aws

ユーザーガイド

Amazon EC2 Auto Scaling

Amazon EC2 Auto Scaling: ユーザーガイド

Table of Contents

Amazon EC2 Auto Scaling とは 1
Amazon EC2 Auto Scaling の機能 1
Amazon EC2 Auto Scaling の料金 3
使用を開始する 3
Auto Scaling グループの使用 4
Auto Scaling の利点 5
例: 変化する需要に対応する 5
例: ウェブアプリアーキテクチャ 7
例: 複数のアベイラビリティーゾーン全体にインスタンスを分散させる 9
インスタンスのライフサイクル 12
スケールアウト 13
インスタンスが実行中 13
スケールイン 14
インスタンスのデタッチ 15
インスタンスのアタッチ 15
ライフサイクルフック 15
スタンバイを入力し終了します。 16
Amazon EC2 Auto Scaling クォータ 16
Amazon EC2 Auto Scaling API のリクエストスロットリング 18
EC2 終了レート 18
その他のサービス 19
セットアップする 20
Amazon EC2 を使用するための準備を整える 20
を使用するための準備 AWS CLI 20
使用を開始する 21
チュートリアル: 最初の Auto Scaling グループを作成する 22
チュートリアルのための準備 22
ステップ 1: 起動テンプレートを作成する 23
ステップ 2: 単一インスタンスの Auto Scaling グループを作成する 24
ステップ 3: Auto Scaling グループを検証する 25
ステップ 4: Auto Scaling グループのインスタンスを終了します 26
ステップ 5: 次のステップ 27
ステップ 6: クリーンアップする 27

チュートリアル: スケーリングとロードバランシングを使用するアプリケーションのセットアップ ... 28
前提条件 ... 30
ステップ 1: 起動テンプレートまたは起動設定を設定する ... 31
ステップ 2: Auto Scaling グループを作成する ... 34
ステップ 3: ロードバランサーがアタッチされたことを確認する ... 36
ステップ 4: 次のステップ ... 36
ステップ 5 : クリーンアップ ... 37
関連リソース ... 38
Amazon EC2 Auto Scaling 起動テンプレート ... 40
起動テンプレートを操作するアクセス許可 ... 41
起動テンプレートでサポートされている API オペレーション ... 41
Auto Scaling グループの起動テンプレートを作成する ... 41
起動テンプレートを作成する (コンソール) ... 42
デフォルトのネットワークインターフェイス設定を変更する (コンソール) ... 45
ストレージ設定を変更する (コンソール) ... 47
既存のインスタンスから起動テンプレートを作成する (コンソール) ... 50
関連リソース ... 51
制限事項 ... 51
詳細設定を使用して起動テンプレートを作成する ... 51
必須の設定 ... 52
[詳細設定] ... 52
スポットインスタンスをリクエストする ... 57
Capacity Blocks ML 用 ... 59
Auto Scaling グループを起動テンプレートに移行する ... 64
ステップ 1: 起動設定を使用する Auto Scaling グループを検索する ... 64
ステップ 2: 起動設定を起動テンプレートにコピーする ... 67
ステップ 3: 起動テンプレートを使用するように Auto Scaling グループを更新する ... 68
ステップ 4: インスタンスを置き換える ... 69
追加情報 ... 69
CloudFormation スタックを起動テンプレートに移行する ... 69
起動設定を使用している Auto Scaling グループを検索する ... 70
起動テンプレートを使用するようにスタックを更新する ... 71
スタックリソースの更新時の動作 ... 75
移行を追跡する ... 75
起動設定のマッピングリファレンス ... 76

AWS CLI 起動テンプレートの使用例 77
使用例 78
基本的な起動テンプレートを作成する 79
起動時にインスタンスにタグ付けするタグを指定する 79
インスタンスに渡す IAM ロールを指定する 80
パブリック IP アドレスを割り当てる 80
起動時にインスタンスを設定するユーザーデータスクリプトを指定する 81
ブロックデバイスマッピングを指定する 81
外部ベンダーからソフトウェアライセンスを取得するための Dedicated Hosts を指定する 81
既存のネットワークインターフェイスを指定する 82
複数のネットワークインターフェイスを作成する 82
起動テンプレートを管理する 83
起動テンプレートを使用するように Auto Scaling グループを更新する 86
AMI IDs 86
AMI のパラメータを指定する起動テンプレートを作成する 87
起動テンプレートが正しい AMI ID を取得することを確認する 92
関連リソース 93
制限事項 93
起動設定 95
「起動設定を作成する」 95
「起動設定を作成する」 96
IMDS を設定する 99
EC2 インスタンスを使用した起動設定の作成 101
起動設定を変更する 106
「Auto Scaling グループ」 107
起動テンプレートを使用して Auto Scaling グループを作成する 108
起動テンプレートを使用してグループを作成する 109
EC2 起動ウィザードを使用してグループを作成する 112
複数のインスタンスタイプと購入オプションを使用する 116
起動設定を使用して Auto Scaling グループを作成する 162
起動設定を使用してグループを作成する 163
EC2 インスタンスを使用してグループを作成する 166
Auto Scaling グループを更新する 172
Auto Scaling インスタンスの更新 173
グループとインスタンスへのタグ付け 174

タグの命名と使用制限 ... 175
EC2 インスタンスのタグ付けライフサイクル ... 175
Auto Scaling グループにタグを付ける ... 176
タグの削除 ... 179
セキュリティ用のタグ ... 180
タグへのアクセスを制御する ... 181
タグを使用して Auto Scaling グループをフィルタリングする ... 182
インスタンスのメンテナンスポリシー ... 186
概要 ... 186
グループにインスタンスメンテナンスポリシーを設定する ... 193
ライフサイクルフック ... 198
ライフサイクルフックの可用性 ... 199
考慮事項と制約事項 ... 200
関連リソース ... 201
ライフサイクルフックの動作 ... 202
ライフサイクルフックを追加するための準備 ... 204
ターゲットライフサイクル状態を取得する ... 212
ライフサイクルフックを追加する ... 214
ライフサイクルアクションを完了する ... 218
チュートリアル: インスタンスメタデータを使用してターゲットライフサイクル状態を取得するようにユーザーデータを設定する ... 220
チュートリアル:Lambda 関数を呼び出すライフサイクルフックの設定 ... 228
ウォームプール ... 238
主要概念 ... 238
前提条件 ... 241
ウォームプール内のインスタンスを更新する ... 242
関連リソース ... 242
制限事項 ... 243
ライフサイクルフックを使用する ... 244
Auto Scaling グループのためにウォームプールを作成する ... 247
ヘルスチェックステータスを表示する ... 249
AWS CLI ウォームプールの使用例 ... 253
インスタンスのデタッチ ... 255
インスタンスのデタッチに関する考慮事項 ... 256
インスタンスをアタッチする際の考慮事項 ... 257
デタッチとアタッチを使用してインスタンスを別のグループに移動する ... 258

インスタンスを一時的に削除する ... 263
スタンバイ状態の仕組み ... 264
考慮事項 ... 264
スタンバイ状態のインスタンスのヘルスステータス ... 265
インスタンスをスタンバイに設定して一時的に削除する ... 264
Auto Scaling インフラストラクチャを削除する ... 270
Auto Scaling グループの削除 ... 270
(オプション) 起動設定の削除 ... 271
(オプション) 起動テンプレートの削除 ... 272
(オプション) ロードバランサーとターゲットグループの削除 ... 272
(オプション) CloudWatch アラームの削除 ... 273
AWS Auto Scaling グループを操作するための SDK の例 ... 274
Auto Scaling グループを作成する ... 274
Auto Scaling グループを更新する ... 290
Auto Scaling グループの説明 ... 301
Auto Scaling グループを削除する ... 315
インスタンスをリサイクルする ... 329
インスタンスの更新 ... 329
インスタンスの更新の仕組み ... 330
デフォルト値について説明する ... 336
インスタンスの更新の開始 ... 339
インスタンスの更新をモニタリングする ... 351
インスタンスの更新のキャンセル ... 354
ロールバックで変更の取り消し ... 355
スキップマッチングの使用 ... 360
チェックポイントの追加 ... 369
最大インスタンス有効期間 ... 375
考慮事項 ... 375
最大インスタンスライフタイムを設定する ... 376
制限事項 ... 377
グループをスケールする ... 379
スケーリング方法を選択する ... 379
スケーリング制限を設定する ... 381
デフォルトのインスタンスウォームアップを設定する ... 383
パフォーマンスのスケーリングに関する考慮事項 ... 383
デフォルトのインスタンスウォームアップ時間を選択する ... 384

グループに対するインスタンスのデフォルトウォームアップを有効にする 385
グループに対するインスタンスのデフォルトウォームアップを検証する 387
以前にインスタンスのウォームアップ時間を設定してスケーリングポリシーを検索する 388
以前に設定したスケーリングポリシーのインスタンスウォームアップをクリアする 389
手動スケーリング 390
Auto Scaling グループの希望する容量を変更する 390
Auto Scaling グループのインスタンスを終了する (AWS CLI) 394
スケジュールされたスケーリング 395
スケジュールされたスケーリングのしくみ 396
繰り返しのスケジュール 396
Time zone (タイムゾーン) 397
考慮事項 398
スケジュールされたアクションの作成 398
スケジュールされたアクションの詳細を表示する 400
スケーリングアクティビティを検証する 401
スケジュールされたアクションの削除 402
制限事項 402
動的なスケーリング 403
動的スケーリングポリシーが機能する方法 404
複数の動的スケーリングポリシー 405
ターゲット追跡スケーリングポリシー 406
ステップスケーリングポリシーおよび簡易スケーリングポリシー 420
スケーリングクールダウン 437
Amazon SQS に基づくスケーリング 440
スケーリングアクティビティを検証する 448
スケーリングポリシーを無効化する 450
スケーリングポリシーを削除する 452
AWS CLI スケーリングポリシーの例 455
予測スケーリング 458
予測スケーリングの仕組み 459
予測スケーリングポリシーを作成する 462
予測スケーリングポリシーの評価 470
予測を上書きする 479
カスタムメトリクスを使用する 484
インスタンスの終了を制御する 495
終了ポリシーのシナリオ 496

終了ポリシーを設定する 500
Lambda を使用したカスタム終了ポリシーを作成する 505
インスタンスのスケールイン保護を使用する 512
インスタンスを正常に終了するために設計する 516
プロセスを中断して再開する 520
プロセスのタイプ 520
考慮事項 522
プロセスを中断する 522
プロセスを再開する 523
中断されたプロセスが他のプロセスに与える影響 524
モニター 528
ヘルスチェック 530
ヘルスチェックについて 531
ヘルスチェックの猶予期間を設定する 539
ヘルスチェックが失敗した理由を表示する 541
異常なインスタンスのトラブルシューティング 543
でモニタリングする AWS Health Dashboard 546
CloudWatch メトリクスのモニタリング 548
Amazon EC2 Auto Scaling コンソールでモニタリンググラフを表示する 548
CloudWatch Amazon EC2 Auto Scaling の メトリクス 553
Auto Scaling インスタンスのモニタリングを設定する 561
API 呼び出しを以下のように記録します。 AWS CloudTrail 563
の Amazon EC2 Auto Scaling 情報 CloudTrail 564
Amazon EC2 Auto Scaling のログファイルエントリについて 565
関連リソース 566
Amazon SNS 通知オプション 566
Amazon SNS と Amazon EC2 Auto Scaling 567
他のサービスと連携する 574
キャパシティーの再調整 574
概要 575
キャパシティーの再調整の動作 576
考慮事項 577
キャパシティの再調整のを有効にする (コンソール) 579
キャパシティの再調整を有効にする (AWS CLI) 580
関連リソース 585
制限事項 585

キャパシティ予約 585
ステップ 1: キャパシティ予約を作成する 586
ステップ 2: キャパシティ予約グループを作成する 588
ステップ 3: 起動テンプレートを作成する 590
ステップ 4: Auto Scaling グループを作成する 592
関連リソース 594
AWS CloudShell 594
AWS CloudFormation 595
Amazon EC2 Auto Scaling AWS CloudFormation とテンプレート 595
以下についてさらに詳しく AWS CloudFormation 596
Compute Optimizer 596
制限事項 597
結果 597
推奨事項の表示 598
推奨事項の評価に関する考慮事項 599
Elastic Load Balancing 600
Elastic Load Balancing のタイプ 601
ロードバランサーをアタッチする準備をする 602
ロードバランサーをアタッチする 605
Amazon EC2 Auto Scaling コンソールからロードバランサーを設定する 609
アタッチメントステータスを確認する 610
アベイラビリティーゾーンを追加する 611
AWS CLI Elastic Load Balancing の使用例 615
VPC Lattice 623
ターゲットグループをアタッチする準備を行うには 625
VPC Lattice ターゲットグループをアタッチする 628
アタッチメントステータスを確認する 633
EventBridge 634
Amazon EC2 Auto Scaling イベントリファレンス 635
ウォームプールのイベントとパターンの例 646
EventBridge ルールの作成 651
Amazon VPC 657
デフォルト VPC 658
デフォルトではない VPC 658
VPC サブネットを選択する場合の考慮事項 658
VPC での IP アドレス指定 659

VPC 内のネットワークインターフェイス ... 659
インスタンスのプレイスメントテナンシー ... 660
AWS Outposts ... 660
VPC に関するその他のリソース ... 661
セキュリティ ... 662
インフラストラクチャセキュリティ ... 663
関連リソース ... 663
耐障害性 ... 663
関連リソース ... 665
データ保護 ... 665
AWS KMS keys を使用して Amazon EBS ボリュームを暗号化する ... 666
関連リソース ... 667
AWS KMS 暗号化されたボリュームで使用する キーポリシー ... 667
Identity and Access Management ... 673
アクセスコントロール ... 674
Amazon EC2 Auto Scaling と IAM の連携 ... 674
API アクセス許可 ... 684
マネージドポリシー ... 686
サービスリンクロール ... 691
アイデンティティベースポリシーの例 ... 696
サービス間の混乱した代理の防止 ... 705
起動テンプレートのサポート ... 707
Amazon EC2 インスタンスで実行中のアプリケーション用の IAM ロール ... 715
コンプライアンス検証 ... 718
PCI DSS コンプライアンス ... 720
プライベート接続のための VPC エンドポイントを使用する ... 720
インターフェイス VPC エンドポイントを作成する ... 721
VPC エンドポイントポリシーを作成する ... 721
トラブルシューティング ... 723
エラーメッセージを取得する ... 723
スケーリングアクティビティをオフにする ... 725
その他のトラブルシューティングリソース ... 726
インスタンス起動の失敗 ... 727
リクエストされた設定は現在サポートされていません。 ... 728
セキュリティグループ <セキュリティグループ名> は存在しません。EC2 インスタンスの起動に失敗しました。 ... 729

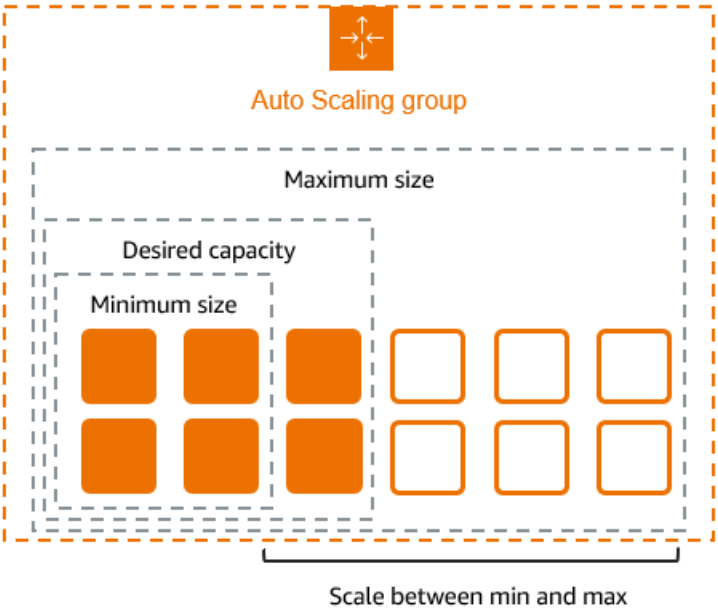
キーペア <お使いの EC2 インスタンスに関連付けられているキーペア> は存在しません。EC2 インスタンスの起動に失敗しました。 729
リクエストされたインスタンスタイプ (<インスタンスタイプ>) は、リクエストされたアベイラビリティーゾーン (<インスタンスのアベイラビリティーゾーン>) ではサポートされません。 730
設定したポットリクエスト料金 0.015 は、スポットリクエストに最低限必要な料金 0.0735 を下回っています...。 730
デバイス名 <device name> が無効です/無効なデバイス名をアップロードしようとしています。EC2 インスタンスの起動に失敗しました。 731
パラメータ virtualName の値 (<インスタンスストレージのデバイスに関連付けられている名前>) は無効です。EC2 インスタンスの起動に失敗しました。 731
EBS ブロックデバイスマッピングはインスタンスストア AMI ではサポートされません。 .. 732
プレイスメントグループは、「<instance type>」タイプのインスタンスでは使用できません。EC2 インスタンスの起動に失敗しました。 732
Client.InternalError: 起動時のクライアントエラー。 732
現在、お客様がリクエストしたアベイラビリティゾーン内には、<instance type> の十分なキャパシティーがありません..。EC2 インスタンスの起動に失敗しました。 734
リクエストされた予約には、このリクエストに十分な互換性と使用可能な容量がありません。EC2 インスタンスの起動に失敗しました。 734
キャパシティブロック予約 <予約 ID> はまだアクティブではありません。EC2 インスタンスの起動に失敗しました。 735
リクエストに適合した、使用可能なスポットキャパシティーはありません。EC2 インスタンスの起動に失敗しました。 735
<インスタンス数> 個のインスタンスがすでに実行中です。EC2 インスタンスの起動に失敗しました。 736
AMI に関する問題 736
AMI ID <お使いの AMI の ID> は存在しません。EC2 インスタンスの起動に失敗しました。 737
AMI <AMI ID> は保留中のため実行できません。EC2 インスタンスの起動に失敗しました。 737
デバイス名 <device name> が無効です。EC2 インスタンスの起動に失敗しました。 738
指定されたインスタンスタイプのアーキテクチャ「arm64」が、指定された AMI のアーキテクチャ「x86_64」と一致しません。EC2 インスタンスの起動に失敗しました。 738
AMI '<AMI ID>' は無効になっており、実行できません。EC2 インスタンスの起動に失敗しました。 740
ロードバランサーに関する問題 740

1 つ以上のターゲットグループが見つかりませんでした。ロードバランサーの設定の確認が失敗しました。 741
Load Balancer <ご使用のロードバランサー> が見つかりません。ロードバランサーの設定の確認が失敗しました。 741
<ロードバランサー名> という名前のアクティブなロードバランサーはありません。ロードバランサーの設定の更新が失敗しました。 742
EC2 インスタンス <インスタンス ID> は VPC にありません。ロードバランサーの設定の更新が失敗しました。 742
起動テンプレートに関する問題 742
完全な形式の有効な起動テンプレートを使用する必要があります (無効な値) 743
起動テンプレートを使用する権限がありません (許可が不十分)。 743
関連情報 745
ドキュメント履歴 748
..... dccxci

Amazon EC2 Auto Scaling とは

Amazon EC2 Auto Scaling は、アプリケーションの負荷を処理するために適切な数の Amazon EC2 インスタンスを利用できるようにします。Auto Scaling グループと呼ばれる EC2 インスタンスの集合を作成します。各 Auto Scaling グループ内のインスタンスの最小数を指定することができ、Amazon EC2 Auto Scaling グループはこのサイズよりも小さくなることはありません。各 Auto Scaling グループ内のインスタンスの最大数を指定することができ、Amazon EC2 Auto Scaling グループはこのサイズよりも大きくなることはありません。グループの作成時、またはそれ以降の任意の時点で、希望するキャパシティーを指定した場合、Amazon EC2 Auto Scaling によって、グループのインスタンス数はこの数に設定されます。スケーリングポリシーを指定する場合、Amazon EC2 Auto Scaling でアプリケーションに対する需要の増減に応じて、インスタンスを起動または終了できます。

例えば、次の Auto Scaling グループの最小サイズは 4 インスタンス、希望する容量は 6 インスタンス、最大サイズは 12 インスタンスです。定義するスケーリングポリシーによって、指定した条件に基づいて、インスタンスの最小数と最大数の間でインスタンス数が調整されます。



Amazon EC2 Auto Scaling の機能

Amazon EC2 Auto Scaling では、EC2 インスタンスは Auto Scaling グループに編成され、スケーリングと管理の目的で論理単位として扱われます。Auto Scaling グループはEC2 インスタンスの設定テンプレートとして起動テンプレート (または起動設定) を使用します。

Amazon EC2 Auto Scaling の主な機能は次のとおりです。

実行中のインスタンスの状態のモニタリング

Amazon EC2 Auto Scaling は、EC2 ヘルスチェックを使用してインスタンスのヘルスと可用性を自動的にモニタリングし、終了または障害が発生したインスタンスを置き換えて、希望する容量を維持します。

カスタムヘルスチェック

組み込みのヘルスチェックに加えて、アプリケーションに固有のカスタムヘルスチェックを定義して、期待どおりに応答していることを確認することができます。インスタンスがカスタムヘルスチェックに失敗すると、希望する容量を維持するために自動的に置き換えられます。

アベイラビリティーゾーン間で容量のバランスをとる

Auto Scaling グループには複数のアベイラビリティーゾーンを指定できます。Amazon EC2 Auto Scaling は、グループのスケールに応じて、アベイラビリティーゾーン間でインスタンスを均等に分散します。これにより、アプリケーションを 1 か所で障害から保護することで、高可用性と耐障害性を実現できます。

複数のインスタンスタイプと購入オプション

1 つの Auto Scaling グループ内で、複数のインスタンスタイプと購入オプション (スポットインスタンスとオンデマンドインスタンス) を起動できるため、スポットインスタンスの使用を通じてコストを最適化できます。リザーブドインスタンスと Savings Plan sの割引を利用するには、グループ内のオンデマンドインスタンスと組み合わせて使用することもできます。

スポットインスタンスの自動交換

グループにスポットインスタンスが含まれている場合、Amazon EC2 Auto Scaling は、スポットインスタンスが中断された場合に代替スポット容量を自動的にリクエストできます。容量の再調整により、Amazon EC2 Auto Scaling は中断のリスクが高いスポットインスタンスをモニタリングし、事前に置き換えることもできます。

負荷分散

Elastic Load Balancing のロードバランシングとヘルスチェックを使用して、アプリケーショントラフィックが正常なインスタンスに均等に分散されるようにできます。インスタンスが起動または終了されるたびに、Amazon EC2 Auto Scaling はロードバランサーからインスタンスを自動的に登録および登録解除します。

スケーラビリティ

Amazon EC2 Auto Scaling には、Auto Scaling グループをスケーリングするいくつかの方法もあります。自動スケーリングを使用すると、ピーク負荷を処理する容量を追加し、需要が低い場合に容量を削除することで、アプリケーションの可用性を維持し、コストを削減できます。必要に応じて Auto Scaling グループのサイズを手動で調整することもできます。

インスタンスの更新

インスタンスの更新機能は、AMI または起動テンプレートを更新するときに、ローリング方式でインスタンスを更新するメカニズムを提供します。Canary デプロイと呼ばれる段階的なアプローチを使用して、グループ全体にロールアウトする前に、少数のインスタンスで新しい AMI または起動テンプレートをテストすることもできます。

ライフサイクルフック

ライフサイクルフックは、新しいインスタンスの起動時またはインスタンスの終了前に呼び出されるカスタムアクションを定義するのに役立ちます。この機能は、イベント駆動型アーキテクチャの構築に特に役立ちますが、ライフサイクルを通じてインスタンスを管理するのにも役立ちます。

ステートフルワークロードのサポート

ライフサイクルフックは、シャットダウン時に状態を永続化するためのメカニズムも提供します。ステートフルなアプリケーションの継続性を確保するために、スケールイン保護ポリシーまたはカスタム終了ポリシーを使用して、実行時間の長いプロセスのインスタンスが早期に終了しないようにすることもできます。

Amazon EC2 Auto Scaling のメリットの詳細については、「アプリケーションアーキテクチャの Auto Scaling の利点」を参照してください。

Amazon EC2 Auto Scaling の料金

Amazon EC2 Auto Scaling では追加料金は発生しないため、簡単に試して、 AWS アーキテクチャにどのようなメリットがあるかを確認してください。お支払いいただくのは、使用した AWS リソース (EC2 インスタンス、EBS ボリューム、 CloudWatch アラームなど) のみです。

使用を開始する

まず、最初の Auto Scaling グループの作成チュートリアルを完了して Auto Scaling グループを作成し、そのグループのインスタンスが終了したときにどのように応答するかを確認します。

Auto Scaling グループの使用

Auto Scaling グループは、以下のインターフェイスのいずれかを使用して、作成、アクセス、および管理することできます。

- AWS Management Console – Auto Scaling グループへのアクセスに使用できるウェブインターフェイスを提供します。にサインアップしている場合は AWS アカウント、 にサインインし AWS Management Console、ナビゲーションバーの検索ボックスを使用して Auto Scaling グループ Auto Scaling を検索し、Auto Scaling グループ を選択します。
- AWS Command Line Interface （ AWS CLI ） – 幅広い セットのコマンドを提供し AWS のサービス、Windows、macOS、Linux でサポートされています。開始するには、 を使用するための準備 AWS CLI を参照してください。詳細については、AWS CLI コマンドリファレンスの「autoscaling」を参照してください。
- AWS Tools for Windows PowerShell – PowerShell 環境でスクリプトを作成するユーザー向けに、幅広い AWS 製品セットのコマンドを提供します。使用を開始する方法については、『AWS Tools for Windows PowerShell ユーザーガイド』を参照してください。詳細については、「AWS Tools for PowerShell Cmdlet Reference」を参照してください。
- AWS SDKs – 言語固有の API オペレーションを提供し、署名の計算、リクエストの再試行処理、エラー処理など、接続の詳細の多くを処理します。詳細については、AWS SDK を参照してください。
- クエリ API – HTTPS リクエスを使用して呼び出す低レベル API アクションを提供します。クエリ API の使用は、 AWS のサービスにアクセスする最も直接的な方法です。ただし、この方法では、リクエストに署名するハッシュの生成やエラー処理など、低レベルの詳細な作業をアプリケーションで処理する必要があります。詳細については、「Amazon EC2 Auto Scaling API Reference」(Amazon EC2 Auto Scaling API リファレンス) を参照してください。
- AWS CloudFormation – CloudFormation テンプレートを使用した Auto Scaling グループの作成をサポートします。詳細については、「AWS CloudFormationで Auto Scaling グループを作成する」を参照してください。

にプログラムで接続するには AWS のサービス、 エンドポイントを使用します。Amazon EC2 Auto Scaling への呼び出しのエンドポイントの詳細については、「 シークレットリージョンユーザーガイド」の「 トップシークレットリージョンの エンドポイントの Amazon Amazon EC2 Auto Scaling エンドポイントとクォータAWS 全般のリファレンス。

アプリケーションアーキテクチャの Auto Scaling の利点

Amazon EC2 Auto Scaling をアプリケーションアーキテクチャに追加するのは、 AWS クラウドの利点を最大化する方法の 1 つです。Amazon EC2 Auto Scaling を使用する場合、アプリケーションには次のようなメリットがあります。

- 耐障害性の向上。Amazon EC2 Auto Scaling では、インスタンスに異常が発生したタイミングを検出し、インスタンスを削除して、その代わりに新しいインスタンスを起動することができます。複数のアベイラビリティーゾーンを使用するように Amazon EC2 Auto Scaling を設定することもできます。1 つのアベイラビリティーゾーンが利用できなくなると、Amazon EC2 Auto Scaling は別のアベイラビリティーゾーンでインスタンスを起動して補正します。
- 可用性の向上。Amazon EC2 Auto Scaling は、現在のトラフィック需要を処理するための適切なキャパシティを、お使いのアプリケーションに常に持たせるのに役立ちます。
- コスト管理の強化。Amazon EC2 Auto Scaling では、必要に応じて動的に処理能力を増減できます。使用する EC2 インスタンスの料金を支払うので、必要なときにインスタンスを起動し、不要な場合は削除することで費用を節約できます。

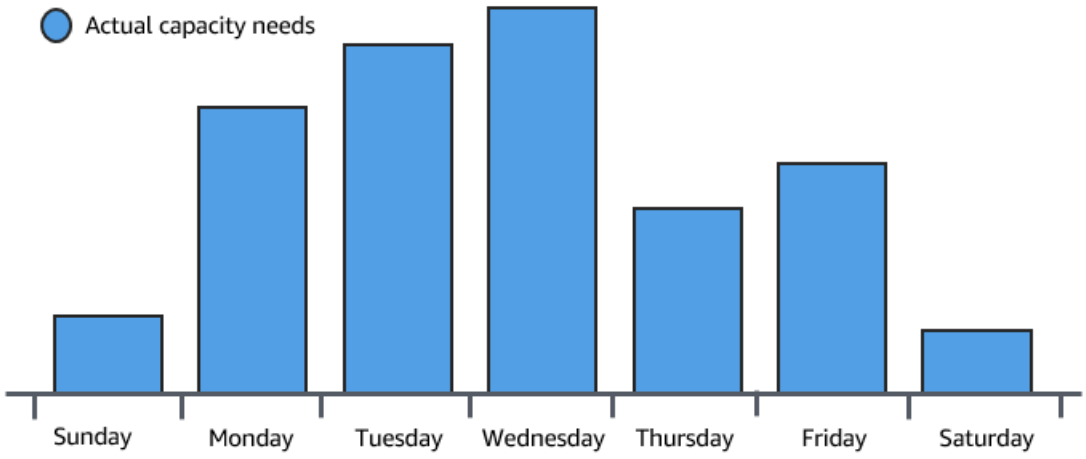
内容

- 例: 変化する需要に対応する
- 例: ウェブアプリアーキテクチャ
- 例: 複数のアベイラビリティーゾーン全体にインスタンスを分散させる
  - インスタンスの分散
  - アクティビティの再分散

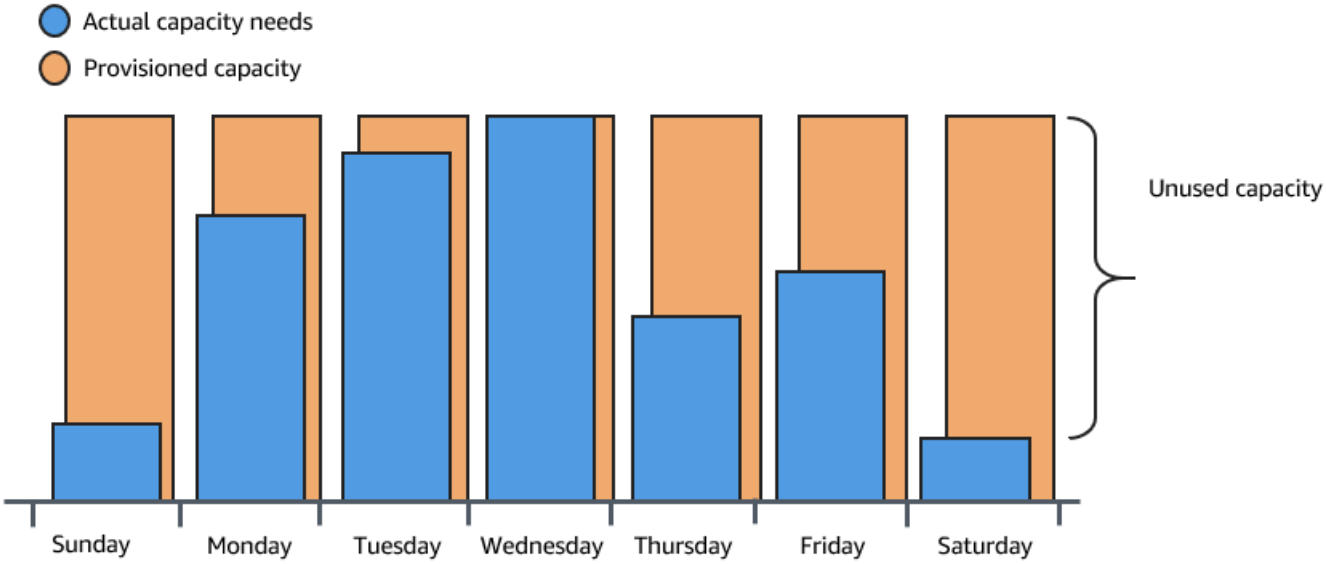
例: 変化する需要に対応する

Amazon EC2 Auto Scaling のいくつかのメリットを示すために、 AWSで実行されている基本的なウェブアプリケーションを例として取り上げます。このアプリケーションでは、従業員は会議に使用できる会議室を検索できます。週の始めと終わりには、このアプリケーションの使用量は最小になります。週の中盤では、より多くの従業員が会議をスケジュールしているため、アプリケーションの使用量が大幅に増加します。

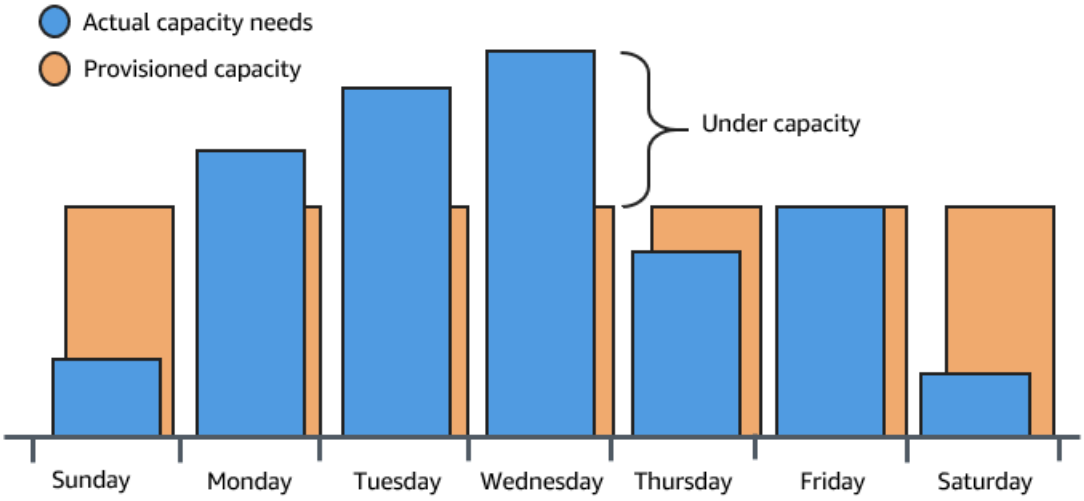
以下のグラフは、1 週間を通じてアプリケーションの処理能力がどのくらい使用されているかを示しています。



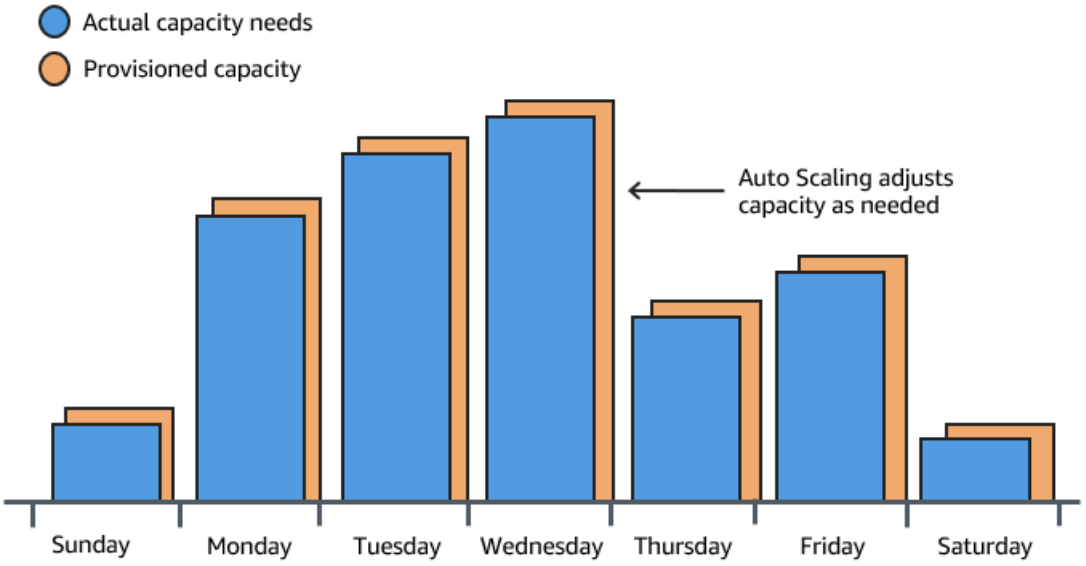
従来、このような処理能力の変化に対応した計画を立てるには 2 通りの方法がありました。最初の方法は、アプリケーションが使用量に応じて十分な処理能力を常に確保できるように、十分な数のサーバーを追加する方法です。ただし、この方法の欠点は、アプリケーションがこの十分な処理能力を必要としない日数が発生することです。余分な処理能力が未使用のままになり、実質的には、アプリケーションの実行を維持するためのコストが増加します。



2 番目の方法は、アプリケーションの平均的な使用量を処理するための十分な処理能力を確保する方法です。この方法では、使用される機会が少ない機器を購入しないため、コストはあまりかかりません。ただし、アプリケーションの使用量が処理能力を超えた場合に、カスタマーエクスペリエンスが低下するというリスクがあります。



このアプリケーションに Amazon EC2 Auto Scaling を追加することによって、3 番目の方法が利用可能になります。必要な場合にのみアプリケーションに新しいインスタンスを追加し、それらのインスタンスが不要になった場合は、インスタンスを終了できます。Amazon EC2 Auto Scaling では EC2 インスタンスを使用するため、インスタンスを使用した場合に、それらのインスタンスの料金のみを支払うだけで済みます。この方法により、費用効率が高いアーキテクチャの利用が可能になります。このようなアーキテクチャでは最適なカスタマーエクスペリエンスが提供され、費用を最小限に抑えることができます。

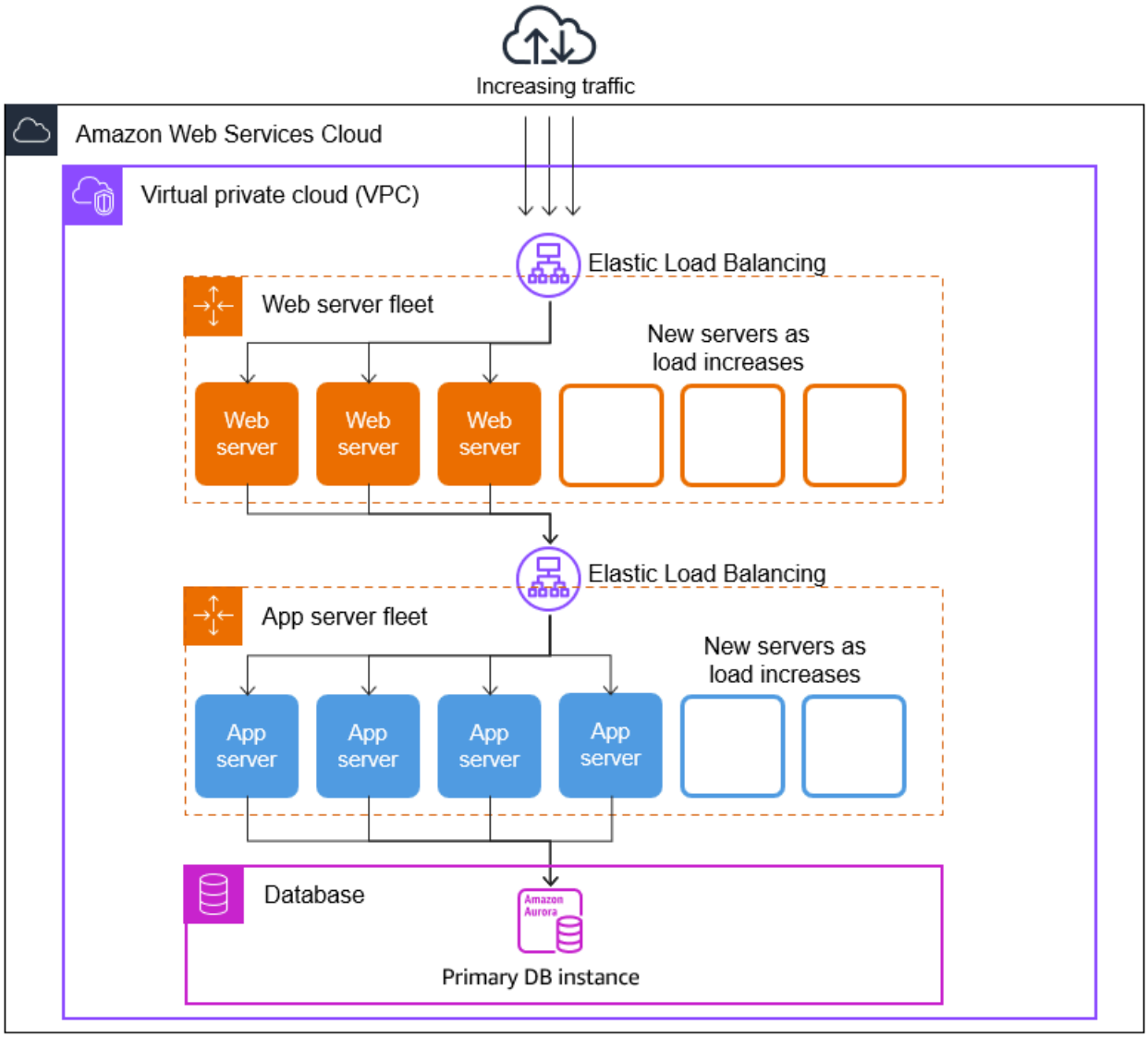


例: ウェブアプリアーキテクチャ

一般的なウェブアプリケーションのシナリオでは、顧客のトラフィックのボリュームに対応するために、アプリケーションの複数のコピーを同時に実行します。このようなアプリケーションの複数のコ

ピーは、同じ EC2 インスタンス (クラウドサーバー) でホストされ、それぞれが顧客のリクエストを処理します。

Amazon EC2 Auto Scaling は、お客様に代わって、これらの EC2 インスタンスの起動と終了を管理します。Auto Scaling グループが EC2 インスタンスを起動または終了するタイミングを決定する一連の条件 (Amazon CloudWatch アラームなど) を定義します。Auto Scaling グループをネットワークアーキテクチャに追加することによって、アプリケーションの可用性と耐障害性を向上させることができます。



Auto Scaling グループは必要な数だけ作成できます。例えば、各層のための Auto Scaling グループを作成できます。

Auto Scaling グループ内のインスタンス間のトラフィックを分散させるために、アーキテクチャにロードバランサーを導入できます。詳細については、「Elastic Load Balancing」を参照してください。

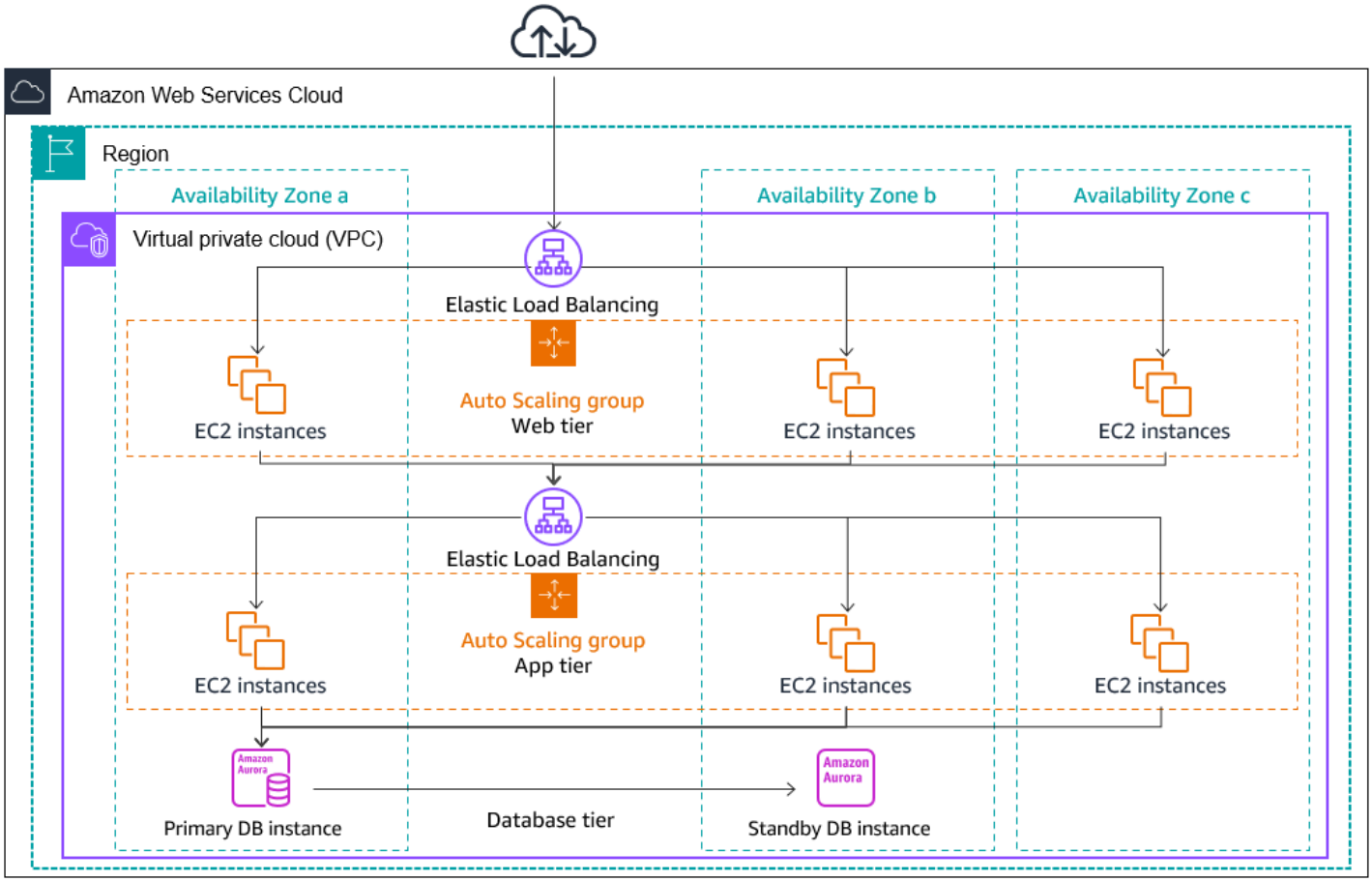
例: 複数のアベイラビリティーゾーン全体にインスタンスを分散させる

アベイラビリティーゾーンは、所定の AWS リージョンにある隔離された場所です。各リージョンには、そのリージョンに対して高可用性を提供するために設計された複数のアベイラビリティーゾーン (AZ) があります。アベイラビリティーゾーンは独立しているため、複数のゾーンを使用するようにアプリケーションを設計すると、アプリケーションの可用性が向上します。詳細については、「Amazon EC2 Auto Scaling のレジリエンス」を参照してください。

アベイラビリティーゾーンは、 AWS リージョン コードの後に文字識別子 (などus-east-1a) が続きます。デフォルトの VPC を使用する代わりに独自の VPC とサブネットを作成する場合は、各アベイラビリティーゾーンに 1 つ、または複数のサブネットを定義できます。各サブネットが完全に 1 つのアベイラビリティーゾーン内に含まれている必要があり、1 つのサブネットが複数のゾーンに、またがることはできません。詳細については、「Amazon VPC ユーザーガイド」の「Amazon VPC の仕組み」を参照してください。

Auto Scaling グループを作成するときは、Auto Scaling グループをデプロイする VPC とサブネットを選択する必要があります。Amazon EC2 Auto Scaling は、選択されたサブネットにインスタンスを作成します。そのため、各インスタンスは Amazon EC2 Auto Scaling によって選択された特定のアベイラビリティーゾーンに関連付けられます。インスタンスを起動すると、Amazon EC2 Auto Scaling は、可用性と信頼性を高めるためにインスタンスをゾーン間で均等に分散させようと試みます。

以下の画像は、3 つのアベイラビリティーゾーン間にデプロイされた多層アーキテクチャの概要です。



インスタンスの分散

Amazon EC2 Auto Scaling は自動的に、有効化された各アベイラビリティーゾーン内で等しい数のインスタンスを維持しようと試みます。Amazon EC2 Auto Scaling は、インスタンス数が最も少ないアベイラビリティーゾーンで新しいインスタンスの起動を試みることによって、これを実行します。アベイラビリティーゾーンに対して複数のサブネットが選択されている場合、Amazon EC2 Auto Scaling はアベイラビリティーゾーンからサブネットをランダムに選択します。この試みが失敗した場合、Amazon EC2 Auto Scaling は成功するまで別のアベイラビリティーゾーンでのインスタンスの起動を試みます。

アベイラビリティーゾーンが異常、または利用不能な状況では、アベイラビリティーゾーン間でのインスタンスの分散が不均等になる可能性があります。アベイラビリティーゾーンが回復すると、Amazon EC2 Auto Scaling は Auto Scaling グループのバランスを自動的に再調整します。これは、インスタンス数が最も少ない有効なアベイラビリティーゾーンでインスタンスを起動し、その他のゾーンでインスタンスを終了することによって行われます。

アクティビティの再分散

再分散アクティビティは、アベイラビリティーゾーンの再調整とキャパシティ再調整の 2 つのカテゴリに分類されます。

アベイラビリティーゾーンの再調整

特定のアクションが発生すると、Auto Scaling グループはアベイラビリティーゾーン間で不均衡になる可能性があります。Amazon EC2 Auto Scaling は、これらのアベイラビリティーゾーンを再調整することによって、これを補正します。再調整アクティビティは、以下のアクションが原因で行われる場合があります。

- Auto Scaling グループに関連付けられたアベイラビリティーゾーンを変更する。
- インスタンスを明示的に終了もしくはデタッチする、またはインスタンスをスタンバイにすることでグループのバランスが悪くなる。
- それまでキャパシティが不足していたアベイラビリティーゾーンが回復し、使用できるキャパシティが増えた。
- これまでスポット価格が上限価格より高かったアベイラビリティーゾーンで、スポット価格が上限価格より低くなった。

再調整するにあたって、Amazon EC2 Auto Scaling は新しいインスタンスを起動してから古いインスタンスを終了します。そうすることで、再調整によってアプリケーションのパフォーマンスや可用性が損なわれることがなくなります。

Amazon EC2 Auto Scaling は古いインスタンスを終了する前に新しいインスタンスの起動を試みるため、指定された最大キャパシティ、またはそれに近いキャパシティが使用されていると、再調整アクティビティが妨げられる、または完全に停止される可能性があります。

この問題を回避するため、システムは、再調整アクティビティの実行中に指定された最大キャパシティを一時的に超過することができます。デフォルトでは、10% のマージンまたは 1 つのインスタンスのいずれか大きい方で指定できます。この超過範囲は、グループのキャパシティが最大またはそれに近く、再調整が必要な場合にのみ拡張されます。この追加キャパシティーは、グループの再分散に要する時間にわたってのみ提供されます (通常は数分)。

または、インスタンスのメンテナンスポリシーを使用して Auto Scaling グループのしきい値を設定し、そのグループはそのしきい値の範囲内でのみ容量を増減できます。これにより、グループが再調整する速度を制御できます。詳細については、「インスタンスのメンテナンスポリシー」を参照してください。

キャパシティーの再調整

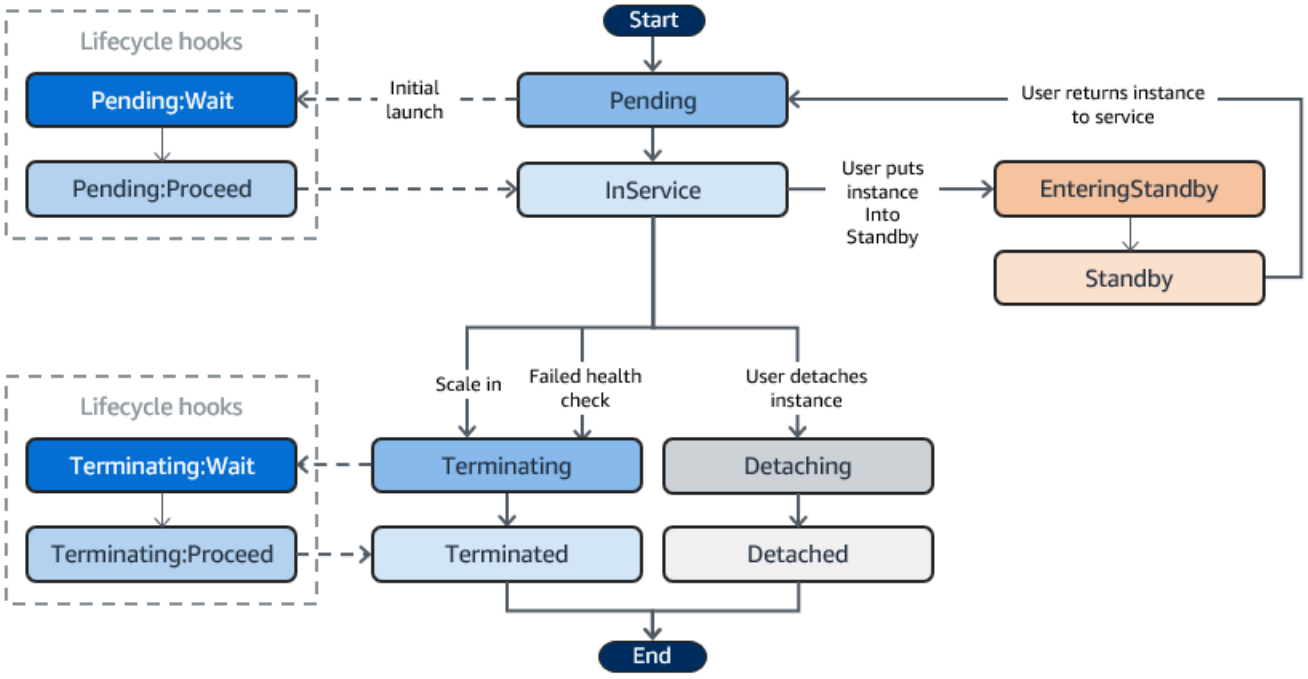
スポットインスタンスを使用するときは、Auto Scaling グループのキャパシティの再調整を有効にできます。これは、Amazon EC2 がスポットインスタンスの中断リスクが高まっていることを報告するときに、Amazon EC2 Auto Scaling がスポットインスタンスの起動を試行できるようにします。新しいインスタンスの起動後、Amazon EC2 Auto Scaling は古いインスタンスを終了します。詳細については、「キャパシティの再調整を使用して Amazon EC2 スポットの中断に対処する」を参照してください。

Amazon EC2 Auto Scaling インスタンスのライフサイクル

Auto Scaling グループの EC2 インスタンスに、他の EC2 インスタンスとは異なるパスまたはライフサイクルがあります。Auto Scaling グループがインスタンスを起動すると、ライフサイクルが起動してサービスに組み込まれます。インスタンスを削除するとライフサイクルが終了するか、または Auto Scaling グループがインスタンスをサービスから削除して終了します。

> **Note**
>
> インスタンスが起動すると、サービスを使用していなくてもすぐに課金されます。

次の図は、Amazon EC2 Auto Scaling ライフサイクルのインスタンス状態の遷移を示しています。



スケールアウト

以下で、イベントを Auto Scaling グループで直接スケールアウトして EC2 インスタンスを起動し、グループにアタッチします。

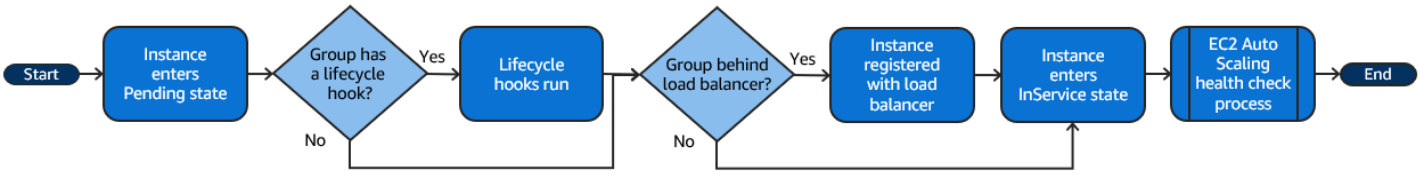
- 手動でグループのサイズを拡大します。詳細については、「既存の Auto Scaling グループの希望する容量を変更する」を参照してください。
- スケーリングポリシーを作成して、指定した需要の増加に基づいてグループのサイズを自動的に拡大します。詳細については、「Amazon EC2 Auto Scaling の動的スケーリング」を参照してください。
- スケジュールでスケーリングを設定して、特定の時間でグループのサイズを拡大します。詳しくは、「Amazon EC2 Auto Scaling のスケジュールされたスケーリング」を参照してください。

スケールアウトイベントが発生した場合、Auto Scaling グループは割り当てられた起動テンプレートを使用して、必要な数の EC2 インスタンスを起動します。これらのインスタンスは `Pending` 状態で起動します。ライフサイクルフックを Auto Scaling グループに追加する場合は、ここでカスタムアクションを実行できます。詳細については、「ライフサイクルフック」を参照してください。

各インスタンスが完全に設定され、Amazon EC2 ヘルスチェックに合格すると、Auto Scaling グループにインスタンスをアタッチして `InService` 状態へ移行します。インスタンスは、Auto Scaling グループで必要なキャパシティにカウントされます。

Elastic Load Balancing ロードバランサーからのトラフィックを受信するように Auto Scaling グループが設定されている場合、Amazon EC2 Auto Scaling は、インスタンスを `InService` としてマークする前にロードバランサーに登録します。

スケールアウトイベントのロードバランサーにインスタンスを登録する手順を以下に示します。



インスタンスが実行中

以下のいずれかが発生するまで、インスタンスは `InService` 状態のままとなります。

- スケールインイベントが発生し、Amazon EC2 Auto Scaling が Auto Scaling グループのサイズを縮小するためにこのインスタンスを削除する。詳細については、「スケールイン中に終了する Auto Scaling インスタンスを制御する」を参照してください。
- ユーザーがインスタンスを Standby 状態にする 詳細については、「スタンバイを入力し終了します。」を参照してください。
- Auto Scaling グループからインスタンスをデタッチします。詳細については、「インスタンスのデタッチまたはアタッチ」を参照してください。
- インスタンスが必要な数のヘルスチェックに失敗したため、Auto Scaling グループから削除され、終了されて置き換えられます。詳しくは、「Auto Scaling グループ内のインスタンスのヘルスチェック」を参照してください。

スケールイン

以下で、イベントを Auto Scaling グループで直接スケールインして EC2 インスタンスをグループからデタッチして、終了します。

- 手動でグループのサイズを縮小します。詳細については、「既存の Auto Scaling グループの希望する容量を変更する」を参照してください。
- スケーリングポリシーを作成して、指定した需要の縮小に基づいてグループのサイズを自動的に縮小します。詳細については、「Amazon EC2 Auto Scaling の動的スケーリング」を参照してください。
- スケジュールでスケーリングを設定して、特定の時間にグループのサイズを縮小します。詳しくは、「Amazon EC2 Auto Scaling のスケジュールされたスケーリング」を参照してください。

作成したそれぞれのスケールアウトイベントについて、対応するスケールインイベントを作成することが重要です。これにより、アプリケーションに割り当てられたリソースが、それらのリソースに対する要求に可能な限り厳密に対応するようになります。

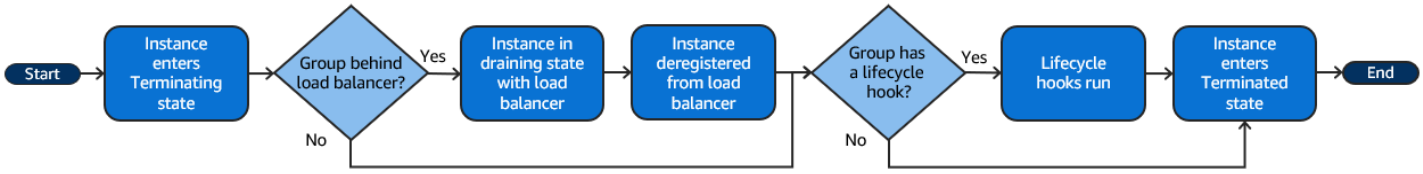
スケールインイベントが発生すると、Auto Scaling グループは 1 つ以上のインスタンスをデタッチします。Auto Scaling グループは終了ポリシーを使用して、終了するインスタンスを決定します。Auto Scaling グループからの終了のプロセス中にあるインスタンスは、Terminating 状態に移行し、稼働状態に戻ることはできません。

Elastic Load Balancing ロードバランサーからのトラフィックを受信するように Auto Scaling グループが設定されている場合、Amazon EC2 Auto Scaling は、終了するインスタンスをロードバランサーからインスタンスを登録解除します。インスタンスの登録を解除すると、新しいリクエストはす

べてロードバランサーのターゲットグループ内の他のインスタンスにリダイレクトされ、インスタンスへの既存の接続は登録解除の遅延の期限が切れるまで継続できます。

ライフサイクルフックを Auto Scaling グループに追加する場合は、終了するインスタンスに対してカスタムアクションを実行できます。詳細については、「ライフサイクルフック」を参照してください。最後に、インスタンスは完全に終了し `Terminated` 状態へ移行します。

スケールインイベントのロードバランサーでインスタンスを登録解除する手順を以下に示します。



インスタンスのデタッチ

Auto Scaling グループからインスタンスをデタッチできます。インスタンスをデタッチした後で、インスタンスを Auto Scaling グループとは別に管理するか、または別の Auto Scaling グループにアタッチできます。

詳細については、「インスタンスのデタッチまたはアタッチ」を参照してください。

インスタンスのアタッチ

特定の基準を満たす実行中の EC2 インスタンスを Auto Scaling グループにアタッチできます。インスタンスがアタッチされると、Auto Scaling グループの一部として管理されます。

詳細については、「インスタンスのデタッチまたはアタッチ」を参照してください。

ライフサイクルフック

インスタンスを起動または終了したときにカスタムアクションを実行できるように、ライフサイクルフックを Auto Scaling グループに追加できます。

Amazon EC2 Auto Scaling はスケールアウトイベントに応答すると、1 つ以上のインスタンスを起動します。これらのインスタンスは `Pending` 状態で起動します。Auto Scaling グループに `autoscaling:EC2_INSTANCE_LAUNCHING` ライフサイクルフックを追加すると、インスタンスは `Pending` 状態から `Pending:Wait` 状態に移行します。ライフサイクルアクションを完了したら、インスタンスは `Pending:Proceed` 状態に移行します。インスタンスが完全に設定されると、Auto Scaling グループにアタッチされて `InService` 状態へ移行します。

