごとに設定されている任意のアラームに追加できます。これには、Amazon CloudWatch によって (`AWS/EC2` 名前空間で) 提供される基本モニタリングや詳細モニタリングのメトリクスが含まれます。また、`InstanceId` ディメンションを含む任意のカスタムメトリクスも (その値が実行中の有効な Amazon EC2 インスタンスを参照する場合に限り) 含まれます。

> **Important**
>
> メトリクスデータポイントが欠落している場合、ステータスチェックアラームは一時的に INSUFFICIENT_DATA の状態になることがあります。これはまれですが、インスタンスが正常であっても、メトリクスレポートシステムに中断がある場合に発生する場合があります。特にインスタンスを停止、終了、再起動、または復旧するようにアラームを設定する場合は、アラーム違反ではなく、INSUFFICIENT_DATA の状態を欠落データとして扱うことをお勧めします。

コンソールのサポート

Amazon EC2 コンソールまたは CloudWatch コンソールを使用してアラームを作成できます。このドキュメントの手順では、Amazon EC2 コンソールを使用します。CloudWatch コンソールを使用する手順については、「Amazon CloudWatch ユーザーガイド」の「インスタンスを停止、終了、再起動、または復旧するアラームを作成する」を参照してください。

アクセス許可

EC2 アラームアクションを実行するアラームを作成または変更するには、`iam:CreateServiceLinkedRole` が必要です。サービスロールとは、サービスがユーザーに代わってアクションを実行するために引き受ける IAM ロールです。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除できます。詳細については、IAM ユーザーガイドの AWS のサービスに許可を委任するロールを作成するを参照してください。

内容

- Amazon CloudWatch アラームへの停止アクションの追加
- Amazon CloudWatch アラームへの終了アクションの追加
- Amazon CloudWatch アラームへの再起動アクションの追加
- Amazon CloudWatch アラームへの復旧アクションの追加
- Amazon CloudWatch のアラームアクションのシナリオ

Amazon CloudWatch アラームへの停止アクションの追加

一定のしきい値に達したときに Amazon EC2 インスタンスを停止するアラームを作成できます。例えば、開発またはテスト用のインスタンスを実行したまま、終了するのを忘れることがたまにあります。平均 CPU 利用率が 24 時間 10% 未満である場合に、インスタンスがアイドル状態で使用され

ていないという信号を発してトリガーするアラームを作成できます。しきい値、持続時間、期間をニーズに合わせて調整し、アラームがトリガーされたときにメールを受信するよう Amazon Simple Notification Service (Amazon SNS) 通知を追加できます。

Amazon EBS ボリュームをルートボリュームとして使用するインスタンスは停止または終了できますが、インスタンスストアをルートボリュームとして使用するインスタンスでは終了のみ行えます。インスタンスストアボリュームのデータは、インスタンスの終了または停止に伴って失われます。

アイドル状態のインスタンスを停止させるアラームを作成するには (Amazon EC2 コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択し、[アクション]、[モニタリングとトラブルシューティング]、[CloudWatch アラームの管理] の順に選択します。

   または、[アラームステータス] 列でプラス記号
   ( )
   を選択できます。
4. [CloudWatch アラームの管理] ページで、次の操作を行います。

   a. [アラームの作成] を選択します。

   b. アラームがトリガーされたときに E メールを受信するには、[アラーム通知] で既存の Amazon SNS トピックを選択します。先ず、Amazon SNS コンソールを使用して Amazon SNS トピックを作成する必要があります。詳細については、Amazon Simple Notification Service デベロッパーガイド のAmazon SNS を使用した Application-to-Person (A2P) メッセージングを参照してください。

   c. [アラームアクション] をオンにして、[停止] を選択します。

   d. [サンプルのグループ化基準] と [サンプリングするデータのタイプ] で、統計とメトリクスを選択します。この例では、[平均] と [CPU 使用率] を選択します。

   e. [アラーム発生時] と [パーセント] で、メトリクスのしきい値を指定します。この例では、<= と 10 % を指定します。

   f. [連続期間] と [期間] で、アラームの評価期間を指定します。この例では、5 分間の 1 連続期間を指定します。

   g. Amazon CloudWatch は、自動的にアラーム名を作成します。名前を変更するには、[アラーム名] に新しい名前を入力します。アラーム名には ASCII 文字のみを使用する必要があります。

> **Note**
>
> アラーム設定は、アラームを作成する前に実際の要件に基づいて調整することも、アラーム作成後に編集することもできます。これにはメトリクス、しきい値、持続時間、アクション、通知設定などがあります。ただし、アラームの作成後のアラーム名の編集はできません。

h. [作成] を選択します。

Amazon CloudWatch アラームへの終了アクションの追加

(インスタンスで終了保護が有効になっていない限り)、一定のしきい値に達したときに EC2 インスタンスを自動的に終了させるアラームを作成することができます。例えば、インスタンスが仕事を終え、再びそのインスタンスを使用する必要がない場合は、インスタンスを削除することをお勧めします。後でインスタンスを使用する可能性がある場合は、インスタンスを削除するのではなく、停止するほうが良いでしょう。インスタンスストアボリュームのデータは、インスタンスの終了に伴って失われます。インスタンスの削除保護の有効化と無効化については、「インスタンスの終了保護を変更する」を参照してください。

アイドル状態のインスタンスを終了するアラームを作成するには (Amazon EC2 コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択し、[アクション]、[モニタリングとトラブルシューティング]、[CloudWatch アラームの管理] の順に選択します。

   または、[アラームステータス] 列でプラス記号
   ( )
   を選択できます。
4. [CloudWatch アラームの管理] ページで、次の操作を行います。

   a. [アラームの作成] を選択します。

   b. アラームがトリガーされたときに E メールを受信するには、[アラーム通知] で既存の Amazon SNS トピックを選択します。先ず、Amazon SNS コンソールを使用して Amazon SNS トピックを作成する必要があります。詳細については、Amazon Simple Notification

Service デベロッパーガイド のAmazon SNS を使用した Application-to-Person (A2P) メッセージングを参照してください。

c. [アラームアクション] をオンにして、[終了] を選択します。

d. [サンプルのグループ化基準] と [サンプリングするデータのタイプ] で、統計とメトリクスを選択します。この例では、[平均] と [CPU 使用率] を選択します。

e. [アラーム発生時] と [パーセント] で、メトリクスのしきい値を指定します。この例では、>= と 10 % を指定します。

f. [連続期間] と [期間] で、アラームの評価期間を指定します。この例では、1 時間の 24 連続期間を指定します。

g. Amazon CloudWatch は、自動的にアラーム名を作成します。名前を変更するには、[アラーム名] に新しい名前を入力します。アラーム名には ASCII 文字のみを使用する必要があります。

> **Note**
>
> アラーム設定は、アラームを作成する前に実際の要件に基づいて調整することも、アラーム作成後に編集することもできます。これにはメトリクス、しきい値、持続時間、アクション、通知設定などがあります。ただし、アラームの作成後のアラーム名の編集はできません。

h. [作成] を選択します。

Amazon CloudWatch アラームへの再起動アクションの追加

Amazon EC2 インスタンスをモニタリングし、自動的に再起動する Amazon CloudWatch アラームを作成できます。再起動アラームアクションは、インスタンスのヘルスチェックが失敗した場合に推奨されます (システムのヘルスチェックが失敗した場合には、復旧アラームアクションが推奨されます) 。インスタンスの再起動は、オペレーティングシステムの再起動と同等です。ほとんどの場合、インスタンスの再起動には数分しかかかりません。インスタンスを再起動すると、インスタンスは同じホスト上で保持されるため、インスタンスのパブリック DNS 名、プライベート IP アドレス、およびインスタンスストアボリューム上のすべてのデータは保持されます。

インスタンスの停止と再起動の場合とは違って、インスタンスを再起動しても、インスタンスの新しい (1 分間分の最低料金がある) 課金期間は開始されません。インスタンスストアボリュームのデータは、インスタンスの再起動しても保持されます。インスタンスストアボリュームは、再起動後に

ファイルシステムに再マウントする必要があります。詳細については、「Amazon EC2 インスタンスを再起動する」を参照してください。

> **Important**
>
> 再起動と復旧アクション間で不具合が発生するのを回避するには、再起動アラームと復旧アラームを同じ評価期間に設定するのを避けます。再起動アラームを各 1 分間の 3 回の評価期間に設定することをお勧めします。詳細については、「Amazon CloudWatch ユーザーガイド」の「アラームを評価する」を参照してください。

インスタンスを再起動するアラームを作成するには (Amazon EC2 コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択し、[アクション]、[モニタリングとトラブルシューティング]、[CloudWatch アラームの管理] の順に選択します。

   または、[アラームステータス] 列でプラス記号 ( ) を選択できます。
4. [CloudWatch アラームの管理] ページで、次の操作を行います。

   a. [アラームの作成] を選択します。

   b. アラームがトリガーされたときに E メールを受信するには、[アラーム通知] で既存の Amazon SNS トピックを選択します。先ず、Amazon SNS コンソールを使用して Amazon SNS トピックを作成する必要があります。詳細については、Amazon Simple Notification Service デベロッパーガイド のAmazon SNS を使用した Application-to-Person (A2P) メッセージングを参照してください。

   c. [アラームアクション] をオンにして、[再起動] を選択します。

   d. [サンプルのグループ化基準] と [サンプリングするデータのタイプ] で、統計とメトリクスを選択します。この例では、[平均] と [ステータスチェックに失敗しました: インスタンス] を選択しています。

   e. [連続期間] と [期間] で、アラームの評価期間を指定します。この例では、[1 分間]の [3] 連続期間と入力します。[1 分が無効] になっている場合は、[詳細なモニタリング] を有効にするか、代わりに [5 分] を選択する必要があります。

f. Amazon CloudWatch は、自動的にアラーム名を作成します。名前を変更するには、[アラーム名] に新しい名前を入力します。アラーム名には ASCII 文字のみを使用する必要があります。

g. [作成] を選択します。

Amazon CloudWatch アラームへの復旧アクションの追加

Amazon EC2 インスタンスをモニタリングする Amazon CloudWatch アラームを作成できます。下層のハードウェア障害または修復に AWS を必要とする問題によりインスタンスが正常に機能しなくなった場合に、自動的にインスタンスを復旧できます。終了したインスタンスは復旧できません。復旧されたインスタンスは、インスタンス ID、プライベート IP アドレス、Elastic IP アドレス、すべてのインスタンスメタデータを含め、元のインスタンスと同じです。

CloudWatch では、復旧アクションをサポートしていないインスタンスにあるアラームに、復旧アクションを追加することはできません。

`StatusCheckFailed_System` アラームがトリガーされ、復旧アクションが開始されると、アラームを作成し、復旧アクションに関連付けたときに選択した Amazon SNS トピックによって通知されます。インスタンスを復旧する際、インスタンスを再起動するときにインスタンスは移行され、メモリ内にあるデータは失われます。プロセスが完了すると、情報はアラームに設定された SNS トピックに発行されます。この SNS トピックをサブスクライブしているすべてのユーザーは、復旧処理のステータスと、それ以降の手順を含むメールの通知を受け取ります。インスタンスが復旧した時点でインスタンスが再起動されたことがわかります。

> **Note**
>
> 復旧アクションは、`StatusCheckFailed_System` でのみ使用できます。`StatusCheckFailed_Instance` では使用できません。

以下の問題が発生すると、システムステータスのチェックに失敗する可能性があります。

- ネットワーク接続の喪失
- システム電源の喪失
- 物理ホストのソフトウェアの問題
- ネットワーク到達可能性に影響する、物理ホスト上のハードウェアの問題

復旧アクションは、特定の特性を持つインスタンスでのみサポートされます。詳細については、「インスタンスの自動復旧」を参照してください。

インスタンスにパブリック IP アドレスが割り当てられている場合、復旧後にパブリック IP アドレスが維持されます。

> **Important**
>
> 再起動と復旧アクション間で不具合が発生するのを回避するには、再起動アラームと復旧アラームを同じ評価期間に設定するのを避けます。復旧アラームを各 1 分間の 2 回の評価期間に設定することをお勧めします。詳細については、「Amazon CloudWatch ユーザーガイド」の「アラームを評価する」を参照してください。

インスタンスを復旧するアラームを作成するには (Amazon EC2 コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択し、[アクション]、[モニタリングとトラブルシューティング]、[CloudWatch アラームの管理] の順に選択します。

   または、[アラームステータス] 列でプラス記号 (+) を選択できます。
4. [CloudWatch アラームの管理] ページで、次の操作を行います。

   a. [アラームの作成] を選択します。

   b. アラームがトリガーされたときに E メールを受信するには、[アラーム通知] で既存の Amazon SNS トピックを選択します。先ず、Amazon SNS コンソールを使用して Amazon SNS トピックを作成する必要があります。詳細については、Amazon Simple Notification Service デベロッパーガイド のAmazon SNS を使用した Application-to-Person (A2P) メッセージングを参照してください。

   > **Note**
   >
   > 今後、アラームがトリガーされたときにメール通知を受信するためには、指定された SNS トピックをサブスクライブする必要があります。AWS アカウントのルートユーザー は、自動インスタンス復旧アクションが発生すると、SNS トピックが指

定されていない場合や、ルートユーザーが指定した SNS トピックにサブスクライブしていない場合でも、常に E メール通知を受信します。

c. [アラームアクション] をオンにして、[復元] を選択します。

d. [サンプルのグループ化基準] と [サンプリングするデータのタイプ] で、統計とメトリクスを選択します。この例では、[平均] と [ステータスチェックに失敗しました: システム] を選択しています。

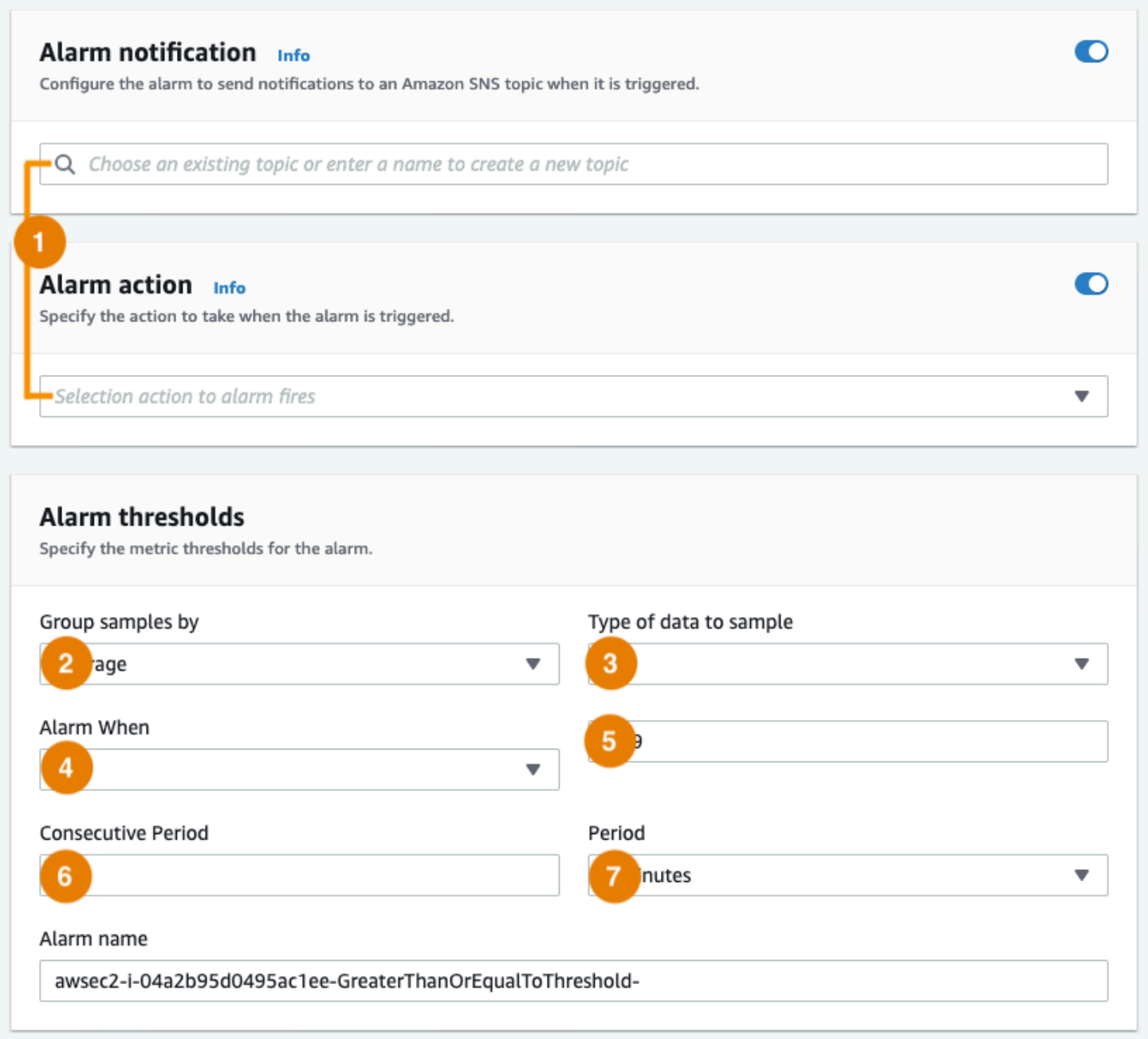
e. [連続期間] と [期間] で、アラームの評価期間を指定します。この例では、[1 分間]の [2] 連続期間と入力します。[1 分が無効] になっている場合は、[詳細なモニタリング] を有効にするか、代わりに [5 分] を選択する必要があります。

f. Amazon CloudWatch は、自動的にアラーム名を作成します。名前を変更するには、[アラーム名] に新しい名前を入力します。アラーム名には ASCII 文字のみを使用する必要があります。

g. [作成] を選択します。

Amazon CloudWatch のアラームアクションのシナリオ

Amazon EC2 (Amazon EC2) コンソールを使用して、一定の条件が満たされたときにインスタンスを停止または終了させるアラームアクションを作成することができます。アラームアクションが設定する以下のコンソールページの画面キャプチャー内に、設定の順番を付けました。また、アクションを適切に作成できるよう、次のシナリオの設定にも順番を付けました。



シナリオ 1: アイドル状態の開発インスタンスおよびテストインスタンスを停止する

ソフトウェアの開発またはテストに使用するインスタンスが 1 時間以上アイドル状態である場合に停止するアラームを作成します。

設定	値
1	停止

設定	値
2	最大
3	CPU 使用率 (%)
4	<=
5	10%
6	1
7	1 時間

シナリオ 2: アイドル状態のインスタンスを停止する

インスタンスが 24 時間アイドル状態である場合、インスタンスを停止し、メールを送信するアラームを作成します。

設定	値
1	停止および E メール
2	平均
3	CPU 使用率 (%)
4	<=
5	5%
6	24
7	1 時間

シナリオ 3: トラフィック量が異常に多いウェブサーバーについて E メールを送信する

インスタンスの 1 日当たりのアウトバウンドネットワークトラフィックが 10 GB を超える場合にメールを送信するアラームを作成します。

設定	値
1	メール
2	合計
3	ネットワーク出力
4	>
5	10 GB
6	24
7	1 時間

シナリオ 4: トラフィック量が異常に多いウェブサーバーを停止する

アウトバウンドトラフィックが 1 時間当たり 1 GB を超えた場合にインスタンスを停止し、テキストメッセージ (SMS) を送信するアラームを作成します。

設定	値
1	Stop and send SMS
2	合計
3	ネットワーク出力
4	>
5	1 GB
6	1
7	1 時間

シナリオ 5: 障害のあるインスタンスを停止する

3 回連続で状態チェック (5 分間隔で実施) が不合格のインスタンスを停止するアラームを作成します。

設定	値
1	停止
2	平均
3	ステータスチェックに失敗: システム
4	-
5	-
6	1
7	15 分

シナリオ 6: バッチ処理ジョブの完了時にインスタンスを削除する

バッチジョブを実行するインスタンスが結果データを送信しなくなったときに、そのインスタンスを削除するアラームを作成します。

設定	値
1	終了
2	最大
3	ネットワーク出力
4	<=
5	100,000 bytes
6	1

設定	値
7	5 分

EventBridge を使用して Amazon EC2 を自動化する

Amazon EventBridge を使用すると、AWS のサービス を自動化し、アプリケーションの可用性の問題やリソースの変更などのシステムイベントに自動的に対応できます。AWS のサービスからのイベントは、ほぼリアルタイムに EventBridge に提供されます。ルールを作成して、注目しているイベントと、イベントがルールに一致した場合に実行するアクションを指定できます。自動的にトリガーできるオペレーションには、以下が含まれます。

- AWS Lambda 関数を呼び出す
- Amazon EC2 Run Command を呼び出す
- Amazon Kinesis Data Streams へのイベントを中継する
- AWS Step Functions ステートマシンをアクティブ化する
- Amazon SNS トピックを通知する
- Amazon SQS キューを通知する

Amazon EC2 での EventBridge の使用例を次に示します。

- インスタンスが実行状態になるたびに Lambda 関数をアクティブ化します。
- Amazon EBS ボリュームの作成時または変更時に Amazon SNS トピックを通知します。
- AWS の別のサービスで特定のイベントが発生するたびに、Amazon EC2 Run Command を使用して 1 つ以上の Amazon EC2 インスタンスにコマンドを送信します。

詳細については、「Amazon EventBridge ユーザーガイド」を参照してください。

Amazon EC2 イベントタイプ

Amazon EC2 は、次のイベントタイプをサポートします。

- EC2 AMI の状態変更
- EC2 の高速起動状態の変更通知
- EC2 フリートエラー

- EC2 フリート情報
- EC2 フリートインスタンスの変更
- EC2 フリートのスポットインスタンスリクエストの変更
- EC2 フリートの状態の変更
- EC2 インスタンスの再調整に関するレコメンデーション
- EC2 インスタンスの状態変更通知
- EC2 スポットフリートのエラー
- EC2 スポットフリート情報
- EC2 スポットフリートインスタンスの変更
- EC2 スポットフリートのスポットインスタンスリクエストの変更
- EC2 スポットフリートの状態の変更
- EC2 スポットインスタンスの中断の警告
- EC2 スポットインスタンスリクエストのフルフィルメント
- EC2 ODCR 低使用率通知

Amazon EBS でサポートされているイベントタイプについては、「Amazon EBS 用 Amazon EventBridge」を参照してください。

AWS CloudTrail を使用して Amazon EC2 API コールをログに記録する

Amazon EC2 API は、ユーザー、ロール、または AWS のサービスが実行したアクションの記録を提供するサービスである AWS CloudTrail と統合されています。CloudTrail は、すべての Amazon EC2 API コールをイベントとしてキャプチャします。キャプチャされる呼び出しには、コンソールからの呼び出しが含まれます。CloudTrail で収集した情報を使用して、Amazon EC2 API に対して行われたリクエスト、リクエスト元の IP アドレス、およびリクエストが行われた日時を確認できます。

各イベントまたはログエントリには、リクエストの生成者に関する情報が含まれます。アイデンティティ情報は、以下を判別するのに役立ちます:

- ルートユーザーまたはユーザー認証情報のどちらを使用してリクエストが送信されたか
- リクエストが IAM Identity Center ユーザーに代わって行われたかどうか。
- リクエストがロールまたはフェデレーションユーザーのテンポラリなセキュリティ認証情報を使用して行われたかどうか。

- リクエストが、別の AWS のサービス によって送信されたかどうか。

アカウントを作成すると、AWS アカウント で CloudTrail がアクティブになり、自動的に CloudTrail の[イベント履歴] にアクセスできるようになります。CloudTrail の [イベント履歴] では、AWS リージョン で過去 90 日間に記録された 管理イベントの表示、検索、およびダウンロードが可能で、変更不可能な記録を確認できます。詳細については、「AWS CloudTrail ユーザーガイド」の「CloudTrail イベント履歴の使用」を参照してください。[イベント履歴] の閲覧には CloudTrail の料金はかかりません。

AWS アカウント で過去 90 日間のイベントを継続的に記録するには、証跡または CloudTrail Lake イベントデータストアを作成します。

CloudTrail 証跡

証跡により、CloudTrail はログファイルを Amazon S3 バケットに配信できます。AWS マネジメントコンソール を使用して作成した証跡はマルチリージョンです。AWS CLI を使用する際は、単一リージョンまたは複数リージョンの証跡を作成できます。アカウント内のすべて AWS リージョン でアクティビティを把握するため、マルチリージョン証跡を作成することをお勧めします。単一リージョンの証跡を作成する場合、証跡の AWS リージョン に記録されたイベントのみを表示できます。証跡の詳細については、「AWS CloudTrail ユーザーガイド」の「AWS アカウント の証跡の作成」および「組織の証跡の作成」を参照してください。

証跡を作成すると、進行中の管理イベントのコピーを 1 つ無料で CloudTrail から Amazon S3 バケットに配信できますが、Amazon S3 ストレージには料金がかかります。CloudTrail の料金の詳細については、「AWS CloudTrail の料金」を参照してください。Amazon S3 の料金に関する詳細については、「Amazon S3 の料金」を参照してください。

CloudTrail Lake イベントデータストア

CloudTrail Lake を使用すると、イベントに対して SQL ベースのクエリを実行できます。CloudTrail Lake は、行ベースの JSON 形式の既存のイベントを Apache ORC 形式に変換します。ORC は、データを高速に取得するために最適化された単票ストレージ形式です。イベントはイベントデータストアに集約されます。イベントデータストアは、高度なイベントセレクタを適用することによって選択する条件に基いた、イベントのイミュータブルなコレクションです。どのイベントが存続し、クエリに使用できるかは、イベントデータストアに適用するセレクタが制御します。CloudTrail Lake の詳細については、「AWS CloudTrail ユーザーガイド」の「Lake の使用AWS CloudTrail」を参照してください。

CloudTrail Lake のイベントデータストアとクエリにはコストがかかります。イベントデータストアを作成する際に、イベントデータストアに使用する料金オプションを選択します。料金オ

プションによって、イベントの取り込みと保存にかかる料金、および、そのイベントデータストアのデフォルトと最長の保持期間が決まります。CloudTrail の料金の詳細については、「AWS CloudTrail の料金」を参照してください。

CloudTrail の Amazon EC2 API 管理イベント

管理イベントでは、AWS アカウント のリソースに対して実行される管理オペレーションについての情報が得られます。これらのイベントは、コントロールプレーンオペレーションとも呼ばれます。CloudTrail は、デフォルトで管理イベントをログ記録します。

すべての Amazon EC2 API アクションは管理イベントとしてログに記録されます。CloudTrail によってログ記録される API アクションのリストについては、「Amazon EC2 API リファレンス」を参照してください。例えば、RunInstances、DescribeInstances、および StopInstances アクションへの呼び出しは、管理イベントとしてログに記録されます。

Amazon EC2 API イベントの例

各イベントは任意の送信元からの単一のリクエストを表し、リクエストされた API オペレーション、オペレーションの日時、リクエストパラメータなどに関する情報を含みます。CloudTrail ログファイルは、パブリック API コールの順序付けられたスタックトレースではないため、イベントは特定の順序で表示されません。

次のログファイルレコードは、ユーザーがインスタンスを終了したことを示しています。


```
{
   "Records":[
      {
         "eventVersion":"1.03",
         "userIdentity":{
            "type":"Root",
            "principalId":"123456789012",
            "arn":"arn:aws:iam::123456789012:root",
            "accountId":"123456789012",
            "accessKeyId":"AKIAIOSFODNN7EXAMPLE",
            "userName":"user"
         },
         "eventTime":"2016-05-20T08:27:45Z",
         "eventSource":"ec2.amazonaws.com",
         "eventName":"TerminateInstances",
         "awsRegion":"us-west-2",
         "sourceIPAddress":"198.51.100.1",
```

```
        "userAgent":"aws-cli/1.10.10 Python/2.7.9 Windows/7botocore/1.4.1",
        "requestParameters":{
           "instancesSet":{
              "items":[{
                 "instanceId":"i-1a2b3c4d"
              }]
           }
        },
        "responseElements":{
           "instancesSet":{
              "items":[{
                 "instanceId":"i-1a2b3c4d",
                 "currentState":{
                    "code":32,
                    "name":"shutting-down"
                 },
                 "previousState":{
                    "code":16,
                    "name":"running"
                 }
              }]
           }
        },
        "requestID":"be112233-1ba5-4ae0-8e2b-1c302EXAMPLE",
        "eventID":"6e12345-2a4e-417c-aa78-7594fEXAMPLE",
        "eventType":"AwsApiCall",
        "recipientAccountId":"123456789012"
     }
   ]
}
```


CloudTrail レコードの内容については、「AWS CloudTrail ユーザーガイド」の「CloudTrail record contents」を参照してください。

EC2 Instance Connect による接続を監査する

AWS CloudTrail を使用して、EC2 Instance Connect によってインスタンスに接続するユーザーを監査します。

AWS CloudTrail コンソールを使用して、EC2 Instance Connect 経由で SSH アクティビティを監査するには

1. CloudTrail コンソール (https://console.aws.amazon.com/cloudtrail/) を開きます。

2. 正しいリージョンを使用していることを確認します。
3. ナビゲーションペインで [Event history (イベント履歴)] を選択します。
4. [Filter (フィルター)] で、[Event source (イベントソース)]、[ec2-instance-connect.amazonaws.com] の順に選択します。
5. (オプション) [Time range (時間範囲)] で、時間範囲を選択します。
6. [Refresh events (イベントの更新)] アイコンを選択します。
7. SendSSHPublicKey API コールに対応するイベントがページに表示されます。矢印を使用してイベントを展開します。ユーザー名、SSH 接続を行うために使用した AWS アクセスキー、ソース IP アドレスなどの詳細が表示されます。
8. すべてのイベント情報を JSON 形式で表示するには、[View event (イベントの表示)] を選択します。[requestParameters] フィールドに、SSH 接続を行うために使用されたターゲットインスタンス ID、OS ユーザー名、およびパブリックキーが表示されます。

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "ABCDEFGONGNOMOOCB6XYTQEXAMPLE",
        "arn": "arn:aws:iam::1234567890120:user/IAM-friendly-name",
        "accountId": "123456789012",
        "accessKeyId": "ABCDEFGUKZHNAW4OSN2AEXAMPLE",
        "userName": "IAM-friendly-name",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-09-21T21:37:58Z"}
        }
    },
    "eventTime": "2018-09-21T21:38:00Z",
    "eventSource": "ec2-instance-connect.amazonaws.com",
    "eventName": "SendSSHPublicKey ",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "123.456.789.012",
    "userAgent": "aws-cli/1.15.61 Python/2.7.10 Darwin/16.7.0 botocore/1.10.60",
    "requestParameters": {
        "instanceId": "i-0123456789EXAMPLE",
        "osUser": "ec2-user",
        "SSHKey": {
            "publicKey": "ssh-rsa ABCDEFGHIJKLMNO01234567890EXAMPLE"
        }
```

```
     },
    "responseElements": null,
    "requestID": "1a2s3d4f-bde6-11e8-a892-f7ec64543add",
    "eventID": "1a2w3d4r5-a88f-4e28-b3bf-30161f75be34",
    "eventType": "AwsApiCall",
    "recipientAccountId": "0987654321"
}
```

CloudTrail イベントを S3 バケットに収集するために AWS アカウントを設定している場合は、プログラムで情報をダウンロードして監査できます。詳細については、「AWS CloudTrail ユーザーガイド」の「CloudTrail ログファイルの取得と表示」を参照してください。

CloudWatch Application Insights を使用した .NET および SQL Server アプリケーションのモニタリング

CloudWatch Application Insights は他の AWS アプリケーションリソースとともに Amazon EC2 インスタンスを使用する .NET および SQL Server アプリケーションをモニタリングするのに役立ちます。アプリケーションのリソース全体およびテクノロジースタック (Microsoft SQL Server データベース、ウェブ (IIS) サーバー、アプリケーションサーバー、OS、ロードバランサー、キューなど) で主要なメトリクスとログおよびアラームを特定して設定します。メトリクスとログを継続的にモニタリングし、異常やエラーを検出して相互に関連付けます。エラーや異常が検出されると、Application Insights は CloudWatch Events を生成します。これを使用して、通知を設定したり、アクションを実行したりできます。トラブルシューティングを支援するために、検出した問題の自動ダッシュボードを作成します。このダッシュボードには相互に関連付けられた異常とログエラー、さらに根本原因を示唆する追加のインサイトが示されます。自動ダッシュボードを使用すると、修復アクションをすばやく実行してアプリケーションを正常な状態に保ち、アプリケーションのエンドユーザーへの影響を防止できます。

検出された問題に関して提供される情報

- 問題の簡単な概要
- 問題の発生日時
- 問題の重大度: 高/中/低
- 検出された問題のステータス: 進行中/解決済み
- インサイト: 検出された問題と考えられる根本原因に関して自動生成されるインサイト

- インサイトに関するフィードバック: CloudWatch Application Insights for .NET and SQL Server で生成されたインサイトの有益性についてユーザーが提供したフィードバック
- 関連する監視結果: さまざまなアプリケーションコンポーネントにまたがる問題に関連するメトリクスの異常とログのエラースニペットの詳細ビュー

フィードバック

検出された問題への自動生成されたインサイトが有益であるかどうかを判定することで、インサイトに対するフィードバックを提供できます。インサイトに対するユーザーからのフィードバックとアプリケーションの診断結果 (メトリクスの異常とログの例外) は今後同様の問題が発生した場合の検出を向上させるために使用されます。

詳細については、「Amazon CloudWatch ユーザーガイド」の「CloudWatch Application Insights」ドキュメントを参照してください。

Amazon EC2 の無料利用枠の使用状況を追跡する

AWS アカウントを作成すると、AWS 無料利用枠を利用して、Amazon EC2 を無料で使い始めることができます。無料利用枠の特典は、アカウントを作成したのが 2025 年 7 月 15 日より前か、2025 年 7 月 15 日以降かによって異なります。詳細については、「AWS Billing ユーザーガイド」の「Explore AWS services with AWS Free Tier」を参照してください。

2025 年 7 月 15 日より前とそれ以降の無料利用枠の特典

次の表は、AWS アカウントの作成日に基づく無料利用枠の特典を比較したものです。

無料利用枠の特典	2025 年 7 月 15 日より前に作成されたアカウント	2025 年 7 月 15 日以降に作成されたアカウント
[無料利用枠の対象] と表示されたインスタンスタイプ	`t2.micro`, `t3.micro`	`t3.micro`, `t3.small`, `t4g.micro` , `t4g.small` , `c7i-flex.large` , `m7i-flex.large`
[無料利用枠の対象] と表示された Amazon EBS ボリュームタイプ	`standard`, `st1`, `sc1`, `gp2`, および `gp3`	`standard`, `st1`, `sc1`, `gp2`, および `gp3`

無料利用枠の特典	2025 年 7 月 15 日より前に作成されたアカウント	2025 年 7 月 15 日以降に作成されたアカウント
[無料利用枠の対象] と表示された AMI	[無料利用枠の対象] と表示された AMI を確認	[無料利用枠の対象] と表示された AMI を確認
使用制限	使用制限が設定され、それを超えた後は従量制料金が請求されます。	サインアップすると 100 USD のクレジットを受け取ることができ、最大 100 USD の追加クレジットを獲得できます。
無料利用枠の期間	無料利用枠は、アカウントの作成日から 12 か月間有効です。この間、無料利用枠の使用制限を超えた場合は、従量制料金が請求されます。	無料利用枠は、アカウントの作成日から 6 か月間、またはクレジットを使い切った時点のいずれか早い方まで有効です。無料利用枠の制限を超えることはできません。

無料利用枠の対象となるインスタンスタイプを一覧表示するには

describe-instance-types コマンドを `free-tier-eligible` フィルターと共に使用します。

```
aws ec2 describe-instance-types \
    --filters Name=free-tier-eligible,Values=true \
    --query "InstanceTypes[*].[InstanceType]" \
    --output text | sort
```

無料利用枠の対象となる AMI を一覧表示する方法

`free-tier-eligible` フィルターで「describe-images」コマンドを使用します。

```
aws ec2 describe-images \
    --filters Name=free-tier-eligible,Values=true \
    --query "Images[*].[ImageId]" \
    --output text | sort
```

2025 年 7 月 15 日より前に作成されたアカウントの無料利用枠の使用状況を追跡する

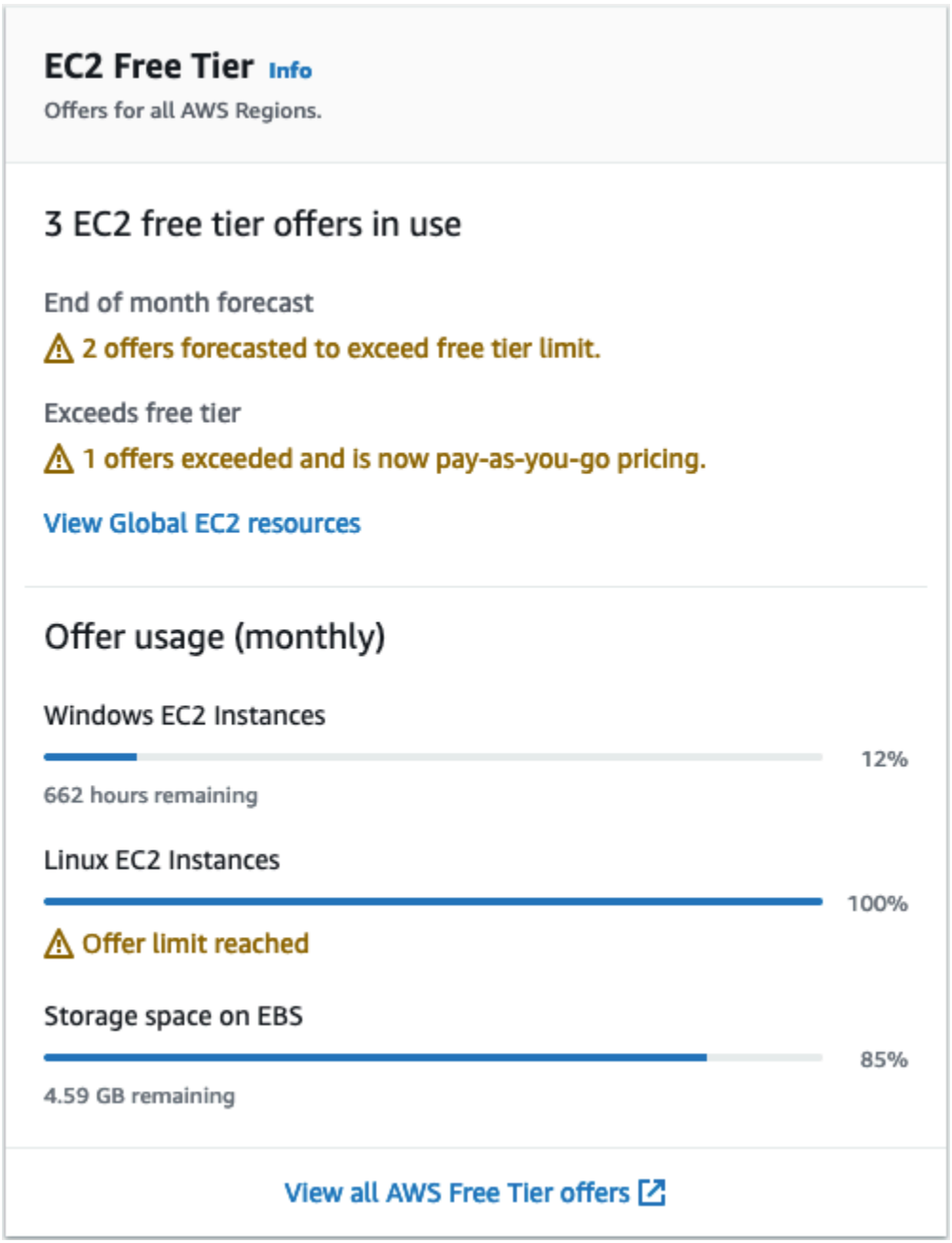
> **Note**
>
> このセクションは、2025 年 7 月 15 日より前に AWS アカウントを作成した無料利用枠ユーザーにのみ適用されます。2025 年 7 月 15 日以降にアカウントを作成した場合は、「AWS Billing ユーザーガイド」の「Tracking your AWS Free Tier usage」を参照してください。

2025 年 7 月 15 日より前にアカウントを作成した場合、AWS のお客様になってから 12 か月未満で、AWS 無料利用枠 使用制限の範囲内であれば、Amazon EC2 は無料で使用できます。予期せぬ請求を避けるには無料利用枠の使用状況を追跡することが重要です。無料利用枠の制限を超える場合、標準の従量課金制料金が発生します。詳細については「AWS 無料利用枠」を参照してください。

> **Note**
>
> 12 か月以上ご利用いただいている AWS カスタマーの場合は無料利用枠の利用資格がなくなり、以下の手順で説明する [EC2 無料利用枠] ボックスも表示されなくなります。

無料利用枠の使用状況を追跡するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[EC2 ダッシュボード] を選択してください。
3. [EC2 無料利用枠] ボックス (右上) を参照してください。



4. [EC2 無料利用枠] ボックスで、次のように無料利用枠の使用状況を確認します。

   - [使用中の EC2 無料利用枠のオファー] の警告に注意してください。
     - [月末の予測] — 現在の使用パターンを継続する場合、今月は料金が発生することを警告しています。
     - [無料利用枠超過] — 無料利用枠の制限を超え、すでに料金が発生していることを警告しています。

- [オファーの使用量 (月額)] で、Linux インスタンス、Windows インスタンス、EBS ストレージの使用状況を書き留めてください。このパーセンテージは今月使用した無料利用枠の制限を示します。100% の場合、それ以降の使用には料金が発生します。

> **Note**
>
> この情報はインスタンスを作成した後にのみ表示されます。ただし、使用状況情報はリアルタイムでは更新されず、1 日 3 回更新されます。

5. さらなる料金の発生を避けるには現在料金が発生しているリソース、または無料利用枠の使用制限を超えた場合に料金が発生するリソースをすべて削除してください。

   - インスタンスを削除する手順については「Amazon EC2 インスタンスを終了する」を参照してください。
   - 料金が発生する可能性のあるリソースが他のリージョンにあるかどうかを確認するには[EC2 無料利用枠] ボックスで [グローバル EC2 リソースを表示] を選択して [EC2 グローバルビュー] を開きます。詳細については「AWS Global View を使用して複数のリージョンにわたるリソースを表示する」を参照してください。

6. AWS 無料利用枠 にあるすべての AWS のサービス のリソース使用量を表示するには[EC2 無料利用枠] ボックスの下部にある [すべての AWS 無料利用枠 オファーを表示] を選択してください。詳細については「AWS Billing User Guide」の「Trying services using AWS 無料利用枠」を参照してください。

請求および使用状況レポートでの Amazon EC2 のコードを理解する

Amazon EC2 を使用すると、AWS 請求および使用状況レポートに関連コードが含められます。これらのコードを確認することは、Amazon EC2 のコストと使用パターンを理解するのに役立ちます。コストを最適化するには、費用の追跡と管理が不可欠です。

以下の表では、請求および使用状況レポートに表示される Amazon EC2 のコードについて説明します。請求および使用状況レポートで使用されるリージョンコードのリストについては、「AWS Region billing codes」を参照してください。

次の請求コード:

- インスタンス

- ベアメタルインスタンス
- 専用ホスト
- Dedicated Instances
- EBS 最適化
- キャパシティ予約

関連リソース

- the section called “請求と購入のオプション”
- the section called “AMI の請求について”
- Amazon EC2 の料金

インスタンス

Code	説明	単位
region-BoxUsage :*instance-type*	オンデマンドインスタンスの実行時間。	時間
region-HostBoxUsage :*instance-type*	専有ホスト上のインスタンスの実行時間。	時間
region-SpotUsage :*instance-type*	スポットインスタンスの実行時間。	時間

ベアメタルインスタンス

Code	説明	単位
region-BoxUsage :*instance-family* .metal	ベアメタルオンデマンドインスタンスの実行時間。	時間

Code	説明	単位
region-HostBoxUsage :*instance-family* .metal	専有ホスト上のベアメタルインスタンスの実行時間。	時間
region-SpotUsage :*instance-family* .metal	ベアメタルスポットインスタンスの実行時間。	時間

専用ホスト

Code	説明	単位
region-HostUsage :*host-type*	専有ホストがプロビジョニングされる時間。	時間
region-ReservedHostUsage :*host-type*	専有ホストの予約が適用される時間。	時間

Dedicated Instances

Code	説明	単位
region-DedicatedUsage :*instance-type*	ハードウェア専有インスタンスの実行時間。	時間 + リージョンごとの料金

EBS 最適化

Code	説明	単位
region-EBSOptimized :*instance-type*	EBS 最適化が有効になっている時間。	時間

キャパシティ予約

Code	説明	単位
region-Reservation :*instance-type*	キャパシティ予約のリザーブドインスタンス時間。	時間
region-UnusedBox :*instance-type*	キャパシティ予約の未使用のリザーブドインスタンス時間。	時間
region-DedicatedRes :*instance-type*	専有キャパシティ予約のリザーブドインスタンス時間。	時間
region-UnusedDed :*instance-type*	専有キャパシティ予約の未使用のリザーブドインスタンス時間。	時間

EC2 インスタンスのインベントリを作成する

Amazon EC2 は、オンデマンドでスケーラブルなコンピューティング能力を提供します。EC2 インスタンスは AWS ネットワーク上の仮想サーバーです。

次の表で EC2 インスタンスの主要な特徴を説明します。システム管理者は、本書に記載されているガイダンスを使用してこれらの特徴に関する情報を取得し、それを利用して、オンプレミスまたは別のクラウドプロバイダーのネットワークで実行される機能的に同等のサーバーを設定できます。

特性	説明
地理的場所	Amazon EC2 は、世界中のすべての AWS リージョンでホストされています。ロケーションは、インスタンスが顧客の近くに配置されるような場所を選択します。複数のロケーションでインスタンスを起動できます。
ハードウェア仕様	Amazon EC2 では、幅広いインスタンスタイプの選択肢があり、さまざまなユースケースに合わせて最適化できます。インスタンスに選択するインスタンスタイプによって、コンピューティング、メモリ、ネットワーキング、およびストレージのリソースが決まります。
イメージ	Amazon マシンイメージ (AMI) には、起動時にインスタンスにインストールするソフトウェアが含まれています。オペレーティングシステム、ソフトウェアパッケージ、カスタム設定などが含まれます。
IP アドレスと DNS ホスト名	インスタンスは、プライベート IP アドレスとプライベート DNS ホスト名を受信します。インスタンスにパブリック IP アドレスを設定すると、パブリック DNS ホスト名も受信します。
「セキュリティグループのルール」	インスタンスに関連付けられたセキュリティグループルールによって、許可されるインバウンドトラフィックとアウトバウンドトラフィックが決まります。
ユーザーデータ	ユーザーデータは、起動時にインスタンスで利用可能になります。シェルスクリプト (Linux) または PowerShell スクリプト (Windows) で構成されます。

地理的場所

Amazon EC2 は、世界中のすべての AWS リージョンで利用できます。各リージョンは、個別の地理的領域です。大部分のユーザーに近いリージョンをサーバー向けに選択すると、ネットワークレイテンシーを低減できます。

Amazon EC2 グローバルビューを使用すると、すべてのリージョンの Amazon EC2 インスタンスを一覧表示できます。AWS マネジメントコンソールおよび API では、各リージョンのインスタンスを個別に一覧表示する必要があります。

これが重要な理由

インスタンスの場所を特定したら、必要に応じて、機能的に同等のサーバーを同じロケーションにデプロイするか、それとも異なるロケーションにデプロイするかを決定できます。

すべてのリージョンの EC2 インスタンスの概要を取得するには

1. Amazon EC2 Global View コンソール (https://console.aws.amazon.com/ec2globalview/home) を開きます。
2. [リージョンエクスプローラー] タブの [概要] で、[インスタンス] のリソース数を確認します。これには、インスタンスの数とリージョンの数が含まれます。下線付きのテキストをクリックして、インスタンスの数がリージョン全体にどのように分散しているかを確認します。
3. [グローバル検索] タブで、クライアントフィルター [リソースタイプ = インスタンス] を選択します。リージョンまたはタグを指定することで、結果をさらにフィルタリングできます。

AWS CLI を使用してリージョンにある EC2 インスタンスの数を取得するには

次の describe-instances コマンドを使用して、指定したリージョンにあるインスタンスの数を調べます。このコマンドは、インスタンスがあるリージョンごとに実行する必要があります。

```
aws ec2 describe-instances \
    --region us-east-2 \
    --query "length(Reservations[*].Instances[])"
```

以下は出力の例です。

```
27
```

インスタンス数ではなくインスタンス ID のリストを取得するには、代わりに次の --query パラメータを使用します。

```
--query "Reservations[*].Instances[].InstanceId"
```

ハードウェア仕様

EC2 インスタンスに指定するインスタンスタイプによって、使用可能なコンピューティング、メモリ、ストレージ、ネットワーキングのリソースが決まります。各インスタンスファミリはコンピューティング、メモリ、ネットワーク、ストレージのリソースが異なるバランスで構成されています。詳細については「Amazon EC2 インスタンスタイプガイド」を参照してください。

これらが重要な理由

インスタンスのハードウェア仕様を決定したら、機能的に同等のサーバーに必要な最小仕様を決定できます。

インスタンスで使用するインスタンスタイプの概要を取得するには

次の describe-instances コマンドを使用します。sed コマンドを実行すると、角括弧を含む行と空の行が削除されます。このコマンドは、インスタンスがあるリージョンごとに実行する必要があります。

```
aws ec2 describe-instances \
    --query "Reservations[*].Instances[].InstanceType" | sed 's/[][]//g;/^$/d' | sort |
 uniq -c | sort -nr
```

以下は出力の例です。

```
  20    "c6i.4xlarge",
   5    "t2.micro",
   2    "g6e.2xlarge",
```

特定のインスタンスタイプに関する情報を取得するには

次の describe-instance-types コマンドを使用すると、インスタンスタイプの説明が表示されます。--query パラメータには、出力内に指定されたフィールドのみが含まれます。これらは基本的な特徴です。必要な追加フィールドを含めるか、「Amazon EC2 インスタンスタイプのガイド」の「Amazon EC2 インスタンスタイプの仕様」を参照することもできます。

```
aws ec2 describe-instance-types \
```

```
    --instance-types c6i.4xlarge t2.micro g6e.2xlarge \
    --query "InstanceTypes[*].
[InstanceType,VCpuInfo.DefaultVCpus,MemoryInfo.SizeInMiB,NetworkInfo.NetworkPerformance]"
 \
    --output table
```

以下は出力の例です。列は、インスタンスタイプ、vCPU、メモリ (GiB)、ネットワーク帯域幅 (Gbps) です。

```
-----------------------------------------------------
|                DescribeInstanceTypes              |
+--------------+-----+--------+----------------------+
|  t2.micro    |  1  |  1024  |  Low to Moderate     |
|  c6i.4xlarge |  16 |  32768 |  Up to 12.5 Gigabit  |
|  g6e.2xlarge |  8  |  65536 |  Up to 20 Gigabit    |
+--------------+-----+--------+----------------------+
```

イメージ

Amazon マシンイメージ (AMI) は、オペレーティングシステム、サービス、開発ツール、アプリケーションなど、起動時にインスタンスにインストールされるソフトウェアを提供します。AMI を他の形式にエクスポートすると、他のサーバーで使用できます。

これらが重要な理由

インスタンスの AMI を決定したら、機能的に同等のサーバーで必要なイメージを計画できます。AMI を他の場所で使用できる形式にエクスポートできる場合があります。そうでない場合、各 AMI から起動したインスタンスに接続し、インストールされている内容と設定を確認して、必要なものがすべてコピーされていることを確認する必要があります。

インスタンスの AMI を取得するには

次の describe-instances コマンドを使用します。sed コマンドを実行すると、角括弧を含む行と空の行が削除されます。このコマンドは、インスタンスがあるリージョンごとに実行する必要があります。

```
aws ec2 describe-instances \
    --query "Reservations[*].Instances[].ImageId" | sed 's/[][]//g;/^$/d' | sort | uniq
 -c | sort -nr
```

以下は出力の例です。

```
20     "ami-0a70b9d193ae8a79",
 5     "ami-07d9cf938edb0739b",
 2     "ami-09245d5773578a1d6",
```

特定の AMI に関する情報を取得するには

次の describe-images コマンドを使用します。--query パラメータには、出力内に指定されたフィールドのみが含まれます。必要な追加フィールドを含めることができます。

```
aws ec2 describe-images \
    --image-id ami-0a70b9d193ae8a79 ami-07d9cf938edb0739b ami-09245d5773578a1d6 \
    --query "Images[*].
{ID:ImageId,CreateDate:CreationDate,Name:Name,Description:Description}"
```

以下は出力の例です。

```
[
    {
        "ID": "ami-0a70b9d193ae8a799",
        "CreateDate": "2025-03-01T02:22:41.000Z",
        "Name": "web-server-3.1",
        "Description": "Image for web servers"
    },
    {
        "ID": "ami-07d9cf938edb0739b",
        "CreateDate": "2025-02-01T23:59:03.000Z",
        "Name": "awesome-application-11.5",
        "Description": "Image for Awesome Application"
    },
    {
        "ID": "ami-09245d5773578a1d6",
        "CreateDate": "2025-01-31T02:22:41.000Z",
        "Name": "monitoring-4.2",
        "Description": "Monitoring software"
    }
]
```

AMI から VM ファイルを作成するには

export-image コマンドを使用します。サポートされているイメージ形式は、VHD (Citrix Xen および Microsoft Hyper-V と互換性あり）、VMDK (VMware ESX および VMware vSphere と互換性あ

り) 、および raw 形式 (KVM および Xen ハイパーバイザーと互換性あり) です。VM Import/Export の要件および制限事項については、「VM Import/Export の要件」を参照してください。

IP アドレスと DNS ホスト名

ユーザーは、パブリック DNS ホスト名を使用してインターネット経由で EC2 インスタンスに接続します。EC2 インスタンスのパブリック DNS ホスト名は、そのパブリック IP アドレスに解決されます。EC2 インスタンスはプライベート IP アドレスを使用して互いに通信できます。

これらが重要な理由

IP アドレスを使用すると、デバイスはローカルネットワークまたはインターネット経由で相互に通信できます。インスタンスの IP アドレスと DNS 名をメモすることで、機能的に同等のサーバーが、インスタンスと同じクライアントまたはデバイスと通信できるようになります。例えば、ロードバランサー設定や、DNS プロバイダー用に作成した DNS レコードを更新できます。

インスタンスの IP アドレスと DNS ホスト名を取得するには

次の describe-instances コマンドを使用します。このコマンドは、インスタンスがあるリージョンごとに実行する必要があります。--query パラメータには、出力内に指定されたフィールドのみが含まれます。必要な追加フィールドを含めることができます。

```
aws ec2 describe-instances \
    --query "Reservations[*].Instances[].[InstanceId,PrivateIpAddress,PublicDnsName]" \
    --output table
```

以下は出力の例です。列は、インスタンス ID、プライベート IPv4 アドレス、パブリック IPv4 DNS 名です。

```
----------------------------------------------------------------------------------
|                                DescribeInstances
 |
+---------------------+------------+-----------------------------------------------
+
| i-0bac57d7472c89bac | 10.0.2.139 | ec2-192-0-2-142.us-east-2.compute.amazonaws.com
 |
| i-0fa8b7678975a3fff | 10.0.14.85 | ec2-198-51-100-56.us-east-2.compute.amazonaws.com
 |
  ...
  ...
```

```
| i-08fd74f3f1595fdbd | 10.0.1.241 | ec2-203.0.113.13.us-east-2.compute.amazonaws.com
 |
+---------------------+------------+-------------------------------------------------
+
```

「セキュリティグループのルール」

セキュリティグループは、仮想ファイアウォールとして機能します。セキュリティグループを EC2 インスタンスに関連付けると、そのルールにより、特定のポートとプロトコルを介した EC2 インスタンスのインバウンドトラフィックとアウトバウンドトラフィックが許可されます。

これらが重要な理由

サーバーに到達できるインバウンドトラフィックとサーバーから出ることができるアウトバウンドトラフィックを決定した後で、機能的に同等のサーバーに必要なファイアウォールルールを計画できます。

インスタンスのセキュリティグループを取得するには

次の describe-instances コマンドを使用します。sed コマンドを実行すると、角括弧を含む行と空の行が削除されます。このコマンドは、インスタンスがあるリージョンごとに実行する必要があります。

```
aws ec2 describe-instances \
    --query "Reservations[*].Instances[].SecurityGroups[].GroupId" | sed 's/[][]//g;/^
$/d' | sort | uniq -c | sort -nr
```

以下は出力の例です。

```
27     "sg-01dd3383691d02f42",
10     "sg-08c77cc910c0b3b2c",
 2     "sg-00f4e409629f1a42d",
```

セキュリティグループのインバウンドルールを取得するには

次の describe-security-group-rules コマンドを使用して、IsEgress が false である指定されたセキュリティグループのルールを表示します。

```
aws ec2 describe-security-group-rules \
```

```
    --filters Name=group-id,Values=sg-01dd3383691d02f42 \
    --query 'SecurityGroupRules[?IsEgress==`false`]'
```

以下は出力の例です。最初のルールはデフォルトのインバウンドルールで、このセキュリティグループに割り当てられたすべてのリソースからのインバウンドトラフィックを許可します。もう 1 つのルールは、指定されたプレフィックスリストの IP アドレスからの SSH トラフィックを許可します。プレフィックスリストの CIDR ブロックを記述するには、describe-prefix-lists コマンドを実行します。

```
[
    {
        "SecurityGroupRuleId": "sgr-06c8b42574a91db1b",
        "GroupId": "sg-01dd3383691d02f42",
        "GroupOwnerId": "123456789012",
        "IsEgress": false,
        "IpProtocol": "-1",
        "FromPort": -1,
        "ToPort": -1,
        "ReferencedGroupInfo": {
            "GroupId": "sg-01dd3383691d02f42",
            "UserId": "123456789012"
        },
        "Tags": [],
        "SecurityGroupRuleArn": "arn:aws:ec2:us-west-2:123456789012:security-group-
rule/sgr-06c8b42574a91db1b"
    },
    {
        "SecurityGroupRuleId": "sgr-0886a5d46afcd1758",
        "GroupId": "sg-01dd3383691d02f42",
        "GroupOwnerId": "123456789012",
        "IsEgress": false,
        "IpProtocol": "tcp",
        "FromPort": 22,
        "ToPort": 22,
        "PrefixListId": "pl-f8a6439125e7bf465",
        "Tags": [],
        "SecurityGroupRuleArn": "arn:aws:ec2:us-east-2:123456789012:security-group-
rule/sgr-0886a5d46afcd1758"
    }
]
```

セキュリティグループのアウトバウンドルールを取得するには

次の describe-security-group-rules コマンドを使用して、IsEgress が true である指定されたセキュリティグループのルールを表示します。

```
aws ec2 describe-security-group-rules \
    --filters Name=group-id,Values=sg-01dd3383691d02f42 \
    --query 'SecurityGroupRules[?IsEgress==`true`]'
```

以下は出力の例です。これには、すべてのアウトバウンド IPv4 トラフィックを許可するデフォルトのアウトバウンドルールのみが含まれます。

```
[
    {
        "SecurityGroupRuleId": "sgr-048f09a719247dce7",
        "GroupId": "sg-01dd3383691d02f42",
        "GroupOwnerId": "123456789012",
        "IsEgress": true,
        "IpProtocol": "-1",
        "FromPort": -1,
        "ToPort": -1,
        "CidrIpv4": "0.0.0.0/0",
        "Tags": [],
        "SecurityGroupRuleArn": "arn:aws:ec2:us-east-2:123456789012:security-group-
rule/sgr-048f09a719247dce7"
    }
]
```

ユーザーデータ

EC2 インスタンスを起動するときに、ユーザーデータを使用してシェルスクリプトをインスタンスに渡すことができます。ユーザーデータは base64 でエンコードされているため、スクリプトを読み取るにはユーザーデータをデコードする必要があることに注意してください。

これが重要な理由

インスタンスの設定の一環として起動時にコマンドを実行している場合は、機能的に同等のサーバーを設定する際にも同じタスクを実行する必要がある場合があります。

インスタンスのデコードされたユーザーデータを表示するには

次の describe-instance-attribute コマンドを使用します。base64 コマンドはユーザーデータをデコードします。

```
aws ec2 describe-instance-attribute \
    --instance-id i-1234567890abcdef0 \
    --attribute userData \
    --output text \
    --query "UserData.Value" | base64 --decode
```

以下は出力の例です。

```
#!/bin/bash
yum update -y
service httpd start
chkconfig httpd on
```

関連リソース

EC2 インスタンスの追加の特徴は次のとおりです。

- キーペア
- [Storage (ストレージ)]
- [タグ]

以下を使用して EC2 インスタンスを起動しているか、EC2 インスタンス間でトラフィックを分散しているかどうかを確認できます。

- 「Auto Scaling グループ」
- EC2 Fleet
- エラスティックロードバランシング

Amazon EC2 インスタンスの問題をトラブルシューティングする

次の手順とヒントはAmazon EC2 インスタンスの問題をトラブルシューティングする際に参考になります。

問題

- Amazon EC2 インスタンスの起動に関する問題のトラブルシューティング
- Amazon EC2 インスタンスの停止に関する問題のトラブルシューティング
- Amazon EC2 インスタンスの終了に関する問題のトラブルシューティング
- 到達できない Amazon EC2 インスタンスのトラブルシューティング
- Amazon EC2 Linux インスタンスへの接続に関する問題のトラブルシューティング
- ステータスチェックに失敗した Amazon EC2 Linux インスタンスをトラブルシューティングする
- Amazon EC2 Linux インスタンスを間違ったボリュームから起動した問題をトラブルシューティングする
- Amazon EC2 Windows インスタンスへの接続に関する問題のトラブルシューティング
- Amazon EC2 Windows インスタンスの開始に関する問題のトラブルシューティング
- Amazon EC2 Windows インスタンスの問題のトラブルシューティング
- ネットワーク経由の Windows インスタンスのカーネルデバッギング
- Amazon EC2 Windows インスタンスの Windows 管理者パスワードをリセットする
- Amazon EC2 Windows インスタンスの Sysprep の問題をトラブルシューティングする
- EC2Rescue を使用して問題のある Amazon EC2 Linux インスタンスのトラブルシューティングを行う
- EC2Rescue を使用して問題のある Amazon EC2 Windows インスタンスのトラブルシューティングを行う
- インスタンス用 EC2 シリアルコンソール
- 診断割り込みを送信して到達不能の Amazon EC2 インスタンスをデバッグする

Amazon EC2 インスタンスの起動に関する問題のトラブルシューティング

次は、Amazon EC2 インスタンスを起動する際の問題の解決に役立つトラブルシューティングのヒントです。

起動に関する問題

- 無効なデバイス名
- インスタンス制限の超過
- インスタンス容量の不足
- リクエストされた設定は現在サポートされていません。サポートされている設定については、ドキュメントを参照してください。
- インスタンスがすぐに終了する
- アクセス権限の不足
- Windows の起動直後に CPU 使用率が高い (Windows インスタンスのみ)
- IMDSv1 が有効化されたインスタンスの起動が失敗する

無効なデバイス名

説明

新しいインスタンスを起動しようとすると、`Invalid device name` *`device_name`* エラーが発生します。

原因

インスタンスを起動しようとしたときにこのエラーが発生した場合、リクエストの 1 つ以上のボリュームのために指定されたデバイス名に無効なデバイス名が含まれています。エラーの原因として以下が考えられます。

- デバイス名は、選択した AMI によって使用されている可能性があります。
- デバイス名は、ルートボリューム用に予約されている可能性があります。
- デバイス名は、リクエスト内の別のボリュームのために使用される可能性があります。
- デバイス名は、オペレーティングシステム向けに有効でない可能性があります。

ソリューション

問題を解決するには:

- デバイス名が、選択した AMI で使用されていないことを確認します。次のコマンドを実行して、AMI によって使用されるデバイス名を表示します。