Amazon EC2 Auto Scaling はスケールインイベントに応答すると、1 つ以上のインスタンスを終了します。これらのインスタンスは Auto Scaling グループからデタッチされ Terminating 状態へ移行します。Auto Scaling グループに autoscaling:EC2_INSTANCE_TERMINATING ライフサイクルフックを追加すると、インスタンスは Terminating 状態から Terminating:Wait 状態に移行します。ライフサイクルアクションを完了したら、インスタンスは Terminating:Proceed 状態に移行します。インスタンスが完全に終了すると、Terminated 状態へ移行します。

詳細については、「Amazon EC2 Auto Scaling のライフサイクルフック」を参照してください。

スタンバイを入力し終了します。

InService 状態にあるインスタンスを、Standby 状態に移行できます。これによりインスタンスをサービスから削除し、トラブルシューティングや変更を加えてから、サービスに戻すことができます。

Standby状態のインスタンスは引き続き Auto Scaling グループによって管理されます。ただし、このようなインスタンスを稼働状態に戻すまで、それらはアプリケーションのアクティブな部分にはなりません。

詳細については、「Auto Scaling グループからインスタンスを一時的に削除する」を参照してください。

Auto Scaling リソースとグループのクォータ

AWS アカウント には、各 AWS サービスについて、以前は制限と呼ばれていたデフォルトのクォータがあります。特に明記されていない限り、クォータは地域固有です。一部のクォータについては引き上げをリクエストできますが、その他のクォータについては引き上げることはできません。

Amazon EC2 Auto Scaling のクォータを表示するには、Service Quotas コンソールを開きます。ナビゲーションペインで、[AWS のサービス] を選択し、[Amazon EC2 Auto Scaling] を選択します。

クォータの引き上げをリクエストするには、Service Quotas ユーザーガイド の「クォータ引き上げリクエスト」を参照してください。Service Quotas でクォータがまだ利用できない場合は、[クォータの引き上げをリクエスト] フォームを使用してください。クォータの引き上げは、リクエストされたリージョンに関連付けられます。

すべてのリクエストは に送信されます AWS Support。 AWS Support コンソールでリクエストケースを追跡できます。

Amazon EC2 Auto Scaling リソース

AWS アカウント には、作成できる Auto Scaling グループと起動設定の数に関連する次のクォータがあります。

リソース	デフォルトのクォータ
リージョンあたりの Auto Scaling グループ	500
リージョンごとの起動設定	200

Auto Scaling グループの設定

AWS アカウント には、Auto Scaling グループの設定に関連する以下のクォータがあります。変更することはできません。

リソース	クォータ
Auto Scaling グループあたりのスケーリングポリシー	50
Auto Scaling グループあたりのスケジュールされたアクション	125
ステップスケーリングポリシーあたりのステップ調整	20
Auto Scaling グループあたりのライフサイクルフック	50
Auto Scaling グループあたりの SNS トピック	10
Auto Scaling グループあたりの Classic Load Balancer	50
Auto Scaling グループごとの Elastic Load Balancing ターゲットグループ	50
Auto Scaling グループごとの VPC Lattice ターゲットグループ	5

Auto Scaling グループ API オペレーション

Amazon EC2 Auto Scaling では、Auto Scaling グループをバッチで変更するための API オペレーションを利用できます。1 回のオペレーションで許可される項目の最大数 (配列メンバーの最大数) に対する API の制限を以下に示します。変更することはできません。

操作	配列メンバーの最大数
AttachInstances	20 個のインスタンス ID
AttachLoadバランサー	10 個のロードバランサー
AttachLoadBalancerTargetグループ	10 個のターゲットグループ
BatchDeleteScheduledAction	50 個のスケジュールに基づくアクション
BatchPutScheduledUpdateGroupAction	50 個のスケジュールに基づくアクション
DetachInstances	20 個のインスタンス ID
DetachLoadバランサー	10 個のロードバランサー
DetachLoadBalancerTargetグループ	10 個のターゲットグループ
EnterStandby	20 個のインスタンス ID
ExitStandby	20 個のインスタンス ID
SetInstance保護	50 個のインスタンス ID

Amazon EC2 Auto Scaling API のリクエストスロットリング

Amazon EC2 Auto Scaling API リクエストは、サービス帯域幅を維持するためにトークンバケットスキームを使用してスロットリングされます。詳細については、Amazon EC2 Auto Scaling API リファレンス」の「API リクエストレート」を参照してください。

EC2 終了レート

Amazon EC2 Auto Scaling は、Auto Scaling グループがスケールインするときに一度に実行できる EC2 インスタンスの終了オペレーションの数を動的に決定します。これは、Auto Scaling グループ間で一度に終了されるインスタンスの数にばらつきが見られる可能性があることを意味します。これ

らのばらつきは、Amazon EC2 Auto Scaling がロードバランサーからインスタンスを登録解除する必要があるかどうかなど、外部の考慮事項によって発生します。

その他のサービス

Amazon EC2 や Amazon VPC などの他のサービスのクォータは、Auto Scaling グループに影響を与える可能性があります。を使用して Service Quotas 、 の EC2 インスタンスおよびその他のリソースのクォータを更新できます AWS アカウント。 Service Quotas コンソールでは、使用可能なすべてのサービスクォータを表示し、それらの引き上げをリクエストできます。詳細については、「Service Quotas ユーザーガイド」の「Requesting a quota increase」(クォータ引き上げのリクエスト) を参照してください。

起動テンプレートに固有のクォータについては、「Amazon EC2 ユーザーガイド」の「起動テンプレートの制限」を参照してください。 Amazon EC2

Amazon EC2 Auto Scaling を使用するようにセットアップする

Amazon EC2 Auto Scaling の使用を開始する前に、以下のタスクを完了してください。

タスク

- Amazon EC2 を使用するための準備を整える
- を使用するための準備 AWS CLI

Amazon EC2 を使用するための準備を整える

Amazon EC2 を使用したことがない場合は、Amazon EC2 のドキュメントで説明されているタスクを完了してください。詳細については、Amazon EC2 ユーザーガイド」のAmazon EC2 でのセットアップ」またはAmazon EC2 ユーザーガイド」の「Amazon EC2 でのセットアップAmazon EC2」を参照してください。

を使用するための準備 AWS CLI

AWS コマンドラインツールを使用して、システムのコマンドラインでコマンドを発行し、Amazon EC2 Auto Scaling やその他の AWS タスクを実行できます。

AWS Command Line Interface （ AWS CLI) を使用するには、 のバージョン 1 または 2 をダウンロード、インストール、設定します AWS CLI。Amazon EC2 Auto Scaling の同じ機能はバージョン 1 および 2 で利用できます。 AWS CLI バージョン 1 をインストールするには、「AWS CLI バージョン 1 ユーザーガイド」の「AWS CLIのインストール、アップデート、アンインストール」を参照してください。 AWS CLI バージョン 2 をインストールするには、「 AWS CLI バージョン 2 ユーザーガイド」の「 の最新バージョンのインストールまたは更新 AWS CLI」を参照してください。

AWS CloudShell では、 AWS CLI 開発環境に をインストールせずに、 AWS Management Console 代わりに で使用できます。インストールが不要になるだけではなく、認証情報の設定およびリージョンの指定も必要ありません。 AWS Management Console セッションはこのコンテキストをに提供します AWS CLI。サポートされている AWS CloudShell で を使用できます AWS リージョン。詳細については、「を使用してコマンドラインから Auto Scaling グループを作成する AWS CloudShell」を参照してください。

詳細については、AWS CLI コマンドリファレンスの「autoscaling」を参照してください。

Amazon EC2 Auto Scaling の使用を開始する

Amazon EC2 Auto Scaling の使用を開始するには、サービスを紹介するチュートリアルに従います。

トピック

- チュートリアル: 最初の Auto Scaling グループを作成する
- チュートリアル: スケーリングとロードバランシングを使用するアプリケーションのセットアップ

Auto Scaling グループ内のインスタンスのライフサイクルを管理するための特定のツールに焦点を当てたその他のチュートリアルについては、以下のトピックを参照してください。

- チュートリアル:Lambda 関数を呼び出すライフサイクルフックの設定。 このチュートリアルでは、Amazon を使用して EventBridge 、Auto Scaling グループ内のインスタンスに発生したイベントに基づいて Lambda 関数を呼び出すルールを作成する方法を示します。
- チュートリアル: インスタンスメタデータを使用してターゲットライフサイクル状態を取得するようにユーザーデータを設定する. このチュートリアルでは、インスタンスメタデータサービス (IMDS) を使用してインスタンス自体からアクションを呼び出す方法を説明します。

アプリケーションで使用する Auto Scaling グループを作成する前に、 AWS クラウドで実行されるアプリケーションを念入りにレビューするようにしてください。以下の点を考慮します。

- Auto Scaling グループに含めるアベイラビリティーゾーンの数。
- 使用できる既存リソース (セキュリティグループ、Amazon マシンイメージ (AMI) など)。
- キャパシティーをスケーリングするか、または、常時実行中のサーバーを一定数確保します。Amazon EC2 Auto Scaling は両方を同時に実行できます。
- アプリケーションのパフォーマンスと最も関連性が高いメトリクス。
- サーバーの起動とプロビジョニングにかかる時間。

アプリケーションの理解が進むにつれて、Auto Scaling アーキテクチャーをより効率的なものにすることができるようになります。

チュートリアル: 最初の Auto Scaling グループを作成する

このチュートリアルでは、 を通じて Amazon EC2 Auto Scaling を実際に紹介します AWS Management Console。EC2 インスタンスを定義する起動テンプレートと、その中に 1 つのインスタンスを持つ Auto Scaling グループを作成します。Auto Scaling グループを起動したら、インスタンスを終了し、インスタンスがサービスから削除され、置き換えられたことを確認します。一定数のインスタンスを維持するために、Amazon EC2 Auto Scaling は Amazon EC2 のヘルスチェックと到達可能性チェックを自動的に検出して応答します。

にサインアップすると AWS、 無料AWS 利用枠 を使用して Amazon EC2 Auto Scaling を無料で開始できます。無料利用枠を使用し、`t2.micro` インスタンスを 12 か月間無料で起動して利用できます (`t2.micro` が利用できないリージョンでは、無料利用枠で `t3.micro` インスタンスを使用できます)。無料利用枠に含まれないインスタンスを起動する場合は、そのインスタンスの通常の Amazon EC2 使用料がかかります。詳細については、「Amazon EC2 料金表」を参照してください。

タスク

- チュートリアルのための準備
- ステップ 1: 起動テンプレートを作成する
- ステップ 2: 単一インスタンスの Auto Scaling グループを作成する
- ステップ 3: Auto Scaling グループを検証する
- ステップ 4: Auto Scaling グループのインスタンスを終了します
- ステップ 5: 次のステップ
- ステップ 6: クリーンアップする

チュートリアルのための準備

このチュートリアルは、EC2 インスタンスの起動について知識があり、キーペアとセキュリティグループを既に作成していることを前提としています。詳細については、Amazon EC2 ユーザーガイド」のAmazon EC2 でのセットアップ」を参照してください。

Amazon EC2 Auto Scaling の使用を開始するには、 のデフォルト VPC を使用できます AWS アカウント。デフォルト VPC には、各アベイラビリティーゾーンのデフォルトのパブリックサブネットと、VPC にアタッチされたインターネットゲートウェイが含まれます。Amazon Virtual Private Cloud (Amazon VPC) コンソールの VPC のページで VPC を表示できます。

ステップ 1: 起動テンプレートを作成する

このステップでは、Amazon EC2 Auto Scaling が作成する EC2 インスタンスのタイプを指定する起動テンプレートを作成します。 Amazon EC2 使用する Amazon Machine Image (AMI) の ID、インスタンスタイプ、キーペア、セキュリティグループなどの情報を含めます。

起動テンプレートを作成するには

1. Amazon EC2 コンソールを開き、「起動テンプレート」ページに移動します。
2. 上部のナビゲーションバーで、[AWS リージョン] を選択します。作成する起動テンプレートと Auto Scaling グループは、指定するリージョンに関連付けられます。
3. [起動テンプレートの作成] を選択します。
4. [起動テンプレート名] を使用する場合、**`my-template-for-auto-scaling`** を入力します。
5. [Auto Scaling ガイダンス] で、チェックボックスを選択します。
6. [Application and OS Images (Amazon Machine Image)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で、[Quick Start] (クイックスタート) リストから Amazon Linux 2 (HVM) のバージョンを選択します。AMI はインスタンスの基本設定テンプレートとして機能します。
7. [インスタンスタイプ] で、指定した AMI と互換性のあるハードウェア設定を選択します。
8. (オプション) [Key pair (login)] (キーペア (ログイン)) で、既存のキーペアを選択します。キーペアは Amazon EC2 インスタンスを SSH に接続するときに使用します。インスタンスへの接続は、このチュートリアルには含まれていません。このため、SSH を使用してインスタンスに接続する予定の場合を除き、キーペアを指定する必要はありません。
9. [Network settings] (ネットワーク設定) で、[Advanced network configuration] (高度なネットワーク設定) を展開し、以下を実行します。

    a. [Add network interface] (ネットワークインターフェイスを追加) を選択して、プライマリネットワークインターフェイスを追加します。

    b. パブリック IP を自動割り当てするには、インスタンスがパブリック IPv4 アドレスを受け取るかどうかを指定します。デフォルトでは、Amazon EC2 EC2 はパブリック IPv4 アドレスを割り当てます。 IPv4 インスタンスに接続する必要がない場合は、 を無効にするを選択します。

    c. セキュリティグループ ID で、Auto Scaling グループの VPC として使用する予定の同じ VPC 内のセキュリティグループを選択します。セキュリティグループを指定しないと、インスタンスは VPC のデフォルトのセキュリティグループに自動的に関連付けられます。

d. 終了時に削除 で、はい を選択して、インスタンスが削除されたときにネットワークインターフェイスを削除します。

10. [起動テンプレートの作成] を選択します。
11. 確認ページで、[Auto Scaling グループの作成] を選択します。

ステップ 2: 単一インスタンスの Auto Scaling グループを作成する

起動テンプレートを作成した後、中断した場所を続行するには、次の手順に従います。

Auto Scaling グループを作成する

1. [Choose launch template or configuration (起動テンプレートまたは設定の選択)] ページで、Auto Scaling グループの名前を **`my-first-asg`** に入力します。
2. [次へ] をクリックします。

   インスタンス起動オプションの選択ページが表示され、Auto Scaling グループで使用する VPC ネットワーク設定を選択したり、オンデマンドインスタンスとスポットインスタンスを起動するオプションを提供したりできます。
3. ネットワークセクションで、VPC を選択した のデフォルト VPC に設定したままにするか AWS リージョン、独自の VPC を選択します。デフォルトの VPC は、インスタンスへのインターネット接続を提供するように自動的に設定されます。この VPC には、リージョンの各アベイラビリティーゾーンのパブリックサブネットが含まれます。
4. [Availability Zones and subnets] (アベイラビリティゾーンとサブネット) で、含める各アベイラビリティーゾーンからサブネットを選択します。複数のアベイラビリティーゾーンのサブネットを使用することで、高可用性を得られます。詳細については、「VPC サブネットを選択する場合の考慮事項」を参照してください。
5. [Instance type requirements] (インスタンスタイプの要件) セクションでは、このステップを簡略化するためにデフォルト設定を使用します。(起動テンプレートを上書きしないでください。) このチュートリアルでは、起動テンプレートで指定されたインスタンスタイプを使用して、オンデマンドインスタンスを 1 つだけ起動します。
6. このチュートリアルの残りの部分はデフォルトのままにして、[Skip to review (スキップして確認)] を選択します。

> **Note**
>
> グループの初期サイズは、希望するキャパシティーによって決まります。デフォルト値は 1 インスタンスです。

7. [Review (確認)] ページでグループの情報を確認し、[Auto Scaling グループの作成] を選択します。

ステップ 3: Auto Scaling グループを検証する

Auto Scaling グループを作成し、グループによって EC2 インスタンスが起動されたことを確認する準備が整いました。

> **Tip**
>
> 以下の手順では、Auto Scaling グループについて [Activity history] (アクティビティ履歴) と [Instances] (インスタンス) の各セクションを調べます。どちらのセクションにも、名前付きの列がすでに表示されているはずです。非表示の列を表示する、または表示される行の数を変更するには、各セクションの右上隅にある歯車アイコンを選択して設定モーダルを開き、必要に応じて設定を更新してから、[Confirm] (確認) を選択します。

Auto Scaling グループが EC2 インスタンスを起動したことを確認するには

1. Amazon EC2 コンソールで Auto Scaling グループのページを開きます。
2. 作成した Auto Scaling グループの横にあるチェックボックスを選択します。

   [Auto Scaling groups] (Auto Scaling グループ) ページの下部にスプリットペインが開きます。使用可能な最初のタブは [詳細] タブで、Auto Scaling グループに関する情報が表示されます。
3. 2 番目のタブ [アクティビティ] を選択します。[アクティビティ履歴] で、Auto Scaling グループに関連付けられているアクティビティの進行状況を表示できます。[ステータス] 列には、インスタンスの現在のステータスが表示されます。インスタンスが起動している間、ステータス列には [`Not yet in service`] と表示されます。ステータスは、インスタンスが起動されると `Successful` に変わります。[Refresh] ボタンを使用して、インスタンスの現在のステータスを表示することもできます。
4. [インスタンス管理] タブの [インスタンス] で、インスタンスのステータスを表示できます。

5. インスタンスが正常に起動したことを確認します。インスタンスの起動には短時間かかります。

   - [ライフサイクル] 列には、インスタンスの状態が表示されます。最初、インスタンスの状態は Pending です。インスタンスがトラフィックを受信できるようになったら、そのステータスは InService です。
   - ヘルスステータス列には、インスタンスの Amazon EC2 Auto Scaling ヘルスチェックの結果が表示されます。

ステップ 4: Auto Scaling グループのインスタンスを終了します

これらのステップを使用して Amazon EC2 Auto Scaling の機能 (具体的には、必要に応じて新しいインスタンスを起動する方法) を詳しく確認できます。このチュートリアルで作成した Auto Scaling グループの最小サイズは、1 インスタンスです。そのため、実行中のインスタンスを終了する場合、Amazon EC2 Auto Scaling はその代わりとなる新しいインスタンスを起動する必要があります。

1. Amazon EC2 コンソールで Auto Scaling グループのページを開きます。
2. Auto Scaling グループの横にあるチェックボックスを選択します。
3. [インスタンス管理] タブの [インスタンス] で、インスタンスの ID を選択します。

   そうすると、Amazon EC2 コンソールの [Instances] (インスタンス) ページが開きます。インスタンスはここで終了できます。
4. [Actions]、[Instance State]、[Terminate] の順に選択します。確認を求めるメッセージが表示されたら、[Yes、Terminate] (はい、終了する) を選択します。
5. ナビゲーションペインの 自動スケーリング で、[Auto Scaling Groups] (Auto Scaling グループ) を選択します。Auto Scaling グループを選択し、[アクティビティ] タブを選択します。

   インスタンスページからインスタンスを終了する場合、新しいインスタンスが起動されるまでにインスタンスを終了してから 1~2 分かかります。アクティビティ履歴で、スケーリングアクティビティが開始すると、最初のインスタンスの削除のエントリおよび新しいインスタンスの起動のエントリが表示されます。新しいエントリが表示されるまで、更新ボタンを使用します。
6. [インスタンス管理] タブの [インスタンス] セクションには、新しいインスタンスのみが表示されます。
7. ナビゲーションペインの [Instances] (インスタンス) で、[Instances] (インスタンス) を選択します。このページには、終了したインスタンスと実行中の新しいインスタンスの両方が表示されます。

ステップ 5: 次のステップ

先ほど作成した基本インフラストラクチャを削除する場合は、次のステップに進みます。それ以外の場合は、ベースとしてこのインフラストラクチャを使用し、次の 1 つ以上を試すことができます。

- Session Manager または SSH を使用した Linux インスタンスへの接続。詳細については、「Amazon EC2 ユーザーガイド」の「Session Manager を使用して Linux インスタンスに接続する」および「SSH を使用して Linux または macOS から Linux インスタンスに接続する」を参照してください。 Amazon EC2
- Auto Scaling グループの起動インスタンス、または終了インスタンスが変わるたびに通知するよう、Amazon SNS 通知を設定します。詳しくは、「Amazon SNS 通知オプション」を参照してください。
- Auto Scaling グループを手動でスケールして、SNS 通知をテストします。詳細については、「Auto Scaling グループの希望する容量を変更する」を参照してください。

また、ターゲット追跡スケーリングポリシー を読んでおくことで、Auto Scaling の概念に慣れておくこともできます。アプリケーションの負荷が変化した場合、Auto Scaling グループは、グループの希望する容量を最小容量制限と最大容量制限の間で調整することで、自動的にスケールアウト (インスタンスを追加する) およびスケールイン (インスタンスの数を減らして実行する) ことができます。これらの制限の設定に関する詳細については、「Auto Scaling グループのスケーリング制限を設定する」を参照してください。

ステップ 6: クリーンアップする

スケーリングインフラストラクチャを削除するか、Auto Scaling グループのみを削除して、起動テンプレートを後で使用するために保持できます。

AWS 無料利用枠外でインスタンスを起動した場合、不要な課金を避けるためにインスタンスを終了する必要があります。インスタンスを終了すると、それに関連付けられたデータも削除されます。

Auto Scaling グループを削除するには

1. Amazon EC2 コンソールで Auto Scaling グループのページを開きます。
2. Auto Scaling グループ (`my-first-asg`) の横にあるチェックボックスを選択します。
3. [削除] を選択します。
4. 確認を求められたら、**`delete`** を入力して指定された Auto Scaling グループの削除を確認し、[Delete] (削除) を選択します。

[Name (名前)] 列のロードアイコンに、Auto Scaling グループが削除されたことが示されます。削除が行われると、[Desired] (必要)、[Min] (最小)、[Max] (最大) 列には、Auto Scaling グループのインスタンス数として `0` と表示されます。インスタンスを終了し、グループを削除するには数分かかります。リストを更新して、現在の状態を確認します。

起動テンプレートを維持する場合は、この手順をスキップします。

起動テンプレートを削除するには

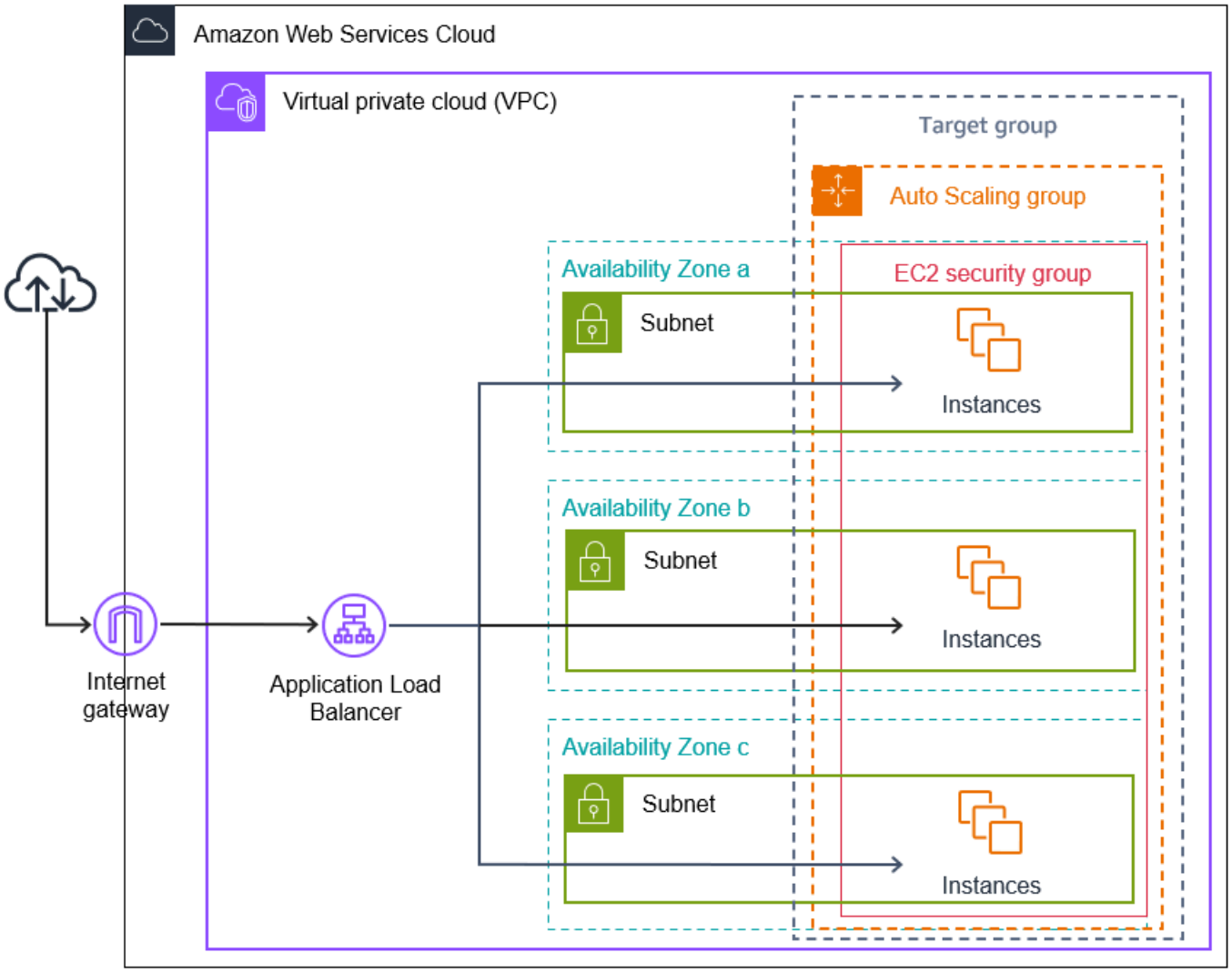
1. Amazon EC2 コンソールの起動テンプレートページを開きます。
2. 起動テンプレートを選択します (`my-template-for-auto-scaling`)。
3. [アクション]、[テンプレートの削除] の順に選択します。
4. 確認を求められたら、**`Delete`** を入力して指定した起動テンプレートの削除を確認し、[Delete] (削除) を選択します。

チュートリアル: スケーリングとロードバランシングを使用するアプリケーションのセットアップ

> **Important**
>
> このチュートリアルを進める前に、まず「最初の Auto Scaling グループの作成」の入門チュートリアルを確認することをお勧めします。

Auto Scaling グループを Elastic Load Balancing ロードバランサーに登録すると、負荷分散されたアプリケーションをセットアップできます。Elastic Load Balancing は Amazon EC2 Auto Scaling と連携し、受信トラフィックを正常な Amazon EC2 インスタンスに分散させます。これにより、アプリケーションのスケーラビリティと可用性が向上します。複数のアベイラビリティーゾーン内で Elastic Load Balancing を有効にして、アプリケーションの耐障害性を向上させることができます。

このチュートリアルでは、Auto Scaling グループの作成時に負荷分散されたアプリケーションをセットアップする基本的なステップについて説明します。完了すると、アーキテクチャは次の図表のようになります。



Elastic Load Balancing では、異なる種類のロードバランサーがサポートされています。このチュートリアルでは、Application Load Balancer を使用することをお勧めします。

アーキテクチャへのロードバランサーの導入の詳細については、「Elastic Load Balancing を使用して Auto Scaling グループ内のインスタンス全体にトラフィックを分散させる」を参照してください。

タスク

- 前提条件
- ステップ 1: 起動テンプレートまたは起動設定を設定する
- ステップ 2: Auto Scaling グループを作成する
- ステップ 3: ロードバランサーがアタッチされたことを確認する
- ステップ 4: 次のステップ

- ステップ 5 : クリーンアップ
- 関連リソース

前提条件

- ロードバランサーおよびターゲットグループ。Auto Scaling グループに使用するロードバランサーと同じアベイラビリティゾーンを選択していることを確認します。詳細については、Elastic Load Balancing ユーザーガイドの Elastic Load Balancing で使用開始 を参照してください。
- 起動テンプレートまたは起動設定に対応したセキュリティグループ。セキュリティグループは、リスナーポート (HTTP トラフィックの場合は通常、ポート 80) および Elastic Load Balancing でヘルスチェックに使用するポートの両方で、ロードバランサーからのアクセスが許可されている必要があります。詳細については、該当するドキュメントを参照してください。
  - Application Load Balancer のユーザーガイド の「ターゲットセキュリティグループ」
  - Network Load Balancer のユーザーガイドの「ターゲットセキュリティグループ」

  インスタンスにパブリック IP アドレスがある場合は、オプションでインスタンスに接続するための SSH トラフィックを許可することもできます。
- (オプション) アプリケーションにアクセス権を付与する IAM ロール。 AWS
- (オプション) Amazon EC2 インスタンスのソーステンプレートとして定義された Amazon マシンイメージ (AMI)。作成するには、インスタンスを起動します。IAM ロール (作成した場合) と、ユーザーデータとして必要な設定スクリプトを指定します。インスタンスに接続し、それをカスタマイズします。たとえば、ソフトウェアやアプリケーションのインストール、データのコピー、追加の EBS ボリュームのアタッチを行うことができます。インスタンスが正しく設定されたことを確認するために、インスタンスでアプリケーションをテストします。この更新された設定をカスタム AMI として保存します。後で使用しないインスタンスは、終了できます。この新しいカスタム AMI から起動されるインスタンスには、AMI の作成時に追加したカスタマイズが含まれます。
- Virtual Private Cloud (VPC)。このチュートリアルではデフォルトの VPC を参照していますが、独自の VPC を使用することもできます。後者の場合は、VPC に、作業中のリージョンの各アベイラビリティーゾーンにマッピングされたサブネットがあることを確認してください。ロードバランサーを作成するには、2 つ以上のパブリックサブネットが必要です。Auto Scaling グループを作成してロードバランサーに登録するには、2 つのプライベートサブネットまたは 2 つのパブリックサブネットが必要です。

ステップ 1: 起動テンプレートまたは起動設定を設定する

このチュートリアルでは、起動テンプレートまたは起動設定を使用します。

トピック

- 起動テンプレートを選択または作成します。
- 起動設定を作成または選択する

起動テンプレートを選択または作成します。

使用する起動テンプレートが既に存在している場合は、以下の手順を使用して起動テンプレートを選択します。

既存の起動テンプレートを選択するには

1. Amazon EC2 コンソールの起動テンプレートページを開きます。
2. 画面上部のナビゲーションバーで、ロードバランサーが作成されたリージョンを選択します。
3. 起動テンプレートを削除します。
4. Actions、[Auto Scaling グループの作成] を選択します。

または、新しい起動テンプレートを作成するために、次の手順を使用します。

起動テンプレートを作成するには

1. Amazon EC2 コンソールの起動テンプレートページを開きます。
2. 画面上部のナビゲーションバーで、ロードバランサーが作成されたリージョンを選択します。
3. [起動テンプレートの作成] を選択します。
4. 名前を入力し、起動テンプレートの最初のバージョンの説明を加えます。
5. [Application and OS Images (Amazon Machine Image)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で、インスタンス用の AMI の ID を選択します。利用可能なすべての AMI を検索するか、[Recent] (最新) または [クイックスタート] リストから選択できます。必要な AMI が表示されない場合は、[その他の AMI を閲覧する] を選択して、完全な AMI カタログを参照します。
6. [インスタンスタイプ] には、指定した AMI と互換性のあるインスタンス用のハードウェア設定を選択します。

7. (オプション) [キーペア (ログイン)] で、インスタンスに接続するときに使用するキーペアを選択します。
8. [Network settings] (ネットワーク設定) で、[Advanced network configuration] (高度なネットワーク設定) を展開し、以下を実行します。
   a. [Add network interface] (ネットワークインターフェイスを追加) を選択して、プライマリネットワークインターフェイスを追加します。
   b. Auto-assign public IP では、インスタンスがパブリック IPv4 アドレスを受け取るかどうかを指定します。デフォルトでは、EC2 インスタンスがデフォルトのサブネットで起動された場合、またはパブリック IPv4 アドレスを自動的に割り当てるように設定されたサブネットでインスタンスが起動された場合、Amazon EC2 はパブリック IPv4 アドレスを割り当てます。インスタンスに接続する必要がない場合は、[Disable] を選択して、グループ内のインスタンスがインターネットから直接トラフィックを受信しないようにできます。この場合、トラフィックの受信はロードバランサーからのみになります。
   c. セキュリティグループ ID を使用する場合、ロードバランサーと同じ VPC のインスタンスのセキュリティグループを指定します。
   d. [Delete on termination (終了時に削除)] を使用する場合、Yes (はい)を選択してください。これにより、Auto Scaling グループをスケールインしてネットワークインターフェイスがアタッチされているインスタンスを終了するときに、ネットワークインターフェイスが削除されます。
9. (オプション) 認証情報をインスタンスに安全に配布するには、[Advanced details (詳細設定)] の [IAM instance profile (IAM インスタンスプロファイル)] に、IAM ロールの Amazon リソースネーム (ARN) を入力します。
10. (オプション) インスタンスにユーザーデータまたは設定スクリプトを指定するには、[Advanced details]、[User data] に設定スクリプトを貼り付けます。
11. [起動テンプレートの作成] を選択します。
12. 確認ページで、[Auto Scaling グループの作成] を選択します。

起動設定を作成または選択する

> **Note**
>
> 新しいアプリケーションでは起動設定を使用しないことを強くお勧めします。なぜなら、起動設定は従来の機能であり、計画的な投資もないからです。また、2023 年 6 月 1 日以降に

作成された新しいアカウントには、コンソールから新しい起動設定を作成することはできません。詳細については、「起動設定」を参照してください。

既存の起動設定を選択するには

1. Amazon EC2 コンソールの [起動設定ページ] を開きます。
2. 上部のナビゲーションバーで、ロードバランサーが作成されたリージョンを選択します。
3. 起動設定を選択します。
4. Actions、[Auto Scaling グループの作成] を選択します。

または、新しい起動設定を作成するために、次の手順を使用します。

起動設定を作成するには

1. Amazon EC2 コンソールの [起動設定ページ] を開きます。確認を求めるプロンプトが表示されたら、[起動設定を表示] を選択して、[起動設定] ページを表示することを確認します。
2. 上部のナビゲーションバーで、ロードバランサーが作成されたリージョンを選択します。
3. [Create launch configuration (起動設定の作成)] を選択して、起動設定の名前を入力します。
4. [Amazon マシンイメージ (AMI)] には、検索基準としてインスタンスの AMI の ID を入力します。
5. [インスタンスタイプ] では、インスタンスのハードウェア設定を選択します。
6. [Additional configuration (追加設定)] 以下のフィールドに注意してください。

    a. (オプション) 認証情報を EC2 インスタンスに安全に配布するには、[IAM インスタンスプロファイル] で、IAM ロールを選択します。詳細については、「Amazon EC2 インスタンスで実行中のアプリケーション用の IAM ロール」を参照してください。

    b. (オプション) インスタンスにユーザーデータまたは設定スクリプトを指定するには、[Advanced details (高度な詳細)]、[User data] に設定スクリプトを貼り付けます。

    c. (オプション) [Advanced details (高度な詳細)] の [IP アドレスタイプ] は、デフォルト値のままにします。Auto Scaling グループを作成する際には、デフォルト VPC 内のデフォルトサブネットなど、パブリック IP アドレッシング属性が有効なサブネットを使用することにより、Auto Scaling グループ内のインスタンスにパブリック IP アドレスを割り当てることができます。または、インスタンスに接続する必要がない場合は、グループ内のインスタンスがインターネットから直接トラフィックを受信することのないよう、[パブリック IP アド

レスをどのインスタンスにも割り当てないでください] を選択することもできます。この場合、トラフィックの受信はロードバランサーからのみになります。

7. セキュリティグループを使用する場合、ロードバランサーと同じ VPC から既存のセキュリティグループを選択します。Create a new security group (新しいセキュリティグループの作成)オプションを選択すると、Linux を実行する Amazon EC2 インスタンスに対してデフォルトの SSH ルールが設定されます。デフォルトの RDP ロールは、Windows を実行する Amazon EC2 インスタンスに対して設定されます。
8. [Key pair (login) (キーペア (ログイン)] で、[Key pair options (キーペアのオプション)] の下にあるオプションを選択します。

   すでに Amazon EC2 インスタンスキーペアを設定している場合は、ここで選択できます。

   Amazon EC2 インスタンスのキーペアがまだない場合は、[Create a new key pair (新しいキーペアの作成)] を選択して、わかりやすい名前を付けます。[Download key pair (キーペアのダウンロード)] を選択し、コンピュータにダウンロードします。

   > **Important**
   >
   > インスタンスに接続する必要がある場合は、[Proceed without a key pair (キーペアなしで続行する)] を選択しないでください。

9. 確認チェックボックスをオンにし、[Create launch configuration (起動設定の作成)] を選択します。
10. 新しい起動設定の名前の横にあるチェックボックスを選択し、アクション,Auto Scaling グループの作成を選択してください。

ステップ 2: Auto Scaling グループを作成する

起動テンプレートまたは起動設定を作成または選択した後、以下の手順に従って中断していた作業を続行します。

Auto Scaling グループを作成する

1. [Choose launch template or configuration (起動テンプレートまたは起動設定を選択する)] ページで [Auto Scaling グループ名] にAuto Scaling グループの名前を入力します。

2. [Launch template only] [起動テンプレート] で、スケールアウト時に Auto Scaling グループで使用する起動テンプレートのバージョン (デフォルト、最新、または特定のバージョン) を選択します。
3. [次へ] をクリックします。

   [Choose instance launch options] (インスタンス起動オプションの選択) ページが表示されます。このページでは、Auto Scaling グループで使用する VPC ネットワーク設定を選択し、オンデマンドインスタンスとスポットインスタンスを起動するためのオプションを選択できます (起動テンプレートを選択した場合)。
4. [Network] (ネットワーク) セクションの [VPC] で、ロードバランサーに使用した VPC を選択します。デフォルトの VPC を選択した場合は、インスタンスへのインターネット接続を提供するように自動的に設定されます。この VPC には、リージョンの各アベイラビリティーゾーンのパブリックサブネットが含まれます。
5. [Availability Zones and subnets] (アベイラビリティゾーンとサブネット) で、ロードバランサーがあるアベイラビリティーゾーンに基づいて、含める各アベイラビリティーゾーンから 1 つ以上のサブネットを選択します。詳細については、「VPC サブネットを選択する場合の考慮事項」を参照してください。
6. [起動テンプレートのみ] [Instance type requirements] (インスタンスタイプの要件) セクションで、デフォルト設定を使用して、この手順を簡略化します。(起動テンプレートを上書きしないでください。) このチュートリアルでは、起動テンプレートで指定されたインスタンスタイプを使用して、オンデマンドインスタンスのみを起動します。
7. [Next] (次へ) を選択して、[Configure advanced options] (詳細オプションの設定) ページに移動します。
8. グループを既存のロードバランサーにアタッチするには、[Load balancing] (ロードバランシング) セクションで [Attach to an existing load balancer] (既存のロードバランサーにアタッチする) を選択します。[Choose from your load balancer target groups] (ロードバランサーのターゲットグループから選択する) または [Choose from Classic Load Balancers] (Classic Load Balancer から選択する) も選択できます。その後、作成した Application Load Balancer または Network Load Balancer のターゲットグループの名前を選択するか、Classic Load Balancer の名前を選択できます。
9. (オプション) Elastic Load Balancing ヘルスチェックを使用するには、[ヘルスチェック] で、ヘルスチェックタイプの[ELB] を選択します。
10. Auto Scaling グループの設定が完了したら、[Skip to review (スキップして確認)] を選択します。
11. [確認] ページで、Auto Scaling グループの詳細を確認します。[Edit] を選択して、変更を加えることができます。完了したら、[Auto Scaling グループの作成] を選択します。

ロードバランサーがアタッチされた Auto Scaling グループを作成すると、ロードバランサーは新しいインスタンスがオンラインになると自動的に登録します。この時点ではインスタンスが 1 つしかないため、登録するインスタンスは多くありません。ただし、グループの希望キャパシティーを更新することで、インスタンスを追加できるようになりました。 step-by-step 手順については、を参照してくださいAuto Scaling グループの希望する容量を変更する。

ステップ 3: ロードバランサーがアタッチされたことを確認する

ロードバランサーがアタッチされたことを確認するには

1. Amazon EC2 コンソールの Auto Scaling グループページ から、Auto Scaling グループの横にあるチェックボックスを選択します。
2. [詳細] タブの [Load balancing] には、アタッチされているロードバランサーターゲットグループまたは Classic Load Balancer が表示されます。
3. [アクティビティ]タブのアクティビティ履歴で、インスタンスが正常に起動したことを確認できます。[Status] 列は Auto Scaling グループがインスタンスを正常に起動したかどうかを表示します。インスタンスの起動に失敗した場合は、Amazon EC2 Auto Scaling をトラブルシューティングする にインスタンスの起動に関する一般的な問題のトラブルシューティングのヒントがあります。
4. [インスタンス管理] タブの [インスタンス] で、インスタンスがトラフィックを受け取る準備ができたことを確認できます。当初、インスタンスの状態は `Pending` です。インスタンスがトラフィックを受信できるようになったら、そのステータスは `InService` です。[Health Status] 列には、インスタンスのAmazon EC2 Auto Scaling ヘルスチェックの結果が表示されます。インスタンスが正常とマークされていても、ロードバランサーは、ロードバランサーのヘルスチェックに合格したインスタンスにのみ、トラフィックを送信します。
5. インスタンスがロードバランサーに登録されていることを確認します。Amazon EC2 コンソールのターゲットグループページを開きます。ターゲットグループを選択し、[ターゲット] タブを選択します。インスタンスの状態が `initial` の場合、おそらく登録中であるか、まだヘルスチェック中です。インスタンスの状態が `healthy` になると、使用できる状態です。

ステップ 4: 次のステップ

このチュートリアルを完了したので、さらに詳しい内容に進むことができます。

- Amazon EC2 Auto Scaling は、Auto Scaling グループが使用するヘルスチェックのステータスに基づいてインスタンスが正常かどうかを判断します。ロードバランサーのヘルスチェックを有効に

し、インスタンスがヘルスチェックに失敗した場合、Auto Scaling グループはそのインスタンスを異常と見なして置き換えます。詳細については、「ヘルスチェック」を参照してください。

- 同じリージョン内の追加のアベイラビリティーゾーンにアプリケーションを拡張して、サービス中断が発生した場合の耐障害性を向上させることができます。詳細については、「アベイラビリティーゾーンを追加する」を参照してください。
- Auto Scaling グループを設定することで、ターゲット追跡スケーリングポリシーを使用できます。これにより、インスタンスの需要の変化に応じて、インスタンスの数が自動的に増減します。これにより、グループはアプリケーションが受信するトラフィック量の変化に対応できます。詳細については、「ターゲット追跡スケーリングポリシー」を参照してください。

ステップ 5：クリーンアップ

このチュートリアル用に作成したリソースを使用し終えたら、不要な料金の発生を回避するため、クリーンアップを検討してください。

Auto Scaling グループを削除するには

1. Amazon EC2 コンソールで Auto Scaling グループのページを開きます。
2. Auto Scaling グループの横にあるチェックボックスを選択します。
3. [削除] をクリックします。
4. 確認を求められたら、**delete** を入力して指定された Auto Scaling グループの削除を確認し、[Delete] (削除) を選択します。

   [Name (名前)] 列のロードアイコンに、Auto Scaling グループが削除されたことが示されます。削除が行われると、[Desired] (必要)、[Min] (最小)、[Max] (最大) 列には、Auto Scaling グループのインスタンス数として `0` と表示されます。インスタンスを終了し、グループを削除するには数分かかります。リストを更新して、現在の状態を確認します。

起動テンプレートを維持する場合は、この手順をスキップします。

起動テンプレートを削除するには

1. Amazon EC2 コンソールの起動テンプレートページを開きます。
2. 起動テンプレートを選択します。
3. [アクション]、[テンプレートの削除] の順に選択します。

4. 確認を求められたら、**Delete** を入力して指定した起動テンプレートの削除を確認し、[Delete] (削除) を選択します。

起動設定を維持する場合は、以下の手順をスキップします。

起動設定を削除するには

1. Amazon EC2 コンソールの [起動設定ページ] を開きます。
2. 起動設定を選択します。
3. [Actions]、[Delete launch configuration] の順に選択します。
4. 確認を求めるメッセージが表示されたら、[削除] を選択します。

将来使用できるように、ロードバランサーを保持する場合は、次の手順をスキップします。

ロードバランサーを削除するには

1. Amazon EC2 コンソールで [ロードバランサーページ] を開きます 。
2. ロードバランサーを選択してから、[Actions (アクション)]、[Delete (削除)] の順に選択します。
3. 確認を求めるメッセージが表示されたら、[Yes、Delete] を選択します。

ターゲットグループを削除するには

1. Amazon EC2 コンソールのターゲットグループページを開きます。
2. ターゲットグループを選択し、[Actions (アクション)]、[Delete (削除)] を選択します。
3. 確認を求めるメッセージが表示されたら、[Yes、Delete] を選択します。

関連リソース

AWS CloudFormationでは、テンプレートファイルを使用してリソースのコレクションを 1 つのユニット (スタック) としてまとめて作成および削除することで、 AWS インフラストラクチャデプロイを予測どおりに繰り返し作成およびプロビジョニングできます。詳細については、『AWS CloudFormation ユーザーガイド』を参照してください。

スタックテンプレートを使用して、Auto Scaling グループと Application Load Balancer をプロビジョニングする方法のチュートリアルは、「AWS CloudFormation ユーザーガイド」の「チュートリアル: スケーラブルなロードバランシングウェブサーバーの作成」を参照してください。このチュー

トリアルとサンプルテンプレートは、実際のニーズに合わせて、同様のテンプレートを作成する際の開始点として使用できます。

Amazon EC2 Auto Scaling 起動テンプレート

起動テンプレートは、インスタンス設定情報を指定する起動設定と似ています。Amazon マシンイメージ (AMI) の ID、インスタンスタイプ、キーペア、セキュリティグループ、その他 EC2 インスタンスを起動するために使用するパラメータが含まれています。ただし、起動設定の代わりに起動テンプレートを定義すると、複数のバージョンの起動テンプレートを使用することができます。

起動テンプレートのバージョン管理では、パラメータのフルセットのサブセットを作成できます。その後、再使用して、同じ起動テンプレートの他のバージョンを作成できます。たとえば、AMI またはユーザーデータスクリプトを使用せずに、基本設定を定義する起動テンプレートを作成できます。起動テンプレートを作成したら、新しいバージョンを作成し、アプリケーションの最新バージョンを持つ AMI とユーザーデータをテスト用に追加できます。これにより、起動テンプレートのバージョンが 2 つになります。基本構成を保存すると、必要な一般構成パラメータを維持するのに役立ちます。基本設定から起動テンプレートの新しいバージョンを、必要に応じて作成することができます。アプリケーションのテストに使用されたバージョンも、不要になったら削除することができます。

最新の機能や改善点にアクセスできるように、起動テンプレートを使用することをお勧めします。起動設定を使用する場合、すべての Amazon EC2 Auto Scaling 機能を使用できるわけではありません。たとえば、スポットインスタンスとオンデマンドインスタンスの両方を起動する Auto Scaling グループや、複数のインスタンスタイプを指定する Auto Scaling グループを作成することはできません。これらの機能を設定するには、起動テンプレートを使用する必要があります。詳細については、「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。

起動テンプレートを使用すると、Amazon EC2 の新しい機能を使用することもできます。これには、Systems Manager パラメータ (AMI ID)、現行世代の EBS プロビジョンド IOPS ボリューム (io2)、EBS ボリュームのタグ付け、T2 Unlimited インスタンス、キャパシティ予約、Capacity Blocks、専有ホストなどが含まれます。

起動テンプレートを作成するときは、すべてのパラメータはオプションです。ただし、起動テンプレートで AMI が指定されていない場合、Auto Scaling グループの作成時に AMI を追加することはできません。AMI を指定してもインスタンスタイプを指定しない場合は、Auto Scaling グループの作成時に 1 つ以上のインスタンスタイプを追加できます。

内容

- 起動テンプレートを操作するアクセス許可
- 起動テンプレートでサポートされている API オペレーション

- Auto Scaling グループの起動テンプレートを作成する
- 詳細設定を使用して起動テンプレートを作成する
- Auto Scaling グループを起動テンプレートに移行する
- スタックを起動テンプレートに移行する AWS CloudFormation
- を使用した起動テンプレートの作成と管理の例 AWS CLI
- 起動テンプレートで AMI IDs の代わりに AWS Systems Manager パラメータを使用する

起動テンプレートを操作するアクセス許可

このセクションの手順では、起動テンプレートを作成するために必要なアクセス許可が既にあることを前提としています。管理者がアクセス許可を付与する方法の詳細については、「Amazon EC2 ユーザーガイド」の「IAM アクセス許可を使用して起動テンプレートへのアクセスを制御する」を参照してください。 Amazon EC2

起動テンプレートで指定されたリソースを使用および作成するための十分なアクセス許可がない場合、Auto Scaling グループに指定しようとすると、起動テンプレートを使用する権限がないというエラーが表示されることに注意してください。詳細については、「Amazon EC2 Auto Scaling をトラブルシューティングする: 起動テンプレート」を参照してください。

起動テンプレートを使用して `CreateAutoScalingGroup`、`UpdateAutoScalingGroup`、および `RunInstances` API オペレーションを呼び出すことができる IAM ポリシーの例については、「」を参照してください起動テンプレートのサポート。

起動テンプレートでサポートされている API オペレーション

起動テンプレートでサポートされている API オペレーションのリストについては、「Amazon EC2 API Reference」の「Amazon EC2 actions」を参照してください。

Auto Scaling グループの起動テンプレートを作成する

起動テンプレートを使用して Auto Scaling グループを作成する前に、Amazon マシンイメージ (AMI) の ID など、インスタンスを起動するための設定情報を含む起動テンプレートを作成する必要があります。

新しい起動テンプレートを作成するには、次の手順を使用します。

内容

- 起動テンプレートを作成する (コンソール)
- デフォルトのネットワークインターフェイス設定を変更する (コンソール)
- ストレージ設定を変更する (コンソール)
- 既存のインスタンスから起動テンプレートを作成する (コンソール)
- 関連リソース
- 制限事項

> **Important**
>
> 起動テンプレートパラメータは、起動テンプレート作成の際には完全には検証されません。パラメータに誤った値を指定した場合、またはサポートされているパラメータの組み合わせを使用しない場合、この起動テンプレートを使用してインスタンスは起動できません。パラメータに正しい値を指定したこと、およびサポートされているパラメータの組み合わせを使用していることを確認します。例えば、ARM ベースの AWS Graviton or Graviton2 AMI を含むインスタンスを起動するには、ARM 互換のインスタンスタイプを指定する必要があります。詳細については、「Amazon EC2 ユーザーガイド」の「起動テンプレートの制限」を参照してください。 Amazon EC2

起動テンプレートを作成する (コンソール)

次の手順では、基本的な起動テンプレートを設定する方法について説明します。

- インスタンスを起動する Amazon マシンイメージ (AMI) を選択します。
- 指定した AMI と互換性のあるインスタンスタイプを選択します。
- インスタンスへの接続時 (SSH を使用) に使用するキーペアを指定します。
- 1 つ以上のセキュリティグループを追加して、インスタンスへのネットワークアクセスを許可します。
- 各インスタンスに追加のボリュームをアタッチするかどうかを指定します。
- インスタンスおよびボリュームにカスタムタグ (キーバリューペア) を追加する。

起動テンプレートを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインで、[インスタンス] の [テンプレートの起動] を選択します。
3. [起動テンプレートの作成] を選択します。名前を入力し、起動テンプレートの最初のバージョンの説明を加えます。
4. (オプション) [Auto Scaling ガイダンス] でチェックボックスを選択すると、Amazon EC2 Auto Scaling で使用するテンプレートの作成に役立つガイダンスが Amazon EC2 で表示されます。
5. [Launch template contents] (起動テンプレートのコンテンツ) で各必須フィールドに入力し、必要に応じてオプションフィールドにも入力します。

   a. [アプリケーションおよび OS イメージ (Amazon マシンイメージ) 情報}: (必須) インスタンス用の AMI の ID を選択します。利用可能なすべての AMI を検索するか、[Recent] (最新) または [クイックスタート] リストから選択できます。必要な AMI が表示されない場合は、[その他の AMI を閲覧する] を選択して、完全な AMI カタログを参照します。

      カスタム AMI を選択するには、最初に、カスタマイズしたインスタンスから AMI を作成する必要があります。詳細については、「Amazon EC2 ユーザーガイド」の「AMI の作成」を参照してください。 Amazon EC2

   b. [インスタンスタイプ] には、指定した AMI との互換性がある単一のインスタンスタイプを選択します。

      あるいは、属性ベースのインスタンスタイプの選択を使用するには、[詳細]、[インスタンスタイプの属性を指定] を選択し、次のオプションを指定します。

      - [Number of vCPUs] (vCPU の数): vCPU の最小数と最大数を入力します。制限を設定しない場合は、最小値に 0 を入力し、最大値を空白のままにします。
      - [Amount of memory (MiB)] (メモリの量 (MiB)): メモリの最小量と最大量を MiB 単位で入力します。制限を設定しない場合は、最小値に 0 を入力し、最大値を空白のままにします。
      - [Optional instance type attributes] (オプションのインスタンスタイプ属性) を展開して [Add attribute] (属性の追加) を選択し、目的のキャパシティーを満たすために使用できるインスタンスのタイプをさらに絞り込みます。各属性の詳細については、Amazon EC2 API リファレンス」のInstanceRequirements「リクエスト」を参照してください。
      - [Resulting instance types] (得られるインスタンスタイプ): vCPU、メモリ、ストレージなど、指定されたコンピューティング要件に適合するインスタンスタイプを表示できます。
      - インスタンスタイプを除外するには、[属性の追加] を選択します。[属性] リストから [除外されたインスタンスタイプ] を選択します。[Attribute value] (属性値) リストから、除外したいインスタンスタイプを選択します。

c. [キーペア (ログイン)]: [キーペア名] で既存のキーペアを選択するか、[新しいキーペアの作成] を選択して新しいキーペアを作成します。詳細については、Amazon EC2 ユーザーガイド」の「Amazon EC2 キーペアと Linux インスタンスAmazon EC2」を参照してください。

d. [ネットワーク設定]: [ファイアウォール (セキュリティグループ)] で 1 つ以上のセキュリティグループを使用するか、空白のままにして、ネットワークインターフェイスの一部として 1 つ以上のセキュリティグループを設定します。詳細については、「Amazon EC2 ユーザーガイド」の「Amazon EC2 security groups for Linux instances」(Linux インスタンス用の Amazon EC2 セキュリティグループ) を参照してください。

   起動テンプレートでセキュリティグループを指定しない場合、Amazon EC2 は、Auto Scaling グループがインスタンスを起動する VPC のデフォルトのセキュリティグループを使用します。デフォルトでは、このセキュリティグループは外部ネットワークからのインバウンドトラフィックを許可しません。詳細については、Amazon VPC ユーザーガイドの「VPC のデフォルトセキュリティグループ」を参照してください。

e. 次のいずれかを行います。

   - デフォルトのネットワークインターフェイス設定を変更します。例えば、サブネット上の自動割り当てパブリック IPv4 アドレス設定を上書きするパブリック IPv4 アドレス指定機能を有効または無効にできます。詳細については、「デフォルトのネットワークインターフェイス設定を変更する (コンソール) 」を参照してください。
   - デフォルトのネットワークインターフェイス設定を維持するには、このステップをスキップします。

f. 次のいずれかを行います。

   - ストレージ設定を変更します。詳細については、「ストレージ設定を変更する (コンソール) 」を参照してください。
   - デフォルトのストレージ設定を保持するには、このステップをスキップします。

g. [Resource tags] (リソースタグ) には、キーと値の組み合わせを入力してタグを指定します。起動テンプレートでインスタンスタグを指定して、Auto Scaling グループのタグをそのインスタンスに伝播することを選択した場合、すべてのタグがマージされます。起動テンプレートのタグと Auto Scaling グループのタグに同じタグキーが指定されている場合、グループのタグ値が優先されます。

6. 詳細設定を構成します (オプション)。たとえば、アプリケーションが他の AWS リソースにアクセスするときに使用できる IAM ロールを選択できます。または、インスタンスの起動後に一般的な自動設定タスクを実行するために使用できるインスタンスユーザーデータを指定できます。詳細については、「詳細設定を使用して起動テンプレートを作成する」を参照してください。

7. 起動テンプレートを作成する準備ができたら、[起動テンプレートを作成] を選択します。
8. Auto Scaling グループを作成するには、[confirmation (確認)] ページで[Auto Scaling グループの作成] を選択します。

デフォルトのネットワークインターフェイス設定を変更する (コンソール)

ネットワークインターフェイスは、VPC およびインターネット内の他のリソースへの接続を提供します。詳細については、「Amazon VPC を使用して Auto Scaling インスタンスにネットワーク接続を提供する」を参照してください。

このセクションでは、デフォルトのネットワークインターフェイス設定を変更する方法について説明します。例えば、サブネット上の自動割り当てパブリック IPv4 設定をデフォルトにする代わりに、パブリック IPv4 アドレスを各インスタンスに割り当てるかどうかを定義できます。

考慮事項と制約事項

デフォルトのネットワークインターフェイス設定を変更する場合、次の考慮事項と制約事項に留意してください。

- セキュリティグループは、テンプレートの [セキュリティグループ] セクションではなく、ネットワークインターフェイスの一部として設定する必要があります。両方の場所でセキュリティグループを指定することはできません。
- ネットワークインターフェイスに 2 番目のプライベート IP アドレス (セカンダリ IP アドレス) を割り当てることはできません。
- 既存のネットワークインターフェイス ID を指定した場合、起動できるインスタンスは 1 つだけです。これを行うには、 AWS CLI または SDK を使用して Auto Scaling グループを作成する必要があります。グループを作成するときは、アベイラビリティーゾーンを指定する必要がありますが、サブネット ID は指定しないでください。また、デバイスインデックスが 0 の場合にのみ、既存のネットワークインターフェイスを指定できます。
- 複数のネットワークインターフェイスを指定した場合、パブリック IPv4 アドレスを自動割り当てすることはできません。また、ネットワークインターフェイス間で重複するデバイスインデックスを指定することもできません。プライマリとセカンダリの両方のネットワークインターフェイスは同じサブネットに存在します。
- インスタンスが起動すると、各ネットワークインターフェイスにプライベートアドレスが自動的に割り当てられます。アドレスは、インスタンスが起動したサブネットの CIDR 範囲から取得されます。VPC またはサブネットの CIDRブロック (IP アドレス範囲) の指定の詳細については、Amazon VPC ユーザーガイドを参照してください。

デフォルトのネットワークインターフェイス設定を変更するには

1. [ネットワーク設定] で [高度なネットワーク設定] を展開します。
2. [ネットワークインターフェイスを追加] を選択して、プライマリネットワークインターフェイスを設定します。以下のフィールドに注意してください。

   a. デバイスインデックス: プライマリネットワークインターフェイス (eth0) に変更を適用するには、デフォルト値の 0 のままにします。

   b. ネットワークインターフェイス: インスタンスが起動したときに Amazon EC2 Auto Scaling で新しいネットワークを自動的に作成するには、デフォルト値 ([新しいインターフェイス]) のままにします。デバイスインデックスが 0 の既存の利用可能なネットワークインターフェイスを選択できます。その場合、Auto Scaling グループは 1 つのインスタンスに制限されます。

   c. 説明: 分かりやすい名前を入力します (オプション)。

   d. サブネット: デフォルトの [[起動テンプレートの設定に含めない] のままにします。

      AMI によってネットワークインターフェイスのサブネットが指定された場合、エラーが発生します。回避策として、[Auto Scaling ガイダンス] をオフにすることが推奨されます。この変更を行った後は、エラーメッセージは表示されません。ただし、サブネットが指定されている場所に関係なく、Auto Scaling グループのサブネット設定が優先され、上書きすることはできません。

   e. パブリック IP の自動割り当て: デバイスインデックスが 0 のネットワークインターフェイスがパブリック IPv4 アドレスを受け取るかどうかを変更します。デフォルトでは、デフォルトのサブネットにあるインスタンスはパブリック IPv4 アドレスを受け取りますが、デフォルト以外のサブネットにあるインスタンスは受け取りません。[Enable] または [Disable] を選択すると、これによりサブネットのデフォルト設定がオーバーライドされます。

   f. セキュリティグループ: ネットワークインターフェイスに対して 1 つ以上のセキュリティグループを選択します。各セキュリティグループは、Auto Scaling グループがインスタンスを起動する VPC 用に設定する必要があります。詳細については、「Amazon EC2 ユーザーガイド」の「Amazon EC2 security groups for Linux instances」(Linux インスタンス用の Amazon EC2 セキュリティグループ) を参照してください。

   g. 終了時に削除: インスタンスが終了したときにネットワークインスタンスを削除するには、[はい] を選択します。ネットワークインスタンスを維持するには、[いいえ] を選択します。

   h. Elastic Fabric Adapter : 高性能コンピューティングと機械学習のユースケースをサポートするには、ネットワークインターフェイスを Elastic Fabric Adapter ネットワークインター

フェイスに変更します。詳細については、「Amazon EC2 ユーザーガイド」の「Elastic Fabric Adapter」を参照してください。 Amazon EC2

i. ネットワークカードのインデックス: デバイスインデックスが 0 のネットワークカードにプライマリネットワークインターフェイスをアタッチするには [0] を選択します。このオプションが使用できない場合は、デフォルト値 ([起動テンプレートに含めない]) のままにします。特定のネットワークカードへのネットワークインターフェイスのアタッチは、サポートされているインスタンスタイプでのみ使用できます。詳細については、「Amazon EC2 ユーザーガイド」の「ネットワークカード」を参照してください。 Amazon EC2

j. ENA Express : ENA Express をサポートするインスタンスタイプでは、Enable を選択して ENA Express を有効にするか、Disable を選択して無効にします。詳細については、「Amazon EC2 ユーザーガイド」の「Linux インスタンスでの ENA Express によるネットワークパフォーマンスの向上」を参照してください。 Amazon EC2

k. ENA Express UDP : ENA Express を有効にすると、オプションで UDP トラフィックに使用できます。Enable を選択して ENA Express UDP を有効にするか、Disable を選択して無効にします。

3. セカンダリネットワークインターフェイスを追加するには、ネットワークインターフェイスを追加するを選択します。

ストレージ設定を変更する (コンソール)

Amazon EBS-backed AMI または instance store-backed AMI から起動するインスタンスのストレージ設定を変更できます。インスタンスにアタッチする追加の EBS ボリュームを指定することもできます。AMI には、ルートボリュームである [Volume 1 (AMI Root)] (ボリューム 1 (AMI Root) を始めとする 1 つまたは複数のストレージボリュームが含まれます。

ストレージ設定を変更するには

1. [Configure storage] (ストレージの設定) でボリュームのサイズまたはタイプを変更します。

   ボリュームのサイズに指定した値がボリュームのタイプの制限外である場合、またはスナップショットのサイズより小さい場合は、エラーメッセージが表示されます。この問題に対処するために、このメッセージには、フィールドに入力できる最小値または最大値が示されます。

   Amazon EBS-backed AMI に関連付けられているボリュームのみが表示されます。instance store-backed AMI から起動するインスタンスのストレージ設定に関する情報を表示するには、[Instance store volumes] (インスタンスストアボリューム) セクションの [詳細を表示] を選択します。

すべての EBS ボリュームパラメータを指定するには、右上隅のリンクから [アドバンスト] ビューに切り替えます。

2. アドバンストオプションで、変更するボリュームを展開し、次のようにボリュームを構成します。

   a. [Storage type] (ストレージタイプ): インスタンスと関連付けるボリュームのタイプ (EBS またはエフェメラル) です。インスタンスストア (エフェメラル) ボリュームタイプは、それをサポートするインスタンスタイプを選択した場合にのみ使用できます。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリューム」および「Amazon EC2 ユーザーガイド」の「Amazon EC2 インスタンスストアAmazon EC2」を参照してください。

   b. [Device name] (デバイス名): ボリュームで利用できるデバイス名の一覧から選択します。

   c. [Snapshot] (スナップショット): ボリュームの作成元となるスナップショットを選択します。[Snapshot] (スナップショット) フィールドにテキストを入力して、利用できる共有スナップショットとパブリックスナップショットを検索することもできます。

   d. [Size (GiB)] (サイズ (GiB)): EBS ボリュームの場合、ストレージサイズを指定できます。無料利用枠の対象となる AMI とインスタンスを選択した場合でも、無料利用枠内に収めるには、合計ストレージを 30 GiB 以下に維持する必要があることに注意してください。詳細については、「Amazon EBS ユーザーガイド」の「EBS ボリュームのサイズと設定の制限」を参照してください。

   e. ボリュームタイプ: EBS ボリュームのボリュームタイプを選択します。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームの種類」を参照してください。

   f. [IOPS]: プロビジョンド IOPS SSD (io1 と io2) あるいは汎用 SSD (gp3) ボリュームタイプを選択した場合、そのボリュームでサポートが可能な 1 秒あたりの I/O オペレーション数 (IOPS) を入力します。これは、io1、io2、gp3 ボリュームに必要です。gp2、st1、sc1、またはスタンダードボリュームではサポートされていません。

   g. 終了時に削除: EBS ボリュームで、インスタンスの終了時にボリュームを削除する場合は [Yes] (はい) を選択します。ボリュームを維持する場合は [No] (いいえ) を選択します。

   h. [Encrypted] (暗号化): インスタンスタイプが EBS 暗号化をサポートしている場合、[Yes] (はい) を選択し、ボリュームの暗号化を有効にできます。このリージョンでデフォルトで暗号化を有効にした場合、暗号化は有効になります。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS 暗号化」および「デフォルトで暗号化を有効にする」を参照してください。

以下の表に示すように、このパラメータを設定した場合のデフォルトの効果は、選択したボリュームのソースによって異なります。いずれの場合も、指定された を使用するためのアクセス許可が必要です AWS KMS key。

暗号化の結果

Encrypted パラメータの設定	ボリュームのソース	デフォルトの暗号化状態	メモ
いいえ	新しい (空の) ボリューム	暗号化されていない*	該当なし
	所有する暗号化されていないスナップショット	暗号化されていない*	
	お客様が所有する暗号化されたスナップショット	同じキーで暗号化されている	
	お客様と共有されている暗号化されていないスナップショット	暗号化されていない*	
	お客様と共有されている暗号化されたスナップショット	デフォルト KMS キーで暗号化	
はい	新しいボリューム	デフォルト KMS キーで暗号化	デフォルト以外の KMS キーを使用するには、[KMS キー] パラメータの値を指定します。
	所有する暗号化されていないスナップショット	デフォルト KMS キーで暗号化	
	お客様が所有する暗号化されたスナップショット	同じキーで暗号化されている	
	お客様と共有されている暗号化されていないスナップショット	デフォルト KMS キーで暗号化	

Encrypted パラメータの設定	ボリュームのソース	デフォルトの暗号化状態	メモ
	お客様と共有されている暗号化されたスナップショット	デフォルト KMS キーで暗号化	

* デフォルトで暗号化が有効な場合、新しく作成されたすべてのボリュームは ([暗号化済み] パラメータが [Yes] (はい) に設定されているかどうかに関係なく)、デフォルトの KMS キーを使用して暗号化されます。[暗号化済み] および [KMS キー] パラメータの両方を設定した場合、デフォルト以外の KMS キーを指定できます。

i. KMS キー: [暗号化済み] で [Yes] (はい) を選択した場合、ボリュームの暗号化に使用するカスタマーマネージド型キーを選択する必要があります。このリージョンでデフォルトの暗号化を有効にした場合は、自動的にデフォルトのカスタマーマネージド型キーが選択されます。別のキーを選択するか、 AWS Key Management Serviceを使用して作成したカスタマーマネージド型キーの ARN を指定できます。

3. この起動テンプレートによって起動されたインスタンスにアタッチする追加のボリュームを指定するには、[新しいボリュームを追加] を選択します。

既存のインスタンスから起動テンプレートを作成する (コンソール)

既存のインスタンスから起動テンプレートを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインの [Instances] (インスタンス) で、[Instances] (インスタンス) を選択します。
3. インスタンスを選び、[Actions (アクション)]、[Image and templates (イメージとテンプレート)]、[Create Template from Instance (インスタンスからテンプレートを作成)] の順に選択します。
4. 名前と説明を入力します。
5. [Auto Scaling ガイダンス] で、[チェックボックス] を選択します。
6. 必要に応じて設定を調整し、[起動テンプレートの作成] を選択します。

7. Auto Scaling グループを作成するには、[confirmation (確認)] ページで[Auto Scaling グループの作成] を選択します。

関連リソース

AWS CloudFormation スタックテンプレートで起動テンプレートを宣言する方法を理解するために使用できる JSON および YAML テンプレートスニペットがいくつか用意されています。詳細については、「 ユーザーガイド」のAWS::EC2::LaunchTemplate「」および「 セクションを含む起動テンプレートの作成 AWS CloudFormationAWS CloudFormation 」を参照してください。

起動テンプレートの詳細については、Amazon EC2 ユーザーガイド」の「起動テンプレートからのインスタンスの起動」を参照してください。

制限事項

- 起動テンプレートではサブネットを指定できますが、起動テンプレートを使用して Auto Scaling グループを作成するだけであれば、サブネットを指定する必要はありません。起動テンプレートでサブネットを指定して、Auto Scaling グループのサブネットを指定することはできません。Auto Scaling グループのサブネットは、Auto Scaling グループの独自のリソース定義から取得されます。
- ユーザー定義のネットワークインターフェースに関するその他の制限については、デフォルトのネットワークインターフェイス設定を変更する (コンソール) を参照してください。

詳細設定を使用して起動テンプレートを作成する

このトピックでは、 から詳細設定を使用して起動テンプレートを作成する方法について説明します AWS Management Console。

詳細設定を使用して起動テンプレートを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインのインスタンス で、起動テンプレート を選択し、起動テンプレートの作成 を選択します。
3. 次のトピックで説明するように、起動テンプレートを設定します。

   - 必須の設定
   - [詳細設定]

4. [起動テンプレートの作成] を選択します。

必須の設定

起動テンプレートを作成するときは、次の必須設定を含める必要があります。

起動テンプレート名

起動テンプレートを説明する一意の名前を入力します。

アプリケーションと OS イメージ (Amazon マシンイメージ)

使用する Amazon マシンイメージ (AMI) を選択します。使用する AMI を検索または参照できます。最適なスケーリング効率を得るには、アプリケーションコードでインスタンスを起動するように完全に設定されており、起動時に変更がほとんど必要なカスタム AMI を選択します。

インスタンスタイプ

AMI と互換性のあるインスタンスタイプを選択します。Auto Scaling グループ独自のリソース定義に埋め込まれている複数のインスタンスタイプを使用する場合は、起動テンプレートへのインスタンスタイプの追加をスキップできます。インスタンスタイプは、混合インスタンスグループを作成する予定がない場合にのみ必要です。

[詳細設定]

詳細設定はオプションです。詳細設定を行わない場合、特定の機能はインスタンスに追加されません。

[高度な詳細] セクションを展開して、詳細設定を表示します。以下のセクションでは、Auto Scaling グループの起動テンプレートを作成するときに焦点を当てる最も有用な詳細設定について説明します。詳細については、「Amazon EC2 https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/create-launch-template.html#lt-advanced-detailsユーザーガイド」の「詳細」を参照してください。Amazon EC2

IAM インスタンスプロファイル

インスタンスプロファイルには、使用する IAM ロールが含まれます。Auto Scaling グループが EC2 インスタンスを起動すると、関連付けられた IAM ロールで定義されたアクセス許可が、インスタンスで実行されているアプリケーションに付与されます。詳細については、「Amazon EC2 インスタンスで実行中のアプリケーション用の IAM ロール」を参照してください。

終了保護

この機能を有効にすると、ユーザーは Amazon EC2 コンソール、CLI コマンド、および API オペレーションを使用してインスタンスを終了できなくなります。終了保護は、偶発的な終了に対する追加の保護を提供します。Amazon EC2 Auto Scaling がインスタンスを終了することを妨げることはありません。Amazon EC2 Auto Scaling を終了できるインスタンスを制御するには、「」を参照してくださいインスタンスのスケールイン保護を使用する。

詳細 CloudWatch モニタリング

EC2 インスタンスの詳細モニタリングを有効にして、1 分間隔でメトリクスデータを Amazon CloudWatch に送信できます。デフォルトでは、EC2 インスタンスはメトリクスデータを CloudWatch 5 分間隔で に送信します。別途 料金がかかります。詳細については、「Auto Scaling インスタンスのモニタリングを設定する」を参照してください。

クレジット仕様

Amazon EC2 は、T2, T3T3a などのバーストパフォーマンスインスタンスを提供します。これにより、アプリケーションは必要に応じてベースライン CPU パフォーマンスを超えてバーストできます。デフォルトでは、これらのインスタンスは、CPU 使用率がスロットリングされる前に期間限定でバーストする可能性があります。オプションで無制限モードを有効にして、インスタンスがベースラインを超えて必要なだけバーストできるようにします。これにより、アプリケーションは必要に応じて高い CPU パフォーマンスを維持できます。追加料金が適用される場合があります。詳細については、Amazon EC2 ユーザーガイド」のAuto Scaling グループを使用してバーストパフォーマンスインスタンスを無制限で起動する」を参照してください。

プレイスメントグループ名

プレイスメントグループを指定し、クラスターまたはパーティション戦略を使用して、インスタンスが AWS データセンターに物理的にどのように配置されているかに影響を与えることができます。小規模な Auto Scaling グループの場合は、スプレッド戦略を使用することもできます。詳細については、「Amazon EC2 ユーザーガイド」の「プレイスメントグループ」を参照してください。 Amazon EC2

Auto Scaling グループでプレイスメントグループを使用する場合は、いくつかの考慮事項があります。

- 起動テンプレートと Auto Scaling グループの両方でプレイスメントグループが指定されている場合、Auto Scaling グループのプレイスメントグループが優先されます。グループが作成されると、Auto Scaling グループ設定で指定されたプレイスメントグループを変更することはできません。

- では AWS CloudFormation、起動テンプレートでプレイスメントグループを定義する場合は注意してください。Amazon EC2 Auto Scaling は、指定されたプレイスメントグループにインスタンスを起動します。ただし、Auto Scaling グループUpdatePolicyで を使用する場合、CloudFormation はそれらのインスタンスからシグナルを受信しません (ただし、これは将来変更される可能性があります)。

購入オプション

スポットインスタンスのリクエストを選択して、オンデマンド料金でスポットインスタンスをリクエストし、カスタマイズを選択してデフォルトのスポットインスタンス設定を変更できます。Auto Scaling グループでは、終了日なしのワンタイムリクエストを指定する必要があります (デフォルト)。詳細については、「耐障害性に優れた柔軟なアプリケーションのためにスポットインスタンスをリクエストする」を参照してください。この設定は特殊な状況では便利ですが、通常は指定せず、代わりに混合インスタンスグループを作成することをお勧めします。詳細については、「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。

起動テンプレートでスポットインスタンスリクエストを指定した場合、混合インスタンスグループを作成することはできません。混合インスタンスグループでスポットインスタンスをリクエストする起動テンプレートを使用しようとすると、次のように `Incompatible launch template: You cannot use a launch template that is set to request Spot Instances (InstanceMarketOptions) when you configure an Auto Scaling group with a mixed instances policy. Add a different launch template to the group and try again.` というエラーメッセージが表示されます。

Capacity Reservation

キャパシティ予約を使用すると、特定のアベイラビリティーゾーンの Amazon EC2 インスタンスのキャパシティーを任意の期間予約できます。詳細については、「Amazon EC2 ユーザーガイド」の「オンデマンドキャパシティ予約」を参照してください。 Amazon EC2

インスタンスを起動するかどうかを選択できます。

- 任意のオープンキャパシティーの予約 (オープン)
- 特定のキャパシティーの予約 (ID によるターゲット)
- キャパシティ予約のグループ (グループ によるターゲット)

特定のキャパシティーの予約をターゲットにするには、起動テンプレートのインスタンスタイプが予約のインスタンスタイプと一致する必要があります。Auto Scaling グループを作成するときは、キャパシティーの予約と同じアベイラビリティーゾーンを使用します。 AWS リージョン 選

択した に応じて、代わりにキャパシティブロックをターゲットにすることを選択できます。詳細については、「機械学習ワークロードCapacity Blocksに を使用する」を参照してください。

キャパシティ予約のグループをターゲットにするには、「」を参照してくださいオンデマンドキャパシティ予約を使用して特定のアベイラビリティーゾーンのキャパシティを予約する。キャパシティ予約のグループをターゲットにすることで、キャパシティを複数のアベイラビリティーゾーンに分散させて回復性を向上させることができます。

テナンシー

Amazon EC2 には、EC2 インスタンスのテナンシーに 3 つのオプションがあります。

- 共有 (共有) — 複数の が同じ物理ハードウェアを共有 AWS アカウント している場合があります。これは、インスタンスを起動する際のデフォルトのテナンシーオプションです。
- 専用インスタンス (専用) — インスタンスはシングルテナントハードウェアで実行されます。同じ AWS 物理サーバーを共有するお客様は他にありません。詳細については、ハードウェア専有インスタンス の Amazon EC2 ユーザーガイドを参照してください。
- Dedicated Hosts (専有ホスト) – インスタンスは、お客様専用の物理サーバーで実行されます。Dedicated Hosts を使用すると、専用のハードウェア要件を持つ独自のライセンス (BYOL) を EC2 に持ち込み、コンプライアンスのユースケースを満たすことが容易になります。このオプションを選択した場合は、テナンシーホストリソースグループ のホストリソースグループを指定する必要があります。詳細については、「Amazon EC2 ユーザーガイド」の「Dedicated Host」を参照してください。

Dedicated Hosts のサポートは、ホストリソースグループを指定する場合にのみ使用できます。特定のホスト ID をターゲットにしたり、ホストのプレイスメントアフィニティを使用したりすることはできません。

- ホスト ID を指定する起動テンプレートを使用しようとすると、次のエラーメッセージが表示されます。 `Incompatible launch template: Tenancy host ID is not supported for Auto Scaling.`
- ホスト配置のアフィニティを指定する起動テンプレートを使用しようとすると、次のエラーメッセージが表示されます。 `Incompatible launch template: Auto Scaling does not support host placement affinity.`

テナンシーホストリソースグループ

を使用すると AWS License Manager、独自のライセンスを に持ち込み AWS 、一元的に管理できます。ホストリソースグループは、特定の License Manager ライセンス設定にリンクされた Dedicated Hosts のグループです。ホストリソースグループを使用すると、ソフトウェアライセンスのニーズに合った Dedicated Hosts で EC2 インスタンスを簡単に起動できます。Dedicated

Hosts は事前に手動で割り当てる必要はありません。必要に応じて自動的に作成されます。AMI をライセンス設定に関連付ける場合、その AMI は一度に 1 つのホストリソースグループにのみ関連付けることができます。詳細については、「License Manager ユーザーガイド」の「AWS License Managerのホストリソースグループ」を参照してください。

ライセンス設定

この設定では、専有ホストにテナンシーを制限せずに、インスタンスのライセンス設定を指定できます。ライセンス設定は、インスタンスにデプロイされたソフトウェアライセンスを追跡するため、ライセンスの使用状況とコンプライアンスをモニタリングできます。詳細については、「License Manager ユーザーガイド」の「セルフマネージドライセンスの作成」を参照してください。

アクセス可能なメタデータ

インスタンスメタデータサービスの HTTP エンドポイントへのアクセスを有効または無効にするかどうかを選択できます。デフォルトでは、HTTP エンドポイントは有効です。エンドポイントを無効にすると、インスタンスメタデータへのアクセスはオフになります。HTTP エンドポイントが有効になっている場合にのみ、IMDSv2 を要求する条件を指定できます。詳細については、「Amazon EC2 ユーザーガイド」の「インスタンスメタデータオプションの設定」を参照してください。 Amazon EC2

メタデータバージョン

インスタンスメタデータをリクエストするときに、インスタンスメタデータサービスバージョン 2 (IMDSv2) の使用を要求できます。値を指定しない場合、デフォルトで IMDSv1 と IMDSv2 の両方がサポートされます。詳細については、「Amazon EC2 ユーザーガイド」の「インスタンスメタデータオプションの設定」を参照してください。 Amazon EC2

メタデータトークンレスポンスホップ制限

メタデータトークンのネットワークホップの許容数を設定できます。値を指定していない場合、デフォルトで 1 が適用されます。詳細については、「Amazon EC2 ユーザーガイド」の「インスタンスメタデータオプションの設定」を参照してください。 Amazon EC2

ユーザーデータ

シェルスクリプトまたは cloud-init ディレクティブをユーザーデータとして指定することで、起動時にインスタンスをカスタマイズして設定を完了できます。ユーザーデータは、インスタンスの初回起動時に実行されるため、起動時にアプリケーション、依存関係、またはカスタマイズを自動的にインストールできます。詳細については、「Amazon EC2 ユーザーガイド」の「起動時に Linux インスタンスでコマンドを実行するAmazon EC2」を参照してください。

大量のダウンロードや複雑なスクリプトがある場合、インスタンスが使用できるようになるまでにかかる時間が長くなります。その場合、インスタンスが完全にプロビジョニングされるまで InService状態になるまで遅延するようにライフサイクルフックを設定する必要がある場合があります。Auto Scaling グループにライフサイクルフックを追加する方法の詳細については、「」を参照してくださいAmazon EC2 Auto Scaling のライフサイクルフック。

耐障害性に優れた柔軟なアプリケーションのためにスポットインスタンスをリクエストする

起動テンプレートには、終了日や使用期間を指定せずにスポットインスタンスをリクエストできるオプションがあります。Amazon EC2 スポットインスタンスは、EC2 オンデマンドの料金と比較して、大幅割引で提供される予備のキャパシティーです。スポットインスタンス は、アプリケーションを実行する時間に柔軟性がある場合や、アプリケーションを中断できる場合に、費用効率の高い選択肢です。スポットインスタンスをリクエストする起動テンプレート作成の詳細については、「詳細設定を使用して起動テンプレートを作成する」を参照してください。

> **Important**
>
> スポットインスタンスは通常、オンデマンドインスタンスを補完するために使用されます。このシナリオでは、スポットインスタンスの起動に使用されるものと同じ設定を、Auto Scaling グループの設定内で指定できます。Auto Scaling グループの一部として設定を指定する場合、特定の数のオンデマンドインスタンスを起動した後のみにスポットインスタンスを起動するようにリクエストできます。その後、グループのスケーリングに応じてオンデマンドインスタンスとスポットインスタンスの組み合わせを継続して起動するようにリクエストできます。詳細については、「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。

このトピックでは、Auto Scaling グループ自体ではなく起動テンプレートにより設定を指定することで、Auto Scaling グループ内にスポットインスタンスのみを起動する方法について説明します。このトピックの情報は、起動設定を使用してスポットインスタンスをリクエストする Auto Scaling グループにも適用されます。違いは、起動設定には上限価格が必須ですが、起動テンプレートの場合、この設定はオプションとなることです。

起動テンプレートを作成してスポットインスタンスのみを起動する場合は、次の考慮事項に留意してください。

- スポット料金 起動するスポットインスタンスには、現在のスポット料金のみが課金されます。この料金は、需要と供給の長期的な傾向に基づいて時間の経過とともに緩やかに変動します。詳細については、Amazon EC2 ユーザーガイド」の「スポットインスタンス」および「料金と割引」を参照してください。
- 上限価格を設定する。起動テンプレートではオプションとして、スポットインスタンスの時間あたりの上限価格を指定することができます。時間あたりの上限価格が現在のスポット料金を上回っていて、利用可能なキャパシティーがある場合には、Amazon EC2 は速やかにスポットインスタンスに対するリクエストに対応します。スポットインスタンスの料金が、お客様の Auto Scaling グループで実行中のインスタンスに設定された上限価格を上回った場合には、インスタンスが終了されます。

> **Warning**
>
> 低すぎる上限価格が設定されるなどの理由で、スポットインスタンスを取得できない場合には、アプリケーションが実行されないことがあります。利用可能なスポットインスタンスを、可能な限り長期にわたり活用するには、上限価格をオンデマンド料金に近い値に設定します。

- アベイラビリティーゾーン間でのバランシング。複数のアベイラビリティーゾーンを指定した場合、Amazon EC2 Auto Scaling は指定したゾーンの間でスポットリクエストを分散します。上限価格が 1 つのアベイラビリティーゾーンでリクエストを落札するには低すぎる場合、Amazon EC2 Auto Scaling は他のアベイラビリティーゾーンでリクエストが落札されたかどうか確認します。その場合、Amazon EC2 Auto Scaling は失敗したリクエストをキャンセルし、リクエストが落札されたアベイラビリティーゾーンの間でリクエストを再分散します。落札されたリクエストがないアベイラビリティーゾーンの料金が、今後のリクエストに成功するために十分に下がった場合、Amazon EC2 Auto Scaling はすべてのアベイラビリティーゾーンの間で再調整します。
- スポットインスタンスの終了 スポットインスタンスは任意のタイミングで終了できます。スポットインスタンスの可用性や料金に変化があった場合は、Amazon EC2 スポットサービスにより、Auto Scaling グループのスポットインスタンスが終了されることがあります。またスケーリングやヘルスチェックの実行時には、Amazon EC2 Auto Scaling が、オンデマンドインスタンスの終了と同じ方法により、スポットインスタンスを終了することがあります。インスタンスが終了された際には、そのためのストレージは削除されます。
- 必要とされるキャパシティーの維持 終了されるスポットインスタンスがあった場合、Amazon EC2 Auto Scaling は、グループ内で必要とされるキャパシティーを維持するために、代わりのインスタンスの起動を試みます。現在のスポット料金が上限価格未満の場合は、スポットインスタン

スが起動されます。スポットインスタンスのリクエストが正常に処理されなかった場合は、その試行が繰り替えされます。

- 上限価格の変更 上限価格を変更するには、起動テンプレートを新規で作成するか、既存の起動テンプレートを新しい上限価格で更新します。その上で、このテンプレートを Auto Scaling グループに関連付けます。これらのインスタンスに使用する起動テンプレートで指定された上限価格が、現在のスポット料金より高い限り、既存のスポットインスタンスが実行され続けます。上限価格を指定しない場合、オンデマンド料金がデフォルトの上限価格となります。

機械学習ワークロードCapacity Blocksに を使用する

Capacity Blocks は、短期間の機械学習 (ML) ワークロードをサポートするために、需要の高い GPU インスタンスを将来予約するのに役立ちます。

の概要Capacity Blocksと仕組みについては、Amazon EC2 ユーザーガイドの「 for Capacity Blocks ML」を参照してください。

の使用を開始するにはCapacity Blocks、特定のアベイラビリティーゾーンにキャパシティ予約を作成します。 Capacity Blocksは、1 つのアベイラビリティーゾーンにtargetedキャパシティ予約として配信されます。起動テンプレートを作成するときは、キャパシティブロックの予約 ID とインスタンスタイプを指定します。次に、作成した起動テンプレートとキャパシティブロックのアベイラビリティーゾーンを使用するように Auto Scaling グループを更新します。キャパシティブロック予約が開始されたら、スケジュールされたスケーリングを使用して、キャパシティブロック予約と同じ数のインスタンスを起動します。

> **Important**
>
> Capacity Blocks は、特定の Amazon EC2 インスタンスタイプ および でのみ使用できます AWS リージョン。詳細については、Amazon EC2 ユーザーガイド」の「前提条件」を参照してください。

内容

- 操作のガイドライン
- 起動テンプレートでキャパシティブロックを指定する
- 制限事項
- 関連リソース

操作のガイドライン

以下は、Auto Scaling グループでキャパシティブロックを使用するに従うべき操作の基本的なガイドラインです。

- キャパシティブロックの予約終了時刻の 30 分以上前に、Auto Scaling グループをゼロにスケールインします。Amazon EC2 は、キャパシティブロックの終了時刻の 30 分前に、実行中のインスタンスを終了します。
- スケジュールされたスケーリングを使用して、適切な予約時間にスケールアウト (インスタンスの追加) およびスケールイン (インスタンスの削除) することをお勧めします。詳細については、「Amazon EC2 Auto Scaling のスケジュールされたスケーリング」を参照してください。
- 必要に応じてライフサイクルフックを追加し、スケールインの際にインスタンス内でアプリケーションを正常にシャットダウンします。キャパシティブロックの予約終了時刻の 30 分前に Amazon EC2 がインスタンスの強制終了を開始するまでに、ライフサイクルアクションが完了できるよう十分な時間を確保してください。詳細については、「Amazon EC2 Auto Scaling のライフサイクルフック」を参照してください。
- Auto Scaling グループが、予約期間全体を通して正しいバージョンの起動テンプレートを指定していることを確認してください。`$Default` または `$Latest` バージョンではなく、特定のバージョンの起動テンプレートを指定することをお勧めします。

> **Note**
>
> 予約が終了するまでキャパシティブロックインスタンスを実行したままにし、Amazon EC2 がそれを再利用した場合、Auto Scaling グループのスケーリングアクティビティは、キャパシティブロックの最後に意図的に再利用された場合でも`taken out of service in response to an EC2 health check that indicated it had been terminated or stopped.`、「」と表示されます。同様に、Amazon EC2 Auto Scaling は、ヘルスチェックに失敗したインスタンスの場合と同じ方法でインスタンスの置き換えを試みます。詳細については、「Auto Scaling グループ内のインスタンスのヘルスチェック」を参照してください。

起動テンプレートでキャパシティブロックを指定する

Auto Scaling グループの特定のキャパシティブロックをターゲットとする起動テンプレートを作成するには、次のいずれかの方法を使用します。

Console

起動テンプレートでキャパシティブロックを指定するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 上部のナビゲーションバーで、キャパシティブロックを作成した AWS リージョン を選択します。
3. ナビゲーションペインで、[インスタンス] の [テンプレートの起動] を選択します。
4. 起動テンプレートの作成 を選択し、起動テンプレートを作成します。必要に応じて、Amazon マシンイメージ (AMI) の ID、インスタンスタイプ、およびその他の起動テンプレート設定を含めます。
5. [高度な詳細] セクションを展開して、詳細設定を表示します。
6. 購入オプション で、キャパシティブロック を選択します。
7. [キャパシティ予約] で [ID 別のターゲット] を選択し、[キャパシティ予約 - ID ごとのターゲット] で、既存のキャパシティブロックのキャパシティ予約 ID を選択します。
8. 完了したら、起動テンプレートの作成 を選択します。

起動テンプレートを使用して Auto Scaling グループを作成する方法については、「」を参照してください起動テンプレートを使用して Auto Scaling グループを作成する。

AWS CLI

起動テンプレートでキャパシティブロックを指定するには (AWS CLI)

次の create-launch-template コマンドを使用して、既存のキャパシティブロックの予約 ID を指定するための起動テンプレートを作成します。各##############を独自の情報に置き換えます。

```
aws ec2 create-launch-template --launch-template-name my-template-for-capacity-block
  \
  --version-description AutoScalingVersion1 --region us-east-2 \
  --launch-template-data file://config.json
```

> **Tip**
>
> このコマンドがエラーをスローする場合は、 を AWS CLI ローカルで最新バージョンに更新していることを確認してください。

config.json の内容。

```
{
    "ImageId": "ami-04d5cc9b88example",
    "InstanceType": "p4d.24xlarge",
    "SecurityGroupIds": [
        "sg-903004f88example"
    ],
    "KeyName": "MyKeyPair",
    "InstanceMarketOptions": {
        "MarketType": "capacity-block"
    },
    "CapacityReservationSpecification": {
        "CapacityReservationTarget": {
            "CapacityReservationId": "cr-02168da1478b509e0"
        }
    }
}
```

以下は出力例です。

```
{
    "LaunchTemplate": {
        "LaunchTemplateId": "lt-068f72b724example",
        "LaunchTemplateName": "my-template-for-capacity-block",
        "CreateTime": "2023-10-27T15:12:44.000Z",
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "DefaultVersionNumber": 1,
        "LatestVersionNumber": 1
    }
}
```

次の describe-launch-template-versions コマンドを使用して、起動テンプレートに関連付けられたキャパシティブロックの予約 ID を確認できます。

```
aws ec2 describe-launch-template-versions --launch-template-names my-template-for-capacity-block \
  --region us-east-2
```

以下は、キャパシティブロックの予約を指定する起動テンプレートの出力例です。

```
{
```

```
    "LaunchTemplateVersions": [
        {
            "LaunchTemplateId": "lt-068f72b724example",
            "LaunchTemplateName": "my-template-for-capacity-block",
            "VersionNumber": 1,
            "CreateTime": "2023-10-27T15:12:44.000Z",
            "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
            "DefaultVersion": true,
            "LaunchTemplateData": {
                "ImageId": "ami-04d5cc9b88example",
                "InstanceType": "p5.48xlarge",
                "SecurityGroupIds": [
                    "sg-903004f88example"
                ],
                "KeyName": "MyKeyPair",
                "InstanceMarketOptions": {
                    "MarketType": "capacity-block"
                },
                "CapacityReservationSpecification": {
                    "CapacityReservationTarget": {
                        "CapacityReservationId": "cr-02168da1478b509e0"
                    }
                }
            }
        }
    ]
}
```

制限事項

- のサポートCapacity Blocksは、Auto Scaling グループに互換性のある設定がある場合にのみ使用できます。混合インスタンスグループおよびウォームプールはサポートされていません。
- 一度にターゲットにできるキャパシティブロックは 1 つだけです。

関連リソース

- P5 インスタンスを使用するための前提条件と推奨事項については、Amazon EC2 ユーザーガイド」のP5 インスタンスの開始方法」を参照してください。

- Amazon EKS は、Amazon EKS クラスターでの短期間Capacity Blocksの機械学習 (ML) ワークロードをサポートするために の使用をサポートしています。詳細については、「Amazon EKS Capacity Blocks ユーザーガイド」の「 for ML」を参照してください。
- は、サポートされているインスタンスタイプとリージョンCapacity Blocksで使用できます。ただし、オンデマンドキャパシティ予約では、他のインスタンスタイプやリージョンのキャパシティを柔軟に予約できます。オンデマンドキャパシティ予約オプションの使用方法を示すチュートリアルについては、「」を参照してくださいオンデマンドキャパシティ予約を使用して特定のアベイラビリティーゾーンのキャパシティを予約する。

Auto Scaling グループを起動テンプレートに移行する

2023 年より、2022 年 12 月 31 日以降にリリースされた新しい Amazon EC2 インスタンスタイプで `CreateLaunchConfiguration` を呼び出すことはできなくなりました。詳細については、「起動設定」を参照してください。

Auto Scaling グループを起動設定から起動テンプレートに移行するには、次のステップを参照してください。

> **Important**
>
> 続行する前に、起動テンプレートを操作するために必要な許可があることを確認してください。詳細については、「起動テンプレートを操作するアクセス許可」を参照してください。

ステップ 1: 起動設定を使用する Auto Scaling グループを検索する

まだ起動設定を使用している Auto Scaling グループがあるかどうかを確認するには、 AWS CLIを使用して次の describe-auto-scaling-groups コマンドを実行します。*REGION* を に置き換えます AWS リージョン。

```
aws autoscaling describe-auto-scaling-groups --region REGION \
  --query 'AutoScalingGroups[?LaunchConfigurationName!=`null`]'
```

以下は出力例です。

```
[
    {
 "AutoScalingGroupName": "group-1",
```

```
"AutoScalingGroupARN": "arn",
"LaunchConfigurationName": "my-launch-config",
"MinSize": 1,
"MaxSize": 5,
"DesiredCapacity": 2,
"DefaultCooldown": 300,
"AvailabilityZones": [
          "us-west-2a",
          "us-west-2b",
          "us-west-2c"
      ],
"LoadBalancerNames": [],
"TargetGroupARNs": [],
"HealthCheckType": "EC2",
"HealthCheckGracePeriod": 300,
"Instances": [
          {
              "ProtectedFromScaleIn": false,
              "AvailabilityZone": "us-west-2a",
              "LaunchConfigurationName": "my-launch-config",
              "InstanceId": "i-05b4f7d5be44822a6",
              "InstanceType": "t3.micro",
              "HealthStatus": "Healthy",
              "LifecycleState": "InService"
          },
          {
              "ProtectedFromScaleIn": false,
              "AvailabilityZone": "us-west-2b",
              "LaunchConfigurationName": "my-launch-config",
              "InstanceId": "i-0c20ac468fa3049e8",
              "InstanceType": "t3.micro",
              "HealthStatus": "Healthy",
              "LifecycleState": "InService"
          }
],
"CreatedTime": "2023-03-09T22:15:11.611Z",
"SuspendedProcesses": [],
"VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782",
"EnabledMetrics": [],
"Tags": [
          {
 "ResourceId": "group-1",
 "ResourceType": "auto-scaling-group",
 "Key": "environment",
```

```
  "Value": "production",
  "PropagateAtLaunch": true
            }
        ],
 "TerminationPolicies": [
     "Default"
 ],
 "NewInstancesProtectedFromScaleIn": false,
 "ServiceLinkedRoleARN": "arn",
       "TrafficSources": []
    },

    ... additional groups ...

]
```

あるいは、Auto Scaling グループ名、それぞれの起動設定およびタグの名前のみを出力に表示するには、次のコマンドを実行します。

```
aws autoscaling describe-auto-scaling-groups --region REGION \
  --query 'AutoScalingGroups[?LaunchConfigurationName!=`null`].{AutoScalingGroupName:
 AutoScalingGroupName, LaunchConfigurationName: LaunchConfigurationName, Tags: Tags}'
```

出力例を次に示します。

```
[
    {
        "AutoScalingGroupName": "group-1",
        "LaunchConfigurationName": "my-launch-config",
        "Tags": [
            {
                "ResourceId": "group-1",
                "ResourceType": "auto-scaling-group",
                "Key": "environment",
                "Value": "production",
                "PropagateAtLaunch": true
            }
        ]
    },

    ... additional groups ...
```

```
]
```

フィルタリングの詳細については、「 AWS Command Line Interface ユーザーガイド」のAWS CLI「出力のフィルタリング」を参照してください。

ステップ 2: 起動設定を起動テンプレートにコピーする

次のステップを実行して、起動設定を起動テンプレートにコピーできます。その後、それを Auto Scaling グループに追加できます。

複数の起動設定をコピーすると、同じ名前の起動テンプレートが作成されます。コピープロセス中に起動テンプレートに付けられた名前を変更するには、起動設定を 1 つずつコピーする必要があります。

> **Note**
>
> コピー機能は、コンソールでのみ使用できます。

起動テンプレートへ起動設定をコピーするには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左のナビゲーションペインの [Auto Scaling] で、[Auto Scaling グループ] を選択します。
3. ページの上部付近にある [起動設定] を選択します。確認を求めるプロンプトが表示されたら、[起動設定を表示] を選択して、[起動設定] ページを表示することを確認します。
4. コピーする起動設定を選択し、[Copy to launch template (起動テンプレートにコピー)、Copy selected (選択した範囲をコピー)] を選択します。これにより、新しい起動テンプレートが、選択した起動設定と同じ名前およびオプションでセットアップされます。
5. [New launch template name (新しい起動テンプレート名)] では、起動設定の名前 (デフォルト) を使用する、または、新しい名前を入力します。起動テンプレート名は一意である必要があります。
6. (オプション) [新しいテンプレートを使用して Auto Scaling グループを作成] を選択します。

   このステップをスキップして、起動設定のコピーを完了することができます。新しい Auto Scaling グループを作成する必要はありません。
7. [コピー] を選択します。

すべての起動設定を起動テンプレートにコピーするには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインの [Auto Scaling] で、[Launch Configurations (起動設定)] を選択します。
3. Copy to launch template (起動テンプレートにコピー)、Copy all (すべてコピー)を選択します。これにより、現在のリージョンの各起動設定が、同じ名前およびオプションで新しい起動テンプレートにコピーされます。
4. [コピー] を選択します。

ステップ 3: 起動テンプレートを使用するように Auto Scaling グループを更新する

起動テンプレートを作成すると、それを Auto Scaling グループに追加する準備が整います。

起動テンプレートを使用するように Auto Scaling グループを更新するには (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの隣にあるチェックボックスを選択します。

   ページ下部に分割ウィンドウが開き、選択したグループの情報が表示されます。
3. [詳細] タブで、[起動設定]、[編集] の順に選択します。
4. 起動テンプレートに切り替えるを選択します。
5. [Launch Template (起動テンプレート)] では、起動テンプレートを選択します。
6. [Version (バージョン)] では、必要に応じて起動テンプレートのバージョンを選択します。起動テンプレートのバージョンを作成したら、スケールアウト時に Auto Scaling グループで起動テンプレートのデフォルトバージョンを使用するか最新バージョンを使用するかを選択できます。
7. [更新] を選択します。

起動テンプレートを使用するように Auto Scaling グループを更新するには (AWS CLI)

次の update-auto-scaling-group コマンドは、指定された Auto Scaling グループを更新して、指定された起動テンプレートの最初のバージョンを使用します。

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
```

```
  --launch-template LaunchTemplateName=my-template-for-auto-scaling,Version='1'
```

CLI コマンドを使用して Auto Scaling グループを更新して起動テンプレートを使用する他の例については、「起動テンプレートを使用するように Auto Scaling グループを更新する」を参照してください。

ステップ 4: インスタンスを置き換える

起動設定を起動テンプレートに置き換えると、新しいインスタンスは、その新しい起動テンプレートを使用するようになります。既存のインスタンスは影響を受けません。

既存のインスタンスを更新するには、一度にいくつかのインスタンスを手動で置き換えるのではなく、インスタンスの更新を開始して Auto Scaling グループ内のインスタンスを置き換えることができます。詳細については、「インスタンスの更新を使用して Auto Scaling グループのインスタンスを更新する」を参照してください。グループが大きい場合、インスタンスの更新は特に便利です。

あるいは、自動スケーリングを許可して、グループの終了ポリシーに基づき既存のインスタンスを新しいインスタンスに徐々に置き換えたり、インスタンスを終了したりすることもできます。手動での終了により、グループが必要とするキャパシティーを維持するため、強制的にAuto Scaling グループが新しいインスタンスを起動します。詳細については、「Amazon EC2 ユーザーガイド」の「インスタンスの終了」を参照してください。 Amazon EC2

追加情報

詳細については、「 コンピューティングブログ」のAmazon EC2 Auto Scaling は、新しい EC2 機能のサポートを 起動設定に追加しなくなります」を参照してください。 AWS

起動設定から起動テンプレートに AWS CloudFormation スタックを移行する方法を説明するトピックについては、「」を参照してくださいスタックを起動テンプレートに移行する AWS CloudFormation。

スタックを起動テンプレートに移行する AWS CloudFormation

既存の AWS CloudFormation スタックテンプレートを起動設定から起動テンプレートに移行できます。これを行うには、起動テンプレートを既存のスタックテンプレートに直接追加した後、そのスタックテンプレート内で、起動テンプレートに Auto Scaling グループを関連付けます。その後、変更したテンプレートを使用してスタックを更新します。

起動テンプレートに移行する場合、このトピックでは、 CloudFormation スタックテンプレートの起動設定を起動テンプレートとして書き換える手順を提供することで、時間を節約できます。起動設定

を起動テンプレートに移行する方法の詳細については、「Auto Scaling グループを起動テンプレートに移行する」を参照してください。

トピック

- 起動設定を使用している Auto Scaling グループを検索する
- 起動テンプレートを使用するようにスタックを更新する
- スタックリソースの更新時の動作
- 移行を追跡する
- 起動設定のマッピングリファレンス

起動設定を使用している Auto Scaling グループを検索する

起動設定を使用している Auto Scaling グループを見つけるには

- 次の describe-auto-scaling-groups コマンドを使用して、指定したリージョン内で起動設定の使用を続けている Auto Scaling グループの名前を一覧表示します。`--filters` オプションを含めて、 CloudFormation スタックに関連付けられたグループに結果を絞り込みます (`aws:cloudformation:stack-name`タグキーでフィルタリングします)。

```
aws autoscaling describe-auto-scaling-groups --region REGION \
  --filters Name=tag-key,Values=aws:cloudformation:stack-name \
  --query 'AutoScalingGroups[?LaunchConfigurationName!
=`null`].AutoScalingGroupName'
```

出力例を次に示します。

```
[
    "{stack-name}-group-1",
    "{stack-name}-group-2",
    "{stack-name}-group-3"
]
```

Auto Scaling グループを検索して出力を移行し、フィルタリングするためのその他の便利な AWS CLI コマンドは、 にありますAuto Scaling グループを起動テンプレートに移行する。

> **Important**
>
> スタックリソースの名前AWSEBに が含まれている場合、これは を通じて作成されたことを意味します AWS Elastic Beanstalk。この場合、Beanstalk 環境を更新して、起動設定を削除して起動テンプレートに置き換えるように Elastic Beanstalk に指示する必要があります。

起動テンプレートを使用するようにスタックを更新する

このセクションのステップに従い、 その手順を実行します。

- 起動設定のプロパティを、起動テンプレートの同等のプロパティに対し記述し直します。
- 新しい起動テンプレートを Auto Scaling グループと関連付けます。
- これらの更新内容をデプロイします。

スタックテンプレートを変更してスタックを更新するには

1. 「AWS CloudFormation ユーザーガイド」の「スタックテンプレートの変更」で説明されている、一般的なスタックテンプレートの変更と同じ手順に従います。
2. 起動設定を起動テンプレートとして記述し直します。次の例を参照してください。

   例: シンプルな起動設定

```
---
Resources:
  myLaunchConfig:
    Type: AWS::AutoScaling::LaunchConfiguration
    Properties:
      ImageId: ami-02354e95b3example
      InstanceType: t3.micro
      SecurityGroups:
        - !Ref EC2SecurityGroup
      KeyName: MyKeyPair
      BlockDeviceMappings:
        - DeviceName: /dev/xvda
          Ebs:
            VolumeSize: 150
            DeleteOnTermination: true
      UserData:
        Fn::Base64: !Sub |
```

```
          #!/bin/bash -xe
          yum install -y aws-cfn-bootstrap
          /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --resource myASG
 --region ${AWS::Region}
```

例: 同等の起動テンプレート

```
---
Resources:
  myLaunchTemplate:
    Type: AWS::EC2::LaunchTemplate
    Properties:
      LaunchTemplateName: !Sub ${AWS::StackName}-launch-template
      LaunchTemplateData:
        ImageId: ami-02354e95b3example
        InstanceType: t3.micro
        SecurityGroupIds:
          - Ref! EC2SecurityGroup
        KeyName: MyKeyPair
        BlockDeviceMappings:
          - DeviceName: /dev/xvda
            Ebs:
              VolumeSize: 150
              DeleteOnTermination: true
        UserData:
          Fn::Base64: !Sub |
            #!/bin/bash -x
            yum install -y aws-cfn-bootstrap
            /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --resource
 myASG --region ${AWS::Region}
```

Amazon EC2 がサポートするすべてのプロパティのリファレンス情報については、「 ユーザーガイドAWS::EC2::LaunchTemplate」のAWS CloudFormation 「」を参照してください。

起動テンプレートには、!Sub ${AWS::StackName}-launch-template 値が指定された LaunchTemplateName プロパティが含まれていることに注意してください。これは、起動テンプレート名にスタック名を含めたい場合に必須です。

3. **IamInstanceProfile** プロパティが起動設定に存在する場合は、それを構造に変換し、インスタンスプロファイルの名前または ARN を指定する必要があります。例については、AWS::EC2::LaunchTemplateを参照してください。

4. 起動設定に **AssociatePublicIpAddress**、**InstanceMonitoring**、または **PlacementTenancy** プロパティがある場合は、これらを構造体に変換する必要があります。例については、「」を参照してくださいAWS::EC2::LaunchTemplate。

   ただし、Auto Scaling グループに使用したサブネットの MapPublicIpOnLaunch プロパティの値が、起動設定の AssociatePublicIpAddress プロパティの値と一致する場合は例外です。この場合は、AssociatePublicIpAddress プロパティを無視できます。AssociatePublicIpAddress プロパティは、MapPublicIpOnLaunch プロパティを上書きして、起動時にインスタンスがパブリック IPv4 アドレスを受信するかどうかを変更する場合にのみ使用されます。

5. **SecurityGroups** プロパティから、セキュリティグループを起動テンプレートの 2 つの場所のどちらかにコピーすることができます。通常は、SecurityGroupIds プロパティにセキュリティグループをコピーします。ただし、起動テンプレート内に AssociatePublicIpAddress プロパティを指定する NetworkInterfaces 構造体を作成する場合は、代わりにネットワークインターフェイスの Groups プロパティに対し、セキュリティグループをコピーする必要があります。

6. **NoDevice** に true が設定された BlockDeviceMapping 構造体が起動設定の中にある場合は、起動テンプレート内の NoDevice に空の文字列を指定して、Amazon EC2 にデバイスを省略させる必要があります。

7. 起動設定内に **SpotPrice** プロパティが含まれている場合、起動テンプレートではそれを省略することをお勧めします。スポットインスタンスは現在のスポット料金で起動します。この価格は、オンデマンド料金を超えることはありません。

   スポットインスタンスをリクエストする際には、下記のように相互に排他的な 2 つのオプションが存在します。

   - 1 つ目は、起動テンプレート内で InstanceMarketOptions 構造体を使用することです (非推奨)。詳細については、「 ユーザーガイドAWS::EC2::LaunchTemplate InstanceMarketOptions」のAWS CloudFormation 「」を参照してください。
   - 2 つ目は、Auto Scaling グループに MixedInstancesPolicy 構造体を追加することです。これを行うと、リクエストの方法に、より多くの選択肢が得られるようになります。起動テンプレートのスポットインスタンスリクエストでは、Auto Scaling グループごとに複数のインスタンスタイプを選択することはできません。ただし、インスタンスポリシーを組み合わせることで、Auto Scaling グループごとに複数のインスタンスタイプを選択できます。インスタンスタイプに複数の選択肢があると、スポットインスタンスリクエストに対するメリットが得られます。詳細については、「 ユーザーガイド」のAWS::AutoScaling::AutoScaling「グループ MixedInstancesPolicyAWS CloudFormation 」を参照してください。

8. AWS::AutoScaling::AutoScalingグループリソースから **LaunchConfigurationName**プロパティを削除します。その代わりに、起動テンプレートを追加します。

   次の例では、Ref 組み込み関数は論理 ID を持つAWS::EC2::LaunchTemplateリソースの ID を取得しますmyLaunchTemplate。GetAtt 関数は、 Versionプロパティの起動テンプレートの最新バージョン番号 (など1) を取得します。

   例: インスタンスポリシーの組み合わせなし

   ```
   ---
   Resources:
     myASG:
       Type: AWS::AutoScaling::AutoScalingGroup
       Properties:
         LaunchTemplate:
           LaunchTemplateId: !Ref myLaunchTemplate
           Version: !GetAtt myLaunchTemplate.LatestVersionNumber
         ...
   ```

   例: インスタンスポリシーの組み合わせあり

   ```
   ---
   Resources:
     myASG:
       Type: AWS::AutoScaling::AutoScalingGroup
       Properties:
         MixedInstancesPolicy:
           LaunchTemplate:
             LaunchTemplateSpecification:
               LaunchTemplateId: !Ref myLaunchTemplate
               Version: !GetAtt myLaunchTemplate.LatestVersionNumber
         ...
   ```

   Amazon EC2 Auto Scaling がサポートするすべてのプロパティのリファレンス情報については、AWS CloudFormation 「 ユーザーガイド」のAWS::AutoScaling::AutoScaling「グループ」を参照してください。

9. これらの更新をデプロイする準備ができたら、 CloudFormation手順に従って、変更されたスタックテンプレートでスタックを更新します。詳細については、「AWS CloudFormation ユーザーガイド」の「スタックテンプレートの変更」を参照してください。

スタックリソースの更新時の動作

CloudFormation は、指定した更新されたテンプレートと、スタックテンプレートの以前のバージョンで説明したリソース設定の変更を比較して、スタックリソースを更新します。変更されていないリソース設定は、更新プロセス中も影響を受けません。

CloudFormation は Auto Scaling グループの UpdatePolicy 属性をサポートします。更新中に `UpdatePolicy`が に設定されている場合`AutoScalingRollingUpdate`、 はこの手順のステップを実行した後に`InService`インスタンスを CloudFormation 置き換えます。`UpdatePolicy` が に設定されている場合`AutoScalingReplacingUpdate`、 は Auto Scaling グループとそのウォームプール (存在する場合) を CloudFormation 置き換えます。

Auto Scaling グループの`UpdatePolicy`属性を指定しなかった場合、起動テンプレートが正しいかどうかはチェックされますが、Auto Scaling グループのインスタンス全体に変更がデプロイ CloudFormation されることはありません。すべての新しいインスタンスは、起動テンプレートを使用するようになります。ただし、既存のインスタンスは、起動した当初の起動設定のままで実行を継続します (起動設定が存在しない場合は除く)。この例外となるのは、組み合わせのインスタンスポリシーを追加するなどして、購入オプションを変更した場合です。この場合、新しい購入オプションに適合するために、Auto Scaling グループにより、徐々に既存のインスタンスが新しいインスタンスに置き換えられます。

移行を追跡する

移行を追跡するには

1. AWS CloudFormation コンソールで、更新したスタックを選択し、[Events] (イベント) タブを選択してスタックイベントを表示します。
2. イベントリストを最新のイベントで更新するには、 CloudFormation コンソールで更新ボタンを選択します。
3. スタックの更新中は、リソースが更新されるたびに複数のイベントが表示されます。起動テンプレートを作成しようとしたときに、[ステータスの理由] 列に例外が表示され問題が発生したことが示された場合は、「Amazon EC2 Auto Scaling をトラブルシューティングする: 起動テンプレート」で可能性のある原因を参照してください。
4. (オプション) `UpdatePolicy` 属性の使用状況に応じて、Amazon EC2 コンソールの Auto Scaling グループのページから Auto Scaling グループの進行状況をモニタリングできます。Auto Scaling グループを選択します。[Activity] (アクティビティ) タブで、[Activity history] (アクティビティ履歴) の下の [Status] (ステータス) 列に、Auto Scaling グループがインスタンスを正常に起動あるいは終了したか、または、依然としてスケーリングが進行中なのかが表示されます。

5. スタックの更新が完了すると、 はUPDATE_COMPLETEスタックイベント CloudFormation を発行します。詳細については、「AWS CloudFormation ユーザーガイド」の「スタック更新の進行状況の監視」を参照してください。
6. スタックの更新が完了したら、Amazon EC2 コンソールの [起動テンプレート] ページと [起動設定] ページを開きます。新しい起動テンプレートが作成され、起動設定が削除されていることが確認できます。

起動設定のマッピングリファレンス

参考までに、次の表にAWS::AutoScaling::LaunchConfiguration、リソース内のすべての最上位プロパティと、AWS::EC2::LaunchTemplateリソース内の対応するプロパティを示します。

起動設定のソースプロパティ	起動テンプレートのターゲットプロパティ
AssociatePublicIpAddress	NetworkInterfaces.AssociatePublicIpAddress
BlockDeviceMappings	BlockDeviceMappings
ClassicLinkVPCId	利用不可
ClassicLinkVPCSecurityGroups	利用不可
EbsOptimized	EbsOptimized
IamInstanceProfile	IamInstanceProfile.Arn または IamInstanceProfile.Name を入力します。両方を入力することはできません。
ImageId	ImageId
InstanceId	InstanceId
InstanceMonitoring	Monitoring.Enabled
InstanceType	InstanceType
KernelId	KernelId

起動設定のソースプロパティ	起動テンプレートのターゲットプロパティ
KeyName	KeyName
LaunchConfigurationName	LaunchTemplateName
MetadataOptions	MetadataOptions
PlacementTenancy	Placement.Tenancy
RamDiskId	RamDiskId
SecurityGroups	SecurityGroupIds または NetworkInterfaces.Groups を入力します。両方を入力することはできません。
SpotPrice	InstanceMarketOptions.SpotOptions.MaxPrice
UserData	UserData

1 EC2-Classic は使用できなくなるため、 ClassicLinkVPCIdおよび ClassicLinkVPCSecurityGroupsプロパティは起動テンプレートでは使用できません。 EC2-Classic

を使用した起動テンプレートの作成と管理の例 AWS CLI

起動テンプレートは、、 AWS Command Line Interface （ AWS CLI ） AWS Management Console、または SDKs。このセクションでは、 から Amazon EC2 Auto Scaling の起動テンプレートを作成および管理する例を示します AWS CLI。

内容

- 使用例
- 基本的な起動テンプレートを作成する
- 起動時にインスタンスにタグ付けするタグを指定する
- インスタンスに渡す IAM ロールを指定する
- パブリック IP アドレスを割り当てる

- 起動時にインスタンスを設定するユーザーデータスクリプトを指定する
- ブロックデバイスマッピングを指定する
- 外部ベンダーからソフトウェアライセンスを取得するための Dedicated Hosts を指定する
- 既存のネットワークインターフェイスを指定する
- 複数のネットワークインターフェイスを作成する
- 起動テンプレートを管理する
- 起動テンプレートを使用するように Auto Scaling グループを更新する

使用例

```
{
    "LaunchTemplateName": "my-template-for-auto-scaling",
    "VersionDescription": "test description",
    "LaunchTemplateData": {
        "ImageId": "ami-04d5cc9b88example",
        "InstanceType": "t2.micro",
        "SecurityGroupIds": [
            "sg-903004f88example"
        ],
        "KeyName": "MyKeyPair",
        "Monitoring": {
            "Enabled": true
        },
        "Placement": {
            "Tenancy": "dedicated"
        },
        "CreditSpecification": {
            "CpuCredits": "unlimited"
        },
        "MetadataOptions": {
            "HttpTokens": "required",
            "HttpPutResponseHopLimit": 1,
            "HttpEndpoint": "enabled"
        }
    }
}
```

基本的な起動テンプレートを作成する

基本的な起動テンプレートを作成するには、create-launch-templateコマンドを次のように使用して変更を加えます：

- インスタンスを起動する AMI の ID をami-04d5cc9b88exampleに置き換えます。
- 指定した AMI と互換性のあるインスタンスタイプをt2.microに置き換えます。

この例では、*my-template-for-Auto Scaling*という名前の起動テンプレートを作成します。この起動テンプレートによって作成されたインスタンスがデフォルト VPC で起動されると、デフォルトでパブリック IP アドレスが割り当てられます。インスタンスがデフォルト以外の VPC で起動される場合は、デフォルトでパブリック IP アドレスは割り当てられません。

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
  --launch-template-data
 '{"ImageId":"ami-04d5cc9b88example","InstanceType":"t2.micro"}'
```

JSON 形式のパラメータで引用する方法については、AWS Command Line Interface ユーザーガイドの「AWS CLIの文字列で引用符を使用する」を参照してください。

または、設定ファイルで JSON 形式のパラメータを指定することもできます。

次の例では、起動テンプレートパラメータ値の設定ファイルを参照して、基本的な起動テンプレートを作成します。

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
  --launch-template-data file://config.json
```

config.json の内容：

```
{
    "ImageId":"ami-04d5cc9b88example",
    "InstanceType":"t2.micro"
}
```

起動時にインスタンスにタグ付けするタグを指定する

次の例では、起動時にタグ (例: purpose=webserver) をインスタンスに追加します。

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
  --launch-template-data '{"TagSpecifications":[{"ResourceType":"instance","Tags":
[{"Key":"purpose","Value":"webserver"}]}],"ImageId":"ami-04d5cc9b88example","InstanceType":"t2.
```

> **Note**
>
> 起動テンプレートでインスタンスタグを指定して、Auto Scaling グループのタグをそのインスタンスに伝播することを選択した場合、すべてのタグがマージされます。起動テンプレートのタグと Auto Scaling グループのタグに同じタグキーが指定されている場合、グループのタグ値が優先されます。

インスタンスに渡す IAM ロールを指定する

次の例では、IAM ロールに関連付けられているインスタンスプロファイルの名前を指定し、起動時にインスタンスに受け渡します。詳細については、「Amazon EC2 インスタンスで実行中のアプリケーション用の IAM ロール」を参照してください。

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
--launch-template-data '{"IamInstanceProfile":{"Name":"my-instance-
profile"},"ImageId":"ami-04d5cc9b88example","InstanceType":"t2.micro"}'
```

パブリック IP アドレスを割り当てる

次の create-launch-template の例では、デフォルト以外の VPC で起動したインスタンスに、パブリックアドレスを割り当てる起動テンプレートを設定します。

> **Note**
>
> ネットワークインターフェイスを指定する場合、Auto Scaling グループがインスタンスを起動する VPC のセキュリティグループに対応するGroupsの値を指定します。Auto Scaling グループのプロパティとして VPC サブネットを指定します。

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
```

```
  --launch-template-data '{"NetworkInterfaces":
[{"DeviceIndex":0,"AssociatePublicIpAddress":true,"Groups":
["sg-903004f88example"],"DeleteOnTermination":true}],"ImageId":"ami-04d5cc9b88example","Instanc
```

起動時にインスタンスを設定するユーザーデータスクリプトを指定する

次の例では、起動時にインスタンスを設定する base64-encoded 文字列としてユーザーデータスクリプトを指定します。create-launch-templateコマンドには、base64-encoded ユーザーデータが必要です。

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
--launch-template-data
 '{"UserData":"IyEvYmluL2Jhc...","ImageId":"ami-04d5cc9b88example","InstanceType":"t2.micro"}'
```

ブロックデバイスマッピングを指定する

次の create-launch-template の例では、ブロックデバイスマッピングを使用して起動テンプレートを作成します。22 GB の EBS ボリュームは /dev/xvdcz にマップされます。/dev/xvdcz ボリュームは、汎用 SSD (gp2) ボリュームタイプを使用し、アタッチされているインスタンスを終了するときに削除されます。

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
  --launch-template-data '{"BlockDeviceMappings":[{"DeviceName":"/dev/xvdcz","Ebs":
{"VolumeSize":22,"VolumeType":"gp2","DeleteOnTermination":true}}],"ImageId":"ami-04d5cc9b88exam
```

外部ベンダーからソフトウェアライセンスを取得するための Dedicated Hosts を指定する

ホストテナンシーを指定すると、ホストリソースグループと License Manager ライセンス構成を指定して、外部ベンダーから適格なソフトウェアライセンスを取得できます。その後、EC2 インスタンスでライセンスを使用するには、次の create-launch-template コマンドを実行します。

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
  --launch-template-data '{"Placement":
{"Tenancy":"host","HostResourceGroupArn":"arn"},"LicenseSpecifications":
[{"LicenseConfigurationArn":"arn"}],"ImageId":"ami-04d5cc9b88example","InstanceType":"t2.micro"
```

既存のネットワークインターフェイスを指定する

以下の create-launch-template の例では、プライマリネットワークインターフェイスを既存のネットワークインターフェイスを使用するように設定します。

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
  --launch-template-data '{"NetworkInterfaces":
[{"DeviceIndex":0,"NetworkInterfaceId":"eni-
b9a5ac93","DeleteOnTermination":false}],"ImageId":"ami-04d5cc9b88example","InstanceType":"t2.mi
```

複数のネットワークインターフェイスを作成する

以下の create-launch-template の例では、セカンダリネットワークインターフェイスを追加します。プライマリネットワークインターフェイスのデバイスインデックスは 0 で、セカンダリネットワークインターフェイスのデバイスインデックスは 1 です。

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
  --launch-template-data '{"NetworkInterfaces":[{"DeviceIndex":0,"Groups":
["sg-903004f88example"],"DeleteOnTermination":true},{"DeviceIndex":1,"Groups":
["sg-903004f88example"],"DeleteOnTermination":true}],"ImageId":"ami-04d5cc9b88example","Instanc
```

複数のネットワークカードおよび Elastic Fabric Adapters (EFA) をサポートするインスタンスタイプを使用する場合は、セカンダリネットワークカードにセカンダリインターフェイスを追加し、次のcreate-launch-templateコマンドを使用して EFA を有効にします。詳細については、「Amazon EC2 ユーザーガイド」の「起動テンプレートへの EFA の追加」を参照してください。 Amazon EC2

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
  --launch-template-data '{"NetworkInterfaces":
[{"NetworkCardIndex":0,"DeviceIndex":0,"Groups":
["sg-7c2270198example"],"InterfaceType":"efa","DeleteOnTermination":true},
{"NetworkCardIndex":1,"DeviceIndex":1,"Groups":
["sg-7c2270198example"],"InterfaceType":"efa","DeleteOnTermination":true}],"ImageId":"ami-09d95
```

> **Warning**
>
> p4d.24xlarge インスタンスタイプは、このセクションの他の例よりも高いコストが発生します。P4d インスタンス の料金の詳細については、「Amazon EC2 P4d インスタンスの料金」を参照してください。