```
aws ec2 describe-images --image-id ami-0abcdef1234567890 --query
 'Images[*].BlockDeviceMappings[].DeviceName'
```

- ルートボリューム用に予約されているデバイス名を使用していないことを確認します。詳細については、「使用できるデバイス名」を参照してください。
- リクエストで指定されている各ボリュームのデバイス名が一意であることを確認します。
- 指定したデバイス名が正しい形式となっていることを確認します。詳細については、「使用できるデバイス名」を参照してください。

インスタンス制限の超過

説明

新しいインスタンスを起動するか、停止したインスタンスを再起動しようとすると、InstanceLimitExceeded エラーが発生する。

原因

新しいインスタンスを起動するか、停止したインスタンスを再起動しようとすると InstanceLimitExceeded エラーが発生する場合、リージョンで起動できるインスタンス数の制限に達しています。AWS アカウントを作成するときに、リージョンごとに、実行できるインスタンスの数についてデフォルトの制限が設定されます。

ソリューション

インスタンス制限の引き上げは、リージョンごとにリクエストできます。詳細については、「Amazon EC2 の Service Quotas」を参照してください。

インスタンス容量の不足

説明

新しいインスタンスを起動するか、停止したインスタンスを再起動しようとすると、InsufficientInstanceCapacity エラーが発生する。

原因

インスタンスを起動したり、停止したインスタンスを再起動したりする際にこのエラーが発生する場合、現在 AWS にはリクエストに対応するために必要とされる十分なオンデマンドキャパシティーがありません。

ソリューション

この問題を解決するには、以下の手順を実行します。

- 数分間待ってからリクエストを再度送信します。容量は頻繁に変化します。
- インスタンス数を減らして新しいリクエストを送信します。たとえば、15 インスタンスを起動する 1 つのリクエストを行っている場合、代わりに 5 つのインスタンスに対する 3 つのリクエストを作成するか、1 つのインスタンスに対する 15 のリクエストを作成してみてください。
- インスタンスを起動する場合は、アベイラビリティーゾーンを指定しないで新しいリクエストを送信します。
- インスタンスを起動する場合は、別のインスタンスタイプを使用して新しいリクエストを送信します (これは後でサイズを変更できます)。詳細については、「Amazon EC2 インスタンスタイプの変更」を参照してください。
- クラスタープレイスメントグループにインスタンスを起動すると、容量不足エラーが発生する場合があります。

リクエストされた設定は現在サポートされていません。サポートされている設定については、ドキュメントを参照してください。

説明

インスタンス設定がサポートされていないため、新しいインスタンスを起動しようとすると、Unsupported エラーが表示されます。

原因

エラーメッセージには、詳細が記載されています。例えば、インスタンスタイプまたはインスタンス購入オプションは、指定されたリージョンまたはアベイラビリティーゾーンではサポートされていない可能性があります。

ソリューション

別のインスタンス設定を試してください。要件を満たすインスタンスタイプを検索するには、「Amazon EC2 インスタンスタイプの検索」を参照してください。

インスタンスがすぐに終了する

説明

インスタンスは `pending` 状態から `terminated` 状態に移行します。

原因

インスタンスがすぐに終了する理由を次にいくつか示します。

- EBS ボリュームの制限を超えた。詳細については、「Amazon EC2 インスタンスの Amazon EBS ボリューム制限」を参照してください。
- EBS スナップショットが破損している。
- ルート EBS ボリュームは暗号化されていて、復号用の KMS キー にアクセスする権限がない。
- AMI のブロックデバイスマッピングで指定されたスナップショットは暗号化されていて、復号するための KMS キー へのアクセス権限がないか、復元されたボリュームを暗号化するための KMS キー へのアクセス権限がありません。
- インスタンスを起動するために使用した Amazon S3-backed AMI で、必要な部分 (image.part.xx ファイル) が見つからない。

詳細については、次のいずれかの方法を使用して、削除された理由を確認します。

Amazon EC2 コンソールを使用して、削除された理由を確認するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Instances] を選択し、インスタンスを選択します。
3. 最初のタブで、[State transition reason (状態遷移の理由)] の横にある理由を見つけます。

AWS CLI を使用して、削除された理由を確認するには

1. describe-instances コマンドを使用して、インスタンス ID を指定します。

```
aws ec2 describe-instances --instance-id i-1234567890abcdef0
```

2. コマンドによって返された JSON レスポンスで、StateReason レスポンス要素の値を確認します。

   次のコードブロックは StateReason レスポンス要素の例を示しています。

```
"StateReason": {
  "Message": "Client.VolumeLimitExceeded: Volume limit exceeded",
  "Code": "Server.InternalError"
},
```

AWS CloudTrail を使用して、削除された理由を確認するには

詳細については、AWS CloudTrail ユーザーガイドの「CloudTrail イベント履歴でのイベントの表示」を参照してください。

ソリューション

削除の理由に応じて、次のアクションの 1 つを実行します。

- **`Client.VolumeLimitExceeded: Volume limit exceeded`** — 未使用のボリュームを削除します。ボリューム制限を引き上げるリクエストを送信できます。
- **`Client.InternalError: Client error on launch`** - ボリュームの復号と暗号化に使用する AWS KMS keys への必要なアクセス権限があることを確認します。詳細については、AWS Key Management Service デベロッパーガイドの「AWS KMS でのキーポリシーの使用」を参照してください。

アクセス権限の不足

説明

新しいインスタンスを起動しようとすると、`"errorMessage": "You are not authorized to perform this operation."` エラーが発生し、起動が失敗します。

原因

インスタンスを起動しようとしたときにこのエラーが表示される場合は、インスタンスを起動するために必要な IAM 権限が不足している可能性があります。

不足している可能性のある権限には次のものがあります。

- `ec2:RunInstances`
- `iam:PassRole`

その他のアクセス許可が不足している可能性もあります。インスタンスの起動に必要な権限のリストについては、例: EC2 インスタンス起動ウィザードの使用 および インスタンスの起動 (RunInstances) の下にある IAM ポリシーの例を参照してください。

ソリューション

問題を解決するには:

- IAM ユーザーとしてリクエストを発行する場合は、以下の条件が満たされていることを確認します。
  - `ec2:RunInstances` でリソースがワイルドカード (*) で定義されている
  - `iam:PassRole` でロールの ARN (`arn:aws:iam::999999999999:role/ExampleRoleName` など) に一致するリソースが定義されている
- 上記の権限がない場合は、IAM ロールまたはユーザーに関連付けられた IAM ポリシーを編集して、不足している必要な権限を追加してください。

問題が解決されず、起動失敗エラーが引き続き表示される場合は、エラーに含まれる認可失敗メッセージをデコードすることができます。デコードされたメッセージには、IAM ポリシーにない権限が含まれています。詳細については、「EC2 インスタンスの起動中に "UnauthorizedOperation" というエラーが発生した後、認可失敗のメッセージをデコードする方法」を参照してください。

Windows の起動直後に CPU 使用率が高い (Windows インスタンスのみ)

> **Note**
>
> このトラブルシューティングのヒントは、Windows インスタンスのみが対象です。

Windows Update が、[更新プログラムを確認するが、ダウンロードとインストールを行うかどうかは選択する] (デフォルトのインスタンス設定) に設定されている場合は、この確認によってインスタンスの CPU の 50 ~ 99% が消費される可能性があります。この CPU の消費によってアプリケーションの問題が発生する場合は、[コントロールパネル] で Windows Update の設定を手動で変更するか、または Amazon EC2 ユーザーデータフィールドで以下のスクリプトを使用できます。

```
reg add "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\WindowsUpdate\Auto Update" /v
 AUOptions /t REG_DWORD /d 3 /f net stop wuauserv net start wuauserv
```

このスクリプトを実行するときに /d の値を指定します。デフォルト値は 3 です。以下に示しているのは、可能な値です。

1. 更新プログラムを確認しない
2. 更新プログラムを確認するが、ダウンロードとインストールを行うかどうかは選択する
3. 更新プログラムをダウンロードするが、インストールを行うかどうかは選択する
4. 更新プログラムを自動的にインストールする

インスタンスのユーザーデータを変更したら、そのインスタンスを実行できます。詳細については、「起動時に Linux インスタンスでコマンドを実行する」を参照してください。

IMDSv1 が有効化されたインスタンスの起動が失敗する

説明

次のメッセージを伴う UnsupportedOperation 例外がスローされます。

```
You can't launch instances with IMDSv1 because httpTokensEnforced
is enabled for this account. Either launch the instance with
httpTokens=required or contact your account owner to disable
httpTokensEnforced using the ModifyInstanceMetadataDefaults API or the
account settings in the EC2 console.
```

原因

このエラーは、EC2 アカウント設定または AWS Organization 宣言型ポリシーが IMDSv2 の使用を強制適用する (`httpTokensEnforced = enabled`) アカウントで新しいインスタンスを IMDSv1 有効 (`httpTokens = optional`) として起動しようとするときにスローされます。

ソリューション

IMDSv2 のみを使用する準備ができたら、IMDSv1 を無効にして (`httpTokens = required`) インスタンスを起動します。準備が整っているかどうかを確認するには、「インスタンスメタデータサービスバージョン 2 の使用への移行」を参照してください。

新規または既存のインスタンスで IMDSv1 のサポートが引き続き必要な場合は、リージョン内のアカウントで IMDSv2 の強制適用を無効にする必要があります。IMDSv2 の強制適用を無効にするには、`HttpTokensEnforced` を `disabled` に設定します。詳細については、「Amazon EC2 API リファレンス」の「ModifyInstanceMetadataDefaults」を参照してください。この設定をコンソールを使用して行う場合は、「アカウントレベルで IMDSv2 を強制適用する」を参照してください。

Amazon EC2 インスタンスの停止に関する問題のトラブルシューティング

Amazon EBS バックアップされたインスタンスが `stopping` 状態でスタックしている場合、基になるホストコンピュータに問題がある可能性があります。

この問題を解決するには次の手順を実行します。

1. インスタンスを強制停止

   Amazon EC2 コンソールまたは AWS CLI を使用し、インスタンスを強制停止します。手順についてはインスタンスを強制停止するを参照してください。

   インスタンスはまず正常なシャットダウンを試みますが、これにはファイルシステムのキャッシュおよびメタデータのフラッシュが含まれます (オプションとして正常なシャットダウンをバイパスすることもできます)。タイムアウト期間内に正常なシャットダウンが完了しない場合、インスタンスはファイルシステムのキャッシュおよびメタデータをフラッシュせずに強制的にシャットダウンします。
2. 強制停止後

   ファイルシステムのチェックおよび修復の手順を実行します。

   > **Important**
   >
   > 強制停止するとファイルシステムのキャッシュおよびメタデータがフラッシュされないため、これらの手順を実行することは重要です。

3. 強制停止が失敗した場合

   10 分経ってもインスタンスが停止しない場合、次の操作を行います。

   a. 「AWS re:Post」でヘルプの依頼を投稿します。迅速な解決のために、インスタンス ID を含めて、既に行った手順について説明してください。
   b. また、サポートプランを契約している場合はサポートセンターでサポートケースを作成できます。
   c. 支援を待っている間、必要に応じて代替のインスタンスを作成できます。手順については(オプション) 代替インスタンスの作成を参照してください。

インスタンスが `stopping` 状態または `running` 以外の状態にある間はインスタンスの使用にコストがかかりません。インスタンスが `running` 状態のときのみ、インスタンスの使用量に対して課金されます。

内容

- インスタンスを強制停止する
- (オプション) 代替インスタンスの作成

インスタンスを強制停止する

インスタンスは強制的に停止できます。10 分経過してもインスタンスが停止しない場合、AWS re:Post にヘルプリクのエストを投稿してください。迅速な解決のために、インスタンス ID を含めて、既に行った手順について説明してください。また、サポートプランを契約している場合はサポートセンターでサポートケースを作成できます。

> **Note**
>
> コンソールを使用する場合、インスタンスを強制的に停止できるのはインスタンスが `stopping` 状態にある間のみです。AWS CLI を使用する場合は、インスタンスが `pending`、`running`、または `stopping` 状態にある間にインスタンスを強制的に停止できます。

Console

インスタンスを強制停止する

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインの [Instances (インスタンス)] を選択し、処理が止まってしまったインスタンスを選択してください。
3. [インスタンスの状態] と [インスタンスの強制停止] を選択します。

   [Force stop instance] (インスタンスの強制停止) はインスタンスが`stopping`状態である場合のみコンソールで利用できることに注意してください。インスタンスが別の状態の場合 (`shutting-down` と `terminated` を除く)はAWS CLI を使用してインスタンスを強制停止します。
4. (オプション) 強制停止中に正常な OS シャットダウンをバイパスするには、[OS シャットダウンをスキップ] チェックボックスをオンにします。
5. [強制停止] を選択します。

AWS CLI

インスタンスを強制停止する

run-instances コマンドを `--force` オプション共に使用します。

```
aws ec2 stop-instances \
    --instance-ids i-1234567890abcdef0 \
    --force
```

強制停止中に正常な OS シャットダウンをバイパスするには、「`--skip-os-shutdown`」オプションを含めます。

```
aws ec2 stop-instances \
    --instance-ids i-1234567890abcdef0 \
    --force \
    --skip-os-shutdown
```

PowerShell

インスタンスを強制停止する

Stop-EC2Instance コマンドレットを使用し、-Enforce を true に設定します。

```
Stop-EC2Instance `
    -InstanceId i-1234567890abcdef0 `
    -Enforce $true
```

強制停止中に正常な OS シャットダウンをバイパスするには、「-SkipOsShutdown $true」を含めます。

```
Stop-EC2Instance `
    -InstanceId i-1234567890abcdef0 `
    -Enforce $true `
    -SkipOsShutdown $true
```

(オプション) 代替インスタンスの作成

AWS re:Post またはサポートセンターからの支援を待っている間に、必要に応じて代わりのインスタンスを作成できます。処理が止まってしまったインスタンスから AMI を作成し、新しい AMI を使用して新しいインスタンスを起動します。

> **Important**
>
> インスタンスのステータスチェックを行うと、壊れたオペレーティングシステムの完全なレプリカが AMI にコピーされることになるため、処理が止まってしまったインスタンスがシステムのステータスチェックのみを生成する場合は代わりのインスタンスを作成することができます。ステータスメッセージを確認したら、AMI を作成し、新しい AMI を使用して新しいインスタンスを起動します。

Console

代替インスタンスを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインの [Instances (インスタンス)] を選択し、処理が止まってしまったインスタンスを選択してください。
3. [アクション]、[Image and templates (イメージとテンプレート)]、[イメージの作成] の順に選択してください。

4. [イメージの作成] ページで、次の操作を行います。

   a. AMI の名前と説明を入力してください。

   b. [インスタンスを再起動] をクリアします。

   c. [イメージを作成] を選択してください。

   詳細についてはthe section called “インスタンスから AMI を作成する”を参照してください。

5. AMI から新しいインスタンスを起動し、その新しいインスタンスが動作していることを確認します。

6. 処理が止まってしまったインスタンスを選択し、[アクション]、[インスタンスの状態]、[インスタンスの終了 (削除)] の順に選択してください。インスタンスの終了処理も止まってしまう場合はAmazon EC2 は数時間以内に自動的にそのインスタンスを強制終了します。

前の手順で説明されたように、インスタンスから AMI を作成できない場合は次のようにして代わりのインスタンスを設定できます。

(代替方法) コンソールを使用して代わりのインスタンスを作成するには

1. インスタンスを選択し、[Description (説明)]、[Block devices (ブロックデバイス)] の順に選択してください。各ボリュームを選択し、そのボリューム ID を書き留めます。必ずどのボリュームがルートボリュームであるかメモしておきます。

2. ナビゲーションペインの [Volumes] を選択してください。インスタンスの各ボリュームを選択し、[アクションs]、[Create Snapshot] の順に選択してください。

3. ナビゲーションペインで、[Snapshots] を選択してください。作成したスナップショットを選択し、[アクションs]、[Create Volume] の順に選択してください。

4. 処理が止まってしまったインスタンスと同じオペレーティングシステムのインスタンスを起動します。そのルートボリュームのボリューム ID とデバイス名をメモしておきます。

5. ナビゲーションペインで、[Instances] を選択し、起動したインスタンスを選択した後で、[Instance state]、[Stop instance] の順に選択してください。

6. ナビゲーションペインで [Volumes] を選択し、停止したインスタンスのルートボリュームを選択した後で、[アクションs]、[Detach Volume] の順に選択してください。

7. 処理が停止してしまったインスタンスから作成したルートボリュームを選択し、[アクションs]、[Attach Volume] の順に選択して、そのルートボリュームとして新しいインスタンスにアタッチします (書き留めたデバイス名を使用)。その他の非ルートボリュームをインスタンスにアタッチします。

8. ナビゲーションペインで、[Instances] を選択し、代わりのインスタンスを選択してください。[インスタンスの状態]、[Start instance (インスタンスの開始)] の順に選択してください。インスタンスが動作していることを確認します。
9. 処理が止まったインスタンスを選択し、[インスタンスの状態]、[インスタンスの終了 (削除)] の順に選択してください。インスタンスの終了処理も止まってしまう場合はAmazon EC2 は数時間以内に自動的にそのインスタンスを強制終了します。

AWS CLI

代替インスタンスを作成するには

1. create-image コマンドを --no-reboot オプションと共に使用して、スタックしたインスタンスから AMI を作成します。

```
aws ec2 create-image \
    --instance-id i-1234567890abcdef0 \
    --name "my-replacement-ami" \
    --description ""AMI for replacement instance" \
    --no-reboot
```

2. run-instances コマンドを使用して、先ほど作成した AMI から新しいインスタンスを起動します。
3. 新しいインスタンスが動作していることを確認します。
4. (オプション) terminate-instances コマンドを使用して、スタックしたインスタンスを終了します。

```
aws ec2 terminate-instances --instance-ids i-1234567890abcdef0
```

PowerShell

代替インスタンスを作成するには

1. New-EC2Image コマンドレットを使用してスタックしたインスタンスから AMI を作成し、-NoReboot を true に設定します。

```
New-EC2Image `
    -InstanceId i-1234567890abcdef0 `
    -Name "my-replacement-ami" `
```

```
    -Description "AMI for replacement instance" `
    -NoReboot $true
```

2. New-EC2Instance コマンドレットを使用して、先ほど作成した AMI から新しいインスタンスを起動します。
3. 新しいインスタンスが動作していることを確認します。
4. (オプション) Remove-EC2Instance コマンドレットを使用して、スタックしたインスタンスを終了します。

```
Remove-EC2Instance -InstanceId i-1234567890abcdef0
```

Amazon EC2 インスタンスの終了に関する問題のトラブルシューティング

インスタンスをシャットダウンまたは削除することは、インスタンスの終了と呼ばれます。以下の情報は、インスタンスを終了するときの問題のトラブルシューティングに役立ちます。

インスタンスが running 状態ではない場合は、インスタンスの使用に対して課金されません。つまり、インスタンスを終了させると、そのステータスが shutting-down に変わるとすぐに、そのインスタンスへの課金は停止します。

インスタンスがすぐに終了する

複数の問題により、起動時にインスタンスがすぐに終了する可能性があります。詳細については、「インスタンスがすぐに終了する」を参照してください。

インスタンスの削除の遅延

インスタンスの shutting-down 状態が数分以上続く場合、次の原因が考えられます。

- インスタンスはシャットダウンスクリプトを実行しています。
- 基盤となるホストコンピュータに問題があります。

shutting-down 状態が数時間以上続いた後、Amazon EC2 によってインスタンスはスタック状態として扱われて強制終了されます。

スタックしたインスタンスを自分で解決するには：

1. インスタンスを強制終了する

   Amazon EC2 コンソールまたは AWS CLI を使用し、インスタンスを強制停止します。手順についてはインスタンスを強制終了するを参照してください。

   インスタンスはまず正常なシャットダウンを試みますが、これにはファイルシステムのキャッシュおよびメタデータのフラッシュが含まれます (オプションとして正常なシャットダウンをバイパスすることもできます)。タイムアウト期間内に正常なシャットダウンが完了しない場合、インスタンスはファイルシステムのキャッシュおよびメタデータをフラッシュせずに強制的にシャットダウンします。

2. 強制終了が失敗した場合

   数時間経ってもインスタンスが終了しておらず、終了処理でスタックした場合、次の操作を行います。

   a. 「AWS re:Post」でヘルプの依頼を投稿します。迅速な解決のために、インスタンス ID を含めて、既に行った手順について説明してください。

   b. また、サポートプランを契約している場合はサポートセンターでサポートケースを作成できます。

インスタンスを強制終了する

インスタンスの終了が停止しているように見える場合は、インスタンスを強制的に終了させることができます。数時間経過してもインスタンスが終了していない場合は、AWSre:Post にヘルプリクエストを投稿してください。迅速な解決のために、インスタンス ID を含めて、既に行った手順について説明してください。また、サポートプランを契約している場合は、サポートセンターでサポートケースを作成できます。

Console

インスタンスを強制終了するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインの [Instances (インスタンス)] を選択し、処理が止まってしまったインスタンスを選択してください。
3. [Instance state (インスタンスの状態)]、[Force terminate instance (インスタンスの強制終了)] の順に選択してください。

[Force stop instance (インスタンスの強制停止)] はインスタンスがstopping状態である場合のみコンソールで利用できることに注意してください。インスタンスが別の状態の場合 (shutting-down と terminated を除く)は、AWS CLI を使用してインスタンスを強制停止します。

4. (オプション) 強制終了中に正常な OS シャットダウンをバイパスするには、[OS シャットダウンをスキップ] チェックボックスをオンにします。
5. [強制終了] を選択します。

AWS CLI

インスタンスを強制終了するには

--force オプションで terminate-instances コマンドを使用します。

```
aws ec2 terminate-instances \
    --instance-ids i-1234567890abcdef0 \
    --force
```

強制終了中に正常な OS シャットダウンをバイパスするには、「--skip-os-shutdown」オプションを含めます。

```
aws ec2 terminate-instances \
    --instance-ids i-1234567890abcdef0 \
    --force \
    --skip-os-shutdown
```

PowerShell

インスタンスを強制終了するには

Remove-EC2Instance コマンドレットを使用し、-Enforce を true に設定します。

```
Remove-EC2Instance `
    -InstanceId i-1234567890abcdef0 `
    -Enforce $true
```

強制終了中に正常な OS シャットダウンをバイパスするには、「-SkipOsShutdown $true」を含めます。

```
Remove-EC2Instance `
    -InstanceId i-1234567890abcdef0 `
    -Enforce $true `
    -SkipOsShutdown $true
```

表示されているインスタンスを削除する

インスタンスの削除後、インスタンスはしばらくの間削除されずに表示されたままとなります。状態は terminated となります。このエントリが数時間経過しても削除されない場合には、サポートに連絡してください。

エラー: インスタンスは終了できない可能性があります。その「disableApiTermination」インスタンス属性を変更します

インスタンスを終了しようとしたときに The instance *i-1234567890abcdef0* may not be terminated. Modify its 'disableApiTermination' instance attribute エラーメッセージが表示される場合は、そのインスタンスの終了保護が有効になっていることを示します。削除保護はインスタンスが誤って削除されないようにします。

インスタンスを終了する前に、終了保護を無効にする必要があります。

詳細については、「インスタンスの終了保護を変更する」を参照してください。

インスタンスが自動的に起動または終了される

通常、以下の動作は、定義した基準に基づいて自動的にコンピューティングリソースをスケールするため、Amazon EC2 Auto Scaling、EC2 フリート、またはスポットフリートを使用していることを意味します。

- インスタンスを終了すると、別のインスタンスが自動的に起動します。
- インスタンスを起動すると、いずれかのインスタンスが自動的に終了します。
- インスタンスを停止すると、そのインスタンスが終了し、別のインスタンスが自動的に起動します。

自動スケーリングを停止するには、インスタンスを起動している Auto Scaling グループまたはフリートを見つけて、その容量を 0 に設定するか、削除します。

到達できない Amazon EC2 インスタンスのトラブルシューティング

以下の情報は到達できない Amazon EC2 インスタンスのトラブルシューティングに役立ちます。問題の診断に役立つようにスクリーンショットをキャプチャするかコンソール出力にアクセスし、インスタンスを再起動する必要があるかどうか判断することができます。到達できない Windows インスタンスの場合はサービスから返されたスクリーンショットを確認してトラブルシューティングを行います。

内容

- インスタンスの再起動
- インスタンスコンソール出力
- 接続できないインスタンスのスクリーンショットの取得
- 到達できない Windows インスタンスをトラブルシューティングするための一般的なスクリーンショット
- ホストコンピュータに障害が発生した場合のインスタンスの復旧
- インスタンスがオフラインになり、予期せず再起動された

インスタンスの再起動

トラブルシューティングにも一般的なインスタンス管理にも、到達できないインスタンスを再起動する方法が重要です。

リセットボタンを押してコンピュータをリセットするように、Amazon EC2 コンソール、CLI、または API を使用して EC2 インスタンスをリセットできます。詳細については Amazon EC2 インスタンスを再起動する を参照してください。

インスタンスコンソール出力

コンソール出力は問題を診断する際に役立つツールで、特に、カーネルの問題やサービス設定の問題のトラブルシューティングを行うときに便利です。これらの問題が発生すると、SSH デーモンの開始前にインスタンスが停止したり、インスタンスに到達不能になったりする可能性があります。

- Linux インスタンス – コンピュータに接続されている物理的なモニタに通常表示されるものとまったく同じコンソール出力がインスタンスコンソール出力に表示されます。コンソール出力はインスタンス遷移状態 (開始、停止、再起動、終了) の直後に投稿されたバッファされた情報を返しま

す。表示される出力は継続的には更新されず、更新する価値があると思われる場合にのみ更新されます。

- Windows インスタンス – インスタンスコンソール出力には直近のシステムイベントログエラーが 3 件含まれます。

インスタンスの所有者のみがコンソール出力にアクセスできます。

インスタンスのライフサイクル中に最新のシリアルコンソールの出力を取得できます。このオプションはNitro ベースのインスタンスでのみサポートされています。

Console

コンソール出力を取得するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左のナビゲーションペインの [インスタンス] を選択してください。
3. インスタンスを選択してください。
4. [アクション]、[モニタリングとトラブルシューティング]、[システムログの取得] の順に選択します。

AWS CLI

コンソール出力を取得するには

get-console-output コマンドを使用します。

```
aws ec2 get-console-output --instance-id i-1234567890abcdef0
```

PowerShell

コンソール出力を取得するには

Get-EC2ConsoleOutput コマンドレットを使用します。

```
Get-EC2ConsoleOutput -InstanceId i-1234567890abcdef0
```

接続できないインスタンスのスクリーンショットの取得

インスタンスに接続できない場合はインスタンスのスクリーンショットをキャプチャして、それをイメージとして表示することができます。このイメージにより、インスタンスのステータスについて可視化されるため、迅速にトラブルシューティングすることができます。

インスタンスの実行中またはクラッシュ後にスクリーンショットを生成できます。イメージは JPG 形式で生成され、100 KB 未満です。スクリーンショットにはデータ転送コストがかかりません。

制限事項

この機能は以下の場合はサポートされません。

- ベアメタルインスタンス (タイプ `*.metal` のインスタンス)
- インスタンスで NVIDIA GRID ドライバーが使用されている
- Arm ベースの Graviton プロセッサを搭載したインスタンス
- AWS Outposts 上の Windows インスタンス
- AWS Local Zones 上の Windows インスタンス

リージョンのサポート

この機能は次のリージョンでは利用できません。

- アジアパシフィック (タイ)
- メキシコ (中部)
- GovCloud リージョン

Console

インスタンスのスクリーンショットを取得するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. 左のナビゲーションペインの [インスタンス] を選択してください。
3. キャプチャするインスタンスを選択してください。
4. [アクション]、[モニタリングとトラブルシューティン]、[インスタンスのスクリーンショットの取得] の順に選択してください。

5. [ダウンロード] を選択するか、イメージを右クリックしてダウンロードして保存します。

AWS CLI

インスタンスのスクリーンショットをキャプチャするには

get-console-screenshot コマンドを使用します。出力は base64 でエンコードされます。

```
aws ec2 get-console-screenshot --instance-id i-1234567890abcdef0
```

PowerShell

インスタンスのスクリーンショットをキャプチャするには

Get-EC2ConsoleScreenshot コマンドレットを使用します。出力は base64 でエンコードされます。

```
Get-EC2ConsoleScreenshot -InstanceId i-1234567890abcdef0
```

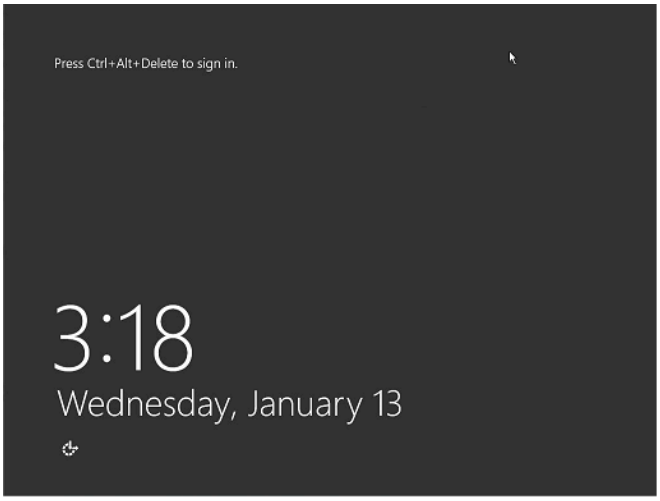
到達できない Windows インスタンスをトラブルシューティングするための一般的なスクリーンショット

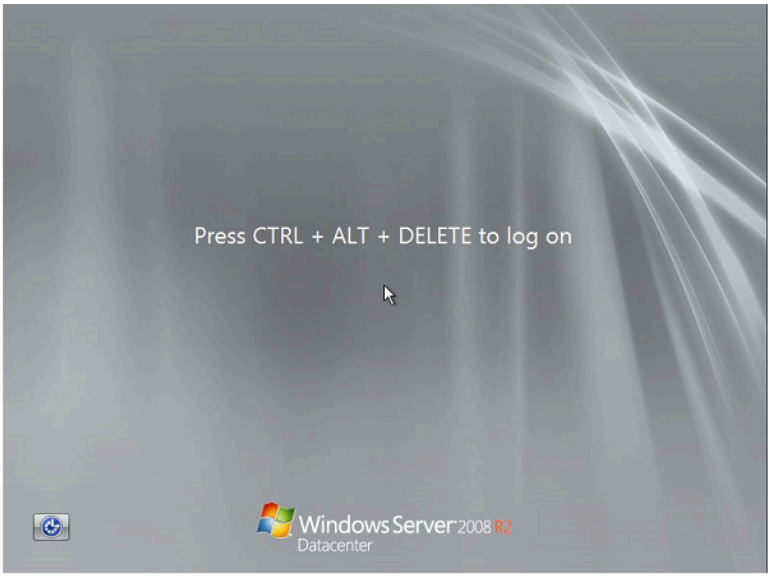
以下の情報を使用して、サービスによって返されたスクリーンショットに基づいて、到達不能な Windows インスタンスをトラブルシューティングすることができます。

- ログイン画面 (Ctrl+Alt+Delete)
- リカバリコンソール画面
- Windows Boot Manager 画面
- Sysprep 画面
- Getting Ready 画面
- Windows 更新 画面
- Chkdsk

ログイン画面 (Ctrl+Alt+Delete)

コンソールスクリーンショットサービスにより以下の内容が返されました。





インスタンスがログイン時に到達不可能になった場合はネットワーク設定または Windows リモートデスクトップサービスに問題がある可能性があります。プロセスが大量の CPU を使用している場合、インスタンスも応答しない可能性があります。

ネットワーク構成

次の情報を使用して、AWS、Microsoft Windows、およびローカル (またはオンプレミス) ネットワーク設定がインスタンスへのアクセスをブロックしていないことを確認します。

AWS ネットワーク設定

設定	検証
セキュリティグループの構成	ポート 3389 がセキュリティグループに対して開かれていることを確認します。適切なパブリック IP アドレスに接続していることを確認します。インスタンスが Elastic IP に関連付けられていない場合、パブリック IP アドレスはインスタンスが停止/起動した後に変更されます。詳細についてはリモートデスクトップからリモートコンピュータに接続できませんを参照してください。
VPC 設定 (ネットワーク ACL)	Amazon VPC のアクセスコントロールリスト (ACL) がアクセスをブロックしていないことを確認します。詳細についてはAmazon VPC

設定	検証
	ユーザーガイドの「ネットワーク ACL」を参照してください。
VPN の設定	仮想プライベートネットワーク (VPN) を使用して VPC に接続している場合、VPN トンネル接続を確認します。詳細については、「Troubleshooting AWS Client VPN: Tunnel connectivity issues to a VPC」を参照してください。

Windows ネットワーク設定

設定	検証
Windows ファイアウォール	Windows ファイアウォールがインスタンスへの接続をブロックしていないことを確認します。リモートデスクトップのトラブルシューティングセクション「リモートデスクトップからリモートコンピュータに接続できません」にある 7 番目の項目で説明されている手順に従って、Windows ファイアウォールを無効にします。
高度な TCP/IP の設定 (静的 IP を使用)	静的 IP アドレスを設定したため、インスタンスが応答しない可能性があります。VPC の場合、ネットワークインターフェイスを作成して、インスタンスにアタッチします。

ローカルまたはオンプレミスのネットワーク設定

ローカルネットワーク設定がアクセスをブロックしていないことを確認します。インスタンスが到達不可能なため、同じ VPC 内の別のインスタンスに接続を試みます。別のインスタンスにアクセスできない場合、ローカルポリシーでアクセスが制限されていないかどうかをローカルネットワーク管理者に確認してください。

リモートデスクトップサービスの問題

ログイン時にインスタンスに接続できない場合はインスタンスのリモートデスクトップサービス (RDS) に問題が存在する可能性があります。

> **Tip**
>
> AWSSupport-TroubleshootRDP Runbook を使用して、リモートデスクトッププロトコル (RDP) での接続に影響を与える可能性のある各種設定をチェックし、変更します。詳細についてはAWS Systems Manager「 自動 ランブックリファレンス」のAWSSupport-TroubleshootRDPを参照してください。

リモートデスクトップサービスの設定

設定	検証
RDS が実行されている	インスタンスで RDS が実行中であることを確認します。Microsoft 管理コンソールの MMC) サービススナップイン (services.msc) を使用してインスタンスに接続します。サービスのリストで、[リモートデスクトップサービス] が [実行中] であることを確認します。そうでない場合はサービスを開始し、スタートアップの種類を [自動] に設定します。サービススナップインを使用してインスタンスに接続できない場合はルートボリュームをインスタンスからデタッチして、ボリュームのスナップショットを取得するか AMI を作成し、元のボリュームを同じアベイラビリティーゾーンの別のインスタンスにセカンダリボリュームとしてアタッチして、Start レジストリキーを変更します。完了したら、元のインスタンスにルートボリュームを再アタッチします。
RDS が有効である	サービスが開始された場合でも、無効になることがあります。ルートボリュームをインスタンスからデタッチして、ボリュームからスナップショットを取得するか AMI を作成し、元のボリュームを同じアベイラビリティーゾーンの別のインスタンスにセカンダリボリュームとしてアタッチして、「リモートレジストリを使用して EC2 インスタンスでリモートデスクトップを有効にする」で説明されているように [Terminal サーバー (ターミ

設定	検証
	ナルサーバー)] レジストリキーを変更してサービスを有効にします。 完了したら、元のインスタンスにルートボリュームを再アタッチします。

高い CPU 使用率

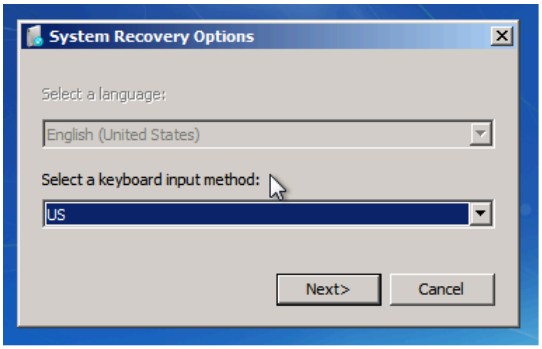
Amazon CloudWatch を使用して [CPUUtilization (Maximum)] メトリクスを確認します。[CPUUtilization (Maximum)] の数値が大きい場合、CPU 使用率が低下するまで待ってから再度接続してみてください。CPU 使用率が高い理由として考えられるものを以下に示します。

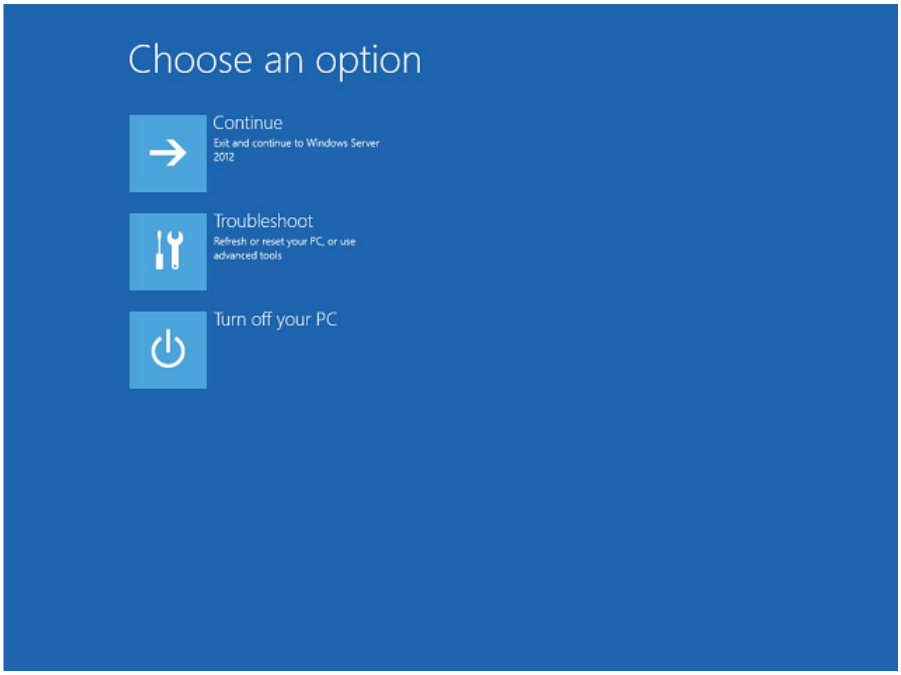
- Windows 更新
- セキュリティソフトウェアによるスキャン
- カスタム起動スクリプト
- タスクスケジューラ

詳細についてはAmazon CloudWatch ユーザーガイドの「特定のリソースの統計を取得する」を参照してください。トラブルシューティングのヒントについてはWindows の起動直後に CPU 使用率が高い (Windows インスタンスのみ)を参照してください。

リカバリコンソール画面

コンソールスクリーンショットサービスにより以下の内容が返されました。





bootstatuspolicy が ignoreallfailures に設定されていない場合、オペレーティングシステムがリカバリコンソールで起動し、この状態でスタックする可能性があります。bootstatuspolicy 設定を ignoreallfailures に変更するには次の手順を使用します。

デフォルトではAWS が提供するパブリック Windows AMI のポリシー設定は ignoreallfailures に設定されています。

1. 接続できないインスタンスを停止します。
2. ルートボリュームのスナップショットを作成します。ルートボリュームが /dev/sda1 としてインスタンスにアタッチされます。

   接続できないインスタンスからルートボリュームをデタッチして、ボリュームのスナップショットを取得するか AMI を作成し、同じアベイラビリティーゾーン内の別のインスタンスにセカンダリボリュームとしてアタッチします。

   > **Warning**
   >
   > 一時インスタンスと元のインスタンスが同じ AMI を使用して起動された場合、追加の手順を完了する必要があります。この手順を実行しないと、ディスク署名の競合によって、ルートボリュームを復元した後、元のインスタンスを起動できなくなります。同じ AMI を使用して一時インスタンスを作成する必要がある場合はディスク署名の競合を回避するため、ディスク署名の衝突 の手順を完了します。

または一時インスタンスとして別の AMI を選択してください。例えば、元のインスタンスで Windows サーバー 2016 用の AMI を使用している場合、Windows サーバー 2019 用の AMI を使用して一時インスタンスを起動します。

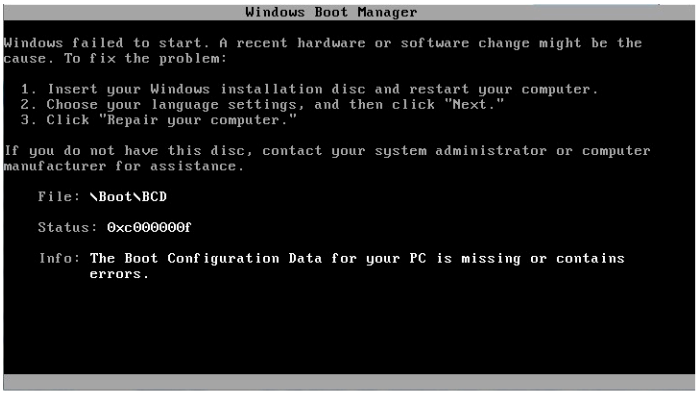
3. インスタンスにログインし、コマンドプロンプトから次のコマンドを実行して bootstatuspolicy 設定を ignoreallfailures に変更します。

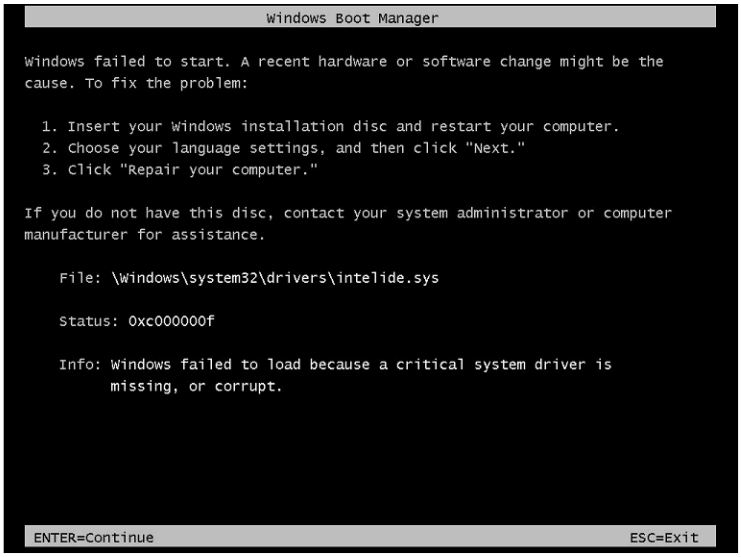
```
bcdedit /store Drive Letter:\boot\bcd /set {default} bootstatuspolicy
 ignoreallfailures
```

4. ボリュームを接続できないインスタンスに再アタッチし、インスタンスを再起動します。

Windows Boot Manager 画面

コンソールスクリーンショットサービスにより以下の内容が返されました。

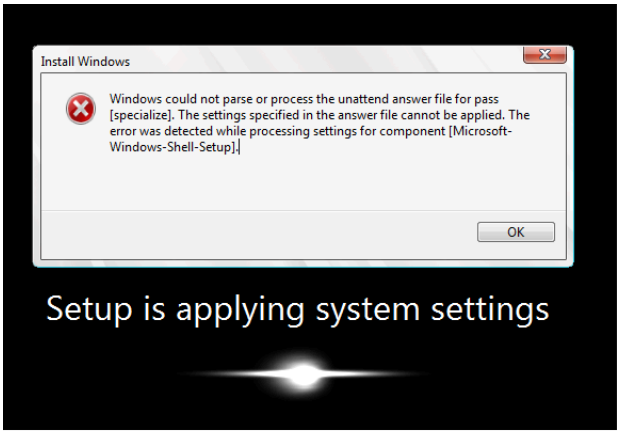


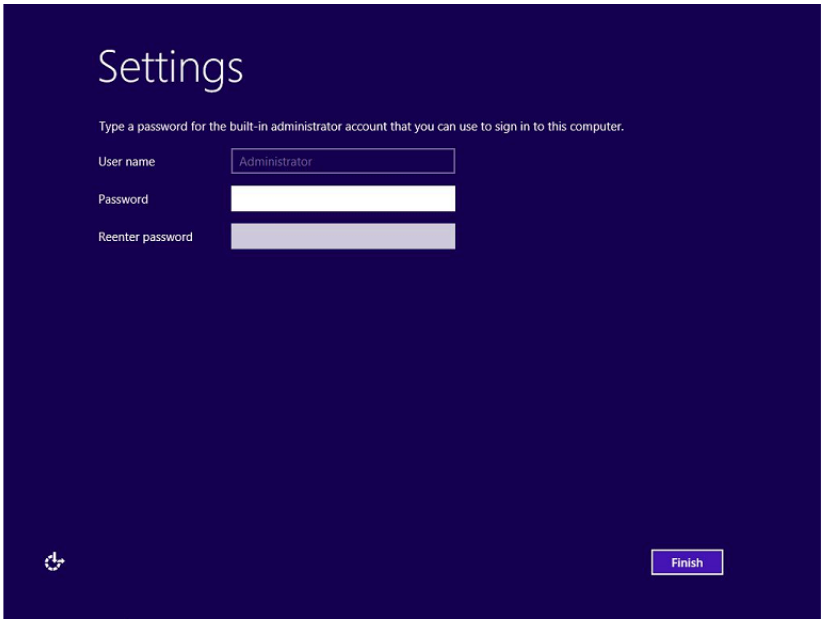


オペレーティングシステムのファイルシステムやレジストリで致命的な破損が発生しました。インスタンスがこの状態でスタックした場合、最新のバックアップ AMI からインスタンスを復旧するか、代わりのインスタンスを起動する必要があります。インスタンスのデータにアクセスする必要がある場合、接続できないインスタンスからルートボリュームをデタッチして、それらのボリュームからスナップショットを取得するか AMI を作成し、同じアベイラビリティーゾーン内の別のインスタンスにセカンダリボリュームとしてアタッチします。

Sysprep 画面

コンソールスクリーンショットサービスにより以下の内容が返されました。

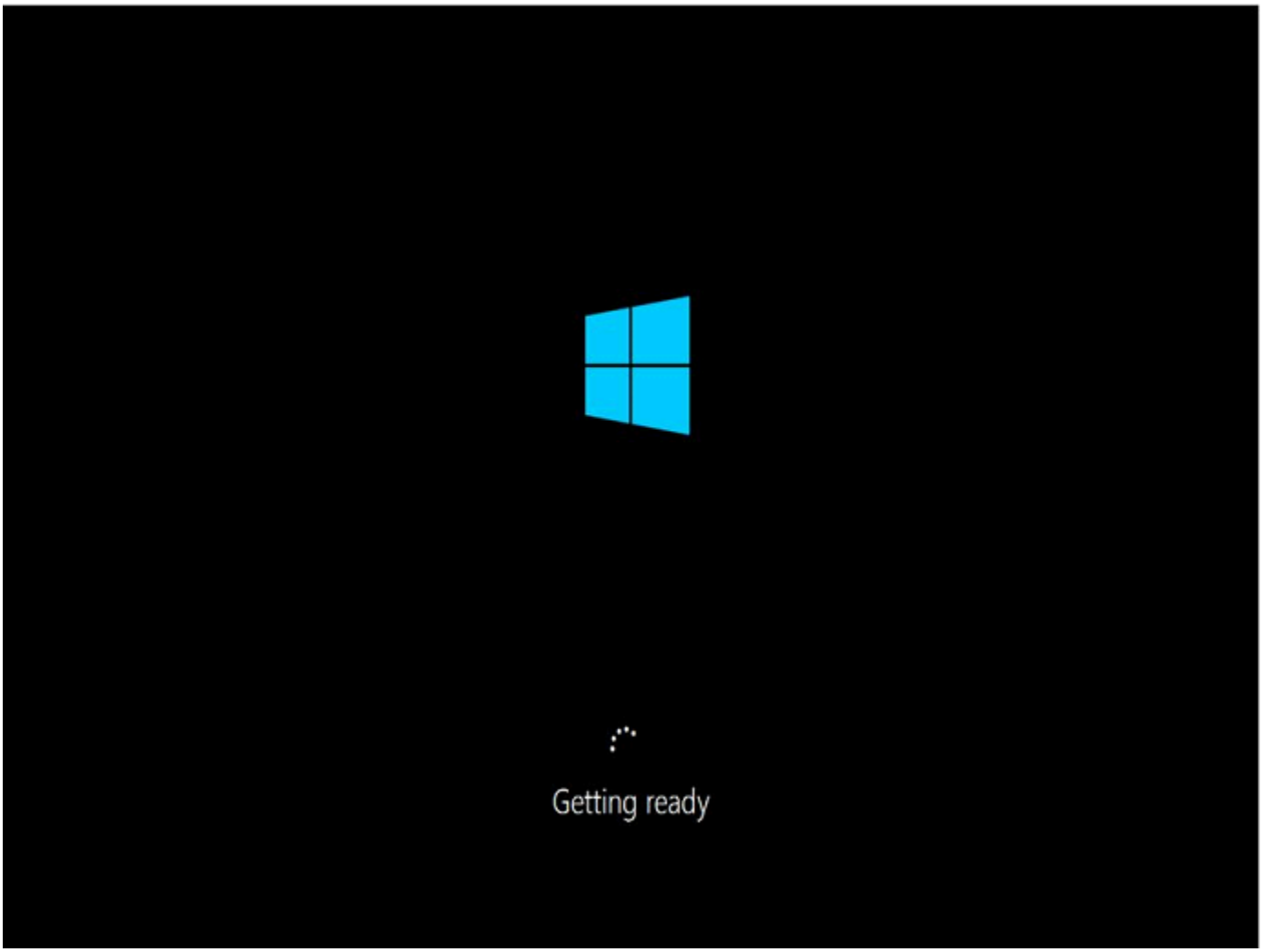




この画面はSysprep の呼び出しに EC2Config サービスを使用していない場合、または Sysprep の実行中にオペレーティングシステムでエラーが発生した場合に表示されることがあります。パスワードはEC2Rescue を使用してリセットすることが可能です。・ それ以外の場合はWindows Sysprep を使用して Amazon EC2 AMI を作成するを参照してください。

Getting Ready 画面

コンソールスクリーンショットサービスにより以下の内容が返されました。



インスタンスコンソールスクリーンショットサービスを繰り返し更新し、進捗状況のリングが回っていることを確認します。リングが回っている場合、オペレーティングシステムが起動するまで待ちます。Amazon CloudWatch を使用してオペレーティングシステムがアクティブであるかどうかを確認することにより、インスタンスの [CPUUtilization (Maximum)] メトリクスも確認します。進行状況のリングが回っていない場合、インスタンスは起動プロセス中にスタックしている可能性があります。インスタンスを再起動します。再起動で問題を解決できない場合はインスタンスを最新のバックアップ AMI から復旧するか、代わりのインスタンスを起動します。インスタンス上のデータにアクセスする必要がある場合、ルートボリュームを接続できないインスタンスからデタッチし、ボリュームのスナップショットを取得するか、AMI を作成します。その後、同じアベイラビリティーゾーン内の別のインスタンスにセカンダリボリュームとしてアタッチします。

Windows 更新 画面

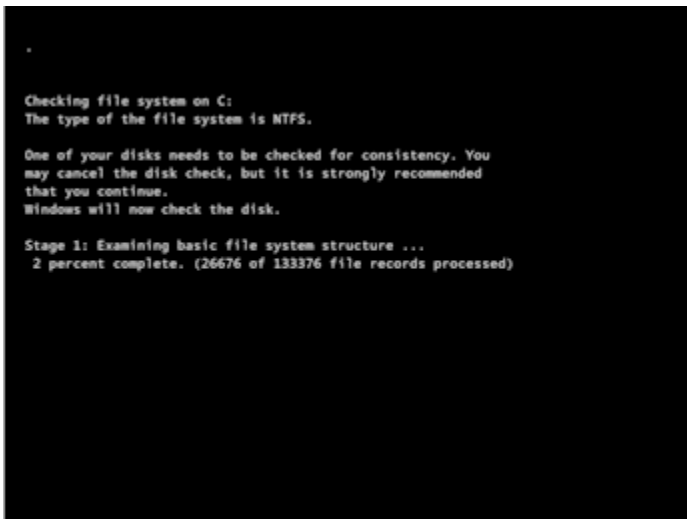
コンソールスクリーンショットサービスにより以下の内容が返されました。

Windows 更新 プロセスがレジストリを更新中です。更新が終了するまで待機してください。更新時にデータが破損する可能性があるため、インスタンスを再起動したり停止したりしないでください。

> **Note**
>
> Windows 更新 プロセスは更新中のサーバーのリソースを消費することがあります。この問題が頻繁に発生する場合、高速なインスタンスタイプと高速な EBS ボリュームを使用することを検討します。

Chkdsk

コンソールスクリーンショットサービスにより以下の内容が返されました。



Windows がドライブで chkdsk ツールを実行し、ファイルシステムの完全性を確認し、論理的なファイルシステムのエラーを修正しています。プロセスが完了するまで待ちます。

ホストコンピュータに障害が発生した場合のインスタンスの復旧

基になるホストコンピュータのハードウェアで復旧不可能な問題が発生した場合、AWS はインスタンスの停止イベントをスケジュールすることがあります。このようなイベントは事前に E メールで通知されます。

障害が発生したホストコンピュータで実行されている Amazon EBS-backed インスタンスを復旧するには

1. インスタンスストアボリュームの重要なデータを Amazon EBS または Amazon S3 にバックアップします。
2. インスタンスを停止します。
3. インスタンスを起動します。
4. 重要なデータを復元します。

詳細については、「Amazon EC2 インスタンスの停止と開始」を参照してください。

障害が発生したホストコンピュータで実行されている、インスタンスストアのルートボリュームを持つインスタンスを復旧するには

1. インスタンスから AMI を作成します。
2. イメージを Amazon S3 にアップロードします。
3. 重要なデータを Amazon EBS または Amazon S3 にバックアップします。
4. インスタンスを終了します。
5. AMI から新しいインスタンスを起動します。
6. 重要なデータを新しいインスタンスに復元します。

インスタンスがオフラインになり、予期せず再起動された

インスタンスがオフラインになっていて、予期せず再起動されたように見える場合はインスタンスの自動復旧が実行されている可能性があります。これは が基盤となるハードウェアまたはソフトウェアの問題によりインスタンスが使用できないことをAWS検出し、インスタンスで簡易自動復旧または CloudWatch アクションベースの復旧が有効になっている場合に発生します。

復旧プロセス中に、 は別のハードウェアに移行することで、インスタンスの可用性の復元AWSを試みます。インスタンスで自動インスタンス復旧が発生したかどうかを確認するには自動インスタンス復旧が発生したかどうかを確認するを参照してください。

> **Note**
>
> ワークロードまたはアプリケーションが応答しない場合はインスタンスで実行されているかどうかを確認します。そうでない場合は手動で起動します。今後この問題を回避するには復旧計画を実装して、インスタンスの復旧後にワークロードまたはアプリケーションが適切に機能するようにします。

Amazon EC2 Linux インスタンスへの接続に関する問題のトラブルシューティング

以下の情報と一般的なエラーはLinux インスタンスへの接続に関するトラブルシューティングに役立ちます。

接続の問題

- 接続の問題の一般的な原因
- インスタンスへの接続エラー: 接続タイムアウト
- エラー: キーを読み込めません..。期待: 任意のプライベートキー
- エラー: ユーザーキーがサーバーによって認識されない
- エラー: アクセス許可が拒否されたか、[インスタンス] ポート 22 によって接続が閉じられました。
- エラー: Unprotected Private キー ファイル (保護されていないプライベートキーファイル)
- エラー: プライベートキーの先頭は「-----BEGIN RSA PRIVATE KEY-----」、末尾は「-----END RSA PRIVATE KEY-----」にする必要があります
- エラー: ホストキーの検証に失敗しました
- エラー: サーバー refused our key または No supported authentication methods available (サーバーはキーを拒否しましたまたは利用可能なサポートされる認証方法はありません)
- インスタンスに対して ping を実行できない
- エラー: サーバーによる予期しないネットワーク接続の閉鎖
- エラー: EC2 インスタントコネクト のホストキーの検証に失敗しました

- EC2 インスタントコネクト を使用して Unbntu インスタンスに接続できない
- プライベートキーを紛失しました。自分のインスタンスに接続するにはどうすればよいですか?

接続の問題の一般的な原因

次のタスクを正確に実行したことを確認して、インスタンス接続の問題のトラブルシューティングを開始することをお勧めします。

インスタンスのユーザー名を確認する

インスタンスに接続するにはユーザーアカウントのユーザー名、またはインスタンスの起動に使用した AMI のデフォルトのユーザー名を使用します。

- ユーザーアカウントのユーザー名を取得します。

  ユーザーアカウントの作成方法については Amazon EC2 Linux インスタンスのシステムユーザーを管理する を参照してください。
- インスタンスの起動に使用した AMI のデフォルトのユーザー名を取得します。

インスタンスの起動に使用される AMI	デフォルトユーザー名
Amazon Linux	`ec2-user`
CentOS	`centos` または `ec2-user`
Debian	`admin`
Fedora	`fedora` または `ec2-user`
FreeBSD	`ec2-user`
RHEL	`ec2-user` または `root`
SUSE	`ec2-user` または `root`
Ubuntu	`ubuntu`
Oracle	`ec2-user`
Bitnami	`bitnami`

インスタンスの起動に使用される AMI	デフォルトユーザー名
Rocky Linux	`rocky`
その他	AMI プロバイダーに確認してください。

セキュリティグループルールでトラフィックが許可されていることを確認する

インスタンスに関連付けられているセキュリティグループで、IP アドレスからの受信 SSH トラフィックが許可されていることを確認します。VPC のデフォルトのセキュリティグループでは着信 SSH トラフィックはデフォルトでは許可されません。インスタンス起動ウィザードで作成されたセキュリティグループではデフォルトで SSH トラフィックが許可されます。Linux インスタンスにインバウンド SSH トラフィックのルールを追加する手順については[コンピュータからのインスタンスへの接続ルール]を参照してください。確認する手順については[インスタンスへの接続エラー: 接続タイムアウト]を参照してください。

インスタンスが準備ができていることを確認する

インスタンスを起動してから接続リクエストを受け入れる準備が整うまでに、数分かかる場合があります。インスタンスをチェックして、それが実行中であり、ステータスチェックに合格していることを確認します。

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [インスタンス] を選択し、インスタンスを選択してください。
3. 以下について確認してください。

    a. [インスタンスの状態] 列で、インスタンスの状態が `running` であることを確認します。

    b. [ステータスチェック] 列で、インスタンスがすべてのステータスチェックに合格したことを確認します。

接続の前提条件をすべて満たしていることを確認する

接続に必要な情報がすべて揃っていることを確認します。詳細については、「[SSH を使用した Linux インスタンスへの接続]」を参照してください。

Linux または macOS X から接続する

ローカルコンピュータのオペレーティングシステムが Linux または macOS X の場合、Linux インスタンスに接続するための固有の前提条件について以下を確認してください。

- SSH クライアント
- EC2 インスタントコネクト
- AWS Systems Manager セッションマネージャー

Windows から接続する

ローカルコンピュータのオペレーティングシステムが Windows の場合、Linux インスタンスに接続するための固有の前提条件について以下を確認してください。

- OpenSSH
- PuTTY
- AWS Systems Manager セッションマネージャー
- Windows Subsystem for Linux

インスタンスが管理されたインスタンスかを確認します

管理されたインスタンスへのユーザー主導の接続は許可されません。インスタンスが管理されているかどうかを判断するにはインスタンスの 管理された フィールドを見つけます。値が 正 の場合、管理されたインスタンスです。詳細については、「Amazon EC2 マネージドインスタンス」を参照してください。

インスタンスへの接続エラー: 接続タイムアウト

インスタンスへ接続しようとして、エラーメッセージ `Network error: Connection timed out` または `Error connecting to [instance], reason: -> Connection timed out: connect` が表示される場合、次を実行します。

セキュリティグループルールを調べます。

ローカルコンピュータの適切なポートのパブリック IPv4 アドレスからのインバウンドトラフィックがセキュリティグループルールで許可されている必要があります。

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [インスタンス] を選択し、インスタンスを選択してください。
3. コンソールページの下部にある [セキュリティ] タブの [インバウンドルール] で、選択したインスタンスで有効なルールのリストを確認します。ローカルコンピュータからポート 22 (SSH) へのトラフィックを許可するルールがあることを確認します。

セキュリティグループに、ローカルコンピュータからのインバウンドトラフィックを許可するルールがない場合はセキュリティグループにルールを追加します。詳細については、「コンピュータからのインスタンスへの接続ルール」を参照してください。

4. インバウンドトラフィックを許可するルールについては[Source] (ソース) フィールドを確認します。値が単一の IP アドレスで、IP アドレスが静的でない場合、コンピュータを再起動するたびに新しい IP アドレスが割り当てられます。これにより、ルールにはコンピュータの IP アドレストラフィックが含まれなくなります。コンピュータが企業ネットワークにある場合またはインターネットサービスプロバイダー (ISP) を通じて接続する場合はコンピュータの IP アドレスが静的ではない可能性があります。つまり、コンピュータの IP アドレスは動的で、コンピュータを再起動するたびに変化します。セキュリティグループルールで、ローカルコンピュータからのインバウンドトラフィックが許可されるようにするには[Source] (ソース) に単一の IP アドレスを指定するのではなく、クライアントコンピュータで使用されている IP アドレスの範囲を指定します。

   セキュリティグループのルールの詳細についてはAmazon VPCユーザーガイドの「セキュリティグループのルール」を参照してください。

サブネットのルートテーブルを確認します。

VPC の外部あてのすべてのトラフィックを VPC のインターネットゲートウェイに送信するにはルートが必要です。

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [インスタンス] を選択し、インスタンスを選択してください。
3. [ネットワーキング] タブで、VPC ID とサブネット ID の値を書き留めます。
4. Amazon VPC コンソールの https://console.aws.amazon.com/vpc/ を開いてください。
5. ナビゲーションペインで、[Internet Gateways] を選択してください。ご使用の VPC にアタッチされているインターネットゲートウェイがあることを確認します。または[インターネットゲートウェイの作成] を選択し、インターネットゲートウェイの名前を入力して [インターネットゲートウェイの作成] を選択してください。次に、作成したインターネットゲートウェイで、[アクション]、[VPC にアタッチ]、[VPC] の順に選択し、[インターネットゲートウェイのアタッチ] をクリックして VPC にアタッチします。
6. ナビゲーションペインで [Subnets] を選択し、サブネットを選択してください。
7. [ルートテーブル] タブで、送信先として `0.0.0.0/0`、ターゲットとして VPC のインターネットゲートウェイが指定されたルートがあることを確認します。IPv6 アドレスを使用してインス

タンスに接続する場合はインターネットゲートウェイを指しているすべての IPv6 トラフィック (`::/0`) 用のルートがあることを確認します。それ以外の場合は以下の作業を行います。

a. ルートテーブルの ID (rtb-xxxxxxxx) を選択して、ルートテーブルに移動します。

b. [ルーター] タブで、[ルーター編集] を選択してください。[ルーター追加] を選択して、`0.0.0.0/0` を送信先として追加し、インターネットゲートウェイをターゲットとして使用します。IPv6 の場合は[ルーター追加] を選択して、`::/0` を送信先として追加し、インターネットゲートウェイをターゲットとして使用します。

c. [ルーター保存] を選択してください。

サブネットのネットワークアクセスコントロールリスト (ACL) を確認します。

ネットワーク ACL がポート 22 でローカル IP アドレスからのインバウンド SSH トラフィックを許可している必要があります。また、一時ポート (1024-65535) へのアウトバウンドトラフィックも許可する必要があります。

1. Amazon VPC コンソールの https://console.aws.amazon.com/vpc/ を開いてください。
2. ナビゲーションペインで、[サブネット] を選択してください。
3. サブネットを選択する
4. [ネットワーク ACL] タブの [インバウンドルール] で、必要なポートでコンピュータからのインバウンドトラフィックを許可しているルールがあることを確認します。それが見つからない場合はコンピュータへのトラフィックをブロックしているルールを削除または変更します。
5. [アウトバウンドルール] で、一時ポートでコンピュータへのトラフィックを許可しているルールがあることを確認します。存在しない場合はコンピュータへのトラフィックをブロックしているルールを削除または変更します。

コンピュータが社内ネットワークに接続されている場合

社内ファイアウォールで、ご使用のコンピュータのインバウンドおよびアウトバウンドのトラフィックがポート 22 で許可されているかどうか、ネットワーク管理者に問い合わせてください。

ご使用のコンピュータにファイアウォールが設定されている場合、そのファイアウォールでコンピュータのインバウンドおよびアウトバウンドのトラフィックがポート 22 で許可されているかどうか確認します。

インスタンスにパブリック IPv4 アドレスがあることを確認します。

そうでない場合はElastic IP アドレスをインスタンスに関連付けることができます。詳細については、「Elastic IP アドレス」を参照してください。

サーバーが過負荷になっている可能性のあるインスタンスの CPU 負荷を確認します。

AWS は自動的に Amazon CloudWatch メトリクスおよびインスタンスステータスなどのデータを提供します。これらを使用することでインスタンスの CPU 負荷を確認でき、必要に応じて負荷の処理方法を調整できます。詳細については、「CloudWatch を使用したインスタンスのモニタリング」を参照してください。

- 負荷が変化する場合、Auto Scaling および エラスティックロードバランシング を使用して、インスタンスの増減を自動的に縮小・拡張できます。
- 負荷が常に増加している場合、大きなインスタンスタイプに移動できます。詳細については、「Amazon EC2 インスタンスタイプの変更」を参照してください。

IPv6 アドレスを使用してインスタンスに接続するには以下のことを確認します。

- サブネットはインターネットゲートウェイへの IPv6 トラフィック (`::/0`) のルートを持つルートテーブルに関連付けられている必要があります。
- セキュリティグループルールではポート 22 でローカル IPv6 アドレスからのインバウンドトラフィックを許可する必要があります。
- ネットワーク ACL ルールではインバウンドおよびアウトバウンドの IPv6 トラフィックを許可する必要があります。
- 古い AMI からインスタンスを起動した場合、DHCPv6 用に設定されていない可能性があります (IPv6 アドレスはネットワークインターフェイスでは自動的に認識されません)。詳細については「Amazon VPC ユーザーガイド」の「インスタンスに IPv6 を設定する」を参照してください。
- ローカルコンピュータに IPv6 アドレスがあり、IPv6 を使用するように設定されている必要があります。

エラー: キーを読み込めません..。期待: 任意のプライベートキー

インスタンスに接続しようとして、エラーメッセージ、`unable to load key ... Expecting: ANY PRIVATE KEY` が表示される場合、プライベートキーが保存されているファイルが正しく設定されていません。プライベートキーファイルが `.pem` で終わる場合でも、正しく設定されていない可能性があります。プライベートキーファイルが正しく設定されていない原因として考えられるのは証明書がないことです。

プライベートキーファイルが正しく設定されていない場合は以下の手順に従ってエラーを解決する

1. 新しいキーペアを作成します。詳細については、「Amazon EC2 を使用してキーペアを作成する」を参照してください。

   > **Note**
   >
   > サードパーティー製のツールを使用して、新しいキーペアを作成することもできます。詳細については、「サードパーティー製のツールを使用してキーペアを作成し、Amazon EC2 にパブリックキーをインポートする」を参照してください。

2. 新しいキーペアをインスタンスに追加します。詳細については、「プライベートキーを紛失しました。自分のインスタンスに接続するにはどうすればよいですか?」を参照してください。
3. 新しいキーペアを使用してインスタンスに接続します。

エラー: ユーザーキーがサーバーによって認識されない

SSH を使用してインスタンスに接続している場合

- `ssh -vvv` を使用して、接続中に 3 倍詳細デバッグ情報を取得します。