> **Note**
>
> 同じサブネットから複数のネットワークインターフェイスをインスタンスにアタッチすると、特に Amazon Linux 以外のバリアントを使用するインスタンスでは、非対称ルーティングが発生する場合があります。このタイプの設定が必要な場合は、OS 内でセカンダリネットワークインターフェイスを設定する必要があります。例については、 AWS ナレッジセンターの「Ubuntu EC2 インスタンスでセカンダリネットワークインターフェイスを機能させるにはどうすればよいですか？」を参照してください。

起動テンプレートを管理する

AWS CLI には、起動テンプレートの管理に役立つ他のコマンドがいくつか含まれています。

内容

- 起動テンプレートをリストして記述する
- 起動テンプレートのバージョンの作成
- 起動テンプレートのバージョンの削除
- 起動テンプレートの削除

起動テンプレートをリストして記述する

起動テンプレートに関する情報を取得するには、describe-launch-templates と describe-launch-template-versions の 2 つの AWS CLI コマンドを使用できます。

describe-launch-templates コマンドを使用すると、作成済みの起動テンプレートのリストを取得できます。オプションを使用すると、起動テンプレート名、作成時間、タグキー、またはタグとキーバリューの組み合わせの結果をフィルタリングできます。このコマンドは、起動テンプレート識別子、最新バージョン、デフォルトバージョンなど、起動テンプレートに関する概要情報を返します。

次の例では、指定された起動テンプレートの概要を示します。

```
aws ec2 describe-launch-templates --launch-template-names my-template-for-auto-scaling
```

以下に、応答の例を示します。

```
{
    "LaunchTemplates": [
        {
            "LaunchTemplateId": "lt-068f72b729example",
            "LaunchTemplateName": "my-template-for-auto-scaling",
            "CreateTime": "2020-02-28T19:52:27.000Z",
            "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
            "DefaultVersionNumber": 1,
            "LatestVersionNumber": 1
        }
    ]
}
```

出力を 1 つの起動テンプレートだけに制限する `--launch-template-names` オプションを使用しない場合は、すべての起動テンプレートの情報が返されます。

次の「describe-launch-template-versions」コマンドは、指定された起動テンプレートのバージョンを説明する情報を提供します。

```
aws ec2 describe-launch-template-versions --launch-template-id lt-068f72b729example
```

以下に、応答の例を示します。

```
{
    "LaunchTemplateVersions": [
        {
            "VersionDescription": "version1",
            "LaunchTemplateId": "lt-068f72b729example",
            "LaunchTemplateName": "my-template-for-auto-scaling",
            "VersionNumber": 1,
            "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
            "LaunchTemplateData": {
                "TagSpecifications": [
                    {
                        "ResourceType": "instance",
                        "Tags": [
```

```
                        {
                            "Key": "purpose",
                            "Value": "webserver"
                        }
                    ]
                }
            ],
            "ImageId": "ami-04d5cc9b88example",
            "InstanceType": "t2.micro",
            "NetworkInterfaces": [
                {
                    "DeviceIndex": 0,
                    "DeleteOnTermination": true,
                    "Groups": [
                        "sg-903004f88example"
                    ],
                    "AssociatePublicIpAddress": true
                }
            ]
        },
        "DefaultVersion": true,
        "CreateTime": "2020-02-28T19:52:27.000Z"
    }
  ]
}
```

起動テンプレートのバージョンの作成

以下の create-launch-template-versions コマンドは、起動テンプレートのバージョン 1 に基づいて新しい起動テンプレートバージョンを作成し、異なる AMI ID を指定します。

```
aws ec2 create-launch-template-version --launch-template-id lt-068f72b729example --
version-description version2 \
  --source-version 1 --launch-template-data "ImageId=ami-c998b6b2example"
```

デフォルトバージョンの起動テンプレートを設定するには、modify-launch-templateコマンドを使用します。

起動テンプレートのバージョンの削除

以下の delete-launch-template-versions コマンドは、指定された起動テンプレートのバージョンを削除します。

```
aws ec2 delete-launch-template-versions --launch-template-id lt-068f72b729example --
versions 1
```

起動テンプレートの削除

起動テンプレートが不要になった場合には、以下のdelete-launch-templateeコマンドを実行します。起動テンプレートを削除すると、すべてのバージョンが削除されます。

```
aws ec2 delete-launch-template --launch-template-id lt-068f72b729example
```

起動テンプレートを使用するように Auto Scaling グループを更新する

update-auto-scaling-group コマンドを使用して、既存の Auto Scaling グループに起動テンプレートを追加できます。

最新バージョンの起動テンプレートを使用するように Auto Scaling グループを更新する

以下の update-auto-scaling-group コマンドは、指定された Auto Scaling グループを更新して、指定された起動テンプレートの最新バージョンを使用します。

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --launch-template LaunchTemplateId=lt-068f72b729example,Version='$Latest'
```

特定バージョンの起動テンプレートを使用するように Auto Scaling グループを更新する

以下の update-auto-scaling-group コマンドは、指定された Auto Scaling グループを更新して、指定された起動テンプレートの特定のバージョンを使用します。

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --launch-template LaunchTemplateName=my-template-for-auto-scaling,Version='2'
```

起動テンプレートで AMI IDs の代わりに AWS Systems Manager パラメータを使用する

このセクションでは、Amazon マシンイメージ (AMI) ID を参照する AWS Systems Manager パラメータを指定する起動テンプレートを作成する方法について説明します。同じ に保存されているパ

ラメータ AWS アカウント、別の から共有されているパラメータ AWS アカウント、または によって管理されているパブリック AMI のパブリックパラメータを使用できます AWS。

Systems Manager パラメータにより、AMI ID が変更されるたびに新しい起動テンプレートまたは新しいバージョンの起動テンプレートを作成しなくても、新しい AMI ID を使用するように Auto Scaling グループを更新できます。AMI が更新されて最新の OS またはソフトウェアアップデートが追加されたときなど、これらの ID は定期的に変更される可能性があります。

の一機能である Parameter Store を使用して、独自の Systems Manager パラメータを作成、更新、AWS Systems Managerまたは削除できます。起動テンプレートで Systems Manager パラメータを使用する前に、作成する必要があります。開始するには、データタイプ `aws:ec2:image` でパラメータを作成し、値については AMI の ID を入力できます。AMI ID は形式 `ami-<identifier>` があり、`ami-123example456` はその例です。正しい AMI ID は、Auto Scaling グループを起動するインスタンスタイプおよび AWS リージョン によって異なります。

AMI ID の有効なパラメータの作成の詳細については、「Systems Manager パラメータの作成」を参照してください。

AMI のパラメータを指定する起動テンプレートを作成する

AMI のパラメータを指定する起動テンプレートを作成するには、次のいずれかの方法を使用します。

Console

AWS Systems Manager パラメータを使用して起動テンプレートを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[起動テンプレート]、[起動テンプレートの作成] の順に選択します。
3. [起動テンプレート名] に、起動テンプレートのわかりやすい名前を入力します。
4. [Application and OS Images (Amazon Machine Image)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で、[Browse more AMIs] (その他の AMI を閲覧する) を選択します。
5. 検索バーの右側にある矢印ボタンを選択したら、[カスタム値を指定 / Systems Manager パラメータ] を選択します。
6. [カスタム値または Systems Manager のパラメータを指定] ダイアログボックスで、次の手順を実行します。

a. [AMI ID または Systems Manager パラメータ文字列] の場合、次の形式のいずれかを使用して Systems Manager パラメータ名を入力します。

パブリックパラメータを参照するには:

- **resolve:ssm:*public-parameter***

同じアカウントに保存されているパラメータを参照するには:

- **resolve:ssm:*parameter-name***
- **resolve:ssm:*parameter-name*:*version-number***
- **resolve:ssm:*parameter-name*:*label***

別の AWS アカウントから共有されたパラメータを参照するには:

- **resolve:ssm:*parameter-ARN***
- **resolve:ssm:*parameter-ARN*:*version-number***
- **resolve:ssm:*parameter-ARN*:*label***

b. [保存] を選択します。

7. 必要に応じて、他の起動テンプレート設定を実行してから [起動テンプレートを作成] を選択します。詳細については、「Auto Scaling グループの起動テンプレートを作成する」を参照してください。

AWS CLI

Systems Manager パラメータを指定する起動テンプレートを作成するには、次のいずれかのサンプルコマンドを使用できます。各*###############*を独自の情報に置き換えます。

例: AWS所有のパブリックパラメータを指定する起動テンプレートを作成する

`resolve:ssm:`*`public-parameter`* という構文を使用します。ただし、`resolve:ssm` は標準のプレフィクスで *`public-parameter`* はパブリックパラメータのパスおよび名前です。

この例では、起動テンプレートは AWSが提供するパブリックパラメータを使用して、プロファイル用に AWS リージョン 設定された で最新の Amazon Linux 2 AMI を使用してインスタンスを起動します。

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling
 --version-description version1 \
  --launch-template-data file://config.json
```

config.json の内容:

```
{
    "ImageId":"resolve:ssm:/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-
x86_64-gp2",
    "InstanceType":"t2.micro"
}
```

以下に、応答の例を示します。

```
{
    "LaunchTemplate": {
        "LaunchTemplateId": "lt-089c023a30example",
        "LaunchTemplateName": "my-template-for-auto-scaling",
        "CreateTime": "2022-12-28T19:52:27.000Z",
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "DefaultVersionNumber": 1,
        "LatestVersionNumber": 1
    }
}
```

例: 同じアカウントに保存されているパラメータを指定する起動テンプレートを作成する

resolve:ssm:*parameter-name* という構文を使用します。ただし、resolve:ssm は標準のプレフィックスで *parameter-name* は Systems Manager のパラメータ名です。

次の例では、*golden-ami* という名前の既存の Systems Manager パラメータから AMI ID を取得する起動テンプレートを作成します。

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling \
  --launch-template-data file://config.json
```

config.json の内容:

```
{
```

```
    "ImageId":"resolve:ssm:golden-ami",
    "InstanceType":"t2.micro"
}
```

パラメータのデフォルトバージョンを指定しない場合、最新バージョンです。

次の例では、*golden-ami* パラメータの特定のバージョンを参照しています。この例では、*golden-ami* パラメータのバージョン *3* を使用していますが、有効なバージョン番号であればどれでも使用できます。

```
{
    "ImageId":"resolve:ssm:golden-ami:3",
    "InstanceType":"t2.micro"
}
```

次の同様の例では、*golden-ami* パラメータの特定のバージョンにマップするパラメータラベル *prod* を参照しています。

```
{
    "ImageId":"resolve:ssm:golden-ami:prod",
    "InstanceType":"t2.micro"
}
```

以下は出力例です。

```
{
    "LaunchTemplate": {
        "LaunchTemplateId": "lt-068f72b724example",
        "LaunchTemplateName": "my-template-for-auto-scaling",
        "CreateTime": "2022-12-27T17:11:21.000Z",
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "DefaultVersionNumber": 1,
        "LatestVersionNumber": 1
    }
}
```

例: 別の から共有されるパラメータを指定する起動テンプレートを作成する AWS アカウント

次の構文を使用します: 。ここでresolve:ssm:*parameter-ARN*、 resolve:ssmは標準プレフィックスで、 *parameter-ARN*は Systems Manager パラメータの ARN です。

次の例では、ARN が の既存の Systems Manager パラメータから AMI ID を取得する起動テンプレートを作成します*arn:aws:ssm:us-east-2:123456789012:parameter/MyParameter*。

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling
 --version-description version1 \
  --launch-template-data file://config.json
```

config.json の内容:

```
{
    "ImageId":"resolve:ssm:arn:aws:ssm:us-east-2:123456789012:parameter/
MyParameter",
    "InstanceType":"t2.micro"
}
```

パラメータのデフォルトバージョンを指定しない場合、最新バージョンです。

次の例では、*MyParameter* パラメータの特定のバージョンを参照しています。この例では、*MyParameter* パラメータのバージョン *3* を使用していますが、有効なバージョン番号であればどれでも使用できます。

```
{
    "ImageId":"resolve:ssm:arn:aws:ssm:us-east-2:123456789012:parameter/
MyParameter:3",
    "InstanceType":"t2.micro"
}
```

次の同様の例では、*MyParameter* パラメータの特定のバージョンにマップするパラメータラベル *prod* を参照しています。

```
{
    "ImageId":"resolve:ssm:arn:aws:ssm:us-east-2:123456789012:parameter/
MyParameter:prod",
    "InstanceType":"t2.micro"
}
```

以下に、応答の例を示します。

```
{
```

```
    "LaunchTemplate": {
        "LaunchTemplateId": "lt-00f93d4588example",
        "LaunchTemplateName": "my-template-for-auto-scaling",
        "CreateTime": "2024-01-08T12:43:21.000Z",
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "DefaultVersionNumber": 1,
        "LatestVersionNumber": 1
    }
}
```

起動テンプレートで Parameter Store からパラメータを指定するには、指定されたパラメータに対する アクセスssm:GetParameters許可が必要です。起動テンプレートを使用するユーザーには、パラメータ値を検証するための アクセスssm:GetParameters許可も必要です。詳細については、「AWS Systems Manager ユーザーガイド」の「IAM ポリシーを使用した Systems Manager パラメータへのアクセスの制限」を参照してください。

起動テンプレートが正しい AMI ID を取得することを確認する

describe-launch-template-versions コマンドを使用して、 パラメータを実際の AMI ID に解決する --resolve-aliasオプションを含めます。

```
aws ec2 describe-launch-template-versions --launch-template-name my-template-for-auto-scaling \
   --versions $Default --resolve-alias
```

この例では、ImageId に AMI ID を返します。この起動テンプレートを使用してインスタンスを起動するとき、AMI ID は ami-0ac394d6a3example に変換されます。

```
{
    "LaunchTemplateVersions": [
        {
            "LaunchTemplateId": "lt-089c023a30example",
            "LaunchTemplateName": "my-template-for-auto-scaling",
            "VersionNumber": 1,
            "CreateTime": "2022-12-28T19:52:27.000Z",
            "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
            "DefaultVersion": true,
            "LaunchTemplateData": {
                "ImageId": "ami-0ac394d6a3example",
                "InstanceType": "t2.micro",
```

```
            }
        }
    ]
}
```

関連リソース

起動テンプレートで Systems Manager パラメータを指定する方法の詳細については、Amazon EC2 ユーザーガイド」の「AMI ID の代わりに Systems Manager パラメータを使用する」を参照してください。

Systems Manager パラメータの使用に関する詳細は、Systems Manager のドキュメントの次のリファレンス資料を参照してください。

- パラメータバージョンとラベルを作成するには、「パラメータバージョンの使用」および「パラメータラベルの使用」を参照してください。
- Amazon EC2 でサポートされている AMI パブリックパラメータを検索する方法については、「Calling AMI public parameters」を参照してください。
- 他の AWS アカウントと、または を介してパラメータを共有する方法については AWS Organizations、「共有パラメータの使用」を参照してください。
- パラメータが正常に作成されたかどうかをモニタリングする方法については、「Native parameter support for Amazon Machine Image IDs」を参照してください。

制限事項

Systems Manager パラメータを使用する場合は、次の制限に注意してください。

- Amazon EC2 Auto Scaling は、AMI ID をパラメータとして指定することのみサポートします。
- Systems Manager パラメータを指定する起動テンプレートを使用した混合インスタンスグループの作成または更新は、現在サポートされていません。
- Auto Scaling グループが Systems Manager パラメータを指定する起動テンプレートを使用している場合、必要な設定またはスキップマッチングを使用してインスタンスの更新を開始することはできません。
- Auto Scaling グループを作成または更新する呼び出しのたびに、Amazon EC2 Auto Scaling は起動テンプレートの Systems Manager パラメータを変換します。高度なパラメータまたはより高いスループット制限を使用している場合、Parameter Store (つまり、`GetParameters` 操作) への頻繁な呼び出しは、Parameter Store API 対話ごとに料金が発生するため、Systems Manager のコスト

が増加する可能性があります。詳細については、「AWS Systems Manager 料金」を参照してください。

起動設定

> **Important**
>
> 2022 年 12 月 31 日以降にリリースされた新しい Amazon EC2 インスタンスタイプで `CreateLaunchConfiguration` を呼び出すことはできません。また、2023 年 6 月 1 日以降に作成された新しいアカウントには、コンソールから新しい起動設定を作成することはできません。今後、新しいアカウントは、コンソール、API、CLI、および を使用して新しい起動設定を作成できなくなります CloudFormation。起動テンプレートに移行して、現在または将来に新しい起動設定を作成する必要がないようにします。Auto Scaling グループを移行してテンプレートを起動する方法については、「Auto Scaling グループを起動テンプレートに移行する」を参照してください。

起動設定は、Auto Scaling グループが EC2 インスタンスを起動するために使用するインスタンス設定テンプレートです。起動設定を作成する場合は、インスタンスの情報を指定します。Amazon マシンイメージ (AMI) の ID、インスタンスタイプ、キーペア、1 つ以上のセキュリティグループ、ブロックデバイスマッピングなどがあります。以前 EC2 インスタンスを起動している場合は、インスタンスを起動するために同じ情報を指定しました。

複数の Auto Scaling グループについて起動設定を指定できます。ただし、1 つの Auto Scaling グループに指定できる起動設定は 1 つだけであり、グループを作成した後で起動設定を変更することはできません。Auto Scaling グループの起動設定を変更するには、起動設定を作成し、それを使用して Auto Scaling グループを更新する必要があります。

内容

- 「起動設定を作成する」
- Auto Scaling グループの起動設定を変更する

「起動設定を作成する」

> **Important**
>
> 2022 年 12 月 31 日以降にリリースされた新しい Amazon EC2 インスタンスタイプで `CreateLaunchConfiguration` を呼び出すことはできません。また、2023 年 6 月 1 日以降に作成された新しいアカウントには、コンソールから新しい起動設定を作成することはで

きません。今後、新しいアカウントは、コンソール、API、CLI、および を使用して新しい起動設定を作成できなくなります CloudFormation。起動テンプレートに移行して、現在または将来に新しい起動設定を作成する必要がないようにします。Auto Scaling グループを移行してテンプレートを起動する方法については、「Auto Scaling グループを起動テンプレートに移行する」を参照してください。

このトピックでは、起動設定を作成する方法について説明します。

起動設定を作成した後は、変更できません。代わりに、新しい起動設定を作成する必要があります。

新しい起動設定を既存の Auto Scaling グループに関連付けるには、「」を参照してくださいAuto Scaling グループの起動設定を変更する。新しい Auto Scaling グループを作成するには、「」を参照してください起動設定を使用して Auto Scaling グループを作成する。

内容

- 「起動設定を作成する」
- インスタンスメタデータオプションの設定
- EC2 インスタンスを使用した起動設定の作成

「起動設定を作成する」

起動設定を作成するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 上部のナビゲーションバーで、 AWS リージョンを選択します。
3. 左のナビゲーションペインの [Auto Scaling] で、[Auto Scaling グループ] を選択します。
4. ページの上部付近にある [起動設定] を選択します。確認を求めるプロンプトが表示されたら、[起動設定を表示] を選択して、[起動設定] ページを表示することを確認します。
5. [Create launch configuration (起動設定の作成)] を選択して、起動設定の名前を入力します。
6. Amazon マシンイメージ (AMI) で、 AMI を選択します。特定の AMI を検索するには、[find a suitable AMI (適切な AMI を見つける)] をクリックし、その ID を書き留め、検索条件として ID を入力します。

   Amazon Linux 2 AMI の ID を取得するには:

   a. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

b. 左側のナビゲーションペインの [インスタンス] で、[インスタンス] を選択し、[インスタンスを起動] を選択します。

c. [Choose an Amazon Machine Image (Amazon Machine Image の選択)] ページの [クイックスタート] タブで、[Amazon Linux 2 AMI (HVM)] の横にある AMI の ID をメモします。

7. [インスタンスタイプのタイプ] で、インスタンスのハードウェア設定を選択します。

8. [Additional configuration (追加設定)] の下にある、次のフィールドに注意してください:

a. (オプション) [Purchasing option (購入オプション)] で、[Request Spot Instances (スポットインスタンスのリクエスト)] を選択し、オンデマンド価格を上額とするスポット料金でスポットインスタンスをリクエストできます。オプションで、スポットインスタンスのインスタンス時間ごとに上限価格を指定できます。

> **Note**
>
> スポットインスタンスは、アプリケーションを実行する時間に柔軟性がある場合や、アプリケーションを中断できる場合に、オンデマンドインスタンスと比べて費用効率の高い選択肢です。詳細については、「耐障害性に優れた柔軟なアプリケーションのためにスポットインスタンスをリクエストする」を参照してください。

b. (オプション) [IAM instance profile (IAM インスタンスプロファイル)] で、インスタンスに関連付けるロールを選択します。詳細については、「Amazon EC2 インスタンスで実行中のアプリケーション用の IAM ロール」を参照してください。

c. (オプション) モニタリング で、詳細モニタリングを有効に CloudWatch して、インスタンスがメトリクスデータを 1 分間隔で Amazon に発行できるようにするかどうかを選択します。別途 料金がかかります。詳細については、「Auto Scaling インスタンスのモニタリングを設定する」を参照してください。

d. (オプション) [Advanced details (詳細情報)]、[User data (ユーザーデータ)] で、起動中にインスタンスを設定するユーザーデータ、またはインスタンスの起動後に設定スクリプトを実行するユーザーデータを指定できます。

e. (オプション) [Advanced details (詳細情報)]、[IP address type (IP アドレスタイプ)]で、パブリック IP アドレスをグループのインスタンスに割り当てるかどうか選択します。値を設定しない場合、デフォルトでは、インスタンスが起動されるサブネットの自動割り当てパブリック IP 設定が使用されます。

9. (オプション) [Storage (volumes) (ストレージボリューム)] で、追加のストレージが必要ない場合、このセクションをスキップできます。それ以外の場合は、AMI で指定されたボリュームに加えてインスタンスにアタッチするボリュームを指定するには、[Add new volume (新しいボ

リュームの追加)] を選択します。次に、デバイス、スナップショット、サイズ、ボリュームタイプ、IOPS、スループット、終了時に削除、および暗号化に必要なオプションと関連する値を選択します。

10. [Security groups (セキュリティグループ)] で、グループのインスタンスに関連付けるセキュリティグループを作成または選択します。選択した [Create a new security group (新しいセキュリティグループの作成)] オプションから離れる場合、Linux を実行する Amazon EC2 インスタンスのデフォルトの SSH ルールが設定されます。デフォルトの RDP ロールは、Windows を実行する Amazon EC2 インスタンスに対して設定されます。
11. [Key pair (login) (キーペア (ログイン)] で、[Key pair options (キーペアのオプション)] の下にあるオプションを選択します。

    すでに Amazon EC2 インスタンスキーペアを設定している場合は、ここで選択できます。

    Amazon EC2 インスタンスのキーペアがまだない場合は、[Create a new key pair (新しいキーペアの作成)] を選択して、わかりやすい名前を付けます。[Download key pair (キーペアのダウンロード)] を選択し、コンピュータにダウンロードします。

    > **Important**
    >
    > インスタンスに接続する必要がある場合は、[Proceed without a key pair (キーペアなしで続行する)] を選択しないでください。

12. 確認チェックボックスをオンにし、[Create launch configuration (起動設定の作成)] を選択します。

既存の起動設定から起動設定を作成するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 上部のナビゲーションバーで、 AWS リージョンを選択します。
3. 左のナビゲーションペインの [Auto Scaling] で、[Auto Scaling グループ] を選択します。
4. ページの上部付近にある [起動設定] を選択します。確認を求めるプロンプトが表示されたら、[起動設定を表示] を選択して、[起動設定] ページを表示することを確認します。
5. 起動設定を選択し、[Actions]、[Copy launch configuration] の順に選択します。これにより、新しい起動設定は元の設定と同じオプションで設定されますが、名前に「コピー」が追加されます。

6. [Copy Launch Configuration] ページで、必要に応じて設定オプションを編集し、[Create launch configuration] を選択します。

コマンドラインを使用して起動設定を作成するには

以下のコマンドのいずれかを使用できます。

- create-launch-configuration (AWS CLI)
- 新しい ASLaunchConfiguration (AWS Tools for Windows PowerShell)

インスタンスメタデータオプションの設定

Amazon EC2 Auto Scaling は、起動設定のインスタンスメタデータサービス (IMDS) の設定をサポートしています。このオプションにより、起動設定を使用して、Auto Scaling グループ内の Amazon EC2 インスタンスを、インスタンスメタデータサービスバージョン 2 (IMDSv2) をリクエストするように設定できます。これは、インスタンスメタデータを要求するためのセッション指向の方法です。IMDSv2 の利点の詳細については、「EC2 インスタンスメタデータサービスに多層防御を追加する拡張機能についての AWS ブログ」の記事を参照してください。

IMDSv2 と IMDSv1 の両方をサポートするように (デフォルト)、または IMDSv2 の使用をリクエストするように IMDS を設定できます。 AWS CLI またはいずれかの SDKs を使用して IMDS を設定している場合は、最新バージョンの AWS CLI または SDK を使用して IMDSv2 の使用を要求する必要があります。

起動設定は次の設定を指定できます。

- インスタンスメタデータをリクエストするときに IMDSv2 の使用を要求する
- PUT レスポンスのホップ制限を指定する
- インスタンスメタデータへのアクセスを無効にする

インスタンスメタデータサービスの設定の詳細については、Amazon EC2 ユーザーガイドの「インスタンスメタデータサービスの設定」を参照してください。 Amazon EC2

起動設定で IMDS オプションを設定するには、以下の手順を使用します。起動設定を作成したら、それを Auto Scaling グループに関連付けることができます。起動設定を既存の Auto Scaling グループに関連付けると、既存の起動設定と Auto Scaling グループとの関連付けが解除され、新しい起動設定で指定した IMDS オプションを使用するには、既存のインスタンスを置き換える必要があります。詳細については、「Auto Scaling グループの起動設定を変更する」を参照してください。

起動設定で IMDS を設定するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 上部のナビゲーションバーで、 AWS リージョンを選択します。
3. 左のナビゲーションペインの [Auto Scaling] で、[Auto Scaling グループ] を選択します。
4. ページの上部付近にある [起動設定] を選択します。確認を求めるプロンプトが表示されたら、[起動設定を表示] を選択して、[起動設定] ページを表示することを確認します。
5. [Create launch configuration (起動設定の作成)] を選択し、通常の方法で起動設定を作成します。Amazon Machine Image (AMI) の ID、インスタンスタイプ、オプションでキーペア、1 つ以上のセキュリティグループ、およびインスタンス用の追加の EBS ボリュームまたはインスタンスストアボリュームなどがあります。
6. この起動設定に関連付けられているすべてのインスタンスのインスタンスメタデータオプションを設定するには、[Additional configuration (追加設定)] の下にある [Advanced details (詳細情報)] で次の作業を行います。

   a. [Metadata accessible (アクセス可能なメタデータ)] で、インスタンスメタデータサービスの HTTP エンドポイントへのアクセスを有効または無効にするか選択します。デフォルトでは、HTTP エンドポイントは有効です。エンドポイントを無効にすると、インスタンスメタデータへのアクセスはオフになります。HTTP エンドポイントが有効になっている場合にのみ、IMDSv2 を要求する条件を指定できます。

   b. [Metadata version] で、インスタンスメタデータをリクエストするときにインスタンスメタデータサービスバージョン 2 (IMDSv2) の使用を必須にすることができます。値を指定しない場合、デフォルトで IMDSv1 と IMDSv2 の両方がサポートされます。

   c. [Metadata token response hop limit] で、メタデータトークンに許容されるネットワークホップ数を設定できます。値を指定していない場合、デフォルトは 1 です。
7. 完了したら、[Create launch configuration (起動設定の作成)] を選択します。

起動設定で IMDSv2 の使用をリクエストするには、 AWS CLIを使用します。

`HttpTokens=required` に設定されている `--metadata-options` と使用して、次の create-launch-configurationコマンドを使用します。`HttpTokens` の値を指定する場合は、`HttpEndpoint` も有効にする必要があります。メタデータの取得リクエストではセキュリティで保護されたトークンヘッダーが必須と設定されるため、インスタンスメタデータをリクエストするときに IMDSv2 の使用をリクエストするようにインスタンスが設定されます。

```
aws autoscaling create-launch-configuration \
```

```
  --launch-configuration-name my-lc-with-imdsv2 \
  --image-id ami-01e24be29428c15b2 \
  --instance-type t2.micro \
      ...
  --metadata-options "HttpEndpoint=enabled,HttpTokens=required"
```

インスタンスメタデータへのアクセスを無効にするには

インスタンスメタデータへのアクセスをオフにするために、次の create-launch-configuration を使用します。アクセスに後で戻るには、modify-instance-metadata-options コマンドを使用します。

```
aws autoscaling create-launch-configuration \
  --launch-configuration-name my-lc-with-imds-disabled \
  --image-id ami-01e24be29428c15b2 \
  --instance-type t2.micro \
      ...
  --metadata-options "HttpEndpoint=disabled"
```

EC2 インスタンスを使用した起動設定の作成

実行中の EC2 インスタンスの属性を使用して起動設定を作成するオプションもあります。

起動設定を新しく作成することと、既存の EC2 インスタンスから起動設定を作成することには違いがあります。起動設定をゼロから作成するときは、イメージ ID、インスタンスタイプ、オプションリソース (ストレージデバイスなど) 、オプション設定 (モニタリングなど) を指定します。実行中のインスタンスから起動設定を作成すると、Amazon EC2 Auto Scaling は指定されたインスタンスから起動設定の属性を取得します。属性は、インスタンスが起動された AMI のブロックデバイスマッピングからも取得されますが、起動後に追加されたブロックデバイスは無視されます。

実行中のインスタンスを使用して起動設定を作成するときは、同じリクエストの一部としてそれらを指定することによって、次の属性を変更できます。AMI、ブロックデバイス、キーペア、インスタンスプロファイル、インスタンスタイプ、カーネル、インスタンスのモニタリング、プレイスメントテナンシー、ラムディスク、セキュリティグループ、スポット (最大) 料金、ユーザーデータ、インスタンスにパブリック IP アドレスがあるか、インスタンスが EBS で最適化されているかどうか。

> **Note**
>
> 指定されたインスタンスに起動設定で現在サポートされていないプロパティがある場合、Auto Scaling グループによって起動されるインスタンスは、元の EC2 インスタンスと同じにならない場合があります。

> **Important**
>
> 指定したインスタンスを起動するために使用される AMI がまだ存在する必要があります。

トピック

- EC2 インスタンスから起動設定を作成する (AWS CLI)
- インスタンスから起動設定を作成し、ブロックデバイスをオーバーライドする (AWS CLI)
- 起動設定を作成し、インスタンスタイプをオーバーライドする (AWS CLI)

EC2 インスタンスから起動設定を作成する (AWS CLI)

create-launch-configuration インスタンスから、そのインスタンスと同じ属性を使用して起動設定を作成するには、次の コマンドを使用します。起動後に追加されたブロックデバイスはすべて無視されます。

```
aws autoscaling create-launch-configuration --launch-configuration-name my-lc-from-instance --instance-id i-a8e09d9c
```

次の describe-launch-configurations コマンドを使用すると、起動設定を記述し、その属性がインスタンスの属性と一致することを確認できます。

```
aws autoscaling describe-launch-configurations --launch-configuration-names my-lc-from-instance
```

以下に、応答の例を示します。

```
{
    "LaunchConfigurations": [
        {
            "UserData": null,
            "EbsOptimized": false,
            "LaunchConfigurationARN": "arn",
            "InstanceMonitoring": {
                "Enabled": false
            },
            "ImageId": "ami-05355a6c",
            "CreatedTime": "2014-12-29T16:14:50.382Z",
```

```
            "BlockDeviceMappings": [],
            "KeyName": "my-key-pair",
            "SecurityGroups": [
                "sg-8422d1eb"
            ],
            "LaunchConfigurationName": "my-lc-from-instance",
            "KernelId": "null",
            "RamdiskId": null,
            "InstanceType": "t1.micro",
            "AssociatePublicIpAddress": true
        }
    ]
}
```

インスタンスから起動設定を作成し、ブロックデバイスをオーバーライドする (AWS CLI)

Amazon EC2 Auto Scaling は、デフォルトで、指定された EC2 インスタンスの属性を使用して起動設定を作成します。ただし、インスタンスではなく、インスタンスの起動に使用された AMI のブロックデバイスが使用されます。起動設定にブロックデバイスを追加するには、起動設定のブロックデバイスマッピングをオーバーライドします。

EC2 インスタンスでカスタムブロックデバイスマッピングを使用して起動設定を作成するには、次の create-launch-configuration コマンドを使用します。

```
aws autoscaling create-launch-configuration --launch-configuration-name my-lc-from-instance-bdm --instance-id i-a8e09d9c \
  --block-device-mappings "[{\"DeviceName\":\"/dev/sda1\",\"Ebs\":{\"SnapshotId\":\"snap-3decf207\"}},{\"DeviceName\":\"/dev/sdf\",\"Ebs\":{\"SnapshotId\":\"snap-eed6ac86\"}}]"
```

起動設定を記述し、それがカスタムブロックデバイスマッピングを使用していることを確認するには、次の describe-launch-configurations コマンドを使用します。

```
aws autoscaling describe-launch-configurations --launch-configuration-names my-lc-from-instance-bdm
```

以下の応答例は、起動設定の詳細を表しています。

```
{
```

```
    "LaunchConfigurations": [
        {
            "UserData": null,
            "EbsOptimized": false,
            "LaunchConfigurationARN": "arn",
            "InstanceMonitoring": {
                "Enabled": false
            },
            "ImageId": "ami-c49c0dac",
            "CreatedTime": "2015-01-07T14:51:26.065Z",
            "BlockDeviceMappings": [
                {
                    "DeviceName": "/dev/sda1",
                    "Ebs": {
                        "SnapshotId": "snap-3decf207"
                    }
                },
                {
                    "DeviceName": "/dev/sdf",
                    "Ebs": {
                        "SnapshotId": "snap-eed6ac86"
                    }
                }
            ],
            "KeyName": "my-key-pair",
            "SecurityGroups": [
                "sg-8637d3e3"
            ],
            "LaunchConfigurationName": "my-lc-from-instance-bdm",
            "KernelId": null,
            "RamdiskId": null,
            "InstanceType": "t1.micro",
            "AssociatePublicIpAddress": true
        }
    ]
}
```

起動設定を作成し、インスタンスタイプをオーバーライドする (AWS CLI)

Amazon EC2 Auto Scaling は、デフォルトで、指定された EC2 インスタンスの属性を使用して起動設定を作成します。要件によっては、インスタンスの属性をオーバーライドし、必要な値を使用する場合があります。たとえば、インスタンスタイプをオーバーライドできます。

EC2 インスタンスを使用して起動設定を作成するために、次の create-launch-configuration コマンドを使用しますが、インスタンス (たとえば t2.micro) よりも異なるインスタンスタイプ (例:t2.medium) を使用します。

```
aws autoscaling create-launch-configuration --launch-configuration-name my-lc-from-instance-changetype \
  --instance-id i-a8e09d9c --instance-type t2.medium
```

起動設定を記述し、インスタンスタイプが上書きされたことを確認するには、次の describe-launch-configurations コマンドを使用します。

```
aws autoscaling describe-launch-configurations --launch-configuration-names my-lc-from-instance-changetype
```

以下の応答例は、起動設定の詳細を表しています。

```
{
    "LaunchConfigurations": [
        {
            "UserData": null,
            "EbsOptimized": false,
            "LaunchConfigurationARN": "arn",
            "InstanceMonitoring": {
                "Enabled": false
            },
            "ImageId": "ami-05355a6c",
            "CreatedTime": "2014-12-29T16:14:50.382Z",
            "BlockDeviceMappings": [],
            "KeyName": "my-key-pair",
            "SecurityGroups": [
                "sg-8422d1eb"
            ],
            "LaunchConfigurationName": "my-lc-from-instance-changetype",
            "KernelId": "null",
            "RamdiskId": null,
            "InstanceType": "t2.medium",
            "AssociatePublicIpAddress": true
        }
    ]
}
```

Auto Scaling グループの起動設定を変更する

> **Important**
>
> 起動設定に関する情報は、起動設定から起動テンプレートにまだ移行していないお客様向けに提供しています。Auto Scaling グループを移行してテンプレートを起動する方法については、「Auto Scaling グループを起動テンプレートに移行する」を参照してください。

このトピックでは、別の起動設定を Auto Scaling グループに関連付ける方法について説明します。

起動設定を変更すると、新しいインスタンスは新しい設定オプションを使用して起動されますが、既存のインスタンスは影響を受けません。詳細については、「Auto Scaling インスタンスの更新」を参照してください。

Auto Scaling グループの起動設定を置き換えるには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左のナビゲーションペインの [Auto Scaling] で、[Auto Scaling グループ] を選択します。
3. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。
4. [Details] (詳細) タブで、[Launch configuration] (起動設定)、[Edit] (編集) の順に選択します。
5. 起動設定 で、起動設定を選択します。
6. 完了したら、[更新] を選択します。

コマンドラインを使用して Auto Scaling グループの起動設定を変更するには

以下のコマンドのいずれかを使用できます。

- update-auto-scaling-group (AWS CLI)
- Update-AS AutoScalingグループ (AWS Tools for Windows PowerShell)

「Auto Scaling グループ」

> **Note**
>
> Auto Scaling グループを初めて使用する場合は、「最初の Auto Scaling グループの作成」チュートリアルの手順を実行して開始し、グループ内のインスタンスが終了したときに Auto Scaling グループがどのように応答するかを確認します。

Auto Scaling グループには、自動スケーリングと管理の目的で論理グループとして扱われる EC2 インスタンスの集合が含まれます。また、Auto Scaling グループによって、ヘルスチェックの置き換えやスケーリングポリシーなど、Amazon EC2 Auto Scaling の機能も使用できます。Auto Scaling グループでのインスタンス数の維持と自動スケーリングの両方が Amazon EC2 Auto Scaling サービスの主な機能です。

Auto Scaling グループのサイズは、希望するキャパシティーとして設定したインスタンスの数によって異なります。手動でまたは自動スケーリングを使用して、需要に合わせてサイズを調整できます。

Auto Scaling グループは起動時に、希望するキャパシティーに対応するために十分な数のインスタンスを起動します。また、グループのインスタンスに対して定期的にヘルスチェックを実行することで、このインスタンス数を維持します。Auto Scaling グループは、インスタンスに異常が発生した場合でも、一定数のインスタンスを維持し続けます。インスタンスに異常が生じた場合、グループは異常のあるインスタンスを終了し、そのインスタンスを置き換える別のインスタンスを起動します。詳細については、「Auto Scaling グループ内のインスタンスのヘルスチェック」を参照してください。

スケーリングポリシーを使用して、状況の変化に合わせてグループのインスタンスの数を動的に増減することができます。スケーリングポリシーが有効になっている場合、Auto Scaling グループは、指定された最小キャパシティ値と最大キャパシティ値の間で、グループの希望するキャパシティを調整し、必要に応じてインスタンスを起動または終了します。スケーリングはスケジュールに従って行うこともできます。詳細については、「スケーリング方法を選択する」を参照してください。

Auto Scaling グループを作成する際に、オンデマンドインスタンス、スポットインスタンス、またはその両方を起動するかどうかを選択できます。起動テンプレートを使用する場合にのみ、Auto Scaling グループに複数の購入オプションを指定できます。詳細については、「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。

スポットインスタンスでは、オンデマンド料金と比べて大幅な割引料金で、未使用の EC2 キャパシティを購入できます。詳細については、「Amazon EC2スポットインスタンス」を参照してください。スポットインスタンスとオンデマンドインスタンスの主な違いは次のとおりです。

- スポットインスタンスの料金は需要に応じて変化する
- スポットインスタンスの可用性や料金が変化したときに、Amazon EC2 が個々のスポットインスタンスを終了できる

スポットインスタンスが削除されると、Auto Scaling グループはグループの希望するキャパシティを維持するために、代わりのインスタンスを起動しようとします。

インスタンスの起動時に複数のアベイラビリティーゾーンを指定した場合は、希望するキャパシティがこれらのアベイラビリティーゾーンに分散されます。スケーリングアクションが発生した場合、Amazon EC2 Auto Scaling は指定したすべてのアベイラビリティーゾーン間でキャパシティーのバランスを自動的に維持します。

内容

- 起動テンプレートを使用して Auto Scaling グループを作成する
- 起動設定を使用して Auto Scaling グループを作成する
- Auto Scaling グループを更新する
- Auto Scaling グループとインスタンスにタグを付ける
- インスタンスのメンテナンスポリシー
- Amazon EC2 Auto Scaling のライフサイクルフック
- Amazon EC2 Auto Scaling のウォームプール
- インスタンスのデタッチまたはアタッチ
- Auto Scaling グループからインスタンスを一時的に削除する
- Auto Scaling インフラストラクチャを削除する
- AWS SDKs を使用した Auto Scaling グループの作成と管理の例

起動テンプレートを使用して Auto Scaling グループを作成する

起動テンプレートを作成した場合は、起動テンプレートを EC2 インスタンスの設定テンプレートとして使用する Auto Scaling グループを作成できます。起動テンプレートは、インスタンスの AMI ID、インスタンスタイプ、キーペア、セキュリティグループ、ブロックデバイスマッピングなどの情

報を指定します。起動テンプレートの作成の詳細については、「Auto Scaling グループの起動テンプレートを作成する」を参照してください。

Auto Scaling グループを作成するための十分な許可が必要です。また、サービスにリンクされたロールが存在しない場合は、Amazon EC2 Auto Scaling がユーザーに代わってアクションを実行するために使用する、サービスにリンクされたロールを作成するための十分な許可も必要です。管理者がアクセス許可を付与するための参照として使用できる IAM ポリシーの例については、アイデンティティベースポリシーの例 および 起動テンプレートのサポート を参照してください。

内容

- 起動テンプレートを使用して Auto Scaling グループを作成する
- Amazon EC2 起動ウィザードを使用して Auto Scaling グループを作成する
- 複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ

起動テンプレートを使用して Auto Scaling グループを作成する

Auto Scaling グループを作成する際、Amazon EC2 インスタンスを設定するために必要な情報、そのインスタンスのアベイラビリティーゾーンと VPC サブネット、希望するキャパシティー、キャパシティー制限の最小値と最大値を指定する必要があります。

Auto Scaling グループによって起動される Amazon EC2 インスタンスを設定するには、起動テンプレートまたは起動設定を指定します。次の手順では、起動テンプレートを使用して Auto Scaling グループを作成する方法を説明します。

前提条件

- 起動テンプレートを作成しておく必要があります。詳細については、「Auto Scaling グループの起動テンプレートを作成する」を参照してください。

起動テンプレートを使用して Auto Scaling グループを作成するには (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. 画面上部のナビゲーションバーで、起動テンプレートの作成時に使用した AWS リージョン ものと同じ を選択します。
3. [Auto Scaling グループの作成] を選択します。
4. Choose launch template or configuration のページで、以下を実行します。

a. Auto Scaling グループ名に　Auto Scaling グループの名前を入力します。

b. [起動テンプレート] で、既存の起動テンプレートを選択します。

c. [起動テンプレートのバージョン] で、スケールアウト時に Auto Scaling グループで使用する起動テンプレートのバージョン (デフォルト、最新、または特定のバージョン) を選択します。

d. 起動テンプレートが、使用する予定のすべてのオプションをサポートしていることを確認し、[次へ] を選択します。

5. [インスタンス起動オプションを選択] ページで、複数のインスタンスタイプを使用していない場合は [インスタンスタイプの要件] セクションをスキップして、起動テンプレートで指定されている EC2 インスタンスタイプを使用できます。

   複数のインスタンスタイプを使用する場合は、「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。

6. [Network] (ネットワーク) の下にある [VPC] で、VPC を選択します。Auto Scaling グループは、起動テンプレートで指定したセキュリティグループと、同じ VPC 内に作成する必要があります。

7. (サブネット)、[Availability Zones and subnets] (アベイラビリティーゾーンとサブネット) で、指定した VPC 内のサブネットを 1 つ以上選択します。複数のアベイラビリティーゾーンのサブネットを使用することで、高可用性を得られます。詳細については、「VPC サブネットを選択する場合の考慮事項」を参照してください。

8. 指定したインスタンスタイプを使用して起動テンプレートを作成した場合は、次の手順に進み、起動テンプレートのインスタンスタイプを使用する Auto Scaling グループを作成できます。

   または、起動テンプレートでインスタンスタイプが指定されていない場合や、Auto Scaling に複数のインスタンスタイプを使用する場合は、[Override launch template] (起動テンプレートを上書きする) オプションを選択できます。詳細については、「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。

9. [Next] (次へ) を選択して、次のステップに進みます。

   または、残りはデフォルトのままにして、[Skip to Review (確認をスキップ)] を選択できます。

10. (オプション) [詳細オプションの設定] ページで、次のオプションを設定し、[次へ] を選択します。

    a. 「追加設定」「モニタリング」で、 CloudWatch グループメトリクスの収集を有効にするかどうかを選択します。これらのメトリクスは、終了インスタンス数や保留中のインスタ

ンスの数など、潜在的な問題の指標となる測定値を提供します。詳細については、「Auto Scaling グループとインスタンスの CloudWatch メトリクスをモニタリングする」を参照してください。

b. デフォルトのインスタンスウォームアップを有効にする で、このオプションを選択し、アプリケーションのウォームアップ時間を選択します。スケーリングポリシーを持つ Auto Scaling グループを作成する場合、デフォルトのインスタンスウォームアップ機能により、動的スケーリングに使用される Amazon CloudWatch メトリクスが改善されます。詳細については、「Auto Scaling グループに対するインスタンスのデフォルトウォームアップを設定する」を参照してください。

11. (オプション) [Configure group size and scaling policies (グループサイズとスケーリングポリシーの設定)] ページで、次のオプションを設定し、[次へ] を選択します。

a. グループサイズ で、希望する容量 に、起動するインスタンスの初期数を入力します。

b. Scaling セクションの Scaling 制限 で、希望する容量の新しい値が希望する最小容量と希望する最大容量 より大きい場合、希望する最大容量は自動的に希望する新しい容量値に増加します。これらの制限は必要に応じて変更できます。詳細については、「Auto Scaling グループのスケーリング制限を設定する」を参照してください。

c. 自動スケーリング では、ターゲット追跡スケーリングポリシーを作成するかどうかを選択します。このポリシーは、Auto Scaling グループの作成後に作成することもできます。

ターゲット追跡スケーリングポリシー を選択した場合は、「」の指示に従ってポリシーターゲット追跡スケーリングポリシーを作成するを作成します。

d. インスタンスメンテナンスポリシー で、インスタンスメンテナンスポリシーを作成するかどうかを選択します。このポリシーは、Auto Scaling グループの作成後に作成することもできます。インスタンスのメンテナンスポリシーを設定する 「」の指示に従ってポリシーを作成します。

e. [Instance scale-in protection (インスタンスのスケールイン保護)] で、インスタンスのスケールイン保護を有効にするかどうかを選択します。詳細については、「インスタンスのスケールイン保護を使用する」を参照してください。

12. (オプション) 通知を受け取るには、[通知の追加] を選択し、通知を設定してから [次へ] を選択します。詳細については、「Amazon EC2 Auto Scaling の Amazon SNS 通知オプション Amazon EC2 Auto Scaling」を参照してください。

13. (オプション) タグを追加するには、[タグの追加] を選択し、各タグのタグキーと値を指定し、[次へ] を選択します。詳細については、「Auto Scaling グループとインスタンスにタグを付ける」を参照してください。

14. [Review (レビュー)]ページで、[Create Auto Scaling group (Auto Scaling グループを作成)] を選択します。

コマンドラインを使用して Auto Scaling グループを作成するには

以下のコマンドのいずれかを使用できます。

- create-auto-scaling-group (AWS CLI)
- 新しい ASAutoScalingGroup (AWS Tools for Windows PowerShell)

Amazon EC2 起動ウィザードを使用して Auto Scaling グループを作成する

次の手順では、Amazon EC2 コンソールで [Launch instance] (インスタンスの作成) ウィザードを使用して Auto Scaling グループを作成する方法を示します。このオプションでは、特定の設定が詳細に入力されている起動テンプレートが、[Launch instance] (インスタンスの作成) ウィザードに自動的に入力されます。

> **Note**
>
> ウィザードは、指定したインスタンス数を Auto Scaling グループに設定しません。起動テンプレートには、Amazon マシンイメージ (AMI) の ID と、インスタンスタイプのみが設定されます。[Create Auto Scaling group] (Auto Scaling グループの作成) ウィザードで、起動するインスタンスの数を指定します。
> AMI には、インスタンスの設定に必要な情報が含まれています。同じ設定で複数のインスタンスが必要な場合は、1 つの AMI から複数のインスタンスを起動できます。Auto Scaling グループに属するインスタンスを再起動した場合に、インスタンスが終了しないように、既にアプリケーションがインストールされているカスタム AMI を使用することをお勧めします。Amazon EC2 Auto Scaling でカスタム AMI を使用するには、まずカスタマイズされたインスタンスから AMI を作成し、次に AMI を使用して、Auto Scaling グループの起動テンプレートを作成する必要があります。

前提条件

- Auto Scaling グループを作成する AWS リージョン のと同じ にカスタム AMI を作成しておく必要があります。詳細については、「Amazon EC2 ユーザーガイド」の「AMI の作成」を参照してください。 Amazon EC2

カスタム AMI をテンプレートとして使用する

このセクションでは、Amazon EC2 起動ウィザードを使用して、起動テンプレートにカスタム AMI を自動的に入力します。また、起動テンプレートを最初から設定する場合や、起動テンプレートに設定できるパラメータの詳細については、起動テンプレートを作成する (コンソール) を参照してください。

カスタム AMI をテンプレートとして使用するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 画面上部のナビゲーションバーに、現在の AWS リージョン が表示されます。Auto Scaling グループを起動するリージョンを選択します。
3. ナビゲーションペインで、[インスタンス] を選択します。
4. [Launch instance] (インスタンスを起動) を選択してから、以下を実行します。

   a. [Name and tags] (名前とタグ) では、[Name] (名前) を空のままにしておきます。名前は、起動テンプレートの作成に使用されるデータの 1 部ではありません。

   b. [Application and OS Images (Amazon Machine Image)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で、[Browse more AMIs] (その他の AMI を閲覧する) を選択して完全な AMI カタログを参照します。

   c. [My AMIs] (自分の AMI) を選択して、作成した AMI を見つけてから、[Select] (選択) を選択します。

   d. [Instance type] (インスタンスタイプ) でインスタンスタイプを選択します。

   > **Note**
   >
   > AMI の作成時に使用したものと同じインスタンスタイプを選択するか、より強力なインスタンスタイプを選択します。

   e. 画面の右側にある [Summary] (概要) で、[Number of instances] (インスタンス数) に任意の数を入力します。ここに入力する数値は重要ではありません。起動するインスタンスの数は、Auto Scaling グループの作成時に指定します。

   [Number of instances] (インスタンス数) フィールドの下に、[When launching more than 1 instance, consider EC2 Auto Scaling] (複数のインスタンスを起動する場合は、EC2 Auto Scaling を検討してください) というメッセージが表示されます。

f. ハイパーリンクテキストの [consider EC2 Auto Scaling] (EC2 Auto Scaling を検討してください) を選択します。

g. [Launch into Auto Scaling Group] (Auto Scaling グループに作成する) 確認ダイアログで [Continue] (次へ) を選択して、インスタンス起動ウィザードで選択した AMI とインスタンスタイプが既に入力されている [Create launch template] (起動テンプレートを作成) ページに移動します。

[Continue] (続行) を選択すると、[Create launch template] (起動テンプレートの作成) ページが開きます。この手順に従って、起動テンプレートの作成を完了します。

起動テンプレートを作成するには

1. [Launch template name and description] (起動テンプレート名と説明) で、新しい起動テンプレートの名前と説明を入力します。
2. (オプション) [Key pair (login)] (キーペア (ログイン)) と [Key pair name] (キーペア名) では、SSH などを使用したインスタンスへの接続時での使用のために以前作成したキーペアの名前を選択します。
3. (オプション) [Network settings] (セットワーク設定) の [Security groups] (セキュリティグループ) では、以前作成したセキュリティグループを 1 つ、または複数選択します。
4. (オプション) [Configure storage] (ストレージを設定) で、ストレージ設定を更新します。デフォルトのストレージ設定は AMI およびインスタンスタイプによって決まります。
5. 起動テンプレートの設定が終わったら、[Create launch template] (起動テンプレートの作成) を選択します。
6. 確認ページで、[Auto Scaling グループの作成] を選択します。

Auto Scaling グループを作成する

> **Note**
>
> このトピックの残りの部分では、Auto Scaling グループ作成の基本的な手順について説明します。Auto Scaling グループに設定できるパラメータの詳細については、「起動テンプレートを使用して Auto Scaling グループを作成する」を参照してください。

[Create Auto Scaling group] (Auto Scaling グループの作成) を選択すると、[Create Auto Scaling group] (Auto Scaling グループの作成) ウィザードが開きます。Auto Scaling グループを作成するには、次の手順を実行します。

Auto Scaling グループを作成する

1. [Choose launch template or configuration (起動テンプレートまたは設定の選択)] ページで、Auto Scaling グループの名前を入力します。
2. 作成した起動テンプレートは、既に選択されています。

   [起動テンプレートのバージョン] で、スケールアウト時に Auto Scaling グループで使用する起動テンプレートのバージョン (デフォルト、最新、または特定のバージョン) を選択します。
3. [Next] (次へ) を選択して、次のステップに進みます。
4. [インスタンス起動オプションを選択] ページで、複数のインスタンスタイプを使用していない場合は [インスタンスタイプの要件] セクションをスキップして、起動テンプレートで指定されている EC2 インスタンスタイプを使用できます。

   複数のインスタンスタイプを使用する場合は、「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。
5. [Network] (ネットワーク) の下にある [VPC] で、VPC を選択します。Auto Scaling グループは、起動テンプレートで指定したセキュリティグループと、同じ VPC 内に作成する必要があります。

   > **Tip**
   >
   > 起動テンプレートでセキュリティグループを指定しなかった場合、指定した VPC のデフォルトのセキュリティグループを使用して、インスタンスが起動されます。デフォルトでは、このセキュリティグループは外部ネットワークからのインバウンドトラフィックを許可しません。
6. (サブネット)、[Availability Zones and subnets] (アベイラビリティーゾーンとサブネット) で、指定した VPC 内のサブネットを 1 つ以上選択します。
7. [Next] (次へ) を2 回選択すると、[Configure group size and scaling policies] (グループサイズとスケーリングポリシーの設定) ページへ移動します。
8. グループサイズ で、希望する容量 (Auto Scaling グループの作成直後に起動するインスタンスの初期数) を定義します。

9. スケーリングセクションのスケーリング制限 で、希望する容量の新しい値が希望する最小容量と希望する最大容量 より大きい場合、希望する最大容量は自動的に希望する新しい容量値に増加します。これらの制限は必要に応じて変更できます。詳細については、「Auto Scaling グループのスケーリング制限を設定する」を参照してください。
10. [Skip to review] を選択します。
11. [Review (レビュー)]ページで、[Create Auto Scaling group (Auto Scaling グループを作成)] を選択します。

次のステップ

アクティビティ履歴を表示することによって、Auto Scaling グループが正しく作成されたことを確認できます。[Activity] (アクティビティ) タブの [Activity history] (アクティビティ履歴) では、[Status] (ステータス) 列に、Auto Scaling グループがインスタンスを正常に起動したかどうかが表示されます。インスタンスの起動に失敗したり、起動してもすぐに終了したりする場合は、次のトピックで、考えられる原因と解決方法を参照してください。

- Amazon EC2 Auto Scaling をトラブルシューティングする: EC2 インスタンス起動の失敗
- Amazon EC2 Auto Scaling をトラブルシューティングする: AMI 問題
- Amazon EC2 Auto Scaling の異常なインスタンスのトラブルシューティング

必要に応じて、Auto Scaling グループと同じリージョンに、ロードバランサーをアタッチできるようになりました。詳細については、「Elastic Load Balancing を使用して Auto Scaling グループ内のインスタンス全体にトラフィックを分散させる」を参照してください。

複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ

1 つの Auto Scaling グループ内で、オンデマンドインスタンスとスポットインスタンスのフリートを起動してオートスケールできます。スポットインスタンスの使用で割引を受けるだけでなく、リザーブドインスタンスまたは Savings Plans を使用して、通常のオンデマンドインスタンスの料金に対する割引の適用を受けることができます。これらの要素は、EC2 インスタンスのコスト削減を最適化し、アプリケーションに必要なスケールとパフォーマンスを得るのに役立ちます。

スポットインスタンスは、EC2 オンデマンド料金と比較して大幅な割引で利用できる予備の容量です。スポットインスタンス は、アプリケーションを実行する時間に柔軟性がある場合や、アプリケーションを中断できる場合に、費用効率の高い選択肢です。耐障害性と柔軟性に優れたさまざま

なアプリケーションに使用できます。例としては、ステートレスウェブサーバー、API エンドポイント、ビッグデータおよび分析アプリケーション、コンテナ化されたワークロード、CI/CD パイプライン、ハイパフォーマンスおよびハイスループットコンピューティング (HPC/HTC)、レンダリングワークロード、その他の柔軟なワークロードなどがあります。

詳細については、「Amazon EC2 ユーザーガイド」の「インスタンス購入オプション」を参照してください。 Amazon EC2

トピック

- 設定の概要
- 配分戦略
- 属性ベースのインスタンスタイプの選択を使用して混合インスタンスグループを作成する
- インスタンスタイプを手動で選択して混合インスタンスグループを作成する
- インスタンスの重みを使用するように Auto Scaling グループを設定する
- インスタンスタイプに異なる起動テンプレートを使用する

設定の概要

このトピックでは、混合インスタンスグループを作成するための概要とベストプラクティスについて説明します。

内容

- 概要
- インスタンスタイプの柔軟性
- アベイラビリティーゾーンの柔軟性
- 最大スポット料金
- プロアクティブなキャパシティの再調整
- スケーリングの動作
- リージョン別のインスタンスタイプの可用性
- 関連リソース
- 制限事項

概要

混合インスタンスグループを作成するには、次の 2 つのオプションがあります。

- 属性ベースのインスタンスタイプの選択 – コンピューティング要件を定義して、特定のインスタンス属性に基づいてインスタンスタイプを自動的に選択します。
- 手動インスタンスタイプの選択 — ワークロードに適したインスタンスタイプを手動で選択します。

Manual selection

次のステップでは、インスタンスタイプを手動で選択して混合インスタンスグループを作成する方法を説明します。

1. EC2 インスタンスを起動するためのパラメータを含む起動テンプレートを選択します。起動テンプレートのパラメータはオプションですが、Amazon マシンイメージ (AMI) ID が起動テンプレートにない場合、Amazon EC2 Auto Scaling はインスタンスを起動できません。
2. 起動テンプレートをオーバーライドするオプションを選択します。
3. ワークロードに合ったインスタンスタイプを手動で選択します。
4. 起動するオンデマンドインスタンスとスポットインスタンスの割合を指定します。
5. Amazon EC2 Auto Scaling が可能なインスタンスタイプからオンデマンドとスポットのキャパシティーを満たす方法を決定する割り当て戦略を選択します。
6. インスタンスを起動するアベイラビリティーゾーンと VPC サブネットを選択します。
7. グループの初期サイズ (希望するキャパシティ) と、グループの最小サイズおよび最大サイズを指定します。

オーバーライドは、起動テンプレートで宣言されたインスタンスタイプをオーバーライドし、Auto Scaling グループ独自のリソース定義に埋め込まれている複数のインスタンスタイプを使用するために必要です。使用可能なインスタンスタイプの詳細については、「Amazon EC2 ユーザーガイド」の「インスタンスタイプ」を参照してください。 Amazon EC2

インスタンスタイプごとに次のオプションパラメータを設定することもできます。

- `LaunchTemplateSpecification` – 必要に応じて、インスタンスタイプに別の起動テンプレートを割り当てることができます。このオプションは現在、コンソールからは利用できません。詳細については、「インスタンスタイプに異なる起動テンプレートを使用する」を参照してください。
- `WeightedCapacity` – グループ内の残りのインスタンスと比較して、インスタンスが希望する容量にどれだけカウントされるかを決定します。1 つのインスタンスタイプに `WeightedCapacity` の値を指定する場合は、すべてのインスタンスタイプに

`WeightedCapacity` の値を指定する必要があります。デフォルトでは、各インスタンスは希望するキャパシティに対して 1 個としてカウントされます。詳細については、「インスタンスの重みを使用するように Auto Scaling グループを設定する」を参照してください。

Attribute-based selection

Amazon EC2 Auto Scaling が特定のインスタンス属性に基づいてインスタンスタイプを自動的に選択できるようにするには、次のステップに従って、コンピューティング要件を指定することによって混合インスタンスグループを作成します。

1. EC2 インスタンスを起動するためのパラメータを含む起動テンプレートを選択します。起動テンプレートのパラメータはオプションですが、Amazon マシンイメージ (AMI) ID が起動テンプレートにない場合、Amazon EC2 Auto Scaling はインスタンスを起動できません。
2. 起動テンプレートをオーバーライドするオプションを選択します。
3. vCPU およびメモリ要件などのコンピューティング要件に一致するインスタンス属性を指定します。
4. 起動するオンデマンドインスタンスとスポットインスタンスの割合を指定します。
5. Amazon EC2 Auto Scaling が可能なインスタンスタイプからオンデマンドとスポットのキャパシティーを満たす方法を決定する割り当て戦略を選択します。
6. インスタンスを起動するアベイラビリティーゾーンと VPC サブネットを選択します。
7. グループの初期サイズ (希望するキャパシティ) と、グループの最小サイズおよび最大サイズを指定します。

オーバーライドは、起動テンプレートで宣言されたインスタンスタイプをオーバーライドし、コンピューティング要件を記述するインスタンス属性のセットを使用するために必要です。サポートされている属性については、InstanceRequirementsAmazon EC2 Auto Scaling API リファレンス」の「」を参照してください。あるいは、既にインスタンス属性が定義されている起動テンプレートを使用することもできます。

必要に応じて、オーバーライド構造内で `LaunchTemplateSpecification` パラメータを設定して、一連のインスタンス要件に別の起動テンプレートを割り当てることもできます。このオプションは現在、コンソールからは利用できません。詳細については、LaunchTemplate「Amazon EC2 Auto Scaling API リファレンスAmazon EC2 Auto Scaling」の「オーバーライド」を参照してください。

デフォルトでは、Auto Scaling グループの希望するキャパシティとしてインスタンスの数を設定します。

あるいは、希望するキャパシティの値を vCPU の数またはメモリ量に設定することもできます。これを実行するには、`CreateAutoScalingGroup` API オペレーションの `DesiredCapacityType` プロパティを使用するか、または AWS Management Consoleの [希望する容量タイプ] ドロップダウンフィールドを使用します。これは、インスタンスの重みに代わる便利な方法です。

インスタンスタイプの柔軟性

可用性を高めるには、複数のインスタンスタイプにアプリケーションをデプロイします。キャパシティ要件を満たすために複数のインスタンスタイプを使用するのがベストプラクティスです。そうすることで、選択したアベイラビリティーゾーンに十分なインスタンスのキャパシティがない場合、Amazon EC2 Auto Scaling は別のインスタンスタイプを起動できます。

スポットインスタンスで十分なインスタンスのキャパシティがない場合、Amazon EC2 Auto Scaling は他のスポットインスタンスプールからの起動を試みようとし続けます。(使用するプールは、インスタンスタイプと配分戦略の選択によって決まります)。Amazon EC2 Auto Scaling では、オンデマンドインスタンスの代わりにスポットインスタンスを起動することで、スポットインスタンスのコスト削減効果を活用できます。

ワークロードごとに少なくとも 10 個のインスタンスを使用して柔軟性を持たせることをお勧めします。インスタンスタイプを選択する際には、最も人気のある新しいインスタンスタイプに限定しないでください。旧世代のインスタンスタイプを選択すると、オンデマンドの顧客からの需要が少ないため、スポット中断が少なくなる傾向があります。

アベイラビリティーゾーンの柔軟性

Auto Scaling グループが複数のアベイラビリティーゾーンにまたがるようにすることを強くお勧めします。複数のアベイラビリティーゾーンを使用すると、ゾーン間で自動的にフェイルオーバーして回復力を高めるアプリケーションを設計できます。

追加のメリットとして、単一のアベイラビリティーゾーン内のグループと比較して、より深い Amazon EC2 キャパシティプールにアクセスできます。キャパシティは各アベイラビリティーゾーンのインスタンスタイプごとに個別に変動するため、多くの場合、より多くのコンピューティングキャパシティを得ることができ、インスタンスタイプとアベイラビリティーゾーンの両方を柔軟に選択できます。

複数のアベイラビリティーゾーンの使用の詳細については、「例: 複数のアベイラビリティーゾーン全体にインスタンスを分散させる」を参照してください。

最大スポット料金

AWS CLI または SDK を使用して Auto Scaling グループを作成する場合、 SpotMaxPriceパラメータを指定できます。SpotMaxPrice パラメータは、お客様がスポットインスタンス時間について支払ってもよいと考える最大料金を決定します。

オーバーライド (またはグループレベルで "DesiredCapacityType": "vcpu" もしくは "DesiredCapacityType": "memory-mib") で WeightedCapacity パラメータを指定すると、最大料金はインスタンス全体の最大料金ではなく、最大単価を表します。

最大料金を指定しないことを強くお勧めします。低すぎる上限価格が設定されるなどの理由で、スポットインスタンスを取得できない場合には、アプリケーションが実行されないことがあります。上限料金を指定しない場合、デフォルトの上限料金はオンデマンド価格となります。起動するスポットインスタンスのスポット価格のみを支払います。スポットインスタンスによって提供される大幅な割引は、そのまま受けることができます。これらの割引を実現できるのは、スポット料金モデルを使用して適用されるスポット料金が安定しているためです。詳細については、「Amazon EC2 ユーザーガイド」の「料金と割引」を参照してください。 Amazon EC2

プロアクティブなキャパシティの再調整

ユースケースで可能な場合は、キャパシティの再調整をお勧めします。容量の再調整は、実行中のスポットインスタンスが 2 分間のスポットインスタンス中断通知を受け取る前に、新しいスポットインスタンスでフリートを事前に拡張することにより、ワークロードの可用性を維持するのに役立ちます。

キャパシティの再調整が有効になっている場合、Amazon EC2 Auto Scaling は、再調整に関するレコメンデーションを受け取ったスポットインスタンスをプロアクティブに置き換えようとします。これは、中断のリスクが低い新しいスポットインスタンスにワークロードを再調整する機会を提供します。

詳細については、「キャパシティの再調整を使用して Amazon EC2 スポットの中断に対処する」を参照してください。

スケーリングの動作

混合インスタンスグループを作成すると、デフォルトでオンデマンドインスタンスが使用されます。スポットインスタンスを使用するには、オンデマンドインスタンスとして起動されるグループの割合

を変更する必要があります。オンデマンドインスタンスの割合には、0 から 100 までの任意の数を指定できます。

オプションで、開始時のオンデマンドインスタンスのベース数を指定することもできます。その場合、Amazon EC2 Auto Scaling は、グループがスケールアウトする際にオンデマンドインスタンスのベースキャパシティが起動された後までは、スポットインスタンスの起動を待機します。ベースキャパシティーを超える場合、オンデマンドインスタンスの割合を使用して、起動するオンデマンドインスタンスとスポットインスタンスの数が決まります。

Amazon EC2 Auto Scaling は、その割合を同等のインスタンス数に変換します。結果が小数になる場合、オンデマンドインスタンスを優先して次の整数に切り上げられます。

次の表は、サイズが増減した場合における Auto Scaling グループの動作について示しています。

例: スケーリング動作

購入オプション	各購入オプションのグループのサイズと実行中のインスタンスの数			
	10	20	30	40
例 1: 10 を基準、50/50% オンデマンド/スポット				
オンデマンドインスタンス (ベース量)	10	10	10	10
オンデマンドインスタンス	0	5	10	15
スポットインスタンス	0	5	10	15
例 2: 0 を基準、0/100% オンデマンド/スポット				

購入オプション	各購入オプションのグループのサイズと実行中のインスタンスの数			
オンデマンドインスタンス (ベース量)	0	0	0	0
オンデマンドインスタンス	0	0	0	0
スポットインスタンス	10	20	30	40
例 3: 0 を基準、60/40% オンデマンド/スポット				
オンデマンドインスタンス (ベース量)	0	0	0	0
オンデマンドインスタンス	6	12	18	24
スポットインスタンス	4	8	12	16
例 4: 0 を基準、100/0% オンデマンド/スポット				
オンデマンドインスタンス (ベース量)	0	0	0	0
オンデマンドインスタンス	10	20	30	40

購入オプション	各購入オプションのグループのサイズと実行中のインスタンスの数			
スポットインスタンス	0	0	0	0
例 5: 12 を基準、0/100% オンデマンド/スポット				
オンデマンドインスタンス (ベース量)	10	12	12	12
オンデマンドインスタンス	0	0	0	0
スポットインスタンス	0	8	18	28

グループのサイズが大きくなると、Amazon EC2 Auto Scaling は、指定されたアベイラビリティーゾーン全体でキャパシティのバランスを均等にすることを試みます。次に、指定された配分戦略に従ってインスタンスタイプを起動します。

グループのサイズが小さくなると、Amazon EC2 Auto Scaling は、まず 2 つのタイプ (スポットまたはオンデマンド) のどちらを終了する必要があるかを特定します。その後、指定されたアベイラビリティーゾーン全体でバランスの取れた方法でインスタンスを終了することを試みます。また、優先的に、割り当て戦略に近い方法でインスタンスを終了します。終了ポリシーの詳細については、「Amazon EC2 Auto Scaling の終了ポリシーを設定する」を参照してください。

リージョン別のインスタンスタイプの可用性

EC2 インスタンスタイプの可用性は、 によって異なります AWS リージョン。例えば、最新世代のインスタンスタイプは、特定のリージョンではまだ利用できない可能性があります。リージョンによってインスタンスの可用性が異なるため、オーバーライドする複数のインスタンスタイプをリージョンで利用できない場合、プログラムでリクエストを行う際に問題が発生する可能性があります。リージョンで利用できない複数のインスタンスタイプを使用すると、リクエストが完全に失敗する可能性があります。この問題を解決するには、異なるインスタンスタイプでリクエストを再試行し、各インスタンスタイプがリージョンで利用可能であることを確認してください。各ロケーションで

利用可能なインスタンスタイプを検索するには、describe-instance-type-offerings コマンドを使用します。詳細については、Amazon EC2 ユーザーガイド」の「Amazon EC2 インスタンスタイプの検索」を参照してください。 Amazon EC2

関連リソース

スポットインスタンスのベストプラクティスの詳細については、「Amazon EC2 ユーザーガイド」の「EC2 スポットのベストプラクティス」を参照してください。 Amazon EC2

制限事項

混合インスタンスポリシー を使用して Auto Scaling グループにオーバーライドを追加した後、`UpdateAutoScalingGroup`API コールでオーバーライドを更新できますが、削除することはできません。 https://docs.aws.amazon.com/autoscaling/ec2/APIReference/API_MixedInstancesPolicy.htmlオーバーライドを完全に削除するには、まず Auto Scaling グループを切り替えて、混合インスタンスポリシーではなく起動テンプレートまたは起動設定を使用する必要があります。その後、オーバーライドなしで混合インスタンスポリシーを再度追加できます。

配分戦略

複数のインスタンスタイプを使用する場合、Amazon EC2 Auto Scaling が可能なインスタンスタイプからオンデマンドおよびスポットキャパシティを満たす方法を管理します。これを実行するには、割り当て戦略を指定します。

混合インスタンスグループのベストプラクティスを確認するには、「」を参照してください設定の概要。

内容

- スポットインスタンス
- オンデマンドインスタンス
- 配分戦略と重みの連携方法

スポットインスタンス

Amazon EC2 Auto Scaling では、スポットインスタンス用に、次の配分戦略を提供しています。

`price-capacity-optimized` (推奨)

価格とキャパシティで最適化された配分戦略では、価格とキャパシティの両方を考慮して、中断される可能性が最も低く、価格ができるだけ低いスポットインスタンスプールを選択します。

使用を開始する際には、この戦略をお勧めします。詳細については、 AWS ブログのEC2 スポットインスタンスの price-capacity-optimized 配分戦略の紹介」を参照してください。

`capacity-optimized`

Amazon EC2 Auto Scaling は、起動するインスタンスの数に最適なキャパシティを持つプールからスポットインスタンスをリクエストします。

スポットインスタンスでは、料金は需要と供給の長期的な傾向に基づいて時間の経過とともに緩やかに変動します。ただし、キャパシティはリアルタイムで変動します。capacity-optimized戦略では、リアルタイムのキャパシティーデータを調べ、可用性の最も高いプールを予測することで、そのプールから スポットインスタンス を自動的に起動します。これにより、作業の再開とチェックポイントに伴う中断コストが高くなる可能性があるワークロードの中断を最小限に抑えることができます。最初に特定のインスタンスタイプを起動する可能性を高めるには、`capacity-optimized-prioritized` を使用します。

`capacity-optimized-prioritized`

起動テンプレートのオーバーライドのリスト内のインスタンスタイプの順序を、優先順位の高いものから順番に (リストの最初から最後まで) 設定することもできます。Amazon EC2 Auto Scaling は、ベストエフォートベースでインスタンスタイプの優先順位を尊重しますが、まずはキャパシティーに合わせて最適化します。これは、中断の可能性を最小限に抑える必要があるワークロードに適したオプションですが、適切なインスタンスタイプを設定することも重要です。オンデマンド配分戦略を `prioritized` に設定した場合、オンデマンドキャパシティを満たす際にも同じ優先順位が適用されます。

`lowest-price`

Amazon EC2 Auto Scaling は、[最低料金のプール] の設定で指定したスポットプールの N 個のプール全体で、アベイラビリティーゾーン内の最低料金のプールを使用してスポットインスタンスをリクエストします。例えば、4 つのインスタンスタイプと 4 つのアベイラビリティーゾーンを指定した場合、Auto Scaling グループは最大 16 個のスポットプールにアクセスできます。(各アベイラビリティーゾーンに 4 個ずつ)。配分戦略で 2 つのスポットプール (N=2) を指定した場合、Auto Scaling グループは、アベイラビリティゾーンあたり 2 つの最低価格のプールを引き出し、スポットキャパシティを満たすことができます。

この戦略では、インスタンスの価格のみが考慮され、キャパシティの可用性は考慮されないため、中断率が高くなる可能性があります。

Amazon EC2 Auto Scaling は、指定された N 個のプールからスポットインスタンスを取得しようとします。ただし、希望するキャパシティを満たす前にプールにスポットキャパシティの残量がなくなった場合、Amazon EC2 Auto Scaling は次に料金の低いプールのキャパシティを利用して

引き続きリクエストを満たします。希望するキャパシティを満たすために、指定した N 個を超えるプールからスポットインスタンスが割り当てられることがあります。同様に、ほとんどのプールにスポットキャパシティーがない場合には、指定した N 個より少ないプールから希望するキャパシティのすべてが割り当てられることがあります。

> **Note**
>
> AMD SEV-SNP を有効にして 起動するようにスポットインスタンスを設定すると、選択したインスタンスタイプのオンデマンド時間料金の 10% に相当する追加の時間単位使用料が請求されます。配分戦略で料金を入力として使用する場合、Amazon EC2 Auto Scaling にはこの追加料金は含まれず、スポット料金のみが使用されます。

オンデマンドインスタンス

Amazon EC2 Auto Scaling では、オンデマンドインスタンスに使用できる、以下の配分戦略を提供しています。

`lowest-price`

Amazon EC2 Auto Scaling は、現在のオンデマンド価格に基づき、各アベイラビリティーゾーンで最も最低価格のインスタンスタイプを自動的にデプロイします。

希望するキャパシティを満たすために、各アベイラビリティーゾーンで複数のタイプのオンデマンドインスタンスを受け取る場合があります。これは、リクエストするキャパシティの大きさによって異なります。

`prioritized`

オンデマンドキャパシティを満たすとき、Amazon EC2 Auto Scaling は、起動テンプレートのオーバーライドのリスト内のインスタンスタイプの順序に基づき、最初に使用するインスタンスタイプを決定します。例えば、3 つの起動テンプレートの優先度を `c5.large`、`c4.large`、`c3.large` と指定するとします。オンデマンドインスタンスが起動すると、Auto Scaling グループは、`c5.large`、`c4.large`、`c3.large` の順にオンデマンドキャパシティを満たします。

オンデマンドインスタンスの優先順位を管理する場合は、次の点を考慮してください。

- Savings Plans またはリザーブドインスタンスを利用して使用料を前払いすることで、オンデマンドインスタンスの大幅な割引を受けることができます。詳細については、「Amazon EC2 料金」ページを参照してください。

- リザーブドインスタンスでは、Amazon EC2 Auto Scaling が一致するインスタンスタイプを起動すると、通常のオンデマンドインスタンス コストの割引料金が適用されます。したがって、c4.large の未使用のリザーブドインスタンスがある場合、インスタンスタイプの優先順位を設定して、リザーブドインスタンスの最も高い優先順位を c4.large インスタンスタイプに付与できます。c4.large インスタンスが起動すると、リザーブドインスタンスの料金が発生します。
- Savings Plans では、Amazon EC2 Instance Savings Plans または Compute Savings Plans を使用する場合、通常のオンデマンドインスタンス コストの割引料金が適用されます。Savings Plans では、インスタンスタイプの優先順位をより柔軟に設定することができます。Savings Plans の対象となるインスタンスタイプであれば、任意の順序で優先順位を設定できます。また、インスタンスタイプの全体の順序を変えることも可能で、その場合でも Savings Plans の割引料金が適用されます。Savings Plans の詳細については、Savings Plans ユーザーガイドを参照してください。

配分戦略と重みの連携方法

オーバーライド (または"DesiredCapacityType": "memory-mib"グループレベルで"DesiredCapacityType": "vcpu"または) で WeightedCapacityパラメータを指定すると、割り当て戦略は他の Auto Scaling グループとまったく同じように機能します。

唯一の違いは、 lowest-priceまたは price-capacity-optimized戦略を選択すると、各アベイラビリティーゾーンのユニットあたりの料金が最も低いインスタンスプールからインスタンスが取得されることです。詳細については、「インスタンスの重みを使用するように Auto Scaling グループを設定する」を参照してください。

例えば、vCPU の量が異なる複数のインスタンスタイプを含む Auto Scaling グループがあるとします。スポットとオンデマンドの配分戦略として lowest-price を使うとします。各インスタンスタイプの vCPU 数に基づいた重みの割り当てを選択する場合、Amazon EC2 Auto Scaling は、フルフィルメントの時点で割り当てられた重みの値 (vCPU など) あたりの料金が最も低いインスタンスタイプを起動します。スポットインスタンスの場合、これは vCPU あたりの最低スポット料金であることを意味します。オンデマンドインスタンスの場合、これは vCPU あたりの最低オンデマンド料金であることを意味します。

属性ベースのインスタンスタイプの選択を使用して混合インスタンスグループを作成する

混合インスタンスグループのインスタンスタイプを手動で選択する代わりに、コンピューティング要件を記述するインスタンス属性のセットを指定できます。Amazon EC2 Auto Scaling がインスタン

スを起動するとき、Auto Scaling グループで使用されるインスタンスタイプは、必要なインスタンス属性と一致している必要があります。これは属性ベースのインスタンスタイプの選択と呼ばれます。

このアプローチは、コンテナやビッグデータ、CI/CD など、柔軟にインスタンスタイプを使用するワークロードとフレームワークに最適です。

属性ベースのインスタンスタイプを選択すると、次の利点があります。

- スポットインスタンスの最適な柔軟性 – Amazon EC2 Auto Scaling は、スポットインスタンスを起動するための幅広いインスタンスタイプから選択できます。これは、インスタンスタイプに対して柔軟であるという、スポットのベストプラクティスを満たしています。これにより、Amazon EC2 スポットサービスが必要なコンピューティング性能を見つけ、割り当てられる機会に恵まれます。
- 適切なインスタンスタイプを簡単に使用 - 利用可能なインスタンスタイプの数が多いため、ワークロードに適したインスタンスタイプを見つけるには時間がかかることがあります。インスタンス属性を指定すると、インスタンスタイプにはワークロードに必要な属性が自動的に設定されます。
- 新しいインスタンスタイプの自動使用 – Auto Scaling グループは、リリース時に新しい世代のインスタンスタイプを使用できます。 Auto Scaling 要件に一致し、かつ Auto Scaling グループのために選択した割り当て戦略にマッチする場合には、新しい世代のインスタンスタイプが自動的に使用されます。

トピック

- 属性ベースのインスタンスタイプ選択の仕組み
- 料金保護
- 前提条件
- 属性ベースのインスタンスタイプ選択を使用して混合インスタンスグループを作成する (コンソール)
- 属性ベースのインスタンスタイプ選択を使用して混合インスタンスグループを作成する (AWS CLI)
- 設定例
- インスタンスタイプをプレビューする
- 関連リソース

属性ベースのインスタンスタイプ選択の仕組み

属性ベースのインスタンスタイプを選択すると、特定のインスタンスタイプのリストを提供する代わりに、インスタンスに必要なインスタンス属性のリストを指定します。例えば、次のようになります。

- vCPU 数 – インスタンスあたりの vCPU の最小数と最大数。
- メモリ — インスタンスあたりのメモリ GiBs の最小と最大。
- ローカルストレージ – EBS ボリュームとインスタンスストアボリュームのどちらをローカルストレージに使用するか。
- バースト可能なパフォーマンス – T4g、T3a、T3、および T2 タイプを含む T インスタンスファミリーを使用するかどうか。

インスタンスの要件を定義するには、多くのオプションを使用できます。各オプションの説明とデフォルト値については、InstanceRequirementsAmazon EC2 Auto Scaling API リファレンス」の「」を参照してください。

Auto Scaling グループは、インスタンスを起動する必要がある場合、指定した属性に一致し、そのアベイラビリティーゾーンで使用可能なインスタンスタイプを検索します。次に、割り当て戦略によって、どのインスタンスタイプを起動するかが決まります。デフォルトでは、属性ベースのインスタンスタイプの選択では、Auto Scaling グループが予算のしきい値を超えるインスタンスタイプを起動できないように料金保護機能が有効になっています。

デフォルトでは、Auto Scaling グループの希望する容量を設定するときに、測定単位としてインスタンス数を使用します。つまり、各インスタンスは 1 ユニットとしてカウントされます。

あるいは、希望するキャパシティの値を vCPU の数またはメモリ量に設定することもできます。これを行うには、 の「希望する容量タイプ」ドロップダウンフィールド、 AWS Management Console または `CreateAutoScalingGroup`または `UpdateAutoScalingGroup` API オペレーションの `DesiredCapacityType`プロパティを使用します。Amazon EC2 Auto Scaling は、必要な vCPU またはメモリ容量を満たすために必要なインスタンスの数を起動します。例えば、vCPUs を目的の容量タイプとして使用し、それぞれ 2 つの vCPUs を持つインスタンスを使用する場合、10 個の vCPUs の希望する容量は 5 つのインスタンスを起動します。これは、インスタンスの重みに代わる便利な方法です。

料金保護

料金保護を使用すると、Auto Scaling グループによって起動された EC2 インスタンスに対して支払う上限料金を指定できます。料金保護は、Auto Scaling グループが、指定した属性に適合した場合でも、高すぎると思われるインスタンスタイプを使用できないようにする機能です。

料金保護はデフォルトで有効になっており、オンデマンドインスタンスとスポットインスタンスの料金しきい値は異なります。Amazon EC2 Auto Scaling が新しいインスタンスを起動する必要がある場合、関連するしきい値を超える料金のインスタンスタイプは起動されません。

トピック

- オンデマンド料金保護
- スポット料金保護
- 料金保護をカスタマイズする

オンデマンド料金保護

オンデマンドインスタンスでは、指定したオンデマンド料金よりも高いパーセンテージで支払うことができる最大オンデマンド料金を定義します。特定されたオンデマンド料金は、指定した属性を持つ最低価格の現行世代の C、M、または R インスタンスタイプの料金です。

オンデマンド料金保護値が明示的に定義されていない場合は、特定されたオンデマンド料金よりも 20% 高いデフォルトの最大オンデマンド料金が使用されます。

スポット料金保護

デフォルトでは、Amazon EC2 Auto Scaling は最適なスポットインスタンス料金保護を自動的に適用し、幅広いインスタンスタイプから一貫して選択します。料金保護を手動で設定することもできます。ただし、Amazon EC2 Auto Scaling に任せることで、スポット容量が満たされる可能性を高めることができます。

料金保護は、次のいずれかのオプションを使用して手動で指定できます。料金保護を手動で設定する場合は、最初のオプションを使用することをお勧めします。

- 特定されたオンデマンド料金の割合 – 特定されたオンデマンド料金は、指定した属性を持つ最低価格の現行世代の C、M、または R インスタンスタイプの料金です。
- 識別されたスポット料金よりも高いパーセンテージ — 識別されたスポット料金は、指定した属性を持つ最低価格の現行世代の C、M、または R インスタンスタイプの料金です。スポット料金は

変動する可能性があるため、このオプションの使用はお勧めしません。したがって、料金保護のしきい値も変動する可能性があります。

料金保護をカスタマイズする

料金保護のしきい値は、Amazon EC2 Auto Scaling コンソール、または AWS CLI または SDKsを使用してカスタマイズできます。

- コンソールで、追加のインスタンス属性 のオンデマンド料金保護とスポット料金保護設定を使用します。
- InstanceRequirements 構造で、オンデマンドインスタンスの料金保護しきい値を指定するには、 `OnDemandMaxPricePercentageOverLowestPrice`プロパティを使用します。スポットインスタンスの料金保護のしきい値を指定するには、 `MaxSpotPriceAsPercentageOfOptimalOnDemandPrice`または`SpotMaxPricePercentageOverLowestPrice`プロパティを使用します。

希望する容量タイプ (`DesiredCapacityType`) を vCPUs または Memory GiB に設定すると、インスタンスあたりの料金ではなく、vCPU あたりの料金またはメモリあたりの料金に基づいて料金保護が適用されます。

料金保護をオフにすることもできます。料金保護のしきい値を指定しない場合は、999999 などの高いパーセンテージ値を指定します。

> **Note**
>
> 指定した属性に一致する現行世代の C、M、または R インスタンスタイプがない場合でも、料金保護は適用されます。一致するものが見つからなかった場合、識別される料金は、最低価格の現行世代のインスタンスタイプであるか、または属性に一致する最低価格の旧世代のインスタンスタイプで失敗します。

前提条件

- 起動テンプレートを作成する。詳細については、「Auto Scaling グループの起動テンプレートを作成する」を参照してください。
- 起動テンプレートがまだスポットインスタンスをリクエストしていないことを確認します。

属性ベースのインスタンスタイプ選択を使用して混合インスタンスグループを作成する (コンソール)

属性ベースのインスタンスタイプの選択を使用して混合インスタンスグループを作成するには、次の手順を実行します。ステップを効率的に進めるために、いくつかのオプションのセクションは省略されています。

ほとんどの汎用的なワークロードでは、必要な vCPU とメモリの数を指定すれば十分です。高度なユースケースでは、ストレージのタイプ、ネットワークインターフェイス、CPU の製造元、アクセラレーターのタイプなどの属性を指定できます。

混合インスタンスグループのベストプラクティスを確認するには、「」を参照してください設定の概要。

混合インスタンスグループを作成するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. 画面の上部のナビゲーションバーで、起動テンプレートを作成したときに使用したのと同じ AWS リージョン を選択します。
3. [Auto Scaling グループの作成] を選択します。
4. [起動テンプレートまたは起動設定を選択する] ページで [Auto Scaling グループ名] にAuto Scaling グループの名前を入力します。
5. 起動テンプレートを選択するには、以下の手順を実行します。
   a. [起動テンプレート] で、既存の起動テンプレートを選択します。
   b. [起動テンプレートのバージョン] で、スケールアウト時に Auto Scaling グループで使用する起動テンプレートのバージョン (デフォルト、最新、または特定のバージョン) を選択します。
   c. 起動テンプレートが、使用する予定のすべてのオプションをサポートしていることを確認し、[次へ] を選択します。
6. [インスタンス起動オプションを選択] ページで、次を実行します。
   a. [Instance type requirements] (インスタンスタイプの要件) で、[Override launch template] (起動テンプレートを上書きする) を選択します。

> **Note**
>
> vCPU やメモリなどのインスタンス属性のセットが既に含まれている起動テンプレートを選択した場合は、インスタンス属性が表示されます。これらの属性は Auto Scaling グループのプロパティに追加され、Amazon EC2 Auto Scaling コンソールからいつでも更新できます。

b. [Specify instance attributes] (インスタンスの属性を指定する) で、まず vCPU とメモリの要件を入力します。

- [vCPUs] に、希望する vCPU の最小数と最大数を入力します。制限なしを指定するには、[No minimum] (最小値なし)、[No maximum] (最大値なし)、または両方を選択します。
- [Memory (GiB)] (メモリ (GiB)) に、希望するメモリの最小値と最大値を入力します。制限なしを指定するには、[No minimum] (最小値なし)、[No maximum] (最大値なし)、または両方を選択します。

c. (オプション) [Additional instance attributes] (その他のインスタンス属性) では、オプションで 1 つ以上の属性を指定して、コンピューティング要件をより詳細に表現できます。追加の属性は、リクエストにさらに制約を追加します。

d. プレビューマッチングインスタンスタイプを展開して、指定した属性を持つインスタンスタイプを表示します。

e. [インスタンスの購入オプション] の [インスタンスの分散] で、オンデマンドインスタンスとスポットインスタンスとして起動するグループの割合をそれぞれ指定します。アプリケーションが、ステートレスでフォールトトレラントであり、中断されるインスタンスを扱える場合は、より高い割合のスポットインスタンスを指定できます。

f. (オプション) スポットインスタンスの割合を指定するときは、[オンデマンドベースキャパシティを含める] を選択してから、オンデマンドインスタンスによって満たされる必要がある Auto Scaling グループの最小初期キャパシティを指定します。ベースキャパシティーを超える場合は、[Instances distribution] (インスタンスの分散) 設定を使用して、起動するオンデマンドインスタンスとスポットインスタンスの数を決定します。

g. [Allocation strategies] (配分戦略) の[Lowest price] (最低価格) は、[On-Demand allocation strategy] (オンデマンドの配分戦略) によって自動的に選択され、変更できません。

h. [Spot allocation strategy] (スポット配分戦略) で、配分戦略を選択します。デフォルトでは、[Price capacity optimized] (価格のキャパシティの最適化) が選択されています。[Lowest price] (最低価格) はデフォルトでは非表示になっており、[Show all strategies] (すべての戦

略を表示) を選択した場合にのみ表示されます。[最低料金] を選択した場合は、[最低料金のプール] に、分散する最低料金のプールの数を入力します。

i. [容量の再分散] で、容量の再分散を有効にするか無効にするかを選択します。キャパシティの再調整を使用すると、スポットインスタンスがスポットの中断によって終了に近づいたときに自動的に応答します。詳細については、「キャパシティの再調整を使用して Amazon EC2 スポットの中断に対処する」を参照してください。

j. [Network] (ネットワーク) の下にある [VPC] で、VPC を選択します。Auto Scaling グループは、起動テンプレートで指定したセキュリティグループと、同じ VPC 内に作成する必要があります。

k. [Availability Zones and subnets] (アベイラビリティーゾーンとサブネット) で、指定した VPC 内のサブネットを 1 つ以上選択します。複数のアベイラビリティーゾーンのサブネットを使用することで、高可用性を得られます。詳細については、「VPC サブネットを選択する場合の考慮事項」を参照してください。

l. [次へ]、[次へ] を選択します。

7. [Configure group size and scaling policies] (グループサイズとスケーリングポリシーを設定する) ステップでは、以下の手順を実行します。

   a. インスタンス以外の単位で希望する容量を測定するには、グループサイズ 、希望する容量タイプ に適切なオプションを選択します。[Units] (ユニット)、[vCPUs]、および [Memory GiB] (メモリ GiB) がサポートされています。デフォルトで、Amazon EC2 Auto Scaling は [Units] (ユニット) を指定します。これはインスタンスの数になります。

   b. 希望する容量 の場合、Auto Scaling グループの初期サイズ。

   c. スケーリングセクションのスケーリング制限 で、希望する容量の新しい値が希望する最小容量と希望する最大容量 より大きい場合、希望する最大容量は自動的に新しい希望する容量値に増加します。これらの制限は必要に応じて変更できます。詳細については、「Auto Scaling グループのスケーリング制限を設定する」を参照してください。

8. [Skip to review] を選択します。

9. [Review (レビュー)]ページで、[Create Auto Scaling group (Auto Scaling グループを作成)] を選択します。

属性ベースのインスタンスタイプ選択を使用して混合インスタンスグループを作成する (AWS CLI)

コマンドラインを使用して混合インスタンスグループを作成するには

以下のいずれかのコマンドを使用します。

- create-auto-scaling-group (AWS CLI)
- 新しい AS AutoScalingグループ (AWS Tools for Windows PowerShell)

設定例

AWS CLIを使用して、属性ベースのインスタンスタイプを選択した Auto Scaling グループを作成するには、次の create-auto-scaling-group コマンドを使用します。

次のインスタンス属性が指定されています。

- VCpuCount — インスタンスタイプには、4 個以上、最大 8 個の vCPU が必要です。
- MemoryMiB – インスタンスタイプには最低 16,384 MiB のメモリが必要です。
- CpuManufacturers — インスタンスタイプには、インテル製の CPU が必要です。

JSON

```
aws autoscaling create-auto-scaling-group --cli-input-json file://~/config.json
```

次は、config.json ファイルの例です。

```
{
    "AutoScalingGroupName": "my-asg",
    "DesiredCapacityType": "units",
    "MixedInstancesPolicy": {
        "LaunchTemplate": {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "$Default"
            },
            "Overrides": [{
                "InstanceRequirements": {
                    "VCpuCount": {"Min": 4, "Max": 8},
                    "MemoryMiB": {"Min": 16384},
                    "CpuManufacturers": ["intel"]
                }
            }]
        },
        "InstancesDistribution": {
            "OnDemandPercentageAboveBaseCapacity": 50,
            "SpotAllocationStrategy": "price-capacity-optimized"
```

```
        }
    },
    "MinSize": 0,
    "MaxSize": 100,
    "DesiredCapacity": 4,
    "DesiredCapacityType": "units",
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
}
```

vCPU の数またはメモリーの総量として、希望するキャパシティー値を設定するには、ファイルで "DesiredCapacityType": "vcpu" または "DesiredCapacityType": "memory-mib" を指定します。希望するキャパシティータイプのデフォルトは units で、これはインスタンスの数を、希望するキャパシティー値として設定します。

YAML

または、次の create-auto-scaling-group コマンドを使用して、Auto Scaling グループを作成します。これは、Auto Scaling グループの唯一のパラメータとして YAML ファイルを参照します。

```
aws autoscaling create-auto-scaling-group --cli-input-yaml file://~/config.yaml
```

次は、config.yaml ファイルの例です。

```
---
AutoScalingGroupName: my-asg
DesiredCapacityType: units
MixedInstancesPolicy:
  LaunchTemplate:
    LaunchTemplateSpecification:
      LaunchTemplateName: my-launch-template
      Version: $Default
    Overrides:
    - InstanceRequirements:
        VCpuCount:
          Min: 2
          Max: 4
        MemoryMiB:
          Min: 2048
        CpuManufacturers:
        - intel
  InstancesDistribution:
    OnDemandPercentageAboveBaseCapacity: 50
```

```
    SpotAllocationStrategy: price-capacity-optimized
MinSize: 0
MaxSize: 100
DesiredCapacity: 4
DesiredCapacityType: units
VPCZoneIdentifier: subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782
```

vCPU の数またはメモリーの総量として、希望するキャパシティー値を設定するには、ファイルで DesiredCapacityType: vcpu または DesiredCapacityType: memory-mib を指定します。希望するキャパシティータイプのデフォルトは units で、これはインスタンスの数を、希望するキャパシティー値として設定します。

インスタンスタイプをプレビューする

インスタンスを起動することなくコンピューティング要件に一致するインスタンスタイプをプレビューでき、必要に応じて要件を調整できます。Amazon EC2 Auto Scaling コンソールで Auto Scaling グループを作成すると、[Choose instance launch options] (インスタンス起動オプションを選択) ページの [Preview matching instance types] (一致するインスタンスタイプのプレビュー) セクションに、インスタンスタイプのプレビューが表示されます。

または、 AWS CLI または SDK を使用して Amazon EC2 GetInstanceTypesFromInstanceRequirements API コールを行うことで、インスタンスタイプをプレビューすることもできます。Auto Scaling グループの作成または更新のリクエストの中で、正しい形式で InstanceRequirements パラメーターを渡します。詳細については、「Amazon EC2 ユーザーガイド」の「指定された属性を持つインスタンスタイプのプレビュー」を参照してください。Amazon EC2

関連リソース

属性ベースのインスタンスタイプの選択の詳細については、 AWS ブログのEC2 Auto Scaling と EC2 フリートの属性ベースのインスタンスタイプの選択」を参照してください。

AWS CloudFormationを使用して Auto Scaling グループを作成する際に、属性ベースのインスタンスタイプの選択を宣言できます。詳細については、「AWS CloudFormation ユーザーガイド」の「Auto Scaling テンプレートスニペット」セクションのサンプルスニペットを参照してください。

インスタンスタイプを手動で選択して混合インスタンスグループを作成する

このトピックでは、インスタンスタイプを手動で選択して、単一の Auto Scaling グループで複数のインスタンスタイプを起動する方法を示します。

インスタンスタイプを選択する基準としてインスタンス属性を使用する場合は、「属性ベースのインスタンスタイプの選択を使用して混合インスタンスグループを作成する」を参照してください。

内容

- 前提条件
- 混合インスタンスグループを作成する (コンソール)
- 混合インスタンスグループを作成する (AWS CLI)
- 設定例

前提条件

- 起動テンプレートを作成する。詳細については、「Auto Scaling グループの起動テンプレートを作成する」を参照してください。
- 起動テンプレートがまだスポットインスタンスをリクエストしていないことを確認します。

混合インスタンスグループを作成する (コンソール)

次の手順を実行して、グループが起動できるインスタンスタイプを手動で選択し、混合インスタンスグループを作成します。ステップを効率的に進めるために、いくつかのオプションのセクションは省略されています。

混合インスタンスグループのベストプラクティスを確認するには、「」を参照してください設定の概要。

混合インスタンスグループを作成するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. 画面の上部のナビゲーションバーで、起動テンプレートを作成したときに使用したのと同じ AWS リージョン を選択します。
3. [Auto Scaling グループの作成] を選択します。
4. [起動テンプレートまたは起動設定を選択する] ページで [Auto Scaling グループ名] にAuto Scaling グループの名前を入力します。
5. 起動テンプレートを選択するには、以下の手順を実行します。

    a. [起動テンプレート] で、既存の起動テンプレートを選択します。

b. [起動テンプレートのバージョン] で、スケールアウト時に Auto Scaling グループで使用する起動テンプレートのバージョン (デフォルト、最新、または特定のバージョン) を選択します。

c. 起動テンプレートが、使用する予定のすべてのオプションをサポートしていることを確認し、[Next] (次へ) を選択します。

6. [インスタンス起動オプションを選択] ページで、次を実行します。

a. [Instance type requirements] (インスタンスタイプの要件) で、[Override launch template] (起動テンプレートを上書きする) を選択してから、[Manually add instance types] (インスタンスタイプを手動で追加する) を選択します。

b. インスタンスタイプを選択します。まずはレコメンデーションを使用できます。デフォルトでは、[Family and generation flexible] (ファミリーと世代が柔軟) が選択されています。

- インスタンスタイプの順序を変更するには、矢印を使用します。優先順位付けをサポートする配分戦略を選択した場合、インスタンスタイプの順序によって起動の優先順位が設定されます。
- インスタンスタイプを削除するには、[X] を選択します。
- (オプション) [重み] 列のボックスで、各インスタンスタイプに相対的な重みを割り当てます。これを行うには、そのタイプのインスタンスがグループの希望するキャパシティにカウントされるユニット数を入力します。これが便利なのは、インスタンスタイプ別に異なる vCPU、メモリ、ストレージ、またはネットワーク帯域幅の機能を設定する場合です。詳細については、「インスタンスの重みを使用するように Auto Scaling グループを設定する」を参照してください。

  [サイズが柔軟] のレコメンデーションを使用することを選択した場合は、このセクションに含まれるすべてのインスタンスタイプに自動的に重みの値が設定されることに留意してください。重みを指定したくない場合は、すべてのインスタンスタイプについて [Weight] (重み) 列のボックスをクリアしてください。

c. [Instance purchase options] (インスタンスの購入オプション) の [Instances distribution] (インスタンスの分散) で、オンデマンドインスタンスとスポットインスタンスとして起動されるグループの割合をそれぞれ指定します。アプリケーションが、ステートレスでフォールトトレラントであり、中断されるインスタンスを扱える場合は、より高い割合のスポットインスタンスを指定できます。

d. (オプション) スポットインスタンスの割合を指定するときは、[オンデマンドベースキャパシティを含める] を選択してから、オンデマンドインスタンスによって満たされる必要がある Auto Scaling グループの最小初期キャパシティを指定します。ベースキャパシティーを

超える場合は、[Instances distribution] (インスタンスの分散) 設定を使用して、起動するオンデマンドインスタンスとスポットインスタンスの数を決定します。

e. [Allocation strategies] (配分戦略) の [On-Demand allocation strategy] (オンデマンドの配分戦略) で、配分戦略を選択します。インスタンスタイプを手動で選択すると、デフォルトで [Prioritized] (高い優先順位で設定済み) が選択されます。

f. [Spot allocation strategy] (スポット配分戦略) で、配分戦略を選択します。デフォルトでは、[Price capacity optimized] (価格のキャパシティの最適化) が選択されています。[Lowest price] (最低価格) はデフォルトでは非表示になっており、[Show all strategies] (すべての戦略を表示) を選択した場合にのみ表示されます。

- [最低料金] を選択した場合は、[最低料金のプール] に、分散する最低料金のプールの数を入力します。
- [キャパシティ最適化] を選択した場合は、必要に応じて [インスタンスタイプの優先順位を設定] ボックスをオンにして、インスタンスタイプのリスト順に基づいて最初に起動するインスタンスタイプを Amazon EC2 Auto Scaling が選択できるようにします。

g. [容量の再分散] で、容量の再分散を有効にするか無効にするかを選択します。キャパシティの再調整を使用すると、スポットインスタンスがスポットの中断によって終了に近づいたときに自動的に応答します。詳細については、「キャパシティの再調整を使用して Amazon EC2 スポットの中断に対処する」を参照してください。

h. [Network] (ネットワーク) の下にある [VPC] で、VPC を選択します。Auto Scaling グループは、起動テンプレートで指定したセキュリティグループと、同じ VPC 内に作成する必要があります。

i. [Availability Zones and subnets] (アベイラビリティーゾーンとサブネット) で、指定した VPC 内のサブネットを 1 つ以上選択します。複数のアベイラビリティーゾーンのサブネットを使用することで、高可用性を得られます。詳細については、「VPC サブネットを選択する場合の考慮事項」を参照してください。

j. [次へ]、[次へ] を選択します。

7. [Configure group size and scaling policies] (グループサイズとスケーリングポリシーを設定する) ステップでは、以下の手順を実行します。

a. グループサイズ で、希望する容量 に、起動するインスタンスの初期数を入力します。

デフォルトでは、希望する容量はインスタンスの数として表されます。インスタンスタイプに重みを割り当てた場合は、この値を vCPUs の数など、重みの割り当てに使用したのと同じ測定単位に変換する必要があります。

b. スケーリングセクションのスケーリング制限 で、希望する容量の新しい値が希望する最小容量と希望する最大容量 より大きい場合、希望する最大容量は自動的に希望する新しい容量値に引き上げられます。これらの制限は必要に応じて変更できます。詳細については、「Auto Scaling グループのスケーリング制限を設定する」を参照してください。

8. [Skip to review] を選択します。
9. [Review (レビュー)]ページで、[Create Auto Scaling group (Auto Scaling グループを作成)] を選択します。

混合インスタンスグループを作成する (AWS CLI)

コマンドラインを使用して混合インスタンスグループを作成するには

以下のいずれかのコマンドを使用します。

- create-auto-scaling-group (AWS CLI)
- 新しい AS AutoScalingグループ (AWS Tools for Windows PowerShell)

設定例

次の設定例は、さまざまなスポット割り当て戦略を使用して混合インスタンスグループを作成する方法を示しています。

> **Note**
>
> これらの例は、JSON または YAML でフォーマットされた設定ファイルの使用方法を示しています。 AWS CLI バージョン 1 を使用する場合は、JSON 形式の設定ファイルを指定する必要があります。 AWS CLI バージョン 2 を使用する場合は、YAML または JSON のいずれかでフォーマットされた設定ファイルを指定できます。

例

- 例 1: capacity-optimized 割り当て戦略を使用して スポットインスタンス を起動する
- 例 2: capacity-optimized-prioritized 割り当て戦略を使用して スポットインスタンス を起動する
- 例 3: 2つのプール間での lowest-price 配分戦略を使用してスポットインスタンスを起動する
- 例 4: 配分戦略を使用して スポットインスタンス を起動するprice-capacity-optimized

例 1: **capacity-optimized** 割り当て戦略を使用して スポットインスタンス を起動する

次の [create-auto-scaling-group] コマンドは、以下を指定する Auto Scaling グループを作成します。

- オンデマンドインスタンスとして起動するグループの割合 (0) と開始時のオンデマンドインスタンスのベース数 (1)
- 優先度に従って起動するインスタンスタイプ (c5.large、c5a.large、m5.large、m5a.large、c4.large、m4.large、c3.large、m3.large)
- インスタンスを起動するサブネット (subnet-5ea0c127、subnet-6194ea3b、subnet-c934b782)。それぞれが異なるアベイラビリティーゾーンに対応します。
- 起動テンプレート (my-launch-template) とそのバージョン ($Default)。

Amazon EC2 Auto Scaling がオンデマンドキャパシティーを満たそうとするとき、まず c5.large インスタンスタイプを起動します。スポットインスタンスは、スポットインスタンスのキャパシティーに基づいて、各アベイラビリティーゾーンの最適なスポットプールから取得されます。

JSON

```
aws autoscaling create-auto-scaling-group --cli-input-json file://~/config.json
```

config.json ファイルには次のコンテンツが含まれます。

```
{
    "AutoScalingGroupName": "my-asg",
    "MixedInstancesPolicy": {
        "LaunchTemplate": {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "$Default"
            },
            "Overrides": [
                {
                    "InstanceType": "c5.large"
                },
                {
                    "InstanceType": "c5a.large"
                },
                {
                    "InstanceType": "m5.large"
```

```
                },
                {
                    "InstanceType": "m5a.large"
                },
                {
                    "InstanceType": "c4.large"
                },
                {
                    "InstanceType": "m4.large"
                },
                {
                    "InstanceType": "c3.large"
                },
                {
                    "InstanceType": "m3.large"
                }
            ]
        },
        "InstancesDistribution": {
            "OnDemandBaseCapacity": 1,
            "OnDemandPercentageAboveBaseCapacity": 0,
            "SpotAllocationStrategy": "capacity-optimized"
        }
    },
    "MinSize": 1,
    "MaxSize": 5,
    "DesiredCapacity": 3,
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
}
```

YAML

または、次の create-auto-scaling-group コマンドを使用して、Auto Scaling グループを作成します。これは、Auto Scaling グループの唯一のパラメータとして YAML ファイルを参照します。

```
aws autoscaling create-auto-scaling-group --cli-input-yaml file://~/config.yaml
```

config.yaml ファイルには次のコンテンツが含まれます。

```
---
AutoScalingGroupName: my-asg
MixedInstancesPolicy:
```

```
  LaunchTemplate:
    LaunchTemplateSpecification:
      LaunchTemplateName: my-launch-template
      Version: $Default
    Overrides:
    - InstanceType: c5.large
    - InstanceType: c5a.large
    - InstanceType: m5.large
    - InstanceType: m5a.large
    - InstanceType: c4.large
    - InstanceType: m4.large
    - InstanceType: c3.large
    - InstanceType: m3.large
  InstancesDistribution:
    OnDemandBaseCapacity: 1
    OnDemandPercentageAboveBaseCapacity: 0
    SpotAllocationStrategy: capacity-optimized
MinSize: 1
MaxSize: 5
DesiredCapacity: 3
VPCZoneIdentifier: subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782
```

例 2: **capacity-optimized-prioritized** 割り当て戦略を使用して スポットインスタンス を起動する

次の [create-auto-scaling-group] コマンドは、以下を指定する Auto Scaling グループを作成します。

- オンデマンドインスタンスとして起動するグループの割合 (0) と開始時のオンデマンドインスタンスのベース数 (1)
- 優先度に従って起動するインスタンスタイプ (c5.large、c5a.large、m5.large、m5a.large、c4.large、m4.large、c3.large、m3.large)
- インスタンスを起動するサブネット (subnet-5ea0c127、subnet-6194ea3b、subnet-c934b782)。それぞれが異なるアベイラビリティーゾーンに対応します。
- 起動テンプレート (my-launch-template) とそのバージョン ($Latest)。

Amazon EC2 Auto Scaling がオンデマンドキャパシティーを満たそうとするとき、まず c5.large インスタンスタイプを起動します。Amazon EC2 Auto Scaling がスポットキャパシティーを満たそうとするとき、ベストエフォートベースでインスタンスタイプの優先順位を尊重します。ただし、最初にキャパシティを最適化します。

JSON

```
aws autoscaling create-auto-scaling-group --cli-input-json file://~/config.json
```

config.json ファイルには次のコンテンツが含まれます。

```
{
    "AutoScalingGroupName": "my-asg",
    "MixedInstancesPolicy": {
        "LaunchTemplate": {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "$Latest"
            },
            "Overrides": [
                {
                    "InstanceType": "c5.large"
                },
                {
                    "InstanceType": "c5a.large"
                },
                {
                    "InstanceType": "m5.large"
                },
                {
                    "InstanceType": "m5a.large"
                },
                {
                    "InstanceType": "c4.large"
                },
                {
                    "InstanceType": "m4.large"
                },
                {
                    "InstanceType": "c3.large"
                },
                {
                    "InstanceType": "m3.large"
                }
            ]
        },
        "InstancesDistribution": {
            "OnDemandBaseCapacity": 1,
```

```
            "OnDemandPercentageAboveBaseCapacity": 0,
            "SpotAllocationStrategy": "capacity-optimized-prioritized"
        }
    },
    "MinSize": 1,
    "MaxSize": 5,
    "DesiredCapacity": 3,
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
}
```

YAML

または、次の create-auto-scaling-group コマンドを使用して、Auto Scaling グループを作成します。これは、Auto Scaling グループの唯一のパラメータとして YAML ファイルを参照します。

```
aws autoscaling create-auto-scaling-group --cli-input-yaml file://~/config.yaml
```

config.yaml ファイルには次のコンテンツが含まれます。

```
---
AutoScalingGroupName: my-asg
MixedInstancesPolicy:
  LaunchTemplate:
    LaunchTemplateSpecification:
      LaunchTemplateName: my-launch-template
      Version: $Default
    Overrides:
    - InstanceType: c5.large
    - InstanceType: c5a.large
    - InstanceType: m5.large
    - InstanceType: m5a.large
    - InstanceType: c4.large
    - InstanceType: m4.large
    - InstanceType: c3.large
    - InstanceType: m3.large
  InstancesDistribution:
    OnDemandBaseCapacity: 1
    OnDemandPercentageAboveBaseCapacity: 0
    SpotAllocationStrategy: capacity-optimized-prioritized
MinSize: 1
MaxSize: 5
DesiredCapacity: 3
```

```
VPCZoneIdentifier: subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782
```

例 3: 2つのプール間での **lowest-price** 配分戦略を使用してスポットインスタンスを起動する

次の [create-auto-scaling-group] コマンドは、以下を指定する Auto Scaling グループを作成します。

- オンデマンドインスタンスとして起動するグループの割合 (50)。(これは、開始時のオンデマンドインスタンスのベース数を指定するものではありません)。
- 優先度に従って起動するインスタンスタイプ (c5.large、c5a.large、m5.large、m5a.large、c4.large、m4.large、c3.large、m3.large)
- インスタンスを起動するサブネット (subnet-5ea0c127、subnet-6194ea3b、subnet-c934b782)。それぞれが異なるアベイラビリティーゾーンに対応します。
- 起動テンプレート (my-launch-template) とそのバージョン ($Latest)。

Amazon EC2 Auto Scaling がオンデマンドキャパシティーを満たそうとするとき、まず c5.large インスタンスタイプを起動します。スポットキャパシティーについては、Amazon EC2 Auto Scaling は、各アベイラビリティーゾーンで 2 つの最低価格のプールのスポットインスタンスを均等に起動しようとします。

JSON

```
aws autoscaling create-auto-scaling-group --cli-input-json file://~/config.json
```

config.json ファイルには次のコンテンツが含まれます。

```
{
    "AutoScalingGroupName": "my-asg",
    "MixedInstancesPolicy": {
        "LaunchTemplate": {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "$Latest"
            },
            "Overrides": [
                {
                    "InstanceType": "c5.large"
                },
                {
                    "InstanceType": "c5a.large"
```

```
                },
                {
                    "InstanceType": "m5.large"
                },
                {
                    "InstanceType": "m5a.large"
                },
                {
                    "InstanceType": "c4.large"
                },
                {
                    "InstanceType": "m4.large"
                },
                {
                    "InstanceType": "c3.large"
                },
                {
                    "InstanceType": "m3.large"
                }
            ]
        },
        "InstancesDistribution": {
            "OnDemandPercentageAboveBaseCapacity": 50,
            "SpotAllocationStrategy": "lowest-price",
            "SpotInstancePools": 2
        }
    },
    "MinSize": 1,
    "MaxSize": 5,
    "DesiredCapacity": 3,
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
}
```

YAML

または、次の create-auto-scaling-group コマンドを使用して、Auto Scaling グループを作成します。これは、Auto Scaling グループの唯一のパラメータとして YAML ファイルを参照します。

```
aws autoscaling create-auto-scaling-group --cli-input-yaml file://~/config.yaml
```

config.yaml ファイルには次のコンテンツが含まれます。

```
---
```

```
AutoScalingGroupName: my-asg
MixedInstancesPolicy:
  LaunchTemplate:
    LaunchTemplateSpecification:
      LaunchTemplateName: my-launch-template
      Version: $Default
    Overrides:
    - InstanceType: c5.large
    - InstanceType: c5a.large
    - InstanceType: m5.large
    - InstanceType: m5a.large
    - InstanceType: c4.large
    - InstanceType: m4.large
    - InstanceType: c3.large
    - InstanceType: m3.large
  InstancesDistribution:
    OnDemandPercentageAboveBaseCapacity: 50
    SpotAllocationStrategy: lowest-price
    SpotInstancePools: 2
MinSize: 1
MaxSize: 5
DesiredCapacity: 3
VPCZoneIdentifier: subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782
```

例 4: 配分戦略を使用して スポットインスタンス を起動する**price-capacity-optimized**

次の [create-auto-scaling-group] コマンドは、以下を指定する Auto Scaling グループを作成します。

- オンデマンドインスタンスとして起動するグループの割合 (30)。(これは、開始時のオンデマンドインスタンスのベース数を指定するものではありません)。
- 優先度に従って起動するインスタンスタイプ (c5.large、c5a.large、m5.large、m5a.large、c4.large、m4.large、c3.large、m3.large)
- インスタンスを起動するサブネット (subnet-5ea0c127、subnet-6194ea3b、subnet-c934b782)。それぞれが異なるアベイラビリティーゾーンに対応します。
- 起動テンプレート (my-launch-template) とそのバージョン ($Latest)。

Amazon EC2 Auto Scaling がオンデマンドキャパシティーを満たそうとするとき、まず c5.large インスタンスタイプを起動します。スポットキャパシティについては、Amazon EC2 Auto Scaling は、可能な限り低料金かつ起動するインスタンス数に最適なキャパシティを持つスポットインスタンスを、スポットインスタンスプールから起動しようとします。

JSON

```
aws autoscaling create-auto-scaling-group --cli-input-json file://~/config.json
```

config.json ファイルには次のコンテンツが含まれます。

```
{
    "AutoScalingGroupName": "my-asg",
    "MixedInstancesPolicy": {
        "LaunchTemplate": {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "$Latest"
            },
            "Overrides": [
                {
                    "InstanceType": "c5.large"
                },
                {
                    "InstanceType": "c5a.large"
                },
                {
                    "InstanceType": "m5.large"
                },
                {
                    "InstanceType": "m5a.large"
                },
                {
                    "InstanceType": "c4.large"
                },
                {
                    "InstanceType": "m4.large"
                },
                {
                    "InstanceType": "c3.large"
                },
                {
                    "InstanceType": "m3.large"
                }
            ]
        },
        "InstancesDistribution": {
            "OnDemandPercentageAboveBaseCapacity": 30,
```

```
            "SpotAllocationStrategy": "price-capacity-optimized"
        }
    },
    "MinSize": 1,
    "MaxSize": 5,
    "DesiredCapacity": 3,
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
}
```

YAML

または、次の create-auto-scaling-group コマンドを使用して、Auto Scaling グループを作成します。これは、Auto Scaling グループの唯一のパラメータとして YAML ファイルを参照します。

```
aws autoscaling create-auto-scaling-group --cli-input-yaml file://~/config.yaml
```

config.yaml ファイルには次のコンテンツが含まれます。

```
---
AutoScalingGroupName: my-asg
MixedInstancesPolicy:
  LaunchTemplate:
    LaunchTemplateSpecification:
      LaunchTemplateName: my-launch-template
      Version: $Default
    Overrides:
    - InstanceType: c5.large
    - InstanceType: c5a.large
    - InstanceType: m5.large
    - InstanceType: m5a.large
    - InstanceType: c4.large
    - InstanceType: m4.large
    - InstanceType: c3.large
    - InstanceType: m3.large
  InstancesDistribution:
    OnDemandPercentageAboveBaseCapacity: 30
    SpotAllocationStrategy: price-capacity-optimized
MinSize: 1
MaxSize: 5
DesiredCapacity: 3
VPCZoneIdentifier: subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782
```

インスタンスの重みを使用するように Auto Scaling グループを設定する

複数のインスタンスタイプを使用する場合、各インスタンスタイプに関連付けるユニット数を指定し、同じ測定単位を持つグループの容量を指定できます。この容量仕様オプションは重みと呼ばれます。

たとえば、少なくとも 8 個の vCPU と 15 GiB の RAM で最高のパフォーマンスを発揮する、コンピューティング集約型のアプリケーションを実行するとします。基本単位として `c5.2xlarge` を使用する場合、以下の EC2 インスタンスタイプのいずれかがアプリケーションのニーズを満たします。

インスタンスタイプの例

インスタンスタイプ	vCPU	メモリ (GiB)
`c5.2xlarge`	8	16
`c5.4xlarge`	16	32
`c5.12xlarge`	48	96
`c5.18xlarge`	72	144
`c5.24xlarge`	96	192

デフォルトでは、すべてのインスタンスタイプは、サイズにかかわらず同じ重みを持ちます。つまり、Amazon EC2 Auto Scaling が起動するインスタンスタイプの規模の大小にかかわらず、各インスタンスは Auto Scaling グループの希望するキャパシティに対して同じようにカウントされます。

ただし、重みでは、各インスタンスタイプに関連付けるユニット数を指定する数値を割り当てます。例えば、インスタンスのサイズが異なる場合、`c5.2xlarge` インスタンスには 2 の重みを付け、`c5.4xlarge` (2 倍大きい) インスタンスには 4 の重みを付けます。その後、Amazon EC2 Auto Scaling がグループをスケールするときに、これらの重みは、各インスタンスが希望するキャパシティとしてカウントするユニット数に変換されます。

重みは、Amazon EC2 Auto Scaling が起動するインスタンスタイプを変更しません。代わりに、割り当て戦略がそれを行います。詳細については、「配分戦略」を参照してください。

> **Important**
>
> vCPU の数または各インスタンスタイプのメモリ量を使用して希望するキャパシティを満たすように Auto Scaling グループを設定するには、属性ベースのインスタンスタイプの選択を使用することをお勧めします。`DesiredCapacityType` パラメータを設定すると、このパラメータに設定した値に基づいて、各インスタンスタイプに関連付けるユニットの数が自動的に指定されます。詳細については、「属性ベースのインスタンスタイプの選択を使用して混合インスタンスグループを作成する」を参照してください。

内容

- 考慮事項
- インスタンスの重みの動作
- 重みを使用するように Auto Scaling グループを設定する
- ユニット時間あたりのスポット料金の例

考慮事項

このセクションでは、重みを効果的に実装するための重要な考慮事項について説明します。

- アプリケーションのパフォーマンスニーズに合ったインスタンスタイプをいくつか選択します。各インスタンスタイプが Auto Scaling グループの希望する容量にカウントする重みを、その機能に基づいて決定します。これらの重みは、現在および将来のインスタンスに適用されます。
- 重みの間に大きな範囲を使用しないでください。例えば、次に大きいインスタンスタイプの重みが 200 の場合、インスタンスタイプに重み 1 を指定しないでください。最小の重みと最大の重みの差も極端であってはなりません。重みの違いは、コストパフォーマンスの最適化に悪影響を及ぼす可能性があります。
- グループの希望する容量をインスタンスではなくユニット単位で指定します。例えば、vCPU ベースの重みを使用する場合は、必要なコア数と最小値と最大値を設定します。
- 重みと希望するキャパシティを設定して、希望するキャパシティが最も大きい重みの少なくとも 2〜3 倍になるようにします。

既存のグループを更新する際には、次の点に注意してください。

- 既存のグループに重みを追加するときは、現在使用中のすべてのインスタンスタイプの重みを含めます。

- 重みを追加または変更すると、Amazon EC2 Auto Scaling はインスタンスを起動または終了して、新しい重み値に基づいて希望する容量に到達します。
- インスタンスタイプを削除すると、そのタイプのインスタンスを実行すると、定義されなくなった場合でも、最後の重みが維持されます。

インスタンスの重みの動作

インスタンスの重みを使用すると、Amazon EC2 Auto Scaling は次のように動作します。

- 現在のキャパシティーは、希望するキャパシティーと同じかそれ以上になります。起動したインスタンスが残りの希望するキャパシティーユニットを超えると、現在のキャパシティーが希望するキャパシティーを超える可能性があります。例えば、2 つのインスタンスタイプ `c5.2xlarge` と `c5.12xlarge` を指定し、`c5.2xlarge` にインスタンスの重み 2 を割り当て、`c5.12xlarge` にインスタンスの重み 12 を割り当てるとします。希望するキャパシティを満たすためのキャパシティが 5 ユニット分残っており、Amazon EC2 Auto Scaling が `c5.12xlarge` をプロビジョニングする場合、希望するキャパシティは 7 ユニット分超過します。
- インスタンスを起動する場合、Amazon EC2 Auto Scaling は、アベイラビリティーゾーン間で容量を分散し、希望する容量を超えるよりも配分戦略を優先します。
- Amazon EC2 Auto Scaling は、希望する配分戦略を使用して、アベイラビリティーゾーン間のバランスを維持するために最大容量制限を超える可能性があります。Amazon EC2 Auto Scaling によって適用されるハード制限は、希望する容量と最大の重みです。

重みを使用するように Auto Scaling グループを設定する

次の AWS CLI の例に示すように、重みを使用するように Auto Scaling グループを設定できます。コンソールを使用する手順については、「インスタンスタイプを手動で選択して混合インスタンスグループを作成する」を参照してください。

重みを使用するように新しい Auto Scaling グループを設定するには (AWS CLI)

create-auto-scaling-group コマンドを使用します。例えば、次のコマンドは新しい Auto Scaling グループを作成し、次を指定して重みを割り当てます。

- オンデマンドインスタンスとして起動するグループの割合 (`0`)
- 各アベイラビリティーゾーンのスポットインスタンスの配分戦略 (`capacity-optimized`)
- 優先度に従って起動するインスタンスタイプ (`m4.16xlarge`、`m5.24xlarge`)
- インスタンスタイプ間の相対的なサイズの違い (vCPU) に対応するインスタンスの重み (16、24)

- インスタンスを起動するサブネット (subnet-5ea0c127、subnet-6194ea3b、subnet-c934b782)。それぞれ異なるアベイラビリティーゾーンに対応
- 起動テンプレート (my-launch-template) とそのバージョン ($Latest)

```
aws autoscaling create-auto-scaling-group --cli-input-json file://~/config.json
```

config.json ファイルには次のコンテンツが含まれます。