```
ssh -vvv -i path/key-pair-name.pem instance-user-
name@ec2-203-0-113-25.compute-1.amazonaws.com
```

次のサンプル出力はサーバーが認識しないキーを使用してインスタンスに接続しようとした場合に表示される可能性があります。

```
open/ANT/myusername/.ssh/known_hosts).
debug2: bits set: 504/1024
debug1: ssh_rsa_verify: signature correct
debug2: kex_derive_keys
debug2: set_newkeys: mode 1
debug1: SSH2_MSG_NEWKEYS sent
debug1: expecting SSH2_MSG_NEWKEYS
debug2: set_newkeys: mode 0
debug1: SSH2_MSG_NEWKEYS received
debug1: Roaming not allowed by server
debug1: SSH2_MSG_SERVICE_REQUEST sent
debug2: service_accept: ssh-userauth
debug1: SSH2_MSG_SERVICE_ACCEPT received
```

```
debug2: key: boguspem.pem ((nil))
debug1: Authentications that can continue: publickey
debug3: start over, passed a different list publickey
debug3: preferred gssapi-keyex,gssapi-with-mic,publickey,keyboard-
interactive,password
debug3: authmethod_lookup publickey
debug3: remaining preferred: keyboard-interactive,password
debug3: authmethod_is_enabled publickey
debug1: Next authentication method: publickey
debug1: Trying private key: boguspem.pem
debug1: read PEM private key done: type RSA
debug3: sign_and_send_pubkey: RSA 9c:4c:bc:0c:d0:5c:c7:92:6c:8e:9b:16:e4:43:d8:b2
debug2: we sent a publickey packet, wait for reply
debug1: Authentications that can continue: publickey
debug2: we did not send a packet, disable method
debug1: No more authentication methods to try.
Permission denied (publickey).
```

PuTTY を使用してインスタンスに接続している場合

- 秘密キー (.pem) ファイルが PuTTY によって認識される形式 (.ppk) に変換されていることを確認します。プライベートキーの変換の詳細については[PuTTY を使用して Linux インスタンスに接続する]を参照してください。

> **Note**
>
> PuTTYgen でプライベートキーファイルをロードし、[Generate] ではなく [Save Private キー] を選択してください。

- AMI 用の適切なユーザー名で接続していることを確認します。[PuTTY Configuration] ウィンドウの [ホスト名] ボックスにユーザー名を入力してください。

インスタンスの起動に使用される AMI	デフォルトユーザー名
Amazon Linux	`ec2-user`
CentOS	`centos` または `ec2-user`
Debian	`admin`

インスタンスの起動に使用される AMI	デフォルトユーザー名
Fedora	`fedora` または `ec2-user`
FreeBSD	`ec2-user`
RHEL	`ec2-user` または `root`
SUSE	`ec2-user` または `root`
Ubuntu	`ubuntu`
Oracle	`ec2-user`
Bitnami	`bitnami`
Rocky Linux	`rocky`
その他	AMI プロバイダーに確認してください。

- 適切なポートへのインバウンドトラフィックを許可しているインバウンドセキュリティグループがあることを確認します。詳細については、「コンピュータからのインスタンスへの接続ルール」を参照してください。

エラー: アクセス許可が拒否されたか、[インスタンス] ポート 22 によって接続が閉じられました。

SSH を使用してインスタンスに接続し、`Host key not found in [directory]`、`Permission denied (publickey)`、`Authentication failed, permission denied`、または `Connection closed by [instance] port 22` のいずれかのエラーが発生した場合はAMI 用の適切なユーザー名で接続していること、およびインスタンス用の適切なプライベートキー (`.pem`) ファイルを指定していることを確認します。

適切なユーザー名は以下のとおりです。

インスタンスの起動に使用される AMI	デフォルトユーザー名
Amazon Linux	`ec2-user`

インスタンスの起動に使用される AMI	デフォルトユーザー名
CentOS	`centos` または `ec2-user`
Debian	`admin`
Fedora	`fedora` または `ec2-user`
FreeBSD	`ec2-user`
RHEL	`ec2-user` または `root`
SUSE	`ec2-user` または `root`
Ubuntu	`ubuntu`
Oracle	`ec2-user`
Bitnami	`bitnami`
Rocky Linux	`rocky`
その他	AMI プロバイダーに確認してください。

例えば、SSH クライアントを使用して Amazon Linux インスタンスに接続するには次のコマンドを使用します。

```
ssh -i /path/key-pair-name.pem instance-user-
name@ec2-203-0-113-25.compute-1.amazonaws.com
```

使用しているプライベートキーファイルが、インスタンスの起動時に選択したキーペアに対応していることを確認します。

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [インスタンス] を選択し、インスタンスを選択してください。
3. [詳細] タブの [インスタンスの詳細] で、[キーペア名] の値を確認します。
4. インスタンスを起動したときにキーペアを指定しなかった場合はキーペアを確実に指定するために、インスタンスを終了してから新しいインスタンスを起動します。それまで使用していたインスタンスで、キーペアに対する .pem ファイルがもう存在しない場合はそのキーペアを新しい

キーペアで置き換えることができます。詳細については、「プライベートキーを紛失しました。自分のインスタンスに接続するにはどうすればよいですか?」を参照してください。

独自のキーペアを生成した場合はキージェネレータが RSA キーを作成するように設定されていることを確認します。DSA キーは受け入れられません。

Permission denied (publickey) エラーが表示され、上のいずれも当てはまらない場合 (例えば、以前は接続できていたなど)、インスタンスのホームディレクトリのアクセス権限が変更された可能性があります。/home/*instance-user-name*/.ssh/authorized_keys のアクセス権限は所有者のみに制限する必要があります。

インスタンスのアクセス権限を検証するには

1. インスタンスを停止し、ルートボリュームをデタッチします。詳細については、「Amazon EC2 インスタンスの停止と開始」を参照してください。
2. 同じアベイラビリティーゾーンにある一時インスタンスを現在のインスタンスとして起動し (現在のインスタンスに使用したのと同様または同じ AMI を使用)、ルートボリュームを一時インスタンスにアタッチします。
3. 一時インスタンスに接続してマウントポイントを作成し、アタッチしたボリュームをマウントします。
4. 一時インスタンスから、アタッチされたボリュームの /home/*instance-user-name*/ ディレクトリのアクセス権限をチェックします。必要に応じて、次のようにアクセス権限を調整します。

```
[ec2-user ~]$ chmod 600 mount_point/home/instance-user-name/.ssh/authorized_keys
```

```
[ec2-user ~]$ chmod 700 mount_point/home/instance-user-name/.ssh
```

```
[ec2-user ~]$ chmod 700 mount_point/home/instance-user-name
```

5. ボリュームをアンマウントして一時インスタンスからデタッチし、元のインスタンスに再アタッチします。ルートボリュームに適切なデバイス名を指定したことを確認します (/dev/xvda など)。
6. インスタンスを起動します。一時インスタンスが必要なくなった場合は終了できます。

エラー: Unprotected Private キー ファイル (保護されていないプライベートキーファイル)

プライベートキーファイルはその他のすべてのユーザーの読み取りおよび書き込み操作から保護されている必要があります。プライベートキーがお客様以外のユーザーによって読み取りまたは書き込みできる場合、SSH はキーを無視し、次の警告メッセージが表示されます。

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@         WARNING: UNPROTECTED PRIVATE KEY FILE!          @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Permissions 0777 for '.ssh/my_private_key.pem' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
bad permissions: ignore key: .ssh/my_private_key.pem
Permission denied (publickey).
```

インスタンスへのログインを試みたときに同様のメッセージが表示された場合はエラーメッセージの最初の行を調べて、インスタンスに対して正しいパブリックキーを使用していることを確認します。上記の例ではプライベートキー `.ssh/my_private_key.pem` をファイル権限 `0777` とともに使用します。これにより、任意のユーザーがこのファイルの読み取りまたは書き込みを行うことができます。この権限レベルの安全性は非常に低いので、SSH はこのキーを無視します。

macOS または Linux から接続する場合にエラーを修正するにはプライベートキーファイルのパスを置き換えて次のコマンドを実行します。

```
[ec2-user ~]$ chmod 0400 .ssh/my_private_key.pem
```

Windows から Linux インスタンスに接続する場合はローカルコンピュータに対して次の手順を実行します。

1. .pem ファイルに移動します。
2. .pem ファイルを右クリックし、[プロパティ] を選択してください。
3. [セキュリティ] タブを選択してください。
4. [詳細] を選択してください。
5. 自分がファイルの所有者であることを確認します。そうでない場合は所有者を自分のユーザー名に変更します。
6. [継承の無効化] および [このオブジェクトから継承されたすべてのアクセス許可を削除] を選択してください。

7. [追加]、[プリンシパルの選択] を選択し、ユーザー名を入力して、[OK] をクリックしてください。
8. [許可エントリ] ウィンドウから、読み取りアクセス許可を付与して、[OK] をクリックしてください。
9. [Apply] (適用) をクリックして、すべての設定が保存されているようにします。
10. [OK] をクリックして、[セキュリティの詳細設定] ウィンドウを閉じます。
11. [OK] をクリックして、[プロパティ] ウィンドウを閉じます。
12. Windows から SSH を使用して Linux インスタンスに接続できるはずです。

Windows のコマンドプロンプトから次のコマンドを実行します。

1. コマンドプロンプトから、.pem ファイルのファイルパスの場所に移動します。
2. 明示的なアクセス許可をリセットおよび削除するには次のコマンドを実行します。

```
icacls.exe $path /reset
```

3. 現在のユーザーに読み取りアクセス許可を付与するには次のコマンドを実行します。

```
icacls.exe $path /GRANT:R "$($env:USERNAME):(R)"
```

4. 継承を無効にし、継承されたアクセス許可を削除するには次のコマンドを実行します。

```
icacls.exe $path /inheritance:r
```

5. Windows から SSH を使用して Linux インスタンスに接続できるはずです。

エラー: プライベートキーの先頭は「-----BEGIN RSA PRIVATE KEY-----」、末尾は「-----END RSA PRIVATE KEY-----」にする必要があります

サードパーティーツール (例: ssh-keygen) を使用して RSA キーペアを作成すると、プライベートキーが OpenSSH キー形式で生成されます。インスタンスに接続する際、OpenSSH 形式のプライベートキーを使用してパスワードを復号すると、エラー (`Private key must begin with "-----BEGIN RSA PRIVATE KEY-----" and end with "-----END RSA PRIVATE KEY-----"`) が発生する場合があります。

エラーを解消するにはプライベートキーは PEM 形式である必要があります。PEM 形式でプライベートキーを作成するには次のコマンドを使用します。

```
ssh-keygen -m PEM
```

エラー: ホストキーの検証に失敗しました

このエラーは、 known_hosts ファイル内のインスタンスに保存されているホストキーとクライアントに保存されているホストキーが一致しない場合に発生します。例えば、あるパブリック IP アドレスを使用してインスタンスに接続し、別のパブリック IP アドレスを使用して再度接続しようとすると、不一致が発生する可能性があります。これは、Elastic IP アドレスを追加または削除した後に発生する可能性があります。理由は、インスタンスのパブリック IP アドレスが変更されるためです。

このエラーを解決するには、まず、インスタンスのホストキーまたはネットワーク設定に予期された変更があったことを確認します。インスタンスに接続する前に、ホストフィンガープリントを確認することもお勧めします。インスタンスに接続したら、 known_hosts ファイルから古いホストキーを削除できます。手順については、インスタンスで使用されている Linux ディストリビューションのドキュメントを参照してください。

エラー: サーバー refused our key または No supported authentication methods available (サーバーはキーを拒否しましたまたは利用可能なサポートされる認証方法はありません)

PuTTY を使用してインスタンスに接続し、[Error: サーバー refused our key] または [Error: No supported authentication methods available] のいずれかのエラーが発生した場合はAMI の適切なユーザー名で接続していることを確認します。[PuTTY Configuration] ウィンドウの [User name] にユーザー名を入力してください。

適切なユーザー名は以下のとおりです。

インスタンスの起動に使用される AMI	デフォルトユーザー名
Amazon Linux	ec2-user
CentOS	centos または ec2-user
Debian	admin

インスタンスの起動に使用される AMI	デフォルトユーザー名
Fedora	`fedora` または `ec2-user`
FreeBSD	`ec2-user`
RHEL	`ec2-user` または `root`
SUSE	`ec2-user` または `root`
Ubuntu	`ubuntu`
Oracle	`ec2-user`
Bitnami	`bitnami`
Rocky Linux	`rocky`
その他	AMI プロバイダーに確認してください。

次の点も確認する必要があります。

- 最新バージョンの PuTTY を使用している。詳細についてはPuTTY のウェブページを参照してください。
- プライベートキー (.pem) ファイルが PuTTY によって認識される形式 (.ppk) に正しく変換されている。プライベートキーの変換の詳細についてはPuTTY を使用して Linux インスタンスに接続するを参照してください。

インスタンスに対して ping を実行できない

`ping` コマンドはICMP トラフィックの一種です。インスタンスに対して Ping を実行できない場合はインバウンドセキュリティグループのルールで、すべてのソースからの、あるいはコマンドを発行しているコンピュータまたはインスタンスからの `Echo Request` メッセージについて、ICMP トラフィックが許可されていることを確認します。

インスタンスから `ping` コマンドを発行できない場合はアウトバウンドセキュリティグループのルールで、すべての宛先への、または Ping の対象であるホストへの `Echo Request` メッセージについて、ICMP トラフィックが許可されていることを確認します。

Ping コマンドはファイアウォールによってブロックされたり、ネットワークのレイテンシーやハードウェアの問題によってタイムアウトしたりすることもあります。詳しいトラブルシューティングについてはローカルネットワークまたはシステム管理者に問い合わせてください。

エラー: サーバーによる予期しないネットワーク接続の閉鎖

PuTTY を使用してインスタンスに接続中に「サーバーによる予期しないネットワーク接続の閉鎖」エラーを受け取った場合、PuTTY 設定の接続ページでキープアライブを有効化して、切断を回避していることを確認してください。一部のサーバーは指定された時間内にデータが一切受信されない場合に、クライアントを切断します。キープアライブ間の秒数を 59 秒に設定します。

キープアライブを有効後にも問題が依然として発生する場合にはPuTTY 設定の接続ページで Nagle のアルゴリズムを無効にすることを試してください。

エラー: EC2 インスタントコネクト のホストキーの検証に失敗しました

インスタンスホストキーをローテーションしても、新しいホストキーは AWS の信頼されたホストキーデータベースに自動的にアップロードされません。これにより、EC2 インスタントコネクト ブラウザベースのクライアントを使用してインスタンスに接続しようとすると、ホストキーの検証が失敗し、インスタンスに接続できなくなります。

エラーを解決するには`eic_harvest_hostkeys` スクリプトをインスタンスで実行する必要があります。これにより、新しいホストキーが EC2 インスタントコネクト にアップロードされます。スクリプトは Amazon Linux 2 インスタンス上の `/opt/aws/bin/` にあり、Ubuntu インスタンスでは `/usr/share/ec2-instance-connect/` にあります。

Amazon Linux 2

Amazon Linux 2 インスタンスでホストキーの検証に失敗したエラーを解決するには

1. SSH を使用してインスタンスに接続します。

   EC2 インスタントコネクト CLI を使用するか、インスタンスの起動時にインスタンスに割り当てた SSH キーペアと、インスタンスの起動に使用した AMI のデフォルトユーザー名を使用して接続できます。Amazon Linux 2 の場合、デフォルトのユーザー名は `ec2-user` です。

   例えば、Amazon Linux 2 を使用してインスタンスを起動した場合、インスタンスのパブリック DNS 名は `ec2-a-b-c-d.us-west-2.compute.amazonaws.com`、キーペアは

my_ec2_private_key.pem です。次のコマンドを使用して SSH 経由でインスタンスに接続します。

```
$ ssh -i my_ec2_private_key.pem ec2-user@ec2-a-b-c-d.us-
west-2.compute.amazonaws.com
```

インスタンスへの接続の詳細については[SSH クライアントを使用して Linux インスタンスに接続する]を参照してください。

2. 次のフォルダに移動します。

```
[ec2-user ~]$ cd /opt/aws/bin/
```

3. インスタンスで次のコマンドを実行します。

```
[ec2-user ~]$ ./eic_harvest_hostkeys
```

呼び出しが成功しても、出力されないことに注意してください。

これで、EC2 インスタントコネクト ブラウザベースのクライアントを使用してインスタンスに接続できます。

Ubuntu

Ubuntuインスタンスでホストキーの検証に失敗したエラーを解決するには

1. SSH を使用してインスタンスに接続します。

EC2 インスタントコネクト CLI を使用するか、インスタンスの起動時にインスタンスに割り当てた SSH キーペアと、インスタンスの起動に使用した AMI のデフォルトユーザー名を使用して接続できます。Ubuntu の場合、デフォルトユーザー名は ubuntu です。

例えば、Ubuntu を使用してインスタンスを起動した場合、インスタンスのパブリック DNS 名は ec2-a-b-c-d.us-west-2.compute.amazonaws.com、キーペアは my_ec2_private_key.pem です。次のコマンドを使用して SSH 経由でインスタンスに接続します。

```
$ ssh -i my_ec2_private_key.pem ubuntu@ec2-a-b-c-d.us-
west-2.compute.amazonaws.com
```

インスタンスへの接続の詳細については[SSH クライアントを使用して Linux インスタンスに接続する]を参照してください。

2. 次のフォルダに移動します。

```
[ec2-user ~]$ cd /usr/share/ec2-instance-connect/
```

3. インスタンスで次のコマンドを実行します。

```
[ec2-user ~]$ ./eic_harvest_hostkeys
```

呼び出しが成功しても、出力されないことに注意してください。

これで、EC2 インスタントコネクト ブラウザベースのクライアントを使用してインスタンスに接続できます。

EC2 インスタントコネクト を使用して Unbntu インスタンスに接続できない

EC2 インスタントコネクト を使用して Ubuntu インスタンスへの接続を試行中にエラーが発生した場合、次の情報を使用して問題の解決を試みることができます。

考えられる原因

インスタンス上の`ec2-instance-connect`パッケージは最新バージョンではありません。

解決策

次のように、インスタンス上の`ec2-instance-connect`パッケージを最新バージョンにアップデートします。

1. EC2 インスタントコネクト 以外の方法でインスタンスに接続します。
2. インスタンス上で次のコマンドを実行して`ec2-instance-connect`パッケージを最新バージョンにアップデートします。

```
apt update && apt upgrade
```

プライベートキーを紛失しました。自分のインスタンスに接続するにはどうすればよいですか?

EBS-Backed インスタンスのプライベートキーを失った場合はインスタンスへのアクセス権を回復することができます。インスタンスを停止し、そのルートボリュームをデタッチし、データボリュームとして別のインスタンスにアタッチし、新しいパブリックキーで authorized_keys ファイルを変更して、ボリュームを元のインスタンスに戻し、インスタンスを再起動する必要があります。インスタンスの起動、接続、および停止の詳細についてはAmazon EC2 インスタンスの状態変更を参照してください。

この手順はEBS ルートボリュームを持つインスタンスでのみサポートされます。インスタンスにインスタンスストアのルートボリュームがある場合、この手順を使用してインスタンスへのアクセスを回復することはできません。インスタンスに接続するにはプライベートキーが必要です。インスタンスのルートボリュームタイプを判断するにはAmazon EC2コンソールを開き、[インスタンス] を選択してインスタンスを選択し、[ストレージ] タブを選択し、[ルートデバイス詳細] セクションで、[ルートデバイスタイプ] の値をチェックします。

この値は EBS または INSTANCE-STORE のどちらかです。

プライベートキーを紛失した場合、以下の手順以外にも Linux インスタンスに接続する方法があります。詳細については最初の起動後に SSH キーペアを紛失した場合、Amazon EC2 インスタンスに接続するにはどうすればよいですか?を参照してください。

別のキーペアを使用して EBS-Backed インスタンスに接続するためのステップ

- ステップ 1: 新しいキーペアを作成する
- ステップ 2: 元のインスタンスとそのルートボリュームに関する情報を取得する
- ステップ 3: 元のインスタンスを停止する
- ステップ 4: 一時インスタンスを起動する
- ステップ 5: 元のインスタンスからルートボリュームをデタッチし、一時インスタンスにアタッチする
- ステップ 6: 一時インスタンスにマウントされた元のボリュームの authorized_keys に、新しいパブリックキーを追加する
- ステップ 7: 一時インスタンスから元のボリュームをアンマウントしてデタッチし、元のインスタンスに再アタッチする
- ステップ 8: 新しいキーペアを使用して元のインスタンスに接続する
- ステップ 9: クリーンアップする

ステップ 1: 新しいキーペアを作成する

Amazon EC2 コンソールまたはサードパーティ製のツールで、新しいキーペアを作成します。新しいキーペアの名前として、紛失したプライベートキーと同じ名前を指定するにはまず既存のキーペアを削除する必要があります。新しいキーペアの作成の詳細については Amazon EC2 を使用してキーペアを作成する または サードパーティー製のツールを使用してキーペアを作成し、Amazon EC2 にパブリックキーをインポートする を参照してください。

ステップ 2: 元のインスタンスとそのルートボリュームに関する情報を取得する

この手順を完了するために必要になるので、次の情報を書き留めます。

元のインスタンスに関する情報を取得するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Instances] を選択し、接続先にするインスタンスを選択します (このインスタンスを元のインスタンスと呼びます)。
3. [Details] タブで、インスタンス ID と AMI ID を書き留めます。
4. [ネットワーキング] タブで、アベイラビリティーゾーンを書き留めます。
5. [Storage] タブの [ルートデバイス名] で、ルートボリュームのデバイス名 (/dev/xvda など) を書き留めます。次に、[Block devices] で、このデバイス名を見つけ、ボリューム ID (vol-0a1234b5678c910de など) を書き留めます。

ステップ 3: 元のインスタンスを停止する

[インスタンスの状態]、[インスタンスの停止] の順に選択してください。このオプションが無効になっている場合はインスタンスが既に停止しているか、またはルートボリュームがインスタンスストアボリュームです。

> **Warning**
>
> インスタンスストアボリューム上のデータは、インスタンスを停止すると失われます。このデータを保持するには、永続ストレージにバックアップしてください。

ステップ 4: 一時インスタンスを起動する

一時インスタンスを起動するには

1. ナビゲーションペインで [Instances] (インスタンス)、[Launch instances] (インスタンスの起動) の順に選択してください。
2. [Name and tags] (名前とタグ) セクションの [Name] (名前) に「Temporary (一時)」と入力してください。
3. [Application and OS Images] (アプリケーションと OS イメージ) セクションで、元のインスタンスの起動に使用したものと同じ AMI を選択してください。その AMI を使用できない場合は停止したインスタンスから使用可能な AMI を作成できます。詳細については、「Amazon EBS-backed AMI を作成する」を参照してください。
4. [Instance type] (インスタンスタイプ) セクションではデフォルトのインスタンスタイプを保持します。
5. [キー pair] (キーペア) セクションの [キー pair name] (キーペア名) で、使用する既存のキーペアを選択するか、新しいキーペアを作成します。
6. [ネットワーク設定] セクションで [編集] を選択し、次に [サブネット] で、元のインスタンスと同じアベイラビリティーゾーンのサブネットを選択してください。
7. [Summary] (サマリー) パネルで、[Launch] (起動) を選択してください。

ステップ 5: 元のインスタンスからルートボリュームをデタッチし、一時インスタンスにアタッチする

1. ナビゲーションペインで [ボリューム] を選択し、元のインスタンスのルートボリュームを選択します (前のステップでそのボリューム ID を書き留めました)。[アクションs] (アクション)、[Detach Volume] (ボリュームのデタッチ)、[Detach] (デタッチする) の順に選択してください。ボリュームの状態が `available` になるまで待ちます ([Refresh] アイコンを選択しなければならない場合があります)。
2. ボリュームを選択したまま [アクションs] (アクション) を選択し、次に [Attach Volume] (ボリュームをアタッチ) を選択してください。一時インスタンスのインスタンス ID を選択し、[Device name] (デバイス名) で指定されたデバイス名 (例: `/dev/sdf`) を書き留めて、[Attach volume] (ボリュームをアタッチ) を選択してください。

> **Note**
>
> 元のインスタンスを AWS Marketplace AMI から起動して、ボリュームに AWS Marketplace のコードが含まれている場合はボリュームをアタッチする前に一時インスタンスを停止する必要があります。

ステップ 6: 一時インスタンスにマウントされた元のボリュームの **authorized_keys** に、新しいパブリックキーを追加する

1. 一時インスタンスに接続します。
2. 一時インスタンスから、そのファイルシステムにアクセスできるように、インスタンスにアタッチしたボリュームをマウントします。例えば、デバイス名が /dev/sdf の場合、次のコマンドを使用してボリュームを /mnt/tempvol としてマウントします。

> **Note**
>
> デバイス名の表示がインスタンスでは異なる場合があります。例えば、/dev/sdf としてマウントされているデバイスが、インスタンスでは /dev/xvdf として表示される場合があります。Red Hat の一部のバージョン (または CentOS などのバリアント) ではさらに末尾の文字が 4 文字インクリメントされる場合があります。例えば、/dev/sd*f* は /dev/xvd*k* になります。

   a. lsblkコマンドを使用して、ボリュームがパーティション分割されているかどうかを判断します。

```
[ec2-user ~]$ lsblk
NAME    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
xvda    202:0    0    8G  0 disk
##xvda1 202:1    0    8G  0 part /
xvdf    202:80   0  101G  0 disk
##xvdf1 202:81   0  101G  0 part
xvdg    202:96   0   30G  0 disk
```

   前述の例では/dev/xvda と /dev/xvdf はパーティション分割されたボリュームで、/dev/xvdg はパーティション分割されていません。ボリュームがパーティション分割され

ている場合は次のステップで raw デバイス (/dev/xvdf1)) の代わりにパーティション(/dev/xvdf) をマウントします。

b. ボリュームをマウントするための一時ディレクトリを作成します。

```
[ec2-user ~]$ sudo mkdir /mnt/tempvol
```

c. 以前に特定したデバイス名またはボリューム名を使用して、一時マウントポイントにボリューム (またはパーティション) をマウントします。必要なコマンドはオペレーティングシステムのファイルシステムによって異なります。注意事項デバイス名の表示がインスタンスでは異なる場合があります。詳細についてはステップ 6 のnoteを参照してください。

- Amazon Linux、Ubuntu、Debian

```
[ec2-user ~]$ sudo mount /dev/xvdf1 /mnt/tempvol
```

- Amazon Linux 2、CentOS、SUSE Linux 12、RHEL 7.x

```
[ec2-user ~]$ sudo mount -o nouuid /dev/xvdf1 /mnt/tempvol
```

> **Note**
>
> ファイルシステムが破損していることを示すエラーが表示された場合は次のコマンドを実行して fsck ユーティリティを使用してファイルシステムをチェックし、問題を修復します。
>
> ```
> [ec2-user ~]$ sudo fsck /dev/xvdf1
> ```

3. 一時インスタンスから、次のコマンドを使用して、一時インスタンスの authorized_keys からの新しいパブリックキーを使用し、マウントされたボリューム上で authorized_keys を更新します。

> **Important**
>
> 以下の例ではAmazon Linux のユーザー名 ec2-user を使用しています。別のユーザー名に置き換える必要がある (Ubuntu インスタンスの場合は ubuntu など) 場合があります。

```
[ec2-user ~]$ cp .ssh/authorized_keys /mnt/tempvol/home/ec2-user/.ssh/
authorized_keys
```

このコピーが正常に終了すると、次のステップに進むことができます。

(オプション) または/mnt/tempvol のファイルを編集するアクセス許可がない場合、sudo を使用してファイルを更新してから、ファイルに対するアクセス許可を確認して、元のインスタンスにログインできるかどうかを確認する必要があります。次のコマンドを使用して、ファイルに対するアクセス許可を確認します。

```
[ec2-user ~]$ sudo ls -l /mnt/tempvol/home/ec2-user/.ssh
total 4
-rw------- 1 222 500 398 Sep 13 22:54 authorized_keys
```

この出力例では*222* はユーザー ID、*500* はグループ ID です。次に、sudo を使用して失敗したコピーコマンドを再実行します。

```
[ec2-user ~]$ sudo cp .ssh/authorized_keys /mnt/tempvol/home/ec2-user/.ssh/
authorized_keys
```

次のコマンドを再度実行して、アクセス許可が変更されているかどうかを判断します。

```
[ec2-user ~]$ sudo ls -l /mnt/tempvol/home/ec2-user/.ssh
```

ユーザー ID とグループ ID が変更されている場合は次のコマンドを実行して復元します。

```
[ec2-user ~]$ sudo chown 222:500 /mnt/tempvol/home/ec2-user/.ssh/authorized_keys
```

ステップ 7: 一時インスタンスから元のボリュームをアンマウントしてデタッチし、元のインスタンスに再アタッチする

1. 一時インスタンスから、元のインスタンスに再アタッチできるように、アタッチしたボリュームをアンマウントします。例えば、/mnt/tempvol のボリュームをアンマウントするには次のコマンドを使用します。

```
[ec2-user ~]$ sudo umount /mnt/tempvol
```

2. (前のステップでアンマウントした) 一時インスタンスからボリュームをデタッチする: Amazon EC2 コンソールのナビゲーションペインで [ボリューム] を選択し、(前のステップでボリューム ID を書き留めた) 元のインスタンスのルートボリュームを選択し、[アクション]、[ボリュームのデタッチ] の順に選択してください。次に、[デタッチ] を選択してください。ボリュームの状態が `available` になるまで待ちます ([Refresh] アイコンを選択しなければならない場合があります)。
3. ボリュームを元のインスタンスに再アタッチする: ボリュームを選択した状態で、[アクション] (アクション)、[Attach Volume] (ボリュームをアタッチ) の順に選択してください。元のインスタンスのインスタンス ID を選択し、元のルートボリュームのアタッチメントについて先程の ステップ 2 で記録したデバイス名 (`/dev/sda1` または `/dev/xvda`) を指定してから、[ボリュームのアタッチ] を選択してください。

> **Important**
>
> 元のアタッチと同じデバイス名を指定しない場合、元のインスタンスを起動することはできません。Amazon EC2 はルートボリュームが `sda1` または `/dev/xvda` であることを想定しています。

ステップ 8: 新しいキーペアを使用して元のインスタンスに接続する

元のインスタンスを選択し、[インスタンスの状態]、[Start instance (インスタンスの開始)] の順に選択してください。インスタンスが `running` 状態になったら、新しいキーペアのプライベートキーファイルを使用して、そのインスタンスに接続できます。

> **Note**
>
> 新しいキーペアおよび対応するプライベートキーファイルの名前が元のキーペアの名前と異なる場合はインスタンスに接続するときに新しいプライベートキーファイルの名前を必ず指定します。

ステップ 9: クリーンアップする

(オプション) 一時インスタンスをそれ以上使用しない場合は終了できます。一時インスタンスを選択し、[インスタンスの状態]、[インスタンスの終了 (削除)] の順に選択してください。

ステータスチェックに失敗した Amazon EC2 Linux インスタンスをトラブルシューティングする

以下の情報はLinux インスタンスでステータスチェックに失敗した場合の問題のトラブルシューティングに役立ちます。まず、アプリケーションで問題が発生しているかどうかを確認します。インスタンスでアプリケーションが正常に実行されていないことを確認した場合はステータスチェック情報とシステムログを確認します。

ステータスチェックの失敗の原因となる問題の例については「Amazon EC2 インスタンスのステータスチェック」を参照してください。

コンテンツ

- ステータスチェック情報の確認
- システムログの取得
- Linux インスタンスに対するシステムログエラーのトラブルシューティング
- メモリ不足: プロセスの終了
- エラー: mmu_update failed (メモリ管理の更新に失敗しました)
- I/O エラー (ブロックデバイス障害)
- I/O エラー: ローカルでもリモートディスクでもありません (破損した分散ブロックデバイス)
- request_module: runaway loop modprobe (古い Linux バージョンでレガシーカーネル modprobe がループしている)
- 「FATAL: kernel too old」および「fsck: No such file or directory while trying to open /dev」 (カーネルと AMI の不一致)
- 「FATAL: Could not load /lib/modules」または「BusyBox」 (カーネルモジュールの欠如)
- エラー: 無効のカーネル (EC2 と互換性のないカーネル)
- fsck: No such file or directory while trying to open..。(ファイルシステムが見つからない。)
- General error mounting filesystems (マウント失敗)
- VFS: Unable to mount root fs on unknown-block (ルートファイルシステム不一致)

- Error: Unable to determine major/minor number of root device..。(ルートファイルシステム/デバイス不一致)
- XENBUS: Device with no driver..。
- ... days without being checked, check forced (ファイルシステムのチェックが必要です)
- fsck died with exit status..。(デバイスが見つからない)
- GRUB プロンプト (grubdom>)
- Bringing up interface eth0: Device eth0 has different MAC address than expected, ignoring。(ハードコードされた MAC アドレス)
- SELinux ポリシーを読み込めません。Machine is in enforcing mode。Halting now。(SELinux の誤設定)
- XENBUS: Timeout connecting to devices (Xenbus タイムアウト)

ステータスチェック情報の確認

Amazon EC2 コンソールを使用して、問題のあるインスタンスを調査するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [インスタンス] を選択し、インスタンスを選択してください。
3. [ステータスとアラーム] タブを選択すると、すべての [システムステータスのチェック]、[インスタンスステータスのチェック]、および [アタッチされた EBS ステータスのチェック] の個別の結果が表示されます。

ステータスチェックに失敗した場合、次のいずれかの方法を試すことができます。

- 失敗したステータスチェックに応じてインスタンスを復旧するアラームを作成します。詳細については「インスタンスを停止、終了、再起動、または復旧するアラームを作成する」を参照してください。
- (インスタンスステータスチェック) インスタンスタイプを Nitro ベースのインスタンスに変更した場合、必要な ENA と NVMe ドライバーがないインスタンスから移行するとステータスチェックが失敗します。詳細については、「インスタンスタイプ変更の互換性」を参照してください。
- EBS ルートボリュームを持つインスタンスの場合、インスタンスを停止して再起動します。詳細については、「Amazon EC2 インスタンスの停止と開始」を参照してください。

- インスタンスストアのルートボリュームを持つインスタンスの場合、インスタンスを終了し、代わりのインスタンスを起動します。詳細については、「Amazon EC2 インスタンスを終了する」を参照してください。
- Amazon EC2 が問題を解決するのを待ちます。
- サポート に問い合わせるか、問題を AWS re:Post に投稿してください。
- インスタンスが Auto Scaling グループに属している場合:
  - (システムステータスのチェックとインスタンスステータスのチェック) デフォルトではAmazon EC2 Auto Scaling は代替インスタンスを自動的に起動します。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling グループ内のインスタンスのヘルスチェック」を参照してください。
  - (アタッチ済みの EBS ステータスチェック) 代替インスタンスを自動的に起動するように Amazon EC2 Auto Scaling を設定する必要があります。詳細については「Amazon EC2 Auto Scaling ユーザーガイド」の「Monitor and replace Auto Scaling instances with impaired Amazon EBS volumes」を参照してください。
- システムログを取得し、エラーを探します。詳細については「システムログの取得」を参照してください。

システムログの取得

インスタンスのステータスチェックに失敗した場合はインスタンスを再起動してシステムログを取得できます。ログから判明したエラーが問題のトラブルシューティングに役立つ場合があります。再起動すると、ログから不要な情報が消去されます。

インスタンスを再起動してシステムログを取得するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Instances] を選択し、インスタンスを選択してください。
3. [インスタンスの状態]、[インスタンスの再起動] の順に選択してください。インスタンスが再起動するまでには数分かかることがあります。
4. 問題がまだ存在することを確認します。再起動によって、問題が解決することがあります。
5. インスタンスが `running` 状態になったら、[アクション]、[モニタリングとトラブルシューティング]、[システムログの取得] の順に選択してください。
6. 画面に表示されるログを確認し、下記の既知のシステムログエラー文のリストを使用して、問題のトラブルシューティングを行います。
7. 問題が解決されない場合は問題を AWS re:Post に投稿できます。

Linux インスタンスに対するシステムログエラーのトラブルシューティング

インスタンスの接続性チェックなどのインスタンスのステータスチェックに失敗した Linux インスタンスについて、上記のステップに従ってシステムログを取得したことを確認します。次のリストは一般的なシステムログエラー、および各エラーの問題解決に対して推奨する対処を示しています。

メモリエラー

- メモリ不足: プロセスの終了
- エラー: mmu_update failed (メモリ管理の更新に失敗しました)

デバイスエラー

- I/O エラー (ブロックデバイス障害)
- I/O エラー: ローカルでもリモートディスクでもありません (破損した分散ブロックデバイス)

カーネルエラー

- request_module: runaway loop modprobe (古い Linux バージョンでレガシーカーネル modprobe がループしている)
- 「FATAL: kernel too old」および「fsck: No such file or directory while trying to open /dev」 (カーネルと AMI の不一致)
- 「FATAL: Could not load /lib/modules」または「BusyBox」 (カーネルモジュールの欠如)
- エラー: 無効のカーネル (EC2 と互換性のないカーネル)

ファイルシステムエラー

- fsck: No such file or directory while trying to open..。(ファイルシステムが見つからない。)
- General error mounting filesystems (マウント失敗)
- VFS: Unable to mount root fs on unknown-block (ルートファイルシステム不一致)
- Error: Unable to determine major/minor number of root device..。(ルートファイルシステム/デバイス不一致)
- XENBUS: Device with no driver..。
- ... days without being checked, check forced (ファイルシステムのチェックが必要です)
- fsck died with exit status..。(デバイスが見つからない)

[オペレーティングシステムエラー]

- GRUB プロンプト (grubdom>)
- Bringing up interface eth0: Device eth0 has different MAC address than expected, ignoring。(ハードコードされた MAC アドレス)
- SELinux ポリシーを読み込めません。Machine is in enforcing mode。Halting now。(SELinux の誤設定)
- XENBUS: Timeout connecting to devices (Xenbus タイムアウト)

メモリ不足: プロセスの終了

メモリ不足エラーは下記のようなシステムログで示されます。

```
[115879.769795] Out of memory: kill process 20273 (httpd) score 1285879
or a child
[115879.769795] Killed process 1917 (php-cgi) vsz:467184kB, anon-
rss:101196kB, file-rss:204kB
```

可能性のある原因

メモリの枯渇

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次のいずれかを行ってください。 • インスタンスを停止し、異なるインスタンスタイプを使用するようにインスタンスを変更した後、インスタンスを再び起動します。例えば、大きいインスタンスタイプやメモリ最適化インスタンスタイプです。 • インスタンスを再起動して、障害のないステータスに戻します。インスタンスタイプを変更しない限り、問題が再び発生する可能性があります。

インスタンスタイプ	操作
Instance store-Backed	次のいずれかを行ってください。 • インスタンスを終了し、別のインスタンスタイプを指定して、新しいインスタンスを起動します。例えば、大きいインスタンスタイプやメモリ最適化インスタンスタイプです。 • インスタンスを再起動して、障害のないステータスに戻します。インスタンスタイプを変更しない限り、問題が再び発生する可能性があります。

エラー: mmu_update failed (メモリ管理の更新に失敗しました)

メモリ管理更新失敗は下記のようなシステムログで示されます。

```
...
Press `ESC' to enter the menu... 0   [H[J  Booting 'Amazon Linux 2011.09
 (2.6.35.14-95.38.amzn1.i686)'


root (hd0)

 Filesystem type is ext2fs, using whole disk

kernel /boot/vmlinuz-2.6.35.14-95.38.amzn1.i686 root=LABEL=/ console=hvc0 LANG=

en_US.UTF-8 KEYTABLE=us

initrd /boot/initramfs-2.6.35.14-95.38.amzn1.i686.img

ERROR: mmu_update failed with rc=-22
```

可能性のある原因

Amazon Linux に関する問題

推奨する対処

AWS に問い合わせるか、問題を re:Postサポート に投稿してください。

I/O エラー (ブロックデバイス障害)

入力/出力エラーは次の例のようなシステムログで示されます。

```
[9943662.053217] end_request: I/O error, dev sde, sector 52428288
[9943664.191262] end_request: I/O error, dev sde, sector 52428168
[9943664.191285] Buffer I/O error on device md0, logical block 209713024
[9943664.191297] Buffer I/O error on device md0, logical block 209713025
[9943664.191304] Buffer I/O error on device md0, logical block 209713026
[9943664.191310] Buffer I/O error on device md0, logical block 209713027
[9943664.191317] Buffer I/O error on device md0, logical block 209713028
[9943664.191324] Buffer I/O error on device md0, logical block 209713029
[9943664.191332] Buffer I/O error on device md0, logical block 209713030
[9943664.191339] Buffer I/O error on device md0, logical block 209713031
[9943664.191581] end_request: I/O error, dev sde, sector 52428280
[9943664.191590] Buffer I/O error on device md0, logical block 209713136
[9943664.191597] Buffer I/O error on device md0, logical block 209713137
[9943664.191767] end_request: I/O error, dev sde, sector 52428288
[9943664.191970] end_request: I/O error, dev sde, sector 52428288
[9943664.192143] end_request: I/O error, dev sde, sector 52428288
[9943664.192949] end_request: I/O error, dev sde, sector 52428288
[9943664.193112] end_request: I/O error, dev sde, sector 52428288
[9943664.193266] end_request: I/O error, dev sde, sector 52428288
...
```

可能性のある原因

インスタンスタイプ	可能性のある原因
Amazon EBS-Backed	障害が発生した Amazon EBS ボリューム
Instance store-Backed	障害が発生した物理ドライブ

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次の手順に従ってください。 1. インスタンスを停止します。 2. ボリュームをデタッチします。 3. ボリュームの回復を試みます。 **Note** Amazon EBS ボリュームのスナップショットを頻繁に作成することをお勧めします。これによって、エラーのためにデータを損失する危険性が大幅に減少します。 4. ボリュームをインスタンスに再アタッチします。 5. インスタンスを起動します。
Instance store-Backed	インスタンスを終了し、新しいインスタンスを起動します。 **Note** データを復旧できない。バックアップから復旧します。 **Note** Amazon S3 または Amazon EBS をバックアップに使用することをお勧めします。インスタンスストアボリュー

インスタンスタイプ	操作
	ムは単一のホストと単一のディスクエラーに直接結びついています。

I/O エラー: ローカルでもリモートディスクでもありません (破損した分散ブロックデバイス)

デバイスでの入力/出力エラーは次の例のようなシステムログで示されます。

```
...
block drbd1: Local IO failed in request_timer_fn. Detaching...

Aborting journal on device drbd1-8.

block drbd1: IO ERROR: neither local nor remote disk

Buffer I/O error on device drbd1, logical block 557056

lost page write due to I/O error on drbd1

JBD2: I/O error detected when updating journal superblock for drbd1-8.
```

可能性のある原因

インスタンスタイプ	可能性のある原因
Amazon EBS-Backed	障害が発生した Amazon EBS ボリューム
Instance store-Backed	障害が発生した物理ドライブ

推奨する対処

インスタンスを終了し、新しいインスタンスを起動します。

Amazon EBS-Backed インスタンスの場合、最新スナップショットからイメージを作成して、データを回復できます。スナップショットを作成した後に追加されたデータは回復できません。

request_module: runaway loop modprobe (古い Linux バージョンでレガシーカーネル modprobe がループしている)

下記のようなシステムログで、この状態が示されます。不安定であるか古い Linux カーネル (例: 2.6.16-xenU) を使用すると、起動時に無限ループが発生することがあります。

```
Linux version 2.6.16-xenU (builder@xenbat.amazonsa) (gcc version 4.0.1
20050727 (Red Hat 4.0.1-5)) #1 SMP Mon May 28 03:41:49 SAST 2007

BIOS-provided physical RAM map:

 Xen: 0000000000000000 - 0000000026700000 (usable)

0MB HIGHMEM available.
...

request_module: runaway loop modprobe binfmt-464c

request_module: runaway loop modprobe binfmt-464c

request_module: runaway loop modprobe binfmt-464c

request_module: runaway loop modprobe binfmt-464c

request_module: runaway loop modprobe binfmt-464c
```

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次のいずれかのオプションを使用して、GRUB ベースまたは静的な新しいカーネルを使用します。 オプション 1: インスタンスを終了し、-kernel および -ramdisk パラメータを指定して新しいインスタンスを起動します。 オプション 2: 1. インスタンスを停止します。

インスタンスタイプ	操作
	2. 新しいカーネルを使用するようカーネルとラムディスク属性を変更します。 3. インスタンスを起動します。
Instance store-Backed	インスタンスを終了し、-kernel および -ramdisk パラメータを指定して新しいインスタンスを起動します。

「FATAL: kernel too old」および「fsck: No such file or directory while trying to open /dev」 (カーネルと AMI の不一致)

下記のようなシステムログで、この状態が示されます。

```
Linux version 2.6.16.33-xenU (root@dom0-0-50-45-1-a4-ee.z-2.aes0.internal)
(gcc version 4.1.1 20070105 (Red Hat 4.1.1-52)) #2 SMP Wed Aug 15 17:27:36 SAST 2007
...
FATAL: kernel too old
Kernel panic - not syncing: Attempted to kill init!
```

可能性のある原因

互換性のないカーネルとユーザーランド

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次の手順に従ってください。 1. インスタンスを停止します。 2. 新しいカーネルを使用するよう設定を変更します。 3. インスタンスを起動します。
Instance store-Backed	次の手順に従ってください。

インスタンスタイプ	操作
	1. より新しいカーネルを使用する AMI を作成します。 2. インスタンスを終了します。 3. 作成した AMI から新しいインスタンスを起動します。

「FATAL: Could not load /lib/modules」または「BusyBox」 (カーネルモジュールの欠如)

下記のようなシステムログで、この状態が示されます。

```
[    0.370415] Freeing unused kernel memory: 1716k freed
Loading, please wait...
WARNING: Couldn't open directory /lib/modules/2.6.34-4-virtual: No such file or
 directory
FATAL: Could not open /lib/modules/2.6.34-4-virtual/modules.dep.temp for writing: No
 such file or directory
FATAL: Could not load /lib/modules/2.6.34-4-virtual/modules.dep: No such file or
 directory
Couldn't get a file descriptor referring to the console
Begin: Loading essential drivers... ...
FATAL: Could not load /lib/modules/2.6.34-4-virtual/modules.dep: No such file or
 directory
FATAL: Could not load /lib/modules/2.6.34-4-virtual/modules.dep: No such file or
 directory
Done.
Begin: Running /scripts/init-premount ...
Done.
Begin: Mounting root file system... ...
Begin: Running /scripts/local-top ...
Done.
Begin: Waiting for root file system... ...
Done.
Gave up waiting for root device.  Common problems:
 - Boot args (cat /proc/cmdline)
   - Check rootdelay= (did the system wait long enough?)
   - Check root= (did the system wait for the right device?)
 - Missing modules (cat /proc/modules; ls /dev)
```

```
FATAL: Could not load /lib/modules/2.6.34-4-virtual/modules.dep: No such file or
 directory
FATAL: Could not load /lib/modules/2.6.34-4-virtual/modules.dep: No such file or
 directory
ALERT! /dev/sda1 does not exist. Dropping to a shell!


BusyBox v1.13.3 (Ubuntu 1:1.13.3-1ubuntu5) built-in shell (ash)
Enter 'help' for a list of built-in commands.

(initramfs)
```

可能性のある原因

次の 1 つ以上の条件によって、この問題が発生する可能性があります。

- ラムディスクが見つからない
- ラムディスクに正しいモジュールが見つからない
- Amazon EBS ルートボリュームが /dev/sda1 として正しくアタッチされていない

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次の手順に従ってください。 1. Amazon EBS ボリュームに対して正しいラムディスクを選択してください。 2. インスタンスを停止します。 3. ボリュームをデタッチし、修復します。 4. ボリュームをインスタンスにアタッチします。 5. インスタンスを起動します。 6. 正しいラムディスクを使用するよう AMI を変更します。
Instance store-Backed	次の手順に従ってください。

インスタンスタイプ	操作
	1. インスタンスを終了してから、正しいラムディスクを使って新たなインスタンスを起動します。 2. 正しいラムディスクを使って新たな AMI を作成します。

エラー: 無効のカーネル (EC2 と互換性のないカーネル)

下記のようなシステムログで、この状態が示されます。

```
...
root (hd0)

 Filesystem type is ext2fs, using whole disk

kernel /vmlinuz root=/dev/sda1 ro

initrd /initrd.img

ERROR Invalid kernel: elf_xen_note_check: ERROR: Will only load images
built for the generic loader or Linux images
xc_dom_parse_image returned -1

Error 9: Unknown boot failure

  Booting 'Fallback'

root (hd0)

 Filesystem type is ext2fs, using whole disk

kernel /vmlinuz.old root=/dev/sda1 ro

Error 15: File not found
```

可能性のある原因

次の一方または両方の条件によって、この問題が発生する可能性があります。

- 指定されたカーネルは GRUB でサポートされていません
- フォールバックカーネルが存在しません

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次の手順に従ってください。 1. インスタンスを停止します。 2. 機能するカーネルに変更します。 3. フォールバックカーネルをインストールします。 4. カーネルを訂正して AMI を変更します。
Instance store-Backed	次の手順に従ってください。 1. インスタンスを終了してから、正しいカーネルを使って新たなインスタンスを起動します。 2. 正しいカーネルを使って AMI を作成します。 3. (オプション) データ復旧の技術サポートについてはサポート にお問い合わせください。

fsck: No such file or directory while trying to open..。(ファイルシステムが見つからない。)

下記のようなシステムログで、この状態が示されます。

```
  Welcome to Fedora
  Press 'I' to enter interactive startup.
Setting clock : Wed Oct 26 05:52:05 EDT 2011 [  OK  ]

Starting udev: [  OK  ]
```

```
Setting hostname localhost:  [  OK  ]

No devices found
Setting up Logical Volume Management: File descriptor 7 left open
  No volume groups found
[  OK  ]

Checking filesystems
Checking all file systems.
[/sbin/fsck.ext3 (1) -- /] fsck.ext3 -a /dev/sda1
/dev/sda1: clean, 82081/1310720 files, 2141116/2621440 blocks
[/sbin/fsck.ext3 (1) -- /mnt/dbbackups] fsck.ext3 -a /dev/sdh
fsck.ext3: No such file or directory while trying to open /dev/sdh

/dev/sdh:
The superblock could not be read or does not describe a correct ext2
filesystem.  If the device is valid and it really contains an ext2
filesystem (and not swap or ufs or something else), then the superblock
is corrupt, and you might try running e2fsck with an alternate superblock:
    e2fsck -b 8193 <device>

[FAILED]


*** An error occurred during the file system check.
*** Dropping you to a shell; the system will reboot
*** when you leave the shell.
Give root password for maintenance
(or type Control-D to continue):
```

可能性のある原因

- ラムディスクファイルシステム定義 /etc/fstab にバグがある
- /etc/fstab のファイルシステム定義の設定が不適切
- ドライブが見つからないかドライブにエラーがある

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次の手順に従ってください。

インスタンスタイプ	操作
	1. インスタンスを停止し、ルートボリュームをデタッチし、/etc/fstab を修復または変更し、ボリュームをインスタンスにアタッチし、インスタンスを起動します。 2. ラムディスクを修正して、変更した /etc/fstab を含めます (適用可能な場合)。 3. 新しいラムディスクを使用するよう AMI を変更します。 fstab の 6 番目のフィールドではマウントの可用性要件を定義します。0 以外の値を指定した場合、そのボリュームに対して fsck を実行して成功しなければならないことを意味します。Amazon EC2 でこのフィールドを使用すると、問題が発生することがあります。これは実行に失敗すると通常は対話的なコンソールプロンプトが表示されますが、このコンソールプロンプトは現在 Amazon EC2 で使用できないためです。この機能は慎重に使用してください。また、fstab の Linux マニュアルページを参照してください。
Instance store-Backed	次の手順に従ってください。 1. インスタンスを終了し、新しいインスタンスを起動します。 2. 障害のある Amazon EBS ボリュームをデタッチし、インスタンスを再起動します。 3. (オプション) データ復旧の技術サポートについてはサポート にお問い合わせください。

General error mounting filesystems (マウント失敗)

下記のようなシステムログで、この状態が示されます。

```
Loading xenblk.ko module
xen-vbd: registered block device major 8

Loading ehci-hcd.ko module
Loading ohci-hcd.ko module
Loading uhci-hcd.ko module
USB Universal Host Controller Interface driver v3.0

Loading mbcache.ko module
Loading jbd.ko module
Loading ext3.ko module
Creating root device.
Mounting root filesystem.
kjournald starting.  Commit interval 5 seconds

EXT3-fs: mounted filesystem with ordered data mode.

Setting up other filesystems.
Setting up new root fs
no fstab.sys, mounting internal defaults
Switching to new root and running init.
unmounting old /dev
unmounting old /proc
unmounting old /sys
mountall:/proc: unable to mount: Device or resource busy
mountall:/proc/self/mountinfo: No such file or directory
mountall: root filesystem isn't mounted
init: mountall main process (221) terminated with status 1

General error mounting filesystems.
A maintenance shell will now be started.
CONTROL-D will terminate this shell and re-try.
Press enter for maintenance
(or type Control-D to continue):
```

可能性のある原因

インスタンスタイプ	可能性のある原因
Amazon EBS-Backed	• Amazon EBS ボリュームがデタッチされているか、ボリュームにエラーがあります。

インスタンスタイプ	可能性のある原因
	• ファイルシステムが破損している。 • ラムディスクと AMI の組み合わせが一致していません (例: Debian ラムディスクと SUSE AMI)。
Instance store-Backed	• ドライブにエラーがあります。 • ファイルシステムが破損しています。 • ラムディスクと AMI の組み合わせが一致していません (例: Debian ラムディスクと SUSE AMI) 。

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次の手順に従ってください。 1. インスタンスを停止します。 2. ルートボリュームをデタッチします。 3. ルートボリュームを既知の動作しているインスタンスにアタッチします。 4. ファイルシステムチェック (fsck -a /dev/...) を実行します 5. エラーを修正します。 6. ボリュームを既知の動作しているインスタンスからデタッチします。 7. 停止したインスタンスにボリュームをアタッチします。 8. インスタンスを起動します。 9. インスタンスのステータスを再確認します。
Instance store-Backed	以下のいずれかを行ってください。

インスタンスタイプ	操作
	• 新しいインスタンスを起動します。 • (オプション) データ復旧の技術サポートについてはサポート にお問い合わせください。

VFS: Unable to mount root fs on unknown-block (ルートファイルシステム不一致)

下記のようなシステムログで、この状態が示されます。

```
Linux version 2.6.16-xenU (builder@xenbat.amazonsa) (gcc version 4.0.1
 20050727 (Red Hat 4.0.1-5)) #1 SMP Mon May 28 03:41:49 SAST 2007
...
Kernel command line:  root=/dev/sda1 ro 4
...
Registering block device major 8
...
Kernel panic - not syncing: VFS: Unable to mount root fs on unknown-block(8,1)
```

可能性のある原因

インスタンスタイプ	可能性のある原因
Amazon EBS-Backed	• デバイスが正しくアタッチされていない。 • ルートデバイスが正しいデバイスポイントにアタッチされていない。 • ファイルシステムのフォーマットが正しくありません。 • レガシーカーネルを使用している (例えば、2.6.16-XenU)。 • インスタンスの最新のカーネル更新 (更新エラーまたは更新バグ)
Instance store-Backed	ハードウェアデバイスのエラー。

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次のいずれかを行ってください。 • インスタンスを停止し、再起動します。 • 正しいデバイスポイントでアタッチするようルートボリュームを変更します。例えば、/dev/sda の代わりに /dev/sda1 を使用します。 • 停止し、新しいカーネルを使用するように変更します。 • 既知の更新バグを確認するにはLinux ディストリビューションのドキュメントを参照してください。カーネルを変更または再インストールします。
Instance store-Backed	インスタンスを終了し、新しいカーネルを使用して、新しいインスタンスを起動します。

Error: Unable to determine major/minor number of root device..。(ルートファイルシステム/デバイス不一致)

下記のようなシステムログで、この状態が示されます。

```
...
XENBUS: Device with no driver: device/vif/0
XENBUS: Device with no driver: device/vbd/2048
drivers/rtc/hctosys.c: unable to open rtc device (rtc0)
Initializing network drop monitor service
Freeing unused kernel memory: 508k freed
:: Starting udevd...
done.
:: Running Hook [udev]
:: Triggering uevents...<30>udevd[65]: starting version 173
done.
Waiting 10 seconds for device /dev/xvda1 ...
```

```
Root device '/dev/xvda1' doesn't exist. Attempting to create it.
ERROR: Unable to determine major/minor number of root device '/dev/xvda1'.
You are being dropped to a recovery shell
    Type 'exit' to try and continue booting
sh: can't access tty; job control turned off
[ramfs /]#
```

可能性のある原因

- 仮想ブロックデバイスドライバーが見つからないか、設定が間違っている
- デバイス列挙が競合している (sda と xvda または sda1 の代わりに sda)
- インスタンスカーネルが正しく選択されていない

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次の手順に従ってください。 1. インスタンスを停止します。 2. ボリュームをデタッチします。 3. デバイスのマッピングの問題を解決します。 4. インスタンスを起動します。 5. AMI を変更して、デバイスのマッピングの問題に対処します。
Instance store-Backed	次の手順に従ってください。 1. 適切な修正を使用して新しい AMI を作成します (ブロックデバイスを正しくマッピングします)。 2. インスタンスを終了し、作成した AMI から新しいインスタンスを起動します。

XENBUS: Device with no driver..。

下記のようなシステムログで、この状態が示されます。

```
XENBUS: Device with no driver: device/vbd/2048
drivers/rtc/hctosys.c: unable to open rtc device (rtc0)
Initializing network drop monitor service
Freeing unused kernel memory: 508k freed
:: Starting udevd...
done.
:: Running Hook [udev]
:: Triggering uevents...<30>udevd[65]: starting version 173
done.
Waiting 10 seconds for device /dev/xvda1 ...
Root device '/dev/xvda1' doesn't exist. Attempting to create it.
ERROR: Unable to determine major/minor number of root device '/dev/xvda1'.
You are being dropped to a recovery shell
    Type 'exit' to try and continue booting
sh: can't access tty; job control turned off
[ramfs /]#
```

可能性のある原因

- 仮想ブロックデバイスドライバーが見つからないか、設定が間違っている
- デバイス列挙が競合している (sda と xvda)。
- インスタンスカーネルが正しく選択されていない

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次の手順に従ってください。 1. インスタンスを停止します。 2. ボリュームをデタッチします。 3. デバイスのマッピングの問題を解決します。 4. インスタンスを起動します。

インスタンスタイプ	操作
	5. AMI を変更して、デバイスのマッピングの問題に対処します。
Instance store-Backed	次の手順に従ってください。 1. 適切な修正を使用して AMI を作成します (ブロックデバイスを正しくマッピングします)。 2. インスタンスを終了し、作成した AMI を使用して新しいインスタンスを起動します。

... days without being checked, check forced (ファイルシステムのチェックが必要です)

下記のようなシステムログで、この状態が示されます。

```
...
Checking filesystems
Checking all file systems.
[/sbin/fsck.ext3 (1) -- /] fsck.ext3 -a /dev/sda1
/dev/sda1 has gone 361 days without being checked, check forced
```

可能性のある原因

ファイルシステムのチェック期間が経過したため、ファイルシステムチェックが強制実行されている。

推奨する対処

- ファイルシステムチェックが完了するまで待ちます。ルートファイルシステムのサイズによってはファイルシステムチェックに時間がかかることがあります。
- tune2fs またはファイルシステムに適したツールを使用してファイルシステムを変更し、ファイルシステムチェック (fsck) の実行を削除します。

fsck died with exit status..。(デバイスが見つからない)

下記のようなシステムログで、この状態が示されます。

```
Cleaning up ifupdown....
Loading kernel modules...done.
...
Activating lvm and md swap...done.
Checking file systems...fsck from util-linux-ng 2.16.2
/sbin/fsck.xfs: /dev/sdh does not exist
fsck died with exit status 8
[31mfailed (code 8).[39;49m
```

可能性のある原因

- 存在しないドライブをラムディスクが検索している
- ファイルシステムの整合性チェックが強制実行されている
- ドライブにエラーがあるか、デタッチされている

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	問題を解決するため、次のいずれかを試します。 • インスタンスを停止し、ボリュームを既存の実行中インスタンスにアタッチします。 • 整合性チェックを手動で実行します。 • ラムディスクを修正して、関連するユーティリティを含めます。 • ファイルシステム調整パラメータを変更して、整合性要件を削除します (お勧めしません)。
Instance store-Backed	問題を解決するため、次のいずれかを試します。

インスタンスタイプ	操作
	• ラムディスクに正しいツールをリバンドリングします。 • ファイルシステム調整パラメータを変更して、整合性要件を削除します (お勧めしません)。 • インスタンスを終了し、新しいインスタンスを起動します。 • (オプション) データ復旧の技術サポートについてはサポート にお問い合わせください。

GRUB プロンプト (grubdom>)

下記のようなシステムログで、この状態が示されます。

```
    GNU GRUB  version 0.97  (629760K lower / 0K upper memory)

       [ Minimal BASH-like line editing is supported.   For

         the   first   word,  TAB  lists  possible  command

         completions.  Anywhere else TAB lists the possible

         completions of a device/filename. ]

grubdom>
```

可能性のある原因

インスタンスタイプ	可能性のある原因
Amazon EBS-Backed	• GRUB 設定ファイルがありません。 • 正しくない GRUB イメージが使用されています。別の場所に GRUB 設定ファイルが必要です。

インスタンスタイプ	可能性のある原因
	• GRUB 設定ファイルを保存するために使用されているファイルシステムがサポートされていません (例えば、ルートファイルシステムを GRUB の以前のバージョンでサポートされていないタイプに変換した)
Instance store-Backed	• GRUB 設定ファイルがありません。 • 正しくない GRUB イメージが使用されています。別の場所に GRUB 設定ファイルが必要です。 • GRUB 設定ファイルを保存するために使用されているファイルシステムがサポートされていません (例えば、ルートファイルシステムを GRUB の以前のバージョンでサポートされていないタイプに変換した)

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	オプション 1: AMI を変更しインスタンスを再作成します。 1. ソース AMI を変更して、標準の場所に GRUB 設定ファイルを作成します (/boot/grub/menu.lst)。 2. GRUB のバージョンが、基になるファイルシステムのタイプをサポートしていることを確認し、必要に応じて GRUB をアップグレードします。 3. 適切な GRUB イメージを選択します (hd0 – 第 1 ドライブまたは hd00 – 第 1 ドライブ、第 1 パーティション)。

インスタンスタイプ	操作
	4. インスタンスを終了し、作成した AMI を使用して新しいインスタンスを起動します。 オプション2: 既存のインスタンスの修正: 1. インスタンスを停止します。 2. ルートファイルシステムをデタッチします。 3. ルートファイルシステムを既知の動作しているインスタンスにアタッチします。 4. ファイルシステムをマウントします。 5. GRUB 設定ファイルを作成します。 6. GRUB のバージョンが、基になるファイルシステムのタイプをサポートしていることを確認し、必要に応じて GRUB をアップグレードします。 7. ファイルシステムをデタッチします。 8. 元のインスタンスにアタッチします。 9. カーネル属性を変更して、適切な GRUB イメージを使用します (第 1 ディスクまたは第 1 ディスクの第 1 パーティション)。 10.インスタンスを起動します。

インスタンスタイプ	操作
Instance store-Backed	オプション 1: AMI を変更しインスタンスを再作成します。 1. GRUB 設定ファイルを使用して、標準の場所に新しい AMI を作成します (/boot/grub/menu.lst)。 2. 適切な GRUB イメージを選択します (hd0 – 第 1 ドライブまたは hd00 – 第 1 ドライブ、第 1 パーティション)。 3. GRUB のバージョンが、基になるファイルシステムのタイプをサポートしていることを確認し、必要に応じて GRUB をアップグレードします。 4. インスタンスを終了し、作成した AMI を使用して新しいインスタンスを起動します。 オプション 2: インスタンスを終了し、正しいカーネルを指定して新しいインスタンスを起動します。 Note 既存のインスタンスからデータを復旧するにはサポート にお問い合わせください。

Bringing up interface eth0: Device eth0 has different MAC address than expected, ignoring。(ハードコードされた MAC アドレス)

下記のようなシステムログで、この状態が示されます。

```
...
Bringing up loopback interface:  [  OK  ]
```

```
Bringing up interface eth0:  Device eth0 has different MAC address than expected,
 ignoring.
[FAILED]

Starting auditd: [  OK  ]
```

可能性のある原因

AMI 設定にハードコードされたインターフェイス MAC がある

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次のいずれかを行ってください。 • AMI を変更してハードコードを削除し、インスタンスを再作成します。 • ハードコードされた MAC アドレスを削除するようにインスタンスを変更します。 または 次の手順に従ってください。 1. インスタンスを停止します。 2. ルートボリュームをデタッチします。 3. ボリュームを別のインスタンスにアタッチし、ハードコードされた MAC アドレスを削除するようにボリュームを変更します。 4. 初期インスタンスにボリュームをアタッチします。 5. インスタンスを起動します。
Instance store-Backed	次のいずれかを行ってください。 • ハードコードされた MAC アドレスを削除するようにインスタンスを変更します。

インスタンスタイプ	操作
	• インスタンスを終了し、新しいインスタンスを起動します。

SELinux ポリシーを読み込めません。Machine is in enforcing mode。Halting now。(SELinux の誤設定)

下記のようなシステムログで、この状態が示されます。

```
audit(1313445102.626:2): enforcing=1 old_enforcing=0 auid=4294967295
Unable to load SELinux Policy. Machine is in enforcing mode. Halting now.
Kernel panic - not syncing: Attempted to kill init!
```

可能性のある原因

SELinux が誤って有効にされた。

- 指定されたカーネルは GRUB でサポートされていません
- フォールバックカーネルが存在しません

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次の手順に従ってください。 1. 障害のあるインスタンスを停止します。 2. 障害の起きたインスタンスのルートボリュームをデタッチします。 3. 別の実行中の Linux インスタンスにルートボリュームをアタッチします (リカバリインスタンスとして扱われます)。 4. リカバリインスタンスを接続し、障害の起きたインスタンスのルートボリュームをマウントします。

インスタンスタイプ	操作
	5. マウントしたルートボリュームの SELinux を無効にします。このプロセスは Linux ディストリビューションによって異なります。詳細については各 OS のドキュメントを参照してください。 **Note** 一部のシステムでは/*mount_point* /etc/sysconfig/selinux ファイルで SELINUX=disabled と設定することで SELinux を無効にできます。*mount_point* 、 はリカバリインスタンス上の、ボリュームをマウントした場所です。 6. リカバリインスタンスからルートボリュームをアンマウントしてデタッチし、元のインスタンスに再アタッチします。 7. インスタンスを起動します。
Instance store-Backed	次の手順に従ってください。 1. インスタンスを終了し、新しいインスタンスを起動します。 2. (オプション) データ復旧の技術サポートについてはサポート にお問い合わせください。

XENBUS: Timeout connecting to devices (Xenbus タイムアウト)

下記のようなシステムログで、この状態が示されます。

```
Linux version 2.6.16-xenU (builder@xenbat.amazonsa) (gcc version 4.0.1
20050727 (Red Hat 4.0.1-5)) #1 SMP Mon May 28 03:41:49 SAST 2007
...
XENBUS: Timeout connecting to devices!
```

```
...
Kernel panic - not syncing: No init found.  Try passing init= option to kernel.
```

可能性のある原因

- ブロックデバイスがインスタンスに接続されていない
- このインスタンスは古いインスタンスカーネルを使用している

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次のいずれかを行ってください。 • AMI とインスタンスを変更して新しいカーネルを使用し、インスタンスを再作成します。 • インスタンスを再起動します。
Instance store-Backed	次のいずれかを行ってください。 • インスタンスを終了します。 • AMI を変更して新しいカーネルを使用し、その AMI を使用して新しいインスタンスを起動します。

Amazon EC2 Linux インスタンスを間違ったボリュームから起動した問題をトラブルシューティングする

状況によっては/dev/xvda または /dev/sda にアタッチしたボリューム以外のボリュームが、Linux インスタンスのルートボリュームになることがあります。これは別のインスタンスのルートボリュームや、ルートボリュームのスナップショットから作成されたボリュームを、既存のルートボリュームのインスタンスにアタッチした場合に起こります。

これは Linux の初期ラムディスクの挙動です。/ で /etc/fstab として定義されたボリュームを選択した場合でも、一部のディストリビューションではボリュームパーティションにアタッチされたラ

ベルによってボリュームが決定されます。例えば、/etc/fstab の内容が次のとおりであったとします。

```
LABEL=/ / ext4 defaults,noatime 1 1
tmpfs /dev/shm tmpfs defaults 0 0
devpts /dev/pts devpts gid=5,mode=620 0 0
sysfs /sys sysfs defaults 0 0
proc /proc proc defaults 0 0
```

両方のボリュームのラベルを確認すれば、両方に / ラベルが含まれることが判ります。

```
[ec2-user ~]$ sudo e2label /dev/xvda1
/
[ec2-user ~]$ sudo e2label /dev/xvdf1
/
```

この例では初期ラムディスクの実行後、意図していた /dev/xvda1 ボリュームからの起動ではなく、/dev/xvdf1 がインスタンスを起動するルートボリュームになる結果となりました。これを解決するために、同じ e2label コマンドを使用して、起動ボリュームではないアタッチ済みのボリュームのラベルを変更できます。

場合によっては/etc/fstab で UUID を指定することで、この問題を解決できます。ただし、両方のボリュームが同じスナップショットから作成された場合、またはセカンダリボリュームがプライマリボリュームのスナップショットから作成されている場合は両ボリュームは UUID を共有します。

```
[ec2-user ~]$ sudo blkid
/dev/xvda1: LABEL="/" UUID=73947a77-ddbe-4dc7-bd8f-3fe0bc840778 TYPE="ext4"
 PARTLABEL="Linux" PARTUUID=d55925ee-72c8-41e7-b514-7084e28f7334
/dev/xvdf1: LABEL="old/" UUID=73947a77-ddbe-4dc7-bd8f-3fe0bc840778 TYPE="ext4"
 PARTLABEL="Linux" PARTUUID=d55925ee-72c8-41e7-b514-7084e28f7334
```

アタッチされた ext4 ボリュームのラベルを変更するには

1. e2label コマンドを使用して、ボリュームのラベルを / 以外のものに変更します。

   ```
   [ec2-user ~]$ sudo e2label /dev/xvdf1 old/
   ```

2. ボリュームに新しいラベルがあることを確認します。

   ```
   [ec2-user ~]$ sudo e2label /dev/xvdf1
   ```

```
old/
```

アタッチされた xfs ボリュームのラベルを変更するには

- xfs_admin コマンドを使用して、ボリュームのラベルを / 以外のものに変更します。

```
[ec2-user ~]$ sudo xfs_admin -L old/ /dev/xvdf1
writing all SBs
new label = "old/"
```

図のようにボリュームラベルを変更した後で、インスタンスを再起動すると、インスタンスの起動時にラムディスクが正しいボリュームを選択するはずです。

> **Important**
>
> 新しいラベルを持つボリュームをデタッチし、別のインスタンスに戻してルートボリュームとして使用する場合は上記の手順をもう一度実行してラベルを元の値に戻す必要があります。これを行わない場合、ラムディスクがラベル / を持つボリュームを見つけることができないため、別のインスタンスが起動しません。

Amazon EC2 Windows インスタンスへの接続に関する問題のトラブルシューティング

以下の情報と一般的なエラーは、Windows インスタンスに接続するときの問題のトラブルシューティングに役立ちます。

接続の問題

- リモートデスクトップからリモートコンピュータに接続できません
- macOS RDP クライアントの使用中にエラーが発生する
- RDP にデスクトップではなく黒い画面が表示される
- 管理者ではないユーザーでインスタンスにリモートでログオンできない
- AWS Systems Manager を使用したリモートデスクトップ問題のトラブルシューティング
- リモートレジストリを使用して EC2 インスタンスでリモートデスクトップを有効にする
- プライベートキーを紛失しました。Windows インスタンスに接続するにはどうすればよいですか?

リモートデスクトップからリモートコンピュータに接続できません

インスタンスへの接続関連の問題を解決するには、以下を実行します。

- 正しいパブリック DNS ホスト名を使用していることを確認します (Amazon EC2 コンソールでは、インスタンスを選択し、詳細ペインの [Public DNS (IPv4) (パブリック DNS (IPv4))] を確認します)。インスタンスが VPC 内にあり、パブリック DNS 名が表示されない場合は、DNS ホスト名を有効にする必要があります。詳細については、「Amazon VPC ユーザーガイド」の「VPC の DNS 属性」を参照してください。
- インスタンスにパブリック IPv4 アドレスがあることを確認します。そうでない場合は、Elastic IP アドレスをインスタンスに関連付けることができます。詳細については、「Elastic IP アドレス」を参照してください。
- IPv6 アドレスを使用してインスタンスに接続するには、ローカルコンピュータに IPv6 があり、IPv6 アドレスを使用するように設定されていることを確認します。詳細については「Amazon VPC ユーザーガイド」の「インスタンスに IPv6 を設定する」を参照してください。
- セキュリティグループに、ポート 3389 での RDP アクセスを許可するルールがあることを確認します。
- パスワードをコピーしたがエラー `Your credentials did not work` が発生した場合、プロンプトが表示されたら手動で入力してください。パスワードをコピーしたときに、1 文字欠けていたり、余分な空白文字が含まれている可能性があります。
- インスタンスのステータスチェックが成功していることを確認します。詳細については、「Amazon EC2 インスタンスのステータスチェック」および「the section called “Linux インスタンスのステータスチェックの失敗”」を参照してください。
- サブネットのルートテーブルに、VPC 外へのすべてのトラフィックを VPC のインターネットゲートウェイに送信するルートがあることを確認します。詳細については、Amazon VPC ユーザーガイド の「カスタムルートテーブルを作成する」(インターネットゲートウェイ) を参照してください。
- Windows ファイアウォール、または他のファイアウォールのソフトウェアによって、インスタンスへの RDP トラフィックがブロックされていないことを確認します。Windows ファイアウォールを無効にし、セキュリティグループのルールを使用してインスタンスへのアクセスを制御することをお勧めします。以下のいずれかのオプションを試行します。
  - AWSSupport-TroubleshootRDP を disable the Windows Firewall profiles using SSM Agent に使用します。このオプションでは、インスタンスが AWS Systems Manager に設定されている必要があります。
  - AWSSupport-ExecuteEC2Rescue を使用します。

  - インスタンスを停止し、ルートボリュームをデタッチして、そのボリュームをデータボリュームとして一時インスタンスにアタッチします。一時インスタンスに接続し、ボリュームをオンラインにします。データボリュームからレジストリハイブをロードします。[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SharedAccess\Parameters\FirewallPolicyStandardProfile] で、[EnableFirewall] を「0」に設定します。レジストリハイブをアンロードしてから、ボリュームをオフラインにします。一時インスタンスからルートボリュームをデタッチし、元のインスタンスにルートボリュームとしてアタッチします。元のインスタンスを起動します。
- Active Directory ドメインの一部ではないインスタンスでネットワークレベル認証が無効になっていることを確認します (AWSSupport-TroubleshootRDP を使用して disable NLA を行います)。
- リモートデスクトップサービス (TermService) のスタートアップタイプが自動であり、サービスが開始されていることを確認します (AWSSupport-TroubleshootRDP を使用して enable and start the RDP service を行います)。
- 正しいリモートデスクトッププロトコルポート (デフォルトは 3389) に接続していることを確認します (AWSSupport-TroubleshootRDP を使用して read the current RDP port および change it back to 3389 を行います)。
- インスタンスでリモートデスクトップ接続が許可されていることを確認します (AWSSupport-TroubleshootRDP を使用して enable Remote Desktop connections を行います)。
- パスワードの有効期限が切れていないことを確認します。パスワードの有効期限が切れている場合、リセットできます。詳細については、「Amazon EC2 Windows インスタンスの Windows 管理者パスワードをリセットする」を参照してください。
- インスタンスで作成したユーザーを使用して接続しようとすると、エラー `The user cannot connect to the server due to insufficient access privileges` が表示される場合、そのユーザーにローカルでログオンする権限が付与されていることを確認してください。詳細については、「メンバーにローカルでログオンする権限を付与する」を参照してください。
- 許可されている最大の同時 RDP セッション数より多くのセッションを使用しようとすると、セッションは終了され、「`Your Remote Desktop Services session has ended. Another user connected to the remote computer, so your connection was lost.`」というメッセージが表示されます。デフォルトでは、インスタンスに許可される同時 RDP セッション数は 2 です。

macOS RDP クライアントの使用中にエラーが発生する

Microsoft ウェブサイトのリモートデスクトップ接続クライアントを使用して Windows Server インスタンスに接続する場合は、次のエラーが発生する可能性があります。

```
Remote Desktop Connection cannot verify the identity of the computer that you want to
 connect to.
```

Mac App Store から Microsoft リモートデスクトップアプリをダウンロードし、そのアプリを使用して、インスタンスに接続します。

RDP にデスクトップではなく黒い画面が表示される

この問題を解決するには、以下の手順を実行します。

- コンソール出力に追加情報がないか、確認します。Amazon EC2 コンソールを使用してインスタンスのコンソール出力を取得するには、インスタンスを選択してから、[アクション]、[モニタリングおよびトラブルシューティング]、[システムログの取得] の順に選択します。
- 最新バージョンの RDP クライアントを実行していることを確認します。
- RDP クライアントのデフォルト設定を使用してください。
- リモートデスクトップ接続を使用している場合、次のように `/admin` オプションを使用して開始してみてください。

  ```
  mstsc /v:instance /admin
  ```

- サーバーで全画面アプリケーションが実行されている場合、応答を停止する可能性があります。Ctrl+Shift+Esc キーを使用して Windows タスクマネージャーを起動し、アプリケーションを閉じます。
- サーバーの使用率が高すぎる場合、応答を停止する可能性があります。Amazon EC2 コンソールを使用してインスタンスを監視するには、インスタンスを選択し、[Monitoring] をクリックします。サイズの大きいインスタンスタイプに変更する必要がある場合は、「Amazon EC2 インスタンスタイプの変更」を参照してください。

管理者ではないユーザーでインスタンスにリモートでログオンできない

管理者アカウントではないユーザーを使用して、Windows インスタンスにリモートでログオンできない場合、ローカルでログオンする権限がユーザーに付与されていることを確認してください。「ユーザーまたはグループにドメインのドメインコントローラーにローカルでログオンする権限を付与する」を参照してください。

AWS Systems Manager を使用したリモートデスクトップ問題のトラブルシューティング

AWS Systems Manager では、RDP を使用して Windows インスタンスに接続する際の問題のトラブルシューティングを行うことができます。

AWSSupport-TroubleshootRDP

AWSSupport-TroubleshootRDP 自動化ドキュメントでは、リモートデスクトッププロトコル (RDP) の接続に影響する可能性があるターゲットインスタンスの一般設定 (RDP ポート、ネットワークレイヤー認証 (NLA)、Windows ファイアウォールプロファイルなど) を確認または修正できます。デフォルトでは、このドキュメントは以上の設定の値を読み取って出力します。

AWSSupport-TroubleshootRDP オートメーションドキュメントは、EC2 インスタンス、オンプレミスインスタンス、および AWS Systems Manager (マネージドインスタンス) との使用が有効になっている仮想マシン (VM) で使用できます。また、Systems Manager との使用が有効になっていない EC2 Windows Server インスタンスでも使用できます。AWS Systems Manager で使用するのインスタンスを有効にする方法については、「AWS Systems Manager ユーザーガイド」の「マネージドノード」を参照してください。

AWSSupport-TroubleshootRDP ドキュメントの使用に関するトラブルシューティングを行うには

1. Systems Manager コンソールにログインします。
2. 障害が発生した インスタンスと同じリージョンで操作していることを確認します。
3. 左側のナビゲーションペインから [Documents] (ドキュメント) を選択します。
4. [Owned by Amazon] (Amazon が所有) タブで、検索フィールドに `AWSSupport-TroubleshootRDP` と入力します。`AWSSupport-TroubleshootRDP` ドキュメントが表示されたら、それを選択します。
5. [Execute automation (自動化の実行)] を選択してください。
6. [Execution Mode (実行モード)] で、[Simple execution (シンプルな実行)] を選択します。
7. [入力パラメーター] の [InstanceId] フィールドで、[Show interactive instance picker (インタラクティブなインスタンスピッカーを表示)] を有効にします。
8. Amazon EC2 インスタンスを選択します。
9. 例を確認し、[Execute (実行)] を選択します。
10. 実行の進行状況をモニタリングするには、[Execution status (実行ステータス)] で、ステータスが [保留中] から [成功] に変わるのを待ちます。[出力] を展開して結果を表示します。個別のス

テップの出力を表示するには、[Executed Steps (実行済みのステップ)] で [Step ID (ステップ ID)] から項目を選択します。

AWSSupport-TroubleshootRDP の例

以下の例では、AWSSupport-TroubleshootRDP を使用して一般的なトラブルシューティングタスクを実行する方法を示します。AWS CLI start-automation-execution コマンドの例を使用するか、提供されている AWS マネジメントコンソール へのリンクを使用できます。

Example例: 現在の RDP のステータスを確認する

AWS CLI:

```
aws ssm start-automation-execution --document-name "AWSSupport-TroubleshootRDP" --
parameters "InstanceId=i-1234567890abcdef0, Action=Custom" --region region_code
```

AWS Systems Manager コンソール:

```
https://console.aws.amazon.com/systems-manager/automation/execute/AWSSupport-
TroubleshootRDP?region=region#documentVersion=$LATEST
```

Example例: Windows ファイアウォールを無効にする

AWS CLI:

```
aws ssm start-automation-execution --document-name "AWSSupport-TroubleshootRDP"
 --parameters "InstanceId=i-1234567890abcdef0, Action=Custom, Firewall=Disable" --
region region_code
```

AWS Systems Manager コンソール:

```
https://console.aws.amazon.com/systems-manager/automation/execute/AWSSupport-
TroubleshootRDP?region=region_code#documentVersion=$LATEST&Firewall=Disable
```

Example例: ネットワークレベル認証を無効にする

AWS CLI:

```
aws ssm start-automation-execution --document-name "AWSSupport-TroubleshootRDP" --
parameters "InstanceId=i-1234567890abcdef0, Action=Custom, NLASettingAction=Disable" --
region region_code
```

AWS Systems Manager コンソール:

```
https://console.aws.amazon.com/systems-manager/automation/execute/AWSSupport-
TroubleshootRDP?region=region_code#documentVersion
```

Example例: RDP サービスのスタートアップタイプを [自動] に設定して RDP サービスを開始する

AWS CLI:

```
aws ssm start-automation-execution --document-name "AWSSupport-TroubleshootRDP" --
parameters "InstanceId=i-1234567890abcdef0, Action=Custom, RDPServiceStartupType=Auto,
 RDPServiceAction=Start" --region region_code
```

AWS Systems Manager コンソール:

```
https://console.aws.amazon.com/systems-manager/automation/execute/
AWSSupport-TroubleshootRDP?region=region_code#documentVersion=
$LATEST&RDPServiceStartupType=Auto&RDPServiceAction=Start
```

Example例: デフォルトの RDP ポート (3389) を復元する

AWS CLI:

```
aws ssm start-automation-execution --document-name "AWSSupport-TroubleshootRDP" --
parameters "InstanceId=i-1234567890abcdef0, Action=Custom, RDPPortAction=Modify" --
region region_code
```

AWS Systems Manager コンソール:

```
https://console.aws.amazon.com/systems-manager/automation/execute/AWSSupport-
TroubleshootRDP?region=region_code#documentVersion=$LATEST&RDPPortAction=Modify
```

Example例: リモート接続を許可する

AWS CLI:

```
aws ssm start-automation-execution --document-name "AWSSupport-TroubleshootRDP" --
parameters "InstanceId=i-1234567890abcdef0, Action=Custom, RemoteConnections=Enable" --
region region_code
```

AWS Systems Manager コンソール:

```
https://console.aws.amazon.com/systems-manager/automation/execute/AWSSupport-
TroubleshootRDP?region=region_code#documentVersion=$LATEST&RemoteConnections=Enable
```

AWSSupport-ExecuteEC2Rescue

AWSSupport-ExecuteEC2Rescue 自動化ドキュメントでは、EC2Rescue for Windows Server を使用して EC2 インスタンスの接続と RDP の問題のトラブルシューティングと復元が自動的に行われます。詳細については、「到達不可能なインスタンスでの EC2Rescue ツールの実行」を参照してください。

AWSSupport-ExecuteEC2Rescue 自動化ドキュメントは、インスタンスの停止と再起動を必要とします。Systems Manager Automation は、インスタンスを停止して Amazon マシンイメージ (AMI) を作成します。インスタンスストアボリュームに保存されているデータは失われます。Elastic IP アドレスを使用していない場合は、パブリック IP アドレスが変わります。詳細については、「AWS Systems Manager ユーザーガイド」の「到達不可能なインスタンスでの EC2Rescue ツールの実行」を参照してください。

AWSSupport-ExecuteEC2Rescue ドキュメントの使用に関するトラブルシューティングを行うには

1. Systems Manager コンソールを開きます。
2. 障害が発生した Amazon EC2 インスタンスと同じリージョンで操作していることを確認します。
3. ナビゲーションペインで、[ドキュメント] を選択します。
4. `AWSSupport-ExecuteEC2Rescue` ドキュメントを検索して選択し、[自動化を実行] を選択します。
5. [Execution Mode (実行モード)] で、[Simple execution (シンプルな実行)] を選択します。
6. [入力パラメーター] セクションの [UnreachableInstanceId] に、到達不可能なインスタンスの Amazon EC2 インスタンス ID を入力します。
7. (オプション) Amazon EC2 インスタンスのトラブルシューティング用にオペレーティングシステムレベルのログを収集する場合は、[LogDestination] に Amazon Simple Storage Service (Amazon S3) バケット名を入力します。ログは、指定したバケットに自動的にアップロードされます。
8. [Execute (実行)] を選択します。
9. 実行の進行状況をモニタリングするには、[Execution status (実行ステータス)] で、ステータスが [保留中] から [成功] に変わるのを待ちます。[出力] を展開して結果を表示します。個別のス

テップの出力を表示するには、[Executed Steps (実行済みのステップ)] で [Step ID (ステップ ID)] を選択します。

リモートレジストリを使用して EC2 インスタンスでリモートデスクトップを有効にする

到達不能なインスタンスが AWS Systems Manager Session Manager で管理されていない場合は、リモートレジストリを使用してリモートデスクトップを有効にできます。

1. EC2 コンソールから、到達不能なインスタンスを停止します。
2. 到達不能なインスタンスのルートボリュームをデタッチし、そのボリュームを、同じアベイラビリティーゾーン内の到達可能なインスタンスにストレージボリュームとしてアタッチします。到達可能なインスタンスが同じアベイラビリティーゾーンにない場合は起動します。到達不能なインスタンス上のルートボリュームのデバイス名を書き留めます。
3. 到達可能なインスタンスで、Disk Management を開きます。これを行うには、コマンドプロンプトウィンドウで次のコマンドを実行します。

   ```
   diskmgmt.msc
   ```

4. 新たにアタッチされた (到達不能なインスタンスからの) ボリュームを右クリックし、[オンライン] を選択します。
5. Windows レジストリエディタを開きます。これを行うには、コマンドプロンプトウィンドウで次のコマンドを実行します。

   ```
   regedit
   ```

6. レジストリエディターで、[HKEY_LOCAL_MACHINE] を選択した後、[ファイル]、[Hive の読み込み] の順に選択します。
7. 接続されているボリュームのドライブを選択し、`\Windows\System32\config\` に移動し、`SYSTEM` を選択して、[開く] を選択します。
8. [キー名] にハイブの一意の名前を入力し、[OK] を選択します。
9. レジストリに変更を加える前に、レジストリ Hive をバックアップします。

   a. レジストリエディターのコンソールツリーで、読み込んだハイブ (HKEY_LOCAL_MACHINE*your-key-name*) を選択します。

   b. [ファイル]、[エクスポート] の順に選択します。

c. [Export Registry File (レジストリファイルのエクスポート)] ダイアログボックスで、バックアップコピーを保存する場所を選択し、[File name (ファイル名)] フィールドにバックアップファイルの名前を入力します。

d. [保存] を選択します。

10. レジストリエディターで HKEY_LOCAL_MACHINE*your key name*\ControlSet001\Control\Terminal Server に移動し、詳細ペインで [fDenyTSConnections] をダブルクリックします。

11. [Edit DWORD (DWORD の編集)] 値ボックスで、[Value data (値のデータ)] フィールドに 0 と入力します。

12. [OK] を選択します。

> **Note**
>
> [Value data](値のデータ) フィールドの値が 1 の場合、インスタンスはリモートデスクトップ接続を拒否します。0 の値は、リモートデスクトップ接続を許可します。

13. レジストリエディターで [HKEY_LOCAL_MACHINE*your-key-name]* を選択した後、[ファイル]、[ハイブのアンロード] の順に選択します。

14. レジストリエディターと Disk Management を閉じます。

15. EC2 コンソールから、ルートボリュームを到達可能なインスタンスからデタッチし、到達不能なインスタンスに再アタッチします。到達不能なインスタンスにボリュームをアタッチする際は、先に保存してあったデバイス名を [デバイス] フィールドに入力します。

16. 到達できないインスタンスを再び開始します。

プライベートキーを紛失しました。Windows インスタンスに接続するにはどうすればよいですか?

新しく起動した Windows インスタンスに接続する際、インスタンスの起動時に指定したキーペアのプライベートキーを使用して、管理者アカウントのパスワードを復号します。

管理者パスワードを紛失し、プライベートキーを使用できなくなった場合は、パスワードをリセットするか、新しいインスタンスを作成する必要があります。詳細については、「Amazon EC2 Windows インスタンスの Windows 管理者パスワードをリセットする」を参照してください。Systems Manager ドキュメントを使用してパスワードをリセットする手順については、「AWS

Systems Manager ユーザーガイド」の「EC2 インスタンスで、パスワードと SSH キーをリセットする」を参照してください。

Amazon EC2 Windows インスタンスの開始に関する問題のトラブルシューティング

次はAmazon EC2 Windows インスタンスのパスワードとアクティベーションに関する問題の解決に役立つトラブルシューティングのヒントです。

問題

- "パスワードは使用できません"
- "パスワードはまだ使用できません"
- "Windows パスワードを取得できません"
- "メタデータサービスを待っています"
- "Windows のライセンス認証ができません"
- "Windows が正規品ではありません (0x80070005) "
- "ライセンスを発行できるターミナルサーバーライセンスサーバーがありません"
- 「一部の設定は当組織によって管理されています」

"パスワードは使用できません"

リモートデスクトップを使用して Windows インスタンスに接続するにはアカウントとパスワードを指定する必要があります。提供されるアカウントとパスワードはインスタンスを起動するときに使用した AMI に基づいています。管理者アカウント用に自動生成されたパスワードを取得することも、AMI が作成された元のインスタンスで使われていたアカウントとパスワードを使用することもできます。

カスタム Windows AMI を使用して起動したインスタンスの管理者アカウントのパスワードを生成できます。パスワードを生成するにはAMI を作成する前にオペレーティングシステムでいくつかの設定を構成する必要があります。詳細についてはAmazon EBS-backed AMI を作成するを参照してください。

ランダムなパスワードを生成するように Windows インスタンスが設定されていない場合、コンソールを使用して自動生成パスワードを取得しようとすると、次のエラーメッセージが表示されます。

```
Password is not available.
The instance was launched from a custom AMI, or the default password has changed. A
password cannot be retrieved for this instance. If you have forgotten your password,
 you can
reset it using the Amazon EC2 configuration service. For more information, see
 Passwords for a
Windows Server instance.
```

インスタンスのコンソール出力を確認して、インスタンスを起動するのに使用した AMI がパスワード生成を無効にして作成されたかどうかを調べます。パスワード生成が無効になっている場合、コンソール出力には次の表示が含まれます。

```
Ec2SetPassword: Disabled
```

パスワード生成が無効で、元のインスタンスのパスワードを覚えていない場合はこのインスタンスのパスワードをリセットできます。詳細については[Amazon EC2 Windows インスタンスの Windows 管理者パスワードをリセットする]を参照してください。

"パスワードはまだ使用できません"

リモートデスクトップを使用して Windows インスタンスに接続するにはアカウントとパスワードを指定する必要があります。提供されるアカウントとパスワードはインスタンスを起動するときに使用した AMI に基づいています。管理者アカウント用に自動生成されたパスワードを取得することも、AMI が作成された元のインスタンスで使われていたアカウントとパスワードを使用することもできます。

パスワードは数分以内に使用可能となります。パスワードが使用できない場合、コンソールを使用して自動生成パスワードを取得すると、次のエラーメッセージが表示されます。

```
Password not available yet.
Please wait at least 4 minutes after launching an instance before trying to retrieve
 the
auto-generated password.
```

4 分以上経ってもまだパスワードを取得できない場合、インスタンスの起動エージェントがパスワードを生成するように設定されていない可能性があります。これはコンソール出力が空であるかチェックすることで、確認できます。詳細については[コンソールの出力を取得できない]を参照してください。

管理ポータル へのアクセスに使用されている AWS Identity and Access Management (IAM) アカウントで、ec2:GetPasswordData アクションが許可されていることも確認します。IAM のアクセス許可の詳細については「IAM とは」を参照してください。

"Windows パスワードを取得できません"

管理者アカウントの自動生成パスワードを取得するにはインスタンスの起動時に指定したキーペアのプライベートキーを指定する必要があります。インスタンスの起動時にキーペアを指定しなかった場合、次のメッセージが表示されます。

```
Cannot retrieve Windows password
```

このインスタンスを終了し、同じ AMI を使用して新しいインスタンスを起動して、キーペアを指定することができます。

"メタデータサービスを待っています"

Windows インスタンスはインスタンスメタデータから情報を取得した後、起動します。デフォルトではWaitForMetaDataAvailable 設定により、必ず EC2Config サービスがインスタンスメタデータにアクセスできるようになってから起動プロセスが続行されます。詳細についてはインスタンスメタデータを使用して EC2 インスタンスを管理するを参照してください。

インスタンスがインスタンスの接続性テストに合格しない場合はこの問題を解決するため、次の操作を試してください。

- VPC の CIDR ブロックを確認します。Windows インスタンスはIP アドレス範囲が 224.0.0.0 から 255.255.255.255 (クラス D とクラス E の IP アドレス範囲) の VPC で起動された場合、正しく起動できません。これらの IP アドレス範囲は予約済みで、ホストデバイスに割り当てることはできません。RFC 1918 に指定されているように、プライベート (パブリックにルーティングできない) IP アドレス範囲からの CIDR ブロックを持つ VPC を作成することをお勧めします。
- システムが静的 IP アドレスで設定されている可能性があります。ネットワークインターフェイスを作成して、インスタンスにアタッチします。
- 接続できない Windows インスタンスで DHCP を有効にするには

  1. 影響のあるインスタンスを停止し、ルートボリュームをデタッチします。
  2. 影響のあるインスタンスと同じアベイラビリティーゾーンで一時インスタンスを起動します。

> **Warning**
>
> 一時インスタンスが元のインスタンスと同じ AMI に基づいている場合、追加の手順を完了する必要があります。この手順を実行しない場合、ディスク署名の競合によって、ルートボリュームを復元した後、元のインスタンスを起動できなくなります。または一時インスタンスとして別の AMI を選択してください。例えば、元のインスタンスで Windows サーバー 2016 用の AWS Windows AMI を使用している場合、Windows サーバー 2019 用の AWS Windows AMI を使用して一時インスタンスを起動します。

3. 影響のあるインスタンスから一時インスタンスにルートボリュームをアタッチします。一時インスタンスに接続し、[ディスク管理] ユーティリティを開いて、ドライブをオンラインにします。
4. 一時インスタンスから [レジエディット] を開き、[HKEY_LOCAL_MACHINE] を選択してください。[ファイル] メニューの [ロードハイブ] を選択してください。ドライブを選択して、ファイル [Windows\System32\config\SYSTEM] を開き、プロンプトが表示されたらキー名を指定します (任意の名前を使用できます)。
5. ロードしたキーを選択し、ControlSet001\Services\Tcpip\Parameters\Interfaces に移動します。GUID ごとに、それぞれのネットワークインターフェイスが表示されています。適切なネットワークインターフェイスを選択してください。DHCP が無効で、静的 IP アドレスが割り当てられている場合、EnableDHCP は 0 に設定されています。DHCP を有効にするにはEnableDHCP を 1 に設定し、次のキーが存在する場合は削除します: NameServer、SubnetMask、IPAddress、DefaultGateway。再度キーを選択し、[ファイル] メニューの [アンロードハイブ] を選択してください。

> **Note**
>
> 複数のネットワークインターフェイスがある場合、DHCP を有効にするに正しいインターフェイスを指定する必要があります。正しいネットワークインターフェイスを特定するには次のキー値 NameServer、SubnetMask、IPAddress、DefaultGateway を確認します。これらの値は前のインスタンスの静的設定を表示します。

6. (オプション) DHCP がすでに有効な場合、メタデータサービスへのルーティングがない可能性があります。EC2Config を更新すると、この問題を解決できます。

a. EC2Config サービスの最新バージョンをダウンロードしてインストールします。このサービスをインストールする方法の詳細についてはthe section called “EC2Config のインストール”を参照してください。

b. .zip ファイルを、アタッチしたドライブの Temp ディレクトリに展開します。

c. [レジエディット] を開き、[HKEY_LOCAL_MACHINE] を選択してください。[ファイル] メニューの [ロードハイブ] を選択してください。ドライブを選択して、ファイル [Windows\System32\config\SOFTWARE] を開き、プロンプトが表示されたらキー名を指定します (任意の名前を使用できます)。

d. ロードしたキーを選択し、Microsoft\Windows\CurrentVersion に移動します。RunOnce キーを選択してください。(このキーが存在しない場合、[CurrentVersion] を右クリックし、[新規] をポイントし、[キー] を選択して、キーに RunOnce という名前をつけてください。) 右クリックして [新規] をポイントし、[tring Value] を選択してください。名前に Ec2Install、データに C:\Temp\Ec2Install.exe -q と入力してください。

e. 再度キーを選択し、[ファイル] メニューの [アンロードハイブ] を選択してください。

7. (オプション) 一時インスタンスが元のインスタンスと同じ AMI に基づいている場合、以下の手順を完了する必要があります。この手順を実行しない場合、ディスク署名の競合のためルートボリュームを復元した後、元のインスタンスを起動できなくなります。

> **Warning**
>
> 以下の手順ではレジストリエディタを使用して Windows レジストリを編集する方法を説明します。Windows レジストリに慣れていない場合や、レジストリエディターを使用して安全に変更する方法については「レジストリを構成する」を参照してください。

a. コマンドプロンプトを開き、「regedit.exe」と入力して、[Enter] を押します。

b. [レジストリエディタ] で、コンテキストメニュー (右クリック) から [HKEY_LOCAL_MACHINE] を選択し、[検索] を選択してください。

c. [Windows Boot Manager] を入力して、[次を検索] を選択してください。

d. 11000001 というキーを選択してください。このキーは前の手順で検索したキーの兄弟です。

e. 右のペインで [Element] を選択し、コンテキストメニュー (右クリック) から[変更] を選択してください。

f. データのオフセット 0x38 で 4 バイトのディスク署名を見つけます。バイトの順序を逆にしてディスク署名を作成し、書き留めます。例えば、次のデータで表されるディスク署名は E9EB3AA5 です。

```
...
0030  00 00 00 00 01 00 00 00
0038  A5 3A EB E9 00 00 00 00
0040  00 00 00 00 00 00 00 00
...
```

g. コマンドプロンプトウィンドウで、次のコマンドを実行して Microsoft DiskPart を起動します。

```
diskpart
```

h. 次の DiskPart コマンドを実行して、ボリュームを選択してください。(ディスク番号が 1 であることを確認するにはディスク管理ユーティリティを使用します。)

```
DISKPART> select disk 1

Disk 1 is now the selected disk.
```

i. 次の DiskPart コマンドを実行して、ディスク署名を取得します。

```
DISKPART>  uniqueid disk

Disk ID: 0C764FA8
```

j. 前の手順で表示されたディスク署名が、以前に書き留めた BCD のディスク署名と一致しない場合は次の DiskPart コマンドを使用してディスク署名を変更して一致させます。

```
DISKPART> uniqueid disk id=E9EB3AA5
```

8. [ディスク管理] ユーティリティを使用して、ドライブをオフラインにします。

> **Note**
>
> 一時インスタンスが影響を受けるインスタンスと同じ OS を実行している場合、ドライブは自動的にオフラインになるため、手動でオフラインにする必要はありません。

9. ボリュームを一時インスタンスからデタッチします。一時インスタンスをそれ以上使用しない場合は終了しても構いません。
10. /dev/sda1 としてボリュームをアタッチして、影響のあるインスタンスのルートボリュームを復元します。
11. 影響のあるインスタンスを開始します。

インスタンスに接続されたら、インスタンスからインターネットブラウザを開いて、次のメタデータサーバーの URL を入力してください。

```
http://169.254.169.254/latest/meta-data/
```

メタデータサーバーに接続できない場合は問題解決のために以下を試してください。

- EC2Config サービスの最新バージョンをダウンロードしてインストールします。このサービスをインストールする方法の詳細についてはthe section called “EC2Config のインストール”を参照してください。
- Windows インスタンスが RedHat PV ドライバーを実行しているかどうかを確認します。実行している場合、Citrix PV ドライバーに更新してください。詳細についてはthe section called “PV ドライバーのアップグレード”を参照してください。
- ファイアウォール、IPSec、およびプロキシ設定により、メタデータサービス (169.254.169.254) または AWS KMS サーバーへの送信トラフィックがブロックされていないことを確認します (アドレスは`TargetKMSServer` の `C:\Program Files\Amazon\Ec2ConfigService\Settings\ActivationSettings.xml` 要素で指定されています)。
- 次のコマンドを使用して、メタデータサービス (169.254.169.254) にルーティングされていることを確認します。

  ```
  route print
  ```

- インスタンスのアベイラビリティーゾーンに影響を与える可能性があるネットワークの問題を確認します。http://status.aws.amazon.com/ に移動します。

"Windows のライセンス認証ができません"

Windows インスタンスはWindows AWS KMS のアクティブ化を使用します。インスタンスから `A problem occurred when Windows tried to activate. Error Code 0xC004F074` サーバーにアクセスできない場合、AWS KMS というメッセージが表示される可能性があります。Windows は 180 日ごとにライセンス認証を行う必要があります。EC2Config では確実に、引き続き Windows でアクティブ化されるように、アクティブ化の期限が切れる前に AWS KMS サーバーに接続します。

Windows のライセンス認証の問題が発生した場合はこの問題を解決するため、次の手順に従います。

EC2Config の場合 (Windows サーバー 2012 R2 AMI 以前)

1. EC2Config サービスの最新バージョンをダウンロードしてインストールします。このサービスをインストールする方法の詳細についてはthe section called “EC2Config のインストール”を参照してください。
2. インスタンスにログインして、`C:\Program Files\Amazon\Ec2ConfigService\Settings\config.xml` のファイルを開きます。
3. `config.xml` ファイルで Ec2WindowsActivate プラグインを見つけます。状態を [Enabled] に変更し、変更を保存します。
4. [Windows Services snap-in] で、EC2Config サービスを再開するか、インスタンスを再起動します。

これによってライセンス認証の問題が解決しない場合は追加の手順に従います。

1. AWS KMS ターゲットを設定: C:\> slmgr.vbs /skms 169.254.169.250:1688
2. Windows を有効化: C:\> slmgr.vbs /ato

EC2Launch の場合 (Windows サーバー 2016 AMI 以降)

1. 管理者権限を持つ PowerShell プロンプトから、EC2Launch モジュールをインポートします。

```
PS C:\> Import-Module "C:\ProgramData\Amazon\EC2-Windows\Launch\Module
\Ec2Launch.psd1"
```

2. Add-ルーター 関数を呼び出して、新しいルートのリストを表示します。

```
PS C:\> Add-Routes
```

3. Set-ActivationSettings 関数を呼び出します。

```
PS C:\> Set-Activationsettings
```

4. 次のスクリプトを実行して、Windows のライセンス認証を行います。

```
PS C:\> cscript "${env:SYSTEMROOT}\system32\slmgr.vbs" /ato
```

EC2Config と EC2Launch の両方で、それでもまだライセンス認証エラーを受け取る場合は次の情報を確認します。

- AWS KMS サーバーにルーティングされていることを確認します。`C:\Program Files\Amazon\Ec2ConfigService\Settings\ActivationSettings.xml` を開き、`TargetKMSServer` エレメントを配置します。以下のコマンドを実行し、これらの AWS KMS サーバーのアドレスが表示されるかどうかを確認します。

  ```
  route print
  ```

- AWS KMS クライアントのキーが設定されていることを確認します。次のコマンドを実行し、出力を確認します。

  ```
  C:\Windows\System32\slmgr.vbs /dlv
  ```

  出力に Error: product key not found が含まれる場合、AWS KMS クライアントのキーは設定されていません。AWS KMS クライアントのキーが設定されていない場合はMicrosoft の記事 (AWS KMS クライアントセットアップキー) の説明に従ってクライアントのキーを検索し、次のコマンドを実行して AWS KMS クライアントのキーを設定します。

  ```
  C:\Windows\System32\slmgr.vbs /ipk client_key
  ```

- システムの時刻とタイムゾーンが正しいことを確認します。UTC 以外のタイムゾーンを使用している場合は時刻が正確であることを確認するために、レジストリキー `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\TimeZoneInformation\RealTimeIsUniversal` を追加して 1 に設定します。
- Windows ファイアウォールが有効な場合、次のコマンドを使用して一時的に無効にします。

```
netsh advfirewall set allprofiles state off
```

"Windows が正規品ではありません (0x80070005) "

Windows インスタンスはWindows AWS KMS のアクティブ化を使用します。インスタンスでライセンス認証のプロセスを完了できない場合、Windows が正規品でないことを示しています。

"Windows のライセンス認証ができません" をお試しください。

"ライセンスを発行できるターミナルサーバーライセンスサーバーがありません"

Windows サーバー ではデフォルトで、同時に 2 人のリモートデスクトップ接続のユーザーにライセンスが付与されます。2 人を超えるユーザーに、Windows インスタンスへのリモートデスクトップ接続による同時接続を提供する必要がある場合はリモートデスクトップサービスクライアントアクセスライセンス (CAL) を購入し、リモートデスクトップセッションホストおよびリモートデスクトップライセンスサーバーの役割をインストールします。

次の点を確認します。

- 最大同時 RDP セッション数を超えている。
- Windows のリモートデスクトップサービスの役割をインストールした。
- ライセンスの有効期限が切れている。ライセンスの有効期限が切れている場合、Windows インスタンスにユーザーとして接続できません。以下を試すことができます。
  - `/admin` パラメータを使用して、コマンドラインから、たとえば次のようにインスタンスに接続します。

    ```
    mstsc /v:instance /admin
    ```

    詳細についてはMicrosoft の記事のコマンドラインからリモートデスクトップにアクセスするを参照してください。
  - データを回復するにはインスタンスを停止し、Amazon EBS ボリュームをデタッチして、同じアベイラビリティーゾーンにある別のインスタンスにアタッチします。

「一部の設定は当組織によって管理されています」

最新の Windows サーバー AMI から起動したインスタンスには「一部の設定は当組織によって管理されています」で始まる Windows 更新 ダイアログメッセージが表示される場合があります。このメッセージは Windows サーバー 内の変更の結果として表示され、Windows 更新 の動作やお客様による更新設定管理能力には影響を及ぼしません。

警告を削除するには

1. `gpedit.msc` を開き、[コンピュータの構成]、[管理用テンプレート]、[Windows コンポーネント]、[Windows 更新] の順に移動します。[Configure Automatic 更新 (自動更新の設定)] を編集し、[enabled (有効)] に設定します。
2. コマンドプロンプトで、gpupdate /force を使用してグループポリシーを更新します。
3. Windows 更新 の設定を閉じて再度開きます。設定が組織によって管理されていることを示す上記のメッセージが表示され、その後に次のように表示されます。"計測対象の接続 (料金がかかる場合があります) を除き、アップデートは自動的にダウンロードされます。その場合、Windows が正常に動作するために必要なアップデートが自動的にダウンロードされます。"
4. `gpedit.msc` に戻り、グループポリシーを [設定されていません] に戻します。gpupdate /force をもう一度実行します。
5. コマンドプロンプトを閉じ、数分待ちます。
6. Windows 更新 の設定を再度開きます。「一部の設定は当組織によって管理されています。」というメッセージは表示されません。

Amazon EC2 Windows インスタンスの問題のトラブルシューティング

次は、Amazon EC2 Windows インスタンスの問題の解決に役立つトラブルシューティングのヒントです。

問題

- AWS Systems Manager Sessions Manager を Windows Server 2025 インスタンスに接続できません
- EBS ボリュームが Windows Server 2016 および 2019 で初期化されない
- ディレクトリサービス復元モード (DSRM) で EC2 Windows インスタンスを起動する

- インスタンスのネットワーク接続が失われる、または、スケジュールされたタスクが予定通りに実行されない
- コンソールの出力を取得できない
- Windows Server 2012 R2 をネットワークで使用できない
- ディスク署名の衝突

AWS Systems Manager Sessions Manager を Windows Server 2025 インスタンスに接続できません

AWS Systems Manager Sessions Manager を Windows Server 2025 インスタンスに接続する際に問題が発生する場合があります。この問題に対処するにはインスタンスにログオンし、`Settings > Apps > Optional Features` に移動して `WMIC` を追加します。SSM エージェントサービスを再起動するか、インスタンスを再起動したら、Sessions Manager が接続詞ます。

次の PowerShell コマンドを使用して同じアクションを実行することもできます。

```
Start-Process -FilePath "$env:SystemRoot\system32\Dism.exe" -ArgumentList @('/
Online', '/Add-Capability', '/CapabilityName:WMIC~~~~') -Wait; Restart-Service -Name
 AmazonSSMAgent
```

EBS ボリュームが Windows Server 2016 および 2019 で初期化されない

Windows Server 2016 および 2019 用の Amazon マシンイメージ (AMI) から作成されたインスタンスでは、さまざまなスタートアップタスク (例: EBS ボリュームの初期化) に EC2Launch v1 エージェントが使用されます。デフォルトでは、EC2Launch v1 はセカンダリボリュームを初期化しません。ただし、これらのディスクを自動的に初期化するには、次のように EC2Launch v1 を設定します。

ドライブ文字をボリュームにマッピングする

1. 設定するインスタンスに接続し、`C:\ProgramData\Amazon\EC2-Windows\Launch\Config\DriveLetterMappingConfig.json` ファイルをテキストエディタで開きます。
2. ボリューム設定を次のように指定します。

```
{
"driveLetterMapping": [
```

```
 {
   "volumeName": "sample volume",
   "driveLetter": "H"
 }]
}
```

3. 変更内容を保存し、ファイルを閉じます。
4. Windows PowerShell を開き、次のコマンドを使用して、ディスクを初期化する EC2Launch v1 スクリプトを実行します。

```
PS C:\>  C:\ProgramData\Amazon\EC2-Windows\Launch\Scripts\InitializeDisks.ps1
```

インスタンスが起動するたびにディスクを初期化するには、-Schedule フラグを次のように追加します。

```
PS C:\>  C:\ProgramData\Amazon\EC2-Windows\Launch\Scripts\InitializeDisks.ps1 -
Schedule
```

EC2Launch v1 エージェントは、initializeDisks.ps1 などのインスタンス初期化スクリプトを InitializeInstance.ps1 スクリプトと並行して実行できます。InitializeInstance.ps1 スクリプトでインスタンスを再起動する場合は、インスタンスのスタートアップ時に実行される他のスケジュールされたタスクが中断される可能性があります。コンフリクトが発生しないように、インスタンスの初期化が最初に完了したことを確認するために initializeDisks.ps1 スクリプトにロジックを追加することをお勧めします。

> **Note**
>
> EC2Launch スクリプトがボリュームを初期化しない場合は、ボリュームがオンラインとなっていることを確認します。ボリュームがオフラインの場合は、次のコマンドを実行してすべてのディスクをオンラインにします。
>
> ```
> PS C:\> Get-Disk | Where-Object IsOffline -Eq $True | Set-Disk -IsOffline
>  $False
> ```

ディレクトリサービス復元モード (DSRM) で EC2 Windows インスタンスを起動する

Microsoft Active Directory を実行するインスタンスでシステム障害やその他の重要な問題が発生した場合、ディレクトリサービス復元モード (DSRM) と呼ばれる特殊なバージョンのセーフモードで起動することで、インスタンスをトラブルシューティングできます。DSRM では、Active Directory を修復または復元できます。

DSRM のドライバーサポート

DSRM を有効にしてインスタンスで起動する方法は、インスタンスが実行されているドライバーによって異なります。EC2 コンソールで、システムログからインスタンスのドライバーバージョンの詳細を表示できます。次の表に、DSRM でサポートされるドライバーを示します。

ドライバーのバージョン	サポートされる DSRM	次のステップ
Citrix PV 5.9	いいえ	バックアップからインスタンスを復元します。DSRM を有効にすることはできません。
AWS PV 7.2.0	いいえ	このドライバーでは DSRM がサポートされていませんが、インスタンスからルートボリュームをデタッチして、ボリュームのスナップショットを取得するか AMI を作成し、同じアベイラビリティーゾーン内の別のインスタンスにセカンダリボリュームとしてアタッチできます。その後、DSRM を有効にすることができます (このセクションで説明します)。
AWS PV 7.2.2 以降	はい	ルートボリュームのデタッチ、別のインスタンスへの接続、DSRM の有効化 (このセクションで説明しています)。
拡張ネットワーキング	はい	ルートボリュームのデタッチ、別のインスタンスへの接続、DSRM の有効化 (このセクションで説明しています)。

拡張ネットワーキングを有効にする方法については、「the section called “Elastic Network Adapter (ENA)”」を参照してください。AWS PV ドライバーのアップグレードについては、「Windows インスタンスでの PV ドライバーのアップグレード」を参照してください。

DSRM で起動するインスタンスを設定する

オペレーティング システムを実行する前に、EC2 Windows インスタンスでネットワーク接続は行われません。このため、キーボードの F8 ボタンを押して起動オプションを選択することはできません。次のいずれかの手順を使用して、DSRM で EC2 Windows Server インスタンスを起動する必要があります。

Active Directory が破損していて、インスタンスがまだ実行されていることが疑われる場合、[System Configuration] ダイアログボックスまたはコマンドプロンプトを使用して、DSRM で起動するようインスタンスを設定できます。

[System Configuration] ダイアログボックスを使用して DSRM でオンラインインスタンスを起動するには

1. [Run] ダイアログボックスで、「`msconfig`」と入力して Enter キーを押します。
2. [Boot] タブを選択します。
3. [Boot options] で、[Safe boot] を選択します。
4. [Active Directory repair] を選択し、[OK] を選択します。サーバーを再起動するよう求められます。

コマンドラインを使用して DSRM でオンラインインスタンスを起動するには

コマンドプロンプトウィンドウから次のコマンドを実行します。

```
bcdedit /set safeboot dsrepair
```

インスタンスがオフラインで到達不可能な場合は、ルートボリュームをデタッチし、別のインスタンスにアタッチして DSRM モードを有効にする必要があります。

DSRM でオフラインインスタンスを起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。

3. 影響を受けるインスタンスを探して選択します。[Instance state (インスタンスの状態)]、[Stop instance (インスタンスの停止)] の順に選択します。
4. [Launch instances (インスタンスの起動)] を選択し、影響のあるインスタンスと同じアベイラビリティーゾーンに一時インスタンスを作成します。別のバージョンの Windows を使用するインスタンスタイプを選択します。例えば、インスタンスが Windows Server 2016 である場合、Windows Server 2019 インスタンスを選択します。

   > **Important**
   >
   > 影響のあるインスタンスと同じアベイラビリティーゾーンにインスタンスを作成しない場合、影響のあるインスタンスのルートボリュームを新しいインスタンスにアタッチできません。

5. ナビゲーションペインの [Volumes] を選択してください。
6. 影響のあるインスタンスのルートボリュームを見つけます。ボリュームをデタッチし、先ほど作成した一時インスタンスにアタッチします。デフォルトのデバイス名 (xvdf) でアタッチしてください。
7. リモートデスクトップを使用して一時インスタンスに接続したら、Disk Management ユーティリティを使用してボリュームを有効にします。
8. コマンドプロンプトを開き、次のコマンドを入力します。D を、アタッチしたセカンダリボリュームの実際のドライブ文字と置き換えます。

   ```
   bcdedit /store D:\Boot\BCD /set {default} safeboot dsrepair
   ```

9. Disk Management ユーティリティで、先ほどアタッチしたドライブを選択し、右クリックコンテキストメニューを開いて、[オフライン] を選択します。
10. EC2 コンソールで、影響のあるボリュームを一時インスタンスからデタッチし、/dev/sda1 というデバイス名で元のインスタンスに再アタッチします。ボリュームをルートボリュームとして指定するには、このデバイス名を指定する必要があります。
11. インスタンスを起動します。
12. インスタンスが EC2 コンソールでヘルスチェックに合格したら、リモートデスクトップを使用してインスタンスに接続し、DSRM モードで起動することを確認します。
13. (オプション) この手順で作成した一時インスタンスを削除するか停止します。

インスタンスのネットワーク接続が失われる、または、スケジュールされたタスクが予定通りに実行されない

インスタンスを再起動するとネットワーク接続が失われる場合は、インスタンスの時刻設定が間違っている可能性があります。

デフォルトで、Windows インスタンスは協定世界時 (UTC) を使用します。別のタイムゾーンにインスタンスの時刻を設定して再起動すると、時刻がずれて、インスタンスの IP アドレスが一時的に失われます。最終的にインスタンスのネットワーク接続は復旧されますが、これには数時間かかることがあります。インスタンスがネットワーク接続を復旧するのにかかる時間は、UTC と他のタイムゾーンとの時差に左右されます。

時刻に関するこの同じ問題によって、スケジュールされたタスクが予定通りに実行されないことがあります。この場合、インスタンスの時刻が正しくないため、スケジュールされたタスクは予定通りに実行されません。

UTC 以外のタイムゾーンを永続的に使用するには、RealTimeIsUniversal レジストリキーを設定する必要があります。このキーがない場合、インスタンスは再起動後、UTC を使用します。

ネットワーク接続を失う原因となる時刻の問題を解決するには

1. 推奨 PV ドライバーを実行していることを確認します。詳細については、「the section called “PV ドライバーのアップグレード”」を参照してください。
2. 次のレジストリキーが存在し、1 に設定されていることを確認します: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\TimeZoneInformation\RealTimeIsUniversal。

コンソールの出力を取得できない

Windows インスタンスの場合は、インスタンスコンソールに Windows 起動プロセス中に実行されたタスクの出力が表示されます。Windows が正常に起動した場合、記録される最後のメッセージは `Windows is Ready to use` です。コンソールでイベントログメッセージを表示することもできますが、この機能は Windows のバージョンによってはデフォルトで有効になっていない場合があります。詳細については、「the section called “Windows 起動エージェント”」を参照してください。

Amazon EC2 コンソールを使用してインスタンスのコンソール出力を取得するには、インスタンスを選択してから、[アクション]、[モニタリングおよびトラブルシューティング]、[システムログの取得] の順に選択します。コマンドラインを使用してコンソール出力を取得するには、get-console-

output (AWS CLI) コマンド、または Get-EC2ConsoleOutput (AWS Tools for Windows PowerShell) コマンドを使用します。

Windows Server 2012 R2 以前を実行しているインスタンスで、コンソール出力が空の場合は、設定ファイルが正しく設定されていない、Windows が正しく起動しなかったなど、EC2Config サービスの問題を示している可能性があります。問題を修正するには、EC2Config の最新バージョンをダウンロードしてインストールします。詳細については、「the section called “EC2Config のインストール”」を参照してください。

Windows Server 2012 R2 をネットワークで使用できない

ネットワークで使用できない Windows Server 2012 R2 インスタンスのトラブルシューティングについては、「Windows Server 2012 R2 でインスタンスの再起動後にネットワークおよびストレージとの接続が失われる」を参照してください。

ディスク署名の衝突

ディスク署名の衝突をチェックして解決するには、EC2Rescue for Windows Server を使用します。または、次の手順を実行して、ディスク署名の問題を手動で解決できます。

> **Warning**
>
> 以下の手順ではレジストリエディタを使用して Windows レジストリを編集する方法を説明します。Windows レジストリに慣れていない場合や、レジストリエディターを使用して安全に変更する方法については「レジストリを構成する」を参照してください。

1. コマンドプロンプトを開き、「regedit.exe」と入力して、[Enter] を押します。
2. [レジストリエディタ] で、コンテキストメニュー (右クリック) から [HKEY_LOCAL_MACHINE] を選択し、[検索] を選択してください。
3. [Windows Boot Manager] を入力して、[次を検索] を選択してください。
4. `11000001` というキーを選択してください。このキーは前の手順で検索したキーの兄弟です。
5. 右のペインで [`Element`] を選択し、コンテキストメニュー (右クリック) から[変更] を選択してください。
6. データのオフセット 0x38 で 4 バイトのディスク署名を見つけます。これは、ブート構成データベース署名 (BCD) です。バイトの順序を逆にしてディスク署名を作成し、書き留めます。例えば、次のデータで表されるディスク署名は `E9EB3AA5` です。

```
...
0030  00 00 00 00 01 00 00 00
0038  A5 3A EB E9 00 00 00 00
0040  00 00 00 00 00 00 00 00
...
```

7. コマンドプロンプトウィンドウで、次のコマンドを実行して Microsoft DiskPart を起動します。

```
diskpart
```

8. `select disk` DiskPart コマンドを実行して、ディスクシグネチャが衝突しているボリュームのディスク番号を指定します。

> **Tip**
>
> ディスクシグネチャが衝突しているボリュームのディスク番号をチェックするには、[ディスク管理] ユーティリティを使用します。コマンドプロンプトを開き、「`compmgmt.msc`」と入力して、[Enter] を押します。左側のナビゲーションパネルで、[ディスク管理] をダブルクリックします。[ディスク管理] ユーティリティで、ディスクシグネチャが衝突しているオフラインボリュームのディスク番号をチェックします。

```
DISKPART> select disk 1
Disk 1 is now the selected disk.
```

9. 次の DiskPart コマンドを実行して、ディスク署名を取得します。

```
DISKPART>  uniqueid disk
Disk ID: 0C764FA8
```

10. 前の手順で表示されたディスクシグネチャが以前に書き留めたディスクシグネチャと一致しない場合は、次の DiskPart コマンドを使用してディスクシグネチャを変更し、一致させます。

```
DISKPART> uniqueid disk id=E9EB3AA5
```

ネットワーク経由の Windows インスタンスのカーネルデバッギング

Elastic Network Adapter (ENA) 用の KDNET 拡張モジュールは、Amazon Elastic Compute Cloud インスタンスの ENA を介してネットワーク経由で Windows カーネルデバッギングを可能にするハードウェアドライバーサポートレイヤーです。Windows デバッガー (WinDbg) で拡張モジュールを使用して、Windows を実行している EC2 インスタンスでカーネルレベルのデバッギングを実行できます。

カーネルデバッギングは、ブルースクリーンエラー (BSOD)、ドライバーの障害、EC2 Windows インスタンスの起動問題などの低レベルのオペレーティングシステムの問題の診断とトラブルシューティングに役立ちます。

内容

- 前提条件
- ステップ 1: デバッグホストに Windows デバッギングツールをインストールする
- ステップ 2: デバッグターゲットをセットアップする
- ステップ 3: デバッグホストでデバッギングセッションを開始する
- ステップ 4: デバッグターゲットを再起動する
- デバッギング後にデバッグ設定をクリーンアップする
- 制限事項
- 追加のメモ

前提条件

作業を開始する前に、次の項目があることを確認します。

同じサブネット内の 2 つの EC2 Windows インスタンス:

- デバッグホストインスタンス — Windows デバッガー (WinDbg) を実行します。
- デバッグターゲットインスタンス — デバッグするインスタンス。

インスタンスの起動方法の詳細については、「開始方法のチュートリアル」を参照してください。

ホストインスタンスとターゲットインスタンスのセキュリティグループは、KDNET デバッギングに使用されるポートでインバウンドおよびアウトバウンド UDP トラフィックを許可する必要がありま

す (推奨範囲: 50000 ~ 50039)。これをセットアップする最も簡単な方法は、それ自体から UDP トラフィックをソースとして許可するインバウンドルールを持つセキュリティグループを作成し、そのセキュリティグループを両方のインスタンスにアタッチすることです。詳細については、「Amazon VPC ユーザーガイド」の「セキュリティグループルール」を参照してください。

デバッグターゲットインスタンスは、次の Windows バージョン (またはそれ以降) のいずれかを実行する必要があります:

- ビルド番号 26100.7462 の Windows Server 2025 (2025 年 12 月パッチ)
- ビルド番号 26100.7309 の Windows 11 24H2
- ビルド番号 26200.7309 の Windows 11 25H2

> **Note**
>
> ENA 用の KDNET 拡張モジュールは Windows の一部として配布され、Windows の毎月の累積更新を通じてのみ更新できます。デバッグターゲットに常に最新の Windows KB がインストールされるようにして、その最新バージョンを持っていることを確認することをお勧めします。
> モジュールがデバッグターゲットに存在することを確認するには、昇格された PowerShell セッションで次のコマンドを実行します:
>
> ```
> Test-Path C:\Windows\system32\kd_02_1d0f.dll
> ```
>
> コマンドが `True` を返す場合、モジュールは使用可能です。

ステップ 1: デバッグホストに Windows デバッギングツールをインストールする

PowerShell セッションで次のコマンドを実行して、Windows デバッギングツールをデバッグホストインスタンスにインストールします:

```
winget install microsoft.windbg
```

インストール手順の詳細については、Microsoft ドキュメントの「Windows デバッガーのインストール」を参照してください。

インストール後、PowerShell セッションで次のコマンドを実行して、デバッガーが動作していることを確認します:

```
windbgx
```

WinDbg ウィンドウが開くはずです。その場合、インストールは成功しているので、ウィンドウを閉じることができます。

ステップ 2: デバッグターゲットをセットアップする

> **Note**
>
> カーネルデバッギングがアクティブな場合、デバッギングセッションに使用される ENA デバイスはデバッガートラフィック専用です。デバッギング中にデバッグターゲットインスタンスでインターネットアクセスを維持する必要がある場合は、開始する前にインスタンスに 2 番目の ENA をアタッチします。