```
{
    "AutoScalingGroupName": "my-asg",
    "MixedInstancesPolicy": {
        "LaunchTemplate": {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "$Latest"
            },
            "Overrides": [
                {
                    "InstanceType": "m4.16xlarge",
                    "WeightedCapacity": "16"
                },
                {
                    "InstanceType": "m5.24xlarge",
                    "WeightedCapacity": "24"
                }
            ]
        },
        "InstancesDistribution": {
            "OnDemandPercentageAboveBaseCapacity": 0,
            "SpotAllocationStrategy": "capacity-optimized"
        }
    },
    "MinSize": 160,
    "MaxSize": 720,
    "DesiredCapacity": 480,
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782",
    "Tags": []
}
```

重みを使用するように既存の Auto Scaling グループを設定するには (AWS CLI)

update-auto-scaling-group コマンドを使用します。例えば、ここで示すコマンドは、次を指定して既存の Auto Scaling グループのインスタンスタイプに重みを割り当てます。

- 優先度に従って起動するインスタンスタイプ (c5.18xlarge、c5.24xlarge、c5.2xlarge、c5.4xlarge)
- インスタンスタイプ間の相対的なサイズの違い (vCPU) に対応するインスタンスの重み (18、24、2、4)
- 増やす新しい希望するキャパシティー (最大重みよりも大きい)

```
aws autoscaling update-auto-scaling-group --cli-input-json file://~/config.json
```

config.json ファイルには次のコンテンツが含まれます。

```
{
    "AutoScalingGroupName": "my-existing-asg",
    "MixedInstancesPolicy": {
        "LaunchTemplate": {
            "Overrides": [
                {
                    "InstanceType": "c5.18xlarge",
                    "WeightedCapacity": "18"
                },
                {
                    "InstanceType": "c5.24xlarge",
                    "WeightedCapacity": "24"
                },
                {
                    "InstanceType": "c5.2xlarge",
                    "WeightedCapacity": "2"
                },
                {
                    "InstanceType": "c5.4xlarge",
                    "WeightedCapacity": "4"
                }
            ]
        }
    },
    "MinSize": 0,
    "MaxSize": 100,
    "DesiredCapacity": 100
```

```
}
```

コマンドラインを使用して重みを検証するには

以下のいずれかのコマンドを使用します。

- describe-auto-scaling-groups (AWS CLI)
- Get-AS AutoScalingグループ (AWS Tools for Windows PowerShell)

ユニット時間あたりのスポット料金の例

次の表では、米国東部 (バージニア北部) の異なるアベイラビリティーゾーンでのスポットインスタンスの時間単位の使用料金と、同じリージョンでのオンデマンドインスタンスの使用料金を比較しています。ここで示している料金は例であり、現在の料金ではありません。これらはインスタンス時間単位のコストです。

例: インスタンス時間単位のスポット料金

インスタンスタイプ	us-east-1a	us-east-1b	us-east-1c	オンデマンド料金
c5.2xlarge	0.180 USD	0.191 USD	0.170 USD	0.34 USD
c5.4xlarge	0.341 USD	0.361 USD	0.318 USD	0.68 USD
c5.12xlarge	0.779 USD	0.777 USD	0.777 USD	2.04 USD
c5.18xlarge	1.207 USD	1.475 USD	1.357 USD	3.06 USD
c5.24xlarge	1.555 USD	1.555 USD	1.555 USD	4.08 USD

インスタンスの重みにより、ユニット時間あたりの使用料金に基づいてコストを評価できます。時間単位の使用料金はインスタンスタイプの料金をその単位となる時間数で割って決定できます。オンデマンドインスタンスの場合、時間単位の使用料金は、1 つのインスタンスタイプをデプロイすると

きと、異なるサイズの同じインスタンスタイプをデプロイするときで同じです。一方、時間単位のスポット料金はスポットプールによって異なります。

次の例は、ユニット時間あたりのスポット料金の計算がインスタンスの重みでどのように機能するかを示しています。計算が容易になるように、スポットインスタンスを us-east-1a でのみ起動するとします。ユニット時間あたりの料金を次の表に示します。

例: 時間単位のスポット料金

インスタンスタイプ	us-east-1a	インスタンスの重み	ユニット時間あたりの価格
c5.2xlarge	0.180 USD	2	0.090 USD
c5.4xlarge	0.341 USD	4	0.085 USD
c5.12xlarge	0.779 USD	12	0.065 USD
c5.18xlarge	1.207 USD	18	0.067 USD
c5.24xlarge	1.555 USD	24	0.065 USD

インスタンスタイプに異なる起動テンプレートを使用する

複数のインスタンスタイプを使用するだけでなく、複数の起動テンプレートを使用することもできます。

例えば、コンピューティング集約型アプリケーション用に Auto Scaling グループを設定し、C5、C5a、C6g のインスタンスタイプを混在させたいとします。ただし、C6g インスタンスは 64 ビット Arm アーキテクチャに基づく AWS Graviton プロセッサを搭載し、C5 および C5a インスタンスは 64 ビット Intel x86 プロセッサで実行されます。C5 インスタンスと C5a インスタンスの AMI はどちらも、これらの各インスタンスで機能しますが、C6g インスタンスでは機能しません。この問題を解決するには、C6g インスタンス用に別の起動テンプレートを使用します。C5 インスタンスと C5a インスタンスには同じ起動テンプレートを引き続き使用できます。

このセクションでは、 を使用して AWS CLI 、複数の起動テンプレートの使用に関連するタスクを実行する手順について説明します。現在、この特徴は AWS CLI または SDK を使用している場合のみ利用可能で、コンソールからは利用できません。

内容

- 複数の起動テンプレートを使用するように Auto Scaling グループを設定する
- 関連リソース

複数の起動テンプレートを使用するように Auto Scaling グループを設定する

次の例に示すように、複数の起動テンプレートを使用するように Auto Scaling グループを設定できます。

複数の起動テンプレートを使用するように新しい Auto Scaling グループを設定するには (AWS CLI)

create-auto-scaling-group コマンドを使用します。例えば、次のコマンドは新しい Auto Scaling グループを作成します。これは、`c5.large`、`c5a.large`、および`c6g.large`のインスタンスタイプを指定し、`c6g.large`のインスタンスタイプに新しい起動テンプレートを定義することで、適切な AMI を使用して Arm インスタンスを起動することを保証します。Amazon EC2 Auto Scalingは、インスタンスタイプの順序を使用して、オンデマンドキャパシティーを満たすときに最初に使用するインスタンスタイプを決定します。

```
aws autoscaling create-auto-scaling-group --cli-input-json file://~/config.json
```

`config.json` ファイルには次のコンテンツが含まれます。

```
{
  "AutoScalingGroupName":"my-asg",
  "MixedInstancesPolicy":{
    "LaunchTemplate":{
      "LaunchTemplateSpecification":{
        "LaunchTemplateName":"my-launch-template-for-x86",
        "Version":"$Latest"
      },
      "Overrides":[
        {
          "InstanceType":"c6g.large",
          "LaunchTemplateSpecification": {
            "LaunchTemplateName": "my-launch-template-for-arm",
            "Version": "$Latest"
          }
        },
        {
          "InstanceType":"c5.large"
        },
        {
```

```
          "InstanceType":"c5a.large"
        }
      ]
    },
    "InstancesDistribution":{
      "OnDemandBaseCapacity": 1,
      "OnDemandPercentageAboveBaseCapacity": 50,
      "SpotAllocationStrategy": "capacity-optimized"
    }
  },
  "MinSize":1,
  "MaxSize":5,
  "DesiredCapacity":3,
  "VPCZoneIdentifier":"subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782",
  "Tags":[ ]
}
```

複数の起動テンプレートを使用するように既存の Auto Scaling グループを設定するには (AWS CLI)

update-auto-scaling-group コマンドを使用します。例えば、次のコマンドは、*my-launch-template-for-arm* という名前の起動テンプレートを、*my-asg* という名前の Auto Scaling グループの *c6g.large* インスタンスタイプに割り当てます。

```
aws autoscaling update-auto-scaling-group --cli-input-json file://~/config.json
```

config.json ファイルには次のコンテンツが含まれます。

```
{
  "AutoScalingGroupName":"my-asg",
  "MixedInstancesPolicy":{
    "LaunchTemplate":{
      "Overrides":[
        {
          "InstanceType":"c6g.large",
          "LaunchTemplateSpecification": {
            "LaunchTemplateName": "my-launch-template-for-arm",
            "Version": "$Latest"
          }
        },
        {
          "InstanceType":"c5.large"
        },
```

```
        {
          "InstanceType":"c5a.large"
        }
      ]
    }
  }
}
```

Auto Scaling グループの起動テンプレートを確認するには

以下のいずれかのコマンドを使用します。

- describe-auto-scaling-groups (AWS CLI)
- Get-AS AutoScalingグループ (AWS Tools for Windows PowerShell)

関連リソース

AWS re:Post のテンプレートで、属性ベースのインスタンスタイプの選択を使用して複数の起動 AWS CloudFormation テンプレートを指定する例を確認できます。

起動設定を使用して Auto Scaling グループを作成する

> **Important**
>
> 2022 年 12 月 31 日以降にリリースされた新しい Amazon EC2 インスタンスタイプで `CreateLaunchConfiguration` を呼び出すことはできません。また、2023 年 6 月 1 日以降に作成された新しいアカウントには、コンソールから新しい起動設定を作成することはできません。今後、新しいアカウントは、コンソール、API、CLI、および を使用して新しい起動設定を作成できなくなります CloudFormation。起動テンプレートに移行して、現在または将来に新しい起動設定を作成する必要がないようにします。Auto Scaling グループを移行してテンプレートを起動する方法については、「Auto Scaling グループを起動テンプレートに移行する」を参照してください。

起動設定または EC2 インスタンスを作成した場合は、起動設定を EC2 インスタンスの設定テンプレートとして使用する Auto Scaling グループを作成できます。起動設定は、インスタンスの AMI ID、インスタンスタイプ、キーペア、セキュリティグループ、ブロックデバイスマッピングなどの情報を指定します。起動設定の作成については、「「起動設定を作成する」」を参照してください。

Auto Scaling グループを作成するための十分な許可が必要です。また、サービスにリンクされたロールが存在しない場合は、Amazon EC2 Auto Scaling がユーザーに代わってアクションを実行するために使用する、サービスにリンクされたロールを作成するための十分な許可も必要です。管理者がアクセス許可を付与するための参照として使用できる IAM ポリシーの例については、アイデンティティベースポリシーの例 を参照してください。

内容

- 起動設定を使用して Auto Scaling グループを作成する
- 既存のインスタンスからのパラメータを使用して Auto Scaling グループを作成する

起動設定を使用して Auto Scaling グループを作成する

> **Important**
>
> 起動設定に関する情報は、起動設定から起動テンプレートにまだ移行していないお客様向けに提供しています。Auto Scaling グループを移行してテンプレートを起動する方法については、「Auto Scaling グループを起動テンプレートに移行する」を参照してください。

Auto Scaling グループを作成する際、Amazon EC2 インスタンスを設定するために必要な情報、そのインスタンスのアベイラビリティーゾーンと VPC サブネット、希望するキャパシティー、キャパシティー制限の最小値と最大値を指定する必要があります。

次の手順では、起動設定を使用して Auto Scaling グループを作成する方法を説明します。起動設定は作成後に変更することはできませんが、Auto Scaling グループの起動設定を置き換えることはできます。詳細については、「Auto Scaling グループの起動設定を変更する」を参照してください。

前提条件

- 起動設定を作成しておく必要があります。詳細については、「「起動設定を作成する」」を参照してください。

起動設定 (コンソール) を使用して Auto Scaling グループを作成するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。

2. 画面上部のナビゲーションバーで、起動設定の作成時に使用した AWS リージョン ものと同じを選択します。
3. [Auto Scaling グループの作成] を選択します。
4. [起動テンプレートまたは起動設定を選択する] ページで [Auto Scaling グループ名] にAuto Scaling グループの名前を入力します。
5. 起動設定を選択するには、次の手順を実行します。

   a. [Launch template] (起動テンプレート) で、[Switch to launch configuration] (起動設定に切り替える) を選択します。

   b. [起動設定] で、既存の起動設定を選択します。

   c. 起動設定が、使用する予定のすべてのオプションをサポートしていることを確認し、[次へ] を選択します。
6. (設定の構成)、[Configure instance launch options] (インスタンス起動オプションの設定) ページの [Network] (ネットワーク) にある [VPC] で、VPC を選択します。Auto Scaling グループは、起動設定で指定したセキュリティグループと、同じ VPC 内に作成する必要があります。
7. (サブネット)、[Availability Zones and subnets] (アベイラビリティーゾーンとサブネット) で、指定した VPC 内のサブネットを 1 つ以上選択します。複数のアベイラビリティーゾーンのサブネットを使用することで、高可用性を得られます。詳細については、「VPC サブネットを選択する場合の考慮事項」を参照してください。
8. [Next] (次へ) を選択します。

   または、残りはデフォルトのままにして、[Skip to Review (確認をスキップ)] を選択できます。
9. (オプション) [詳細オプションの設定] ページで、次のオプションを設定し、[次へ] を選択します。

   a. 「追加設定」「モニタリング」で、 CloudWatch グループメトリクスの収集を有効にするかどうかを選択します。これらのメトリクスは、終了インスタンス数や保留中のインスタンスの数など、潜在的な問題の指標となる測定値を提供します。詳細については、「Auto Scaling グループとインスタンスの CloudWatch メトリクスをモニタリングする」を参照してください。

   b. デフォルトのインスタンスウォームアップを有効にする で、このオプションを選択し、アプリケーションのウォームアップ時間を選択します。スケーリングポリシーを持つ Auto Scaling グループを作成する場合、デフォルトのインスタンスウォームアップ機能により、動的スケーリングに使用される Amazon CloudWatch メトリクスが改善されます。詳細につ

いては、「Auto Scaling グループに対するインスタンスのデフォルトウォームアップを設定する」を参照してください。

10. (オプション) [Configure group size and scaling policies (グループサイズとスケーリングポリシーの設定)] ページで、次のオプションを設定し、[次へ] を選択します。

    a. グループサイズ で、希望する容量 に、起動するインスタンスの初期数を入力します。

    b. スケーリングセクションのスケーリング制限 で、希望する容量の新しい値が希望する最小容量と希望する最大容量 より大きい場合、希望する最大容量は自動的に新しい希望する容量値に増加します。これらの制限は必要に応じて変更できます。詳細については、「Auto Scaling グループのスケーリング制限を設定する」を参照してください。

    c. 自動スケーリング では、ターゲット追跡スケーリングポリシーを作成するかどうかを選択します。このポリシーは、Auto Scaling グループの作成後に作成することもできます。

       ターゲット追跡スケーリングポリシー を選択した場合は、「」の指示に従ってポリシーターゲット追跡スケーリングポリシーを作成するを作成します。

    d. インスタンスメンテナンスポリシー で、インスタンスメンテナンスポリシーを作成するかどうかを選択します。このポリシーは、Auto Scaling グループの作成後に作成することもできます。インスタンスのメンテナンスポリシーを設定する 「」の指示に従ってポリシーを作成します。

    e. [Instance scale-in protection (インスタンスのスケールイン保護)] で、インスタンスのスケールイン保護を有効にするかどうかを選択します。詳細については、「インスタンスのスケールイン保護を使用する」を参照してください。

11. (オプション) 通知を受け取るには、[通知の追加] を選択し、通知を設定してから [次へ] を選択します。詳細については、「Amazon EC2 Auto Scaling の Amazon SNS 通知オプション Amazon EC2 Auto Scaling」を参照してください。

12. (オプション) タグを追加するには、[タグの追加] を選択し、各タグのタグキーと値を指定し、[次へ] を選択します。詳細については、「Auto Scaling グループとインスタンスにタグを付ける」を参照してください。

13. [Review (レビュー)]ページで、[Create Auto Scaling group (Auto Scaling グループを作成)] を選択します。

コマンドラインを使用して Auto Scaling グループを作成するには

以下のコマンドのいずれかを使用できます。

- create-auto-scaling-group (AWS CLI)

- 新しい ASAutoScalingGroup (AWS Tools for Windows PowerShell)

既存のインスタンスからのパラメータを使用して Auto Scaling グループを作成する

> **Important**
>
> 起動設定に関する情報は、起動設定から起動テンプレートにまだ移行していないお客様向けに提供しています。Auto Scaling グループを移行してテンプレートを起動する方法については、「Auto Scaling グループを起動テンプレートに移行する」を参照してください。

Auto Scaling グループを初めて作成する場合は、コンソールを使用して、既存の EC2 インスタンスから起動テンプレートを作成することをお勧めします。次に、起動テンプレートを使用して新しい Auto Scaling グループを作成します。詳しい手順については、「Amazon EC2 起動ウィザードを使用して Auto Scaling グループを作成する」を参照してください。

次の手順は、他のインスタンスを起動するためのベースとして使用する、既存のインスタンスを指定して Auto Scaling グループを作成する方法を示しています。EC2 インスタンスを作成するには、Amazon Machine Image (AMI) ID、インスタンスタイプ、キーペア、セキュリティグループなど、複数のパラメータが必要です。Amazon EC2 Auto Scaling では、スケーリングが必要な場合にユーザーに代わってインスタンスを起動するために、これらの情報もすべて使われます。この情報は、起動テンプレートまたは起動設定のいずれかに保存されます。

既存インスタンスを指定すると、同時に作成された起動設定に基づいて、Amazon EC2 Auto Scaling がインスタンスを起動する Auto Scaling グループを作成します。新しい起動設定には Auto Scaling グループと同じ名前がついていて、インスタンスからの特定の設定についての詳細が含まれます。

次の設定詳細は、特定のインスタンスから起動設定にコピーされます。

- AMI ID
- インスタンスタイプ
- キーペア
- セキュリティグループ
- IP アドレスタイプ (パブリックまたはプライベート)
- IAM インスタンスプロファイル (該当する場合)
- モニタリング (true または false)

- EBS 最適化 (true または false)
- VPC (共有または専用) に起動する場合は、テナンシー設定
- 該当する場合は、カーネル ID および RAM ディスク ID
- ユーザーデータ (指定された場合)
- スポット (最大) 料金

VPC サブネットおよびアベイラビリティーゾーンは、指定のインスタンスから Auto Scaling グループの独自のリソース定義にコピーされます。

特定されたインスタンスがプレイスメントグループ内にある場合、新しい Auto Scaling グループは、特定されたインスタンスと同じプレイスメントグループ内でインスタンスを起動します。起動設定ではプレイスメントグループの指定が許可されないため、プレイスメントグループは新しい Auto Scaling グループの `PlacementGroup` 属性にコピーされます。

次の設定の詳細は、特定のインスタンスからはコピーされません。

- ストレージ: ブロックデバイス (EBS ボリュームとインスタンスストアボリューム) は、特定のインスタンスからコピーされません。代わりに、AMI 作成の一部として作成されたブロックデバイスマッピングによって、使用されるデバイスが決まります。
- ネットワークインターフェイスの数: ネットワークインターフェイスは、特定のインスタンスからコピーされません。代わりにデフォルト設定を使用して、Amazon EC2 Auto Scaling が、プライマリネットワークインターフェイス (eth0) であるネットワークインターフェイスを 1 つ作成します。
- インスタンスメタデータオプション: アクセス可能なメタデータ、メタデータのバージョン、およびトークンレスポンスのホップ制限の設定は、特定のインスタンスからコピーされません。代わりに、Amazon EC2 Auto Scaling はデフォルト設定を使用します。詳細については、「インスタンスメタデータオプションの設定」を参照してください。
- ロードバランサー: 識別されたインスタンスが 1 つ以上のロードバランサーに登録されている場合、このロードバランサーの情報は新しい Auto Scaling グループのロードバランサーあるいはターゲットグループ属性にコピーされません。
- タグ: 特定されたインスタンスにタグが指定されている場合、そのタグは新しい Auto Scaling グループの `Tags` 属性にはコピーされません。

前提条件

EC2 インスタンスは以下の基準を満たす必要があります。

- インスタンスは他の Auto Scaling グループのメンバーではありません。
- インスタンスが `running` 状態であること。
- インスタンスの起動に使用した AMI は引き続き存在している必要があります。

EC2 インスタンスを使用して Auto Scaling グループを作成する (コンソール)

EC2 インスタンスから Auto Scaling グループを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインの [Instances] (インスタンス) で [Instances] (インスタンス) を選択してから、インスタンスを選択します。
3. [Actions]、[Instance Settings]、[Attach to Auto Scaling Group] の順に選択します。
4. [Attach to Auto Scaling group (Auto Scaling グループ にアタッチ)] ページで、[Auto Scaling Group (新しい Auto Scaling Group)] を選択し、そのグループの名前を入力して、[Attach (アタッチ)] を選択します。

   アタッチされたインスタンスは、Auto Scaling グループの一部と見なされます。新しい Auto Scaling グループは、Auto Scaling グループに指定したものと同じ名前で、新しい起動設定を使用して作成されます。Auto Scaling グループには、目的の容量と 1 の最大サイズがあります。
5. (オプション) Auto Scaling グループの設定を編集するには、ナビゲーションペインの [Auto Scaling] で [Auto Scaling Groups] (Auto Scaling グループ) を選択します。新しい Auto Scaling グループの横にあるチェックボックスをオンにし、グループリストの上にある [Edit (編集)] ボタンを選択します。必要に応じて設定を変更し、[Update (更新)] を選択します。

EC2 インスタンスを使用して Auto Scaling グループを作成する (AWS CLI)

次の手順は、CLI コマンドを使用して EC2 インスタンスから Auto Scaling グループを作成する方法を示しています。

この手順では、Auto Scaling グループにインスタンスを追加しません。Auto Scaling グループを作成した後、アタッチするインスタンスに attach-instances コマンドを実行する必要があります。

開始する前に、Amazon EC2 コンソールまたは describe-instances コマンドを使用して EC2 インスタンスの ID を見つけます。

現在のインスタンスをテンプレートとして使用するには

- 以下の create-auto-scaling-group コマンドを使用して、EC2 インスタンスi-0e69cc3f05f825f4fから Auto Scaling グループ、my-asg-from-instance を作成します。

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-asg-from-instance \
  --instance-id i-0e69cc3f05f825f4f --min-size 1 --max-size 2 --desired-capacity 2
```

Auto Scaling グループがインスタンスを起動したことを確認するには

- 以下のdescribe-auto-scaling-group コマンドを使用して、Auto Scaling グループが正常に作成されたことを確認します。

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg-from-instance
```

次のレスポンスの例は、グループの希望するキャパシティーが 2 であり、グループには実行中のインスタンスが 2 つあり、起動設定の名前が my-asg-from-instance であることを示しています。

```
{
  "AutoScalingGroups":[
    {
      "AutoScalingGroupName":"my-asg-from-instance",
      "AutoScalingGroupARN":"arn",
      "LaunchConfigurationName":"my-asg-from-instance",
      "MinSize":1,
      "MaxSize":2,
      "DesiredCapacity":2,
      "DefaultCooldown":300,
      "AvailabilityZones":[
        "us-west-2a"
      ],
      "LoadBalancerNames":[],
      "TargetGroupARNs":[],
      "HealthCheckType":"EC2",
      "HealthCheckGracePeriod":0,
      "Instances":[
```

```
        {
          "InstanceId":"i-06905f55584de02da",
          "InstanceType":"t2.micro",
          "AvailabilityZone":"us-west-2a",
          "LifecycleState":"InService",
          "HealthStatus":"Healthy",
          "LaunchConfigurationName":"my-asg-from-instance",
          "ProtectedFromScaleIn":false
        },
        {
          "InstanceId":"i-087b42219468eacde",
          "InstanceType":"t2.micro",
          "AvailabilityZone":"us-west-2a",
          "LifecycleState":"InService",
          "HealthStatus":"Healthy",
          "LaunchConfigurationName":"my-asg-from-instance",
          "ProtectedFromScaleIn":false
        }
      ],
      "CreatedTime":"2020-10-28T02:39:22.152Z",
      "SuspendedProcesses":[ ],
      "VPCZoneIdentifier":"subnet-6bea5f06",
      "EnabledMetrics":[ ],
      "Tags":[ ],
      "TerminationPolicies":[
        "Default"
      ],
      "NewInstancesProtectedFromScaleIn":false,
      "ServiceLinkedRoleARN":"arn",
      "TrafficSources":[]
    }
  ]
}
```

起動設定を表示するには

- 以下の describe-launch-configurations コマンドを使用して、起動設定の詳細を表示します。

```
aws autoscaling describe-launch-configurations --launch-configuration-names my-asg-from-instance
```

出力例を次に示します。

```
{
  "LaunchConfigurations":[
    {
      "LaunchConfigurationName":"my-asg-from-instance",
      "LaunchConfigurationARN":"arn",
      "ImageId":"ami-0528a5175983e7f28",
      "KeyName":"my-key-pair-uswest2",
      "SecurityGroups":[
        "sg-05eaec502fcdadc2e"
      ],
      "ClassicLinkVPCSecurityGroups":[ ],
      "UserData":"",
      "InstanceType":"t2.micro",
      "KernelId":"",
      "RamdiskId":"",
      "BlockDeviceMappings":[ ],
      "InstanceMonitoring":{
        "Enabled":true
      },
      "CreatedTime":"2020-10-28T02:39:22.321Z",
      "EbsOptimized":false,
      "AssociatePublicIpAddress":true
    }
  ]
}
```

インスタンスを終了するには

- 必要がなくなった場合は、インスタンスを終了できます。以下の terminate-instances コマンドで、インスタンスを終了しますi-0e69cc3f05f825f4f。

```
aws ec2 terminate-instances --instance-ids i-0e69cc3f05f825f4f
```

Amazon EC2 インスタンスを削除すると、再起動できなくなります。削除後、ボリュームに含まれるデータは消去され、ボリューム自体はどのインスタンスにもアタッチできなくなります。インスタンスの終了の詳細については、Amazon EC2 ユーザーガイド」の「インスタンスの終了」を参照してください。

Auto Scaling グループを更新する

Auto Scaling グループの詳細は更新することができます。Auto Scaling グループの名前を更新したり、その を変更したりすることはできません AWS リージョン。

Auto Scaling グループ (コンソール) を更新するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループを選択すると、[詳細]、[アクティビティ]、[自動スケーリング]、[インスタンス管理]、[モニタリング]、[インスタンス更新] のタブでグループに関する情報が表示されます。
3. 目的の設定エリアのタブを選択し、必要に応じて設定を更新します。編集する各設定について、[更新] を選択して Auto Scaling グループの設定に変更を保存します。

   - [詳細] タブ

     以下は Auto Scaling グループの一般的な設定です。これらのルールは、Auto Scaling グループの作成時と同じ方法で編集および管理できます。

     詳細設定セクションには、[終了ポリシー]、[クールダウン]、[一時停止プロセス]、[最大インスタンス有効期間]など、グループの作成時には使用できないオプションがいくつかあります。Auto Scaling グループのプレイスメントグループとサービスにリンクされたロールは表示することはできますが、編集することはできません。

     グループが Elastic Load Balancing リソースに関連付けられている場合は、アベイラビリティーゾーンを変更する前に アベイラビリティーゾーンを追加および削除する を参照してください。ロードバランサーの制限によっては、グループのアベイラビリティーゾーンへの変更をロードバランサーのアベイラビリティーゾーンに適用できない場合があります。

   - [アクティビティ] タブ
     - アクティビティ通知 – Amazon SNS 通知
   - [自動スケーリング] タブ
     - 動的スケーリングポリシー — 動的スケーリングポリシー
     - 予測スケーリングポリシー — 予測スケーリングポリシー
     - スケジュールされたアクション – スケジュールされたアクション
   - [インスタンス管理] タブ

    - ライフサイクルフック – ライフサイクルフック
    - ウォームプール - ウォームプール
  - [モニタリング] タブ
    - このタブには 1 つのオプションしかなく、CloudWatchグループメトリクス収集 を有効または無効にできます。

コマンドラインを使用して Auto Scaling グループを更新するには

以下のコマンドのいずれかを使用できます。

- update-auto-scaling-group (AWS CLI)
- Update-AS AutoScalingグループ (AWS Tools for Windows PowerShell)

Auto Scaling インスタンスの更新

新しい起動テンプレートまたは起動設定を Auto Scaling グループに関連付けると、すべての新しいインスタンスに更新された設定が適用されます。既存のインスタンスは、最初に起動された構成で実行され続けます。既存のインスタンスに変更を適用するには、次のオプションがあります。

- 古いインスタンスを置き換えるためにインスタンスの更新を開始します。詳細については、「インスタンスの更新を使用して Auto Scaling グループのインスタンスを更新する」を参照してください。
- スケーリングアクティビティが、終了ポリシーに基づいて古いインスタンスを新しいインスタンスに徐々に置き換えるのを待ちます。
- 手動で終了させて Auto Scaling グループに置き換えます。

> **Note**
>
> 以下のインスタンス属性を起動テンプレートまたは起動設定の一部として指定することで変更できます。
>
> - [Amazon マシンイメージ (AMI)]
> - ブロックデバイス
> - キーペア
> - インスタンスタイプ

- セキュリティグループ
- ユーザーデータ
- モニタリング
- IAM インスタンスプロファイル
- プレースメントテナンシー
- kernel
- ラムディスク
- インスタンスにパブリック IP アドレスがあるかどうか

Auto Scaling グループとインスタンスにタグを付ける

タグは、 AWS リソースに割り当てる、または AWS に割り当てるカスタム属性ラベルです。各 タグは 2 つの部分で構成されます:

- タグキー (例: `costcenter`、`environment` または `project`)
- タグ値として知られるオプションのフィールド (例: `111122223333` または `production`)

タグは、以下のことに役立ちます。

- AWS コストを追跡します。 AWS Billing and Cost Management ダッシュボードでこれらのタグをアクティブ化します。 AWS はタグを使用してコストを分類し、毎月のコスト配分レポートを配信します。詳細については、AWS Billing ユーザーガイド の「コスト配分タグの使用」を参照してください。
- タグに基づいて、Auto Scaling グループへのアクセスを制御します。IAM ポリシーで条件を使用して、そのグループのタグに基づき、Auto Scaling グループへのアクセスを制御できます。詳しくは、「セキュリティ用のタグ」を参照してください。
- 追加したタグに基づく Auto Scaling グループのフィルタリングと検索。詳細については、「タグを使用して Auto Scaling グループをフィルタリングする」を参照してください。
- AWS リソースを特定して整理します。多くの はタグ付け AWS のサービス をサポートしているため、異なるサービスのリソースに同じタグを割り当てて、リソースが関連していることを示すことができます。

新規または既存の Auto Scaling グループにタグを付けることができます。Auto Scaling グループから、起動する EC2 インスタンスにタグを伝播することもできます。

タグは Amazon EBS ボリュームには伝達されません。Amazon EBS ボリュームにタグを追加するには、起動テンプレートでタグを指定します。詳細については、「Auto Scaling グループの起動テンプレートを作成する」を参照してください。

タグは、 AWS Management Console、 AWS CLI、または SDKs。

内容

- タグの命名と使用制限
- EC2 インスタンスのタグ付けライフサイクル
- Auto Scaling グループにタグを付ける
- タグの削除
- セキュリティ用のタグ
- タグへのアクセスを制御する
- タグを使用して Auto Scaling グループをフィルタリングする

タグの命名と使用制限

タグには以下のようなベーシック制限があります。

- リソースあたりのタグの最大数は 50 です。
- 単一の呼び出しを使用して追加または削除できるタグの最大数は 25 です。
- キーの最大長は Unicode 文字で 128 文字です。
- 値の最大長は Unicode 文字で 256 文字です。
- タグのキーと値は大文字と小文字が区別されます。ベストプラクティスとして、タグを大文字にするための戦略を決定し、その戦略をすべてのリソースタイプにわたって一貫して実装します。
- タグ名またはタグ値に aws: プレフィックスを使用しないでください。これは、 AWS 用に予約されているためです。このプレフィックスが含まれるタグの名前または値は編集または削除できません。これらはリソースクォータあたりのタグに対して計算されません。

EC2 インスタンスのタグ付けライフサイクル

EC2 インスタンスにタグを伝播することを選択した場合、タグは以下のように管理されます。

- Auto Scaling グループがインスタンスを起動すると、リソースの作成後ではなく作成中に、インスタンスにタグが追加されます。
- Auto Scaling グループでは、キーに `aws:autoscaling:groupName`、値に Auto Scaling グループ名が使用され、インスタンスにタグが自動的に追加されます。
- 起動テンプレートでインスタンスタグを指定し、グループのタグをそのインスタンスに伝播することを選択した場合、すべてのタグがマージされます。起動テンプレートのタグと Auto Scaling グループのタグに同じタグキーが指定されている場合、グループのタグ値が優先されます。
- 既存のインスタンスをアタッチするときに、Auto Scaling グループはタグをインスタンスに追加し、同じタグキーを持つ既存のタグを上書きします。また、キーが `aws:autoscaling:groupName` で、値が Auto Scaling グループ名のタグも追加されます。
- Auto Scaling グループからインスタンスからデタッチした場合は、`aws:autoscaling:groupName` タグのみが削除されます。

Auto Scaling グループにタグを付ける

Auto Scaling グループにタグを追加する際、Auto Scaling グループで起動するインスタンスに追加するかどうかを指定できます。タグを変更する場合は、変更後にその Auto Scaling グループで起動されたインスタンスには更新されたタグのバージョンが追加されます。Auto Scaling グループのタグを作成または変更しても、これらの変更内容は既に Auto Scaling グループで実行中のインスタンスには加えられません。

内容

- タグの追加または変更 (コンソール)
- タグの追加または変更 (AWS CLI)

タグの追加または変更 (コンソール)

Auto Scaling グループの作成時にタグを付けるには

Amazon EC2 コンソールを使用して Auto Scaling グループを作成する場合、Auto Scaling グループの作成ウィザードの [Add Tags (タグの追加)] ページでタグのキーと値を指定できます。Auto Scaling グループで起動されるインスタンスにタグを付けるには、[Tag New Instances (新しいインスタンスのタグ付け)] オプションが選択されたままにしてください。タグを付けない場合は、このオプションの選択を解除できます。

既存の Auto Scaling グループのタグを追加または変更するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   [Auto Scaling groups] (Auto Scaling グループ) ページの下部にスプリットペインが開きます。
3. [詳細] タブで、[タグ]、[編集] の順に選択します。
4. 既存のタグを変更するには、[Key] と [Value] フィールドを編集します。
5. 新しいタグを追加するには、[Add tag] を選択し、[Key] と [Value] フィールドを選択します。[Tag New Instances (新しいインスタンスにタグ付けする)] を選択したままにして Auto Scaling グループで起動されるインスタンスに自動的にタグを追加することも、選択解除して追加しないこともできます。
6. タグの追加を完了したら、[保存] を選択します

タグの追加または変更 (AWS CLI)

次の例は、 を使用して Auto Scaling グループを作成するときにタグ AWS CLI を追加し、既存の Auto Scaling グループのタグを追加または変更する方法を示しています。

Auto Scaling グループの作成時にタグを付けるには

create-auto-scaling-group を使用して新しい Auto Scaling グループを作成し、タグを追加します (例: **environment=production** を Auto Scaling グループに追加)。タグは、Auto Scaling グループで起動されるインスタンスにも追加されます。

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-asg \
  --launch-configuration-name my-launch-config --min-size 1 --max-size 3 \
  --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782" \
  --tags Key=environment,Value=production,PropagateAtLaunch=true
```

既存の Auto Scaling グループのタグを作成または変更するには

create-or-update-tags コマンドを使用して、タグを作成または変更します。例えば、以下のコマンドは **Name=my-asg** および **costcenter=cc123** タグを追加します。この変更後、それらのタグは Auto Scaling グループ内で起動されるすべてのインスタンスに追加されます。いずれかのキーを持つタグがすでに存在する場合、既存のタグは置き換えられます。Amazon EC2 コンソールでは、各インスタンスの表示名は、Name キーに指定された名前に関連付けられます (大文字と小文字は区別)。

```
aws autoscaling create-or-update-tags \
  --tags ResourceId=my-asg,ResourceType=auto-scaling-group,Key=Name,Value=my-
asg,PropagateAtLaunch=true \
  ResourceId=my-asg,ResourceType=auto-scaling-
group,Key=costcenter,Value=cc123,PropagateAtLaunch=true
```

Auto Scaling グループのタグを記述する (AWS CLI)

特定の Auto Scaling グループに適用されているタグを表示する場合は、次のいずれかのコマンドを使用できます。

- describe-tags – Auto Scaling グループ名を指定して、指定したグループのタグのリストを表示します。

  ```
  aws autoscaling describe-tags --filters Name=auto-scaling-group,Values=my-asg
  ```

  以下に、応答の例を示します。

  ```
  {
      "Tags": [
          {
              "ResourceType": "auto-scaling-group",
              "ResourceId": "my-asg",
              "PropagateAtLaunch": true,
              "Value": "production",
              "Key": "environment"
          }
      ]
  }
  ```

- describe-auto-scaling-groups – Auto Scaling グループ名を指定して、タグを含む指定されたグループの属性を表示します。

  ```
  aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
  ```

  以下に、応答の例を示します。

  ```
  {
      "AutoScalingGroups": [
  ```

```
        {
            "AutoScalingGroupName": "my-asg",
            "AutoScalingGroupARN": "arn",
            "LaunchTemplate": {
                "LaunchTemplateId": "lt-0b97f1e282EXAMPLE",
                "LaunchTemplateName": "my-launch-template",
                "Version": "$Latest"
            },
            "MinSize": 1,
            "MaxSize": 5,
            "DesiredCapacity": 1,
            ...
            "Tags": [
                {
                    "ResourceType": "auto-scaling-group",
                    "ResourceId": "my-asg",
                    "PropagateAtLaunch": true,
                    "Value": "production",
                    "Key": "environment"
                }
            ],
            ...
        }
    ]
}
```

タグの削除

Auto Scaling グループに関連付けられたタグは、いつでも削除できます。

内容

- タグの削除 (コンソール)
- タグの削除 (AWS CLI)

タグの削除 (コンソール)

タグを削除するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。

2. 既存のグループの横にあるチェックボックスをオンにします。

   [Auto Scaling groups] (Auto Scaling グループ) ページの下部にスプリットペインが開きます。
3. [詳細] タブで、[タグ]、[編集] の順に選択します。
4. タグの横にある [削除] を選択します。
5. [更新] を選択します。

タグの削除 (AWS CLI)

delete-tags コマンドを使用してタグを削除します。例えば、以下のコマンドは **environment** キーを持つタグを削除します。

```
aws autoscaling delete-tags --tags "ResourceId=my-asg,ResourceType=auto-scaling-
group,Key=environment"
```

タグのキーは指定する必要がありますが、値を指定する必要はありません。指定した値が正しくない場合、タグは削除されません。

セキュリティ用のタグ

タグを使用してリクエスター (IAM ユーザーまたはロールなど) が特定の Auto Scaling グループを作成、変更、削除する許可を持っていることを確認します。以下の条件キーを 1 つ以上使用して、IAM ポリシーの条件要素にタグ情報を指定します。

- 特定のタグを持つ Auto Scaling グループに対してユーザーアクションを許可 (または拒否) するには、`autoscaling:ResourceTag/`*`tag-key`*`: `*`tag-value` を使用します。
- リクエストに特定のタグが含める (または含めない) ことを要求するには、`aws:RequestTag/`*`tag-key`*`: `*`tag-value` を使用します。
- リクエストに特定のタグキーを含める (または含めない) ことを要求するには、`aws:TagKeys [`*`tag-key`*`, ...]` を使用します。

例えば、以下の例に示すように、キー **environment** および 値 **production** を持つタグを含むすべての Auto Scaling グループへのアクセスを拒否できます。

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Deny",
            "Action": [
                "autoscaling:CreateAutoScalingGroup",
                "autoscaling:UpdateAutoScalingGroup",
                "autoscaling:DeleteAutoScalingGroup"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {"autoscaling:ResourceTag/environment": "production"}
            }
        }
    ]
}
```

条件キーを使用して Auto Scaling グループへのアクセスを制御する方法の詳細については、「Amazon EC2 Auto Scaling と IAM の連携」を参照してください。

タグへのアクセスを制御する

タグを使用してリクエスター (IAM ユーザーまたはロールなど) が Auto Scaling グループのタグを追加、変更、削除する許可を持っていることを確認します。

次の IAM ポリシーの例では、Auto Scaling グループから **temporary** キーを持つタグのみを削除する許可をプリンシパルに与えます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "autoscaling:DeleteTags",
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": { "aws:TagKeys": ["temporary"] }
            }
        }
    ]
}
```

Auto Scaling グループに指定されたタグに制約を強制する IAM ポリシーの例については、「使用できるタグキーとタグ値を制御する」を参照してください。

> **Note**
>
> ユーザーによる Auto Scaling グループに対するタグ付け (または、タグの削除) 操作の実行を制限するポリシーを持っていても、ユーザーはインスタンスの起動後にタグを手動で変更できます。EC2 インスタンスのタグへのアクセスを制御する例については、Amazon EC2 ユーザーガイド」の「例: リソースのタグ付け」を参照してください。

タグを使用して Auto Scaling グループをフィルタリングする

次の例では、describe-auto-scaling-groups コマンドでフィルターを使用して、Auto Scaling グループを特定のタグで記述する方法を示しています。タグによるフィルタリングは、 AWS CLI または SDK に限定され、コンソールからは利用できません。

フィルタリングの考慮事項

- 1 つのリクエストで複数のフィルターと複数のフィルタの値を指定できます。
- フィルターの値にワイルドカードを使用することはできません。
- フィルターの値は大文字と小文字が区別されます。

例: 特定のタグキーと値のペアを持つ Auto Scaling グループを記述する

次のコマンドは、**environment=production** のタグキーと値のペアを持つ Auto Scaling グループのみを表示するように、結果をフィルタリングする方法を示しています。

```
aws autoscaling describe-auto-scaling-groups \
  --filters Name=tag-key,Values=environment Name=tag-value,Values=production
```

以下に、応答の例を示します。

```
{
    "AutoScalingGroups": [
        {
            "AutoScalingGroupName": "my-asg",
            "AutoScalingGroupARN": "arn",
            "LaunchTemplate": {
                "LaunchTemplateId": "lt-0b97f1e282EXAMPLE",
                "LaunchTemplateName": "my-launch-template",
                "Version": "$Latest"
```

```
            },
            "MinSize": 1,
            "MaxSize": 5,
            "DesiredCapacity": 1,
            ...
            "Tags": [
                {
                    "ResourceType": "auto-scaling-group",
                    "ResourceId": "my-asg",
                    "PropagateAtLaunch": true,
                    "Value": "production",
                    "Key": "environment"
                }
            ],
            ...
        },

    ... additional groups ...

    ]
}
```

または `tag:<key>` フィルターを使用して、タグを指定することもできます。例えば、次のコマンドは、**environment=production** のタグキーと値のペアを持つ Auto Scaling グループのみを表示するように、結果をフィルタリングする方法を示しています。このフィルターは、次の形式になります: `Name=tag:<key>,Values=<value>` (**<key>** および **<value>** はタグキーと値のペアを表します。)

```
aws autoscaling describe-auto-scaling-groups \
  --filters Name=tag:environment,Values=production
```

`--query` オプションを使用して AWS CLI 出力をフィルタリングすることもできます。次の例は、前のコマンドの AWS CLI 出力をグループ名、最小サイズ、最大サイズ、および希望する容量属性のみに制限する方法を示しています。

```
aws autoscaling describe-auto-scaling-groups \
  --filters Name=tag:environment,Values=production \
  --query "AutoScalingGroups[].{AutoScalingGroupName: AutoScalingGroupName, MinSize:
 MinSize, MaxSize: MaxSize, DesiredCapacity: DesiredCapacity}"
```

以下に、応答の例を示します。

```
[
    {
        "AutoScalingGroupName": "my-asg",
        "MinSize": 0,
        "MaxSize": 10,
        "DesiredCapacity": 1
    },

    ... additional groups ...

]
```

フィルタリングの詳細については、「 AWS Command Line Interface ユーザーガイド」のAWS CLI「出力のフィルタリング」を参照してください。

例: 指定したタグキーと一致するタグを持つ Auto Scaling グループを記述する

次のコマンドは、タグ値に関係なく **environment** タグを持つ Auto Scaling グループのみを表示するように、結果をフィルタリングする方法を示しています。

```
aws autoscaling describe-auto-scaling-groups \
  --filters Name=tag-key,Values=environment
```

例: 指定したタグキーのセットに一致するタグを持つ Auto Scaling グループを記述する

次のコマンドは、タグ値に関係なく **environment** および **project** のタグを持つ Auto Scaling グループのみを表示するように、結果をフィルタリングする方法を示しています。

```
aws autoscaling describe-auto-scaling-groups \
  --filters Name=tag-key,Values=environment Name=tag-key,Values=project
```

例: 指定したタグキーのうち少なくとも 1 つに一致するタグを持つ Auto Scaling グループを記述する

次のコマンドは、タグ値に関係なく **environment** または **project** のタグを持つ Auto Scaling グループのみを表示するように、結果をフィルタリングする方法を示しています。

```
aws autoscaling describe-auto-scaling-groups \
  --filters Name=tag-key,Values=environment,project
```

例: 指定したタグ値を持つ Auto Scaling グループを記述する

次のコマンドは、タグキーに関係なくタグ値 **production** を持つ Auto Scaling グループのみを表示するように、結果をフィルタリングする方法を示しています。

```
aws autoscaling describe-auto-scaling-groups \
  --filters Name=tag-value,Values=production
```

例: 指定したタグ値のセットを持つ Auto Scaling グループを記述する

次のコマンドは、タグキーに関係なくタグ値 **production** および **development** を持つ Auto Scaling グループのみを表示するように、結果をフィルタリングする方法を示しています。

```
aws autoscaling describe-auto-scaling-groups \
  --filters Name=tag-value,Values=production Name=tag-value,Values=development
```

例: 指定したタグ値の少なくとも 1 つに一致するタグを持つ Auto Scaling グループを記述する

次のコマンドは、タグキーに関係なくタグ値 **production** または **development** を持つ Auto Scaling グループのみを表示するように、結果をフィルタリングする方法を示しています。

```
aws autoscaling describe-auto-scaling-groups \
  --filters Name=tag-value,Values=production,development
```

例: 複数のタグキーおよびタグ値が一致するタグを持つ Auto Scaling グループを記述する

フィルターを組み合わせてカスタムの AND および OR ロジックを作成し、より複雑なフィルタリングを実行することもできます。

次のコマンドは、特定のタグのセットを持つ Auto Scaling グループのみを表示するように、結果をフィルタリングする方法を示しています。一方のタグキーは **environment** AND でタグ値は (**production** OR **development**) AND、もう一方のタグキーは **costcenter** AND でタグ値は **cc123** です。

```
aws autoscaling describe-auto-scaling-groups \
  --filters Name=tag:environment,Values=production,development
 Name=tag:costcenter,Values=cc123
```

インスタンスのメンテナンスポリシー

インスタンスの更新やヘルスチェックプロセスなど、インスタンスが置き換えられるイベント中に特定の容量要件を満たすように Auto Scaling グループのインスタンスメンテナンスポリシーを設定できます。

例えば、インスタンス数が少ない Auto Scaling グループがあるとします。ヘルスチェックでインスタンスの障害が示された場合に、インスタンスを終了して置き換えることによる潜在的な中断を回避したいと考えています。インスタンスメンテナンスポリシーを使用すると、Amazon EC2 Auto Scaling が最初に新しいインスタンスを起動し、そのインスタンスが完全に準備できるようになるのを待ってから、異常なインスタンスを終了できます。

インスタンスメンテナンスポリシーは、複数のインスタンスが同時に置き換えられる場合に発生する可能性のある中断を最小限に抑えるのにも役立ちます。ポリシーの最小および最大正常率パラメータを設定すると、Auto Scaling グループはインスタンスを置き換えるときに、その最小/最大範囲内でのみ容量を増減できます。範囲を大きくすると、同時に置き換えることができるインスタンスの数が増えます。

内容

- インスタンスメンテナンスポリシーの概要
- Auto Scaling グループにインスタンスメンテナンスポリシーを設定する

インスタンスメンテナンスポリシーの概要

このトピックでは、使用可能なオプションの概要と、インスタンスメンテナンスポリシーを作成するときに考慮すべき事項について説明します。

内容

- 概要
- 重要な概念
- インスタンスのウォームアップ
- ヘルスチェックの猶予期間
- Auto Scaling グループのスケーリング
- シナリオ例

概要

Auto Scaling グループのインスタンスメンテナンスポリシーを作成すると、そのポリシーは Amazon EC2 Auto Scaling イベントに影響し、インスタンスが置き換えられます。これにより、同じ Auto Scaling グループ内でより一貫した置換動作が得られます。また、必要に応じて可用性やコストに合わせてグループを最適化することもできます。

コンソールでは、次の設定オプションを使用できます。

- 終了前に起動 — 既存のインスタンスを終了する前に、新しいインスタンスを最初にプロビジョニングする必要があります。このアプローチは、コスト削減よりも可用性を優先するアプリケーションに適しています。
- 終了と起動 — 新しいインスタンスは、既存のインスタンスが終了すると同時にプロビジョニングされます。このアプローチは、可用性よりもコスト削減を優先するアプリケーションに適しています。また、インスタンスを置き換える場合でも、現在利用可能な容量よりも多くの容量を起動すべきではないアプリケーションにも適しています。
- カスタムポリシー — このオプションを使用すると、インスタンスを置き換えるときに使用可能な容量のカスタム最小範囲と最大範囲を使用してポリシーを設定できます。このアプローチは、コストと可用性の適切なバランスを実現するのに役立ちます。

Auto Scaling グループのデフォルトは、インスタンスメンテナンスポリシーがないことです。これにより、インスタンスメンテナンスイベントにデフォルトの動作で応答します。デフォルトの動作については、次の表で説明します。

インスタンスメンテナンスイベントのデフォルト動作

イベント	説明	デフォルトの動作
ヘルスチェックの失敗	インスタンスがヘルスチェックに失敗すると自動的に発生します。Amazon EC2 Auto Scaling は、ヘルスチェックに失敗したインスタンスを置き換えます。ヘルスチェックの失敗の原因については、「」を参照してくださいAuto Scaling グループ内のインスタンスのヘルスチェック。	を終了して起動します。

イベント	説明	デフォルトの動作
インスタンスの更新	インスタンスの更新を開始すると発生します。設定に応じて、インスタンスの更新により、インスタンスは一度に 1 つずつ、複数のインスタンスを一度に、またはすべてを一度に置き換えます。詳細については、「インスタンスの更新を使用して Auto Scaling グループのインスタンスを更新する」を参照してください。	を終了して起動します。
最大インスタンス有効期間	インスタンスが Auto Scaling グループに指定した最大インスタンスライフタイムに達すると自動的に発生します。Amazon EC2 Auto Scaling は、インスタンスの最大有効期間に達したインスタンスを置き換えます。詳細については、「インスタンスの最大存続期間に基づいて Auto Scaling インスタンスを置き換える」を参照してください。	を終了して起動します。

イベント	説明	デフォルトの動作
リバランシング	グループのバランスが崩れる原因となる根本的な変更がある場合に自動的に発生します。Amazon EC2 Auto Scaling は、以下の状況でグループのバランスを再調整します。 • 以前に容量が不足していたアベイラビリティーゾーンが回復するか、アベイラビリティーゾーンをグループに追加または削除します。この場合、Auto Scaling グループは、アベイラビリティーゾーン間で均等にバランスを取るように試みます。詳細については、「アクティビティの再分散」を参照してください。 • Auto Scaling グループでキャパシティの再調整を有効にし、スポットインスタンスの可用性が変化すると既存のスポットインスタンスが中断される前に、新しいスポットインスタンスを起動しようとします。詳細については、「キャパシティの再調整を使用して Amazon EC2 スポットの中断に対処する」を参照してください。	終了する前に を起動します。 Amazon EC2 Auto Scaling は、グループのサイズ制限を最大容量 の最大 10% 超えることができます。ただし、キャパシティーの再調整を使用している場合、これらの制限を超えることができるのは、希望するキャパシティー の最大 10% のみです。

イベント	説明	デフォルトの動作
	• Auto Scaling グループを更新すると、混合インスタンスポリシーの更新時に選択した新しい購入オプションに合わせてインスタンスが徐々に置き換えられます。詳細については、「Auto Scaling グループを更新する」を参照してください。	

Amazon EC2 Auto Scaling は、以下の状況でも引き続きデフォルトで終了および起動されます。したがって、これらの状況のいずれかが発生すると、グループの容量がインスタンスメンテナンスポリシーの下限しきい値を下回る可能性があります。

- 例えば、人間のアクションによってインスタンスが予期せず終了した場合。Amazon EC2 Auto Scaling は、実行されなくなったインスタンスを直ちに置き換えます。詳細については、「Amazon EC2 ヘルスチェック」を参照してください。
- Amazon EC2 Auto Scaling が代替インスタンスを起動する前に、スケジュールされたイベントの一部として Amazon EC2 がインスタンスを再起動、停止、または廃止する場合。これらのイベントの詳細については、「Amazon EC2 ユーザーガイド」の「インスタンスのスケジュールされたイベント」を参照してください。 Amazon EC2
- Amazon EC2 スポットサービスがスポットインスタンスの中断を開始し、スポットインスタンスが強制終了されると、

スポットインスタンスでは、Auto Scaling グループでキャパシティの再調整を有効にした場合、インスタンスには、スポット中断を開始する前に起動した別のスポットプールから保留中のインスタンスが既に存在する可能性があります。容量の再調整の仕組みの詳細については、「」を参照してくださいキャパシティの再調整を使用して Amazon EC2 スポットの中断に対処する。

ただし、スポットインスタンスは引き続き使用可能である保証はなく、2 分間のスポットインスタンス中断通知で終了できるため、新しいインスタンスが起動する前にインスタンスが中断された場合、インスタンスメンテナンスポリシーのしきい値を低く超える可能性があります。

重要な概念

開始する前に、以下の主要な概念と用語を理解してください。

希望する容量

希望する容量は、作成時の Auto Scaling グループの容量です。また、グループにスケーリング条件がアタッチされていない場合に、グループが維持しようとする容量でもあります。

インスタンスメンテナンスポリシー

インスタンスメンテナンスポリシーは、インスタンスのメンテナンスイベントで既存のインスタンスが終了する前に、インスタンスを最初にプロビジョニングするかどうかを制御します。また、Auto Scaling グループが同時に複数のインスタンスを置き換えるために、希望する容量をどれだけ下回るか、超えるかも決定します。

最大正常率

最大正常率は、インスタンスを置き換えるときに Auto Scaling グループが増やすことができる希望する容量の割合です。ワークロードをサポートするために、稼働中および正常または保留中のグループの最大パーセンテージを表します。コンソールでは、Launch before terminating オプションまたは Custom policy オプションのいずれかを使用すると、最大正常率を設定できます。有効な値は 100～200% です。

最小正常率

最小正常率は、インスタンスを置き換えるときにワークロードをサポートするために、稼働中、正常、準備が整った状態を維持するのに必要な容量の割合です。インスタンスは、最初のヘルスチェックが正常に完了し、指定されたウォームアップ時間が経過すると、正常と見なされ、使用できる状態になります。コンソールでは、Terminate and launch オプションまたは Custom policy オプションのいずれかを使用すると、最小正常率を設定できます。有効な値は 0～100% です。

> **Note**
>
> インスタンスをより迅速に置き換えるには、最小正常率を低く指定できます。ただし、正常なインスタンスが十分に実行されていない場合、可用性が低下する可能性があります。複数のインスタンスが置き換えられる状況でも可用性を維持するために、妥当な値を選択することをお勧めします。

インスタンスのウォームアップ

インスタンスが `InService`状態になった後に初期化する時間が必要な場合は、Auto Scaling グループのデフォルトのインスタンスウォームアップを有効にします。デフォルトのインスタンスウォームアップでは、インスタンスが準備が完了する前に最小の正常な割合にカウントされないようにできます。これにより、Amazon EC2 Auto Scaling は、既存のインスタンスを終了する前に、ワークロードをサポートするのに十分なキャパシティーがあるまでの時間を考慮します。

追加の利点として、インスタンスのデフォルトウォームアップを有効にすると、動的スケーリングに使用される Amazon CloudWatch メトリクスを改善できます。Auto Scaling グループにスケーリングポリシーがある場合、グループがスケールアウトすると、初期化が完了する前にインスタンスが CloudWatch メトリクスにカウントされないように、同じデフォルトのウォームアップ期間が使用されます。

詳細については、「Auto Scaling グループに対するインスタンスのデフォルトウォームアップを設定する」を参照してください。

ヘルスチェックの猶予期間

Amazon EC2 Auto Scaling は、Auto Scaling グループが使用するヘルスチェックのステータスに基づいてインスタンスが正常かどうかを判断します。詳細については、「Auto Scaling グループ内のインスタンスのヘルスチェック」を参照してください。

これらのヘルスチェックをできるだけ早く開始するには、グループのヘルスチェック猶予期間をあまり長く設定しないでください。ただし、Elastic Load Balancing ヘルスチェックがターゲットがリクエストを処理できるかどうかを判断するのに十分な長さに設定します。詳細については、「Auto Scaling グループにヘルスチェックの猶予期間を設定する」を参照してください。

Auto Scaling グループのスケーリング

インスタンスメンテナンスポリシーは、インスタンスメンテナンスイベントにのみ適用され、グループが手動または自動的にスケーリングされるのを防ぐものではありません。

Auto Scaling グループにスケーリングポリシーまたはスケジュールされたアクションがアタッチされている場合、インスタンスのメンテナンスイベントの発生中に並行して実行できます。その場合、グループの希望する容量を増減できますが、定義したスケーリング制限内に限られます。これらの制限の詳細については、「」を参照してくださいAuto Scaling グループのスケーリング制限を設定する。

シナリオ例

一般的なシナリオでは、インスタンスのメンテナンスポリシーと希望する容量は次のようになります。

- 最小正常率 = 90%
- 最大正常率 = 120%
- 希望する容量 = 100

インスタンスのメンテナンスイベント中、Auto Scaling グループのインスタンス数は最大 90 個、最大 120 個になる場合があります。イベント後、グループは 100 個のインスタンスを持つに戻ります。

ウォームプールを持つ Auto Scaling グループでインスタンスメンテナンスポリシーを使用する場合、最小および最大正常率は Auto Scaling グループとウォームプールに個別に適用されます。

例えば、次の設定があるとします。

- 最小正常率 = 90%
- 最大正常率 = 120%
- 希望する容量 = 100
- ウォームプールサイズ = 10

インスタンスの更新を開始してグループのインスタンスをリサイクルする場合、Amazon EC2 Auto Scaling は最初に Auto Scaling グループのインスタンスを置き換え、次にウォームプール内のインスタンスを置き換えます。Amazon EC2 Auto Scaling が Auto Scaling グループ内のインスタンスの置き換えにまだ取り組んでいる間、グループには最大 90 個のインスタンスと最大 120 個のインスタンスが含まれる場合があります。グループで終了すると、Amazon EC2 Auto Scaling はウォームプール内のインスタンスの置き換えを処理できます。この間、ウォームプールには 9 つのインスタンスと 12 のインスタンスしか存在しない場合があります。

Auto Scaling グループにインスタンスメンテナンスポリシーを設定する

Auto Scaling グループを作成するときに、インスタンスメンテナンスポリシーを作成できます。既存のグループ用に作成することもできます。

Auto Scaling グループにインスタンスメンテナンスポリシーを設定することで、インスタンスメンテナンスポリシーを上書きしない限り、インスタンス更新機能の最小および最大正常率パラメータの値を指定する必要がなくなります。

コンソールでは、Amazon EC2 Auto Scaling が使用開始に役立つオプションを提供します。

内容

- インスタンスのメンテナンスポリシーを設定する
- インスタンスのメンテナンスポリシーを削除する

インスタンスのメンテナンスポリシーを設定する

Auto Scaling グループにインスタンスメンテナンスポリシーを設定するには、次のいずれかの方法を使用します。

Console

新しいグループにインスタンスメンテナンスポリシーを設定するには (コンソール)

1. の手順に従って起動テンプレートを使用して Auto Scaling グループを作成する、ステップ 11 までの手順の各ステップを完了します。
2. グループサイズとスケーリングポリシーの設定 で、希望する容量 に、起動するインスタンスの初期数を入力します。
3. スケーリングセクションのスケーリング制限 で、希望する容量の新しい値が希望する最小容量と希望する最大容量 より大きい場合、希望する最大容量は自動的に希望する新しい容量値に増加します。これらの制限は必要に応じて変更できます。
4. 自動スケーリング では、ターゲット追跡スケーリングポリシーを作成するかどうかを選択します。このポリシーは、Auto Scaling グループの作成後に作成することもできます。

   ターゲット追跡スケーリングポリシー を選択した場合は、「」の指示に従ってポリシーターゲット追跡スケーリングポリシーを作成するを作成します。
5. インスタンスメンテナンスポリシーセクションで、使用可能なオプションのいずれかを選択します。

   - 終了前に起動： 既存のインスタンスを終了する前に、新しいインスタンスを最初にプロビジョニングする必要があります。これは、コスト削減よりも可用性を優先するアプリケーションに適しています。

- の終了と起動: 新しいインスタンスは、既存のインスタンスが終了すると同時にプロビジョニングされます。これは、可用性よりもコスト削減を優先するアプリケーションに適しています。また、現在利用可能な容量よりも多くの容量を起動すべきではないアプリケーションにも適しています。
- カスタムポリシー : このオプションを使用すると、インスタンスを置き換えるときに使用可能な容量のカスタム最小範囲と最大範囲を使用してポリシーを設定できます。これにより、コストと可用性の適切なバランスを実現できます。

6. 正常なパーセンテージを設定する には、次のフィールドの一方または両方に値を入力します。有効なフィールドは、前のステップで選択したオプションによって異なります。

   - 最小 : インスタンスの置き換えに進むために必要な最小正常率を設定します。
   - 最大 : インスタンスを置き換えるときに可能な最大正常率を設定します。

7. 希望するキャパシティーセクションに基づいて置き換え中にキャパシティーを表示 セクションを展開し、最小と最大 の値がグループにどのように適用されるかを確認します。使用される正確な値は、希望する容量値によって異なります。これは、グループがスケールすると変化します。
8. 起動テンプレートを使用して Auto Scaling グループを作成する のステップを続行します。

AWS CLI

新しいグループにインスタンスメンテナンスポリシーを設定するには (AWS CLI)

create-auto-scaling-group コマンドに `--instance-maintenance-policy` オプションを追加します。次の例では、 という名前の新しい Auto Scaling グループにインスタンスメンテナンスポリシーを設定します*my-asg*。

```
aws autoscaling create-auto-scaling-group \
  --launch-template LaunchTemplateName=my-launch-template,Version='1' \
  --auto-scaling-group-name my-asg \
  --min-size 1 \
  --max-size 10 \
  --desired-capacity 5 \
  --default-instance-warmup 20 \
  --instance-maintenance-policy '{
      "MinHealthyPercentage": 90,
      "MaxHealthyPercentage": 120
    }' \
  --vpc-zone-identifier "subnet-5e6example,subnet-613example,subnet-c93example"
```