デバッグターゲットで、昇格された PowerShell セッションを開き、次のステップを使用してカーネルデバッギングを設定します。

次のコマンドを実行して、インスタンスにアタッチされた ENA アダプターのバス、デバイス、および機能番号を一覧表示します。

```
Get-NetAdapter -Physical |
    Where-Object -Property PnPDeviceID -Match -Value '^PCI\\VEN_1D0F&DEV_EC2[01]&' |
    Get-NetAdapterHardwareInfo |
    Select-Object InterfaceDescription, BusNumber, DeviceNumber, FunctionNumber |
    Format-List
```

インスタンスに複数の ENA アダプターがアタッチされている場合

インスタンスに複数の ENA アダプターがアタッチされている場合は、次のコマンドを使用して、各物理アダプターをプライベート IP アドレスにマッピングします。これらの詳細は、EC2 → インスタンス → [インスタンス ID] → ネットワーク → ネットワークインターフェイスの AWS マネジメントコンソール で相互参照できます。これにより、特定のネットワークインターフェイス ID、プライベート IP、およびセキュリティグループを、ターゲットデバッギング用の OS レベルアダプタと関連付けることができます。

```
Get-NetAdapter -Physical |
```

```
    Where-Object PnPDeviceID -Match '^PCI\\VEN_1D0F&DEV_EC2[01]&' |
    ForEach-Object {
        $adapter = $_
        $hwInfo = $adapter | Get-NetAdapterHardwareInfo
        $ipInfo = Get-NetIPAddress -InterfaceIndex $adapter.InterfaceIndex -
AddressFamily IPv4
        [PSCustomObject]@{
            InterfaceDescription = $adapter.InterfaceDescription
            IPAddress            = $ipInfo.IPAddress
            BusNumber            = $hwInfo.BusNumber
            DeviceNumber         = $hwInfo.DeviceNumber
            FunctionNumber       = $hwInfo.FunctionNumber
        }
    } | Format-List
```

出力からのデバッギングに使用する ENA アダプターの BusNumber、DeviceNumber、および FunctionNumber の値を記録してください。

次のコマンドを実行して、カーネルデバッギングを設定します。プレースホルダー値を、ユーザー固有の設定に置き換えてください。

```
bcdedit /debug on
bcdedit /set loadoptions FORCEHVTONOTSHAREDEBUGDEVICE
bcdedit /dbgsettings net hostip:host-private-ip port:port-number key:encryption-key
 busparams:b.d.f
```

> **Note**
>
> 次のコマンドを実行して、loadoptions が存在するかどうかを確認します。値が返された場合は、その文字列をコピーして、;FORCEHVTONOTSHAREDEBUGDEVICE をそれに追加します。
>
> ```
> (bcdedit /enum) -match "loadoptions"
> ```

説明は以下のとおりです。

- *host-private-ip* — デバッグホストインスタンスのプライベート IPv4 アドレス。インスタンスが IPv6 のみのサブネットで起動される場合、KDNET での IPv6 の使用の詳細については、Microsoft ドキュメントの「IPv6 での KDNET のセットアップ」を参照してください。

- *port-number* — デバッギングセッションに使用するポート。推奨範囲は 50000 ~ 50039 (例: 50000) です。
- *encryption-key* — デバッギング接続を暗号化するために使用される 256 ビットキー。ピリオドで区切られた 4 つの 64 ビット値として指定されます。各 64 ビット値は、小文字 a ~ z と数字 0 ~ 9 のみを使用して、最大 13 文字にすることができます。特殊文字は使用できません。暗号化キーの例: 1kdnet2keys3.4kdnet5keys6.7kdnet8keys9.10kdnet11ke。
- *b.d.f* — デバッギングに使用される、bus.device.function の形式でフォーマットされている (0.3.0 など) ENA デバイスのバス、デバイス、および機能番号。

> **Tip**
>
> Windows ブートプロセスをデバッグするには、次のコマンドも実行します:
>
> ```
> bcdedit /bootdebug on
> ```

ステップ 3: デバッグホストでデバッギングセッションを開始する

ホストでデバッギングトラフィックを許可するには、WinDbg アプリケーションまたは特定の UDP ポートのファイアウォールルールを作成できます。

オプション 1: WinDbg アプリケーションを許可する

次のコマンドを実行して、WinDbg 実行可能ファイルを許可します。

```
$WinDbgxPath = "$env:LocalAppData\Microsoft\WindowsApps\WinDbgX.exe"
New-NetFirewallRule -DisplayName "Allow Inbound KDNET Connection" -Action Allow -
Program $WinDbgxPath
```

オプション 2: 特定の UDP ポートを許可する

または、カーネルデバッギング用に設定されたポートへのインバウンド UDP トラフィックを許可することもできます。*port-number* を、選択した KDNET ポートに置き換えます:

```
$DebugPort = port-number
New-NetFirewallRule -DisplayName "Allow Inbound KDNET Connection" -Direction Inbound -
LocalPort $DebugPort -Protocol UDP -Action Allow
```

> **Note**
>
> ファイアウォール設定は、ドメイングループポリシー (GPO) によって制限されるか、昇格された管理者アクセス許可が必要になる場合があります。コマンドが失敗した場合は、ネットワーク管理者にお問い合わせください。

デバッグターゲット設定に一致するポートとキーを使用して WinDbg を起動します。ユースケースに合わせて、Microsoft WinDbg コマンドラインリファレンスに記載されている追加のカーネルデバッギングオプションを指定できます。

```
windbgx -k net:port=port-number,key=encryption-key
```

WinDbg が開き、デバッグターゲットが接続するのを待機します。

ステップ 4: デバッグターゲットを再起動する

デバッグターゲットで、インスタンスを再起動してデバッギング接続を開始します。

```
shutdown -r -t 0
```

デバッグターゲットが再起動すると、デバッグホストの WinDbg は自動的にターゲットに接続します。WinDbg を使用して、カーネルの状態を検査し、ブレークポイントを設定し、問題を診断できるようになりました。

デバッギング後にデバッグ設定をクリーンアップする

デバッグターゲットで

デバッギングが終了したら、デバッグターゲットからカーネルデバッギング設定を削除して、通常の起動動作を復元します。デバッグターゲットで、昇格された PowerShell セッションを開き、次のコマンドを実行します:

```
bcdedit /debug off
bcdedit /dbgsettings LOCAL
bcdedit /deletevalue loadoptions
```

FORCEHVTONOTSHAREDEBUGDEVICE 以外の既存の `loadoptions` がある場合は、元の `loadoptions` で `bcdedit /set loadoptions` を実行して設定を復元する必要があります。

ブートデバッギングを有効にした場合は、以下も実行します:

```
bcdedit /bootdebug off
```

変更を有効にするためにインスタンスを再起動します。

デバッグホストで

以下を実行して、ファイアウォールルールを削除します:

```
Remove-NetFirewallRule -DisplayName "Allow Inbound KDNET Connection"
```

制限事項

ENA 用の KDNET 拡張モジュールは、現在以下をサポートしていません:

- 第 8 世代ノンメタル x86_64 インスタンスタイプ (例: `m8a.xlarge`)
- 第 7 世代 48xlarge ノンメタル x86_64 インスタンスタイプ (例: `m7a.48xlarge`)
- u7i インスタンスタイプ

モジュールは現在、Secure Boot が有効になっているインスタンスをサポートしていません。昇格された PowerShell セッションで `Confirm-SecureBootUEFI` を実行して、ステータスを確認できます。出力が `True` の場合、Secure Boot はアクティブです。「TPM」プレフィックスを持つ Amazon 提供のすべてのイメージでは、デフォルトで Secure Boot が有効になっていることに注意してください。

追加のメモ

デバッガーをターゲットインスタンスに接続する際に問題が発生した場合は、以下を確認してください:

- 必要な Windows Server ビルドバージョンや拡張モジュールの存在など、このガイドに記載されているすべての前提条件が満たされている。
- 両方のインスタンスにアタッチされたセキュリティグループは、設定されたデバッギングポートでそれらのインスタンス間のトラフィックを許可するように正しく設定されている。
- ホストインスタンスの Windows ファイアウォールルールは、設定されたポート上の 2 つのインスタンス間のネットワークトラフィックをブロックしない。

その他のガイダンスについては、Microsoft ドキュメントの「KDNET ネットワークカーネルデバッギングを手動でセットアップする」を参照してください。

Amazon EC2 Windows インスタンスの Windows 管理者パスワードをリセットする

Windows 管理者パスワードを紛失したり、パスワードが期限切れになったりしたため、Amazon EC2 Windows インスタンスに接続できなくなった場合、パスワードをリセットできます。

> **Note**
>
> ローカル管理者パスワードのリセットに必要な手動の手順を自動的に適用する AWS Systems Manager のオートメーションドキュメントがあります。詳細については、「AWS Systems Manager ユーザーガイド」の「EC2 インスタンスでのパスワードと SSH キーのリセット」を参照してください。

管理者パスワードを手動でリセットする場合は、EC2Launch v2、EC2Config、EC2Launch のいずれかを使用します。

- すべてのサポートされている Windows AMI (EC2Launch v2 エージェントを含む) の場合は、EC2Launch v2 を使用します。
- Windows Server 2016 より前の Windows AMI の場合は、EC2Config サービスを使用します。
- Windows Server 2016 以降の AMI の場合は、EC2Launch サービスを使用します。

これらの手順では、インスタンスの作成に使用したキーペアを紛失した場合に、インスタンスに接続する方法についても示します。Amazon EC2 では、パブリックキーを使用してパスワードなどのデータを暗号化し、プライベートキーを使用してそのデータを復号します。パブリックキーとプライベートキーは、キーペアと呼ばれます。Windows インスタンスでは、キーペアを使用して管理者パスワードを取得してから、RDP を使用してログインします。

> **Note**
>
> インスタンスでローカル管理者アカウントを無効にし、インスタンスが Systems Manager 用に設定されている場合は、EC2Rescue および Run Command を使用してローカル管理者パスワードを再度有効にしたり、リセットすることもできます。詳細については、

「Systems Manager Run Command での EC2Rescue for Windows Server の使用」を参照してください。

内容

- EC2Launch v2 を使用して EC2 インスタンスの Windows 管理者パスワードをリセットする
- EC2Launch を使用して EC2 インスタンスの Windows 管理者パスワードをリセットする
- EC2Config を使用して EC2 インスタンスの Windows 管理者パスワードをリセットする

EC2Launch v2 を使用して EC2 インスタンスの Windows 管理者パスワードをリセットする

Windows 管理者パスワードを紛失した場合、サポートされており EC2Launch v2 エージェントを含む Windows AMI を使用している場合は、EC2Launch v2 を使用して新しいパスワードを生成できます。

EC2Launch v2 エージェントを含まない Windows Server 2016 以降の AMI を使用している場合は、「EC2Launch を使用して EC2 インスタンスの Windows 管理者パスワードをリセットする」を参照してください。

EC2Launch v2 エージェントを含まない Windows Server 2016 より前の Windows Server AMI を使用している場合は、「EC2Config を使用して EC2 インスタンスの Windows 管理者パスワードをリセットする」を参照してください。

> **Note**
>
> インスタンスでローカル管理者アカウントを無効にし、インスタンスが Systems Manager 用に設定されている場合は、EC2Rescue および Run Command を使用してローカル管理者パスワードを再度有効にしたり、リセットすることもできます。詳細については、「Systems Manager Run Command での EC2Rescue for Windows Server の使用」を参照してください。

> **Note**
>
> ローカル管理者パスワードのリセットに必要な手動の手順を自動的に適用する AWS Systems Manager のオートメーションドキュメントがあります。詳細については、「AWS

Systems Manager ユーザーガイド」の「EC2 インスタンスでのパスワードと SSH キーのリセット」を参照してください。

EC2Launch v2 を使用して Windows 管理者パスワードをリセットするには、次の操作が必要です。

- ステップ 1: EC2Launch v2 エージェントが実行されていることを確認する
- ステップ 2: ルートボリュームをインスタンスからデタッチします
- ステップ 3: ボリュームを一時インスタンスにアタッチします。
- ステップ 4: .run-once ファイルを削除する
- ステップ 5: 元のインスタンスを再起動します。

ステップ 1: EC2Launch v2 エージェントが実行されていることを確認する

管理者パスワードのリセットを試みる前に、EC2Launch v2 エージェントがインストールされ、実行されていることを確認します。このセクションで後ほど、EC2Launch v2 エージェントを使用して管理者パスワードをリセットします。

EC2Launch v2 エージェントが実行されていることを確認するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [インスタンス] を選択し、パスワードのリセットが必要なインスタンスを選択します。この手順では、このインスタンスを元のインスタンスと呼びます。
3. [アクション]、[モニタリングとトラブルシューティング]、[システムログの取得] の順に選択します。
4. EC2 起動エントリ (例: Launch: EC2Launch v2 service v2.0.124) を見つけます。このエントリが表示されている場合、EC2Launch v2 サービスは実行されています。

   システムのログ出力が空であるか、EC2Launch v2 エージェントが実行されていない場合は、Instance Console Screenshot サービスを使用してインスタンスのトラブルシューティングを行います。詳細については「接続できないインスタンスのスクリーンショットの取得」を参照してください。

ステップ 2: ルートボリュームをインスタンスからデタッチします

パスワードの保存先のボリュームがルートボリュームとしてインスタンスにアタッチされている場合、EC2Launch v2 を使用して管理者パスワードをリセットすることはできません。一時インスタンスにセカンダリボリュームとしてアタッチする前に、元のインスタンスからボリュームをデタッチする必要があります。

ルートボリュームをインスタンスからデタッチするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. パスワードのリセットが必要なインスタンスを選択し、[インスタンスの状態]、[インスタンスを停止] を選択します。インスタンスのステータスが [停止] に変わったら、次のステップに進みます。
4. (オプション) このインスタンスの起動時に指定したプライベートキーがある場合は、次のステップに進みます。それ以外の場合は、次の手順を使用して、新しいキーペアで起動する新しいインスタンスでインスタンスを置き換えます。

    a. Amazon EC2 コンソールで、新しいキーペアを作成します。紛失したプライベートキーと同じ名前を新しいキーペアに指定するには、まず既存のキーペアを削除する必要があります。

    b. 置き換えるインスタンスを選択します。インスタンスのインスタンスタイプ、VPC、サブネット、セキュリティグループ、および IAM ロールを書き留めます。

    c. インスタンスを選択した状態で、[アクション]、[イメージとテンプレート]、[イメージの作成] の順に選択します。イメージの名前と説明を入力して、[イメージの作成] を選択します。

    d. ナビゲーションペインで [AMIs] を選択します。イメージのステータスが [使用可能] に変わるまで待ちます。そして、イメージを選択し、[AMI からインスタンスを起動] を選択します。

    e. インスタンスを起動するためのフィールドを完了し、置き換えるインスタンスと同じインスタンスタイプ、VPC、サブネット、セキュリティグループ、および IAM ロールを選択し、[インスタンスを起動] を選択します。

    f. プロンプトが表示されたら、新しいインスタンス用に作成したキーペアを選択し、[インスタンスを起動] を選択します。

g. (オプション) 元のインスタンスに Elastic IP アドレスが関連付けられていた場合は、それを新しいインスタンスに関連付けます。元のインスタンスにルートボリュームに加えて EBS ボリュームがある場合は、それらを新しいインスタンスに転送します。

5. 次のように、元のインスタンスからルートボリュームをデタッチします。

   a. 元のインスタンスを選択し、[ストレージ] タブを選択します。[ルートデバイス名] の下のルートデバイスの名前を書き留めます。[ブロックデバイス] の下でこのデバイス名のボリュームを探し、そのボリューム ID を書き留めます。

   b. ナビゲーションペインの [Volumes] を選択してください。

   c. ボリュームのリストで、ルートデバイスとして書き留めたボリュームを選択し、[アクション]、[ボリュームのデタッチ] を選択します。ボリュームのステータスが [利用可能] に変わったら、次のステップに進みます。

6. 元のインスタンスと置き換えるために新しいインスタンスを作成した場合は、元のインスタンスをすぐに終了することができます。元のインスタンスはもう必要ありません。この手順の残りの部分では、元のインスタンスへのすべてのリファレンスが、作成した新しいインスタンスに適用されます。

ステップ 3: ボリュームを一時インスタンスにアタッチします。

次に、一時インスタンスを起動し、ボリュームにセカンダリボリュームとして接続します。これは、設定ファイルを変更するために使用するインスタンスです。

一時インスタンスを起動してボリュームをアタッチするには

1. 次のように一時インスタンスを起動します。

   a. ナビゲーションペインで、[インスタンス]、[インスタンスを起動] の順に選択し、AMI を選択します。

   > **Important**
   >
   > ディスク署名の競合を回避するには、Windows 用の異なるバージョンの AMI を選択する必要があります。たとえば、元のインスタンスが Windows Server 2019 を実行している場合、Windows Server 2016 用の AMI を使用して一時インスタンスを起動します。

   b. デフォルトのインスタンスタイプのまま、[次: インスタンスの詳細の設定] を選択します。

c. [インスタンスの詳細の設定] ページの [サブネット] で、元のインスタンスと同じアベイラビリティーゾーンを選択し、[確認して起動] を選択します。

> **Important**
>
> 一時インスタンスは、元のインスタンスと同じアベイラビリティーゾーンで起動する必要があります。一時インスタンスが別のアベイラビリティーゾーンにある場合、元のインスタンスのルートボリュームをアタッチすることはできません。

d. [Review Instance Launch] ページで、[Launch] を選択します。

e. プロンプトが表示されたら新しいキーペアを作成し、コンピュータ上の安全な場所にダウンロードして、[インスタンスを起動] を選択します。

2. 次のように、ボリュームをセカンダリボリュームとして一時インスタンスにアタッチします。

a. ナビゲーションペインで [ボリューム] を選択し、元のインスタンスからデタッチしたボリュームを選択した後で、[アクション]、[ボリュームのアタッチ] の順に選択します。

b. [インスタンス] の [ボリュームのアタッチ] ダイアログボックスで、一時インスタンスの名前または ID の入力を開始し、リストからインスタンスを選択します。

c. [デバイス] で、**xvdf** (まだない場合) を入力し、[アタッチ] を選択します。

ステップ 4: .run-once ファイルを削除する

ここで、インスタンスにアタッチされたオフラインボリュームから `.run-once` ファイルを削除する必要があります。これにより、EC2Launch v2 は頻度を `once` とするすべてのタスク (管理者パスワードの設定を含む) を実行します。アタッチしたセカンダリボリュームでのファイルパスは、`D:\ProgramData\Amazon\EC2Launch\state\.run-once` と同様になります。

.run-once ファイルを削除するには

1. [ディスクの管理] ユーティリティを開き、「Amazon EBS ボリュームを使用できるようにする」の指示に従ってドライブをオンラインにします。
2. オンラインにしたディスク内の `.run-once` ファイルを探します。
3. .run-once ファイルを削除します。

> **Important**
>
> 一回実行するように設定されたスクリプトは、このアクションによってトリガーされます。

ステップ 5: 元のインスタンスを再起動します。

`.run-once` ファイルの削除後に、ボリュームをルートボリュームとして元のインスタンスに再アタッチし、そのキーペアを使用してインスタンスに接続して管理者パスワードを取得します。

1. 初期インスタンスにボリュームを再度アタッチします。
   a. ナビゲーションペインで [ボリューム] を選択し、一時インスタンスからデタッチしたボリュームを選択した後で、[アクション]、[ボリュームのアタッチ] の順に選択します。
   b. [インスタンス] の [ボリュームのアタッチ] ダイアログボックスで、元のインスタンスの名前または ID の入力し、インスタンスを選択します。
   c. [デバイス] で、**`/dev/sda1`** を入力します。
   d. [アタッチ] を選択します。ボリュームのステータスが `in-use` に変わったら、次のステップに進みます。
2. ナビゲーションペインで、[インスタンス] を選択します。元のインスタンスを選択し、[インスタンスの状態]、[インスタンスの開始] の順に選択します。インスタンスの状態が `Running` に変わったら、次のステップに進みます。
3. 新しいキーペアのプライベートキーを使用して、新しい Windows 管理者パスワードを取得し、インスタンスに接続します。詳細については「RDP を使用した Windows インスタンスへの接続」を参照してください。

   > **Important**
   >
   > インスタンスを停止して起動すると、新しいパブリック IP アドレスが取得されます。必ず、現在のパブリック DNS 名を使用してインスタンスに接続してください。詳細については「Amazon EC2 インスタンスの状態変更」を参照してください。

4. (オプション) 一時インスタンスをそれ以上使用しない場合は、終了できます。一時インスタンスを選択し、[インスタンスの状態]、[インスタンスの終了] の順に選択します。

EC2Launch を使用して EC2 インスタンスの Windows 管理者パスワードをリセットする

Windows 管理者パスワードを紛失した場合、Windows Server 2016 以降の AMI を使用している場合は、EC2Launch サービスを使用する EC2Rescue ツールを使用して、新しいパスワードを生成できます。

EC2Launch v2 エージェントを含まない Windows Server 2016 以降の AMI を使用している場合は、EC2Launch v2 を使用して新しいパスワードを生成できます。

Windows Server 2016 より前の Windows Server AMI を使用している場合は、「EC2Config を使用して EC2 インスタンスの Windows 管理者パスワードをリセットする」を参照してください。

Warning

インスタンスストアボリューム上のデータは、インスタンスを停止すると失われます。このデータを保持するには、永続ストレージにバックアップしてください。

Note

インスタンスでローカル管理者アカウントを無効にし、インスタンスが Systems Manager 用に設定されている場合は、EC2Rescue および Run Command を使用してローカル管理者パスワードを再度有効にしたり、リセットすることもできます。詳細については、「Systems Manager Run Command での EC2Rescue for Windows Server の使用」を参照してください。

Note

ローカル管理者パスワードのリセットに必要な手動の手順を自動的に適用する AWS Systems Manager のオートメーションドキュメントがあります。詳細については、「AWS Systems Manager ユーザーガイド」の「EC2 インスタンスでのパスワードと SSH キーのリセット」を参照してください。

EC2Launch を使用して Windows 管理者パスワードをリセットするには、次の操作が必要です。

- ステップ 1: ルートボリュームをインスタンスからデタッチします
- ステップ 2: ボリュームを一時インスタンスにアタッチします。
- ステップ 3: 管理者パスワードをリセットする
- ステップ 4: 元のインスタンスを再起動します。

ステップ 1: ルートボリュームをインスタンスからデタッチします

パスワードが保存されているボリュームが、ルートボリュームとしてインスタンスにアタッチされている場合、EC2Launch を使用して管理者パスワードをリセットすることはできません。一時インスタンスにセカンダリボリュームとしてアタッチする前に、元のインスタンスからボリュームをデタッチする必要があります。

ルートボリュームをインスタンスからデタッチするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. パスワードのリセットが必要なインスタンスを選択し、[インスタンスの状態]、[インスタンスを停止] を選択します。インスタンスのステータスが [停止] に変わったら、次のステップに進みます。
4. (オプション) このインスタンスの起動時に指定したプライベートキーがある場合は、次のステップに進みます。それ以外の場合は、次の手順を使用して、新しいキーペアで起動する新しいインスタンスでインスタンスを置き換えます。

    a. Amazon EC2 コンソールで、新しいキーペアを作成します。紛失したプライベートキーと同じ名前を新しいキーペアに指定するには、まず既存のキーペアを削除する必要があります。

    b. 置き換えるインスタンスを選択します。インスタンスのインスタンスタイプ、VPC、サブネット、セキュリティグループ、および IAM ロールを書き留めます。

    c. インスタンスを選択した状態で、[アクション]、[イメージとテンプレート]、[イメージの作成] の順に選択します。イメージの名前と説明を入力して、[イメージの作成] を選択します。

    d. ナビゲーションペインで [AMIs] を選択します。イメージのステータスが [使用可能] に変わるまで待ちます。そして、イメージを選択し、[AMI からインスタンスを起動] を選択します。

e. インスタンスを起動するためのフィールドを完了し、置き換えるインスタンスと同じインスタンスタイプ、VPC、サブネット、セキュリティグループ、および IAM ロールを選択し、[インスタンスを起動] を選択します。

f. プロンプトが表示されたら、新しいインスタンス用に作成したキーペアを選択し、[インスタンスを起動] を選択します。

g. (オプション) 元のインスタンスに Elastic IP アドレスが関連付けられていた場合は、それを新しいインスタンスに関連付けます。元のインスタンスにルートボリュームに加えて EBS ボリュームがある場合は、それらを新しいインスタンスに転送します。

5. 次のように、元のインスタンスからルートボリュームをデタッチします。

a. 元のインスタンスを選択し、[ストレージ] タブを選択します。[ルートデバイス名] の下のルートデバイスの名前を書き留めます。[ブロックデバイス] の下でこのデバイス名のボリュームを探し、そのボリューム ID を書き留めます。

b. ナビゲーションペインの [Volumes] を選択してください。

c. ボリュームのリストで、ルートデバイスとして書き留めたボリュームを選択し、[アクション]、[ボリュームのデタッチ] を選択します。ボリュームのステータスが [利用可能] に変わったら、次のステップに進みます。

6. 元のインスタンスと置き換えるために新しいインスタンスを作成した場合は、元のインスタンスをすぐに終了することができます。元のインスタンスはもう必要ありません。この手順の残りの部分では、元のインスタンスへのすべてのリファレンスが、作成した新しいインスタンスに適用されます。

ステップ 2: ボリュームを一時インスタンスにアタッチします。

次に、一時インスタンスを起動し、ボリュームにセカンダリボリュームとして接続します。これは EC2Launch の実行に使用するインスタンスです。

一時インスタンスを起動してボリュームをアタッチするには

1. 次のように一時インスタンスを起動します。

a. ナビゲーションペインで、[インスタンス]、[インスタンスを起動] の順に選択し、AMI を選択します。

> **Important**
>
> ディスク署名の競合を回避するには、Windows 用の異なるバージョンの AMI を選択する必要があります。たとえば、元のインスタンスが Windows Server 2019 を実行している場合、Windows Server 2016 用の AMI を使用して一時インスタンスを起動します。

b. デフォルトのインスタンスタイプのまま、[次: インスタンスの詳細の設定] を選択します。

c. [インスタンスの詳細の設定] ページの [サブネット] で、元のインスタンスと同じアベイラビリティーゾーンを選択し、[確認して起動] を選択します。

> **Important**
>
> 一時インスタンスは、元のインスタンスと同じアベイラビリティーゾーンで起動する必要があります。一時インスタンスが別のアベイラビリティーゾーンにある場合、元のインスタンスのルートボリュームをアタッチすることはできません。

d. [Review Instance Launch] ページで、[Launch] を選択します。

e. プロンプトが表示されたら新しいキーペアを作成し、コンピュータ上の安全な場所にダウンロードして、[インスタンスを起動] を選択します。

2. 次のように、ボリュームをセカンダリボリュームとして一時インスタンスにアタッチします。

a. ナビゲーションペインで [ボリューム] を選択し、元のインスタンスからデタッチしたボリュームを選択した後で、[アクション]、[ボリュームのアタッチ] の順に選択します。

b. [インスタンス] の [ボリュームのアタッチ] ダイアログボックスで、一時インスタンスの名前または ID の入力を開始し、リストからインスタンスを選択します。

c. [デバイス] で、**xvdf** (まだない場合) を入力し、[アタッチ] を選択します。

ステップ 3: 管理者パスワードをリセットする

次に、一時インスタンスに接続し、EC2Launch を使用して管理者パスワードをリセットします。

管理者パスワードをリセットするには

1. 次のように、一時インスタンスに接続し、そのインスタンスで EC2Rescue for Windows Server ツールを使用して管理者パスワードをリセットします。

a. EC2Rescue for Windows Server zip ファイルをダウンロードして、内容を展開し、[EC2Rescue.exe] を実行します。

b. [ライセンス契約] 画面で、使用許諾書をお読みの上、条件に同意する場合は、[同意します] を選択します。

c. [EC2Rescue for Windows Server へようこそ] 画面で、[次へ] を選択します。

d. [Select mode (モードの選択)] 画面で、[Offline instance (オフラインインスタンス)] を選択します。

e. [Select a disk (ディスクの選択)] 画面で、[xvdf] デバイスを選択して、[次へ] を選択します。

f. ディスクの選択を確認し、[Yes] を選択します。

g. ボリュームがロードされたら、[OK] を選択します。

h. [Select Offline Instance Option (オフラインインスタンスオプションの選択)] 画面で、[Diagnose and Rescue (診断とレスキュー)] を選択します。

i. [概要] 画面で情報を確認し、[次へ] を選択します。

j. [Detected possible issues (検出された潜在的な問題)] 画面で [Reset Administrator Password (管理者パスワードのリセット)] を選択し、[次へ] を選択します。

k. [確認] 画面で、[Rescue (レスキュー)] を選択して、[OK] を選択します。

l. [完了] 画面で、[終了] を選択します。

m. EC2Rescue for Windows Server ツールを閉じて、一時インスタンスから切断して、Amazon EC2 コンソールに戻ります。

2. 次のように、一時インスタンスからセカンダリ (xvdf) ボリュームをデタッチします。

a. ナビゲーションペインで、[インスタンス] を選択し、一時インスタンスを選択します。

b. 一時インスタンスの [ストレージ] タブで、xvdf として表示される EBS ボリュームの ID を書き留めます。

c. ナビゲーションペインの [Volumes] を選択します。

d. ボリュームのリストで、前のステップで記録したボリュームを選択し、[アクション]、[ボリュームのデタッチ] の順に選択します。ボリュームのステータスが [利用可能] に変わったら、次のステップに進みます。

ステップ 4: 元のインスタンスを再起動します。

EC2Launch を使用して管理者パスワードをリセットした後、元のインスタンスにボリュームをルートボリュームとして再アタッチし、そのキーペアを使用してインスタンスに接続して管理者パスワードを取得します。

元のインスタンスを再起動するには

1. 初期インスタンスにボリュームを再度アタッチします。
   a. ナビゲーションペインで [ボリューム] を選択し、一時インスタンスからデタッチしたボリュームを選択した後で、[アクション]、[ボリュームのアタッチ] の順に選択します。
   b. [インスタンス] の [ボリュームのアタッチ] ダイアログボックスで、元のインスタンスの名前または ID の入力し、インスタンスを選択します。
   c. [デバイス] で、**/dev/sda1** を入力します。
   d. [アタッチ] を選択します。ボリュームのステータスが `in-use` に変わったら、次のステップに進みます。
2. ナビゲーションペインで、[インスタンス] を選択します。元のインスタンスを選択し、[インスタンスの状態]、[インスタンスの開始] の順に選択します。インスタンスの状態が `Running` に変わったら、次のステップに進みます。
3. 新しいキーペアのプライベートキーを使用して、新しい Windows 管理者パスワードを取得し、インスタンスに接続します。詳細については「RDP を使用した Windows インスタンスへの接続」を参照してください。
4. (オプション) 一時インスタンスをそれ以上使用しない場合は、終了できます。一時インスタンスを選択し、[インスタンスの状態]、[インスタンスの終了] の順に選択します。

EC2Config を使用して EC2 インスタンスの Windows 管理者パスワードをリセットする

Windows 管理者パスワードを紛失した場合、Windows Server 2016 以前の Windows AMI を使用している場合は、EC2Config エージェントを使用して新しいパスワードを生成できます。

Windows Server 2016 以降の AMI を使用している場合は、「EC2Launch を使用して EC2 インスタンスの Windows 管理者パスワードをリセットする」を参照してください。また、EC2Launch サービスを使用する EC2Rescue ツールを使用して、新しいパスワードを生成できます。

> **Note**
>
> インスタンスでローカル管理者アカウントを無効にし、インスタンスが Systems Manager 用に設定されている場合は、EC2Rescue および Run Command を使用してローカル管理者パスワードを再度有効にしたり、リセットすることもできます。詳細については、「Systems Manager Run Command での EC2Rescue for Windows Server の使用」を参照してください。

> **Note**
>
> ローカル管理者パスワードのリセットに必要な手動の手順を自動的に適用する AWS Systems Manager のオートメーションドキュメントがあります。詳細については、「AWS Systems Manager ユーザーガイド」の「EC2 インスタンスでのパスワードと SSH キーのリセット」を参照してください。

EC2Config を使用して Windows 管理者パスワードをリセットするには、次の操作が必要です。

- ステップ 1: EC2Config サービスが実行中であることを確認します
- ステップ 2: ルートボリュームをインスタンスからデタッチします
- ステップ 3: ボリュームを一時インスタンスにアタッチします。
- ステップ 4: 設定ファイルを変更する
- ステップ 5: 元のインスタンスを再起動します。

ステップ 1: EC2Config サービスが実行中であることを確認します

管理者パスワードのリセットを試みる前に、EC2Config サービスがインストールされ、実行されていることを確認します。このセクションの後で、EC2Config サービスを使用して管理者パスワードをリセットします。

EC2Config サービスが実行中であることを確認するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [インスタンス] を選択し、パスワードのリセットが必要なインスタンスを選択します。この手順では、このインスタンスを元のインスタンスと呼びます。

3. [アクション]、[モニタリングとトラブルシューティング]、[システムログの取得] の順に選択します。
4. EC2 Agent エントリ (例: EC2 Agent: Ec2Config service v3.18.1118) を見つけます。このエントリが表示される場合、EC2Config サービスは実行中です。

   システムログ出力が空であるか、EC2Config サービスが実行されていない場合は、インスタンスコンソールスクリーンショットサービスを使用してインスタンスをトラブルシューティングします。詳細については「接続できないインスタンスのスクリーンショットの取得」を参照してください。

ステップ 2: ルートボリュームをインスタンスからデタッチします

パスワードが保存されているボリュームが、ルートボリュームとしてインスタンスにアタッチされている場合、EC2Config を使用して管理者パスワードをリセットすることはできません。一時インスタンスにセカンダリボリュームとしてアタッチする前に、元のインスタンスからボリュームをデタッチする必要があります。

ルートボリュームをインスタンスからデタッチするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. パスワードのリセットが必要なインスタンスを選択し、[インスタンスの状態]、[インスタンスを停止] を選択します。インスタンスのステータスが [停止] に変わったら、次のステップに進みます。
4. (オプション) このインスタンスの起動時に指定したプライベートキーがある場合は、次のステップに進みます。それ以外の場合は、次の手順を使用して、新しいキーペアで起動する新しいインスタンスでインスタンスを置き換えます。

   a. Amazon EC2 コンソールで、新しいキーペアを作成します。紛失したプライベートキーと同じ名前を新しいキーペアに指定するには、まず既存のキーペアを削除する必要があります。
   b. 置き換えるインスタンスを選択します。インスタンスのインスタンスタイプ、VPC、サブネット、セキュリティグループ、および IAM ロールを書き留めます。
   c. インスタンスを選択した状態で、[アクション]、[イメージとテンプレート]、[イメージの作成] の順に選択します。イメージの名前と説明を入力して、[イメージの作成] を選択します。

d. ナビゲーションペインで [AMIs] を選択します。イメージのステータスが [使用可能] に変わるまで待ちます。そして、イメージを選択し、[AMI からインスタンスを起動] を選択します。

e. インスタンスを起動するためのフィールドを完了し、置き換えるインスタンスと同じインスタンスタイプ、VPC、サブネット、セキュリティグループ、および IAM ロールを選択し、[インスタンスを起動] を選択します。

f. プロンプトが表示されたら、新しいインスタンス用に作成したキーペアを選択し、[インスタンスを起動] を選択します。

g. (オプション) 元のインスタンスに Elastic IP アドレスが関連付けられていた場合は、それを新しいインスタンスに関連付けます。元のインスタンスにルートボリュームに加えて EBS ボリュームがある場合は、それらを新しいインスタンスに転送します。

5. 次のように、元のインスタンスからルートボリュームをデタッチします。

a. 元のインスタンスを選択し、[ストレージ] タブを選択します。[ルートデバイス名] の下のルートデバイスの名前を書き留めます。[ブロックデバイス] の下でこのデバイス名のボリュームを探し、そのボリューム ID を書き留めます。

b. ナビゲーションペインの [Volumes] を選択してください。

c. ボリュームのリストで、ルートデバイスとして書き留めたボリュームを選択し、[アクション]、[ボリュームのデタッチ] を選択します。ボリュームのステータスが [利用可能] に変わったら、次のステップに進みます。

6. 元のインスタンスと置き換えるために新しいインスタンスを作成した場合は、元のインスタンスをすぐに終了することができます。元のインスタンスはもう必要ありません。この手順の残りの部分では、元のインスタンスへのすべてのリファレンスが、作成した新しいインスタンスに適用されます。

ステップ 3: ボリュームを一時インスタンスにアタッチします。

次に、一時インスタンスを起動し、ボリュームにセカンダリボリュームとして接続します。これは、設定ファイルを変更するために使用するインスタンスです。

一時インスタンスを起動してボリュームをアタッチするには

1. 次のように一時インスタンスを起動します。

a. ナビゲーションペインで、[インスタンス]、[インスタンスを起動] の順に選択し、AMI を選択します。

> **Important**
>
> ディスク署名の競合を回避するには、Windows 用の異なるバージョンの AMI を選択する必要があります。たとえば、元のインスタンスが Windows Server 2019 を実行している場合、Windows Server 2016 用の AMI を使用して一時インスタンスを起動します。

b. デフォルトのインスタンスタイプのまま、[次: インスタンスの詳細の設定] を選択します。

c. [インスタンスの詳細の設定] ページの [サブネット] で、元のインスタンスと同じアベイラビリティーゾーンを選択し、[確認して起動] を選択します。

> **Important**
>
> 一時インスタンスは、元のインスタンスと同じアベイラビリティーゾーンで起動する必要があります。一時インスタンスが別のアベイラビリティーゾーンにある場合、元のインスタンスのルートボリュームをアタッチすることはできません。

d. [Review Instance Launch] ページで、[Launch] を選択します。

e. プロンプトが表示されたら新しいキーペアを作成し、コンピュータ上の安全な場所にダウンロードして、[インスタンスを起動] を選択します。

2. 次のように、ボリュームをセカンダリボリュームとして一時インスタンスにアタッチします。

a. ナビゲーションペインで [ボリューム] を選択し、元のインスタンスからデタッチしたボリュームを選択した後で、[アクション]、[ボリュームのアタッチ] の順に選択します。

b. [インスタンス] の [ボリュームのアタッチ] ダイアログボックスで、一時インスタンスの名前または ID の入力を開始し、リストからインスタンスを選択します。

c. [デバイス] で、**xvdf** (まだない場合) を入力し、[アタッチ] を選択します。

ステップ 4: 設定ファイルを変更する

一時インスタンスにボリュームをセカンダリボリュームとして添付したら、設定ファイルの `Ec2SetPassword` プラグインを変更します。

設定ファイルを変更するには

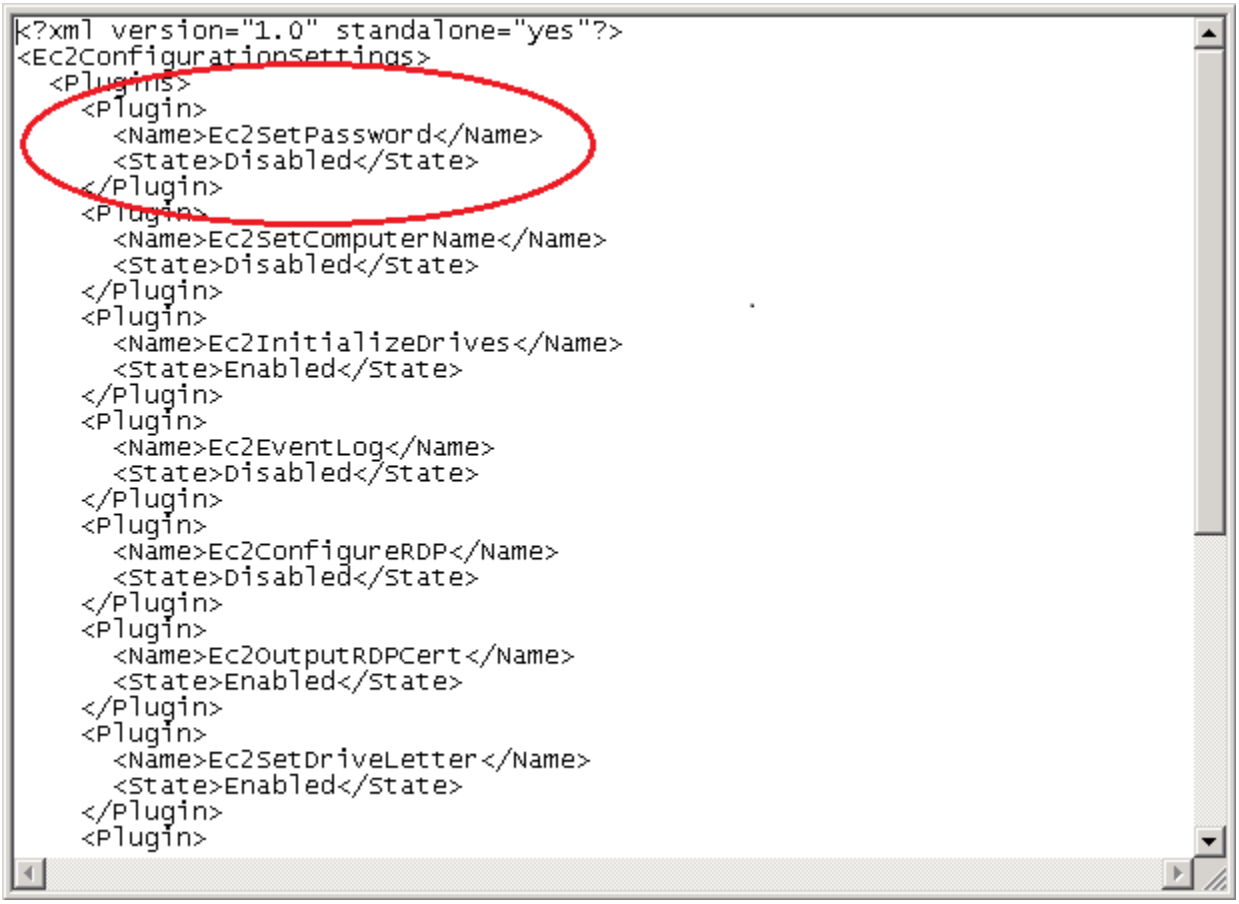
1. 一時インスタンスから、次のようにセカンダリボリュームの設定ファイルを変更します。

a. 一時インスタンスを起動して接続します。

b. ドライブをオンラインにするには、以下の手順に従います。Amazon EBS ボリュームを使用できるようにします。

c. セカンダリボリュームに移動し、メモ帳などのテキストエディタを使用して \Program Files\Amazon\Ec2ConfigService\Settings\config.xml を開きます。

d. ファイルの先頭で、スクリーンショットに示すような Ec2SetPassword という名前のプラグインを見つけます。状態を Disabled から Enabled に変更して、ファイルを保存します。



2. 設定ファイルを変更した後、次のように、一時インスタンスからセカンダリボリュームをデタッチします。

a. [ディスク管理] ユーティリティを使用して、ボリュームをオフラインにします。

b. 一時インスタンスから切断し、Amazon EC2 コンソールに戻ります。

c. ナビゲーションペインで、[ボリューム] を選択してボリュームを選択し、[アクション]、[ボリュームのデタッチ] の順に選択します。ボリュームのステータスが [available] に変わったら、次のステップに進みます。

ステップ 5: 元のインスタンスを再起動します。

設定ファイルを変更した後、元のインスタンスにボリュームをルートボリュームとして再アタッチし、そのキーペアを使用してインスタンスに接続して管理者パスワードを取得します。

1. 初期インスタンスにボリュームを再度アタッチします。
   a. ナビゲーションペインで [ボリューム] を選択し、一時インスタンスからデタッチしたボリュームを選択した後で、[アクション]、[ボリュームのアタッチ] の順に選択します。
   b. [インスタンス] の [ボリュームのアタッチ] ダイアログボックスで、元のインスタンスの名前または ID の入力し、インスタンスを選択します。
   c. [デバイス] で、**/dev/sda1** を入力します。
   d. [アタッチ] を選択します。ボリュームのステータスが `in-use` に変わったら、次のステップに進みます。
2. ナビゲーションペインで、[インスタンス] を選択します。元のインスタンスを選択し、[インスタンスの状態]、[インスタンスの開始] の順に選択します。インスタンスの状態が `Running` に変わったら、次のステップに進みます。
3. 新しいキーペアのプライベートキーを使用して、新しい Windows 管理者パスワードを取得し、インスタンスに接続します。詳細については「RDP を使用した Windows インスタンスへの接続」を参照してください。

   > **Important**
   >
   > インスタンスを停止して起動すると、新しいパブリック IP アドレスが取得されます。必ず、現在のパブリック DNS 名を使用してインスタンスに接続してください。詳細については「Amazon EC2 インスタンスの状態変更」を参照してください。

4. (オプション) 一時インスタンスをそれ以上使用しない場合は、終了できます。一時インスタンスを選択し、[インスタンスの状態]、[インスタンスの終了] の順に選択します。

Amazon EC2 Windows インスタンスの Sysprep の問題をトラブルシューティングする

イメージ準備中に問題が発生したり、エラーメッセージを受け取った場合、次のログを確認します。ログの場所はSysprep で EC2Config または EC2Launch v1、あるいは EC2Launch v2のいずれを実行しているかにより異なります。

- %WINDIR%\Panther\Unattendgc (EC2Config、EC2Launch v1、および EC2Launch v2)
- %WINDIR%\System32\Sysprep\Panther (EC2Config、EC2Launch v1、および EC2Launch v2)
- C:\Program Files\Amazon\Ec2ConfigService\Logs\Ec2ConfigLog.txt (EC2Config のみ)
- C:\ProgramData\Amazon\Ec2Config\Logs (EC2Config のみ)
- C:\ProgramData\Amazon\EC2-Windows\Launch\Log\EC2Launch.log (EC2Launch v1 のみ)
- %ProgramData%\Amazon\EC2Launch\log\agent.log (EC2Launch v2 のみ)

Sysprep でイメージ準備中にエラーメッセージを受け取った場合、OS にアクセスできないことがあります。ログファイルを確認するにはインスタンスを停止し、別の正常なインスタンスにルートボリュームをセカンダリボリュームとしてアタッチして、上記のログをセカンダリボリュームで確認します。ログファイルの名前別の詳しい用途についてはMicrosoft ドキュメントの「Windows セットアップ関連のログファイル」を参照してください。

Unattendgc のログファイルでエラーを見つけた場合はMicrosoft Error Lookup Tool を使用してエラーの詳細にアクセスします。Unattendgc のログファイルでレポートされた次の事項ではインスタンス内の 1 つ以上の破損しているユーザープロフィールがよくある原因です。

```
Error [Shell Unattend] _FindLatestProfile failed (0x80070003) [gle=0x00000003]
Error [Shell Unattend] CopyProfile failed (0x80070003) [gle=0x00000003]
```

この問題を解決するためには 2 つのオプションがあります。

オプション 1

インスタンスで Regedit を使用して、次のキーを検索します。削除されたユーザーのプロファイルレジストリキーがないことを確認します。

```
[HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion
\ProfileList\
```

オプション 2

1. 関連ファイルを以下のように編集します。
   - Windows Server 2012 R2 以前 – EC2Config 応答ファイル (C:\Program Files\Amazon\Ec2ConfigService\sysprep2008.xml) を編集します。

- Windows Server 2016 および 2019 – unattend.xml 応答ファイル (`C:\ProgramData\Amazon\EC2-Windows\Launch\Sysprep\Unattend.xml`) を編集します。
- Windows Server 2022 – unattend.xml 応答ファイル (`C:\ProgramData\Amazon\EC2Launch\sysprep\unattend.xml`) を編集します。

2. `<CopyProfile>true</CopyProfile>` を `<CopyProfile>false</CopyProfile>` に変更します。
3. Sysprep を再度実行します。この設定の変更はSysprep が完了した後で、組み込まれた管理者ユーザープロフィールを削除することに注意してください。

EC2Rescue を使用して問題のある Amazon EC2 Linux インスタンスのトラブルシューティングを行う

Linux 用 EC2Rescue は、使いやすいオープンソースのツールであり、Amazon EC2 Linux インスタンスで実行し、100 を超えるモジュールのライブラリを使用して一般的な問題を診断、トラブルシューティング、修正できます。モジュールは、BASH または Python スクリプト、および必要なメタデータを含む YAML ファイルです。

Linux インスタンス用 EC2Rescue の一般的な使用例には、次のようなものがあります。

- syslog ログとパッケージマネージャーログの収集
- リソース使用率データの収集
- 既知の問題のあるカーネルパラメータと一般的な OpenSSH の問題の診断と修復

> **Note**
>
> `AWSSupport-TroubleshootSSH` AWS Systems Manager Automation ランブックは Linux 用 EC2Rescue をインストールし、そのツールを使用して、Linux インスタンスへの SSH 接続を妨げる、一般的な問題を確認しその修正を試みます。詳細については、「AWSSupport-TroubleshootSSH」を参照してください。

Windows インスタンスを使用している場合は、「the section called “EC2Rescue for Windows インスタンス”」を参照してください。

トピック

- Amazon EC2 Linux インスタンスに EC2Rescue をインストールする
- Amazon EC2 Linux インスタンスで EC2Rescue コマンドを実行する
- Amazon EC2 Linux インスタンス用の EC2Rescue モジュールを開発する

Amazon EC2 Linux インスタンスに EC2Rescue をインストールする

Linux 用 EC2Rescue ツールは、次の前提要件を満たす Amazon EC2 Linux インスタンスにインストールできます。

前提条件

- サポートされるオペレーティングシステム
  - Amazon Linux 2
  - Amazon Linux 2016.09+
  - SUSE Linux Enterprise Server 12+
  - RHEL 7+
  - Ubuntu 16.04+
- ソフトウェア要件
  - Python 2.7.9+ または 3.2+

EC2Rescue をインストールする

`AWSSupport-TroubleshootSSH` ランブックは Linux 用 EC2Rescue をインストールし、そのツールを使用して、Linux マシンへの SSH 経由でのリモートからの接続を妨げる、一般的な問題を確認しその修正を試みます。より詳しい情報を確認し、この自動化を実行するには、サポート-TroubleshootSSH をご参照ください。

システムに必要な Python バージョンがある場合は、標準ビルドをインストールできます。それ以外の場合は、Python の最小のコピーを含むバンドル済みのビルドをインストールできます。

標準ビルドをインストールするには

1. 作動している Linux インスタンスから、Linux 用 EC2Rescue ツールをダウンロードします。

```
curl -O https://s3.amazonaws.com/ec2rescuelinux/ec2rl.tgz
```

2. (オプション) Linux 用 EC2Rescue のインストールファイルの署名を検証します。詳細については、「(オプション) Linux 用 EC2Rescue の署名を検証する」を参照してください。
3. sha256 ハッシュファイルをダウンロードします。

```
curl -O https://s3.amazonaws.com/ec2rescuelinux/ec2rl.tgz.sha256
```

4. tarball の整合性を確認します。

```
sha256sum -c ec2rl.tgz.sha256
```

5. Tarball を解凍します。

```
tar -xzvf ec2rl.tgz
```

6. ヘルプファイルを表示してインストールを検証します。

```
cd ec2rl-<version_number>
./ec2rl help
```

バンドル済みのビルドをインストールするには

ダウンロードのリンクと制限については、github の「Linux 用 EC2Rescue」を参照してください。

(オプション) Linux 用 EC2Rescue の署名を検証する

以下に、Linux ベースのオペレーティングシステム用の Linux 用 EC2Rescue パッケージの有効性を検証するための推奨されるプロセスを示します。

インターネットからアプリケーションをダウンロードする場合は、ソフトウェア発行元のアイデンティティを認証し、アプリケーションの発行後に改ざん、あるいは破損がないか確認することをお勧めします。これにより、ウイルスやマルウェアに感染したバージョンのアプリケーションをインストールせずに済みます。

このトピックのステップを実行した後に Linux 用 EC2Rescue のソフトウェアが変更または破損していることが判明した場合は、インストールファイルを実行しないでください。このような場合は Amazon Web Services にご連絡ください。

Linux ベースのオペレーティングシステム用の Linux 用 EC2Rescue ファイルの署名には、GnuPG が使用されています。これは安全なデジタル署名のための、オープンソース実装のプリティグッドプ

ライバシー (OpenPGP) 標準です。GnuPG (GPG とも呼ばれます) では、デジタル署名を通じて認証と整合性のチェックが行われます。ダウンロードした Linux 用 EC2Rescue パッケージの検証に使用できるパブリックキーと署名は、AWS により公開されています。PGP と GnuPG (GPG) の詳細については、「https://www.gnupg.org/」を参照してください。

まず、ソフトウェア発行元との信頼を確立します。ソフトウェア発行元のパブリックキーをダウンロードし、キー所有者が一致していることを確認してから、キーリングに追加します。キーリングとは、既知のパブリックキーの集合です。真正性が確立されたパブリック キーは、アプリケーションの署名を確認するために使用できます。

タスク

- パブリック キーを認証およびインポートする
- パッケージの署名の確認

パブリック キーを認証およびインポートする

次の手順では、Linux 用 EC2Rescue のパブリックキーを認証し、信頼されたキーとして GPG キーリングへ追加します。

Linux 用 EC2Rescue のパブリックキーを認証してインポートするには

1. コマンドプロンプトで、次のコマンドを使用して当社のパブリック GPG ビルドキーのコピーを取得します。

   ```
   curl -O https://s3.amazonaws.com/ec2rescuelinux/ec2rl.key
   ```

2. ec2rl.key を保存したディレクトリのコマンドプロンプトで、次のコマンドを使用して Linux 用 EC2Rescue のパブリックキーをキーリングにインポートします。

   ```
   gpg2 --import ec2rl.key
   ```

   コマンドで次のような結果が返されます。

   ```
   gpg: /home/ec2-user/.gnupg/trustdb.gpg: trustdb created
   gpg: key 2FAE2A1C: public key "ec2autodiag@amazon.com <EC2 Rescue for Linux>"
    imported
   gpg: Total number processed: 1
   gpg:               imported: 1  (RSA: 1)
   ```

> **Tip**
>
> コマンドが見つからないというエラーが表示された場合は、`apt-get install gnupg2` (Debian ベースの Linux) または `yum install gnupg2` (Red Hat ベースの Linux) を使用して GnuPG ユーティリティをインストールします。

パッケージの署名の確認

GPG ツールをインストール後、Linux 用 EC2Rescue パブリックキーを認証してインポートし、Linux 用 EC2Rescue パブリック キーが信頼済みであることを確認すると、Linux 用 EC2Rescue インストールスクリプトの署名を確認できるようになります。

Linux 用 EC2Rescue インストールスクリプトの署名を確認するには

1. コマンド プロンプトで次のコマンドを実行し、インストール スクリプトの署名ファイルをダウンロードします。

   ```
   curl -O https://s3.amazonaws.com/ec2rescuelinux/ec2rl.tgz.sig
   ```

2. `ec2rl.tgz.sig` と Linux 用 EC2Rescue インストールファイルを保存したディレクトリのコマンドプロンプトで次のコマンドを実行し、署名を確認します。ファイルが2つとも存在している必要があります。

   ```
   gpg2 --verify ./ec2rl.tgz.sig
   ```

   出力は次のようになります。

   ```
   gpg: Signature made Thu 12 Jul 2018 01:57:51 AM UTC using RSA key ID 6991ED45
   gpg: Good signature from "ec2autodiag@amazon.com <EC2 Rescue for Linux>"
   gpg: WARNING: This key is not certified with a trusted signature!
   gpg:          There is no indication that the signature belongs to the owner.
   Primary key fingerprint: E528 BCC9 0DBF 5AFA 0F6C  C36A F780 4843 2FAE 2A1C
        Subkey fingerprint: 966B 0D27 85E9 AEEC 1146  7A9D 8851 1153 6991 ED45
   ```

   出力に「`Good signature from "ec2autodiag@amazon.com <EC2 Rescue for Linux>"`」という句が含まれる場合は、署名が正常に確認されており、Linux 用 EC2Rescue のインストールスクリプトを実行できることを意味しています。

出力結果に「`BAD signature`」という句が含まれる場合、手順が正しいことをもう一度確認してください。この応答が続く場合は、Amazon Web Services に連絡してください。以前にダウンロードしたインストール ファイルを実行しないでください。

以下は、表示される可能性のある警告の詳細です。

- WARNING: This key is not certified with a trusted signature! There is no indication that the signature belongs to the owner. これは、Linux 用 EC2Rescue の認証済みパブリック キーを所有していると考えるユーザーの個人レベルの信頼を参照します。本来は、ユーザーが Amazon Web Services オフィスを訪問してキーを受け取ることが理想的です。しかし、キーは多くの場合 ウェブ サイトからダウンロードされます。この場合、ウェブサイトは Amazon Web Services ウェブサイトです。
- gpg2: no ultimately trusted keys found. これは、特定のキーがユーザー (またはユーザーが信頼する他のユーザー) によって「最終的に信頼された」キーでないことを意味します。

詳細については、「https://www.gnupg.org/」を参照してください。

Amazon EC2 Linux インスタンスで EC2Rescue コマンドを実行する

EC2Rescue はコマンドラインツールです。Linux インスタンスに EC2Rescue をインストールすると、`./ec2rl help` を実行してツールの使用方法に関する一般的なヘルプを得ることができます。`./ec2rl list` を実行して使用可能なモジュールを表示し、`./ec2rl help` *`module_name`* を実行して特定のモジュールに関するヘルプを取得できます。

ここでは、このツールを使い始めるために実行できる一般的なタスクについて説明します。

タスク

- EC2Rescue モジュールを実行する
- EC2Rescue モジュールの結果をアップロードする
- Amazon EC2 Linux インスタンスのバックアップを作成する

EC2Rescue モジュールを実行する

すべての EC2Rescue モジュールを実行するには

追加のパラメータを指定せずに、./ec2rl run コマンドを使用します。一部のモジュールには、ルートアクセスが必要です。ルートユーザーでない場合は、コマンドを実行するときに sudo を使用します。

```
./ec2rl run
```

特定の EC2Rescue モジュールを実行するには

./ec2rl run コマンドを使用し、--only-modules には実行するモジュールの名前を指定します。一部のモジュールでは、それらを使用するための引数が必要です。

```
./ec2rl run --only-modules=module_name --arguments
```

例えば、dig モジュールを実行して amazon.com ドメインをクエリするには、次のコマンドを使用します。

```
./ec2rl run --only-modules=dig --domain=amazon.com
```

EC2Rescue モジュールの結果を表示するには

モジュールを実行し、cat /var/tmp/ec2rl/*logfile_location* のログファイルを表示します。例えば、dig モジュールのログファイルは次の場所にあります。

```
cat /var/tmp/ec2rl/timestamp/mod_out/run/dig.log
```

EC2Rescue モジュールの結果をアップロードする

サポート が EC2Rescue モジュールの結果をリクエストした場合は、EC2Rescue ツールを使用してログファイルをアップロードできます。結果は、サポート が提供する場所にアップロードすることも、所有している Amazon S3 バケットにアップロードすることもできます。

サポート から提供された場所に結果をアップロードするには

./ec2rl upload コマンドを使用します。--upload-directory にはログファイルの場所を指定します。--support-url には、サポート によって提供される URL を指定します。

```
./ec2rl upload --upload-directory=/var/tmp/ec2rl/logfile_location --support-
url="url_provided_by_aws_support"
```

Amazon S3 バケットに結果をアップロードするには

./ec2rl upload コマンドを使用します。--upload-directory には ログファイルの場所を指定します。--presigned-url には、S3 バケットの署名付き URL を指定します。Amazon S3 に署名付きの URL を生成するための詳細については、「署名付き URL を使用したオブジェクトのアップロード」を参照してください。

```
./ec2rl upload --upload-directory=/var/tmp/ec2rl/logfile_location --presigned-
url="presigned_s3_url"
```

Amazon EC2 Linux インスタンスのバックアップを作成する

EC2Rescue を使用して Linux インスタンスをバックアップするには、AMI を作成するか、アタッチされたボリュームのスナップショットを作成します。

AMI を作成するには

./ec2rl run コマンドを使用し、--backup には ami を指定します。

```
./ec2rl run --backup=ami
```

アタッチされたすべてのボリュームのマルチボリュームスナップショットを作成するには

./ec2rl run コマンドを使用し、--backup には allvolumes を指定します。

```
./ec2rl run --backup=allvolumes
```

特定のアタッチされたボリュームのスナップショットを作成するには

./ec2rl run コマンドを使用し、--backup にはバックアップするボリュームの ID を指定します。

```
./ec2rl run --backup=vol-01234567890abcdef
```

Amazon EC2 Linux インスタンス用の EC2Rescue モジュールを開発する

モジュールは、データシリアル化スタンダードである YAML デ書き込まれます。モジュールの YAML ファイルは、モジュールとその属性を示す単一のドキュメントで構成されます。

モジュール属性の追加

次の表には、利用できるモジュールの属性が一覧表示されます。

属性	説明
name	モジュールの名前。この名前は、長さが 18 文字以下である必要があります。
version	モジュールのバージョン番号。
タイトル	モジュールの短い説明タイトルです。この値は、長さが 50 文字以下である必要があります。

helptext	モジュールの拡張された説明。各列は、長さが 75 文字以下である必要があります。必須あるいはオプションでモジュールが引数を消費する場合、helptext 値にこの引数を含めます。
	
	次に例を示します。
	
	`helptext: !!str |`
	`  Collect output from ps for system analysis`
	`  Consumes --times= for number of times to repeat`
	`  Consumes --period= for time period between repetition`

placement	モジュールが実行されるべきステージ。サポートされる値。 • prediagnostic • run • postdiagnostic
language	モジュールコードが書き込まれている言語。サポートされる値。 • bash • python

属性	説明
	Note Python コードは、Python 2.7.9+ および Python 3.2+ の両方と互換性がある必要があります。
修復	モジュールが修復をサポートするかどうかを示します。サポートされている値は `True` または `False` です。 この値がない場合、モジュールのデフォルトは `False` です。修復をサポートしないそれらのモジュールのオプション属性となります。
コンテンツ	全スクリプトコード。
制約	制約値を含むオブジェクトの名前。
ドメイン	モジュールがどのようにグループ化または分類されているかの説明。含まれているモジュール一連は次のドメインを使用します。 • 同時接続の • net • os • パフォーマンス
class	モジュールによって実行されるタスクの種類の説明。含まれているモジュール一連は次のクラスを使用します。 • 回収 (プログラムからの出力を回収します) • 診断 (一連の基準の達成/未達成) • 収集 (ファイルのコピーと特定のファイルへの書き込み)

属性	説明
distro	このモジュールがサポートする Linux ディストリビューションの一覧。含まれているモジュール一連は次のディストリビューションを使用します。 • alami (Amazon Linux) • rhel • Ubuntu • suse
必須	CLI オプションからモジュールが消費する必要な引数。
optional	モジュールが使用できるオプションの引数。
ソフトウェア	モジュールで使用される実行可能なソフトウェア。この属性は、デフォルトでインストールされないソフトウェアの特定を行います。Linux 用 EC2Rescue ロジックは、モジュールを実行する前に、このプログラムが存在し、実行可能であることを確認します。
package	実行ファイル用のソースソフトウェアパッケージ。この属性は、ソフトウェアのパッケージにダウンロード用 URL やそのほかの詳細などの詳しい情報を提供するためのものです。
sudo	ルートアクセスがモジュールの実行に必要であるかどうかを示します。 モジュールスクリプトで sudo チェックを行う必要はありません。値が true になると、Linux 用 EC2Rescue ロジックは実行しているユーザーがルートアクセスを所持している場合にのみモジュールを実行します。

属性	説明
perfimpact	モジュールが実行している環境に重要な影響を及ぼす可能性があるかどうかを示します。値が true であり、`--perfimpact=true` 引数が存在しない場合、モジュールはスキップされます。
parallelexclusive	相互占有を必要とするプログラムを特定します。例えば、「bpf」を指定するすべてのモジュールはシリアル方法で実行します。

環境変数の追加

次の表には、利用できるモジュールの属性が一覧表示されます。

環境変数	説明
`EC2RL_CALLPATH`	`ec2rl.py` へのパス。このパスを使用すると、lib ディレクトリを見つけて、ベンダーの Python モジュールを使用できます。
`EC2RL_WORKDIR`	診断ツールの主要な tmp ディレクトリ。 デフォルト値: `/var/tmp/ec2rl` 。
`EC2RL_RUNDIR`	すべての出力が保存されているディレクトリ。 デフォルト値: `/var/tmp/ec2rl/<date&timestamp>` 。
`EC2RL_GATHEREDDIR`	収集されたモジュールデータを配置するルートディレクトリ。 デフォルト値:`/var/tmp/ec2rl/<date&timestamp>/mod_out/gathered/` 。

環境変数	説明
EC2RL_NET_DRIVER	初めて使用されるドライバーが、インスタンスの非仮想ネットワークインターフェースでアルファベット順に順序付けされます。 例: • xen_netfront • ixgbevf • ena
EC2RL_SUDO	Linux 用 EC2Rescue がルートとして実行されている場合には true、そうでない場合には false。
EC2RL_VIRT_TYPE	インスタンスメタデータから提供される仮想化タイプ。 例: • default-hvm • default-paravirtual
EC2RL_INTERFACES	システム上のインターフェースの列挙一覧。この値は、eth0 や eth1 などの名前が含まれる文字列です。これは functions.bash を介して生成され、これをソースとするモジュールのみで利用できます。

YAML 構文の使用

モジュール YAML ファイルを構築する際、以下に注意してください。

- 3 つのハイフン (---) は、ドキュメントの明示的な開始を示します。
- !ec2rlcore.module.Module タグは、データストリームからオブジェクトを作成する際にどのコンストラクタを呼び出すかを YAML パーサーに伝えます。module.py ファイル内コンストラクタを検索できます。

- !!str タグは、データの種類を決定する試行を行なわず、代わりにコンテンツを文字列リテラルとして解釈するように YAML パーサーに伝えます。
- パイプ文字 (|) は、値がリテラル形式のスカラーであることを YAML パーサーに伝えます。この場合、パーサーにはすべての空白が含まれます。インデントと改行文字が保持されるため、これはモジュールにとって重要です。
- YAML スタンダードインデントは 2 つのスペースとなり、次の例で示されます。スクリプトでスタンダードインデント (例えば、Python では 4 つの空白) を維持して、モジュールファイル内で全コンテンツを 2 つのスペースでインデントすることを確認します。

モジュールの例

例 1 (mod.d/ps.yaml):

```
--- !ec2rlcore.module.Module
# Module document. Translates directly into an almost-complete Module object
name: !!str ps
path: !!str
version: !!str 1.0
title: !!str Collect output from ps for system analysis
helptext: !!str |
  Collect output from ps for system analysis
  Requires --times= for number of times to repeat
  Requires --period= for time period between repetition
placement: !!str run
package:
  - !!str
language: !!str bash
content: !!str |
  #!/bin/bash
  error_trap()
  {
      printf "%0.s=" {1..80}
      echo -e "\nERROR: "$BASH_COMMAND" exited with an error on line ${BASH_LINENO[0]}"
      exit 0
  }
  trap error_trap ERR

  # read-in shared function
  source functions.bash
  echo "I will collect ps output from this $EC2RL_DISTRO box for $times times every
 $period seconds."
```

```
  for i in $(seq 1 $times); do
      ps auxww
      sleep $period
  done
constraint:
  requires_ec2: !!str False
  domain: !!str performance
  class: !!str collect
  distro: !!str alami ubuntu rhel suse
  required: !!str period times
  optional: !!str
  software: !!str
  sudo: !!str False
  perfimpact: !!str False
  parallelexclusive: !!str
```

EC2Rescue を使用して問題のある Amazon EC2 Windows インスタンスのトラブルシューティングを行う

EC2Rescue for Windows Server は、Amazon EC2 Windows Server インスタンス上で動作し、潜在的な問題の診断とトラブルシューティングを行うことができる使いやすいツールです。ログファイルを収集して問題を解決するだけでなく、問題がありそうな部分をプロアクティブに検索することができ、便利です。他のインスタンスから Amazon EBS ルートボリュームを調べて、そのボリュームを使用する Windows Server インスタンスをトラブルシューティングするために必要なログを収集することもできます。EC2Rescue が対処できる一般的な問題は次のとおりです。

- ファイアウォール、リモートデスクトッププロトコル (RDP)、またはネットワークインターフェイスの設定が原因で発生したインスタンス接続の問題
- 停止エラー、起動ループ、またはレジストリの破損によって生じたオペレーティングシステムの起動に関する問題
- 高度なログ分析とトラブルシューティングが必要な問題

EC2Rescue for Windows Server には以下の 2 つのモジュールがあります。

- さまざまなソースからデータを収集するデータコレクターモジュール
- 収集されたデータを一連の事前定義ルールと照合して解析し、問題を特定し、解決方法を提案するアナライザーモジュール

EC2Rescue for Windows Server ツールは、Windows Server 2012 以降を実行している Amazon EC2 インスタンスでのみ実行されます。ツールを起動すると、ツールが Amazon EC2 インスタンスで実行されているかどうかが確認されます。

> **Note**
>
> `AWSSupport-ExecuteEC2Rescue` AWS Systems Manager Automation ランブックでは、EC2Rescue ツールを使用してトラブルシューティングを行い、可能な場合は、特定の EC2 インスタンスの一般的な接続の問題を修復します。より詳しい情報を確認し、このオートメーションを実行するには、「AWSSupport-ExecuteEC2Rescue」をご覧ください。

Linux インスタンスを使用している場合は、「the section called “Linux インスタンス用 EC2Rescue”」を参照してください。

トピック

- EC2Rescue GUI を使用して問題のある Windows インスタンスのトラブルシューティングを行う
- EC2Rescue CLI を使用して問題のある Windows インスタンスのトラブルシューティングを行う
- EC2Rescue と Systems Manager を使用して問題のある Windows インスタンスのトラブルシューティングを行う

EC2Rescue GUI を使用して問題のある Windows インスタンスのトラブルシューティングを行う

EC2Rescue for Windows Server は、オフラインインスタンスで次の分析を実行できます。

オプション	説明
診断とレスキュー	EC2Rescue for Windows Server は、次のサービス設定を使用して、問題を検出し、対応できます。 • システム時刻 • [RealTimeisUniversal] - `RealTimeisUniversal` レジストリキーが有効かどうかを検出します。無効の場合、タイム

オプション	説明

	ゾーンが UTC 以外の値に設定されていると、Windows システム時刻が狂います。
	
	• Windows ファイアウォール
	  • [ドメインネットワーク] - この Windows ファイアウォールプロファイルが有効であるか無効であるかを検出します。
	  • [プライベートネットワーク] - この Windows ファイアウォールプロファイルが有効であるか無効であるかを検出します。
	  • [ゲストまたはパブリックネットワーク] - この Windows ファイアウォールプロファイルが有効であるか無効であるかを検出します。
	
	• リモートデスクトップ
	  • [サービスの開始] - リモートデスクトップサービスが有効かどうかを検出します。
	  • [リモートデスクトップ接続] - これが有効かどうかを検出します。
	  • [TCP ポート] - リモートデスクトップサービスがリッスンしているポートを検出します。
	
	• EC2Config (Windows Server 2012 R2 以前)
	  • [インストール] - インストールされている EC2Config バージョンを検出します。
	  • [サービスの開始] - EC2Config サービスが有効かどうかを検出します。
	  • [Ec2SetPassword] - 新しい管理者パスワードを生成します。

オプション	説明

	• [Ec2HandleUserData] - インスタンスの次回の起動時にユーザーデータスクリプトを実行できます。
	
	• EC2Launch (Windows Server 2016 以降)
	  • [インストール] - インストールされている EC2Launch バージョンを検出します。
	  • [Ec2SetPassword] - 新しい管理者パスワードを生成します。
	
	• ネットワークインターフェイス
	  • [DHCP サービススタートアップ] - DHCP サービスが有効かどうかを検出します。
	  • [イーサネットの詳細] - 検出された場合、ネットワークドライバーバージョンに関する情報を表示します。
	  • [イーサネットでの DHCP] - DHCP が有効かどうかを検出します。
	• ディスク署名のステータス
	  • [Signature on disk] (ディスク上の署名) および [Signature on Boot Configuration Database (BCD)] (ブート構成データベース (BCD) の署名) - ディスク署名と BCD 署名が同じかどうかを検出します。値が異なる場合、EC2Rescue は BCD の署名でディスク署名を上書きしようとします。

オプション	説明
Restore	次のいずれかのアクションを実行します。 • [最後に既知の良好な設定] - 最後に既知のブート可能状態でインスタンスの起動を試みます。 • [バックアップからレジストリを復元] - `\Windows\System32\config\RegBack` からレジストリを復元します。
ログのキャプチャ	分析用にインスタンスでログをキャプチャできます。

EC2Rescue for Windows Server は、アクティブインスタンスおよびオフラインインスタンスから次のデータを収集できます。

項目	説明
イベントログ	アプリケーション、システム、および EC2Config のイベントログを収集します。
[Registry]	`SYSTEM` および `SOFTWARE` Hive を収集します。
[Windows Update Log]	Windows Update によって生成されたログファイルを収集します。 **Note** Windows Server 2016 以降では、ログは Windows イベントトレーシング (ETW) 形式で収集されます。
[Sysprep Log]	Windows システム準備ツールによって生成されたログファイルを収集します。

項目	説明
ドライバセットアップログ	Windows SetupAPI ログ (`setupapi.dev.log` および `setupapi.setup.log`) を収集します。
[Boot Configuration]	`HKEY_LOCAL_MACHINE\BCD00000000` を収集します。
[Memory Dump]	インスタンスに存在するメモリダンプファイルを収集します。
[EC2Config File]	EC2Config サービスによって生成されたログファイルを収集します。
[EC2Launch File]	EC2Launch スクリプトによって生成されたログファイルを収集します。
[SSM Agent File]	SSM Agent、および Patch Manager ログによって生成されたログファイルを収集します。
EC2 ElasticGPU ファイル	Elastic GPU に関連するイベントログを収集します。
ECS	Amazon ECS に関連するログを収集します。
CloudEndure	CloudEndure エージェントに関連するログファイルを収集します。
MGN または DRS ログファイルの AWS レプリケーションエージェント	AWS Transform MGN または AWS Elastic Disaster Recovery に関連するログファイルを収集します。

EC2Rescue for Windows Serverは、アクティブインスタンスから、次の追加データを収集できます。

項目	説明
[System Information]	MSInfo32 を収集します。
グループポリシーの結果	グループポリシーレポートを収集します。

オフラインインスタンスの分析

[Offline Instance] オプションは、Windows インスタンスの起動に関する問題のデバッグに役立ちます。

オフラインインスタンスでアクションを実行するには

1. 動作中の Windows Server インスタンスから、EC2Rescue for Windows Server ツールをダウンロードしてファイルを抽出します。

   Internet Explorer セキュリティ強化の構成 (ESC) を変更せずに EC2Rescue をダウンロードするには、次の PowerShell コマンドを実行します。

   ```
   Invoke-WebRequest https://s3.amazonaws.com/ec2rescue/windows/EC2Rescue_latest.zip -
   OutFile $env:USERPROFILE\Desktop\EC2Rescue_latest.zip
   ```

   このコマンドによって、現在ログインしているユーザーのデスクトップに EC2Rescue.zip ファイルがダウンロードされます。

   > **Note**
   >
   > ファイルのダウンロード時にエラーが表示され、Windows Server 2016 以前のバージョンを使用している場合は、PowerShell ターミナルで TLS 1.2 を有効にする必要がある場合があります。次のコマンドで現在の PowerShell セッションの TLS 1.2 を有効にしてから、もう一度試してください。
   >
   > ```
   > [Net.ServicePointManager]::SecurityProtocol =
   >  [Net.SecurityProtocolType]::Tls12
   > ```

2. 問題のあるインスタンスを、まだ停止していない場合は停止します。

3. 問題のあるインスタンスから EBS ルートボリュームをデタッチし、EC2Rescue for Windows Server がインストールされている動作中の Windows インスタンスにボリュームをアタッチします。
4. 動作しているインスタンスで EC2Rescue for Windows Server ツールを実行して、[Offline Instance (オフラインインスタンス] を選択してください。
5. 新しくマウントされたボリュームのディスクを選択し、[Next] を選択してください。
6. ディスクの選択を確認し、[Yes] を選択してください。
7. 実行するオフラインインスタンスオプションを選択し、[Next] を選択してください。

EC2Rescue for Windows Server ツールはボリュームをスキャンして、選択されたログファイルに基づいてトラブルシューティング情報を収集します。

アクティブなインスタンスからのデータの収集

アクティブなインスタンスからログなどのデータを収集できます。

アクティブなインスタンスからデータを収集するには

1. Windows インスタンスに接続します。
2. EC2Rescue for Windows Server ツールを Windows インスタンスにダウンロードして、ファイルを展開します。

   Internet Explorer セキュリティ強化の構成 (ESC) を変更せずに EC2Rescue をダウンロードするには、次の PowerShell コマンドを実行します。

   ```
   Invoke-WebRequest https://s3.amazonaws.com/ec2rescue/windows/EC2Rescue_latest.zip -
   OutFile $env:USERPROFILE\Desktop\EC2Rescue_latest.zip
   ```

   このコマンドによって、現在ログインしているユーザーのデスクトップに EC2Rescue.zip ファイルがダウンロードされます。

   > **Note**
   >
   > ファイルのダウンロード時にエラーが表示され、Windows Server 2016 以前のバージョンを使用している場合は、PowerShell ターミナルで TLS 1.2 を有効にする必要がある場合があります。次のコマンドで現在の PowerShell セッションの TLS 1.2 を有効にしてから、もう一度試してください。

```
[Net.ServicePointManager]::SecurityProtocol =
 [Net.SecurityProtocolType]::Tls12
```

3. EC2Rescue for Windows Server アプリケーションを開き、使用許諾契約に同意します。
4. [Next]、[Current instance]、[Capture logs] を選択してください。
5. 収集するデータ項目を選択し、[Collect...] を選択してください。警告を読み、[Yes] を選択して続行します。
6. ZIP ファイルのファイル名と場所を選択し、[保存] を選択してください。
7. EC2Rescue for Windows Server が完了したら、[Open Containing Folder (含まれているフォルダを開く)] を選択して ZIP ファイルを表示します。
8. [Finish] を選択してください。

EC2Rescue CLI を使用して問題のある Windows インスタンスのトラブルシューティングを行う

EC2Rescue for Windows Server コマンドラインインターフェース (CLI) を使用すると、EC2Rescue for Windows Server プラグイン (「アクション」と呼ばれます) をプログラムで実行できます。

EC2Rescue for Windows Server ツールには、次の 2 つの実行モードがあります。

- [/online] — EC2Rescue for Windows Server がインストールされているインスタンス上で、ログファイルの収集などのアクションを実行できます。
- /offline:<device_id> — EC2Rescue for Windows Server がインストールされている別個の Amazon EC2 Windows インスタンスにアタッチされているオフラインルートボリュームに対してアクションを実行できます。

EC2Rescue for Windows Server ツールを Windows インスタンスにダウンロードして、ファイルを展開します。次のコマンドを使用してヘルプファイルを表示できます。

```
EC2RescueCmd.exe /help
```

EC2Rescue for Windows Server は、Amazon EC2 Windows インスタンス上で次のアクションを実行できます。

- 収集アクション

- レスキューアクション
- 復元アクション

収集アクション

> **Note**
> すべてのログ、ロググループ全体、またはグループ内の個々のログを収集できます。

EC2Rescue for Windows Server は、アクティブインスタンスおよびオフラインインスタンスから、次のデータを収集できます。

ロググループ	使用可能なログ	説明
`all`		利用可能なすべてのログを収集します。
`eventlog`	• `'Application'` • `'System'` • `'EC2ConfigService'`	アプリケーション、システム、および EC2Config のイベントログを収集します。
`memory-dump`	• `'Memory Dump File'` • `'Mini Dump Files'`	インスタンスに存在するメモリダンプファイルを収集します。
`ec2config`	• `'Log Files'` • `'Configuration Files'`	EC2Config サービスによって生成されたログファイルを収集します。
`ec2launch`	• `'Logs'` • `'Config'`	EC2Launch スクリプトによって生成されたログファイルを収集します。
`ssm-agent`	• `'Log Files'` • `'Patch Baseline Logs'` • `'InstanceData'`	SSM エージェント、および Patch Manager ログによって

ロググループ	使用可能なログ	説明
		生成されたログファイルを収集します。
sysprep	'Log Files'	Windows システム準備ツールによって生成されたログファイルを収集します。
driver-setup	• 'SetupAPI Log Files' • 'DPInst Log File' • 'AWS PV Setup Log File'	Windows SetupAPI ログ (setupapi.dev.log および setupapi.setup.log) を収集します。
registry	• 'SYSTEM' • 'SOFTWARE' • 'BCD'	SYSTEM および SOFTWARE Hive を収集します。
egpu	• 'Event Log' • 'System Files'	Elastic GPU に関連するイベントログを収集します。
boot-config	'BCDEDIT Output'	HKEY_LOCAL_MACHINE\BCD00000000 を収集します。
windows-update	'Log Files'	Windows Update によって生成されたログファイルを収集します。 **Note** Windows Server 2016 以降では、ログは Windows イベントトレーシング (ETW) 形式で収集されます。

ロググループ	使用可能なログ	説明
cloudendure	• 'Migrate Script Logs' • 'Driver Logs' • 'CloudEndure File List'	CloudEndure エージェントに関連するログファイルを収集します。

EC2Rescue for Windows Serverは、アクティブインスタンスから次の追加データを収集できます。

ロググループ	使用可能なログ	説明
system-info	'MSInfo32 Output'	MSInfo32 を収集します。
gpresult	'GPResult Output'	グループポリシーレポートを収集します。

以下のオプションが利用できます。

- [/output:<outputFilePath>] - 収集したログファイルを zip 形式で保存するために必須の送信先ファイルパスの場所。
- [/no-offline] - オフラインモードで使用する省略可能な属性。アクション完了後もボリュームをオフラインに設定しません。
- [/no-fix-signature] - オフラインモードで使用する省略可能な属性。アクション完了後にディスク署名の競合が発生した場合でも修正しません。

例

以下に、EC2Rescue for Windows Server CLI の使用例を示します。

オンラインモードの例

利用可能なすべてのログを収集します。

```
EC2RescueCmd /accepteula /online /collect:all /output:<outputFilePath>
```

特定ロググループのみ収集します。

```
EC2RescueCmd /accepteula /online /collect:ec2config /output:<outputFilePath>
```

ロググループ内の個々のログを収集します。

```
EC2RescueCmd /accepteula /online /collect:'ec2config.Log Files,driver-setup.SetupAPI
 Log Files' /output:<outputFilePath>
```

オフラインモードの例

EBS ボリュームから利用可能なすべてのログを収集します。ボリュームは、device_id 値を使用して指定します。

```
EC2RescueCmd /accepteula /offline:xvdf /collect:all /output:<outputFilePath>
```

特定ロググループのみ収集します。

```
EC2RescueCmd /accepteula /offline:xvdf /collect:ec2config /output:<outputFilePath>
```

レスキューアクション

EC2Rescue for Windows Server は、次のサービス設定を使用して、問題を検出し、対応できます。

サービスグループ	使用可能なアクション	説明
all		
system-time	'RealTimeIsUniversal'	システム時刻 • [RealTimeisUniversal] - RealTimeisUniversal レジストリキーが有効かどうかを検出します。無効の場合、タイムゾーンが UTC 以外の値に設定されていると、Windows システム時刻が狂います。
firewall	• 'Domain networks'	Windows ファイアウォール

サービスグループ	使用可能なアクション	説明
	• 'Private networks' • 'Guest or public networks'	• [ドメインネットワーク] - この Windows ファイアウォールプロファイルが有効であるか無効であるかを検出します。 • [プライベートネットワーク] - この Windows ファイアウォールプロファイルが有効であるか無効であるかを検出します。 • [ゲストまたはパブリックネットワーク] - この Windows ファイアウォールプロファイルが有効であるか無効であるかを検出します。
rdp	• 'Service Start' • 'Remote Desktop Connections' • 'TCP Port'	リモートデスクトップ • [サービスの開始] - リモートデスクトップサービスが有効かどうかを検出します。 • [リモートデスクトップ接続] - これが有効かどうかを検出します。 • [TCP ポート] - リモートデスクトップサービスがリッスンしているポートを検出します。

サービスグループ	使用可能なアクション	説明
ec2config	• 'Service Start' • 'Ec2SetPassword' • 'Ec2HandleUserData'	EC2Config • [サービスの開始] - EC2Config サービスが有効かどうかを検出します。 • [Ec2SetPassword] - 新しい管理者パスワードを生成します。 • [Ec2HandleUserData] - インスタンスの次回の起動時にユーザーデータスクリプトを実行できます。
ec2launch	'Reset Administrator Password'	新しい Windows 管理者パスワードを生成します。
network	'DHCP Service Startup'	ネットワークインターフェイス • [DHCP サービススタートアップ] - DHCP サービスが有効かどうかを検出します。

以下のオプションが利用できます。

- [/level:<level>] - アクションをトリガーする必要があるチェックレベルの省略可能な属性。許容値は、information、warning、error、all のいずれかです。デフォルトでは、error に設定されます。
- [/check-only] - レポートは生成するが、オフラインボリュームに変更は加えない省略可能な属性。

> **Note**
>
> EC2Rescue for Windows Server がディスク署名の衝突の可能性を検出すると、/check-only オプションを使用した場合でも、デフォルトではオフラインプロセスの完了後に署

名を修正します。修正されないようにするには、/no-fix-signature オプションを使用する必要があります。

- [/no-offline] - アクションの完了後にボリュームをオフラインに設定しないための省略可能な属性。
- [/no-fix-signature] - アクションの完了後にディスク署名の競合が発生しても修正しないための省略可能な属性。

レスキューの例

以下に、EC2Rescue for Windows Server CLI の使用例を示します。ボリュームは、device_id 値を使用して指定します。

ボリューム上で特定されたすべての問題の修正を試みる。

```
EC2RescueCmd /accepteula /offline:xvdf /rescue:all
```

ボリューム上のサービスグループ内のすべての問題の修正を試みる。

```
EC2RescueCmd /accepteula /offline:xvdf /rescue:firewall
```

ボリューム上の特定のサービスグループ内の特定の項目の修正を試みる。

```
EC2RescueCmd /accepteula /offline:xvdf /rescue:rdp.'Service Start'
```

ボリューム上で修正を試みる複数の問題を指定する。

```
EC2RescueCmd /accepteula /offline:xvdf /rescue:'system-
time.RealTimeIsUniversal,ec2config.Service Start'
```

復元アクション

EC2Rescue for Windows Server は、次のサービス設定を使用して、問題を検出し、対応できます。

サービスグループ	使用可能なアクション	説明
最後の既知の正常な設定の復元	lkgc	[最後に既知の良好な設定] - 最後に既知のブート可能状態で

サービスグループ	使用可能なアクション	説明
		インスタンスの起動を試みます。
最新のバックアップからの Windows レジストリの復元	`regback`	[バックアップからレジストリを復元] - `\Windows\System32\config\RegBack` からレジストリを復元します。

以下のオプションが利用できます。

- [/no-offline] — アクションの完了後にボリュームをオフラインに設定しないための省略可能な属性。
- [/no-fix-signature] — アクションの完了後にディスク署名の競合が発生しても修正しないための省略可能な属性。

復元の例

以下に、EC2Rescue for Windows Server CLI の使用例を示します。ボリュームは、device_id 値を使用して指定します。

ボリューム上の最後の既知の正常な設定を復元します。

```
EC2RescueCmd /accepteula /offline:xvdf /restore:lkgc
```

ボリューム上の最新の Windows レジストリバックアップを復元します。

```
EC2RescueCmd /accepteula /offline:xvdf /restore:regback
```

EC2Rescue と Systems Manager を使用して問題のある Windows インスタンスのトラブルシューティングを行う

サポート では、Systems Manager Run Command のドキュメントを用意しています。これにより、ご使用の Systems Manager 対応インスタンスとのインターフェイスを使用して、EC2Rescue

for Windows Server を実行できます。この Run Command のドキュメントは、`AWSSupport-RunEC2RescueForWindowsTool` と呼ばれます。

この Systems Manager Run Command のドキュメントでは、次のタスクを実行します。

- EC2Rescue for Windows Server のダウンロードと検証。
- PowerShell モジュールをインポートしてツールとの対話を簡素化。
- 提供されたコマンドとパラメーターで EC2RescueCmd を実行。

Systems Manager Run Command のドキュメントでは、次の 3 つのパラメーターを指定できます。

- [コマンド] — EC2Rescue for Windows Server アクション。現在許可された値は次のとおりです。
  - [ResetAccess] — ローカル管理者のパスワードをリセットします。現在のインスタンスのローカル管理者パスワードはリセットされ、ランダムに生成されたパスワードが `/EC2Rescue/Password/<INSTANCE_ID>` としてパラメータストアに安全に保存されます。このアクションを選択したがパラメータを指定しない場合、パスワードはデフォルトの KMS キー で自動的に暗号化されます。必要に応じて、パラメータに KMS キー ID を指定して、独自のキーでパスワードを暗号化できます。
  - [CollectLogs] — `/collect:all` アクションを指定して、EC2Rescue for Windows Server を実行します。このアクションを選択する場合は、`Parameters` に必ずログのアップロード先となる Amazon S3 バケット名を含むようにしてください。
  - [FixAll] — `/rescue:all` アクションを指定して、EC2Rescue for Windows Server を実行します。このアクションを選択する場合は、`Parameters` に、レスキューするブロックデバイス名が含まれている必要があります。
- [パラメーター] — 指定したコマンドに渡す PowerShell パラメーターを指定します。

要件

ResetAccess 暗号化されたパスワードを Parameter Store に書き込むアクセス許可を付与するポリシーが Amazon EC2 インスタンスにアタッチされている必要があります。ポリシーをアタッチした後、このポリシーを該当する IAM ロールにアタッチしてから、インスタンスのパスワードがリセットされるまで数分かかります。

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:PutParameter"
      ],
      "Resource": [
        "arn:aws:ssm:us-east-1:111122223333:parameter/EC2Rescue/
Passwords/<instanceid>"
      ]
    }
  ]
}
```

デフォルトの KMS キーではなくカスタム KMS キーを使用している場合、代わりにこのポリシーを使用します。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:PutParameter"
      ],
      "Resource": [
        "arn:aws:ssm:us-east-1:111122223333:parameter/EC2Rescue/
Passwords/<instanceid>"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:Encrypt"
      ],
      "Resource": [
        "arn:aws:kms:us-east-1:111122223333:key/<kmskeyid>"
      ]
```

```
    }
  ]
}
```

ドキュメントの JSON を表示する

次の手順では、このドキュメントの JSON を表示する方法について説明します。

Systems Manager Run Command のドキュメントの JSON を表示するには

1. AWS Systems Manager コンソール (https://console.aws.amazon.com/systems-manager/) を開きます。
2. ナビゲーションペインで、[変更管理ツール] を展開し、[ドキュメント] を選択します。
3. 検索バーで `AWSSupport-RunEC2RescueForWindowsTool` と入力し、`AWSSupport-RunEC2RescueForWindowsTool` ドキュメントを選択します。
4. [Content] タブを選択します。

例

以下に、Systems Manager Run Command のドキュメントを使用して、AWS CLI で EC2Rescue for Windows Server を実行する方法をいくつかの例を挙げて説明します。AWS CLI でコマンドを送信する方法の詳細については、 コマンドリファレンスを参照してください。

例

- オフラインルートボリュームで特定されたすべての問題の修正を試みる
- 現在の Amazon EC2 Windows インスタンスから ログを収集する
- ローカル管理者のパスワードをリセットする

オフラインルートボリュームで特定されたすべての問題の修正を試みる

Amazon EC2 Windows インスタンスにアタッチされたオフラインルートボリュームに関するすべての特定された問題の修正を試みます。

```
aws ssm send-command --instance-ids "i-0cb2b964d3e14fd9f" --document-name "AWSSupport-
RunEC2RescueForWindowsTool" --parameters "Command=FixAll, Parameters='xvdf'" --output
 text
```

現在の Amazon EC2 Windows インスタンスからログを収集する

現在のオンライン Amazon EC2 Windows インスタンスからすべてのログを収集し、Amazon S3 バケットにアップロードします。

```
aws ssm send-command --instance-ids "i-0cb2b964d3e14fd9f" --document-name "AWSSupport-RunEC2RescueForWindowsTool" --parameters "Command=CollectLogs, Parameters='amzn-s3-demo-bucket'" --output text
```

ローカル管理者のパスワードをリセットする

次の例では、ローカル管理者パスワードをリセットするために使用するメソッドを示しています。パラメータストアへのリンクが表示され、ランダムに生成された安全なパスワードが発行されます。この RDP はローカル管理者として Amazon EC2 Windows インスタンスで使用します。

デフォルトの AWS KMS key/エイリアス/aws/ssm を使用してオンラインインスタンスのローカル管理者パスワードをリセットする場合:

```
aws ssm send-command --instance-ids "i-0cb2b964d3e14fd9f" --document-name "AWSSupport-RunEC2RescueForWindowsTool" --parameters "Command=ResetAccess" --output text
```

KMS キー を使用してオンラインインスタンスのローカル管理者パスワードをリセットする場合:

```
aws ssm send-command --instance-ids "i-0cb2b964d3e14fd9f" --document-name "AWSSupport-RunEC2RescueForWindowsTool" --parameters "Command=ResetAccess, Parameters=a133dc3c-a2g4-4fc6-a873-6c0720104bf0" --output text
```

> **Note**
>
> この例では、KMS キー は a133dc3c-a2g4-4fc6-a873-6c0720104bf0 です。

インスタンス用 EC2 シリアルコンソール

EC2 シリアルコンソールを使用すると、Amazon EC2 インスタンスのシリアルポートにアクセスできます。このシリアルポートを使用して、起動、ネットワーク設定、およびその他の問題をトラブルシューティングできます。シリアルコンソールではインスタンスにネットワーク機能を持たせる必要はありません。シリアルコンソールを使用すると、キーボードとモニターがインスタンスのシリア

ルポートに直接接続されているかのように、インスタンスにコマンドを入力できます。シリアルコンソールセッションはインスタンスの再起動中および停止中も継続します。再起動中は起動時のすべてのブートメッセージを表示できます。

デフォルトではシリアルコンソールにアクセスできません。組織はアカウントにシリアルコンソールへのアクセスを許可し、IAM ポリシーを設定して、ユーザーにシリアルコンソールへのアクセスを許可する必要があります。シリアルコンソールへのアクセスはインスタンス ID、リソースタグ、その他の IAM レバーを使用して、きめ細かいレベルで制御できます。詳細については「EC2 シリアルコンソールへのアクセスを設定する」を参照してください。

シリアルコンソールにはEC2 コンソールまたは AWS CLI を使用してアクセスできます。

シリアルコンソールは追加料金なしで利用できます。

トピック

- EC2 シリアルコンソールの前提条件
- EC2 シリアルコンソールへのアクセスを設定する
- EC2 シリアルコンソールに接続する
- EC2 シリアルコンソールからの切断
- EC2 シリアルコンソールを使用して Amazon EC2 インスタンスをトラブルシューティングする

EC2 シリアルコンソールの前提条件

EC2 シリアルコンソールに接続し、選択したツールでトラブルシューティングを行うには、以下の前提条件を満たしている必要があります。

- AWS リージョン
- Wavelength ゾーンと AWS Outposts
- ローカルゾーン
- インスタンスのタイプ
- アクセス権を付与する
- ブラウザベースのクライアントのサポート
- インスタンスの状態
- Amazon EC2 Systems Manager
- トラブルシューティングツールを選択

AWS リージョン

すべての AWS リージョン でサポートされています。

Wavelength ゾーンと AWS Outposts

サポート外。

ローカルゾーン

すべての Local Zones ではサポートされています。

インスタンスのタイプ

サポートされるインスタンスタイプ:

- Linux
  - Nitro システム上に構築されたすべての仮想化インスタンス。
  - 以下を除くすべてのベアメタルインスタンス:
    - 汎用: a1.metal、mac1.metal、mac2.metal、mac2-m1ultra.metal、mac2-m2.metal、mac2-m2pro.metal、mac-m3ultra.metal、mac-m4.metal、mac-m4max.metal、および mac-m4pro.metal
    - 高速コンピューティング: g5g.metal
    - メモリ最適化: u-6tb1.metal、u-9tb1.metal、u-12tb1.metal、u-18tb1.metal、u-24tb1.metal
- Windows
  - Nitro システム上に構築されたすべての仮想化インスタンス。
  - 以下を除くすべてのベアメタルインスタンス:
    - 汎用: a1.metal、mac1.metal、mac2.metal、mac2-m1ultra.metal、mac2-m2.metal、mac2-m2pro.metal、mac-m3ultra.metal、mac-m4.metal、mac-m4max.metal、および mac-m4pro.metal
    - 高速コンピューティング: g5g.metal
    - メモリ最適化: u-6tb1.metal、u-9tb1.metal、u-12tb1.metal、u-18tb1.metal、u-24tb1.metal

アクセス権を付与する

EC2 シリアルコンソールへのアクセス権を付与する設定タスクを完了する必要があります。詳細については「EC2 シリアルコンソールへのアクセスを設定する」を参照してください。

ブラウザベースのクライアントのサポート

ブラウザベースのクライアントを使用してシリアルコンソールに接続するにはそのブラウザで WebSocket をサポートしている必要があります。お使いのブラウザが WebSocket をサポートしていない場合は独自のキーとSSH クライアントを使用してシリアルコンソールに接続します。

ブラウザベースのクライアントはポート 443 に接続し、リージョンの EC2 Instance Connect プロキシエンドポイントにアクセスします。エンドポイントの形式はパーティションによって異なります。

- 商用 AWS リージョンと AWS GovCloud (米国) リージョンは、エンドポイント形式を使用します (`prod.region.oneclickv2-proxy.ec2.aws.dev`)。
- 中国リージョンはエンドポイント形式 を使用します (`prod.region.oneclickv2-proxy.ec2.a2z.org.cn`)。

次の例では、各パーティションのエンドポイントが示されています。

- 米国東部 (バージニア北部) リージョンのエンドポイントは `prod.us-east-1.oneclickv2-proxy.ec2.aws.dev` です。
- 中国 (北京) リージョンのエンドポイントは `prod.cn-north-1.oneclickv2-proxy.ec2.a2z.org.cn` です。

> **Note**
>
> ブラウザはこのエンドポイントに到達する必要があります。ファイアウォールまたは VPN が接続をブロックした場合、接続は失敗します。

インスタンスの状態

`running` を指定してください。

インスタンスが `pending`、`stopping`、`stopped`、`shutting-down`、または `terminated` 状態の場合、シリアルコンソールに接続できません。

インスタンスステータスの詳細については「Amazon EC2 インスタンスの状態変更」を参照してください。

Amazon EC2 Systems Manager

インスタンスで Amazon EC2 Systems Manager を使用する場合はSSM Agent のバージョン 3.0.854.0 以降を、そのインスタンスにインストールする必要があります。SSM Agent の詳細についてはAWS Systems Manager ユーザーガイドの「SSM Agentを使用する」を参照してください。

トラブルシューティングツールを選択

シリアルコンソールを使用してインスタンスのトラブルシューティングを行う際に、Linux インスタンスでは GRUB または SysRq、Windows インスタンスでは Special Admin Console (SAC) を使用できます。これらのツールを使用できるようにするにはツールを使用するすべてのインスタンスで設定手順を実行する必要があります。

インスタンスのオペレーティングシステムの手順を使用して、選択したトラブルシューティングツールを設定します。

(Linux インスタンス) GRUB を設定する

GRUB を設定するにはインスタンスの起動に使用された AMI に基づいて、次のいずれかの手順を選択してください。

Amazon Linux 2

Amazon Linux 2 インスタンスで GRUB を設定するには

1. SSH を使用した Linux インスタンスへの接続
2. `/etc/default/grub` で、次のオプションを追加または変更します。
   - `GRUB_TIMEOUT=1` を設定します。
   - `GRUB_TERMINAL="console serial"` を追加します。
   - `GRUB_SERIAL_COMMAND="serial --speed=115200"` を追加します。

   `/etc/default/grub` の例を次に示します。場合によってはシステム設定に基づいて設定を変更することが必要な場合があります。

```
GRUB_CMDLINE_LINUX_DEFAULT="console=tty0 console=ttyS0,115200n8 net.ifnames=0
 biosdevname=0 nvme_core.io_timeout=4294967295 rd.emergency=poweroff rd.shell=0"
```

```
GRUB_TIMEOUT=1
GRUB_DISABLE_RECOVERY="true"
GRUB_TERMINAL="console serial"
GRUB_SERIAL_COMMAND="serial --speed=115200"
```

3. 次のコマンドを実行して、更新された設定を適用します。

```
[ec2-user ~]$ sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

Ubuntu

Ubuntu インスタンスで GRUB を設定するには

1. インスタンスに接続します。
2. /etc/default/grub.d/50-cloudimg-settings.cfg で、次のオプションを追加または変更します。

   - GRUB_TIMEOUT=1 を設定します。
   - GRUB_TIMEOUT_STYLE=menu を追加します。
   - GRUB_TERMINAL="console serial" を追加します。
   - GRUB_HIDDEN_TIMEOUT を削除します。
   - GRUB_SERIAL_COMMAND="serial --speed=115200" を追加します。

   /etc/default/grub.d/50-cloudimg-settings.cfg の例を次に示します。場合によってはシステム設定に基づいて設定を変更することが必要な場合があります。

```
# Cloud Image specific Grub settings for Generic Cloud Images
# CLOUD_IMG: This file was created/modified by the Cloud Image build process

# Set the recordfail timeout
GRUB_RECORDFAIL_TIMEOUT=0

# Do not wait on grub prompt
GRUB_TIMEOUT=1
GRUB_TIMEOUT_STYLE=menu

# Set the default commandline
GRUB_CMDLINE_LINUX_DEFAULT="console=tty1 console=ttyS0
 nvme_core.io_timeout=4294967295"
```

```
# Set the grub console type
GRUB_TERMINAL="console serial"
GRUB_SERIAL_COMMAND="serial --speed 115200"
```

3. 次のコマンドを実行して、更新された設定を適用します。

```
[ec2-user ~]$ sudo update-grub
```

RHEL

RHEL インスタンスで GRUB を設定するには

1. インスタンスに接続します。
2. /etc/default/grub で、次のオプションを追加または変更します。
   - GRUB_TERMINAL_OUTPUT を削除します。
   - GRUB_TERMINAL="console serial" を追加します。
   - GRUB_SERIAL_COMMAND="serial --speed=115200" を追加します。

   /etc/default/grub の例を次に示します。場合によってはシステム設定に基づいて設定を変更することが必要な場合があります。

```
GRUB_TIMEOUT=1
GRUB_DISTRIBUTOR="$(sed 's, release .*$,,g' /etc/system-release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_CMDLINE_LINUX="console=tty0 console=ttyS0,115200n8 net.ifnames=0
 rd.blacklist=nouveau nvme_core.io_timeout=4294967295 crashkernel=auto"
GRUB_DISABLE_RECOVERY="true"
GRUB_ENABLE_BLSCFG=true
GRUB_TERMINAL="console serial"
GRUB_SERIAL_COMMAND="serial --speed=115200"
```

3. 次のコマンドを実行して、更新された設定を適用します。

```
[ec2-user ~]$ sudo grub2-mkconfig -o /boot/grub2/grub.cfg --update-bls-cmdline
```

RHEL 9.2 を使用している場合は次のコマンドを使用します。

```
[ec2-user ~]$ sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

CentOS

CentOS AMI を使用して起動されるインスタンスの場合、GRUB はデフォルトでシリアルコンソール用に設定されます。

`/etc/default/grub` の例を次に示します。構成はシステム設定によって異なる場合があります。

```
GRUB_TIMEOUT=1
GRUB_DISTRIBUTOR="$(sed 's, release .*$,,g' /etc/system-release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL="serial console"
GRUB_SERIAL_COMMAND="serial --speed=115200"
GRUB_CMDLINE_LINUX="console=tty0 crashkernel=auto console=ttyS0,115200"
GRUB_DISABLE_RECOVERY="true"
```

(Linux インスタンス) SysRq を設定する

SysRq を設定するには現在のブートサイクルで SysRq コマンドを有効にします。設定を永続化するために、後続の起動で SysRq コマンドを有効にすることもできます。

現在のブートサイクルについてすべての SysRq コマンドを有効にするには

1. インスタンスに接続します。
2. 次のコマンドを実行します。

```
[ec2-user ~]$ sudo sysctl -w kernel.sysrq=1
```

この設定は次回の再起動時にクリアされます。

後続の起動についてすべての SysRq コマンドを有効にするには

1. `/etc/sysctl.d/99-sysrq.conf` ファイルを作成し、お気に入りのエディタで開きます。

```
[ec2-user ~]$ sudo vi /etc/sysctl.d/99-sysrq.conf
```

2. 次の行を追加します。

```
kernel.sysrq=1
```

3. インスタンスを再起動して、変更を適用します。

```
[ec2-user ~]$ sudo reboot
```

4. login プロンプトで、前に設定したパスワードベースのユーザーのユーザー名を入力し、Enter キーを押します。
5. Password プロンプトで、パスワードを入力し、Enter キーを押します。

(Windows インスタンス) SAC とブートメニューを有効にする

> **Note**
>
> インスタンスで SAC を有効にすると、パスワードの取得に依存する EC2 サービスは Amazon EC2 コンソールから操作できません。Amazon EC2 起動エージェント (EC2Config、EC2Launch v1、EC2Launch v2) での Windowsはシリアルコンソールを使用してさまざまなタスクを実行します。インスタンスで SAC を有効にすると、これらのタスクは正常に実行されません。Amazon EC2 の起動エージェント上の Windows の詳細については「the section called “Windows インスタンスの設定”」を参照してください。SAC を有効にする場合は後で無効にすることができます。詳細については「SAC とブートメニューを無効にする」を参照してください。

インスタンスで SAC とブートメニューを有効にするには次のいずれかの方法を使用します。

PowerShell

Windows インスタンスで SAC とブートメニューを有効にするには

1. インスタンスに接続し、昇格された PowerShell コマンドラインから以下の手順を実行します。
2. SAC を有効にします。

```
bcdedit /ems '{current}' on
bcdedit /emssettings EMSPORT:1 EMSBAUDRATE:115200
```

3. ブートメニューを有効にします。

```
bcdedit /set '{bootmgr}' displaybootmenu yes
bcdedit /set '{bootmgr}' timeout 15
bcdedit /set '{bootmgr}' bootems yes
```

4. インスタンスを再起動して、更新された設定を適用します。

```
shutdown -r -t 0
```

Command prompt

Windows インスタンスで SAC とブートメニューを有効にするには

1. インスタンスに接続し、コマンドプロンプトから次の手順を実行します。
2. SAC を有効にします。

```
bcdedit /ems {current} on
bcdedit /emssettings EMSPORT:1 EMSBAUDRATE:115200
```

3. ブートメニューを有効にします。

```
bcdedit /set {bootmgr} displaybootmenu yes
bcdedit /set {bootmgr} timeout 15
bcdedit /set {bootmgr} bootems yes
```

4. インスタンスを再起動して、更新された設定を適用します。

```
shutdown -r -t 0
```

EC2 シリアルコンソールへのアクセスを設定する

シリアルコンソールへのアクセスを設定するにはアカウントレベルでシリアルコンソールへのアクセスを許可し、ユーザーにアクセス権を付与するように IAM ポリシーを設定する必要がありま

す。Linux インスタンスではユーザーがトラブルシューティングでシリアルコンソールを使用できるように、すべてのインスタンスでパスワードベースのユーザーの設定もする必要があります。

EC2 シリアルコンソールは、仮想シリアルポート接続を使用してインスタンスとやり取りします。この接続はインスタンス VPC から独立しているため、起動障害やネットワーク設定の問題が発生した場合でも、インスタンスオペレーティングシステムを操作しトラブルシューティングツールを実行することができます。この接続は VPC ネットワークの外部にあるため、EC2 シリアルコンソールは、インスタンスへのトラフィックを承認する際は、インスタンスセキュリティグループまたはサブネットネットワーク ACL を使用しません。

[開始する前に]

前提条件が満たされていることを確認します。

内容

- EC2 シリアルコンソールへのアクセスのレベル
- EC2 シリアルコンソールへのアカウントアクセスを管理する
- EC2 シリアルコンソールのアクセスについての IAM ポリシーを設定する
- Linux インスタンスで OS ユーザーパスワードを設定する

EC2 シリアルコンソールへのアクセスのレベル

デフォルトではアカウントレベルでシリアルコンソールにアクセスすることはできません。アカウントレベルでシリアルコンソールへのアクセスを明示的に許可する必要があります。詳細については「EC2 シリアルコンソールへのアカウントアクセスを管理する」を参照してください。

サービスコントロールポリシー (SCP) を使用して、組織内でシリアルコンソールへのアクセスを許可できます。その後、IAM ポリシーを使用してアクセスをコントロールすることで、ユーザーレベルできめ細かいアクセスコントロールを行うことができます。SCP ポリシーと IAM ポリシーを組み合わせて使用することで、シリアルコンソールに対するさまざまなレベルのアクセス制御が可能になります。

組織レベル

サービスコントロールポリシー (SCP) を使用して、組織内のメンバーアカウントのためにシリアルコンソールへのアクセスを許可できます。SCP の詳細についてはAWS Organizations ユーザーガイド の「サービスコントロールポリシー」を参照してください。

インスタンスレベル

IAM PrincipalTag および ResourceTag 構造を使用し、ID でインスタンスを指定することで、シリアルコンソールのアクセスポリシーを設定できます。詳細については「EC2 シリアルコンソールのアクセスについての IAM ポリシーを設定する」を参照してください。

ユーザーレベル

特定のインスタンスのシリアルコンソールサービスに SSH パブリックキーをプッシュするためのアクセス権限を指定ユーザーに許可または拒否するように IAM ポリシーを設定することで、ユーザーレベルでアクセスを設定できます。詳細については「EC2 シリアルコンソールのアクセスについての IAM ポリシーを設定する」を参照してください。

OS レベル (Linux インスタンスのみ)

ユーザーパスワードはゲスト OS レベルで設定できます。これにより、一部のユースケースのためにシリアルコンソールにアクセス権を付与します。ただし、ログをモニタリングするにはパスワードベースのユーザーは必要ありません。詳細については「Linux インスタンスで OS ユーザーパスワードを設定する」を参照してください。

EC2 シリアルコンソールへのアカウントアクセスを管理する

デフォルトではアカウントレベルでシリアルコンソールにアクセスすることはできません。アカウントレベルでシリアルコンソールへのアクセスを明示的に許可する必要があります。

この設定はアカウントレベルで直接、または宣言ポリシーを使用して設定されます。シリアルコンソールへのアクセスを許可する各 AWS リージョン で設定する必要があります。宣言型ポリシーを使用すると、複数のリージョンと複数のアカウントで同時に設定を適用できます。宣言ポリシーが使用されている場合、アカウント内で直接設定を変更することはできません。このトピックでは、アカウント内で設定を直接設定する方法について説明します。宣言ポリシーの使用の詳細についてはAWS Organizations「 ユーザーガイド」の「宣言ポリシー」を参照してください。

目次

- ユーザーにアカウントアクセスを管理するための許可を付与する
- シリアルコンソールへのアカウントアクセスのステータスを表示する
- シリアルコンソールへのアカウントアクセスを許可する
- シリアルコンソールへのアカウントアクセスを拒否する

ユーザーにアカウントアクセスを管理するための許可を付与する

ユーザーが EC2 シリアルコンソールへのアカウントアクセスを管理できるようにするには必要な IAM 許可をユーザーに付与する必要があります。

次のポリシーはアカウントステータスを表示するためのアクセス権限、ならびに EC2 シリアルコンソールへのアカウントアクセスを許可および禁止するためのアクセス権限を付与します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:GetSerialConsoleAccessStatus",
                "ec2:EnableSerialConsoleAccess",
                "ec2:DisableSerialConsoleAccess"
            ],
            "Resource": "*"
        }
    ]
}
```

詳細については、「IAM ユーザーガイド」の「IAM ポリシーの作成」を参照してください。

シリアルコンソールへのアカウントアクセスのステータスを表示する

Console

シリアルコンソールにアカウントアクセスを表示するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、ダッシュボードを選択してください。
3. [アカウント属性] カードの [設定] で、[EC2 シリアルコンソール] を選択します。
4. [EC2 シリアルコンソール] タブで、[EC2 シリアルコンソールアクセス] の値は許可または禁止のいずれかです。

AWS CLI

シリアルコンソールにアカウントアクセスを表示するには

get-serial-console-access-status コマンドを使用します。

```
aws ec2 get-serial-console-access-status
```

以下は出力の例です。true の値は、アカウントがシリアルコンソールへのアクセスを許可されていることを示します。

```
{
    "SerialConsoleAccessEnabled": true,
    "ManagedBy": "account"
}
```

ManagedBy フィールドは、設定を構成したエンティティを示します。指定できる値は account (直接設定) または declarative-policy です。詳細については、「AWS OrganizationsIAM ユーザーガイド」の「管理されたポリシー」を参照してください。

PowerShell

シリアルコンソールにアカウントアクセスを表示するには

Get-EC2SerialConsoleAccessStatus コマンドレットを使用します。

```
Get-EC2SerialConsoleAccessStatus -Select *
```

以下は出力の例です。True の値は、アカウントがシリアルコンソールへのアクセスを許可されていることを示します。

```
ManagedBy SerialConsoleAccessEnabled
--------- --------------------------
account   True
```

ManagedBy フィールドは、設定を構成したエンティティを示します。指定できる値は account (直接設定) または declarative-policy です。詳細については「AWS OrganizationsIAM ユーザーガイド」の「マネージドポリシー」を参照してください。

シリアルコンソールへのアカウントアクセスを許可する

Console

シリアルコンソールへのアカウントアクセスを許可するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、ダッシュボードを選択してください。
3. [アカウント属性] カードの [設定] で、[EC2 シリアルコンソール] を選択します。
4. [管理] をクリックしてください。
5. アカウントにあるすべてのインスタンスの EC2 シリアルコンソールにアクセスを許可するには[許可] チェックボックスをオンにします。
6. [更新] を選択してください。

AWS CLI

シリアルコンソールへのアカウントアクセスを許可するには

enable-serial-console-access コマンドを使用します。

```
aws ec2 enable-serial-console-access
```

以下は出力の例です。

```
{
    "SerialConsoleAccessEnabled": true
}
```

PowerShell

シリアルコンソールへのアカウントアクセスを許可するには

Enable-EC2SerialConsoleAccess コマンドレットを使用します。

```
Enable-EC2SerialConsoleAccess
```

以下は出力の例です。

```
True
```

シリアルコンソールへのアカウントアクセスを拒否する

Console

シリアルコンソールへのアカウントアクセスを拒否するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、ダッシュボードを選択してください。
3. [アカウント属性] カードの [設定] で、[EC2 シリアルコンソール] を選択します。
4. [管理] をクリックしてください。
5. アカウントにあるすべてのインスタンスの EC2 シリアルコンソールにアクセスを禁止するには[許可] チェックボックスをオフにします。
6. [更新] を選択してください。

AWS CLI

シリアルコンソールへのアカウントアクセスを拒否するには

disable-serial-console-access コマンドを使用します。

```
aws ec2 disable-serial-console-access
```

以下は出力の例です。

```
{
    "SerialConsoleAccessEnabled": false
}
```

PowerShell

シリアルコンソールへのアカウントアクセスを拒否するには

Disable-EC2SerialConsoleAccess コマンドレットを使用します。

```
Disable-EC2SerialConsoleAccess
```

以下は出力の例です。

```
False
```

EC2 シリアルコンソールのアクセスについての IAM ポリシーを設定する

デフォルトではユーザーはシリアルコンソールにアクセスできません。組織は IAM ポリシーを設定して、ユーザーに必要なアクセスを許可する必要があります。詳細については、「IAM ユーザーガイド」の「IAM ポリシーの作成」を参照してください。

シリアルコンソールのアクセスについて、ec2-instance-connect:SendSerialConsoleSSHPublicKey アクションを含む JSON ポリシードキュメントを作成します。このアクションはシリアルコンソールセッションを開始するシリアルコンソールサービスにパブリックキーをプッシュするための許可をユーザーに付与します。特定の EC2 インスタンスへのアクセスを制限することをお勧めします。それ以外の場合、この許可を持つすべてのユーザーはすべての EC2 インスタンスのシリアルコンソールに接続できます。

IAM ポリシーの例

- シリアルコンソールへのアクセスを明示的に許可する
- シリアルコンソールへのアクセスを明示的に拒否する
- リソースタグを使用してシリアルコンソールへのアクセスを制御する

シリアルコンソールへのアクセスを明示的に許可する

デフォルトでは誰もシリアルコンソールにアクセスできません。シリアルコンソールへのアクセスを許可するには明示的にアクセスを許可するようにポリシーを設定する必要があります。特定のインスタンスへのアクセスを制限するポリシーを設定することをお勧めします。

次のポリシーはインスタンス ID によって識別される特定のインスタンスのシリアルコンソールへのアクセスを許可します。

DescribeInstances、DescribeInstanceTypes、GetSerialConsoleAccessStatus アクションはリソースレベルの権限をサポートしていないため、これらのアクションには * (アスタリスク) で示されるすべてのリソースを指定する必要があることに注意してください。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowDescribeInstances",
```

```
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceTypes",
                "ec2:GetSerialConsoleAccessStatus"
            ],
             "Resource": "*"
        },
        {
            "Sid": "AllowinstanceBasedSerialConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "ec2-instance-connect:SendSerialConsoleSSHPublicKey"
            ],
            "Resource": "arn:aws:ec2:us-
east-1:111122223333:instance/i-0598c7d356eba48d7"
        }
    ]
}
```

シリアルコンソールへのアクセスを明示的に拒否する

次の IAM ポリシーは* (アスタリスク) で示されるすべてのインスタンスのシリアルコンソールへのアクセスを許可し、ID によって識別される特定のインスタンスのシリアルコンソールへのアクセスを明示的に拒否します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSerialConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "ec2-instance-connect:SendSerialConsoleSSHPublicKey",
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceTypes",
                "ec2:GetSerialConsoleAccessStatus"
            ],
            "Resource": "*"
```

```
        },
        {
            "Sid": "DenySerialConsoleAccess",
            "Effect": "Deny",
            "Action": [
                "ec2-instance-connect:SendSerialConsoleSSHPublicKey"
            ],
            "Resource": "arn:aws:ec2:us-
east-1:111122223333:instance/i-0598c7d356eba48d7"
        }
    ]
}
```

リソースタグを使用してシリアルコンソールへのアクセスを制御する

リソースタグを使用して、インスタンスのシリアルコンソールへのアクセスを制御できます。

属性ベースアクセス制御はユーザーおよび AWS リソースにアタッチできるタグに基づいてアクセス権限を定義する認可戦略です。例えば、次のポリシーはインスタンスのリソースタグとプリンシパルのタグがタグキーについて同じ SerialConsole の値を持っている場合に限り、ユーザーがインスタンスのシリアルコンソール接続を開始することを許可します。

AWS リソースへのアクセスを制御するタグの使用の詳細については、「IAM ユーザーガイド」の「AWS リソースへのアクセス制御」を参照してください。

DescribeInstances、DescribeInstanceTypes、GetSerialConsoleAccessStatus アクションはリソースレベルの権限をサポートしていないため、これらのアクションには * (アスタリスク) で示されるすべてのリソースを指定する必要があることに注意してください。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowDescribeInstances",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceTypes",
```

```
                "ec2:GetSerialConsoleAccessStatus"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowTagBasedSerialConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "ec2-instance-connect:SendSerialConsoleSSHPublicKey"
            ],
            "Resource": "arn:aws:ec2:us-east-1:111122223333:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/SerialConsole":
 "${aws:PrincipalTag/SerialConsole}"
                }
            }
        }
    ]
}
```

Linux インスタンスで OS ユーザーパスワードを設定する

パスワードなしでシリアルコンソールに接続できます。ただし、Linux インスタンスのトラブルシューティングでシリアルコンソールを使用するにはインスタンスにパスワードベースの OS ユーザーがある必要があります。

root ユーザーを含む任意の OS ユーザーに対し、パスワードを設定できます。root ユーザーはすべてのファイルを変更できますが、それ以外の OS ユーザーは権限が制限されていることに注意してください。

シリアルコンソールを使用するすべてのインスタンスについて、ユーザーパスワードを設定する必要があります。これはインスタンスごとに 1 回のみ必要なセットアップです。

> **Note**
>
> デフォルトでは、AWS が提供する AMI には、パスワードベースのユーザーが設定されていません。既にルートユーザーパスワードが設定されている AMI を使用してインスタンスを起動した場合はこれらの手順を省略できます。

Linux インスタンスで OS ユーザーパスワードを設定するには

1. インスタンスに接続します。EC2 シリアルコンソールの接続方法を除き、インスタンスへの接続には任意の方法を使用できます。
2. ユーザーのパスワードを設定するにはpasswd コマンドを使用します。次の例ではユーザーは root です。

   ```
   [ec2-user ~]$ sudo passwd root
   ```

   以下は出力の例です。

   ```
   Changing password for user root.
   New password:
   ```

3. New password のプロンプトに従って、新しいパスワードを入力してください。
4. プロンプトに従って、パスワードを再入力してください。

EC2 シリアルコンソールに接続する

Amazon EC2 コンソールまたは SSH を使用して EC2 インスタンスのシリアルコンソールに接続できます。シリアルコンソールに接続したら、起動、ネットワーク設定、およびその他の問題のトラブルシューティングに使用できます。トラブルシューティングの詳細については「EC2 シリアルコンソールを使用して Amazon EC2 インスタンスをトラブルシューティングする」を参照してください。

考慮事項

- インスタンスごとにサポートされるアクティブなシリアルコンソール接続は 1 つだけです。
- シリアルコンソール接続はユーザーが解除しない限り、通常 1 時間継続します。ただし、システムメンテナンス中はAmazon EC2 によりシリアルコンソールのセッションが切断されます。

  接続の期間は、IAM 認証情報の期間によって決定されません。IAM 認証情報の有効期限が切れても、シリアルコンソール接続の最大期間に達するまで接続は維持されます。EC2 シリアルコンソールコンソールエクスペリエンスを使用しているときに IAM 認証情報の有効期限が切れた場合は、ブラウザページを閉じて接続を終了します。
- シリアルコンソールから切断した後、新しいセッションを許可するためにセッションを終了処理するには30 秒かかります。

- サポートされているシリアルコンソールポート: `ttyS0` (Linux インスタンス) および `COM1` (Windows インスタンス)
- シリアルコンソールに接続すると、インスタンスのスループットがわずかに低下することがあります。

トピック

- ブラウザベースのクライアントを使用した接続
- 独自のキーと SSH クライアントを使用して接続する
- EC2 シリアルコンソールエンドポイントとフィンガープリント

ブラウザベースのクライアントを使用した接続

ブラウザベースのクライアントを使用して、EC2 インスタンスのシリアルコンソールに接続できます。これを行うにはAmazon EC2 コンソールでインスタンスを選択し、シリアルコンソールへの接続を選択してください。ブラウザベースのクライアントはアクセス権限を処理し、正常な接続を提供します。

EC2 シリアルコンソールはほとんどのブラウザで動作し、キーボードとマウスの入力をサポートしています。

接続する前に、前提条件を満たしていることを確認してください。

ブラウザベースのクライアントを使用してインスタンスのシリアルポートに接続するには (Amazon EC2 コンソール)

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで、[インスタンス] を選択してください。
3. インスタンスを選択し、[Actions] (アクション)、[Monitor and troubleshoot] (モニタリングとトラブルシューティング)、[EC2 Serial Console] (EC2 シリアルコンソール)、[Connect] (接続) の順に選択してください。

   またはインスタンスを選択し、[Connect] (接続)、[EC2 Serial Console] (EC2 シリアルコンソール)、[Connect] (接続) の順に選択してください。

   ブラウザ内ターミナルウィンドウが開きます。
4. Enter キーを押します。ログインプロンプトが返された場合はシリアルコンソールに接続されています。

画面が黒いままの場合はシリアルコンソールへの接続に関する問題の解決に役立てるために次の情報を使用できます。

- シリアルコンソールへのアクセスが設定されていることを確認します。詳細については「EC2 シリアルコンソールへのアクセスを設定する」を参照してください。
- (Linux インスタンスのみ) SysRq を使用してシリアルコンソールに接続します。SysRq ではブラウザベースのクライアント経由で接続する必要はありません。詳細については「(Linux インスタンス) SysRq を使用してインスタンスをトラブルシューティングする」を参照してください。
- (Linux インスタンスのみ) getty を再起動します。インスタンスへの SSH アクセスがある場合はSSH を使用してインスタンスに接続し、次のコマンドを使用して getty を再起動します。

```
[ec2-user ~]$ sudo systemctl restart serial-getty@ttyS0
```

- インスタンスを再起動します。SysRq (Linux インスタンス)、EC2 コンソール、または AWS CLI を使用することで、インスタンスを再起動できます。詳細については「(Linux インスタンス) SysRq を使用してインスタンスをトラブルシューティングする」(Linux インスタンス) または「Amazon EC2 インスタンスを再起動する」を参照してください。

5. (Linux インスタンスのみ) `login` プロンプトで、前に設定したパスワードベースのユーザーのユーザー名を入力し、Enter キーを押します。
6. (Linux インスタンスのみ) `Password` プロンプトで、パスワードを入力し、Enter を押します。

独自のキーと SSH クライアントを使用して接続する

シリアルコンソール API の使用中に、独自の SSH キーを使用して、選択した SSH クライアントからインスタンスに接続できます。これにより、インスタンスにパブリックキーをプッシュするシリアルコンソール機能を活用できます。

SSH キーをインスタンスにプッシュした後は、SSH 接続は、ユーザーに EC2 シリアルコンソールアクセスを許可するように設定した IAM ポリシーの対象ではなくなります。

[開始する前に]

前提条件が満たされていることを確認します。

SSH を使用してインスタンスのシリアルコンソールに接続するには

1. SSH パブリックキーをインスタンスにプッシュして、シリアルコンソールのセッションを開始する

   send-serial-console-ssh-public-key コマンドを使用して、SSH パブリックキーをインスタンスにプッシュします。これにより、シリアルコンソールのセッションが開始されます。

   このインスタンスについてシリアルコンソールのセッションが既に開始されている場合、一度に 1 つのセッションしか開くことができないため、コマンドは失敗します。シリアルコンソールから切断した後、新しいセッションを許可するためにセッションを終了処理するには30 秒かかります。

   ```
   aws ec2-instance-connect send-serial-console-ssh-public-key \
       --instance-id i-001234a4bf70dec41EXAMPLE \
       --serial-port 0 \
       --ssh-public-key file://my_key.pub \
       --region us-east-1
   ```

2. プライベートキーを使用してシリアルコンソールに接続する

   パブリックキーをシリアルコンソールサービスから削除する前に、ssh コマンドを使用してシリアルコンソールに接続します。削除されるまで 60 秒かかります。

   パブリックキーに対応するプライベートキーを使用します。

   ユーザー名の形式は `instance-id.port0` であり、これはインスタンス ID とポート 0 で構成されます。次の例ではユーザー名は `i-001234a4bf70dec41EXAMPLE.port0` です。

   シリアルコンソールサービスのエンドポイントはリージョンごとに異なります。各リージョンのエンドポイントについてはEC2 シリアルコンソールエンドポイントとフィンガープリント の表を参照してください。次の例ではシリアルコンソールサービスが us-east-1 リージョンにあります。

   ```
   ssh -i my_key i-001234a4bf70dec41EXAMPLE.port0@serial-console.ec2-instance-
   connect.us-east-1.aws
   ```

   次の例では、`timeout 3600` を使用して SSH セッションが 1 時間後に終了するように設定します。セッション中に開始されたプロセスは、セッション終了後もインスタンスで実行し続ける場合があります。

```
timeout 3600 ssh -i my_key i-001234a4bf70dec41EXAMPLE.port0@serial-console.ec2-instance-connect.us-east-1.aws
```

3. (オプション) フィンガープリントを検証する

   シリアルコンソールの初回接続時に、フィンガープリントを検証するように求めるメッセージが表示されます。シリアルコンソールのフィンガープリントと、検証のために表示されるフィンガープリントを比較できます。これらのフィンガープリントが一致しない場合、「中間者 (MITM) 」攻撃を受けている可能性があります。一致する場合は確信をもってシリアルコンソールに接続できます。

   次のフィンガープリントはus-east-1 リージョンのシリアルコンソールサービス用です。各リージョンのフィンガープリントについてはEC2 シリアルコンソールエンドポイントとフィンガープリント をご参照ください。