Console

既存のグループにインスタンスメンテナンスポリシーを設定するには (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. 画面の上部のナビゲーションバーで、Auto Scaling グループを作した AWS リージョン を選択します。
3. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。
4. 詳細 タブで、インスタンスメンテナンスポリシー 、編集 を選択します。
5. グループにインスタンスメンテナンスポリシーを設定するには、使用可能なオプションのいずれかを選択します。

   - 終了前に起動： 既存のインスタンスを終了する前に、新しいインスタンスを最初にプロビジョニングする必要があります。これは、コスト削減よりも可用性を優先するアプリケーションに適しています。
   - の終了と起動: 新しいインスタンスは、既存のインスタンスが終了すると同時にプロビジョニングされます。これは、可用性よりもコスト削減を優先するアプリケーションに適しています。また、現在利用可能な容量よりも多くの容量を起動すべきではないアプリケーションにも適しています。
   - カスタムポリシー : このオプションを使用すると、インスタンスを置き換えるときに使用可能な容量のカスタム最小範囲と最大範囲を使用してポリシーを設定できます。これにより、コストと可用性の適切なバランスを実現できます。
6. 正常率の設定 では、次のフィールドの一方または両方に値を入力します。有効なフィールドは、前のステップで選択したオプションによって異なります。

   - 最小 : インスタンスの置き換えに進むために必要な最小正常率を設定します。
   - 最大 : インスタンスを置き換えるときに可能な最大正常率を設定します。
7. 希望するキャパシティーセクションに基づいて置き換え中にキャパシティーを表示 セクションを展開し、最小と最大 の値がグループにどのように適用されるかを確認します。使用される正確な値は、希望する容量値によって異なります。これは、グループがスケールすると変化します。
8. [更新] を選択します。

AWS CLI

既存のグループにインスタンスメンテナンスポリシーを設定するには (AWS CLI)

update-auto-scaling-group コマンドに `--instance-maintenance-policy` オプションを追加します。次の例では、指定された Auto Scaling グループにインスタンスメンテナンスポリシーを設定します。

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --instance-maintenance-policy '{
      "MinHealthyPercentage": 90,
      "MaxHealthyPercentage": 120
    }'
```

インスタンスのメンテナンスポリシーを削除する

Auto Scaling グループでインスタンスメンテナンスポリシーの使用を停止する場合は、削除できます。

Console

インスタンスメンテナンスポリシーを削除するには (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. 画面の上部のナビゲーションバーで、Auto Scaling グループを作した AWS リージョン を選択します。
3. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。
4. 詳細 タブで、インスタンスメンテナンスポリシー 、編集 を選択します。
5. インスタンスメンテナンスポリシーなし を選択します。
6. [更新] を選択します。

AWS CLI

インスタンスメンテナンスポリシーを削除するには (AWS CLI)

update-auto-scaling-group コマンドに `--instance-maintenance-policy` オプションを追加します。次の例では、指定された Auto Scaling グループからインスタンスメンテナンスポリシーを削除します。

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --instance-maintenance-policy '{
      "MinHealthyPercentage": -1,
      "MaxHealthyPercentage": -1
    }'
```

Amazon EC2 Auto Scaling のライフサイクルフック

Amazon EC2 Auto Scaling は、Auto Scaling グループにライフサイクルフックを追加する機能を提供します。これらのフックにより、Auto Scaling インスタンスライフサイクルのイベントを認識し、対応するライフサイクルイベントが発生したときにカスタムアクションを実行するソリューションを作成できます。ライフサイクルフックでは、インスタンスが次の状態に移行する前に、ライフサイクルアクションの完了を待つ時間 (デフォルトでは 1 時間) が指定されます。

Auto Scaling インスタンスでライフサイクルフックを使用する例として、以下を実行します。

- スケールアウト イベントが発生すると、新しく起動したインスタンスはスタートアップ シーケンスを完了し、待機状態に移行します。インスタンスが待機状態の間に、アプリケーションに必要なソフトウェアパッケージをダウンロードしてインストールするスクリプトを実行して、トラフィックの受信をスタートする前に、インスタンスの準備がすべて完了していることを確認できます。スクリプトがソフトウェアのインストールを終了すると、complete-lifecycle-actionコマンドを実行して続行します。
- スケールインイベントが発生すると、ライフサイクルフックはインスタンスが終了する前に一時停止し、Amazon を使用して通知を送信します EventBridge。インスタンスが待機状態にある間は、AWS Lambda 関数を呼び出したり、インスタンスに接続してログやその他のデータをダウンロードしたりしてから、インスタンスが完全に終了することができます。

ライフサイクルフックの一般的な使用方法は、インスタンスが Elastic Load Balancing に登録されるタイミングを制御することです。Auto Scaling グループに起動ライフサイクルフックを追加すると、ライフサイクルフックの最後にロードバランサーに登録される前に、ブートストラップスクリプトが正常に完了していて、インスタンス上のアプリケーションがトラフィックを受け入れる準備ができていることを確認できます。

内容

- ライフサイクルフックの可用性
- ライフサイクルフックの考慮事項と制限
- 関連リソース
- ライフサイクルフックの動作
- ライフサイクルフックを Auto Scaling グループに追加するための準備
- インスタンスメタデータを使用してターゲットライフサイクル状態を取得する
- ライフサイクルフックを追加する
- ライフサイクルアクションを完了する
- チュートリアル: インスタンスメタデータを使用してターゲットライフサイクル状態を取得するようにユーザーデータを設定する
- チュートリアル:Lambda 関数を呼び出すライフサイクルフックの設定

ライフサイクルフックの可用性

次の表は、さまざまなシナリオで利用できるライフサイクルフックを示しています。

イベント	インスタンスの起動または終了[1]	インスタンスの最大存続期間: 置き換えインスタンス	インスタンスの更新: 置き換えインスタンス	キャパシティーの再調整: 置き換えインスタンス	ウォームプール: ウォームプールに出入りするインスタンス
インスタンスの起動	✓	✓	✓	✓	✓
インスタンスの削除	✓	✓	✓	✓	✓

[1] 自動的に開始されたか、または手動で開始されたかにかかわらず (SetDesiredCapacity もしくは TerminateInstanceInAutoScalingGroup オペレーションを呼び出す場合など)、すべての起動と終了に適用されます。インスタンスのアタッチまたはデタッチ、インスタンスのスタンバイモードへの切り替え、または強制削除オプションを使用したグループの削除には適用されません。

ライフサイクルフックの考慮事項と制限

ライフサイクルフックを使用する場合は、以下の注意事項と制限事項に留意してください。

- Amazon EC2 Auto Scaling は、Auto Scaling グループの管理に役立つ独自のライフサイクルを提供します。このライフサイクルは、他の EC2 インスタンスのライフサイクルとは異なります。詳細については、「Amazon EC2 Auto Scaling インスタンスのライフサイクル」を参照してください。ウォームプール内のインスタンスには、ウォームプール内のインスタンスのライフサイクル状態の移行 で説明されている独自のライフサイクルもあります。
- スポットインスタンスによりライフサイクルフックを使用できますが、使用可能なキャパシティがなくなった場合、ライフサイクルフックはインスタンスの終了を防ぐことができません。これは、2 分間の中断通知により、いつでも起こり得ます。詳細については、『Amazon EC2 ユーザーガイド』の「スポットインスタンス中断」を参照してください。ただし、キャパシティの再調整を有効にして、Amazon EC2 スポットサービスから再調整のレコメンデーションを受け取ったスポットインスタンスを積極的に置換することができます。これは、スポットインスタンスの中断リスクが高まったときに送信される信号です。詳細については、「キャパシティの再調整を使用して Amazon EC2 スポットの中断に対処する」を参照してください。
- インスタンスは一定期間、待機状態に維持できます。ライフサイクルフックのデフォルトタイムアウトは 1 時間です (ハートビートタイムアウト)。インスタンスを待機状態に保つことができる最大時間を指定するグローバルタイムアウトもあります。グローバルタイムアウトは、48 時間か、ハートビートタイムアウトの 100 倍の時間のどちらか短い方になります。
- ライフサイクルフックの結果は、中止または続行のどちらかになります。インスタンスが起動中の場合、「続行」はアクションが正常に実行され、Amazon EC2 Auto Scaling がインスタンスをサービス開始できることを示します。それ以外の場合、中止はカスタムアクションが失敗し、インスタンスを終了して置き換えることができることを示します。インスタンスが終了する場合、「中止」と「続行」の両方でインスタンスを終了できます。ただし、「中止」は他のライフサイクルフックなど残りのアクションをすべて停止し、「続行」は他のライフサイクルフックを完了することを許可します。
- ライフサイクルフックが繰り返し失敗する場合、Amazon EC2 Auto Scaling はインスタンスの起動を許可する頻度を制限するので、ライフサイクルアクションをテストして永続的なエラーを修正するようにしてください。
- AWS CLI、、または SDK を使用してライフサイクルフックを作成および更新すると AWS CloudFormation、 からライフサイクルフックを作成するときに使用できないオプションが提供されます AWS Management Console。例えば、SNS トピックまたは SQS キューの ARN を指定する フィールドはコンソールに表示されません。これは、Amazon EC2 Auto Scaling が既にイベントを Amazon に送信しているためです EventBridge。これらのイベントはフィルタリングされ、必

要に応じて Lambda、Amazon SNS、Amazon SQS などの AWS サービスにリダイレクトできます。

- 、、 AWS CLI AWS CloudFormationまたは SDK を使用して CreateAutoScalingGroup API を呼び出すことで、Auto Scaling グループの作成中に複数のライフサイクルフックを追加できます。ただし、各フックの通知ターゲットと IAM ロール (指定されている場合) が同じである必要があります。異なる通知ターゲットと異なるロールを持つライフサイクルフックを作成するには、Hook API への個別の呼び出しでライフサイクルフックを一度に 1 PutLifecycleつずつ作成します。
- インスタンス起動用のライフサイクルフックを追加すると、ヘルスチェックの猶予期間は、インスタンスが `InService` 状態に到達するとすぐに開始されます。詳細については、「Auto Scaling グループにヘルスチェックの猶予期間を設定する」を参照してください。

スケーリングに関する考慮事項

- 動的スケーリングポリシーは、複数のインスタンスに集約された CPU やネットワーク I/O などの CloudWatch メトリクスデータに応じてスケールインおよびスケールアウトします。スケールアウトしても、Auto Scaling グループで集計するインスタンスメトリックスに、Amazon EC2 Auto Scaling が新しいインスタンスを追加するわけではありません。インスタンスが `InService` 状態に到達し、ウォームアップが完了するまで待機します。詳細については、デフォルトインスタンスのウォームアップのトピックの「パフォーマンスのスケーリングに関する考慮事項」を参照してください。
- スケールインでは、終了するインスタンスの削除が集約インスタンスメトリックスに直ちに反映されない場合があります。Amazon EC2 Auto Scaling の終了ワークフローが開始すると、終了するインスタンスは、グループの集計インスタンスメトリクスへのカウントを停止します。
- ほとんどの場合、ライフサイクルフックが呼び出されると、簡易スケーリングポリシーに起因するスケーリングアクティビティは、ライフサイクルアクションが完了し、クールダウン期間が終了するまで一時停止されます。クールダウン期間に長い間隔を設定すると、スケーリングが再開されるまでに時間がかかることになります。詳細については、クールダウントピックの「追加の遅延を発生させる可能性のあるライフサイクルフック」を参照してください。一般的に、ステップスケーリングポリシーまたはターゲット追跡スケーリングポリシーを使用できる場合には、簡易スケーリングポリシーを使わないことをおすすめします。

関連リソース

紹介ビデオについては、AWS 「 re:Invent 2018: Capacity Management Made Easy with Amazon EC2 Auto Scaling on 」を参照してくださいYouTube。

AWS CloudFormation スタックテンプレートでライフサイクルフックを宣言する方法を理解するために使用できる JSON および YAML テンプレートスニペットがいくつか用意されています。詳細については、「 ユーザーガイド」の「 AWS::AutoScaling::LifecycleHookリファレンス」を参照してください。 AWS CloudFormation

GitHub リポジトリにアクセスして、ライフサイクルフックのサンプルテンプレートとユーザーデータスクリプトをダウンロードすることもできます。

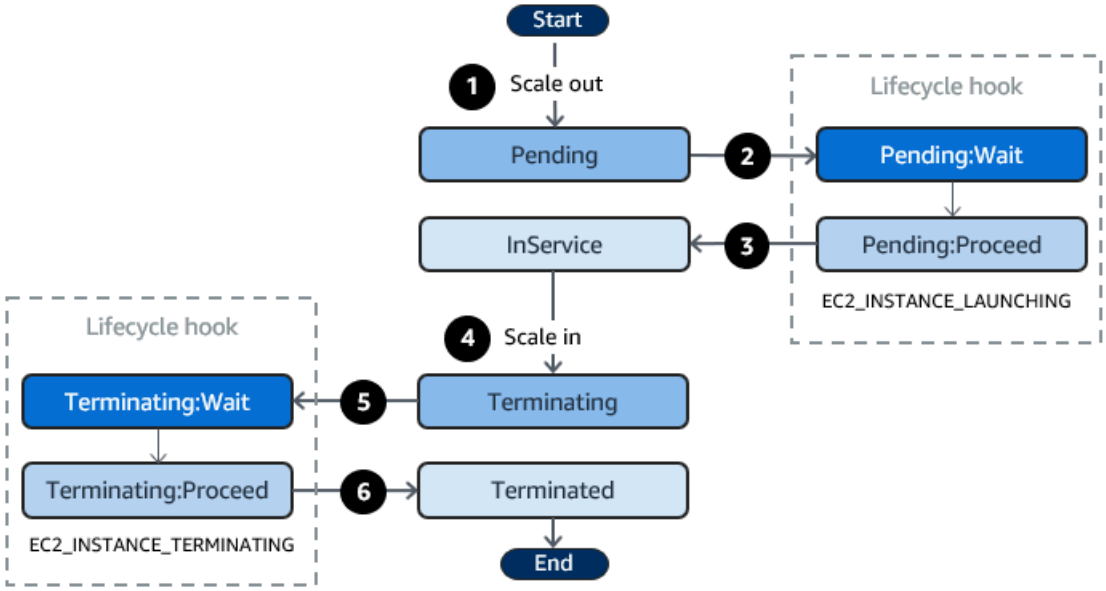
ライフサイクルフックの使用例については、次のブログ記事を参照してください。

- Lambda と Amazon EC2 Run コマンドを使用したスケーリングインスタンスのバックアップシステムの構築
- EC2 Auto Scaling インスタンスを終了する前にコードを実行します。

ライフサイクルフックの動作

Amazon EC2 インスタンスは、起動から終了まで、さまざまな状態に移行します。Auto Scaling グループ用にカスタムアクションを作成して、ライフサイクルフックによってインスタンスが待機状態へ移行しているときに、動作させることができます。

次の図は、スケールアウトとスケールインにライフサイクルフックを使用する場合の Auto Scaling インスタンスの状態の遷移を示しています。



(前の図には示されているように)。

1. Auto Scaling グループはスケールアウト イベントに応答し、インスタンスの起動を開始します。

2. ライフサイクルフックはインスタンスを待機状態 (Pending:Wait) にし、カスタムアクションを実行します。

   インスタンスは、ライフサイクルアクションが完了するまで、またはタイムアウト期間が終了するまで待機状態のままになります。デフォルトでは、インスタンスは 1 時間にわたり待機状態になった後、Auto Scaling グループは起動処理を継続します(Pending:Proceed)。さらに時間が必要な場合は、ハートビートを記録することにより、タイムアウト期間を再開できます。カスタムアクションは完了したが、タイムアウト期間はまだ終了していないという場合にライフサイクルアクションを完了すると、タイムアウト期間が終了し、Auto Scaling グループは起動プロセスを続行します。

3. インスタンスはInService状態になり、ヘルスチェックの猶予期間がスタートします。ただし、Auto Scaling グループが Elastic Load Balancing ロードバランサーに関連付けられている場合、インスタンスが InService 状態に到達する前にインスタンスはロードバランサーに登録され、ロードバランサーはそのヘルスチェックを開始します。ヘルスチェックの猶予期間が終了すると、Amazon EC2 Auto Scaling はインスタンスのヘルス状態のチェックを開始します。

4. Auto Scaling グループはスケールイン イベントに応答し、インスタンスの終了を開始します。Auto Scaling グループが Elastic Load Balancing で使用されている場合、まず、終了するインスタンスがロードバランサーから登録解除されます。ロードバランサーの Connection Draining が有効になっている場合、インスタンスは新しい接続の受け入れを停止し、既存の接続のドレインを待ってから登録解除プロセスを完了します。

5. ライフサイクルフックはインスタンスを待機状態 (Terminating:Wait) にし、カスタムアクションを実行します。

   インスタンスは、ライフサイクルアクションを完了するまで、またはタイムアウト期間が終了するまで (デフォルトでは 1 時間)、待機状態のままになります。ライフサイクルフックを完了するか、タイムアウト期間が終了すると、インスタンスは次の状態 (Terminating:Proceed)に移行します。

6. インスタンスが終了しました。

> **Important**
>
> ウォームプール内のインスタンスには、ウォームプール内のインスタンスのライフサイクル状態の移行 で説明されている、対応する待機状態を備えた独自のライフサイクルもあります。

ライフサイクルフックを Auto Scaling グループに追加するための準備

Auto Scaling グループにライフサイクルフックを追加する前に、ユーザーデータスクリプト、または通知ターゲットまたはが正しく設定されていることを確認するようにしてください。

- インスタンスの起動中にユーザーデータスクリプトを実行してインスタンスでカスタムアクションを使用するために、通知ターゲットを設定する必要はありません。ただし、ユーザーデータスクリプトを指定する起動テンプレートまたは起動設定を作成し、Auto Scaling グループに関連付けておく必要があります。ユーザーデータスクリプトの詳細については、Amazon EC2 ユーザーガイド」の「起動時に Linux インスタンスでコマンドを実行する」を参照してください。
- ライフサイクルアクションが完了したときに Amazon EC2 Auto Scaling に通知するには、スクリプトに CompleteLifecycleAction API コールを追加し、Auto Scaling インスタンスがこの API を呼び出すことを許可するポリシーを使用して IAM ロールを手動で作成する必要があります。起動テンプレートまたは起動設定では、起動時に Amazon EC2 インスタンスにアタッチされる IAM インスタンスプロファイルを使用して、このロールを指定する必要があります。詳細については、「ライフサイクルアクションを完了する」および「Amazon EC2 インスタンスで実行中のアプリケーション用の IAM ロール」を参照してください。
- Lambda などのサービスを使用してカスタムアクションを実行するには、 EventBridge ルールを作成し、Lambda 関数をターゲットとして指定しておく必要があります。詳細については、「ライフサイクル通知の通知ターゲットを設定する」を参照してください。
- ライフサイクルアクションが完了したときに Lambda が Amazon EC2 Auto Scaling に通知できるようにするには、アクション API CompleteLifecycleコールを関数コードに追加する必要があります。関数の実行ロールに、ライフサイクルフックを完了する許可を Lambda に付与する IAM ポリシーをアタッチする必要もあります。詳細については、「チュートリアル:Lambda 関数を呼び出すライフサイクルフックの設定」を参照してください。
- Amazon SNS や Amazon SQS などのサービスを使用してカスタムアクションを実行するには、SNS トピックまたは SQS キューを作成し、その Amazon リソースネーム (ARN) を準備しておく必要があります。また、SNS トピックまたは SQS ターゲットへの Amazon EC2 Auto Scaling アクセスを許可する IAM ロールを作成し、その ARN を準備しておく必要があります。詳細については、「ライフサイクル通知の通知ターゲットを設定する」を参照してください。

> **Note**
>
> デフォルトでは、コンソールにライフサイクルフックを追加すると、Amazon EC2 Auto Scaling はライフサイクルイベント通知を Amazon に送信します EventBridge。EventBridge またはユーザーデータスクリプトを使用することが推奨されるベストプラクティスです。Amazon SNS または Amazon SQS に直接通知を送信するライフサイクル

フックを作成するには、 AWS CLI、 AWS CloudFormation、または SDK を使用してライフサイクルフックを追加します。

ライフサイクル通知の通知ターゲットを設定する

ライフサイクルフックを Auto Scaling グループに追加して、インスタンスが待機状態になったときにカスタムアクションを実行できます。希望する開発アプローチに応じてターゲットサービスを選択し、これらのアクションを実行できます。

最初のアプローチでは EventBridge 、Amazon を使用して、目的のアクションを実行する Lambda 関数を呼び出します。2 つ目のアプローチでは、通知が発行される Amazon Simple Notification Service (Amazon SNS) トピックを作成することです。クライアントは SNS トピックに登録し、サポートされているプロトコルを使用して公開されたメッセージを受信できます。最後のアプローチでは、Amazon Simple Queue Service (Amazon SQS) を使用します。これは、ポーリングモデルを介してメッセージを交換するために、分散アプリケーションで使用されるメッセージングシステムです。

ベストプラクティスとして、 を使用することをお勧めします EventBridge。Amazon SNS および Amazon SQS に送信される通知には、Amazon EC2 Auto Scaling が に送信する通知と同じ情報が含まれています EventBridge。以前は EventBridge、SNS または SQS に通知を送信し、別のサービスを SNS または SQS と統合してプログラムによるアクションを実行することが標準的な方法でした。現在、 EventBridge では、ターゲットにできるサービスのオプションが増え、サーバーレスアーキテクチャを使用してイベントを簡単に処理できるようになりました。

次の手順では、通知ターゲットの設定方法について説明します。

起動テンプレートまたは起動設定に、起動時にインスタンスを設定するユーザーデータスクリプトがある場合は、インスタンスでカスタムアクションを実行するための通知を受け取る必要はありません。

内容

- を使用して Lambda に通知をルーティングする EventBridge
- Amazon SNS を使用した通知の受信
- Amazon SQS を使用した通知の受信
- Amazon SNS と Amazon SQS の通知メッセージの例

> **Important**
>
> ライフサイクルフックで使用する EventBridge ルール、Lambda 関数、Amazon SNS トピック、および Amazon SQS キューは、常に Auto Scaling グループを作成したのと同じリージョンに存在する必要があります。

を使用して Lambda に通知をルーティングする EventBridge

インスタンスが待機状態になったときに Lambda 関数を呼び出すように EventBridge ルールを設定できます。Amazon EC2 Auto Scaling は、起動または終了するインスタンス EventBridge に関するライフサイクルイベント通知と、ライフサイクルアクションを制御するために使用できるトークンをに発行します。これらのイベントの例については、Amazon EC2 Auto Scaling イベントリファレンス を参照してください。

> **Note**
>
> を使用してイベントルール AWS Management Console を作成すると、コンソールは Lambda 関数を呼び出すためのアクセス許可を付与するために必要な IAM アクセス EventBridge 許可を自動的に追加します。 AWS CLIを使用してイベントルールを作成する場合は、この権限を明示的に付与する必要があります。
> EventBridge コンソールでイベントルールを作成する方法については、「Amazon EventBridge ユーザーガイド」の「イベントに反応する Amazon ルールの作成 EventBridge」を参照してください。
> - または -
> コンソールユーザー向けの初歩的なチュートリアルについては、「チュートリアル:Lambda 関数を呼び出すライフサイクルフックの設定」を参照してください。このチュートリアルでは、起動イベントをリッスンしてログに書き出すシンプルな Lambda CloudWatch 関数を作成する方法を示します。

Lambda 関数を呼び出す EventBridge ルールを作成するには

1. [Lambda コンソール] を使用して Lambda 関数を作成し、Amazon リソースネーム (ARN) を書き留めておきます。例えば `arn:aws:lambda:region:123456789012:function:my-function` です。 EventBridge ターゲットを作成するには ARN が必要です。詳細については、[AWS Lambda デベロッパーガイド] の [Lambda の開始方法] を参照してください。

2. インスタンス起動のイベントに一致するルールを作成するには、次の put-rule コマンドを使用します。

```
aws events put-rule --name my-rule --event-pattern file://pattern.json --state
 ENABLED
```

次の例は、インスタンス起動ライフサイクルアクションの pattern.json を示しています。*####*のテキストは、Auto Scaling グループの名前に置き換えてください。

```
{
  "source": [ "aws.autoscaling" ],
  "detail-type": [ "EC2 Instance-launch Lifecycle Action" ],
  "detail": {
      "AutoScalingGroupName": [ "my-asg" ]
   }
}
```

コマンドが正常に実行されると、 はルールの ARN に EventBridge 応答します。この ARN をメモします。これは、ステップ 4 で入力する必要があります。

他のイベントに一致するルールを作成するには、イベントパターンを変更します。詳細については、「EventBridge を使用して Auto Scaling イベントを処理する」を参照してください。

3. ルールのターゲットとして使用する Lambda 関数を指定するには、次のput-argetコマンドを実行します。

```
aws events put-targets --rule my-rule --targets
 Id=1,Arn=arn:aws:lambda:region:123456789012:function:my-function
```

上記のコマンドで、*my-rule* はステップ 2 でルールに指定した名前になり、Arn パラメータの値はステップ 1 で作成した関数の ARN になります。

4. ルールが Lambda 関数を呼び出せるようにする許可を追加するには、以下の Lambda add-permission コマンドを使用します。このコマンドは、 EventBridge サービスプリンシパル (events.amazonaws.com) を信頼し、指定されたルールにアクセス許可をスコープします。

```
aws lambda add-permission --function-name my-function --statement-id my-unique-id \
  --action 'lambda:InvokeFunction' --principal events.amazonaws.com --source-arn
 arn:aws:events:region:123456789012:rule/my-rule
```

上記のコマンドでは:

- *my-function*は、ルールがターゲットとして使用する Lambda 関数の名前です。
- *my-unique-id* は、Lambda 関数ポリシーのステートメントを記述するためにユーザーが定義する一意の識別子です。
- source-arn は EventBridge ルールの ARN です。

コマンドが正常に実行された場合は、次のような出力が表示されます。

```
{
  "Statement": "{\"Sid\":\"my-unique-id\",
    \"Effect\":\"Allow\",
    \"Principal\":{\"Service\":\"events.amazonaws.com\"},
    \"Action\":\"lambda:InvokeFunction\",
    \"Resource\":\"arn:aws:lambda:us-west-2:123456789012:function:my-function\",
    \"Condition\":
      {\"ArnLike\":
        {\"AWS:SourceArn\":
         \"arn:aws:events:us-west-2:123456789012:rule/my-rule\"}}}"
}
```

Statement 値は、Lambda 関数ポリシーに追加されたステートメントの JSON 文字列バージョンです。

5. これらの指示に従った後、次のステップとして「ライフサイクルフックを追加する」に進みます。

Amazon SNS を使用した通知の受信

Amazon SNS を使用して、ライフサイクルアクションが発生したときに、通知を受け取るよう、通知ターゲット (SNS トピック) をセットアップできます。次に、Amazon SNS は登録された受信者に通知を送信します。登録が確認されるまで、トピックに対して発行された通知は受信者に送信されません。

Amazon SNS を使用して通知をセットアップするには

1. Amazon SNS トピックを作成するには、[Amazon SNS コンソール] または次の [トピックの作成] のコマンドを使用します。このトピックが、使用している Auto Scaling グループと同じリー

ジョンにあることを確認します。詳細については、[Amazon Simple 通知サービスデベロッパーガイド] の [Amazon SNS の使用開始] を参照してください。

```
aws sns create-topic --name my-sns-topic
```

2. その Amazon リソースネーム (ARN) をメモします、例えば、`arn:aws:sns:region:123456789012:my-sns-topic`。ライフサイクルフックを作成するには、それが必要です。
3. IAM サービスロールを作成して、Amazon EC2 Auto Scaling に Amazon SNS 通知ターゲットへのアクセス権を付与します。

   SNS トピックへの Amazon EC2 Auto Scaling のアクセス権を付与するには

   a. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
   b. 左側のナビゲーションペインで、[Roles] を選択します。
   c. [ロールの作成] を選択します。
   d. [Select trusted entity] (信頼されたエンティティの選択) で、[AWS のサービス] を選択します。
   e. ユースケースでは、[他の AWS サービスのユースケース] で [EC2 Auto Scaling]、[EC2 Auto Scaling Notification Access] の順に選択します。
   f. [Next] (次へ) を 2 回選択して、[Name, review, and create] (名前、確認、および作成) ページに進みます。
   g. [Role name] (ロール名) にロールの名前 (**my-notification-role** など) を入力して、[Create role] (ロールを作成) を選択します。
   h. [Roles (ロール)] ページで作成したロールを選択して、[Summary (概要)] ページを開きます。ロールの [ARN] をメモします。例えば `arn:aws:iam::123456789012:role/my-notification-role` です。ライフサイクルフックを作成するには、それが必要です。
4. これらの指示に従った後、次のステップとして「ライフサイクルフックを追加する (AWS CLI)」に進みます。

Amazon SQS を使用した通知の受信

ライフサイクルアクションが発生したときにメッセージを受け取るよう、Amazon SQS を使用して通知ターゲットをセットアップできます。キューコンシューマーは、SQS キューをポーリングしてこれらの通知を処理する必要があります。

> **Important**
>
> FIFO キューはライフサイクルフックとの互換性がありません。

Amazon SQS を使用して通知をセットアップするには

1. [Amazon SQS コンソール] を使用して、Amazon SQS キューを作成します。キューが,使用している Auto Scaling グループと同じリージョンにあることを確認します。詳細については、[Amazon Simple キューサービス デベロッパーガイド] の [Amazon SQS の使用開始] を参照してください。
2. キューの ARN をメモします。例えば、`arn:aws:sqs:us-west-2:123456789012:my-sqs-queue`。ライフサイクルフックを作成するには、それが必要です。
3. IAM サービスロールを作成して、Amazon EC2 Auto Scaling に Amazon SQS 通知ターゲットへのアクセス権を付与します。

   SQS キューへ Amazon EC2 Auto Scaling のアクセス権を付与するには

   a. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
   b. 左側のナビゲーションペインで、[Roles] を選択します。
   c. [ロールの作成] を選択します。
   d. [Select trusted entity] (信頼されたエンティティの選択) で、[AWS のサービス] を選択します。
   e. ユースケースでは、[他の AWS サービスのユースケース] で [EC2 Auto Scaling]、[EC2 Auto Scaling Notification Access] の順に選択します。
   f. [Next] (次へ) を 2 回選択して、[Name, review, and create] (名前、確認、および作成) ページに進みます。
   g. [Role name] (ロール名) にロールの名前 (**`my-notification-role`** など) を入力して、[Create role] (ロールを作成) を選択します。
   h. [Roles (ロール)] ページで作成したロールを選択して、[Summary (概要)] ページを開きます。ロールの [ARN] をメモします。例えば `arn:aws:iam::123456789012:role/my-notification-role` です。ライフサイクルフックを作成するには、それが必要です。
4. これらの指示に従った後、次のステップとして「ライフサイクルフックを追加する (AWS CLI)」に進みます。

Amazon SNS と Amazon SQS の通知メッセージの例

インスタンスが待機状態のときに、Amazon SNSまたはAmazon SQSの通知ターゲットにメッセージが発行されます。メッセージには、次に示す情報が含まれます。

- `LifecycleActionToken` - ライフサイクル アクション トークン。
- `AccountId` — AWS アカウント ID。
- `AutoScalingGroupName` - Auto Scaling グループの名前。
- `LifecycleHookName` - ライフサイクルフックの名前。
- `EC2InstanceId` - EC2 インスタンスの ID。
- `LifecycleTransition` - ライフサイクルフックタイプ。
- `NotificationMetadata` — 通知メタデータ。

通知メッセージの例を次に示します。

```
Service: AWS Auto Scaling
Time: 2021-01-19T00:36:26.533Z
RequestId: 18b2ec17-3e9b-4c15-8024-ff2e8ce8786a
LifecycleActionToken: 71514b9d-6a40-4b26-8523-05e7ee35fa40
AccountId: 123456789012
AutoScalingGroupName: my-asg
LifecycleHookName: my-hook
EC2InstanceId: i-0598c7d356eba48d7
LifecycleTransition: autoscaling:EC2_INSTANCE_LAUNCHING
NotificationMetadata: hook message metadata
```

テスト通知メッセージの例

最初にライフサイクルフックを追加すると、テスト通知メッセージが通知ターゲットに発行されます。テスト通知メッセージの例を次に示します。

```
Service: AWS Auto Scaling
Time: 2021-01-19T00:35:52.359Z
RequestId: 18b2ec17-3e9b-4c15-8024-ff2e8ce8786a
Event: autoscaling:TEST_NOTIFICATION
AccountId: 123456789012
AutoScalingGroupName: my-asg
```

```
AutoScalingGroupARN: arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:042cba90-
ad2f-431c-9b4d-6d9055bcc9fb:autoScalingGroupName/my-asg
```

> **Note**
>
> Amazon EC2 Auto Scaling から に配信されるイベントの例については EventBridge、「」を参照してくださいAmazon EC2 Auto Scaling イベントリファレンス。

インスタンスメタデータを使用してターゲットライフサイクル状態を取得する

起動する各 Auto Scaling インスタンスは、いくつかのライフサイクル状態を経過します。特定のライフサイクル状態移行で実行されるようにインスタンス上のカスタムアクションをインスタンス内から呼び出すには、インスタンスメタデータを使用してターゲットライフサイクル状態を取得する必要があります。

例えば、インスタンスが終了する前にインスタンス上で何らかのコードを実行するために、インスタンス内部からのインスタンスの終了を検出するメカニズムが必要な場合があります。これには、インスタンスからインスタンスのライフサイクル状態を直接ポーリングするコードを書きます。次に、ライフサイクル フックを Auto Scaling グループに追加して、コードが complete-lifecycle-action コマンドを送信して続行するまでインスタンスを実行し続けることができます。

Auto Scaling インスタンスのライフサイクルには、2 つの主な定常状態 (`InService` および `Terminated`) と、2 つの副次的な定常状態 (`Detached` および `Standby`) があります。ウォームプールを使用する場合、ライフサイクルにはさらに 4 つの定常状態 (`Warmed:Hibernated`、`Warmed:Running`、`Warmed:Stopped`、および `Warmed:Terminated`) があります。

インスタンスが前述の定常状態のいずれかに移行する準備を行うと、Amazon EC2 Auto Scaling がインスタンスメタデータ項目の `autoscaling/target-lifecycle-state` の値を更新します。インスタンス内からターゲットライフサイクル状態を取得するには、インスタンスメタデータサービスを使用してインスタンスメタデータから状態を取得する必要があります。

> **Note**
>
> インスタンスメタデータは、アプリケーションがインスタンス情報をクエリするために使用できる、Amazon EC2 インスタンスに関するデータです。インスタンスメタデータサービス

は、ローカルコードがインスタンスメタデータにアクセスするために使用する、インスタンス上のコンポーネントです。ローカルコードには、インスタンスで実行されているユーザーデータスクリプトまたはアプリケーションなどがあります。

ローカルコードは、インスタンスメタデータサービスバージョン 1 (IMDSv1) とインスタンスメタデータサービスバージョン 2 (IMDSv2) の 2 つの方法のいずれかを使用して、実行中のインスタンスからのインスタンスメタデータにアクセスできます。IMDSv2 はセッション指向のリクエストを使用し、インスタンスメタデータへのアクセス試行に利用される可能性がある数種類の脆弱性を緩和します。これら 2 つの方法の詳細については、「Amazon ECIMDSv22 の使用」を参照してください。Amazon EC2

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" -v http://169.254.169.254/latest/meta-data/autoscaling/target-lifecycle-state
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/autoscaling/target-lifecycle-state
```

以下は出力例です。

```
InService
```

ターゲットライフサイクル状態とは、インスタンスの移行先状態のことです。現在のライフサイクル状態は、インスタンスの今の状態です。これらは、ライフサイクルアクションが完了し、インスタンスがターゲットライフサイクル状態への移行を完了した後で同じになる場合があります。インスタンスメタデータからインスタンスの現在のライフサイクル状態を取得することはできません。

Amazon EC2 Auto Scaling は、2022 年 3 月 10 日にターゲットライフサイクル状態の生成を開始しました。この日付以降にインスタンスがターゲットライフサイクル状態のいずれかに移行した場合は、インスタンスメタデータにターゲットライフサイクル状態項目が存在します。移行しなかった場合は項目が存在しないため、HTTP 404 エラーが発生します。

インスタンスメタデータの取得の詳細については、Amazon EC2 ユーザーガイド」の「インスタンスメタデータの取得」を参照してください。

ターゲットライフサイクル状態を使用するユーザーデータスクリプトでカスタムアクションを伴うライフサイクルフックを作成する方法を説明するチュートリアルについては、「チュートリアル: インスタンスメタデータを使用してターゲットライフサイクル状態を取得するようにユーザーデータを設定する」を参照してください。

> **Important**
>
> カスタムアクションをできるだけ早く呼び出しできるようにするには、ローカルコードで IMDS を頻繁にポーリングし、エラーがあれば再試行する必要があります。

ライフサイクルフックを追加する

Auto Scaling インスタンスを待機状態にしてカスタムアクションを実行するために、Auto Scaling グループにライフサイクルフックを追加できます。カスタムアクションは、インスタンスの起動時または終了前に実行されます。インスタンスは、ライフサイクルアクションが完了するまで、またはタイムアウト期間が終了するまで待機状態のままになります。

から Auto Scaling グループを作成したら AWS Management Console、1 つ以上のライフサイクルフックを追加でき、最大合計 50 個のライフサイクルフックを追加できます。 AWS CLI、、または SDK を使用して AWS CloudFormation、作成時に Auto Scaling グループにライフサイクルフックを追加することもできます。

デフォルトでは、コンソールにライフサイクルフックを追加すると、Amazon EC2 Auto Scaling はライフサイクルイベント通知を Amazon に送信します EventBridge。 EventBridge またはユーザーデータスクリプトを使用することが推奨されるベストプラクティスです。Amazon SNS または Amazon SQS に直接通知を送信するライフサイクルフックを作成するには、このトピックの例にあるように、put-lifecycle-hook コマンドを使用できます。

目次

- ライフサイクルフックを追加する (コンソール)
- ライフサイクルフックを追加する (AWS CLI)

ライフサイクルフックを追加する (コンソール)

Auto Scaling グループにライフサイクル フックを追加するには、次の手順に従います。スケールアウト (インスタンスの起動) とスケールイン (インスタンスの終了またはウォーム プールへの復帰) のためのライフサイクルフックを追加するには、2 つの個別のフックを作成する必要があります。

作業を開始する前に、「ライフサイクルフックを Auto Scaling グループに追加するための準備」で説明されているように必要に応じてカスタムアクションがセットアップされていることを確認してください。

スケールアウト用のライフサイクルフックを追加するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。ページの下部にスプリットペインが開きます。
3. [Instance management (インスタンス管理)] タブの [Lifecycle hooks (ライフサイクルフック)] で、[Create lifecycle hook (ライフサイクルフックを作成)] を選択します。
4. スケールアウト (インスタンスが起動) のライフサイクルフックを定義するには、以下を実行してください。

   a. [Lifecycle hook name (ライフサイクルフック名)] で、ライフサイクルフックの名前を指定します。

   b. [Lifecycle transition (ライフサイクルの移行)] で、[Instance launch (インスタンスの起動)] を選択します。

   c. [ハートビートのタイムアウト] には、スケールアウト時にフックがタイムアウトするまで待機状態を維持する時間を秒単位で指定します。この時間の範囲は 30～7200 秒です。タイムアウト期間を長く設定すると、カスタムアクションが完了する時間が長くなります。その後、タイムアウト期間終了前に終了した場合は、complete-lifecycle-action コマンドを送信して、インスタンスが次の状態に進むことを許可します。

   d. [Default result] (デフォルトの結果) には、ライフサイクルフックのタイムアウト時間を過ぎた場合、または予期しない障害が発生した場合に実行するアクションを指定します。[続行] または 中止 のどちらかを選択できます。

      - [続行] を選択すると、Auto Scaling グループは他のライフサイクル フックを続行し、インスタンスをサービスに投入できます。

- [中止] を選択する場合、Auto Scaling グループは残りのアクションをすべて停止し、インスタンスをただちに終了します。

e. (オプション) [通知メタデータ] には、Amazon EC2 Auto Scaling が通知ターゲットにメッセージを送信するときに含めるその他の情報を指定します。

5. [作成] を選択します。

スケールイン用のライフサイクルフックを追加するには

1. スケールアウト用のライフサイクル フックを作成した後、[ライフサイクル フックの作成] を選択して、中断したところから続行します。
2. スケールイン (インスタンスを終了またはウォームプールに戻す) のライフサイクルフックを定義するには、以下を実行してください。

   a. [Lifecycle hook name (ライフサイクルフック名)] で、ライフサイクルフックの名前を指定します。

   b. [Lifecycle transition (ライフサイクルの移行)] で、[Instance terminate (インスタンスの終了)] を選択します。

   c. [ハートビートのタイムアウト] には、スケールアウト時にフックがタイムアウトするまで待機状態を維持する時間を秒単位で指定します。から EC2 ログを取得するなど、最終タスクの実行に必要な時間に応じて、30 ~ 120秒の短いタイムアウト期間をお勧めします CloudWatch。

   d. [Default result] (デフォルトの結果) には、タイムアウト時間を超過した、または予期しない失敗が発生した場合に Auto Scaling グループが実行するアクションを指定します。[ABANDON] (中止) と [CONTINUE] (続行) は、どちらもインスタンスを終了します。

      - [CONTINUE] (続行) を選択する場合、Auto Scaling グループは、終了前に他のライフサイクルフックなどの残りのアクションを続行できます。
      - [中止] を選択すると、Auto Scaling グループはインスタンスをただちに終了します。

   e. (オプション) [通知メタデータ] には、Amazon EC2 Auto Scaling が通知ターゲットにメッセージを送信するときに含めるその他の情報を指定します。

3. [Create] を選択します。

ライフサイクルフックを追加する (AWS CLI)

[put-lifecycle-hook] コマンドを使用して、ライフサイクルフックを作成および更新ができます。

スケールアウトでアクションを実行するには、以下のコマンドを使用します。

```
aws autoscaling put-lifecycle-hook --lifecycle-hook-name my-launch-hook  \
  --auto-scaling-group-name my-asg \
  --lifecycle-transition autoscaling:EC2_INSTANCE_LAUNCHING
```

スケールインでアクションを実行するには、以下のコマンドを使用します。

```
aws autoscaling put-lifecycle-hook --lifecycle-hook-name my-termination-hook  \
  --auto-scaling-group-name my-asg \
  --lifecycle-transition autoscaling:EC2_INSTANCE_TERMINATING
```

Amazon SNS または Amazon SQS を使用して通知を受信するには、`--notification-target-arn` および `--role-arn` オプションを追加します。

以下の例は、*my-sns-topic* という名前の SNS トピックを通知ターゲットとして指定するライフサイクルフックを作成します。

```
aws autoscaling put-lifecycle-hook --lifecycle-hook-name my-termination-hook  \
  --auto-scaling-group-name my-asg \
  --lifecycle-transition autoscaling:EC2_INSTANCE_TERMINATING \
  --notification-target-arn arn:aws:sns:region:123456789012:my-sns-topic \
  --role-arn arn:aws:iam::123456789012:role/my-notification-role
```

トピックは、以下のキーと値のペアが含まれたテスト通知を受け取ります。

```
"Event": "autoscaling:TEST_NOTIFICATION"
```

デフォルトで、put-lifecycle-hook コマンドは、ハートビートタイムアウトが 3600 秒 (1 時間) のライフサイクルフックを作成します。

既存のライフサイクルフックのハートビートタイムアウトを変更するには、以下の例にあるように、`--heartbeat-timeout` オプションを追加します。

```
aws autoscaling put-lifecycle-hook --lifecycle-hook-name my-termination-hook \
  --auto-scaling-group-name my-asg --heartbeat-timeout 120
```

インスタンスがすでに待機状態になっている場合は、record-lifecycle-action-heartbeat CLI コマンドを使用して、ハートビートを記録することによってライフサイクルフックがタイムアウトしないよう

にすることができます。これにより、ライフサイクルフックを作成するときに指定するタイムアウト時間によってタイムアウト期間が延長されます。タイムアウト期間が終わる前に終了した場合は、インスタンスが次の状態に進むことを許可する complete-lifecycle-action CLI コマンドを送信することができます。詳細な説明と例については、「ライフサイクルアクションを完了する」を参照してください。

ライフサイクルアクションを完了する

Auto Scaling グループがライフサイクルイベントに応答する際、Auto Scaling グループはインスタンスを待機状態にしてイベント通知を送信します。インスタンスが待機状態にあるときに、カスタムアクションを実行できます。

ライフサイクルアクションをCONTINUE の結果で完了させることで、タイムアウト期間が経過する前に終了させることができます。ライフサイクル アクションを完了しない場合、タイムアウト期間が終了すると、ライフサイクル フックはデフォルトの結果コードに指定したステータスになります。

内容

- ライフサイクルアクションを完了する (手動)
- ライフサイクルアクションを完了する (自動)

ライフサイクルアクションを完了する (手動)

次の手順はコマンドライン インターフェイスに関するもので、コンソールではサポートされていません。インスタンス ID や Auto Scaling グループの名前など、置き換える必要がある情報は、斜体で表示されます。

ライフサイクルアクションを完了するには (AWS CLI)

1. カスタムアクション完了までにさらに時間が必要な場合は、record-lifecycle-action-heartbeat コマンドを使用してタイムアウト時間を再開し、インスタンスを待機状態に維持します。例えば、タイムアウト期間が 1 時間に設定されており、30 分後にこのコマンドを呼び出す場合、インスタンスはさらに 1 時間 (合計で 90 分) 待機状態のままになります。

   以下のコマンドに示すように、通知と共に受信したライフサイクルアクショントークンを指定できます。

   ```
   aws autoscaling record-lifecycle-action-heartbeat --lifecycle-hook-name my-launch-hook \
   ```

```
  --auto-scaling-group-name my-asg --lifecycle-action-
token bcd2f1b8-9a78-44d3-8a7a-4dd07d7cf635
```

または、以下のコマンドに示すように、通知とともに受信したインスタンスの ID を指定できます。

```
aws autoscaling record-lifecycle-action-heartbeat --lifecycle-hook-name my-launch-
hook \
  --auto-scaling-group-name my-asg --instance-id i-1a2b3c4d
```

2. [complete-lifecycle-action] コマンドを使用してタイムアウト期間が終了する前にカスタムアクションを終了すると、Auto Scaling グループは起動を継続するか、インスタンスを終了することができます。以下のコマンドに示すように、ライフサイクルアクショントークンを指定できます。

```
aws autoscaling complete-lifecycle-action --lifecycle-action-result CONTINUE \
  --lifecycle-hook-name my-launch-hook --auto-scaling-group-name my-asg \
  --lifecycle-action-token bcd2f1b8-9a78-44d3-8a7a-4dd07d7cf635
```

または、以下のコマンドに示すように、インスタンスの ID を指定できます。

```
aws autoscaling complete-lifecycle-action --lifecycle-action-result CONTINUE \
  --instance-id i-1a2b3c4d --lifecycle-hook-name my-launch-hook \
  --auto-scaling-group-name my-asg
```

ライフサイクルアクションを完了する (自動)

起動後にインスタンスを設定するユーザーデータスクリプトがある場合は、ライフサイクルアクションを手動で完了する必要はありません。complete-lifecycle-action コマンドをスクリプトに追加します。スクリプトは、インスタンスのメタデータからインスタンス ID を取得し、ブートストラップスクリプトが正常に完了したときに Amazon EC2 Auto Scaling に通知します。

まだそうしていない場合は、インスタンスメタデータからインスタンスのインスタンス ID を取得するためのスクリプトを更新します。詳細については、「Amazon EC2 ユーザーガイド」の「インスタンスメタデータの取得」を参照してください。 Amazon EC2

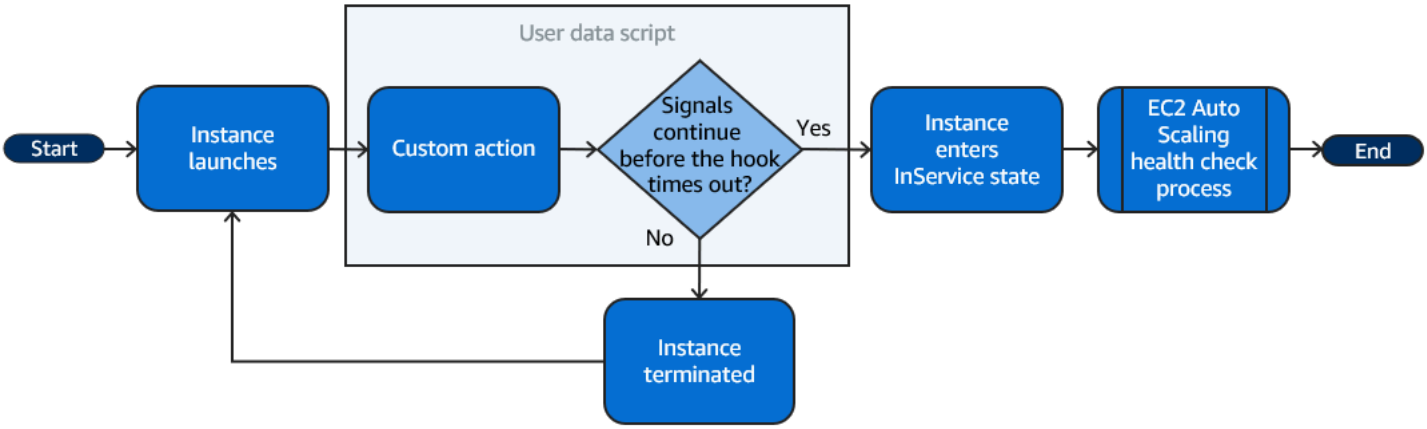
Lambda を使用する場合は、カスタムアクションが成功した際にインスタンスのライフサイクルを続行できるように、関数のコードにコールバックを設定することもできます。詳細については、「チュートリアル:Lambda 関数を呼び出すライフサイクルフックの設定」を参照してください。

チュートリアル: インスタンスメタデータを使用してターゲットライフサイクル状態を取得するようにユーザーデータを設定する

ライフサイクルフックのカスタムアクションを作成する一般的な方法は、Amazon EC2 Auto Scaling が Amazon などの他の サービスに送信する通知を使用することです EventBridge。その代わりに、インスタンスを設定してライフサイクルアクションを完了するコードをインスタンスそのものに移動させるユーザーデータスクリプトを使用することによって、追加のインフラストラクチャを作成する手間を省くことができます。

以下のチュートリアルは、ユーザーデータスクリプトとインスタンスメタデータを使用して開始する方法を説明します。グループ内のインスタンスのターゲットライフサイクル状態を読み取り、インスタンスのライフサイクルの特定のフェーズでコールバックアクションを実行して起動プロセスを続行するユーザーデータスクリプトを使用した、基本的な Auto Scaling グループ設定を作成します。

次の図は、ユーザーデータスクリプトを使用してカスタムアクションを実行する際のスケールアウトイベントのフローをまとめたものです。インスタンスの起動後、インスタンスのライフサイクルは、タイムアウトするか、続行するシグナルを受信する Amazon EC2 Auto Scaling によって、ライフサイクルフックが完了するまで一時停止されます。



内容

- ステップ 1: ライフサイクルアクションを完了するための許可を持つ IAM ロールを作成する
- ステップ 2: 起動テンプレートを作成して IAM ロールとユーザーデータスクリプトを含める
- ステップ 3: Auto Scaling グループを作成する
- ステップ 4: ライフサイクルフックを追加する
- ステップ 5: 機能をテストして検証する
- ステップ 6: クリーンアップする
- 関連リソース

ステップ 1: ライフサイクルアクションを完了するための許可を持つ IAM ロールを作成する

AWS CLI または AWS SDK を使用してライフサイクルアクションを完了するためのコールバックを送信する場合は、ライフサイクルアクションを完了するためのアクセス許可を持つ IAM ロールを使用する必要があります。

ポリシーを作成するには

1. IAM コンソールの [ポリシーページ] を開き、[ポリシーの作成] を選択します。
2. [JSON] タブを選択します。
3. [Policy Document] (ポリシードキュメント) ボックスで、以下のポリシードキュメントをコピーし、ボックスに貼り付けます。*sample text* を、お使いのアカウント番号と、作成する Auto Scaling グループの名前 (**TestAutoScalingEvent-group**) に置き換えます。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "autoscaling:CompleteLifecycleAction"
      ],
      "Resource":
 "arn:aws:autoscaling:*:123456789012:autoScalingGroup:*:autoScalingGroupName/
TestAutoScalingEvent-group"
    }
  ]
}
```

4. [次へ] をクリックします。
5. [ポリシー名] に「**TestAutoScalingEvent-policy**」と入力します。[ポリシーの作成] を選択します。

ポリシーの作成が完了したら、それを使用するロールを作成できます。

ロールを作成するには

1. 左側のナビゲーションペインで、[Roles] を選択します。

2. [ロールの作成] を選択します。
3. [Select trusted entity] (信頼されたエンティティの選択) で、[AWS のサービス] を選択します。
4. ユースケースに [EC2] を選択してから、[Next] (次へ) を選択します。
5. 「アクセス許可の追加」で、作成したポリシー (TestAutoScalingEvent-policy) を選択します。[次へ] を選択します。
6. [Name, review, and create] (名前、確認、および作成) ページで、[Role name] (ロール名) に **`TestAutoScalingEvent-role`** を入力し、[Create role] (ロールを作成) を選択します。

ステップ 2: 起動テンプレートを作成して IAM ロールとユーザーデータスクリプトを含める

Auto Scaling グループで使用する起動テンプレートを作成します。作成した IAM ロールと、提供されたサンプルユーザーデータスクリプトを含めます。

起動テンプレートを作成するには

1. Amazon EC2 コンソールの起動テンプレートページを開きます。
2. [起動テンプレートの作成] を選択します。
3. [起動テンプレート名] を使用する場合、**`TestAutoScalingEvent-template`** を入力します。
4. [Auto Scaling ガイダンス] で、チェックボックスを選択します。
5. [アプリケーションおよび OS イメージ (Amazon マシンイメージ)] で、[クイックスタート] から Amazon Linux 2 (HVM)、SSD Volume Type、64 ビット (x86) を選択します。
6. [Instance type] (インスタンスタイプ) には、Amazon EC2 インスタンスのタイプ (「t2.micro」など) を選択します。
7. [詳細設定] を使用する場合、セクションを展開してフィールドを表示します。
8. IAM インスタンスプロファイル で、IAM ロールの IAM インスタンスプロファイル名 (TestAutoScalingEvent-role) を選択します。インスタンスプロファイルは IAM ロールのコンテナであり、インスタンスの起動時に Amazon EC2 インスタンスに IAM ロール情報を渡すために使用できます。

   IAM コンソールを使用して IAM ロールを作成すると、コンソールが対応するロールと同じ名前を使用したインスタンスプロファイルを自動的に作成します。
9. [User data] (ユーザーデータ) では、以下のサンプルユーザーデータスクリプトをコピーして、フィールドに貼り付けます。のサンプルテキスト`group_name`を、作成する Auto Scaling グ

ループの名前regionに置き換え、 を Auto Scaling グループで使用する に置き換え AWS リージョン ます。

```
#!/bin/bash

function get_target_state {
    echo $(curl -s http://169.254.169.254/latest/meta-data/autoscaling/target-
lifecycle-state)
}

function get_instance_id {
    echo $(curl -s http://169.254.169.254/latest/meta-data/instance-id)
}

function complete_lifecycle_action {
    instance_id=$(get_instance_id)
    group_name='TestAutoScalingEvent-group'
    region='us-west-2'

    echo $instance_id
    echo $region
    echo $(aws autoscaling complete-lifecycle-action \
      --lifecycle-hook-name TestAutoScalingEvent-hook \
      --auto-scaling-group-name $group_name \
      --lifecycle-action-result CONTINUE \
      --instance-id $instance_id \
      --region $region)
}

function main {
    while true
    do
        target_state=$(get_target_state)
        if [ \"$target_state\" = \"InService\" ]; then
            # Change hostname
            export new_hostname="${group_name}-$instance_id"
            hostname $new_hostname
            # Send callback
            complete_lifecycle_action
            break
        fi
        echo $target_state
        sleep 5
```

```
    done
}

main
```

このシンプルなユーザーデータスクリプトは、以下を実行します。

- インスタンスメタデータを呼び出して、インスタンスメタデータからターゲットライフサイクル状態とインスタンス ID を取得する
- ターゲットライフサイクル状態が `InService` に変更されるまで、状態を繰り返し取得する
- ターゲットライフサイクル状態が `InService` の場合、インスタンスのホスト名を Auto Scaling グループの名前が先頭に付加されたインスタンス ID に変更する
- complete-lifecycle-action CLI コマンドを呼び出すことによってコールバックを送信し、EC2 起動プロセスを `CONTINUE` するように Amazon EC2 Auto Scaling に通知する

10. [起動テンプレートの作成] を選択します。
11. 確認ページで、[Auto Scaling グループの作成] を選択します。

> **Note**
>
> ユーザーデータスクリプトを開発するためのリファレンスとして使用できるその他の例については、Amazon EC2 Auto Scaling のGitHub リポジトリ」を参照してください。

ステップ 3: Auto Scaling グループを作成する

起動テンプレートを作成したら、Auto Scaling グループを作成します。

Auto Scaling グループを作成する

1. [Choose launch template or configuration] (起動テンプレートまたは起動設定を選択する) ページで、[Auto Scaling group name] (Auto Scaling グループ名) に Auto Scaling グループの名前 (**`TestAutoScalingEvent-group`**) を入力します。
2. [Next] (次へ) を選択して、[Choose instance launch options] (インスタンス起動オプションを選択) ページに進みます。
3. [Network] (ネットワーク) で VPC を選択します。

4. [Availability Zones and subnets] (アベイラビリティーゾーンとサブネット) で、1 つ、または複数のアベイラビリティーゾーンから 1 つ、または複数のサブネットを選択します。
5. [Instance type requirements] (インスタンスタイプの要件) セクションでは、このステップを簡略化するためにデフォルト設定を使用します。(起動テンプレートを上書きしないでください。) このチュートリアルでは、起動テンプレートで指定されたインスタンスタイプを使用して、オンデマンドインスタンスを 1 つだけ起動します。
6. 画面の最下部にある [Skip to review] (スキップして確認) を選択します。
7. [Review] (確認) ページで Auto Scaling グループの詳細を確認してから、[Create Auto Scaling group] (Auto Scaling グループを作成) を選択します。

ステップ 4: ライフサイクルフックを追加する

ライフサイクルアクションが完了するまでインスタンスを待機状態に維持するライフサイクルフックを追加します。

ライフサイクルフックを追加するには

1. Amazon EC2 コンソールで Auto Scaling グループのページを開きます。
2. Auto Scaling グループの横にあるチェックボックスを選択します。ページの下部にスプリットペインが開きます。
3. 下部のペインで、[Instance management (インスタンス管理)] タブの [Lifecycle hooks (ライフサイクルフック)] で、[Create lifecycle hook (ライフサイクルフックを作成)] を選択します。
4. スケールアウト (インスタンスが起動) のライフサイクルフックを定義するには、以下を実行してください。

   a. [ライフサイクルフック名] で、**`TestAutoScalingEvent-hook`**を入力します。
   b. [Lifecycle transition (ライフサイクルの移行)] で、[Instance launch (インスタンスの起動)] を選択します。
   c. [Heartbeat timeout] (ハートビートタイムアウト) に、ユーザーデータスクリプトからのコールバックを待つ秒数として **`300`** を入力します。
   d. [デフォルトの結果] で、[中止] を選択します。フックがユーザーデータスクリプトからのコールバックを受け取ることなくタイムアウトすると、Auto Scaling グループは新しいインスタンスを終了します。
   e. (オプション) [Notification metadata] (通知メタデータ) を空のままにしておきます。

5. [作成] を選択します。

ステップ 5: 機能をテストして検証する

機能をテストするには、Auto Scaling グループの希望キャパシティを 1 つ増やすことによって Auto Scaling グループを更新します。インスタンスの起動直後にユーザーデータスクリプトが実行され、インスタンスのターゲットライフサイクル状態のチェックが開始されます。スクリプトは、ターゲットライフサイクル状態が `InService` になるとホスト名を変更し、コールバックアクションを送信します。これには通常、完了まで数秒しかかかりません。

Auto Scaling グループのサイズを増やすには

1. Amazon EC2 コンソールで Auto Scaling グループのページを開きます。
2. Auto Scaling グループの横にあるチェックボックスを選択します。上部ペインの最上部の行が見えている状態で、下部ペインの詳細を確認します。
3. 下部のペインの [詳細] タブで、[グループの詳細] 、[編集] を順に選択します。
4. [Desired capacity (希望するキャパシティ)] の場合は、現在の値を 1 ずつ増やします。
5. [更新] を選択します。インスタンスの起動中は、上部ペインの [Status (ステータス)] 列に [Updating capacity (キャパシティの更新)] というステータスが表示されます。

希望キャパシティを増やした後で、スケーリングアクティビティの説明からインスタンスが正常に起動され、終了されていないことを確認できます。

スケーリングを表示するには

1. [Auto Scaling グループ] ページに戻り、グループを選択します。
2. [アクティビティ] タブの [アクティビティ履歴] では、[ステータス] 列に、Auto Scaling グループがインスタンスを正常に起動したかどうかが表示されます。
3. ユーザーデータスクリプトが失敗した場合は、タイムアウト期間が過ぎたときに、ステータスが `Canceled` のスケーリングアクティビティと、`Instance failed to complete user's Lifecycle Action: Lifecycle Action with token e85eb647-4fe0-4909-b341-a6c42EXAMPLE was abandoned: Lifecycle Action Completed with ABANDON Result` のステータスメッセージが表示されます。

ステップ 6: クリーンアップする

このチュートリアルのために作成したリソースでの作業が完了したら、次の手順を実行してそれらを削除してください。

ライフサイクルフックを削除するには

1. Amazon EC2 コンソールで Auto Scaling グループのページを開きます。
2. Auto Scaling グループの横にあるチェックボックスを選択します。
3. [Instance management (インスタンス管理)] タブの [Lifecycle hooks (ライフサイクルフック)] で、[lifecycle hook (ライフサイクルフック)] を選択します。(`TestAutoScalingEvent-hook`)
4. [アクション]、[削除] の順に選択します。
5. 確認のために、もう一度 [削除] を選択します。

起動テンプレートを削除するには

1. Amazon EC2 コンソールの起動テンプレートページを開きます。
2. 起動テンプレート (`TestAutoScalingEvent-template`) を選択してから、[Actions] (アクション)、[Delete template] (テンプレートを削除) の順に選択します。
3. 確認を求められたら、**`Delete`** を入力して指定した起動テンプレートの削除を確認し、[Delete] (削除) を選択します。