   ```
   SHA256:dXwn5ma/xadVMeBZGEru5l2gx+yI5LDiJaLUcz0FMmw
   ```

   フィンガープリントはシリアルコンソールへの初回接続時のみ表示されます。

4. Enter キーを押します。プロンプトが返された場合はシリアルコンソールに接続されています。

   画面が黒いままの場合はシリアルコンソールへの接続に関する問題の解決に役立てるために次の情報を使用できます。

   - シリアルコンソールへのアクセスが設定されていることを確認します。詳細については「EC2 シリアルコンソールへのアクセスを設定する」を参照してください。
   - (Linux インスタンスのみ) SysRq を使用してシリアルコンソールに接続します。SysRq では SSH 経由で接続する必要はありません。詳細については「(Linux インスタンス) SysRq を使用してインスタンスをトラブルシューティングする」を参照してください。
   - (Linux インスタンスのみ) getty を再起動します。インスタンスへの SSH アクセスがある場合はSSH を使用してインスタンスに接続し、次のコマンドを使用して getty を再起動します。

     ```
     [ec2-user ~]$ sudo systemctl restart serial-getty@ttyS0
     ```

   - インスタンスを再起動します。SysRq (Linux インスタンスのみ)、EC2 コンソール、または AWS CLI を使用することで、インスタンスを再起動できます。詳細については「(Linux インスタンス) SysRq を使用してインスタンスをトラブルシューティングする」(Linux インスタンスのみ) または 「Amazon EC2 インスタンスを再起動する」を参照してください。

5. (Linux インスタンスのみ) `login` プロンプトで、前に設定したパスワードベースのユーザーのユーザー名を入力し、Enter キーを押します。
6. (Linux インスタンスのみ) `Password` プロンプトで、パスワードを入力し、Enter を押します。

EC2 シリアルコンソールエンドポイントとフィンガープリント

EC2 シリアルコンソールのサービスエンドポイントとフィンガープリントは次のとおりです。インスタンスのシリアルコンソールにプログラムで接続するにはEC2 シリアルコンソールエンドポイントを使用します。EC2 シリアルコンソールエンドポイントとフィンガープリントはAWS リージョンごとに一意です。

リージョン名	リージョン	エンドポイント	Fingerprint
米国東部 (オハイオ)	us-east-2	serial-console.ec2-instance-connect.us-east-2.aws	SHA256:EhwPkTzRtTY7TRSzz26XbB0/HvV9jRM7mCZN0xw/d/0
米国東部 (バージニア北部)	us-east-1	serial-console.ec2-instance-connect.us-east-1.aws	SHA256:dXwn5ma/xadVMeBZGEru5l2gx+yI5LDiJaLUcz0FMmw
米国西部 (北カリフォルニア)	us-west-1	serial-console.ec2-instance-connect.us-west-1.aws	SHA256:OHldlcMET8u7QLSX3jmRTRAPFHVtqbyoLZBMUCqiH3Y
米国西部 (オレゴン)	us-west-2	serial-console.ec2-instance-connect.us-west-2.aws	SHA256:EMCIe23TqKaBI6yGHainqZcMwqNkDhhAVHa1O2JxVUc
アフリカ (ケープタウン)	af-south-1	ec2-serial-console.af-south-1.api.aws	SHA256:RMWWZ2fVePe

リージョン名	リージョン	エンドポイント	Fingerprint
			JUqzjO5jL2KIgXsczoHlz21Ed00biiWI
アジアパシフィック (香港)	ap-east-1	ec2-serial-console.ap-east-1.api.aws	SHA256:T0Q1lpiXxChoZHplnAkjbP7tkm2xXViC9bJFsjYnifk
アジアパシフィック (台北)	ap-east-2	ec2-serial-console.ap-east-2.api.aws	SHA256:z1rsF5DE8aQqsHwBr/D40tR2GnyMqnScCUa1z1eFwDQ
アジアパシフィック (ハイデラバード)	ap-south-2	ec2-serial-console.ap-south-2.api.aws	SHA256:WJgPBSwV4/shN+OPITValoewAuYj15DVW845JEhDKRs
アジアパシフィック (ジャカルタ)	ap-southeast-3	ec2-serial-console.ap-southeast-3.api.aws	SHA256:5ZwgrCh+lfns32XITqL/4O0zIfbx4bZgsYFqy3o8mIk
アジアパシフィック (マレーシア)	ap-southeast-5	ec2-serial-console.ap-southeast-5.api.aws	SHA256:cQXTHQMRcqRdIjmAGoAMBSExeoRobYyRwT67yTjnEiA
アジアパシフィック (ニュージーランド)	ap-southeast-6	ec2-serial-console.ap-southeast-6.api.aws	SHA256:wNltdH0gVfM5uHJKjrw+ElFuKoJgalcSKqjCS+lLkv4
アジアパシフィック (メルボルン)	ap-southeast-4	ec2-serial-console.ap-southeast-4.api.aws	SHA256:Avaq27hFgLvjn5gTSShZ0oV7h90p0GG46wfOeT6ZJvM

リージョン名	リージョン	エンドポイント	Fingerprint
アジアパシフィック (ムンバイ)	ap-south-1	serial-console.ec2-instance-connect.ap-south-1.aws	SHA256:oB LXcYmklqH HEbliARxEgH8IsO51r ezTPiSM35BsU40
アジアパシフィック (大阪)	ap-northeast-3	ec2-serial-console.ap-northeast-3.api.aws	SHA256:Am0/ jiBKBnBuFnHr9aXs gEV3G8Tu/ vVHFXE/3UcyjsQ
アジアパシフィック (ソウル)	ap-northeast-2	serial-console.ec2-instance-connect.ap-northeast-2.aws	SHA256:FoqWXNX +DZ++GuNTztg9 PK49WYMqBX +FrcZM2dSrqrI
アジアパシフィック (シンガポール)	ap-southeast-1	serial-console.ec2-instance-connect.ap-southeast-1.aws	SHA256:PL FNn7WnCQD Hx3qmwLu1Gy/ O8TUX7LQgZuaC6L 45CoY
アジアパシフィック (シドニー)	ap-southeast-2	serial-console.ec2-instance-connect.ap-southeast-2.aws	SHA256:yF vMwUK9lEU QjQTRoXXzuN+cW9/ VSe9W984Cf5Tgzo4
アジアパシフィック (タイ)	ap-southeast-7	ec2-serial-console.ap-southeast-7.api.aws	SHA256:KC AZiRYrR1Q 2lqsg7vTw ixWmvc2wm jVT31XRgSdEfDY

リージョン名	リージョン	エンドポイント	Fingerprint
アジアパシフィック (東京)	ap-northeast-1	serial-console.ec2-instance-connect.ap-northeast-1.aws	SHA256:RQfsDCZTOfQawewTRDV1t9Em/HMrFQe+CRlIOT5um4k
カナダ (中部)	ca-central-1	serial-console.ec2-instance-connect.ca-central-1.aws	SHA256:P2O2jOZwmpMwkpO6YW738FIOTHdUTyEv2gczYMMO7s4
カナダ西部 (カルガリー)	ca-west-1	ec2-serial-console.ca-west-1.api.aws	SHA256:s3rc8Il2xhbhr3iedjJNxGAFLPGOLjx7IxxXrGckk6Q
中国 (北京)	cn-north-1	ec2-serial-console.cn-north-1.api.amazonwebservices.com.cn	SHA256:2gHVFy4H7uU3+WaFUxD28v/ggMeqjvSlgngpgLgGT+Y
中国 (寧夏)	cn-northwest-1	ec2-serial-console.cn-northwest-1.api.amazonwebservices.com.cn	SHA256:TdgrNZkiQOdVfYEBUhO4SzUA09VWI5rYOZGTogpwmiM
欧州 (フランクフルト)	eu-central-1	serial-console.ec2-instance-connect.eu-central-1.aws	SHA256:aCMFS/yIcOdOlkXvOl8AmZ1Toe+bBnrJJ3Fy0k0De2c

リージョン名	リージョン	エンドポイント	Fingerprint
欧州 (アイルランド)	eu-west-1	serial-console.ec2-instance-connect.eu-west-1.aws	SHA256:h2AaGAWO4Hathhtm6ezs3Bj7udgUxi2qTrHjZAwCW6E
欧州 (ロンドン)	eu-west-2	serial-console.ec2-instance-connect.eu-west-2.aws	SHA256:a69rd5CE/AEG4Amm53I6lkD1ZPvS/BCV3tTPW2RnJg8
欧州 (ミラノ)	eu-south-1	ec2-serial-console.eu-south-1.api.aws	SHA256:IC0kOVJnpgFyBVrxn0A7n99ecLbXSX95cuuS7X7QK30
欧州 (パリ)	eu-west-3	serial-console.ec2-instance-connect.eu-west-3.aws	SHA256:q8ldnAf9pymeNe8BnFVngY3RPAr/kxswJUzfrlxeEWs
欧州 (スペイン)	eu-south-2	ec2-serial-console.eu-south-2.api.aws	SHA256:GoCW2DFRlu669QNxqFxEcsR6fZUz/4F4n7T45ZcwoEc
欧州 (ストックホルム)	eu-north-1	serial-console.ec2-instance-connect.eu-north-1.aws	SHA256:tkGFFUVUDvocDiGSS3Cu8Gdl6w2ul32EPNpKFKLwX84
欧州 (チューリッヒ)	eu-central-2	ec2-serial-console.eu-central-2.api.aws	SHA256:8Ppx2mBMf6WdCw0NUlzKfwM4/IfRz4OaXFutQXWp6mk

リージョン名	リージョン	エンドポイント	Fingerprint
イスラエル (テルアビブ)	il-central-1	ec2-serial-console.il-central-1.api.aws	SHA256:JR 6q8v6kNNP i8+QSFQ4d j5dimNmZP TgwgsM1SNvtYyU
メキシコ (中部)	mx-central-1	ec2-serial-console .mx-central-1.api.aws	SHA256:BC uVl13iQNk+CcVnt18E f4p2ZHUrBBAOxlFetB 32GS0
中東 (バーレーン)	me-south-1	ec2-serial-console .me-south-1.api.aws	SHA256:nP jLLKHu2Qn LdUq2kVAr soK5xvPJO MRJKCBzCDqC3k8
中東 (アラブ首長国連邦)	me-central-1	ec2-serial-console .me-central-1.api.aws	SHA256:zpb5duKiBZ +l0dFwPeyy kB4MPBYhl/ XzXNeFSDKBvLE
南米 (サンパウロ)	sa-east-1	serial-console.ec2-instance-connect.sa-east-1.aws	SHA256:rd2+/32Ognj ew1yVIemENaQzC +Botbih62OqAPDq1dI
AWS GovCloud (US-East)	us-gov-east-1	serial-console.ec2 -instance-connect. us-gov-east-1.amaz onaws.com	SHA256:tI we19GWsoy LClrtvu38YEEh+DHIk qnDcZnmtebvF28
AWS GovCloud (US-West)	us-gov-west-1	serial-console.ec2 -instance-connect. us-gov-west-1.amaz onaws.com	SHA256:kf OFRWLaOZfB +utbd3bRf8OlPf8nG O2YZLqXZiIw5DQ

EC2 シリアルコンソールからの切断

インスタンスの EC2 シリアルコンソール に接続する必要がなくなった場合は接続を切断できます。シリアルコンソールとの接続を切断しても、インスタンスで実行中のすべてのシェルセッションは引き続き実行されます。シェルセッションを終了したい場合はシリアルコンソールとの接続を切断する前に、そのセッションを終了しておく必要があります。

考慮事項

- シリアルコンソール接続はユーザーが解除しない限り、通常 1 時間継続します。ただし、システムメンテナンス中はAmazon EC2 によりシリアルコンソールのセッションが切断されます。
- シリアルコンソールから切断した後、新しいセッションを許可するためにセッションを終了処理するには30 秒かかります。

シリアルコンソールとの接続解除の方法はクライアントによって異なります。

ブラウザベースのクライアント

シリアルコンソールから切断するにはシリアルコンソールのブラウザ内ターミナルウィンドウを閉じます。

標準 OpenSSH クライアント

シリアルコンソールから切断するには次のコマンドを使用して SSH 接続を閉じます。このコマンドは新しい行の直後に実行する必要があります。

```
~.
```

SSH 接続を閉じるために使用するコマンドは使用している SSH クライアントによって異なる場合があります。

EC2 シリアルコンソールを使用して Amazon EC2 インスタンスをトラブルシューティングする

EC2 シリアルコンソールを使用して、インスタンスのシリアルポートに接続することで、起動、ネットワーク設定、およびその他の問題をトラブルシューティングできます。

インスタンスのオペレーティングシステムと、インスタンスで設定したツールの手順を使用します。

ツール

- (Linux インスタンス) GRUB を使用してインスタンスをトラブルシューティングする
- (Linux インスタンス) SysRq を使用してインスタンスをトラブルシューティングする
- (Windows インスタンス) SAC を使用してインスタンスをトラブルシューティングする

前提条件

始める前に、選択したトラブルシューティングツールの設定など、前提条件を満たしていることを確認してください。

(Linux インスタンス) GRUB を使用してインスタンスをトラブルシューティングする

GNU GRUB (GNU GRand Unified Bootloader の略。一般に GRUB と呼ばれます) はほとんどの Linux オペレーティングシステムのデフォルトのブートローダーです。GRUB メニューから、起動先のカーネルを選択したり、メニューエントリを変更してカーネルの起動方法を変更したりできます。これは障害が発生したインスタンスをトラブルシューティングする際に役立ちます。

GRUB メニューはブートプロセス中に表示されます。通常の SSH ではメニューにアクセスできませんが、EC2 シリアルコンソールからアクセスできます。

シングルユーザーモードまたは緊急モードで起動できます。シングルユーザーモードではカーネルを低めの実行レベルで起動します。例えば、ファイルシステムをマウントしても、ネットワークをアクティブ化しない場合があり、インスタンスの修正に必要なメンテナンスを実行することができます。緊急モードはシングルユーザーモードと似ていますが、カーネルは可能な限り低い実行レベルで実行される点が異なります。

シングルユーザーモードで起動するには

1. インスタンスのシリアルコンソールに接続します。
2. 次のコマンドを実行して、インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

3. 再起動時に GRUB メニューが表示されたら、任意のキーを押してブートプロセスを停止します。
4. GRUB メニューで、矢印キーを使用して起動先のカーネルを選択し、キーボードの e を押します。

5. 矢印キーを使用して、カーネルを含む行にカーソルを置きます。行はインスタンスの起動に使用された AMI に応じて、`linux` または `linux16` のいずれかで始まります。Ubuntu の場合、2 つの行は `linux` で始まります。どちらも次のステップで変更する必要があります。
6. 行の最後に、単語 `single` を追加します。

   Amazon Linux 2 の例を次に示します。

   ```
   linux /boot/vmlinuz-4.14.193-149.317.amzn2.aarch64 root=UUID=d33f9c9a-\
   dadd-4499-938d-ebbf42c3e499 ro  console=tty0 console=ttyS0,115200n8 net.ifname\
   s=0 biosdevname=0 nvme_core.io_timeout=4294967295 rd.emergency=poweroff rd.she\
   ll=0 single
   ```
7. シングルユーザーモードで起動するにはCtrl+X キーを押します。
8. `login` プロンプトで、前に設定したパスワードベースのユーザーのユーザー名を入力し、Enter キーを押します。
9. `Password` プロンプトで、パスワードを入力し、Enter キーを押します。

緊急モードで起動するには

シングルユーザーモードと同じ手順に従い、ステップ 6 で `single` の代わりに `emergency` という単語を追加します。

(Linux インスタンス) SysRq を使用してインスタンスをトラブルシューティングする

システムリクエスト (SysRq) キーは「マジック SysRq」とも呼ばれ、シェルの外部でカーネルにコマンドを直接送信するために使用でき、カーネルが何をしているかにかかわらず、カーネルは応答します。例えば、インスタンスが応答を停止した場合、SysRq キーを使用して、カーネルにクラッシュまたは再起動するように指示できます。詳細についてはWikipedia の「マジック SysRq キー」を参照してください。

SysRq コマンドはEC2 シリアルコンソールブラウザベースのクライアントまたは SSH クライアントで使用できます。中断リクエストを送信するコマンドはクライアントごとに異なります。

SysRq を使用するには使用しているクライアントに基づいて、次のいずれかの手順を選択してください。

Browser-based client

シリアルコンソールのブラウザベースのクライアントで SysRq を使用するには

1. インスタンスのシリアルコンソールに接続します。
2. 中断リクエストを送信するにはCTRL+0 (ゼロ) を押します。キーボードがサポートしている場合はPause キーまたは Break キーを使用して中断リクエストを送信することもできます。

```
[ec2-user ~]$ CTRL+0
```

3. SysRq コマンドを発行するには必要なコマンドに対応するキーボードのキーを押します。例えば、SysRq コマンドのリストを表示するにはh を押します。

```
[ec2-user ~]$ h
```

h コマンドは次のような内容を出力します。

```
[ 1169.389495] sysrq: HELP : loglevel(0-9) reboot(b) crash(c) terminate-all-
tasks(e) memory-full-oom-kill(f) kill-all-tasks(i) thaw-filesystems
(j) sak(k) show-backtrace-all-active-cpus(l) show-memory-usage(m) nice-all-RT-
tasks(n) poweroff(o) show-registers(p) show-all-timers(q) unraw(r
) sync(s) show-task-states(t) unmount(u) show-blocked-tasks(w) dump-ftrace-
buffer(z)
```

SSH client

SSH クライアントで SysRq を使用するには

1. インスタンスのシリアルコンソールに接続します。
2. 中断リクエストを送信するには~B (チルダ、その後に大文字の B) を押します。

```
[ec2-user ~]$ ~B
```

3. SysRq コマンドを発行するには必要なコマンドに対応するキーボードのキーを押します。例えば、SysRq コマンドのリストを表示するにはh を押します。

```
[ec2-user ~]$ h
```

h コマンドは次のような内容を出力します。

```
[ 1169.389495] sysrq: HELP : loglevel(0-9) reboot(b) crash(c) terminate-all-
tasks(e) memory-full-oom-kill(f) kill-all-tasks(i) thaw-filesystems
(j) sak(k) show-backtrace-all-active-cpus(l) show-memory-usage(m) nice-all-RT-
tasks(n) poweroff(o) show-registers(p) show-all-timers(q) unraw(r
) sync(s) show-task-states(t) unmount(u) show-blocked-tasks(w) dump-ftrace-
buffer(z)
```

> **Note**
>
> 中断リクエストの送信に使用するコマンドは使用している SSH クライアントによって異なる場合があります。

(Windows インスタンス) SAC を使用してインスタンスをトラブルシューティングする

Windows の Special Admin Console (SAC) 機能を使用して、Windows インスタンスをトラブルシューティングできます。インスタンスのシリアルコンソールに接続して SAC を使用すると、ブートプロセスを中断し、Windows をセーフモードで起動できます。

> **Note**
>
> インスタンスで SAC を有効にすると、パスワードの取得に依存する EC2 サービスは Amazon EC2 コンソールから操作できません。Amazon EC2 起動エージェント (EC2Config、EC2Launch v1、EC2Launch v2) での Windowsはシリアルコンソールを使用してさまざまなタスクを実行します。インスタンスで SAC を有効にすると、これらのタスクは正常に実行されません。Amazon EC2 の起動エージェント上の Windows の詳細については「the section called “Windows インスタンスの設定”」を参照してください。SAC を有効にする場合は後で無効にすることができます。詳細については「SAC とブートメニューを無効にする」を参照してください。

タスク

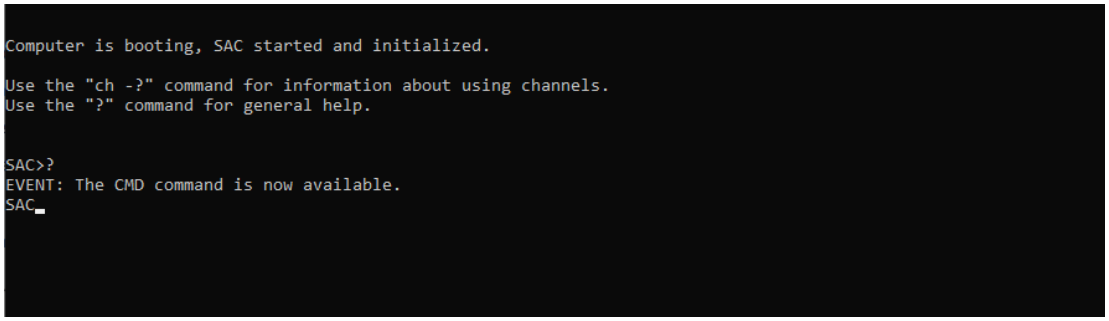
- SAC を使用する
- ブートメニューを使用する
- SAC とブートメニューを無効にする

SAC を使用する

SAC を使用するには

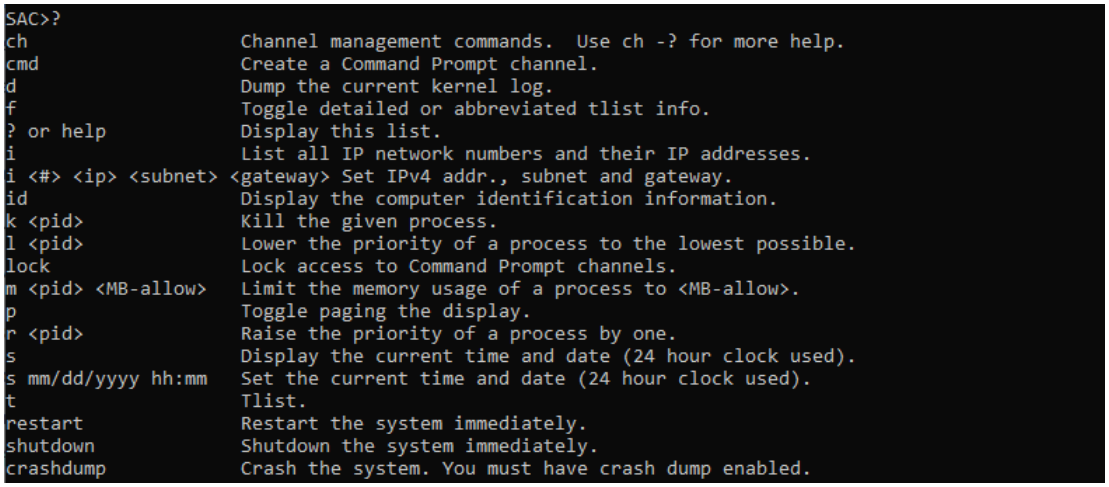
1. シリアルコンソールに接続します。

   インスタンスで SAC が有効になっている場合、シリアルコンソールには SAC> プロンプトが表示されます。



2. SAC コマンドを表示するには と入力し、 Enter.を押します。

   正常な出力



3. コマンドプロンプトチャネル cmd0001 やなど cmd0002を作成するには と入力し、 Enter.を押します。
4. コマンドプロンプトチャネルを表示するにはESC を押してから TAB を押します。

   正常な出力

```
Name:                 Cmd0001
Description:          Command
Type:                 VT-UTF8
Channel GUID:         ef9f20a0-1287-11eb-82b0-0e4ba51872e5
Application Type GUID: 63d02271-8aa4-11d5-bccf-00b0d014a2d0

Press <esc><tab> for next channel.
Press <esc><tab>0 to return to the SAC channel.
Use any other key to view this channel.
```

5. チャネルを切り替えるにはESC + TAB + チャネル番号を同時に押します。例えば、cmd0002 チャネルに切り替えるには (チャネルが作成されている場合)、ESC + TAB + 2 を押します。
6. コマンドプロンプトチャネルに必要な認証情報を入力してください。

```
Please enter login credentials.
Username: Administrator
Domain  : .
Password: *******
```

コマンドプロンプトは既に出力された文字の読み取りを許可しない点を除いて、デスクトップ上で取得するのと同じフル機能のコマンドシェルです。

```
Microsoft Windows [Version 10.0.17763.1457]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>diskpart

Microsoft DiskPart version 10.0.17763.1

Copyright (C) Microsoft Corporation.
On computer: EC2AMAZ-ASR4SAI

DISKPART> list disk

  Disk ###  Status         Size     Free     Dyn  Gpt
  --------  -------------  -------  -------  ---  ---
  Disk 0    Online           30 GB      0 B
  Disk 1    Online           46 GB    46 GB

DISKPART>
```

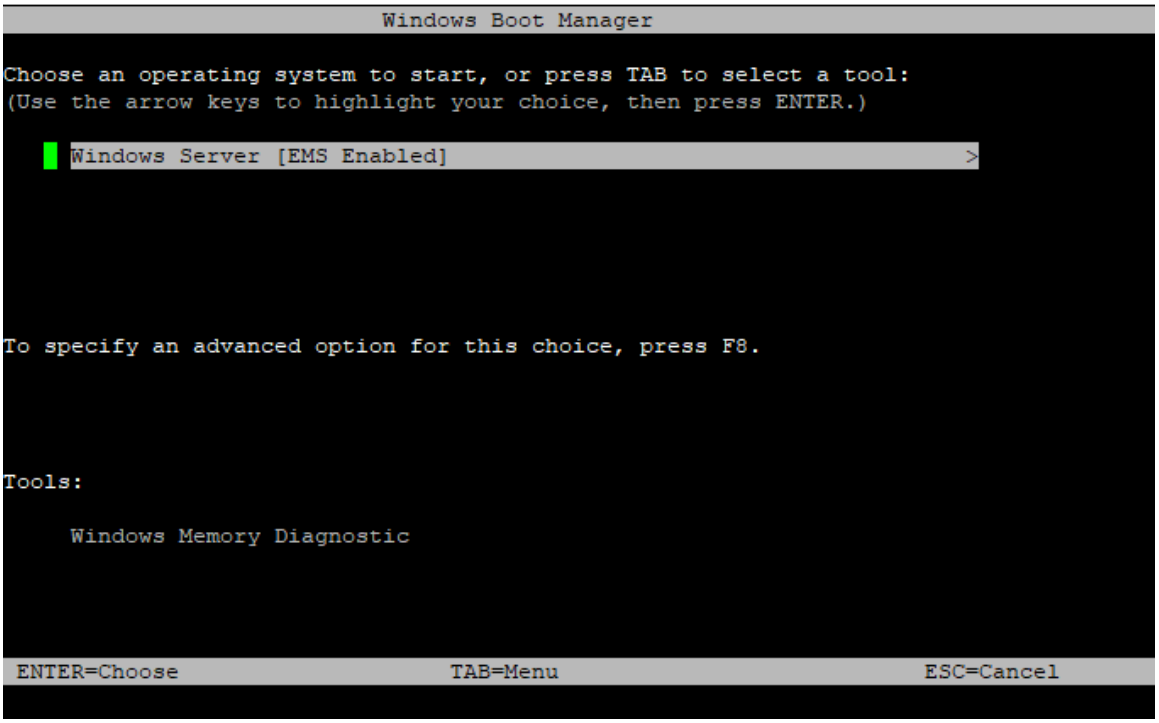
PowerShell はコマンドプロンプトからも使用できます。

進行状況の詳細設定をサイレントモードに設定する必要がある場合があります。

```
PS C:\Windows\system32> $ProgressPreference="SilentlyContinue"
PS C:\Windows\system32> $computerInfo = Get-ComputerInfo
PS C:\Windows\system32> $computerInfo.Csprocessors[0].Name
Intel(R) Xeon(R) Platinum 8124M CPU @ 3.00GHz
PS C:\Windows\system32> $computerInfo.Csprocessors[0].Description
Intel64 Family 6 Model 85 Stepping 4
PS C:\Windows\system32>
```

ブートメニューを使用する

インスタンスでブートメニューが有効になっていて、SSH 経由で接続した後に再起動した場合は次のようにブートメニューが表示されます。



ブートメニューのコマンド

ENTER

オペレーティングシステムの選択したエントリを開始します。

TAB

[Tools] (ツール) メニューに切り替えます。

ESC

インスタンスをキャンセルして再起動します。

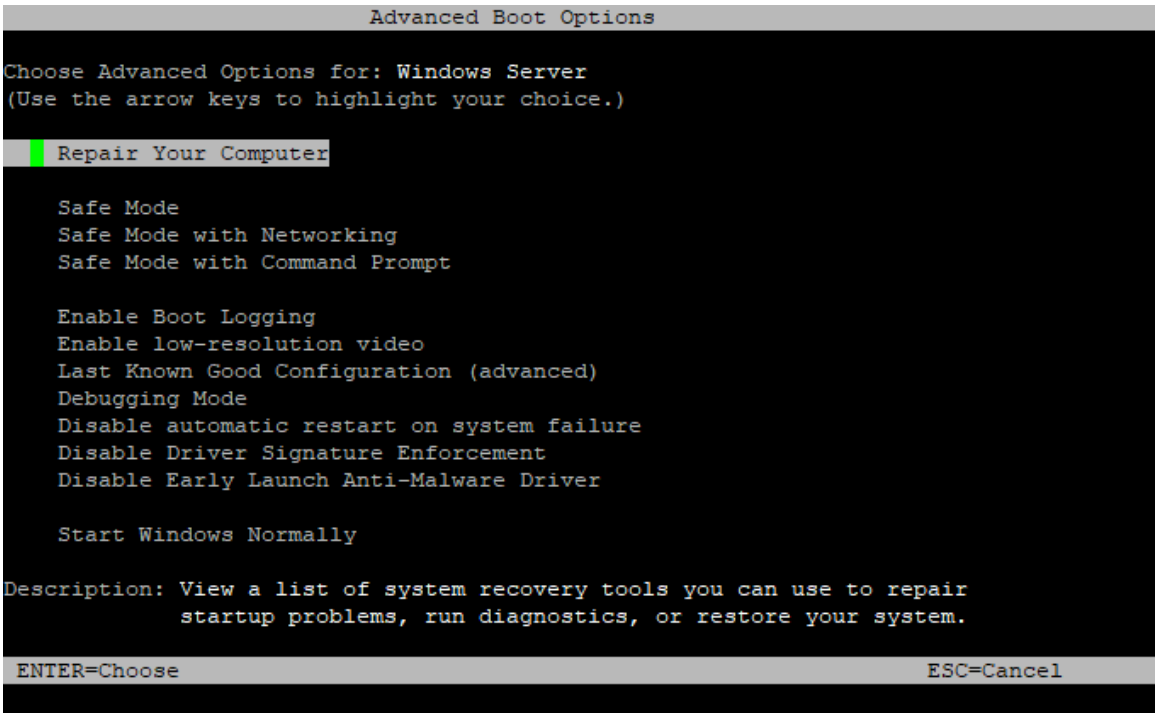
ESC、その後に 8

[F8] を押す操作に相当します。選択した項目の詳細オプションを表示します。

ESC キー + 左矢印

最初のブートメニューに戻ります。

ESC キーだけではWindows が エスケープシーケンスが進行中かどうかを確認するために待機しているため、メインメニューに戻ることはありません。



SAC とブートメニューを無効にする

SAC とブートメニューを有効にする場合、これらの機能を後で無効にできます。

インスタンスで SAC とブートメニューを無効にするには次のいずれかの方法を使用します。

PowerShell

Windows インスタンスで SAC とブートメニューを無効にするには

1. インスタンスに接続し、昇格された PowerShell コマンドラインから以下の手順を実行します。
2. まず、値を no に変更して、ブートメニューを無効にします。

```
bcdedit /set '{bootmgr}' displaybootmenu no
```

3. その後、値を off に変更して SAC を無効にします。

```
bcdedit /ems '{current}' off
```

4. インスタンスを再起動して、更新された設定を適用します。

```
shutdown -r -t 0
```

Command prompt

Windows インスタンスで SAC とブートメニューを無効にするには

1. インスタンスに接続し、コマンドプロンプトから次の手順を実行します。
2. まず、値を no に変更して、ブートメニューを無効にします。

```
bcdedit /set {bootmgr} displaybootmenu no
```

3. その後、値を off に変更して SAC を無効にします。

```
bcdedit /ems {current} off
```

4. インスタンスを再起動して、更新された設定を適用します。

```
shutdown -r -t 0
```

診断割り込みを送信して到達不能の Amazon EC2 インスタンスをデバッグする

> **Warning**
> 診断割り込みは上級ユーザーが使用することを目的としています。不適切な使用はインスタンスに悪影響を与える可能性があります。診断割り込みをインスタンスに送信すると、インスタンスがクラッシュして再起動し、データが失われる可能性があります。

到達できない、または応答しないインスタンスに診断割り込みを送信して、Linux インスタンスのカーネルパニック、または Windows インスタンスの停止エラー (通称ブルースクリーンエラー) を手動でトリガーできます。

Linux インスタンス

Linux オペレーティングシステムは一般的に、カーネルパニックが発生するとクラッシュして再起動されます。ただし、オペレーティングシステムの具体的な動作は設定によって異なります。カーネルパニックはインスタンスのオペレーティングシステムカーネルでクラッシュダンプファイルの生成などのタスクを実行するためにも使用できます。このクラッシュダンプファイル内の情報を使用すると、根本原因解析を実施してインスタンスのデバッグを行うことができます。クラッシュダンプデータはインスタンスの代わりにオペレーティングシステムによってローカルで生成されます。

Windows インスタンス

一般的に、Windows オペレーティングシステムは停止エラーが発生するとクラッシュして再起動されますが、具体的な動作は設定によって異なります。停止エラーが発生すると、オペレーティングシステムからカーネルメモリダンプなどのデバッグ情報がファイルに出力されることもあります。この情報を使用すると、根本原因解析を実施してインスタンスのデバッグを行うことができます。メモリダンプデータはインスタンスの代わりにオペレーティングシステムによってローカルで生成されます。

インスタンスに診断割り込みを送信する前に、OS のドキュメントを参照し、必要な設定変更を行うことをお勧めします。

コンテンツ

- サポートされるインスタンスタイプ
- 前提条件
- 診断割り込みの送信

サポートされるインスタンスタイプ

診断割り込みはNitro ベースのすべてのインスタンスタイプでサポートされます。ただし、AWS Graviton プロセッサで動作するものを除きます。詳細については「Instances built on the AWS Nitro System」と「AWS Graviton」を参照してください。

前提条件

診断割り込みを使用する前に、インスタンスのオペレーティングシステムを設定する必要があります。こうすることで、カーネルパニック (Linux インスタンス) または停止エラー (Windows インスタンス) の発生時に必要なアクションが実行されるようになります。

Linux インスタンス

カーネルパニックの発生時にクラッシュダンプが生成されるように Amazon Linux 2 または Amazon Linux 2023 を設定するには

1. インスタンスに接続します。
2. kexec と kdump をインストールします。

```
[ec2-user ~]$ sudo yum install kexec-tools -y
```

3. セカンダリカーネル用に適切な量のメモリが予約されるようにカーネルを設定します。予約するメモリの量はインスタンスで使用可能な合計メモリによって異なります。適切なテキストエディタを使用して /etc/default/grub ファイルを開き、GRUB_CMDLINE_LINUX_DEFAULT から始まる行を見つけて、crashkernel という形式で crashkernel=*memory_to_reserve* パラメータを追加します。例えば、256MB を予約するにはgrub ファイルを次のように変更します。

```
GRUB_CMDLINE_LINUX_DEFAULT="crashkernel=256M console=tty0 console=ttyS0,115200n8
 net.ifnames=0 biosdevname=0 nvme_core.io_timeout=4294967295 rd.emergency=poweroff
 rd.shell=0"
GRUB_TIMEOUT=0
GRUB_DISABLE_RECOVERY="true"
```

4. 変更内容を保存し、grub ファイルを閉じます。
5. GRUB2 設定ファイルを再構築します。

```
[ec2-user ~]$ sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

6. Intel および AMD プロセッサをベースとしたインスタンスの場合、send-diagnostic-interrupt コマンドを実行すると 不明なマスク不可割り込み (NMI、unknown non-maskable interrupt) がインスタンスに送信されます。不明な NMI を受信した際にはクラッシュするようにカーネルを設定しておく必要があります。適切なテキストエディタを使用して /etc/sysctl.conf ファイルを開き、以下を追加します。

```
kernel.unknown_nmi_panic=1
```

7. インスタンスを再起動して再接続します。
8. 正しいcrashkernel パラメータを使用してカーネルが起動されていることを確認します。

```
$ grep crashkernel /proc/cmdline
```

次の出力例は適切な設定を示しています。

```
BOOT_IMAGE=/boot/vmlinuz-4.14.128-112.105.amzn2.x86_64 root=UUID=a1e1011e-
e38f-408e-878b-fed395b47ad6 ro crashkernel=256M console=tty0 console=ttyS0,115200n8
 net.ifnames=0 biosdevname=0 nvme_core.io_timeout=4294967295 rd.emergency=poweroff
 rd.shell=0
```

9. kdump サービスが実行中であることを確認します。

```
[ec2-user ~]$ systemctl status kdump.service
```

次の出力例はkdump サービスが実行中である場合の結果を示しています。

```
kdump.service - Crash recovery kernel arming
   Loaded: loaded (/usr/lib/systemd/system/kdump.service; enabled; vendor preset:
 enabled)
   Active: active (exited) since Fri 2019-05-24 23:29:13 UTC; 22s ago
  Process: 2503 ExecStart=/usr/bin/kdumpctl start (code=exited, status=0/SUCCESS)
 Main PID: 2503 (code=exited, status=0/SUCCESS)
```

> **Note**
>
> デフォルトではクラッシュダンプファイルは /var/crash/ に保存されます。保存先を変更するには適切なテキストエディタを使用して /etc/kdump.conf ファイルを変更します。

SUSE Linux Enterprise、Ubuntu、または Red Hat Enterprise Linux を設定するには

Intel および AMD プロセッサをベースとしたインスタンスの場合、send-diagnostic-interrupt コマンドを実行すると 不明なマスク不可割り込み (NMI、unknown non-maskable interrupt) がインスタンスに送信されます。オペレーティングシステムの設定ファイルを調整して、不明な NMI を受信したときにカーネルがクラッシュするように設定する必要があります。カーネルをクラッシュするように設定する方法についてはオペレーティングシステムのドキュメントを参照してください。

- SUSE Linux Enterprise
- Ubuntu
- Red Hat Enterprise Linux (RHEL)

Windows インスタンス

停止エラーの発生時に Windows によってメモリダンプが生成されるように設定するには

1. インスタンスに接続します。
2. [コントロールパネル] を開き、[システム]、[システムの詳細設定] の順に選択してください。
3. [システムのプロパティ] ダイアログボックスの [詳細設定] タブを選択してください。
4. [起動と回復] セクションで、[設定...] を選択してください。
5. [システムエラー] セクションで必要に応じて設定を行い、[OK] を選択してください。

Windows 停止エラーの設定の詳細については「Overview of memory dump file options for Windows」を参照してください。

診断割り込みの送信

必要な設定変更を完了したら、AWS CLI または Amazon EC2 API を使用して、診断割り込みをインスタンスに送信できます。

AWS CLI

診断割り込みをインスタンスに送信するには

send-diagnostic-interrupt コマンドを使用します。

```
aws ec2 send-diagnostic-interrupt --instance-id i-1234567890abcdef0
```

PowerShell

診断割り込みをインスタンスに送信するには

Send-EC2DiagnosticInterrupt コマンドレットを使用します。

```
Send-EC2DiagnosticInterrupt -InstanceId i-1234567890abcdef0
```

Amazon EC2 ユーザーガイドのドキュメント履歴

次の表は、2019 年以降の Amazon EC2 ユーザーガイドへの重要な追加項目を記載しています。また、お客様からいただいたフィードバックに対応するために、このガイドを頻繁に更新しています。

変更	説明	日付
既存の EBS ボリュームでの現在のルートボリュームの置き換え	Amazon EC2 インスタンスの現在のルートボリュームを既存の設定済み Amazon EBS ボリュームで置き換えることができます。	2026 年 7 月 9 日
「インスタント」EC2 フリート起動テンプレートの追加のオーバーライド	タイプ `instant` の EC2 フリートが起動テンプレートオーバーライドの追加パラメータ (`IamInstanceProfile` 、`KeyName`、`MetadataOptions`) をサポートするようになりました。	2026 年 7 月 9 日
パブリック AMI の SSM パラメータの検索	パブリック AMI に関連付けられた AWS Systems Manager パラメータを使用して、その AMI の最新バージョンを自動的に参照できます。	2026 年 7 月 7 日
専有ホスト上の SEV-SNP	AMD SEV-SNP が専有ホストでサポートされるようになり、お客様専用の物理サーバーで機密コンピューティングワークロードを実行することが可能になりました。	2026 年 7 月 1 日
AWS グローバルビュー	EC2 グローバルビューの名前が AWS グローバルビューに	2026 年 6 月 30 日

	変更されました。これを使用して、すべてのリージョンにわたる EC2 リソースやその他のサポートされている AWS リソースを 1 つのビューで表示できます。	
精度タイムプレイスメントグループ	精度タイムプレイスメントグループを使用して、AWS インフラストラクチャの高精度のタイムソースに直接アクセス可能なハードウェアにインスタンスを配置できます。	2026 年 6 月 29 日
許可された AMI でのウォーターマークの使用	許可された AMI で AMI ウォーターマークを使用して、指定されたウォーターマークフィルターを持つ AMI のみにインスタンスの起動を制限できるようになりました。新しい許可された AMI パラメータは `ImageWatermarks` です。`MaximumDaysSinceSourceImageCreated` 、`MaximumDaysSinceWatermarkCreated` 、`SourceImageRegion` 、`WatermarkKey` のフィルターが使用可能です。	2026 年 6 月 22 日
Windows 用 AWS VMClock デバイス	AWS は、オペレーティングシステムがデバイスを認識できるようにする Windows 用の null (機能なし) ドライバーパッケージを提供します。	2026 年 6 月 17 日

AMI ウォーターマーク	AMI ウォーターマークを使用し、出所を追跡してプライベート AMI のガバナンスポリシーを実施できます。ウォーターマークは、インスタンスで AMI を作成、AMI をコピー、他のアカウントに AMI を共有するときに保持されます。	2026 年 6 月 10 日
キャパシティ予約のキャンセル	将来の日付のキャパシティ予約をキャンセルできます。キャンセル時期によっては、料金が適用される場合があります。	2026 年 6 月 2 日
ENA Express のクロス AZ サポート	ENA Express は、同じリージョン内の異なるアベイラビリティーゾーンにある Amazon EC2 インスタンス間のトラフィックをサポートするようになりました。	2026 年 5 月 11 日
マネージドリソースの可視性	AWS サービスがユーザーに代わってプロビジョニングするリソースが Amazon EC2 コンソールビューおよび API リストオペレーションに表示されるかどうかを制御できます。	2026 年 4 月 22 日
EFA と NIXL の開始方法	分散推論ワークロードに対して、Elastic Fabric Adapter (EFA) と NIXL を組み合わせて使用できます。	2026 年 3 月 17 日

チュートリアル: 中断可能なキャパシティ予約での EC2 フリートの起動	新しいチュートリアルでは、中断可能なキャパシティ予約を使用して、中断可能なワークロードで未使用のリザーブドキャパシティを利用できるようにする EC2 フリートを作成する方法を説明します。	2026 年 3 月 16 日
ローカルゾーンのキャパシティブロック	機械学習用のキャパシティブロックがローカルゾーンで利用可能になりました。	2026 年 2 月 25 日
IMDSv2 の強制適用	各 AWS リージョンに対し、アカウントレベルで IMDSv2 の使用を強制適用できます。使用が強制されると、インスタンスを起動できるのは IMDSv2 を必須とするようにインスタンスが設定されている場合のみになります。	2026 年 2 月 24 日
ネストされた仮想化	仮想化された Amazon EC2 インスタンス内では、Hyper-V や KVM などのハイパーバイザーを実行できます。	2026 年 2 月 16 日
セカンダリネットワーク	セカンダリネットワークは、特殊なネットワークユースケース専用の仮想ネットワークです。一部の EC2 インスタンスを VPC とセカンダリネットワークの両方で起動できます。	2026 年 2 月 9 日
キャパシティブロックの共有	EC2 キャパシティブロックを AWS 組織内の他の AWS アカウントと共有できます。	2026 年 2 月 2 日

中断可能なキャパシティ予約	中断可能なキャパシティ予約は、アカウント内の他のワークロードで未使用のキャパシティを一時的に利用できるようにするのに役立ちます。	2025 年 11 月 21 日
P6-B300 インスタンスのインスタンストポロジー	`p6-b300.48xlarge` インスタンスのインスタンストポロジー情報を取得できます。	2025 年 11 月 18 日
AMI 祖先	すべての祖先 AMI のリストを取得することで、AMI のオリジンをトレースできるようになりました。	2025 年 11 月 12 日
キャパシティ予約のトポロジー	DescribeCapacityReservationTopology API を使用して、インスタンスを起動する前に、ネットワーク階層内でリザーブドキャパシティが配置される場所を相互に相対的に記述できます。	2025 年 10 月 28 日
ライセンスコストを節約するために CPU を最適化する	ライセンス込みの Windows および SQL Server インスタンスのライセンスコストを節約するためにアクティブな CPU を設定できるようになりました。	2025 年 10 月 15 日

新しい EC2 Capacity Manager	オンデマンドインスタンス、スポットインスタンス、キャパシティ予約のキャパシティ使用状況を 1 か所から監視、分析、管理できる Amazon EC2 Capacity Manager が Amazon EC2 に追加されました。	2025 年 10 月 15 日
AWSEC2CapacityManagerServiceRolePolicy – 新しいマネージドポリシー	ユーザーがキャパシティリソースを管理できるようにし、ユーザーに代わって AWS Organizations と統合するための AWSEC2CapacityManagerServiceRolePolicy マネージドポリシーが Amazon EC2 に追加されました。	2025 年 10 月 15 日
GPUPowerUtilization メトリクス	高速コンピューティングインスタンス向けの新しい GPUPowerUtilization メトリクス。	2025 年 10 月 13 日
EC2 Instance Connect エンドポイントが IPv6 をサポート	EC2 Instance Connect エンドポイントが IPv6 接続をサポートするようになりました。デュアルスタックまたは IPv6 のみをサポートするように EC2 Instance Connect エンドポイントを設定することで、IPv6 アドレスを持つインスタンスに接続できるようになりました。	2025 年 10 月 2 日

許可された AMI – 新しいパラメータ	許可された AMI (AWS アカウント 内で AMI の検出と使用を制限できる機能) は、AWS Marketplace 製品コード、AMI 名、AMI が非推奨になってからの最大経過期間、AMI の最大有効期間という 4 つの新しいパラメータをサポートするようになりました。	2025 年 9 月 25 日
インスタンスストアボリュームの詳細なパフォーマンス統計	Amazon EC2 は、Nitro ベースの インスタンスにアタッチされた NVMe インスタンスストアボリュームに対するリアルタイムかつ高解像度のパフォーマンス統計を提供します。	2025 年 9 月 16 日
Amazon EC2 Mac 専有ホストの更新	Amazon EC2 Mac 専有ホストは、専有ホストの自動復旧と再起動ベースのホストメンテナンスをサポートします。	2025 年 8 月 28 日
ローカルゾーンの AMI コピー	Amazon EBS-backed AMI をローカルゾーンの間でコピーできるようになりました。	2025 年 8 月 28 日
AmazonEC2ImageReferencesAccessPolicy – 新しいマネージドポリシー	Amazon EC2 は、AmazonEC2ImageReferencesAccessPolicy マネージドポリシーを追加し、EC2 DescribeImageReferences API でサポートされているすべてのリソースタイプをスキャンするアクセス許可を付与しました。	2025 年 8 月 26 日

AMI の使用状況レポート	Amazon マシンイメージ (AMI) を他の AWS アカウント (特定の AWS アカウント、組織、組織単位 (OU)、またはパブリック) と共有する場合、AMI の使用状況レポートを作成することで、これらの AMI がどのように使用されているかを確認できます。このレポートでは、EC2 インスタンスまたは起動テンプレートで AMI を使用している AWS アカウントと、各 AMI を参照しているインスタンスまたは起動テンプレートの数を確認できます。	2025 年 8 月 26 日
AMI の参照	指定した Amazon マシンイメージ (AMI) を参照する AWS リソースは、AMI がパブリックかプライベートか、または誰が所有しているかに関係なく識別できます。この可視性により、リソースが最新の準拠 AMI を使用することが保証されます。	2025 年 8 月 26 日
EC2 Instance Connect エンドポイントの変更	AWS CLI または SDK を使用して既存の EC2 Instance Connect エンドポイントを変更できるようになりました。	2025 年 8 月 14 日
キャパシティ予約の共有の機能強化	クラスタープレイスメントグループで作成するキャパシティ予約を共有できるようになりました。	2025 年 8 月 12 日

EC2 Instance Connect エンドポイントの更新された管理ポリシー	既存の EC2 Instance Connect エンドポイントの変更をサポートするため、Amazon EC2 によって IPv6 アドレスの割り当てと割り当て解除、ならびに EC2 Instance Connect エンドポイントによって作成されたネットワークインターフェイスのセキュリティグループを変更するアクセス許可を追加する `Ec2InstanceConnectEndpoint` ポリシーが追加されました。また、Amazon EC2 はこのポリシーを更新して、`Null` 条件演算子を `StringLike` 条件演算子に置き換えました。	2025 年 7 月 31 日
インスタンストポロジ – 追加のリージョンサポート	DescribeInstanceTopology API がアジアパシフィック (メルボルン) で利用可能になりました。	2025 年 7 月 31 日
強制終了	インスタンスを強制終了できるようになりました。	2025 年 7 月 31 日
インスタンストポロジ – 追加のリージョンサポート	DescribeInstanceTopology API が AWS GovCloud (US-West) で利用可能になりました。	2025 年 7 月 24 日
インスタンスの終了時に OS シャットダウンをスキップする	インスタンスを終了する際、正常なオペレーティングシステム (OS) のシャットダウンをスキップできます。	2025 年 7 月 23 日

インスタンスの停止中に OS シャットダウンをスキップする	インスタンスを停止する際、正常なオペレーティングシステム (OS) のシャットダウンをスキップできます。	2025 年 7 月 23 日
us-east-1 で P6-B200 キャパシティブロック	P6-B200 キャパシティブロックは米国東部 (バージニア北部) で利用可能になりました – `us-east-1` 。	2025 年 7 月 23 日
EC2 Fast Launch のサービスリンクロールのマネージドポリシーが更新されました	孤立したリソースの発生を防ぐため、Amazon EC2 ではこのポリシーが更新され、ボリューム、ボリューム属性、ネットワークインターフェイスを記述し、EC2 Fast Launch によって作成されたボリュームおよびネットワークインターフェイスを削除するアクセス許可が追加されました。	2025 年 7 月 17 日
EC2 シリアルコンソール – 追加のリージョンサポート	EC2 シリアルコンソールは、アジアパシフィック (タイ) およびメキシコ (中部) でも利用可能になりました。	2025 年 7 月 15 日
EC2 Instance Connect – 追加のリージョンサポート	EC2 Instance Connect は、アジアパシフィック (タイ) およびメキシコ (中部) でも利用可能になりました。	2025 年 7 月 15 日
ハードウェアパケットタイムスタンプ	EC2 インスタンスでハードウェアパケットタイムスタンプを有効にして、すべての受信ネットワークパケットに 64 ビットのナノ秒精度のタイムスタンプを追加できます。	2025 年 6 月 23 日

AMI の登録解除時にスナップショットを削除する	AMI の登録解除時に、関連付けられたスナップショットを同時に自動で削除できます。	2025 年 5 月 28 日
ユーザー操作による再起動移行	インスタンスが EC2 の再起動イベント予定通知を受信している状態で、イベント予定日時の前にユーザーがインスタンスを再起動すると、インスタンスの移行がデフォルトで有効になります。	2025 年 5 月 20 日
Amazon EC2 Mac インスタンスの SIP 設定	x86 Mac インスタンスと Apple シリコン Mac インスタンスのシステム整合性保護 (SIP) を設定できます。	2025 年 5 月 20 日
ルートボリュームの所有権の委任	Apple シリコン Mac インスタンスでオペレーティングシステムを更新するときに、Amazon EBS ルートボリュームのルートボリューム所有権の委任を自動化できるようになりました。	2025 年 5 月 20 日
P6-B200 インスタンスのインスタンストポロジー	`p6-b200.48xlarge` インスタンスのインスタンストポロジー情報を取得できます。	2025 年 5 月 15 日

EC2 Fast Launch サービスリンクロールのマネージドポリシーを更新	Amazon EventBridge イベントルールを作成して維持するためのアクセスと、EC2 Fast Launch エクスペリエンスを強化するためのその他のアクションで EC2 Fast Launch サービスリンクロールポリシーを更新しました。サービスリンクロールが付与する許可の詳細については、「EC2FastLaunchServiceRolePolicy によって付与されるアクセス許可」を参照してください。	2025 年 5 月 14 日
EC2 Fast Launch フルアクセスポリシーのマネージドポリシーを更新	EC2 Fast Launch フルアクセスポリシーが、起動テンプレートの作成オペレーションと変更オペレーションを行う許可で更新されました。これは、ネットワークとセキュリティグループのリソースを作成する、イングレスルールを承認する、および EC2 Fast Launch が作成したリソースを削除するためのものです。このポリシーはさらに、CloudFormation スタックを記述して作成し、EC2 Fast Launch が作成したスタックを削除することもできます。	2025 年 5 月 14 日
EC2 Fast Launch のオンボーディングを簡素化	オンボーディングプロセスを簡素化するために EC2 Fast Launch を更新しました。	2025 年 5 月 14 日

許可された AMI との AWS Config 統合	ec2-instance-launched-with-allowed-ami AWS Config ルールを使用し、実行中または停止中のインスタンスが、許可された AMI 基準を満たす AMI で起動されたかどうかを確認します。	2025 年 3 月 11 日
AWSEC2CapacityReservationFleetRolePolicy 管理ポリシーの更新	Amazon EC2 では、`StringLike` 条件演算子の代わりに `ArnLike` 条件演算子を使用するように `AWSEC2CapacityReservationFleetRolePolicy` 管理ポリシーを更新しました。	2025 年 3 月 3 日
時間ベースの AMI コピー	AMI コピーが特定の期間内に完了するように、EBS-backed AMI コピー操作の完了期間をリクエストできるようになりました。	2025 年 2 月 25 日
AmazonEC2ReadOnlyAccess ポリシーの更新	Amazon EC2 は、既存の AmazonEC2ReadOnlyAccess ポリシーに `GetSecurityGroupsForVpc` オペレーションを追加しました。	2024 年 12 月 27 日
管理されたインスタンス	Amazon EC2 EC2 管理されたインスタンスを表示できるようになりました。	2024 年 12 月 1 日

宣言ポリシー	宣言ポリシーを使用して、複数のリージョンとアカウントにアカウントレベルの設定を同時に適用できるようになりました。宣言型ポリシーは、VPC、AMI、 AMI スナップショットの EC2 シリアルコンソールアクセス、許可された AMI、IMDS のデフォルト VPCs 、ブロックパブリックアクセスの設定でサポートされています。詳細については、AWS Organizations「 ユーザーガイド」の「宣言ポリシー」およびサポートされている各機能の特定のドキュメントを参照してください。	2024 年 12 月 1 日
許可された AMI	AMI が満たす必要がある基準を指定することで、Amazon EC2 での AMI の検出と使用を制御できるようになりました。	2024 年 12 月 1 日
即時のキャパシティブロック	ML のキャパシティブロックを予約できるようになり、30 分で開始できます。	2024 年 11 月 21 日
将来の日付のキャパシティ予約	将来の日付にキャパシティ予約をリクエストできるようになりました。	2024 年 11 月 21 日
キャパシティブロックを延長	既存のキャパシティブロックの期間を延長できるようになりました。	2024 年 11 月 21 日

6 ヶ月のキャパシティブロック	ML のキャパシティブロックを最大 6 ヶ月間 (182 日間) 予約できるようになりました。	2024 年 11 月 21 日
保存されたフィルターセット	フィルターのカスタマイズされたグループを作成し、EC2 リソースを効率的に表示するために再利用できるようになりました。	2024 年 11 月 20 日
パフォーマンス保護	EC2 フリートまたはスポットフリートに属性ベースのインスタンスタイプの選択を使用するとき、パフォーマンス保護を有効にして、選択したインスタンスタイプが指定されたパフォーマンスベースラインと同等以上にできるようになりました。	2024 年 11 月 20 日
キャパシティ予約のみ	インスタンスがキャパシティ予約またはキャパシティ予約リソースグループでのみ実行されるように指定できるようになりました。	2024 年 11 月 20 日
ソース AMI を特定	AMI の作成に使用されたソース AMI を特定できるようになりました。	2024 年 11 月 13 日
キャパシティの分割	既存のキャパシティ予約からキャパシティを分割し、新規の予約を作成できます。	2024 年 10 月 30 日
キャパシティを移動	キャパシティをキャパシティ予約間で移動できるようになりました。	2024 年 10 月 30 日

初心者向けチュートリアル	初心者向けに、「最初の EC2 インスタンスを起動する」と「テスト EC2 インスタンスを起動し接続する」の 2 つのチュートリアルが新たに提供されるようになりました。	2024 年 10 月 21 日
Windows Server 2025 のサポート	Windows Server 2025 にサポートを追加	2024 年 10 月 16 日
EC2 グローバルビュー	リージョン全体にわたってアカウント内のキャパシティ予約とキャパシティブロックを EC2 グローバルビューに表示できるようになりました。	2024 年 10 月 16 日
ライブ移行ホストメンテナンス	Amazon EC2 専有ホストがライブ移行ホストメンテナンスをサポートするようになりました。これにより、サポート対象のインスタンスはいったん停止してから再起動することなく、障害が発生している専有ホストから代替専有ホストに自動的に移行されます。	2024 年 10 月 15 日
共有キャパシティ予約の請求割り当て	共有キャパシティ予約のうち利用可能な分のキャパシティの請求を、同じ AWS 組織に属するコンシューマーアカウントに割り当てられるようになりました。	2024 年 10 月 14 日

PTP ハードウェアクロック - 追加のリージョンサポート	PTP ハードウェアクロックは、米国東部 (オハイオ) およびアジアパシフィック (マレーシア) でも利用可能になりました。	2024 年 9 月 23 日
EC2 Instance Connect が IPv6 をサポート	EC2 Instance Connect を使用して、インスタンスのパブリック IPv6 アドレスに接続できるようになりました。	2024 年 9 月 23 日
EC2 Instance Connect プレフィックスリスト	セキュリティグループでルールを作成するときに IPv4 アドレスまたは IPv6 アドレスの管理されたプレフィックスリストを選択して、EC2 Instance Connect サービスからの SSH トラフィックを許可できるようになりました。	2024 年 9 月 23 日
オンデマンドキャパシティ予約を管理する新機能	キャパシティ予約を分割したり、キャパシティ予約間でキャパシティを移動したり、キャパシティ予約のインスタンス適格性属性を変更したりできるようになりました。	2024 年 8 月 14 日
C6g、C6gn、C6gd、C7g、C7gd、M6g、M6gd、M7g、M7gd、R6g、R6gd の休止のサポート	C6g、C6gn、C6gd、C7g、C7gd、M6g、M6gd、M7g、M7gd、R6g、R6gd インスタンスタイプで動作中の新たに起動したインスタンスを休止します。	2024 年 7 月 30 日

Graviton インスタンスタイプをサポートする AMI の休止のサポート	Graviton インスタンスタイプをサポートする Amazon Linux または Ubuntu AMI から新たに起動したインスタンスを休止します。	2024 年 7 月 30 日
Credential Guard でサポートされている追加のインスタンスタイプ	C7i、C7-フレックス、M7i、M7i-フレックス、R7i、R7i-フレックス、および T3 インスタンスで Credential Guard を有効にできるようになりました。	2024 年 6 月 26 日
EC2 インスタンスタイプファインダー – 追加のパラメータ	EC2 インスタンスタイプファインダーに、ワークロードのより詳細な要件を指定するための追加のパラメータが備わりました。	2024 年 6 月 5 日
EC2 Fast Launch の新しい管理されたポリシー	インスタンスから EC2 Fast Launch 機能に関連する API アクションを実行する EC2FastLaunchFullAccess ポリシーが追加されました。	2024 年 5 月 14 日
AMI の登録解除保護	AMI の登録解除保護をオンにして、偶発的な削除や悪意のある削除を防ぐことができます。	2024 年 4 月 23 日
PTP ハードウェアクロック — インスタンスタイプのサポート	PTP ハードウェアクロックは M7a、M7g、M7i、R7a、R7i のインスタンスタイプで利用可能になりました。	2024 年 4 月 22 日

ネットワークを強化するための Nitro のパフォーマンスに関する考慮事項を追加	このページでは、Nitro ベースの Amazon EC2 インスタンスのパフォーマンスチューニングに役立つネットワーク上の考慮事項に焦点を当てています。	2024 年 4 月 4 日
VSS ベースの EBS スナップショットの新しい管理ポリシー	Amazon EC2 VSS に新しい IAM 管理ポリシーが登場しました。これをインスタンスプロファイルロールに追加することで、アクセス許可を最新の状態に保ち、ベストプラクティスに従うことができます。	2024 年 3 月 28 日
PTP ハードウェアクロック — 米国東部 (バージニア北部)	PTP ハードウェアクロックが米国東部 (バージニア北部) リージョンで利用可能になりました。	2024 年 3 月 26 日
IMDSv2 をアカウントのデフォルトとして設定する	デフォルトでインスタンスメタデータサービスバージョン 2 (IMDSv2) を使用するように、アカウント内のすべての新しい EC2 インスタンスの起動を設定できます。	2024 年 3 月 25 日
スナップショットから作成された新しい Linux AMI にタグを付ける	スナップショットから Linux AMI を作成する場合、新しい AMI にタグを付けることができます。	2024 年 3 月 7 日

コピー時に新しい AMI とスナップショットにタグを付ける	AMI をコピーするときに、新しい AMI とスナップショットに同じタグを付けるか、異なるタグを付けることができます。	2024 年 3 月 7 日
AWS Management Pack ページを削除する	AWS Management Pack は、主に Windows Server 2012 以前で使用されていました。このレガシー OS のプラットフォームバージョンのサポートは終了しました。AWS とオンプレミスで実行しているサーバーのフリートの管理およびトラブルシューティングについては、「AWS Systems Manager Fleet Manager」を参照してください。	2024 年 2 月 12 日
EC2 Instance Connect は macOS AMI にプリインストールされています	EC2 Instance Connect が macOS Sonoma 14.2.1 以降、macOS Ventura 13.6.3 以降、および macOS Monterey 12.7.2 以降の AMI にプリインストールされるようになりました。	2024 年 1 月 26 日
CentOS、macOS、RHEL の EC2 Instance Connect サポート	サポートされている CentOS、macOS、RHEL AMI に対して EC2 Instance Connect をインストールできるようになりました。	2023 年 12 月 6 日

C7a、C7i、R7a、R7i、および R7iz の休止状態のサポート	C7a、C7i、R7a、R7i、および R7iz インスタンスタイプで実行する、新しく起動したインスタンスを休止状態にします。	2023 年 12 月 1 日
Amazon Q EC2 インスタンスタイプセレクター	Amazon Q EC2 インスタンスタイプセレクターは、優先するユースケース、ワークロードタイプ、CPU メーカーだけではなく、価格とパフォーマンスの優先順位も考慮します。次に、このデータを使用して、新しいワークロードに最適な Amazon EC2 インスタンスタイプのガイダンスと提案を提供します。	2023 年 11 月 28 日
EC2 無料利用枠	EC2 無料利用枠の使用状況は EC2 ダッシュボードから追跡できます。	2023 年 11 月 26 日
Console-to-Code	Console-to-Code は、自動化コードの使用を開始する際に役立ちます。Console-to-Code はコンソールのアクションを記録し、生成 AI を使用して優先する infrastructure-as code 形式のコードを提案します。このコードを出発点として使用し、特定のユースケースに合わせて本番環境に対応できるようにカスタマイズできます。	2023 年 11 月 26 日

設定可能なアイドル接続追跡タイムアウト	セキュリティグループの接続がアイドル状態のままになると、接続追跡が使い果たされることで接続が追跡されなくなり、また、パケットがドロップされる原因となります。今回、Elastic Network Interface で、セキュリティグループの接続追跡のタイムアウトを秒単位で設定できるようになりました。	2023 年 11 月 17 日
PTP ハードウェアクロック	サポート対象のインスタンスに対し、プレシジョンタイムプロトコル (PTP) のハードウェアクロックが搭載されました。PTP ハードウェアクロックでは、NTP または直接 PTP 接続のいずれかがサポートされています。	2023 年 11 月 16 日
休止が有効になっているインスタンスのインスタンスタイプ変更	休止状態が有効になっていて `stopped` 状態にあるインスタンスで、インスタンスのタイプを変更できるようになりました。	2023 年 11 月 16 日
インスタンストポロジー	DescribeInstanceTopology API を使用してインスタンスの場所を検出できます。この情報は、HPC ジョブと ML ジョブを相互に物理的に近いインスタンスで実行することで、それらのジョブを最適化するために使用できます。	2023 年 11 月 13 日

EC2 Fast Launch の共有 AMI サポート	ユーザーにより共有された AMI で、EC2 Fast Launch を有効にできるようになりました。共有した AMI で EC2 Fast Launch を有効にした場合、高速起動の事前プロビジョニングされたスナップショットが、自分のアカウントで作成されます。	2023 年 11 月 6 日
機械学習用のキャパシティブロック	短期間の機械学習 (ML) ワークロードをサポートするために、未来の日付で GPU インスタンスを予約できるようになりました。	2023 年 10 月 31 日
スポットインスタンスの休止状態	オンデマンドインスタンスで現在利用できる休止機能およびインスタンスファミリーと同様のものを使用して、スポットインスタンスを休止できるようになりました。	2023 年 10 月 24 日
AMI のパブリックアクセスのブロックをデフォルトに設定	すべての新規アカウントとパブリック AMI を持たない既存のアカウントで、AMI のパブリックアクセスのブロックがデフォルトで有効になりました。	2023 年 10 月 20 日
Amazon EC2 グローバルビュー	Amazon EC2 Global View が、追加のリソースタイプとカスタマイズ可能な表示オプションをサポート。	2023 年 10 月 18 日

Ubuntu 22.04.2 LTS (Jammy Jellyfish) 休止のサポート	Ubuntu 22.04.2 LTS (Jammy Jellyfish) AMI から新たに起動したインスタンスを休止します。	2023 年 10 月 16 日
AMI の無効化	AMI を無効にして、インスタンスの起動に使用されないようにできます。	2023 年 10 月 12 日
アタッチ済みの EBS ステータスチェック	アタッチ済みの EBS ステータスチェックを使用して、インスタンスにアタッチされている Amazon EBS ボリュームが到達可能かどうかをモニタリングできます。	2023 年 10 月 11 日
Red Hat Enterprise Linux 9 の休止状態サポート	新しく Red Hat Enterprise Linux 9 AMI から起動されたインスタンスでは休止状態が利用可能です。	2023 年 10 月 2 日
Microsoft Windows Server 2022 の休止状態サポート	新しく Microsoft Windows Server 2022 AMI から起動されたインスタンスでは休止状態が利用可能です。	2023 年 10 月 2 日
AL2023 の休止状態サポート	新しく AL2023 AMI から起動されたインスタンスでは休止状態が利用可能です。	2023 年 10 月 2 日
スポットフリート内のスポットインスタンスの中断を開始する	Amazon EC2 コンソールでスポットフリートを選択してフリート内のスポットインスタンスの中断を実行すると、スポットインスタンス上のアプリケーションでの中断に関する処理をテストできます。	2023 年 9 月 21 日

AMI へのパブリックアクセスをブロックする	AMI のパブリックアクセスのブロックをアカウントレベルで有効にして、AMI を公開しようとするあらゆる試みをブロックできます。	2023 年 9 月 12 日
M7i および M7i-フレックス の休止サポート	M7i および M7i-フレックス インスタンスタイプで実行されている新しく起動したインスタンスを休止状態にします。	2023 年 8 月 22 日
EC2-Classic は廃止されました	EC2-Classic では、EC2 インスタンスが他のお客様と共有される単一のフラットネットワーク内で実行されました。Amazon VPC が EC2 クラシックに取って代わります。Amazon VPC では、インスタンスは AWS アカウントから論理的に独立した仮想プライベートクラウド (VPC) で稼働します。	2023 年 8 月
Dedicated Hosts	Dedicated Hosts は、Outpost の特定のハードウェアアセットに割り当てることができます。	2023 年 6 月 20 日
EC2 Instance Connect エンドポイント	インスタンスにパブリック IPv4 アドレスがなくても、SSH または RDP 経由でインスタンスに接続できるようになりました。	2023 年 6 月 13 日

IMDS パッケージアナライザー	IMDS パケットアナライザーを使用して、EC2 インスタンスの IMDSv1 呼び出しのソースを特定できるようになりました。	2023 年 6 月 1 日
EC2 シリアルコンソールのベアメタルインスタンス	EC2 シリアルコンソールは、選択したベアメタルインスタンスのシリアルポートへの接続をサポートするようになりました。	2023 年 4 月 11 日
起動テンプレートのクォータ	起動テンプレートのクォータと起動テンプレートバージョンのクォータを、Service Quotas コンソールと Service Quotas CLI を使用して表示できるようになりました。	2023 年 4 月 3 日
キャパシティ予約使用率の通知	AWS Health は、アカウントのキャパシティ予約のキャパシティ使用率が 20% を下回ったときに通知を送信するようになりました。	2023 年 4 月 3 日
キャパシティ予約グループ	自身が所有するキャパシティ予約グループに、共有を受けるキャパシティ予約を追加できるようになりました。	2023 年 3 月 30 日
インスタンスメタデータオプションの変更	Amazon EC2 コンソールを使用して、インスタンスメタデータオプションを変更することができるようになりました。	2023 年 3 月 20 日

macOS オペレーティングシステムのインプレースアップデート	Apple MacOS オペレーティングシステムのインプレース更新を、M1 Mac インスタンスで実行できるようになりました。	2023 年 3 月 14 日
UEFI Preferred	統合拡張ファームウェアインターフェイス (UEFI) とレガシー BIOS ブートモードの両方をサポートする単一の AMI を作成できるようになりました。	2023 年 3 月 3 日
IMDSv2 用に AMI を変更する	既存の AMI から起動されるインスタンスがデフォルトで IMDSv2 を必須とするように AMI を変更します。	2023 年 2 月 28 日
Windows 仮想化ベースのセキュリティ - Credential Guard	サポートされている Amazon EC2 インスタンスでは、仮想化ベースのセキュリティ (VBS) 機能である Credential Guard を有効にできます。	2023 年 1 月 31 日
EC2 Instance Connect エンドポイントの新しい管理ポリシー	`Ec2InstanceConnectEndpoint` ポリシーが追加されました。これは AWSServiceRoleForEC2InstanceConnect というサービスリンクロールにアタッチされ、EC2 Instance Connect エンドポイントを作成するときにユーザーに変わって Amazon EC2 がアクションを実行できるようにします。	2023 年 1 月 24 日

起動テンプレートの AMI エイリアス	起動テンプレートで AMI ID の代わりに AWS Systems Manager パラメータを指定すると、AMI ID が変更されるたびにテンプレートを更新する必要がなくなります。	2023 年 1 月 19 日
C6i、i3en、および M6i の休止サポート	C6i、I3en、M6i インスタンスタイプで実行されている新しく起動したインスタンスを休止状態にします。	2022 年 12 月 19 日
Torn Write Prevention	ブロックストレージ機能である Torn Write Prevention 機能により、データの回復力に悪影響を及ぼすことなく、I/O 負荷の高いリレーショナルデータベースワークロードのパフォーマンスを向上させ、レイテンシーを削減します。	2022 年 11 月 29 日
ENA Express	ENA Express を使用することで EC2 インスタンス間のネットワークトラフィックのスループットを向上させ、テールレイテンシーを最小限に抑えます。	2022 年 11 月 28 日
AMI タグのコピー	AMI をコピーすると、ユーザー定義の AMI タグも同時にコピーできます。	2022 年 11 月 18 日
保存と復元用の AMI サイズ	Amazon S3 バケットに保存、復元できる AMI のサイズ (圧縮前) が、最大 5,000GB まで可能になりました。	2022 年 11 月 16 日

スポットインスタンスの priceCapacityOptimized 配分戦略	`priceCapacityOptimized` 配分戦略を使用するスポットフリートは、価格と容量の両方を考慮し、中断する可能性が最も低く、価格が最も低いスポットインスタンスプールを選択してください。	2022 年 11 月 10 日
スポットインスタンスの price-capacity-optimized 配分戦略	`price-capacity-optimized` 配分戦略を使用する EC2 フリートは、価格と容量の両方を考慮し、中断する可能性が最も低く、価格が最も低いスポットインスタンスプールを選択してください。	2022 年 11 月 10 日
アカウントと AMI の共有をキャンセルする	AMI が AWS アカウント と共有されていて、そのアカウントとの共有が不要になった場合は、AMI の起動許可からアカウントを削除できます。	2022 年 11 月 4 日
Elastic IP アドレスを移管する	Elastic IP アドレスをある AWS アカウントから別のアカウントに移せるようになりました。	2022 年 10 月 31 日
ルートボリュームを置き換える	AMI を使用して実行中のインスタンスのルート Amazon EBS ボリュームを置き換えることができます。	2022 年 10 月 27 日

インスタンスをデータベースに自動接続する	自動接続機能を使用して、1 つ以上の EC2 インスタンスを RDS データベースにすばやく接続し、これらの間のトラフィックを許可することができます。	2022 年 10 月 10 日
AMI クォータ	AMI を作成および共有する際には、クォータが適用されます。	2022 年 10 月 10 日
IMDSv2 用に AMI を設定する	AMI から起動されるインスタンスがデフォルトで IMDSv2 を必須とするように AMI を設定します。	2022 年 10 月 3 日
スポットインスタンスの中断を開始する	Amazon EC2 コンソールでスポットインスタンスを選択して中断を実行すると、スポットインスタンス上のアプリケーションでの中断に関する処理をテストできます。	2022 年 9 月 26 日
検証済み AMI プロバイダー	Amazon EC2 コンソールでは、Amazon または検証済み Amazon パートナーが所有するパブリック AMI には [Verified provider] (検証済みプロバイダー) のマークが付されます。	2022 年 7 月 22 日
AWS Outposts のプレイスメントグループ	Outpost のプレイスメントグループにホストの分散戦略が追加されました。	2022 年 6 月 30 日
AWS Outposts での Dedicated Hosts	AWS Outposts に Dedicated Hosts を割り当てることができます。	2022 年 5 月 31 日

インスタンス停止の防止	インスタンスが誤って停止するのを防ぐために、インスタンスに対する停止保護を有効にすることができます。	2022 年 5 月 24 日
UEFI Secure Boot	UEFI Secure Boot は、Amazon EC2 の長期にわたって使用されてきたセキュアブートプロセスを基に構築されており、さらなる多重防衛をすることで、再起動後も持続する脅威からソフトウェアを保護します。	2022 年 5 月 10 日
NitroTPM	Nitro Trusted Platform Module (NitroTPM) は、AWS Nitro System によって提供される TPM 2.0 仕様に準拠した仮想デバイスです。	2022 年 5 月 10 日
AMI 状態変更イベント	Amazon EC2 で、AMI の状態が変更したときにイベントが生成されるようになりました。Amazon EventBridge を使用することで、これらのイベントの検出と対応が行えるようになります。	2022 年 5 月 9 日
パブリックキーの説明	パブリックキーと Amazon EC2 キーペアの作成日をクエリできます。	2022 年 4 月 28 日
キーペアを作成する	新しいキーペアを作成するときに、キーの形式 (PEM または PPK) を指定できます。	2022 年 4 月 28 日

起動時に Amazon FSx ファイルシステムをマウントする	新しいインスタンス起動ウィザードを使用して起動時に新規または既存の Amazon FSx for NetApp ONTAP ファイルシステムまたは Amazon FSx for OpenZFS ファイルシステムをマウントできます。	2022 年 4 月 12 日
新しいインスタンス起動ウィザード	Amazon EC2 コンソールの新たに改善された起動エクスペリエンスにより、EC2 インスタンスをすばやく簡単に起動できます。	2022 年 4 月 5 日
パブリック AMI を自動的に非推奨にする	すべてのパブリック AMI を非推奨にする日をデフォルトで AMI 作成日の 2 年後とします。	2022 年 3 月 31 日
インスタンスメタデータカテゴリ: autoscaling/target-lifecycle-state	Auto Scaling グループを使用しているときは、インスタンスメタデータから インスタンスのターゲットライフサイクル状態を取得できます。	2022 年 3 月 24 日
AMI の最終起動時間	`lastLaunchedTime` は、インスタンスの起動のために AMI が最後に使用された時間を示します。	2022 年 2 月 28 日
ED25519 キー	ED25519 キーは EC2 インスタンス Connect および EC2 シリアルコンソールでサポートされるようになりました。	2022 年 1 月 20 日

キャパシティ予約用の追加の RHEL プラットフォーム	オンデマンドキャパシティ予約用の追加の Red Hat Enterprise Linux プラットフォーム。	2022 年 1 月 11 日
Windows AMI を高速起動用に構成する	事前プロビジョニングされたスナップショットを使用して、インスタンスを最大 65% 高速に起動できるように Windows AMI を構成します。	2022 年 1 月 10 日
インスタンスメタデータのインスタンスタグ	インスタンスのメタデータからインスタンスのタグにアクセスできます。	2022 年 1 月 6 日
クラスタープレイスメントグループでのキャパシティ予約	クラスタープレイスメントグループでキャパシティ予約を作成できます。	2022 年 1 月 6 日
スポットフリート launch-before-terminate	スポット群は、新しい置換スポットインスタンスが起動された後に、リバランス通知を受信するスポットインスタンスを終了できます。	2021 年 11 月 4 日
EC2 フリート launch-before-terminate	EC2 フリートは、新しい代替スポットインスタンスが起動された後に、再調整通知を受信したスポットインスタンスを終了できます。	2021 年 11 月 4 日
タイムスタンプの比較	Amazon EC2 Linux インスタンスのタイムスタンプを ClockBound と比較することで、イベントの実際の時刻を判断できます。	2021 年 11 月 2 日

AMI を組織および OU と共有	AMI を次の AWS リソースと共有できるようになりました: 組織と組織単位 (OU)	2021 年 10 月 29 日
スポットプレイスメントスコア	スポットキャパシティ要件に基づく AWS リージョンまたはアベイラビリティーゾーンのレコメンデーションを入手します。	2021 年 10 月 27 日
スポットフリートの属性ベースのインスタンスタイプの選択	インスタンスが持つ必要がある属性を指定すると、Amazon EC2 はそれらの属性を持つすべてのインスタンスタイプを識別します。	2021 年 10 月 27 日
EC2 フリートの属性ベースのインスタンスタイプの選択	インスタンスが持つ必要がある属性を指定すると、Amazon EC2 はそれらの属性を持つすべてのインスタンスタイプを識別します。	2021 年 10 月 27 日
オンデマンドのキャパシティー予約フリート	キャパシティー予約フリートを使用して、キャパシティー予約のグループまたはフリートを起動できます。	2021 年 10 月 5 日
Ubuntu 20.04 LTS での休止状態のサポート – Focal	新しく Ubuntu 20.04 LTS から起動されたインスタンスでは休止状態が利用可能です – Focal AMI。	2021 年 10 月 4 日
EC2 フリートとターゲットを絞ったオンデマンドキャパシティ予約	EC2 フリートは、オンデマンドインスタンスを`targeted` Capacity Reservations に起動することができます。	2021 年 9 月 22 日

Dedicated Hosts での T3 インスタンス	Amazon EC2 Dedicated Hosts での T3 インスタンスのサポート。	2021 年 9 月 14 日
RHEL、Fedora、および CentOS の休止状態のサポート	RHEL、Fedora、および CentOS AMI から起動された新しく起動されたインスタンスを休止状態にします。	2021 年 9 月 9 日
Amazon EC2 グローバルビュー	Amazon EC2 グローバルビューを使用すると、複数の AWSリージョンの VPC、サブネット、インスタンス、セキュリティグループ、およびボリュームを 1 つのコンソールで表示します。	2021 年 9 月 1 日
C5d、M5d、R5d の休止状態のサポート	C5d、M5d、R5d インスタンスタイプで実行されている新しく起動したインスタンスを休止状態にします。	2021 年 8 月 19 日
Amazon EC2 のキーペア	Amazon EC2 は、Linux および Mac インスタンスで ED25519 キーをサポートするようになりました。	2021 年 8 月 17 日
ネットワークインターフェイスのプレフィクス	プライベート IPv4 または IPv6 CIDR 範囲は、自動または手動で、ネットワークインターフェイスに割り当てることができます。	2021 年 7 月 22 日

イベント Windows	スケジュールされたイベントに対して、週ごとに繰り返されるカスタムのイベントウィンドウを定義して、Amazon EC2 インスタンスを再起動、停止、終了させることができます。	2021 年 7 月 15 日
セキュリティグループルールのリソース ID とタグ付けについてのサポート	リソース ID により、セキュリティグループルールを参照することができます。また、セキュリティグループにはタグも追加できます。	2021 年 7 月 7 日
AMI を非推奨にする	AMI を非推奨にするタイミングを指定できるようになりました。	2021 年 6 月 11 日
Windows に対する 1 秒単位の課金	Amazon EC2 では、Windows および SQL Server ベースの使用量に対し、秒単位で課金 (最低料金 1 分間分) されます。	2021 年 6 月 10 日
AWS Outposts でのキャパシティ予約	AWS Outposts で、キャパシティー予約を使用できるようになりました。	2021 年 5 月 24 日
キャパシティ予約の共有	Local Zones および Wavelength Zones で作成されたキャパシティーの予約を共有できるようになりました。	2021 年 5 月 24 日
ルートボリュームの置換	これで、ルートボリューム置換タスクを使用して、実行中のインスタンスのルート EBS ボリュームを置き換えることができます。	2021 年 4 月 22 日

S3 を使用して AMI を保存および復元する	EBS-backed AMI を S3 に保存し、S3 から復元して AMI のクロスパーティションコピーを有効にします。	2021 年 4 月 6 日
EC2 シリアルコンソール	インスタンスのシリアルポートへの接続を確立することにより、起動およびネットワーク接続の問題をトラブルシューティングします。	2021 年 3 月 30 日
ブートモード	Amazon EC2で、選択した AMD および Intel ベースの EC2 インスタンス上で、UEFI ブートがサポートされるようになりました。	2021 年 3 月 22 日
逆引き DNS レコードを作成する	Elastic IP アドレス用に、逆引き DNS ルックアップを設定できるようになりました。	2021 年 2 月 3 日
AMI 作成時の AMI とスナップショットのタグ付け	AMI を作成するときに、AMI とスナップショットに同じタグを付けるか、異なるタグでタグを付けることができます。	2020 年 12 月 4 日
Amazon EventBridge を使用したスポットフリートイベントのモニタリング	スポットフリート状態の変更やエラーに応じてプログラムによるアクションをトリガーする EventBridge ルールを作成します。	2020 年 11 月 20 日
Amazon EventBridge を使用した EC2 フリートイベントのモニタリング	EC2 フリート状態の変更やエラーに応じてプログラムによるアクションをトリガーする EventBridge ルールを作成します。	2020 年 11 月 20 日

instantフリートの削除	タイプ EC2 フリート の `instant` を削除し、1 回の API 呼び出しでフリート内のすべてのインスタンスを終了します。	2020 年 11 月 18 日
T3 および T3a の休止のサポート	T3 および T3a インスタンスタイプで実行されている新しく起動したインスタンスを休止状態にします。	2020 年 11 月 17 日
Amazon EFS の簡易版の作成	Amazon EFS クイック作成を使用することで、起動時にインスタンスに Amazon EFS ファイルシステムを作成してマウントできます。	2020 年 11 月 9 日
インスタンスメタデータのカテゴリ: events/recommendations/rebalance	EC2 インスタンスの再調整推奨通知がインスタンスに対して送信されるおおよその時間 (UTC)。	2020 年 11 月 4 日
EC2 インスタンスの再調整に関するレコメンデーション	スポットインスタンスが中断するリスクが高い場合に通知するシグナル	2020 年 11 月 4 日
Wavelength Zone 内のキャパシティ予約	Wavelength Zone で キャパシティーの予約 を作成して使用できるようになりました。	2020 年 11 月 4 日
キャパシティーの再調整	Amazon EC2 がリバランス推奨を発行したときに、代替スポットインスタンスを起動するようにスポットフリートまたは EC2 フリートを設定します。	2020 年 11 月 4 日

I3、M5ad、および R5ad の休止状態のサポート	I3、M5ad、R5ad インスタンスタイプで実行されている新しく起動したインスタンスを休止状態にします。	2020 年 10 月 21 日
スポットインスタンスの vCPU 制限	スポットインスタンスの上限は、実行中のスポットインスタンスが使用していている vCPU の数、または未処理のリクエストが完了するまでに使用する vCPU の数に関して、管理されるようになりました。	2020 年 10 月 1 日
ローカルゾーンでのキャパシティ予約	Local Zones でキャパシティーの予約を作成して使用できるようになりました。	2020 年 9 月 30 日
M5a および R5a の休止状態のサポート	M5a インスタンスタイプと R5a インスタンスタイプで実行されている新しく起動したインスタンスを休止状態にします。	2020 年 8 月 28 日
インスタンスメタデータにより、インスタンスの場所と配置情報を提供	`placement` カテゴリの新しいインスタンスメタデータフィールド: リージョン、プレイスメントグループ名、パーティション番号、ホスト ID、アベイラビリティーゾーン ID。	2020 年 8 月 24 日

キャパシティ予約グループ	AWS Resource Groups を使用してキャパシティー予約の論理コレクションを作成し、それらのグループ内でターゲットインスタンスを起動できます。	2020 年 7 月 29 日
EC2Launch v2	EC2Launch v2 を使用すると、インスタンスの立ち上げ時、インスタンスが停止した後の起動時、またはインスタンスの再起動時にタスクを実行したり、オンデマンドでタスクを実行したりできます。EC2Launch v2 は Windows Server のすべてのバージョンをサポートしており、これにより、EC2Launch と EC2Config が置き換えられます。	2020 年 6 月 30 日
自分の IPv6 アドレスを使用する	自分の IPv6 アドレス範囲の一部またはすべてを、オンプレミスのネットワークから AWS アカウントに導入できます。	2020 年 5 月 21 日
Systems Manager パラメータを使用してインスタンスを起動する	インスタンスの起動時に、AMI の代わりに AWS Systems Manager パラメータを指定できます。	2020 年 5 月 5 日
スケジュールされたイベント通知のカスタマイズ	スケジュールされたイベント通知をカスタマイズして、メール通知にタグを含めることができます。	2020 年 5 月 4 日

Dedicated Hosts の Windows Server	Amazon が提供する Windows サーバー AMI を使用して、最新バージョンの Windows サーバーを Dedicated Hosts で実行することができます。	2020 年 4 月 7 日
スポットインスタンスの停止と開始	停止中断動作に依存するのではなく、Amazon EBS によってバックアップされたスポットインスタンスを停止し、いつでも開始します。	2020 年 1 月 13 日
リソースへのタグ付け	Egress-only インターネットゲートウェイ、ローカルゲートウェイ、ローカルゲートウェイルートテーブル、ローカルゲートウェイ仮想インターフェイス、ローカルゲートウェイ仮想インターフェイスグループ、ローカルゲートウェイルートテーブル VPC の関連付け、およびローカルゲートウェイルートテーブル仮想インターフェイスグループの関連付けをタグ付けできます。	2020 年 1 月 10 日
Session Manager を使用してインスタンスに接続する	Amazon EC2 コンソールからインスタンスで Session Manager セッションを開始できます。	2019 年 12 月 18 日
Dedicated Hosts およびホストリソースグループ	Dedicated Hosts がホストリソースグループで使用できるようになりました。	2019 年 12 月 2 日

Dedicated Hosts 共有	AWS アカウント間で、Dedicated Hosts を共有できるようになりました。	2019 年 12 月 2 日
アカウントレベルでのデフォルトのクレジット指定	AWS リージョンごとにアカウントレベルで、バースト可能な各パフォーマンスインスタンスファミリーに対し、デフォルトのクレジット仕様を設定できます。	2019 年 11 月 25 日
インスタンスタイプの検出	ニーズに合ったインスタンスタイプを見つけることができます。	2019 年 11 月 22 日
Dedicated Hosts	インスタンスファミリー内にある複数のインスタンスタイプをサポートするように Dedicated Hostを設定できるようになりました。	2019 年 11 月 21 日
インスタンスメタデータサービスバージョン 2	インスタンスメタデータのリクエストに、セッション志向な方法であるインスタンスメタデータサービスバージョン2を使用できます。	2019 年 11 月 19 日
Elastic Fabric Adapter	Elastic Fabric Adapters を インテル MPI 2019 Update 6 と併用できるようになりました。	2019 年 11 月 15 日
オンデマンド Windows インスタンスの休止のサポート	オンデマンド Windows インスタンスを休止できます。	2019 年 10 月 14 日

リザーブドインスタンスの購入予約のキュー登録	リザーブドインスタンス の購入予約を最大 3 年先までキューに入れることができます。	2019 年 10 月 4 日
診断割り込み	到達できないまたは応答しないインスタンスに診断割り込みを送信して、カーネルパニックを手動でトリガーできます。	2019 年 8 月 14 日
容量が最適化された割り当て戦略	EC2 フリートまたはスポットフリートを使用して、起動するインスタンス数に最適な容量で、スポットプールからスポットインスタンスを起動できます。	2019 年 8 月 12 日
オンデマンドによるキャパシティ予約の共有	AWS アカウント間で、キャパシティーの予約を共有できるようになりました。	2019 年 7 月 29 日
Elastic Fabric Adapter	EFA では、Open MPI 3.1.4 および Intel MPI 2019 Update 4 がサポートされるようになりました。	2019 年 7 月 26 日
EC2 インスタントコネクト	EC2 Instance Connect は、Secure Shell (SSH) を使用してインスタンスに接続するシンプルで安全な方法です。	2019 年 6 月 27 日
ホスト復旧	Dedicated Host で予期しないハードウェア障害が発生した場合にインスタンスを新しいホストで自動的に再起動します。	2019 年 6 月 5 日

VSS アプリケーションコンシステントなスナップショット	AWS Systems Manager Run Command を使用して、Windows インスタンスにアタッチされたすべての Amazon EBS ボリュームのアプリケーションコンシステントスナップショットを取得します。	2019 年 5 月 13 日
Microsoft SQL Server データベースでの Windows から Linux へのプラットフォーム変更アシスタント	既存の Microsoft SQL Server ワークロードを Windows から Linux オペレーティングシステムに移行します。更新後のリンクは、「Microsoft SQL Server on Amazon EC2 User Guide」を指します。	2019 年 5 月 8 日
Windows 自動アップグレード	AWS Systems Manager を使用して EC2 Windows インスタンスの自動アップグレードを実行します。	2019 年 5 月 6 日
Elastic Fabric Adapter	High Performance Computing (HPC) アプリケーションを高速化するには、Elastic Fabric Adapter をインスタンスに接続します。	2019 年 4 月 29 日

Amazon EC2 のインスタンスタイプのリリースについては、「Amazon EC2 インスタンスタイプガイド」の「ドキュメント履歴」を参照してください。

2018 年以前の履歴

次の表は、2018 年以前の Amazon EC2 ユーザーガイドへの重要な追加項目を記載しています。

機能	API バージョン	説明	リリース日
パーティションプレイスメントグループ	2016-11-15	パーティションプレイスメントグループはインスタンスを複数の論理パーティションに分散させ、基盤となるハードウェアを 1 つのパーティション内のインスタンスが他のパーティション内のインスタンスと共有しないようにします。詳細については、「パーティションプレイスメントグループ」を参照してください。	2018 年 12 月 20 日
EC2 Linux インスタンスの休止	2016-11-15	休止が有効になっており、前提条件を満たしている場合は、Linux インスタンスを休止状態にすることができます。詳細については、「Amazon EC2 インスタンスの休止」を参照してください。	2018 年 11 月 28 日
Amazon Elastic Inference アクセラレーター	2016-11-15	Amazon EI アクセラレーターをインスタンスにアタッチし、GPU アクセラレーションを追加することで、深層学習の推論を実行するコストを削減できます。	2018 年 11 月 28 日
スポットコンソールはインスタンス群を推奨します	2016-11-15	スポットコンソールは、アプリケーションのニーズに合わせた最小限のハードウェア仕様 (vCPU、メモリ、およびストレージ) を満たすために、スポットのベストプラクティス (インスタンスの多様化) に基づいたインスタンス群を推奨します。詳細については、「スポットフリートを作成する」を参照してください。	2018 年 11 月 20 日
新しい EC2 フリート リクエストタイプ: instant	2016-11-15	EC2 フリート は、インスタンスタイプと購入モデル全体で同時にキャパシティーをプロビジョニングするために使用できる新しいリクエストタイプ、`instant` をサポートしています。`instant` リクエストは、インスタンスを起動するかどうかおよびいつ起動するかについて制御する、API レスポンスで起動されたイン	2018 年 11 月 14 日

機能	API バージョン	説明	リリース日
		スタンスを返します。それ以上のアクションは実行しません。詳細については、「EC2 フリートとスポットフリートのリクエストタイプ」を参照してください。	
スポット削減額情報	2016-11-15	スポットインスタンスを 1 つのスポットフリートまたはすべてのスポットインスタンスに対して使用することで得られる節約額を確認できます。詳細については、「スポットインスタンス購入による削減額」を参照してください。	2018 年 11 月 05 日
CPU オプションを最適化するためのコンソールでのサポート	2016-11-15	インスタンスを起動するとき、Amazon EC2 を使用して特定のワークロードやビジネスニーズに合うように CPU オプションを最適化できます。詳細については、「Amazon EC2 インスタンスに関する CPU オプション」を参照してください。	2018 年 10 月 31 日
インスタンスから起動テンプレートを作成するためのコンソールでのサポート	2016-11-15	Amazon EC2 コンソールを使用した新しい起動テンプレートのためのベースとしてインスタンスを使用して起動テンプレートを作成できます。詳細については、「Amazon EC2 起動テンプレートを作成する」を参照してください。	2018 年 10 月 30 日
On-Demand Capacity Reservations	2016-11-15	特定のアベイラビリティーゾーンの Amazon EC2 インスタンスに対して任意の期間キャパシティーを予約できます。これにより、リザーブドインスタンス (RI) が提供する請求割引とは独立して、キャパシティー予約を登録および管理することができます。詳細については、「EC2 オンデマンドキャパシティ予約を使用してコンピューティングキャパシティを予約」を参照してください。	2018 年 10 月 25 日

機能	API バージョン	説明	リリース日
自分の IP アドレスを使用する (BYOIP)	2016-11-15	すべての公開 IPv4 アドレス範囲の一部またはすべてを、オンプレミスのネットワークから AWS アカウントに導入できます。アドレス範囲を AWS に設定すると、そのアドレス範囲はアドレスプールとしてアカウントに表示されます。アドレスプールから Elastic IP アドレスを作成し、それを AWS リソースで使用できます。詳細については、「Amazon EC2 に自分の IP アドレスを持ち込む (BYOIP)」を参照してください。	2018 年 10 月 23 日
作成時の Dedicated Host タグとコンソールのサポート	2016-11-15	作成時に Dedicated Hosts タグを付けることができ、Amazon EC2 コンソールを使用して Dedicated Host タグを管理できます。詳細については、「アカウントで使用するために Amazon EC2 専有ホストを割り当てる」を参照してください。	2018 年 10 月 08 日
スポットフリートのスケジュールされたスケーリングのコンソールサポート	2016-11-15	日付と時刻に基づいて、フリート現在の容量を増減させます。詳細については、「スケジュールされたスケーリング: スケジュールに従ってスポットフリートをスケールする」を参照してください。	2018 年 9 月 20 日
EC2 フリート の配分戦略	2016-11-15	オンデマンド容量を料金 (最初に最低価格) または優先度 (最初に最も高い優先度) に従って達成するかどうかを指定できます。ターゲットスポット容量を割り当てる先のスポットプールの数を指定できます。詳細については、「配分戦略を使用して、EC2 フリートまたはスポットフリートがどのようにスポットキャパシティとオンデマンドキャパシティを満たすかを決定します。」を参照してください。	2018 年 7 月 26 日

機能	API バージョン	説明	リリース日
スポットフリート の配分戦略	2016-11-15	オンデマンド容量を料金 (最初に最低価格) または優先度 (最初に最も高い優先度) に従って達成するかどうかを指定できます。ターゲットスポット容量を割り当てる先のスポットプールの数を指定できます。詳細については、「配分戦略を使用して、EC2 フリートまたはスポットフリートがどのようにスポットキャパシティとオンデマンドキャパシティを満たすかを決定します。」を参照してください。	2018 年 7 月 26 日
スナップショットライフサイクルの自動化	2016-11-15	Amazon Data Lifecycle Manager を使用して、EBS ボリュームをバックアップするスナップショットの作成と削除を自動化できます。詳細については、「Amazon Data Lifecycle Manager」を参照してください。	2018 年 7 月 12 日
起動テンプレートの CPU オプション	2016-11-15	コマンドラインツールを使用して起動テンプレートを作成すると、特定のワークロードまたはビジネスニーズに合わせて CPU オプションを最適化できます。詳細については、「Amazon EC2 起動テンプレートを作成する」を参照してください。	2018 年 7 月 11 日
Dedicated Hosts のタグ付け	2016-11-15	Dedicated Hosts にタグを付けることができます。	2018 年 7 月 3 日
最新のコンソール出力を取得する	2016-11-15	get-console-output AWS CLI コマンドを使用すると、一部のインスタンスタイプの最新のコンソール出力を取得できます。	2018 年 5 月 9 日
CPU オプションの最適化	2016-11-15	インスタンスを起動するとき、特定のワークロードやビジネスニーズに合うように CPU オプションを最適化できます。詳細については、「Amazon EC2 インスタンスに関する CPU オプション」を参照してください。	2018 年 5 月 8 日

機能	API バージョン	説明	リリース日
EC2 Fleet	2016-11-15	EC2 フリートを使用すると、異なる EC2 インスタンスタイプとアベイラビリティーゾーン間、オンデマンドインスタンス、リザーブドインスタンス、スポットインスタンスの購入モデル間でインスタンスのグループを起動できます。詳細については、「EC2 フリートとスポットフリート」を参照してください。	2018 年 5 月 2 日
スポットフリート のオンデマンドインスタンス	2016-11-15	常にインスタンス容量を確保するため、スポットフリートリクエストにオンデマンドキャパシティーのリクエストを含められます。詳細については、「EC2 フリートとスポットフリート」を参照してください。	2018 年 5 月 2 日
作成中の EBS スナップショットをタグ付けする	2016-11-15	スナップショット作成時にタグを適用できます。	2018 年 4 月 2 日
プレイスメントグループの変更	2016-11-15	インスタンスをプレイスメントグループ内またはプレイスメントグループ外に移動させたり、プレイスメントグループを変更したりできます。詳細については、「EC2 インスタンスの配置を変更する」を参照してください。	2018 年 3 月 1 日
長いリソース ID	2016-11-15	より多くのリソースタイプに対して長い ID 形式を有効にできます。	2018 年 2 月 9 日
ネットワークパフォーマンスの向上	2016-11-15	クラスタプレイスメントグループ外部のインスタンスにおいて、他のインスタンスまたは Amazon S3 との間でネットワークトラフィックを送信または受信する際に、より大きな帯域幅から利点を得られるようになりました。	2018 年 1 月 24 日

機能	API バージョン	説明	リリース日
Elastic IP アドレスにタグを付ける	2016-11-15	Elastic IP アドレスにタグを付けることができます。	2017 年 12 月 21 日
Amazon Time Sync Service のご紹介	2016-11-15	Amazon Time Sync Service を使用して、インスタンスの正確な時刻を維持できます。詳細については、「EC2 インスタンス上の正確なクロックと時刻の同期」を参照してください。	2017 年 11 月 29 日
T2 無制限	2016-11-15	T2 無制限インスタンスは、ベースラインを超えるレベルで必要なだけバーストさせることができます。詳細については、「バーストパフォーマンスインスタンス」を参照してください。	2017 年 11 月 29 日
起動テンプレート	2016-11-15	起動テンプレートにインスタントを起動させるパラメータの全体または一部を含めることで、インスタンス起動のたびに指定する必要がなくなります。詳細については、「Amazon EC2 起動テンプレートにインスタンス起動パラメータを保存する」を参照してください。	2017 年 11 月 29 日
スプレッドプレイスメント	2016-11-15	スプレッドプレイスメントグループは、少数の重要なインスタンスが互いに分離して保持される必要があるアプリケーションに推奨されます。詳細については、「スプレッドプレイスメントグループ」を参照してください。	2017 年 11 月 29 日
スポットインスタンスの休止状態	2016-11-15	スポットサービスは、中断が発生した場合スポットインスタンスを休止させることができます。	2017 年 11 月 28 日

機能	API バージョン	説明	リリース日
スポットフリートのターゲット追跡	2016-11-15	スポットフリートのターゲット追跡スケーリングポリシーを設定できます。詳細については、「ターゲット追跡スケーリング: 特定のメトリクスの値を対象にしてスポットフリートをスケールする」を参照してください。	2017 年 11 月 17 日
スポットフリートは Elastic Load Balancing と統合されます。	2016-11-15	スポットフリートに 1 つ以上のロードバランサーをアタッチできます。	2017 年 11 月 10 日
コンバーティブルリザーブドインスタンス のマージと分割	2016-11-15	2 つ以上の コンバーティブルリザーブドインスタンス を新しい コンバーティブルリザーブドインスタンス に交換 (マージ) することができます。変更プロセスを使って、コンバーティブルリザーブドインスタンス をより小さな予約に分割することもできます。詳細については、「コンバーティブルリザーブドインスタンスの交換」を参照してください。	2017 年 11 月 6 日
VPC のテナント属性を変更する	2016-11-15	VPC インスタンスのテナント属性を `dedicated` から `default` に変更することができます。詳細については、「VPC のインスタンステナンシーを変更する」を参照してください。	2017 年 10 月 16 日
1 秒単位の請求	2016-11-15	Amazon EC2 は、Linux ベースの秒単位での課金 (最低 1 分間分) を行います。	2017 年 10 月 2 日
中断時に停止	2016-11-15	中断時に Amazon EC2 が スポットインスタンス を停止または終了するかを指定できます。詳細については、「スポットインスタンスの中断の動作」を参照してください。	2017 年 9 月 18 日

機能	API バージョン	説明	リリース日
NAT ゲートウェイのタグ付け	2016-11-15	NAT ゲートウェイにタグを付けることができます。詳細については、「リソースのタグ付け」を参照してください。	2017 年 9 月 7 日
セキュリティグループルールの説明	2016-11-15	説明をセキュリティグループに追加できます。	2017 年 8 月 31 日
Elastic Graphics	2016-11-15	インスタンスに Elastic Graphics アクセラレーターをアタッチすると、アプリケーションのグラフィック性能が向上します。	2017 年 8 月 29 日
Elastic IP アドレスの復元	2016-11-15	VPC で使用するために Elastic IP アドレスを解放した場合、復元できる可能性があります。	2017 年 8 月 11 日
スポットフリートのインスタンスにタグを付けます。	2016-11-15	起動するインスタンスに自動的にタグを付けるように、スポットフリートを設定できます。	2017 年 7 月 24 日
リソース作成時のタグ付け	2016-11-15	インスタンスやボリュームの作成時にタグを適用できます。詳細については、「リソースのタグ付け」を参照してください。さらに、タグベースのリソースレベルアクセス権限を使用して、適用されているタグを制御できます。詳細については、「Amazon EC2 リソース作成時にタグ付けするアクセス許可の付与」を参照してください。	2017 年 3 月 28 日
アタッチされた EBS ボリュームで変更を行う	2016-11-15	ほとんどの EC2 インスタンスにアタッチされたほとんどの EBS ボリュームでは、ボリュームをデタッチしたりインスタンスを停止したりせずに、ボリュームのサイズ、タイプ、IOPS を変更できます。	2017 年 2 月 13 日

機能	API バージョン	説明	リリース日
IAM ロールをアタッチする	2016-11-15	既存のインスタンスの IAM ロールをアタッチ、デタッチ、または置換できます。詳細については、「Amazon EC2 の IAM ロール」を参照してください。	2017 年 2 月 9 日
専有 スポットインスタンス	2016-11-15	Virtual Private Cloud (VPC) のシングルテナントハードウェアで、スポットインスタンス を実行できます。詳細については、「シングルテナントハードウェアで起動する」を参照してください。	2017 年 1 月 19 日
IPv6 サポート	2016-11-15	VPC とサブネットに IPv6 CIDR を関連付け、VPC のインスタンスに IPv6 アドレスを割り当てることができます。詳細については、「Amazon EC2 インスタンスの IP アドレス指定」を参照してください。	2016 年 12 月 1 日
スポットフリートの自動スケーリング		スポットフリートのスケーリングポリシーを設定できるようになりました。詳細については、「スポットフリートの自動スケーリングについて」を参照してください。	2016 年 9 月 1 日
Elastic Network Adapter (ENA)	2016-04-01	ENA を使用してネットワーキングを強化できるようになりました。詳細については、「Amazon EC2 インスタンスでの拡張ネットワーキング」を参照してください。	2016 年 6 月 28 日
長い ID の表示および変更の拡張サポート	2016-04-01	他の IAM ユーザー、IAM ロール、ルートユーザーの長い ID 設定を表示および変更できるようになりました。	2016 年 6 月 23 日
暗号化された Amazon EBS スナップショットと AWS アカウント間のコピー	2016-04-01	AWS アカウント間で、暗号化された EBS スナップショットをコピーできるようになりました。	2016 年 6 月 21 日

機能	API バージョン	説明	リリース日
インスタンスコンソールのスクリーンショットの取得	2015-10-01	到達不可のインスタンスをデバッグするときに、追加の情報を取得できるようになりました。詳細については、「接続できないインスタンスのスクリーンショットの取得」を参照してください。	2016 年 5 月 24 日
新しい 2 タイプの EBS ボリューム	2015-10-01	スループット最適化 HDD (st1) と Cold HDD (sc1) ボリュームを作成できるようになりました。	2016 年 4 月 19 日
Amazon EC2 用の新しい NetworkPacketsIn と NetworkPacketsOut のメトリクスを追加しました。		Amazon EC2 用の新しい NetworkPacketsIn と NetworkPacketsOut のメトリクスを追加しました。詳細については、「インスタンスメトリクス」を参照してください。	2016 年 3 月 23 日
スポットフリートの CloudWatch メトリクス		スポットフリートの CloudWatch メトリクスを取得できるようになりました。詳細については、「CloudWatch を使用して EC2 フリートまたはスポットフリートをモニタリングする」を参照してください。	2016 年 3 月 21 日
スケジュールされたインスタンス	2015-10-01	スケジュールされたリザーブドインスタンス (スケジュールされたインスタンス) によって、毎日、毎週、または毎月ベースの指定された開始時間および期間で繰り返しキャパシティー予約を購入できます。	2016 年 1 月 13 日
長いリソース ID	2015-10-01	一部の Amazon EC2 および Amazon EBS リソースタイプに、段階的に長い ID を導入しています。オプトイン期間中に、サポートされるリソースタイプに対して長い ID 形式を有効にできます。	2016 年 1 月 13 日

機能	API バージョン	説明	リリース日
ClassicLink DNS サポート	2015-10-01	VPC の ClassicLink DNS サポートを有効にして、リンクされた EC2-Classic インスタンスと VPC のインスタンス間で対応された DNS ホスト名がプライベート IP アドレスに解決され、パブリック IP アドレスに解決されないようにします。	2016 年 1 月 11 日
Dedicated Host	2015-10-01	Amazon EC2 Dedicated Host は、インスタンス容量を利用したお客様専用の物理サーバーです。詳細については、「Amazon EC2 専有ホスト」を参照してください。	2015 年 11 月 23 日
スポットインスタンスの継続時間	2015-10-01	スポットインスタンス で実行時間を指定できるようになりました。スポットブロックはサポートされていません (2023 年 1 月)。	2015 年 10 月 6 日
スポットフリートの変更リクエスト	2015-10-01	スポットフリートリクエストのターゲット容量が変更できるようになりました。詳細については、「スポットフリートリクエストを変更します。」を参照してください。	2015 年 9 月 29 日
スポットフリートの分散配分戦略	2015-04-15	単一のスポットフリートリクエストを使用して、複数のスポットプールにスポットインスタンスを分散することができるようになりました。詳細については、「配分戦略を使用して、EC2 フリートまたはスポットフリートがどのようにスポットキャパシティとオンデマンドキャパシティを満たすかを決定します。」を参照してください。	2015 年 9 月 15 日

機能	API バージョン	説明	リリース日
スポットフリートインスタンスの分量指定	2015-04-15	アプリケーションのパフォーマンスに影響する各インスタンスのキャパシティーユニットを定義し、スポットプールごとに スポットインスタンス の支払料金を調整することができるようになりました。詳細については、「インスタンスの重み付けを使用して、EC2 フリートまたはスポットフリートのコストとパフォーマンスを管理する」を参照してください。	2015 年 8 月 31 日
新しい再起動アラームアクションと、アラームアクションで使用する新しい IAM ロール		再起動アラームアクションと、アラームアクションで使用する新しい IAM ロールが追加されました。詳細については、「インスタンスを停止、終了、再起動、または復旧するアラームを作成する」を参照してください。	2015 年 7 月 23 日
Spot Fleets	2015-04-15	個別のスポットインスタンスリクエストを管理する代わりに、スポットインスタンスの集合またはフリートを管理できます。詳細については、「EC2 フリートとスポットフリート」を参照してください。	2015 年 5 月 18 日
Elastic IP アドレスを EC2-Classic に移行する	2015-04-15	EC2-Classic で使用するために割り当てた Elastic IP アドレスを、VPC で使用するために移行できます。	2015 年 5 月 15 日
複数のディスクで構成された VM を AMI としてインポート	2015-03-01	VM Import プロセスにより、複数のディスクで構成された VM を AMI としてインポートできるようになりました。詳細については、VM Import/Export ユーザーガイドの「VM Import/Export を使用してイメージとして VM をインポート」を参照してください。	2015 年 4 月 23 日
Systems Manager		Systems Manager を使用すると、EC2 インスタンスを設定して管理できます。	2015 年 2 月 17 日

機能	API バージョン	説明	リリース日
Systems Manager for Microsoft SCVMM 1.5		Systems Manager for Microsoft SCVMM を使用して、インスタンスを起動し、SCVMM から Amazon EC2 に VM をインポートできるようになりました。	2015 年 1 月 21 日
EC2 インスタンスの自動復旧		Amazon EC2 インスタンスをモニタリングする Amazon CloudWatch アラームを作成できます。この機能により、基盤ハードウェアの障害や、AWS による修復を必要とする問題によりインスタンスが正常に機能しなくなった場合に、自動的にそのインスタンスを復旧できます。復旧されたインスタンスは、インスタンス ID、IP アドレス、すべてのインスタンスメタデータを含め、元のインスタンスと同じです。 詳細については、「インスタンスの自動復旧」を参照してください。	2015 年 1 月 12 日
ClassicLink	2014-10-01	ClassicLink を使用すると、EC2-Classic インスタンスをアカウント内の VPC にリンクできます。これによって、VPC のセキュリティグループを EC2-Classic インスタンスに関連付け、プライベート IP アドレスを使用して EC2-Classic インスタンスと VPC 内のインスタンスが通信できるようになります。	2015 年 1 月 7 日
スポットインスタンスの終了通知		スポットインスタンスの中断から保護する最善の方法は、アプリケーションを耐障害性のある設計にすることです。さらに、スポットインスタンスの終了通知機能を活用できます。これは Amazon EC2 がスポットインスタンスを終了する 2 分前に警告を発します。 詳細については、「スポットインスタンスの中断通知」を参照してください。	2015 年 1 月 5 日

機能	API バージョン	説明	リリース日
Systems Manager for Microsoft SCVMM		Systems Manager for Microsoft SCVMM は、Microsoft SCVMM から EC2 インスタンスなどの AWS リソースを管理する、シンプルで使いやすいインターフェイスを提供します。	2014 年 10 月 29 日
`DescribeVolumes` ページ分割サポート	2014-09-01	`DescribeVolumes` API 呼び出しで、`MaxResults` パラメータと `NextToken` パラメータによる結果のページ分割がサポートされるようになりました。詳細については、Amazon EC2 API Reference の「DescribeVolumes」を参照してください。	2014 年 10 月 23 日
Amazon CloudWatch Logs のサポートの追加		Amazon CloudWatch Logs を使用して、インスタンスまたはその他のソースのシステム、アプリケーション、およびカスタムログファイルの監視、保存、アクセスができます。その後、Amazon CloudWatch コンソール、AWS CLI の CloudWatch Logs コマンド、または CloudWatch Logs SDK を使用して、関連するログデータを CloudWatch Logs から取得できます。	2014 年 7 月 10 日
新しい [EC2 Service Limits] ページ		Amazon EC2 コンソールの [EC2 Service Limits] ページを使用して、Amazon EC2 や Amazon VPC から提供されるリソースに対するリージョンごとの現在の制限を表示できます。	2014 年 6 月 19 日

機能	API バージョン	説明	リリース日
Amazon EBS 汎用 SSD ボリューム	2014-05-01	汎用 SSD ボリュームは、さまざまなワークロードに対応できるコスト効率の高いストレージとして使用できます。これらのボリュームは、1 桁ミリ秒のレイテンシー、長時間にわたる 3,000 IOPS へのバースト機能、最大 3 IOPS/GiB のベースパフォーマンスを実現しています。汎用 SSD ボリュームのサイズ範囲は、1 GiB ~ 1 TiB です。	2014 年 6 月 16 日
AWS マネジメントパック		AWS マネジメントパックが System Center Operations Manager 2012 R2 をサポートするようになりました。	2014 年 5 月 22 日
Amazon EBS encryption	2014-05-01	Amazon EBS 暗号化 により、EBS データボリュームとスナップショットがシームレスに暗号化されるため、セキュアキー管理インフラストラクチャを構築および維持する必要がなくなります。EBS 暗号化サービスは、AWS マネージドキー を使用してデータを暗号化することにより、保管中のデータのセキュリティを確保します。EC2 インスタンスをホストするサーバーで暗号化が行われるため、EC2 インスタンスと EBS ストレージとの間を移動するデータが暗号化されます。	2014 年 5 月 21 日
Amazon EC2 使用状況レポート		Amazon EC2 使用状況レポートは、EC2 の使用コストと使用状況データを示す一連のレポートです。	2014 年 1 月 28 日
Linux 仮想マシンのインポート	2013-10-15	VM Import プロセスが、Linux インスタンスのインポートをサポートするようになりました。詳細については、「VM Import/Export ユーザーガイド」を参照してください。	2013 年 12 月 16 日

機能	API バージョン	説明	リリース日
RunInstances に関するリソースレベルの許可	2013-10-15	AWS Identity and Access Management でポリシーを作成して、Amazon EC2 RunInstances API アクションについてリソースレベルでアクセス許可を制御できるようになりました。詳細とポリシー例については、「Amazon EC2 の Identity and Access Management」を参照してください。	2013 年 11 月 20 日
AWS Marketplace からのインスタンスの起動		Amazon EC2 Launch Wizard を使用して、AWS Marketplace からインスタンスを起動できるようになりました。詳細については、「AWS Marketplace AMI から Amazon EC2 インスタンスを起動する」を参照してください。	2013 年 11 月 11 日
新しいlaunch wizard		設計し直された新しい EC2 Launch Wizard が付属します。詳細については、「コンソールのインスタンス起動ウィザードを使用して EC2 インスタンスを起動する」を参照してください。	2013 年 10 月 10 日
リザーブドインスタンスのインスタンスタイプの変更	2013-10-01	同一ファミリー内 (例えば、M1、M2、M3、C1) であれば Linux リザーブドインスタンスのインスタンスタイプを変更できるようになりました。詳細については、「リザーブドインスタンス の変更」を参照してください。	2013 年 10 月 09 日
Amazon EC2 リザーブドインスタンスの変更	2013-08-15	リージョンのリザーブドインスタンスを変更できるようになりました。詳細については、「リザーブドインスタンス の変更」を参照してください。	2013 年 9 月 11 日

機能	API バージョン	説明	リリース日
パブリック IP アドレスの割り当て	2013-07-15	VPC でインスタンスを起動するときに、パブリック IP アドレスを割り当てることができるようになりました。詳細については、「起動時にパブリック IPv4 アドレスを割り当てる」を参照してください。	2013 年 8 月 20 日
リソースレベルのアクセス許可の付与	2013-06-15	Amazon EC2 は、新しい Amazon Resource Name (ARN) および条件キーをサポートします。詳細については、「Amazon EC2 のアイデンティティベースのポリシー」を参照してください。	2013 年 7 月 8 日
インクリメンタルスナップショットコピー	2013-02-01	インクリメンタルスナップショットコピーを実行できるようになりました。	2013 年 6 月 11 日
AWS マネジメントパック		AWS マネジメントパックは、Amazon EC2 インスタンスと、その中で動作する Windows または Linux OS をリンクします。AWS マネジメントパックは Microsoft System Center Operations Manager 向けの拡張パックです。	2013 年 5 月 8 日
新しい [Tags] ページ		Amazon EC2 コンソールに新しい [Tags] ページを追加しました。詳細については、「Amazon EC2 リソースのタグ付け」を参照してください。	2013 年 04 月 4 日
リージョン間での AMI のコピー	2013-02-01	AMI をリージョン間でコピーして、整合性のあるインスタンスを、複数の AWS リージョンですばやく簡単に起動できます。 詳細については、「Amazon EC2 AMI のコピー」を参照してください。	2013 年 3 月 11 日

機能	API バージョン	説明	リリース日
デフォルトの VPC へのインスタンスの起動	2013-02-01	AWS アカウントは、リージョンごとに、EC2-Classic と VPC のいずれか、または VPC のみにインスタンスを起動できます。インスタンスを起動できるのが VPC だけである場合は、デフォルトの VPC が自動的に作成されます。お客様がインスタンスを起動するときは、デフォルトの VPC で起動されるようになります。ただし、お客様が非デフォルト VPC を作成してその VPC でインスタンスを起動するよう指定した場合を除きます。	2013 年 3 月 11 日
EBS スナップショットのコピー	2012-12-01	スナップショットのコピーを使用して、データのバックアップを作成したり、新しい Amazon EBS ボリュームを作成したり、Amazon マシンイメージ (AMI) を作成したりすることができます。	2012 年 12 月 17 日
Provisioned IOPS SSD ボリューム用の EBS メトリクスおよびステータスチェックの更新	2012-10-01	Provisioned IOPS SSD ボリュームの 2 つの新しいメトリクスを含むように EBS メトリクスが更新されました。また、Provisioned IOPS SSD ボリューム用の新しいステータスチェックを追加しました。	2012 年 11 月 20 日
スポットインスタンスリクエストステータス	2012-10-01	スポットインスタンスリクエストステータスにより、スポットリクエストの状態を簡単に確認できます。	2012 年 10 月 14 日

機能	API バージョン	説明	リリース日
Amazon EC2 リザーブドインスタンス Marketplace	2012-08-15	リザーブドインスタンス Marketplace は、不要になった Amazon EC2 リザーブドインスタンスを持つ販売者と、追加の容量の購入を検討する購入者をマッチングします。リザーブドインスタンス Marketplace を通じて売買されたリザーブドインスタンスは、他のリザーブドインスタンス同様に動作します。ただし、完全な標準期間より残りの期間が短く、販売価格が異なる可能性がある点が違います。	2012 年 9 月 11 日
Amazon EBS のプロビジョンド IOPS SSD	2012-07-20	Provisioned IOPS SSD ボリュームは、整合性と応答時間の短さが重要な、データベースアプリケーションなど I/O を多用する作業負荷に対して、予測可能なハイパフォーマンスを提供します。	2012 年 7 月 31 日
Amazon EC2 インスタンスの IAM ロール	2012-06-01	Amazon EC2 向けの IAM ロールは次の機能を提供します。 • Amazon EC2 インスタンスで実行中のアプリケーション用の AWS アクセスキー。 • Amazon EC2 インスタンスの AWS アクセスキーの自動ローテーション。 • Amazon EC2 インスタンスで実行中の、AWS サービスにリクエストを送信しているアプリケーションに対するきめの細かい権限管理。	2012 年 6 月 11 日

機能	API バージョン	説明	リリース日
スポットインスタンスの特徴は、開始や中断の可能性への対処を容易にすることです。		次のように、スポットインスタンス を管理できるようになりました。 • Auto Scaling 起動設定を使用して スポットインスタンス に支払う金額を指定し、スポットインスタンス に支払う金額を指定するスケジュールを設定します。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「スポットインスタンスのリクエスト」を参照してください。 • インスタンスの起動または終了の通知を受け取ることができます。 • CloudFormation テンプレートを使用して、AWS リソースのスタック内のスポットインスタンスを起動します。	2012 年 6 月 7 日
Amazon EC2 のステータスチェックのための EC2 インスタンスのエクスポートとタイムスタンプ	2012-05-01	当初 EC2 にインポートした Windows Server インスタンスのエクスポートのサポートを追加しました。 ステータスチェックが失敗した日時を示す、インスタンスステータスとシステムステータスのタイムスタンプのサポートを追加しました。	2012 年 5 月 25 日
EC2 インスタンスのエクスポート、および Amazon VPC に対するインスタンスとシステムのステータスチェックのタイムスタンプ	2012-05-01	Citrix Xen、Microsoft Hyper-V、および VMware vSphere 向けの EC2 インスタンスのエクスポートのサポートを追加しました。 インスタンスとシステムのステータスチェックのタイムスタンプのサポートを追加しました。	2012 年 5 月 25 日

機能	API バージョン	説明	リリース日
AWS Marketplace AMI	2012-04-01	AWS Marketplace AMI のサポートが追加されました。	2012 年 4 月 19 日
リザーブドインスタンス料金範囲	2011-12-15	リザーブドインスタンス料金範囲に組み込まれた割引料金の利用方法に関する新しいセクションを追加しました。	2012 年 3 月 5 日
Amazon Virtual Private Cloud での EC2 インスタンスのための エラスティックネットワークインターフェイスs (ENI)	2011-12-01	VPC での EC2 インスタンスのための エラスティックネットワークインターフェイスs (ENI) についての新しいセクションを追加しました。詳細については、「Elastic Network Interface」を参照してください。	2011 年 12 月 21 日
新しい Amazon EC2 リザーブドインスタンスの提供タイプ	2011-11-01	インスタンスの使用目的に応じて、さまざまなリザーブドインスタンス提供タイプを選択できるようになりました。	2011 年 12 月 1 日
Amazon EC2 インスタンスのステータス	2011-11-01	AWS で予定されている、インスタンスに影響を及ぼす可能性のあるイベントなど、インスタンスのステータスについて追加の詳細情報を表示できます。これら運用上の作業には、ソフトウェアアップデートやセキュリティパッチを適用するために必要なインスタンスの再起動や、ハードウェアに問題が生じた場合に必要となるインスタンスの廃棄などが含まれます。詳細については、「Amazon EC2 インスタンスのステータスのモニタリング」を参照してください。	2011 年 11 月 16 日

機能	API バージョン	説明	リリース日
Amazon VPC の スポットインスタンス	2011-07-15	Amazon VPC での スポットインスタンス のサポートについての情報を追加しました。この更新により、ユーザーは Virtual Private Cloud (VPC) で スポットインスタンス を起動できます。スポットインスタンス のユーザーは スポットインスタンス を起動することで、Amazon VPC の利点を得ることができます。	2011 年 10 月 11 日
CLI ツールのユーザーのための VM Import 処理の簡素化	2011-07-15	VM Import 処理を簡素化し、`ImportInstance` と `ImportVolume` の機能を拡張しました。これにより、インポートタスクの作成後に、イメージが Amazon EC2 にアップロードされます。さらに、`ResumeImport` の導入により、アップロードが途中で止まった場合にその個所から再開できるようになりました。	2011 年 9 月 15 日
VHD ファイル形式でのインポートのサポート		VM Import で、仮想マシンイメージファイルを VHD 形式でインポートできるようになりました。VHD ファイル形式は、Citrix Xen および Microsoft Hyper-V 仮想化プラットフォームと互換性があります。VM Import はこのリリースで、RAW、VHD、および VMDK (VMware ESX 互換) イメージ形式をサポートしています。詳細については、「VM Import/Export ユーザーガイド」を参照してください。	2011 年 8 月 24 日
VMware vCenter 用の Amazon EC2 VM Import Connector のアップデート		Amazon EC2 VM Import Connector for VMware vCenter 仮想アプライアンス (Connector) の1.1 バージョンについての情報を追加しました。このアップデートには、インターネットアクセスのためのプロキシサポート、エラー処理の向上、タスクプログレスバーの精度向上、および数件のバグ修正が含まれます。	2011 年 6 月 27 日

機能	API バージョン	説明	リリース日
スポットインスタンス のアベイラビリティーゾーンの料金変更	2011-05-15	スポットインスタンス のアベイラビリティーゾーンの料金機能についての情報を追加しました。このリリースでは、スポットインスタンスリクエストおよびスポット料金履歴のクエリを実行したときに返される情報の一部として、新しいアベイラビリティーゾーンの料金オプションが追加されました。これにより、特定のアベイラビリティーゾーンでスポットインスタンスを起動するために必要な料金を判断しやすくなりました。	2011 年 5 月 26 日
AWS Identity and Access Management		AWS Identity and Access Management (IAM) に関する情報を追加しました。ユーザーが Amazon EC2 リソースで通常使用できる Amazon EC2 アクションを、指定できるようにします。詳細については、「Amazon EC2 の Identity and Access Management」を参照してください。	2011 年 4 月 26 日
専用インスタンス		専用インスタンスは、ホストのハードウェアレベルで物理的に隔離されているインスタンスであり、Amazon Virtual Private Cloud (Amazon VPC) 内で起動します。ハードウェア専有インスタンスを使うと、Amazon VPC と AWS クラウドの利点を活かすことができます。このインスタンスでは、Amazon EC2 のコンピューティングインスタンスをハードウェアレベルで隔離しながら、伸縮自在なオンデマンドでのプロビジョニングや、従量課金の料金システムをご利用いただけます。詳細については、「Amazon EC2 ハードウェア専有インスタンス」を参照してください。	2011 年 3 月 27 日

機能	API バージョン	説明	リリース日
リザーブドインスタンスに関する AWS マネジメントコンソールの更新		AWS マネジメントコンソールが更新されました。これにより、リザーブドインスタンスの表示や、ハードウェア専有リザーブドインスタンスを含む追加のリザーブドインスタンスの購入が、より簡単に行えるようになりました。	2011 年 3 月 27 日
メタデータ情報	2011-01-01	2011-01-01 リリースでの変更を反映するため、メタデータについての情報を追加しました。詳細については、「インスタンスメタデータを使用して EC2 インスタンスを管理する」および「インスタンスメタデータのカテゴリ」を参照してください。	2011 年 3 月 11 日
Amazon EC2 VM Import Connector for VMware vCenter		Amazon EC2 VM Import Connector for VMware vCenter 仮想アプライアンス (Connector) についての情報を追加しました。このコネクタは、VMware vSphere Client と統合するための、VMware vCenter のプラグインです。コネクタの GUI を使用して、VMware 仮想マシンを Amazon EC2 にインポートすることができます。	2011 年 3 月 3 日
ボリュームの強制デタッチ		AWS マネジメントコンソール を使用して、インスタンスから Amazon EBS ボリュームを強制的にデタッチできるようになりました。	2011 年 2 月 23 日
インスタンス終了の防止		AWS マネジメントコンソール を使用して、インスタンスの削除を防止できるようになりました。詳細については、「インスタンスの終了保護を変更する」を参照してください。	2011 年 2 月 23 日
VM Import	2010-11-15	仮想マシンまたはボリュームを Amazon EC2 にインポートする VM Import についての情報を追加しました。詳細については、「VM Import/Export ユーザーガイド」を参照してください。	2010 年 12 月 15 日

機能	API バージョン	説明	リリース日
インスタンスの基本モニタリング	2010-08-31	EC2 インスタンスの基本モニタリングについての情報を追加しました。	2010 年 12 月 12 日
フィルタとタグ	2010-08-31	リソースの一覧表示、フィルタリング、およびタグ付けについての情報を追加しました。詳細については、「Amazon EC2 リソースの検索」および「Amazon EC2 リソースのタグ付け」を参照してください。	2010 年 9 月 19 日
インスタンス起動時の多重実行禁止	2010-08-31	インスタンスを実行する際に多重実行を禁止する方法についての情報を追加しました。	2010 年 9 月 19 日
Amazon EC2 用の AWS Identity and Access Management		Amazon EC2 が AWS Identity and Access Management (IAM) と統合されるようになりました。詳細については、「Amazon EC2 の Identity and Access Management」を参照してください。	2010 年 9 月 2 日
Amazon VPC IP アドレス指定	2010-06-15	Amazon VPC ユーザーは、VPC 内で起動されたインスタンスに割り当てる IP アドレスを指定できるようになりました。	2010 年 7 月 12 日
Amazon CloudWatch Amazon EBS ボリュームのモニタリング		Amazon EBS ボリュームに対する Amazon CloudWatch モニタリングが自動的に利用できるようになりました。	2010 年 6 月 14 日
Windows でのリザーブドインスタンス		Amazon EC2 は Windows でリザーブドインスタンスをサポートするようになりました。	2010 年 2 月 22 日