サンプル Auto Scaling グループでの作業が完了したら、グループを削除してください。作成した IAM ロールと許可ポリシーも削除できます。

Auto Scaling グループを削除するには

1. Amazon EC2 コンソールで Auto Scaling グループのページを開きます。
2. Auto Scaling グループ (`TestAutoScalingEvent-group`) の横にあるチェックボックスをオンにして、[Delete] (削除) を選択します。
3. 確認を求められたら、**`delete`** を入力して指定された Auto Scaling グループの削除を確認し、[Delete] (削除) を選択します。

   [Name (名前)] 列のロードアイコンに、Auto Scaling グループが削除されたことが示されます。インスタンスを終了してグループを削除するには数分かかります。

IAM ロールを削除するには

1. IAM コンソールの [Roles (ロール)] ページを開きます。
2. 関数のロール (`TestAutoScalingEvent-role`) を選択します。

3. [削除] を選択します。
4. 確認を求められたら、ロールの名前を入力し、[Delete] (削除) を選択します。

IAM ポリシーを削除するには

1. IAM コンソールのポリシーページを開きます。
2. 作成したポリシーを選択します。(`TestAutoScalingEvent-policy`)
3. [アクション]、[削除] の順に選択します。
4. 確認を求められたら、ポリシーの名前を入力し、[Delete] (削除) を選択します。

関連リソース

以下の関連トピックは、インスタンスメタデータで利用可能なデータに基づいてインスタンスに対してアクションを呼び出すコードを開発する際に役立ちます。

- インスタンスメタデータを使用してターゲットライフサイクル状態を取得する. このセクションでは、インスタンスの終了など、他のユースケースのライフサイクルステータスについて説明します。
- ライフサイクルフックを追加する (コンソール). この手順では、スケールアウト (インスタンスの起動) とスケールイン (インスタンスの終了またはウォームプールへの復帰) の両方にライフサイクルフックを追加する方法を示します。
- 「Amazon EC2 ユーザーガイド」の「インスタンスメタデータカテゴリ」。 Amazon EC2 このトピックでは、EC2 インスタンスでアクションを呼び出すために使用できるインスタンスメタデータのすべてのカテゴリを一覧表示します。

Amazon を使用して、Auto Scaling グループのインスタンスに発生したイベントに基づいて Lambda 関数を呼び出すルール EventBridge を作成する方法を示すチュートリアルについては、「」を参照してくださいチュートリアル:Lambda 関数を呼び出すライフサイクルフックの設定。

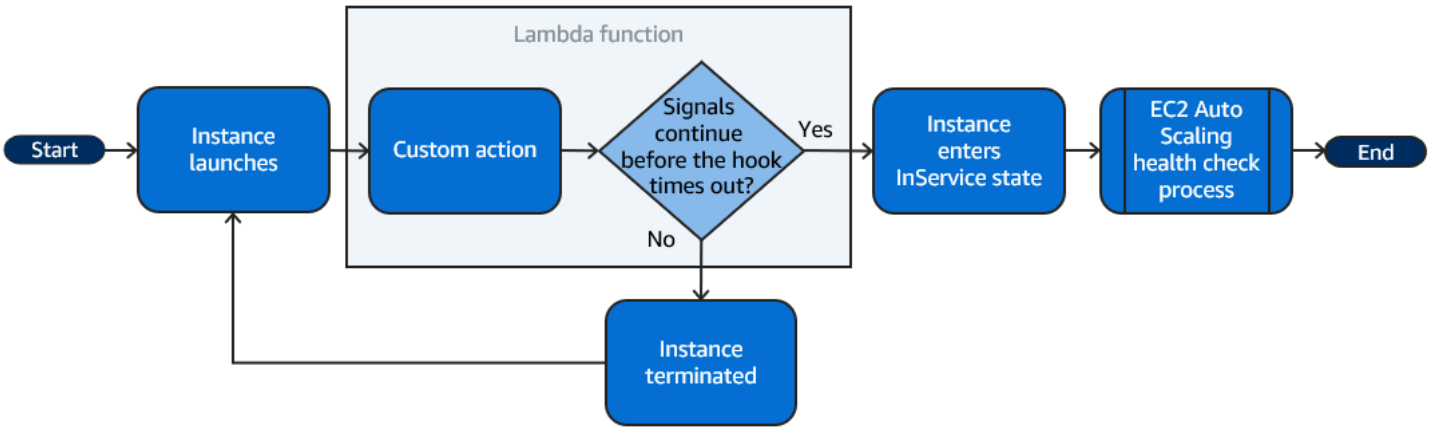
チュートリアル:Lambda 関数を呼び出すライフサイクルフックの設定

この演習では、一致したときに EventBridge ルールターゲットとして AWS Lambda 関数を呼び出すフィルターパターンを含む Amazon ルールを作成します。使用するフィルターパターンとサンプル関数コードを提供します。

すべてが正しく設定されると、このチュートリアルの最後に、インスタンスの起動時に Lambda 関数はカスタムアクションを実行します。カスタムアクションは、Lambda 関数に関連付けられた CloudWatch ログストリームにイベントを記録するだけです。

Lambda 関数もコールバックを実行して、このアクションが成功するとインスタンスのライフサイクルを続行しますが、アクションが失敗するとインスタンスは起動を中止し、終了させます。

次の図は、Lambda 関数を使用してカスタムアクションを実行する際のスケールアウトイベントのフローをまとめたものです。インスタンスの起動後、インスタンスのライフサイクルは、タイムアウトするか、続行するシグナルを受信する Amazon EC2 Auto Scaling によって、ライフサイクルフックが完了するまで一時停止されます。



内容

- 前提条件
- ステップ 1: ライフサイクルアクションを完了するための許可を持つ IAM ロールを作成する
- ステップ 2: Lambda 関数を作成する
- ステップ 3: EventBridge ルールを作成する
- ステップ 4: ライフサイクルフックを追加する
- ステップ 5: イベントをテストし、検証する
- ステップ 6: クリーンアップする
- 関連リソース

前提条件

このチュートリアルを開始する前に、Auto Scaling グループがまだない場合は、作成します。Auto Scaling グループを作成するには、Amazon EC2 コンソールで、Auto Scaling グループのページを開き、[Auto Scaling グループの作成] を選択します。

ステップ 1: ライフサイクルアクションを完了するための許可を持つ IAM ロールを作成する

Lambda 関数を作成する前に、実行ロールと許可ポリシーを作成して、Lambda がライフサイクルフックを完了できるようにする必要があります。

ポリシーを作成するには

1. IAM コンソールの [ポリシーページ] を開き、[ポリシーの作成] を選択します。
2. [JSON] タブを選択します。
3. [ポリシードキュメント] ボックスに、次のポリシードキュメントを貼り付け、[######] のテキストはアカウント番号と Auto Scaling グループの名前に置き換えます。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "autoscaling:CompleteLifecycleAction"
      ],
      "Resource":
 "arn:aws:autoscaling:*:123456789012:autoScalingGroup:*:autoScalingGroupName/my-
asg"
    }
  ]
}
```

4. [次へ] をクリックします。
5. [ポリシー名] に「**LogAutoScalingEvent-policy**」と入力します。[ポリシーの作成] を選択します。

ポリシーの作成が完了したら、それを使用するロールを作成できます。

ロールを作成するには

1. 左側のナビゲーションペインで、[Roles] を選択します。
2. [ロールの作成] を選択します。
3. [Select trusted entity] (信頼されたエンティティの選択) で、[AWS のサービス] を選択します。

4. ユースケースに [Lambda] を選択してから、[Next] (次へ) を選択します。
5. アクセス許可の追加 で、作成したポリシー (LogAutoScalingEvent-policy) と という名前のポリシーを選択しますAWSLambdaBasicExecutionRole。[次へ] を選択します。

> **Note**
>
> AWSLambdaBasicExecutionRole ポリシーには、関数がログを CloudWatch ログに書き込むために必要なアクセス許可があります。

6. [Name, review, and create] (名前、確認、および作成) ページで、[Role name] (ロール名) に **`LogAutoScalingEvent-role`** を入力し、[Create role] (ロールを作成) を選択します。

ステップ 2: Lambda 関数を作成する

イベントの対象となる Lambda 関数を作成します。Node.js で記述されたサンプル Lambda 関数は、一致するイベントが Amazon EC2 Auto Scaling によって出力された EventBridge ときに によって呼び出されます。

Lambda 関数を作成するには

1. Lambda コンソールで [Functions (関数)] ページを開きます。
2. [関数の作成] を選択し、[一から作成] を選択します。
3. [基本的な情報] の [関数名] に「**`LogAutoScalingEvent`**」と入力します。
4. [ランタイム] で [Node.js 18.x] を選択します。
5. スクロールして [デフォルトの実行ロールの変更]を選択し、[実行ロール] で、[既存のロールを使用] を選択します。
6. 既存のロール で、LogAutoScalingEvent-role を選択します。
7. 他はデフォルト値のままにしておきます。
8. [機能の作成]を選択します。関数のコードと設定に戻ります。
9. コンソールで `LogAutoScalingEvent` 関数を開いたまま、エディタの [コードソース] で、index.mjs という名前のファイルに次のサンプルコードを貼り付けます。

```
import { AutoScalingClient, CompleteLifecycleActionCommand } from "@aws-sdk/client-
auto-scaling";
export const handler = async(event) => {
  console.log('LogAutoScalingEvent');
  console.log('Received event:', JSON.stringify(event, null, 2));
```

```
  var autoscaling = new AutoScalingClient({ region: event.region });
  var eventDetail = event.detail;
  var params = {
    AutoScalingGroupName: eventDetail['AutoScalingGroupName'], /* required */
    LifecycleActionResult: 'CONTINUE', /* required */
    LifecycleHookName: eventDetail['LifecycleHookName'], /* required */
    InstanceId: eventDetail['EC2InstanceId'],
    LifecycleActionToken: eventDetail['LifecycleActionToken']
  };
  var response;
  const command = new CompleteLifecycleActionCommand(params);
  try {
    var data = await autoscaling.send(command);
    console.log(data); // successful response
    response = {
      statusCode: 200,
      body: JSON.stringify('SUCCESS'),
    };
  } catch (err) {
    console.log(err, err.stack); // an error occurred
    response = {
      statusCode: 500,
      body: JSON.stringify('ERROR'),
    };
  }
  return response;
};
```

このコードは単にイベントをログに記録するだけで、このチュートリアルの最後に、この Lambda 関数に関連付けられている CloudWatch Logs ログストリームにイベントが表示されます。

10. [デプロイ] を選択します。

ステップ 3: EventBridge ルールを作成する

Lambda 関数を実行する EventBridge ルールを作成します。の使用の詳細については、EventBridge「」を参照してください EventBridge を使用して Auto Scaling イベントを処理する。

コンソールを使用してルールを作成するには

1. EventBridge コンソールを開きます。

2. ナビゲーションペインで Rules] (ルール) を選択します。
3. ルールの作成 を選択します。
4. [Define rule detail] (詳細の定義) で、次の操作を行います。
   a. [名前] に**LogAutoScalingEvent-rule**と入力します。
   b. [イベントバス] として、[デフォルト] を選択します。 AWS のサービス アカウントの がイベントを生成すると、常にアカウントのデフォルトのイベントバスに送られます。
   c. ルールタイプ では、イベントパターンを持つルール] を選択します。
   d. 次へ をクリックします。
5. [Build event pattern] (イベントパターンの作成) で、次の操作を行います。
   a. イベントソース で、AWS イベント または EventBridge パートナーイベント を選択します。
   b. [イベントパターン] までスクロールして、次の操作を行います。
   c. i. イベントソース で AWS のサービス を選択します。
      ii. [AWS のサービス] には、[Auto Scaling] を選択します。
      iii. [イベントタイプ] で、[インスタンスの起動と終了] を選択します。
      iv. デフォルトで、ルールはすべてのスケールインイベントまたはスケールアウトイベントに一致します。スケールアウトイベントが発生し、ライフサイクルフックに基づいてインスタンスが待機状態になったときに通知するルールを作成するには、[Specific instance event(s)] (特定のインスタンスイベント) を選択してから、[EC2 Instance-launch Lifecycle Action] (EC2 インスタンス起動ライフサイクルアクション) を選択します。
      v. デフォルトでは、このルールはリージョン内のすべての Auto Scaling グループと一致します。ルールを特定の Auto Scaling グループに一致させるには、[特定のグループ名] を選択してグループを選択します。
      vi. [次へ] をクリックします。
6. [Select target(s)] (ターゲットを選択) で、以下の操作を行います。
   a. [Target types] (ターゲットタイプ) には、[AWS のサービス] を選択します。
   b. [Select a target] (ターゲットを選択) では、[Lambda function] (Lambda 関数) を選択します。
   c. 関数 で、 を選択しますLogAutoScalingEvent。
   d. [次へ] を 2 回選択します。

7. [Review and create] (確認して作成) ページで、[Create rule] (ルールの作成) を選択します。

ステップ 4: ライフサイクルフックを追加する

このセクションでは、Lambda が起動時にインスタンスで関数を実行できるように、ライフサイクルフックを追加します。

ライフサイクルフックを追加するには

1. Amazon EC2 コンソールで Auto Scaling グループのページを開きます。
2. Auto Scaling グループの横にあるチェックボックスを選択します。ページの下部にスプリットペインが開きます。
3. 下部のペインで、[Instance management (インスタンス管理)] タブの [Lifecycle hooks (ライフサイクルフック)] で、[Create lifecycle hook (ライフサイクルフックを作成)] を選択します。
4. スケールアウト (インスタンスが起動) のライフサイクルフックを定義するには、以下を実行してください。

   a. [ライフサイクルフック名] で、**`LogAutoScalingEvent-hook`**を入力します。

   b. [Lifecycle transition (ライフサイクルの移行)] で、[Instance launch (インスタンスの起動)] を選択します。

   c. [ハートビートのタイムアウト] で、Lambda 関数からのコールバックを待機する秒数として**`300`**を入力します。

   d. [デフォルトの結果] で、[中止] を選択します。つまり、Lambda 関数からコールバックを受け取らずにフックがタイムアウトすると、Auto Scaling グループは新しいインスタンスを終了します。

   e. （オプション）[通知メタデータ] を空にします。に渡すイベントデータには、Lambda 関数を呼び出すために必要なすべての情報 EventBridge が含まれています。

5. [作成] を選択します。

ステップ 5: イベントをテストし、検証する

イベントをテストするには、Auto Scaling グループで希望するキャパシティを 1 増やして Auto Scaling グループを更新します。Lambda 関数は、希望するキャパシティーを増やしてから数秒以内に呼び出されます。

Auto Scaling グループのサイズを増やすには

1. Amazon EC2 コンソールで Auto Scaling グループのページを開きます。
2. Auto Scaling グループの横にあるチェックボックスを選択すると、下部のペインに詳細が表示され、上部のペインの一番上の行が表示されます。
3. 下部のペインの [詳細] タブで、[グループの詳細] 、[編集] を順に選択します。
4. [Desired capacity (希望するキャパシティ)] の場合は、現在の値を 1 ずつ増やします。
5. [更新] を選択します。インスタンスの起動中は、上部ペインの [Status (ステータス)] 列に [Updating capacity (キャパシティの更新)] というステータスが表示されます。

希望するキャパシティーに増やしたら、Lambda 関数が呼び出されたことを確認できます。

Lambda 関数からの出力を表示するには

1. CloudWatch コンソールのロググループページを開きます。
2. Lambda 関数 (`/aws/lambda/LogAutoScalingEvent`) のロググループの名前を選択します。
3. ライフサイクルアクションの関数によって提供されるデータを表示するログのストリーミング名を選択します。

次に、スケーリング アクティビティの説明から、インスタンスが正常に起動したことを確認できます。

スケーリングを表示するには

1. [Auto Scaling グループ] ページに戻り、グループを選択します。
2. [アクティビティ] タブの [アクティビティ履歴] では、[ステータス] 列に、Auto Scaling グループがインスタンスを正常に起動したかどうかが表示されます。

   - アクションが成功した場合、スケーリング アクティビティのステータスは「成功」になります。
   - 失敗した場合、数分待ってから、ステータスが「キャンセル済み」のスケーリング アクティビティが表示され、「インスタンスはユーザーのライフサイクルアクションを完了できませんでした:トークンE85EB647-4FE0-4909-B341-A6C42の例は中止されました:ライフサイクルアクションは中止された結果で完了しました」というステータスメッセージが表示されます。

Auto Scaling グループのサイズを縮小するには

このテスト用に起動した追加のインスタンスが必要なくなった場合は、[詳細] タブを開き、[Desired capacity (希望するキャパシティ)] を 1 減らすことができます。

ステップ 6: クリーンアップする

このチュートリアル専用に作成したリソースで作業が完了したら、次の手順に従ってリソースを削除します。

ライフサイクルフックを削除するには

1. Amazon EC2 コンソールで Auto Scaling グループのページを開きます。
2. Auto Scaling グループの横にあるチェックボックスを選択します。
3. [Instance management (インスタンス管理)] タブの [Lifecycle hooks (ライフサイクルフック)] で、[lifecycle hook (ライフサイクルフック)] を選択します。(`LogAutoScalingEvent-hook`)
4. [アクション]、[削除] の順に選択します。
5. 確認のために、もう一度 [削除] を選択します。

Amazon EventBridge ルールを削除するには

1. Amazon EventBridge コンソールでルールページを開きます。
2. [Event bus (イベントバス)] で、ルール (`Default`) に関連付けられているイベントバスを選択します。
3. `LogAutoScalingEvent-rule` ルールの横にあるチェックボックスをオンにします。
4. [削除] を選択します。
5. 確認を求められたら、ルールの名前を入力し、[Delete] (削除) を選択します。

サンプル関数の使用が終了したら、削除します。関数のログを保存するためのロググループや、作成した実行ロールや許可ポリシーも削除できます。

Lambda 関数を削除するには

1. Lambda コンソールで [Functions (関数)] ページを開きます。
2. 関数 (`LogAutoScalingEvent`) を選択します。
3. [アクション]、[削除] の順に選択します。
4. 確認を求められたら、**`delete`** を入力して指定した関数の削除を確認し、[Delete] (削除) を選択します。

ロググループを削除するには

1. CloudWatch コンソールのロググループページを開きます。
2. 関数のロググループ (/aws/lambda/LogAutoScalingEvent) を選択します。
3. [アクション]、[ロググループの削除] の順にクリックします。
4. ロググループの削除ダイアログボックスで、[削除] をクリックします。

実行ロールを削除するには

1. IAM コンソールの [Roles (ロール)] ページを開きます。
2. 関数のロール (LogAutoScalingEvent-role) を選択します。
3. [削除] を選択します。
4. 確認を求められたら、ロールの名前を入力し、[Delete] (削除) を選択します。

IAM ポリシーを削除するには

1. IAM コンソールのポリシーページを開きます。
2. 作成したポリシーを選択します。(LogAutoScalingEvent-policy)
3. [アクション]、[削除] の順に選択します。
4. 確認を求められたら、ポリシーの名前を入力し、[Delete] (削除) を選択します。

関連リソース

以下の関連トピックは、Auto Scaling グループ内のインスタンスに発生したイベントに基づいて EventBridge ルールを作成する際に役立ちます。

- EventBridge を使用して Auto Scaling イベントを処理する. このセクションでは、スケールイン用のイベントなど、他のユースケースのイベントの例を示します。
- ライフサイクルフックを追加する (コンソール). この手順では、スケールアウト (インスタンスの起動) とスケールイン (インスタンスの終了またはウォームプールへの復帰) の両方にライフサイクルフックを追加する方法を示します。

インスタンスメタデータサービス (IMDS) を使用してインスタンス自体からアクションを呼び出す方法を示すチュートリアルについては、「チュートリアル: インスタンスメタデータを使用してターゲットライフサイクル状態を取得するようにユーザーデータを設定する」を参照してください。

Amazon EC2 Auto Scaling のウォームプール

ウォームプールを使用すると、インスタンスが大量のデータをディスクに書き込む必要があるなど、起動時間が非常に長いアプリケーションのレイテンシーを低減できます。ウォームプールの使用によって、アプリケーションのパフォーマンスを向上させるために、レイテンシーを管理するために Auto Scaling グループを過剰にプロビジョニングする必要がなくなりました。詳細については、ブログ記事 Scaling your applications faster with EC2 Auto Scaling Warm Pools (EC2 Auto Scaling ウォームプールを使用してアプリケーションをより高速にスケーリングする) をご覧ください。

> **Important**
>
> 必要のないときにウォームプールを作成すると、不要なコストが発生する可能性があります。最初の起動時にアプリケーションに顕著なレイテンシーの問題が発生しない場合は、おそらくウォームプールを使用する必要はありません。

トピック

- 主要概念
- 前提条件
- ウォームプール内のインスタンスを更新する
- 関連リソース
- 制限事項
- ウォームプールでライフサイクルフックを使用する
- Auto Scaling グループのためにウォームプールを作成する
- ヘルスチェックのステータスとヘルスチェックの失敗理由を表示する
- を使用したウォームプールの作成と管理の例 AWS CLI

主要概念

開始する前に、以下の主要概念を理解してください。

ウォームプール

ウォームプールは、Auto Scaling グループに接続された初期化済みの EC2 インスタンスのプールです。アプリケーションがスケールアウトする必要があるときはいつでも、Auto Scaling グルー

プはウォームプールに描画して、新しい希望する容量を満たすことができます。これにより、インスタンスがアプリケーショントラフィックを迅速化し、スケールアウトイベントへの応答を高速化できるようになります。インスタンスは、ウォームプールから離れたときに、グループの希望する容量にカウントされます。これは、ウォームスタートとして知られています。

インスタンスがウォームプールにある間、スケーリングポリシーは、InService 状態のインスタンスのメトリクス値がスケーリングポリシーのアラーム上限しきい値 (ターゲット追跡スケーリングポリシーのターゲット使用率と同じ値) を超える場合のみスケールアウトします。

ウォームプールのサイズ

デフォルトでは、ウォームプールのサイズは、Auto Scaling グループの最大容量と希望する容量の数値の差として計算されます。例えば、Auto Scaling グループの希望する容量が 6 で、最大容量が 10 の場合、ウォームプールを最初にセットアップし、プールが初期化されるときに、ウォームプールのサイズは 4 になります。

ウォームプールの最大容量を個別に指定するには、カスタム仕様 (MaxGroupPreparedCapacity) オプションを使用して、グループの現在の容量よりも大きいカスタム値を設定します。カスタム値を指定すると、ウォームプールのサイズは、カスタム値とグループの現在の希望する容量の差として計算されます。例えば、Auto Scaling グループの希望する容量が 6 の場合、最大容量が 20 の場合、カスタム値が 8 の場合、ウォームプールを最初にセットアップし、プールを初期化するときに、ウォームプールのサイズは 2 になります。

大規模な Auto Scaling グループでウォームプールを使用するコスト上の利点を管理する場合にのみ、カスタム仕様 (MaxGroupPreparedCapacity) オプションを使用する必要がある場合があります。例えば、インスタンス 1,000 個、最大容量 1,500 個 (トラフィックの急増時に追加の容量を提供するため)、およびインスタンス 100 個のウォームプールがある Auto Scaling グループは、ウォームプール内に将来使用するインスタンスを 500 個予約しておくよりも、目標の達成に役立つ場合があります。

ウォームプールサイズの最小サイズ

最小サイズ設定を使用して、ウォームプール内で維持するインスタンスの最小数を静的に設定することを検討してください。デフォルトでは最小サイズは設定されていません。

ウォームプールインスタンスの状態

ウォームプール内のインスタンスは、次の 3 つの状態のいずれかで保持できます: Stopped、Running、Hibernated。インスタンスをStopped状態で保持することは、コストを最小限に抑えるための効果的な方法です。停止したインスタンスでは、使用したボリュームとインスタンスにアタッチされた Elastic IP アドレスの分だけ料金が発生します。

または、インスタンスを Hibernated 状態に保持して、メモリコンテンツ (RAM) を削除せずにインスタンスを停止することができます。インスタンスが休止状態になると、RAM コンテンツを Amazon EBS ルートボリュームに保存するようオペレーティングシステムに通知されます。インスタンスを再起動すると、ルートボリュームは以前の状態に復元され、RAM コンテンツがリロードされます。インスタンスが休止状態になっている間は、RAM コンテンツのストレージ、インスタンスにアタッチされた Elastic IP アドレスなどの EBS ボリュームに対してのみ料金が発生します。

ウォームプール内のインスタンスを Running 状態にしておくことも可能ですが、不必要な料金の発生避けるためにも、そうしておかないことを強くお勧めします。インスタンスを停止または休止状態にしておくと、インスタンス自体のコストが削減されます。インスタンスの料金は、インスタンスが実行されている場合にのみ発生します。

ライフサイクルフック

ライフサイクルフックを使用して、インスタンスに対してカスタムアクションを実行できるように、インスタンスを待機状態にします。カスタムアクションは、インスタンスの起動時または終了前に実行されます。

ウォームプール設定では、ライフサイクルフックは、初期化が完了するまで、スケールアウトイベント中にインスタンスが停止または休止されたり、稼働したりするのを遅延させます。ライフサイクルフックなしで Auto Scaling グループにウォームプールを追加すると、初期化の完了までに長い時間がかかるインスタンスは停止または休止状態になり、準備が整う前にスケールアウトイベントが開始する可能性があります。

インスタンスの再利用ポリシー

デフォルトでは、Amazon EC2 Auto Scaling は、Auto Scaling グループがスケールインするとインスタンスを終了します。次に、新しいインスタンスをウォームプールで起動し、終了したインスタンスを置換します。

置換する代わりに、インスタンスをウォームプールに戻す場合は、インスタンスの再利用ポリシーを指定します。これにより、アプリケーショントラフィックを処理するように設定されたインスタンスを再利用できます。ウォームプールが過剰プロビジョニングされないように、Amazon EC2 Auto Scaling はウォームプール内のインスタンスを終了し、その設定に基づいて、必要以上に大きくなったときにそのサイズを減らすことができます。ウォームプール内のインスタンスを終了する際に、Amazon EC2 Auto Scaling はデフォルトの終了ポリシーを使用して、最初に終了するインスタンスを選択します。

> **Important**
>
> スケールイン時にインスタンスを休止し、Auto Scaling グループに既存のインスタンスがある場合は、インスタンスの休止要件を満たしている必要があります。要件を満たしていない場合、インスタンスがウォームプールに戻ると、休止状態ではなく停止状態にフォールバックします。

> **Note**
>
> 現在、インスタンスの再利用ポリシーを指定できるのは、 AWS CLI または SDK のみです。この機能はコンソールからは利用できません。

前提条件

Auto Scaling グループのためにウォームプールを作成する前に、ライフサイクルフックを使用して新しいインスタンスを適切な初期状態で初期化する方法を決定します。

インスタンスがライフサイクルフックを理由として待機状態にあるときに、インスタンスに対してカスタムアクションを実行するには、次の 2 つのオプションがあります。

- 起動時にインスタンスでコマンドを実行する単純なシナリオでは、Auto Scaling グループの起動テンプレートまたは起動設定の作成時にユーザーデータスクリプトを含めることができます。ユーザーデータスクリプトは、インスタンスの起動時に cloud-init により実行される通常のシェルスクリプトまたは cloud-init ディレクティブです。このスクリプトは、実行されるインスタンスの ID を使用して、インスタンスが次の状態に移行するタイミングを制御することもできます。まだそうしていない場合は、インスタンスメタデータからインスタンスのインスタンス ID を取得するためのスクリプトを更新します。詳細については、「Amazon EC2 ユーザーガイド」の「インスタンスメタデータの取得」を参照してください。 Amazon EC2

  > **Tip**
  >
  > インスタンスの再起動時にユーザーデータスクリプトを実行するには、ユーザーデータを MIME マルチパート形式で指定し、ユーザーデータの `#cloud-config` セクションで以下を指定します。

  ```
  #cloud-config
  ```

```
cloud_final_modules:
 - [scripts-user, always]
```

- インスタンスがウォームプールに出入りするときに何かを実行するなど AWS Lambda 、高度なシナリオでは、Auto Scaling グループのライフサイクルフックを作成し、ライフサイクル通知に基づいてカスタムアクションを実行するようにターゲットサービスを設定できます。詳細については、「サポートされている通知ターゲット」を参照してください。

インスタンス休止のための準備

Hibernated プール状態を使用するように Auto Scaling インスタンスを準備するには、Amazon EC2 ユーザーガイドの休止の前提条件トピックで説明されているように、インスタンスの休止をサポートするために正しく設定された新しい起動テンプレートまたは起動設定を作成します。次に、新しい起動テンプレートまたは起動設定を Auto Scaling グループに関連付けてインスタンスの更新を開始し、以前の起動テンプレートまたは起動設定に関連付けられているインスタンスを置換します。詳細については、「インスタンスの更新を使用して Auto Scaling グループのインスタンスを更新する」を参照してください。

ウォームプール内のインスタンスを更新する

ウォームプール内のインスタンスを更新するには、新しい起動テンプレートまたは起動設定を作成し、それを Auto Scaling グループに関連付けます。新しいインスタンスは、起動テンプレートまたは起動設定で指定された新しい AMI およびその他の更新を使用して起動されますが、既存のインスタンスは影響を受けません。

新しい起動テンプレートまたは起動設定を使用する代替ウォームプールインスタンスを強制的に起動するには、インスタンスの更新を開始してグループのローリング更新を行うことができます。インスタンスの更新は、最初にInServiceインスタンスを置き換えます。その後、ウォームプール内のインスタンスが置き換えられます。詳細については、「インスタンスの更新を使用して Auto Scaling グループのインスタンスを更新する」を参照してください。

関連リソース

ウォームプールのライフサイクルフックの例については、GitHubリポジトリを参照してください。

制限事項

- 混合インスタンスポリシー を持つ Auto Scaling グループにウォームプールを追加することはできません。また、起動テンプレートまたはスポットインスタンスをリクエストする起動設定を持つ Auto Scaling グループにウォームプールを追加することはできません。
- Amazon EC2 Auto Scaling は、ルートデバイスとして Amazon EBS ボリュームを持つ場合にのみ、インスタンスを `Stopped` または `Hibernated` 状態にすることができます。ルートデバイスにインスタンスストアを使用するインスタンスは停止または休止できません。
- Amazon EC2 Auto Scaling は、Amazon EC2 ユーザーガイドの休止の前提条件トピックに記載されているすべての要件を満たしている場合にのみ、インスタンスを `Hibernated`状態にすることができます。
- スケールアウトイベントがあるときにウォームプールが枯渇した場合、インスタンスは Auto Scaling グループ内に直接起動されます (コールドスタート)。また、アベイラビリティーゾーンが容量不足の場合にコールドスタートが発生する可能性があります。
- ウォームプール内のインスタンスで起動プロセス中に問題が発生し、 `InService`状態にならない場合、インスタンスは起動に失敗したと見なされ、終了します。これは、容量不足エラーやその他の要因など、根本的な原因に関係なく適用されます。
- Amazon Elastic Kubernetes Service (Amazon EKS) マネージドノードグループでウォームプールを使用しようとすると、まだ初期化中のインスタンスが Amazon EKS クラスターに登録される可能性があります。その結果、このクラスターは、インスタンスが停止または休止の準備を行っているときにインスタンスでジョブをスケジュールする場合があります。
- 同様に、Amazon ECS クラスターでウォームプールを使用しようとすると、初期化が完了する前にインスタンスがクラスターに登録される可能性があります。この問題を解決するには、ユーザーデータに特別なエージェント設定変数が含まれる起動テンプレートまたは起動設定を設定する必要があります。詳細については、「Amazon Elastic Container Service デベロッパーガイド」の「Auto Scaling グループでウォームプールを使用する」を参照してください。
- ウォームプールの休止サポートは、以下を除く AWS リージョン Amazon EC2 Auto Scaling と休止が利用可能なすべての商用 で利用できます。
  - アジアパシフィック (ハイデラバード)
  - アジアパシフィック (メルボルン)
  - カナダ西部 (カルガリー)
  - 中国 (北京) リージョン
  - 中国 (寧夏) リージョン
  - 欧州 (スペイン)

- イスラエル (テルアビブ)

ウォームプールでライフサイクルフックを使用する

ウォームプールのインスタンスは、移行ごとに適切なカスタムアクションを作成できるように、独自の独立したライフサイクルを維持します。このライフサイクルは、インスタンスがまだ初期化中、およびサービスを開始する前に、ターゲットサービス (Lambda 関数など) でアクションを呼び出すように設計されています。

> **Note**
>
> ライフサイクルフックと完全なライフサイクルアクションの追加と管理に使用する API オペレーションは変更されません。インスタンスのライフサイクルのみが変更されます。

ライフサイクルフック追加の詳細については、ライフサイクルフックを追加する を参照してください。ライフサイクルアクション完了の詳細については、ライフサイクルアクションを完了する を参照してください。

ウォームプールに入るインスタンスは、次のいずれかの理由でライフサイクルフックが必要になる場合があります。

- 初期化の完了に時間がかかる AMI から EC2 インスタンスを起動する。
- ユーザーデータスクリプトを実行して EC2 インスタンスをブートストラップする。

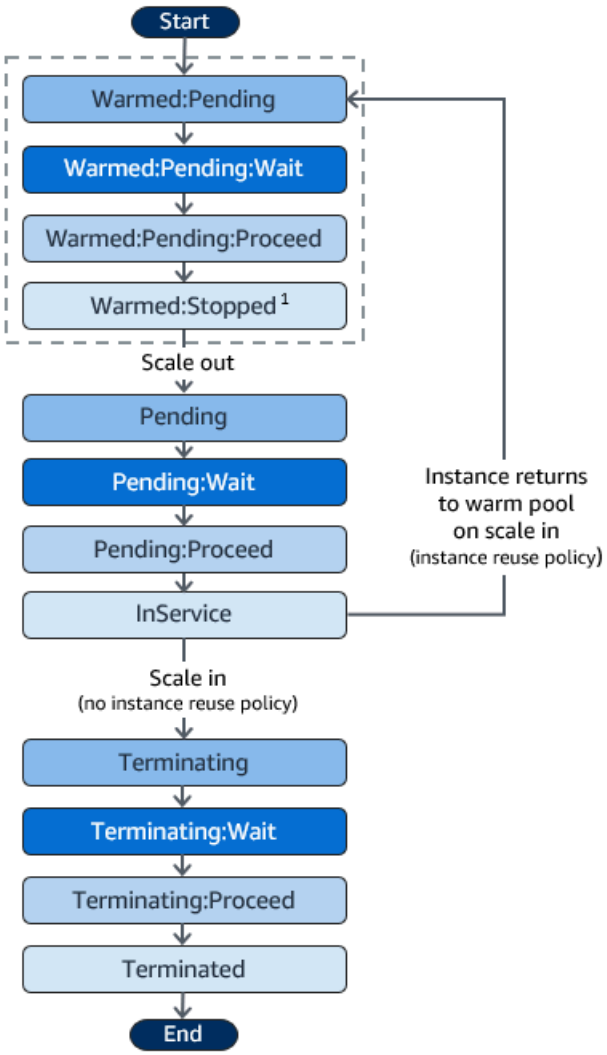
ウォームプールを離れるインスタンスは、次のいずれかの理由でライフサイクルフックが必要になる場合があります。

- EC2 インスタンスの使用準備に、時間を追加することができます。例えば、インスタンスの再起動時に、アプリケーションが正常に動作する前に、開始する必要があるサービスがあるとします。
- キャッシュデータを事前入力して、新しいサーバーが空のキャッシュで起動しないようにすることができます。
- 新しいインスタンスをマネージドインスタンスとして設定管理サービスに登録する場合。

ウォームプール内のインスタンスのライフサイクル状態の移行

オートスケーリングインスタンスは、ライフサイクルの一環として多くの状態に移行します。

次の図表は、ウォームプール使用時のオートスケーリング状態間の移行を示しています。



[1] この状態は、ウォームプールのプール状態設定によって異なります。プール状態が Running に設定されている場合、この状態は代わりに Warmed:Running になります。プール状態が Hibernated に設定されている場合、この状態は代わりに Warmed:Hibernated になります。

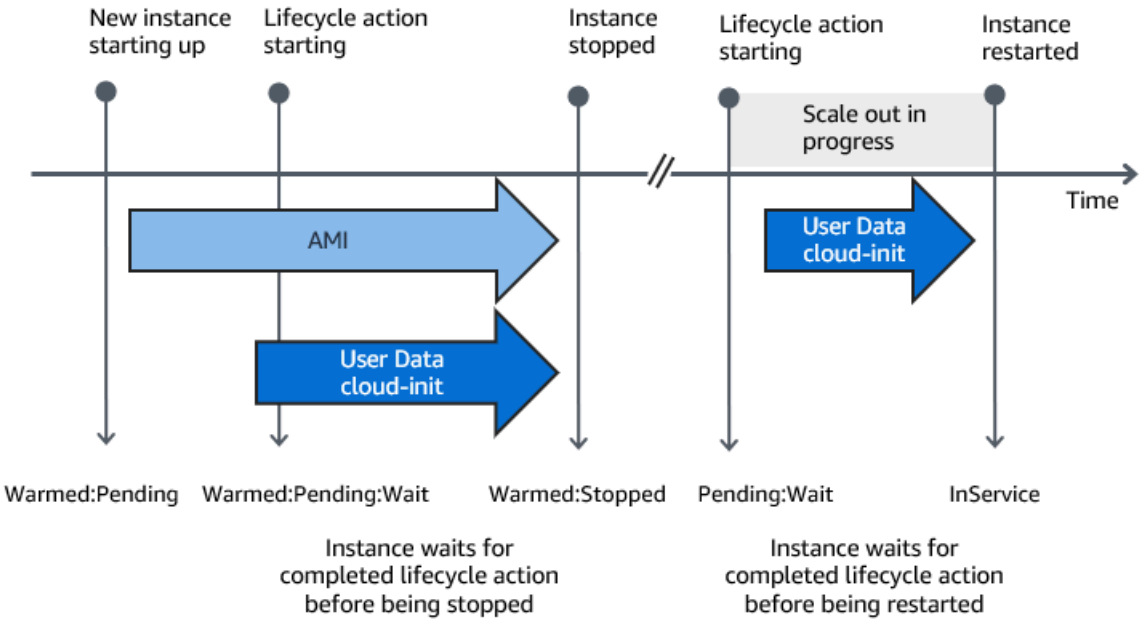
ライフサイクルフックを追加するときは、次の点を考慮してください。

- ライフサイクルフックが autoscaling:EC2_INSTANCE_LAUNCHING ライフサイクルアクションに対して設定されている場合、新しく起動したインスタンスは、Warmed:Pending:Wait 状態に達したときに一時停止してカスタムアクションを実行します。その後、インスタンスが再開して Pending:Wait 状態になったときに再度一時停止してカスタムアクションを実行します。
- ライフサイクルフックが EC2_INSTANCE_TERMINATING ライフサイクルアクションに対して設定されている場合、終了するインスタンスは、Terminating:Wait 状態に達したときに一時停止してカスタムアクションを実行します。ただし、スケールインでインスタンスをウォームプー

ルに戻すインスタンスの再使用ポリシーを指定した場合、ウォームプールに戻るインスタンスは `Warmed:Pending:Wait` 状態で一時停止して `EC2_INSTANCE_TERMINATING` ライフサイクルアクションにカスタムアクションを実行します。

- アプリケーションの要求によってウォームプールが枯渇した場合、Amazon EC2 Auto Scaling はグループが最大容量に到達していない限り、 Auto Scaling グループでインスタンスを直接起動できます。インスタンスがグループ内で直接起動する場合、インスタンスは `Pending:Wait` 状態でのみ一時停止してカスタムアクションを実行します。
- 次の状態に遷移するまでにインスタンスが待機状態を維持する時間を制御するには、complete-lifecycle-action コマンドを使用するようカスタムアクションを設定します。ライフサイクルフックを使用すると、インスタンスは、指定されたライフサイクルアクションが完了したことをユーザーが Amazon EC2 Auto Scaling に通知するか、タイムアウト期間が終了するまで (デフォルトでは 1 時間) 待機状態のままになります。

スケールアウトイベントのフローの概要を次に示します。



インスタンスが待機状態になると、Amazon EC2 Auto Scaling は通知を送信します。これらの通知の例は、このガイドの EventBridge セクションで確認できます。詳細については、「ウォームプールのイベントとパターンの例」を参照してください。

サポートされている通知ターゲット

Amazon EC2 Auto Scaling は、ライフサイクル通知の通知ターゲットとして、次のいずれかを定義するためのサポートを提供します。

- EventBridge ルール
- Amazon SNS トピック
- Amazon SQS キュー

> **Important**
> 起動時にインスタンスを設定するユーザーデータ (cloud-init) スクリプトが起動テンプレートまたは起動設定にある場合、起動または再起動されるインスタンスでカスタムアクションを実行するための通知を受け取る必要はありません。

次のセクションでは、通知ターゲットの設定方法について説明しているドキュメントへのリンクを示します。

EventBridge ルール : Amazon EC2 Auto Scaling がインスタンスを待機状態にするときにコードを実行するには、 EventBridge ルールを作成し、ターゲットとして Lambda 関数を指定します。異なるライフサイクル通知に基づいて異なる Lambda 関数を呼び出すには、複数のルールを作成し、各ルールを特定のイベントパターンおよび Lambda 関数に関連付けます。詳細については、「ウォームプールイベントの EventBridge ルールを作成する」を参照してください。

Amazon SNS トピック: インスタンスが待機状態になったときに通知を受け取るには、Amazon SNS トピックを作成し、メッセージ属性に基づいてライフサイクル通知を異なる方法で配信するように、Amazon SNS メッセージフィルタリングを設定します。詳細については、「Amazon SNS を使用した通知の受信」を参照してください。

Amazon SQS キュー: 関連するコンシューマーがライフサイクル通知を受け取って処理できる配信ポイントを設定するには、Amazon SQS キュー、および SQS キューからのメッセージを処理するキューコンシューマーを作成します。キューコンシューマーに、メッセージ属性に基づいてライフサイクル通知を別々に処理させる場合は、特定の属性が目的の値と一致する際にメッセージを解析、処理するようにキューコンシューマーを設定する必要があります。詳細については、「Amazon SQS を使用した通知の受信」を参照してください。

Auto Scaling グループのためにウォームプールを作成する

このトピックでは、Auto Scaling グループのウォームプールを作成する方法について説明します。

> **Important**
>
> 続行する前に、ウォームプールを作成するための前提条件を満たし、Auto Scaling グループのためにライフサイクルフックが作成されていることを確認します。

ウォームプールを作成する

Auto Scaling グループのためにウォームプールを作成するには、次の手順を実行します。

ウォームプールを作成するには (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. 既存のグループの横にあるチェックボックスをオンにします。

   ページの下部に分割されたペインが開きます。
3. [インスタンス管理] タブを選択します。
4. [ウォームプール] で、ウォームプールの作成を選択します。
5. ウォームプールを設定するには、次の手順を実行します。

   a. ウォームプールインスタンスの状態で、インスタンスがウォームプールに入ったときに、どの状態に移行するかを選択します。デフォルト値は Stopped です。

   b. 最小ウォームプールサイズに、ウォームプールに維持するインスタンスの最小数を入力します。

   c. インスタンスの再利用 では、スケールイン時に再利用 チェックボックスをオンにして、Auto Scaling グループ内のインスタンスがスケールイン時にウォームプールに戻ることができるようにします。

   d. ウォームプールサイズ で、使用可能なオプションのいずれかを選択します。

   - デフォルトの仕様: ウォームプールのサイズは、Auto Scaling グループの最大容量と希望する容量の差によって決まります。このオプションは、ウォームプール管理を効率化します。ウォームプールを作成すると、グループの最大容量を調整するだけで、そのサイズを簡単に更新できます。
   - カスタム仕様 : ウォームプールのサイズは、カスタム値と Auto Scaling グループの希望する容量の差によって決まります。このオプションを使用すると、ウォームプールのサイズをグループの最大容量とは別に柔軟に管理できます。

6. 現在の設定に基づく推定ウォームプールサイズセクションを表示して、デフォルトまたはカスタム仕様がウォームプールのサイズにどのように適用されるかを確認します。ウォームプールのサイズはAuto Scaling グループの希望するキャパシティーによって異なり、グループのスケーリングによって変化します。
7. [作成] を選択します。

ウォームプールを削除する

ウォームプールが不要になった場合は、次の手順にしたがって削除します。

ウォームプールを削除するには (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. 既存のグループの横にあるチェックボックスをオンにします。

   ページの下部に分割されたペインが開きます。
3. [インスタンス管理] タブを選択します。
4. [Warm pool] (ウォームプール) で、[Actions] (アクション)、[Delete] (削除) の順に選択します。
5. 確認を求めるメッセージが表示されたら、[削除] を選択します。

ヘルスチェックのステータスとヘルスチェックの失敗理由を表示する

ヘルスチェックにより、Amazon EC2 Auto Scaling は、インスタンスが異常であり、終了する必要があるタイミングを判断できます。ウォームプールインスタンスが`Stopped`状態の場合、Amazon EBS が`Stopped`インスタンスの可用性を確認して、異常なインスタンスを特定します。これは、`DescribeVolumeStatus`API を使用して、インスタンスにアタッチされている EBS ボリュームのステータスを判別できます。ウォームプールインスタンスの`Running`状態では、EC2 ステータスチェックに依存して、インスタンスの健全性を判断します。ウォームプールインスタンスのヘルスチェック猶予期間はありませんが、Amazon EC2 Auto Scaling はライフサイクルフックが終了するまで、インスタンスのヘルスチェックを開始しません。

インスタンスが異常であることが判明した場合、Amazon EC2 Auto Scaling は自動的に異常インスタンスを削除し、新しいインスタンスを作成して置き換えます。インスタンスは、通常、ヘルスチェックに失敗してから数分以内に終了します。詳細については、「ヘルスチェックが失敗した理由を表示する」を参照してください。

カスタムヘルスチェックもサポートされています。これは、インスタンスの状態を検出し、この情報を Amazon EC2 Auto Scaling に送信できる独自のヘルスチェックシステムがある場合に役立ちます。詳細については、「カスタムヘルスチェック」を参照してください。

Amazon EC2 Auto Scaling コンソールで、ウォームプールインスタンスのステータス (正常または異常) を、ウォームプールインスタンスのステータスを表示できます。 AWS CLI または SDKs のいずれかを使用して、ヘルスステータスを表示することもできます。

ウォームプールインスタンスのステータスを表示するには (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   [Auto Scaling groups] (Auto Scaling グループ) ページの下部にスプリットペインが開きます。
3. [Instance management (インスタンス管理)] タブにある、[Warm pool instances (ウォームプールインスタンス)] の [Lifecycle (ライフサイクル)] 列にインスタンスの状態が表示されます。

   -ヘルスステータス列には、Amazon EC2 Auto Scaling がインスタンスの健全性に対して行った評価が表示されます。

   > **Note**
   >
   > 新しいインスタンスは正常に起動します。ライフサイクルフックが終了するまで、インスタンスの健全性はチェックされません。

ヘルスチェックの失敗の理由を表示するには (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   [Auto Scaling groups] (Auto Scaling グループ) ページの下部にスプリットペインが開きます。
3. [Activity (アクティビティ)] タブの [Activity history (アクティビティ履歴)] の下の [Status (ステータス)] 列に、Auto Scaling グループがインスタンスを正常に起動したか、終了したかが表示されます。

正常でないインスタンスを終了した場合、原因列には、終了の日時、およびヘルスチェックが失敗した理由が表示されます。例えば、「2021-04-01T 21:48:35 Z で、EBS ボリュームのヘルスチェックの失敗に応じて、インスタンスがサービス停止されました」と表示されます。

ウォームプールインスタンスのステータスを表示するには (AWS CLI)

Auto Scaling グループのウォームプールを表示するには、以下を使用します。describe-warm-poolコマンドを実行します。

```
aws autoscaling describe-warm-pool --auto-scaling-group-name my-asg
```

出力例。

```
{
    "WarmPoolConfiguration": {
        "MinSize": 0,
        "PoolState": "Stopped"
    },
    "Instances": [
        {
            "InstanceId": "i-0b5e5e7521cfaa46c",
            "InstanceType": "t2.micro",
            "AvailabilityZone": "us-west-2a",
            "LifecycleState": "Warmed:Stopped",
            "HealthStatus": "Healthy",
            "LaunchTemplate": {
                "LaunchTemplateId": "lt-08c4cd42f320d5dcd",
                "LaunchTemplateName": "my-template-for-auto-scaling",
                "Version": "1"
            }
        },
        {
            "InstanceId": "i-0e21af9dcfb7aa6bf",
            "InstanceType": "t2.micro",
            "AvailabilityZone": "us-west-2a",
            "LifecycleState": "Warmed:Stopped",
            "HealthStatus": "Healthy",
            "LaunchTemplate": {
                "LaunchTemplateId": "lt-08c4cd42f320d5dcd",
                "LaunchTemplateName": "my-template-for-auto-scaling",
                "Version": "1"
```

```
            }
        }
    ]
}
```

ヘルスチェックの失敗理由を表示するには (AWS CLI)

以下の describe-scaling-activities コマンドを実行します。

```
aws autoscaling describe-scaling-activities --auto-scaling-group-name my-asg
```

以下に、応答の例を示します。Descriptionは、Auto Scaling グループがインスタンスを終了したことを示し、Causeは、ヘルスチェックが失敗した理由を示します。

スケーリングアクティビティは、開始時刻順に並べられます。まだ進行中のアクティビティを最初に説明します。

```
{
  "Activities": [
    {
      "ActivityId": "4c65e23d-a35a-4e7d-b6e4-2eaa8753dc12",
      "AutoScalingGroupName": "my-asg",
      "Description": "Terminating EC2 instance: i-04925c838b6438f14",
      "Cause": "At 2021-04-01T21:48:35Z an instance was taken out of service in
 response to EBS volume health check failure.",
      "StartTime": "2021-04-01T21:48:35.859Z",
      "EndTime": "2021-04-01T21:49:18Z",
      "StatusCode": "Successful",
      "Progress": 100,
      "Details": "{\"Subnet ID\":\"subnet-5ea0c127\",\"Availability Zone\":\"us-west-2a
\"...}",
      "AutoScalingGroupARN": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:283179a2-
f3ce-423d-93f6-66bb518232f7:autoScalingGroupName/my-asg"
    },
...
  ]
}
```

を使用したウォームプールの作成と管理の例 AWS CLI

ウォームプールを作成および管理するには、、 AWS Command Line Interface （ AWS CLI ） AWS Management Console、または SDKsを使用します。

次の例では、 AWS CLIを使用してウォームプールを作成、管理する方法を示します。

内容

- 例 1: インスタンスを Stopped 状態に保つ
- 例 2: インスタンスを Running 状態に保つ
- 例 3: インスタンスを Hibernated 状態に保つ
- 例 4: スケールイン時にインスタンスをウォームプールに戻す
- 例 5: ウォームプール内のインスタンスの最小数を指定する
- 例 6: カスタム仕様を使用してウォームプールのサイズを定義する
- 例 7: 絶対的なウォームプールサイズを定義する
- 例 8: ウォームプールを削除する

例 1: インスタンスを **Stopped** 状態に保つ

以下の put-warm-pool の例では、インスタンスを Stopped 状態に保持するウォームプールを作成します。

```
aws autoscaling put-warm-pool --auto-scaling-group-name my-asg /
  --pool-state Stopped
```

例 2: インスタンスを **Running** 状態に保つ

以下の put-warm-pool の例では、インスタンスを Stopped 状態の代わりに Running 状態に保持するウォームプールを作成します。

```
aws autoscaling put-warm-pool --auto-scaling-group-name my-asg /
  --pool-state Running
```

例 3: インスタンスを **Hibernated** 状態に保つ

以下の put-warm-pool の例では、インスタンスを Stopped 状態の代わりに Hibernated 状態に保持するウォームプールを作成します。これにより、メモリコンテンツ (RAM) を削除せずにインスタンスを停止できます。

```
aws autoscaling put-warm-pool --auto-scaling-group-name my-asg /
  --pool-state Hibernated
```

例 4: スケールイン時にインスタンスをウォームプールに戻す

以下の put-warm-pool の例では、インスタンスを Stopped 状態に保持し、--instance-reuse-policy オプションを含むウォームプールを作成します。インスタンスの再利用ポリシー値 '{"ReuseOnScaleIn": true}' は Amazon EC2 Auto Scaling に対し、Auto Scaling グループがスケールインしたときにインスタンスをウォームプールに戻すよう指示します。

```
aws autoscaling put-warm-pool --auto-scaling-group-name my-asg /
  --pool-state Stopped --instance-reuse-policy '{"ReuseOnScaleIn": true}'
```

例 5: ウォームプール内のインスタンスの最小数を指定する

以下の put-warm-pool の例では、4 つ以上のインスタンスを保持できるウォームプールを作成し、トラフィックスパイクの処理に使用可能なインスタンスを 4 つ以上保持します。

```
aws autoscaling put-warm-pool --auto-scaling-group-name my-asg /
  --pool-state Stopped --min-size 4
```

例 6: カスタム仕様を使用してウォームプールのサイズを定義する

デフォルトでは、Amazon EC2 Auto Scaling はAuto Scaling グループの最大容量と希望する容量の差としてウォームプールのサイズを管理します。ただし、 --max-group-prepared-capacityオプションを使用して、ウォームプールのサイズをグループの最大容量とは別に管理できます。

次の put-warm-pool の例では、ウォームプールを作成し、ウォームプールと Auto Scaling グループの両方に同時に存在できるインスタンスの最大数を設定します。グループの希望する容量が 800 の場合、ウォームプールのサイズは、このコマンドの実行後に初期化されるときに最初は 100 になります。

```
aws autoscaling put-warm-pool --auto-scaling-group-name my-asg /
```

```
  --pool-state Stopped --max-group-prepared-capacity 900
```

ウォームプール内のインスタンスの最小数を維持するには、次のように、コマンドを使用して--min-sizeオプションを、含めます。

```
aws autoscaling put-warm-pool --auto-scaling-group-name my-asg /
  --pool-state Stopped --max-group-prepared-capacity 900 --min-size 25
```

例 7: 絶対的なウォームプールサイズを定義する

--max-group-prepared-capacity および --min-size オプションを同じ値に設定すると、ウォームプールは絶対サイズになります。以下の put-warm-pool の例では、10 個のインスタンスのウォームプールサイズを一定に維持するウォームプールを作成します。

```
aws autoscaling put-warm-pool --auto-scaling-group-name my-asg /
  --pool-state Stopped --min-size 10 --max-group-prepared-capacity 10
```

例 8: ウォームプールを削除する

以下の delete-warm-pool コマンドを使用して、ウォームプールを削除します。

```
aws autoscaling delete-warm-pool --auto-scaling-group-name my-asg
```

ウォームプールにインスタンスがある場合、またはスケーリングアクティビティが進行中の場合は、delete-warm-poolコマンドを--force-deleteオプションで使用します。このオプションにより、Amazon EC2 インスタンスおよび未処理のライフサイクルアクションも終了します。

```
aws autoscaling delete-warm-pool --auto-scaling-group-name my-asg --force-delete
```

インスタンスのデタッチまたはアタッチ

Auto Scaling グループからインスタンスをデタッチできます。インスタンスがデタッチされると、そのインスタンスは独立し、単独で管理することも、属していた元のグループとは別に別の Auto Scaling グループにアタッチすることもできます。これは、例えば、すでにアプリケーションを実行している既存のインスタンスを使用してテストを実行する場合に便利です。

このトピックでは、インスタンスをデタッチおよびアタッチする方法について説明します。インスタンスをアタッチするときは、デタッチされたインスタンスではなく既存のインスタンスを使用することもできます。

インスタンスをデタッチして同じグループに再アタッチする代わりに、スタンバイ手順を使用してインスタンスをグループから一時的に削除することをお勧めします。詳細については、「Auto Scaling グループからインスタンスを一時的に削除する」を参照してください。

内容

- インスタンスのデタッチに関する考慮事項
- インスタンスをアタッチする際の考慮事項
- デタッチとアタッチを使用してインスタンスを別のグループに移動する

インスタンスのデタッチに関する考慮事項

インスタンスをデタッチするときは、次の点に注意してください。

- インスタンスをデタッチできるのは、インスタンスが `InService`状態にある場合のみです。
- インスタンスをデタッチした後も、インスタンスの実行が継続され、料金が発生します。不要な料金が発生しないように、デタッチされたインスタンスが不要になったら、再アタッチまたは終了してください。
- デタッチするインスタンスの数に応じて、希望する容量を減らすことができます。容量を減らしない場合、Amazon EC2 Auto Scaling は新しいインスタンスを起動して、デタッチされたインスタンスを置き換え、必要な容量を維持します。
- デタッチするインスタンスの数によって Auto Scaling グループが最小容量を下回る場合は、最小容量を減らす必要があります。
- 必要な容量を減らさずに同じアベイラビリティーゾーンから複数のインスタンスをデタッチすると、`AZRebalance`プロセスを停止しない限り、グループ自体のバランスが再調整されます。詳細については、「Amazon EC2 Auto Scaling プロセスの一時停止と再開」を参照してください。
- ロードバランサーターゲットグループまたは Classic Load Balancer にアタッチした Auto Scaling グループからインスタンスをデタッチすると、インスタンスはロードバランサーから登録解除されます。ロードバランサーで Connection Draining (登録解除の遅延) が有効になっている場合、Amazon EC2 Auto Scaling は未処理のリクエストが完了するまで待機します。

> **Note**
>
> Standby 状態にあるインスタンスをデタッチする場合は注意してください。Standby 状態にしたインスタンスをデタッチしようとすると、他のインスタンスが予期せず終了することがあります。

インスタンスをアタッチする際の考慮事項

インスタンスをアタッチするときは、次の点に注意してください。

- Amazon EC2 Auto Scaling は、アタッチされたインスタンスをグループ自体によって起動されたインスタンスと同じように扱います。つまり、アタッチされたインスタンスは、スケールインイベント中に選択された場合に終了できます。AWSServiceRoleForAutoScaling サービスにリンクされたロールによって付与されたアクセス許可により、Amazon EC2 Auto Scaling はこれを行うことができます。
- インスタンスをアタッチすると、アタッチされるインスタンスの数によって、グループの必要なキャパシティーは増加します。新しいインスタンスを追加した後に必要な容量がグループの最大サイズを超えると、より多くのインスタンスをアタッチするリクエストは失敗します。
- アベイラビリティーゾーン間で分散が不均一になるインスタンスをグループに追加すると、Amazon EC2 Auto Scaling はグループのバランスを再調整して、AZRebalanceプロセスを停止しない限り、均等な分散を再確立します。詳細については、「Amazon EC2 Auto Scaling プロセスの一時停止と再開」を参照してください。
- インスタンスをロードバランサーターゲットグループまたは Classic Load Balancer にアタッチした Auto Scaling グループにアタッチする場合、インスタンスはロードバランサーに登録されます。

アタッチするインスタンスについては、次の条件を満たす必要があります。

- インスタンスが Amazon EC2 で running 状態であること。
- インスタンスの起動に使用する AMI が引き続き存在していること。
- インスタンスは他の Auto Scaling グループのメンバーではありません。
- インスタンスは、Auto Scaling グループで定義されたアベイラビリティーゾーンの 1 つで起動されます。

- Auto Scaling グループにアタッチされたロードバランサーターゲットグループまたは Classic Load Balancer がある場合は、インスタンスおよびロードバランサーは両方とも同じ VPC にある必要があります。

デタッチとアタッチを使用してインスタンスを別のグループに移動する

次のいずれかの手順を使用して、Auto Scaling グループからインスタンスをデタッチし、別の Auto Scaling グループにアタッチします。

デタッチされたインスタンスから新しい Auto Scaling グループを作成するには、「」を参照してください 既存のインスタンスからのパラメータを使用して Auto Scaling グループを作成する (推奨されません。起動設定を作成します)。

Console

Auto Scaling グループからインスタンスをデタッチするには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。
3. [Instance management (インスタンス管理)] タブの [Instances (インスタンス)] でインスタンスを選択し、[Actions (アクション)]、[Detach (デタッチ)] の順に選択します。
4. [インスタンスをデタッチ] ダイアログボックスで、[インスタンスを置き換える] チェックボックスをオンのままにして、置換インスタンスを起動します。必要なキャパシティを減らすには、チェックボックスをオフにします。
5. 確認を求めるプロンプトが表示されたら、指定したインスタンスを Auto Scaling グループから削除することを確認するために **detach** と入力し、[インスタンスのデタッチ] を選択します。

インスタンスを別の Auto Scaling グループにアタッチできるようになりました。

Auto Scaling グループにインスタンスをアタッチするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. (オプション) ナビゲーションペインの [Auto Scaling] で、[Auto Scaling グループ] を選択します。Auto Scaling グループを選択し、Auto Scaling グループの最大サイズが別のインスタンスを追加できる十分な大きさであることを確認します。大きさが十分でない場合は、[詳細] タブで最大キャパシティーを増やします。
3. ナビゲーションペインの [Instances] (インスタンス) で [Instances] (インスタンス) を選択してから、インスタンスを選択します。
4. [Actions]、[Instance Settings]、[Attach to Auto Scaling Group] の順に選択します。
5. [Attach to Auto Scaling Group (Auto Scaling Group にアタッチ)] ページで、[Auto Scaling group (Auto Scalingグループ)] を選択し、[Attach (アタッチ)] を選択します。
6. インスタンスがこの基準を満たさない場合、エラーメッセージとその詳細が表示されます。例えば、インスタンスが Auto Scaling グループと同じアベイラビリティーゾーンにない可能性があります。閉じる を選択して、基準を満たす Auto Scaling グループでもう一度試してください。

AWS CLI

インスタンスをデタッチおよびアタッチするには、次のコマンド例を使用します。各*##########*を独自の情報に置き換えます。

Auto Scaling グループからインスタンスをデタッチするには

1. 現在のインスタンスを記述するには、次の describe-auto-scaling-instances コマンドを使用します。

```
aws autoscaling describe-auto-scaling-instances \
  --query 'AutoScalingInstances[?AutoScalingGroupName==`my-asg`]'
```

次の例は、このコマンドを実行したときに生成される出力を示しています。

グループから削除するインスタンスの ID を書き留めます。この ID は次のステップで必要になります。

```
{
    "AutoScalingInstances": [
        {
            "ProtectedFromScaleIn": false,
            "AvailabilityZone": "us-west-2a",
            "LaunchTemplate": {
```

```
                "LaunchTemplateName": "my-launch-template",
                "Version": "1",
                "LaunchTemplateId": "lt-050555ad16a3f9c7f"
            },
            "InstanceId": "i-05b4f7d5be44822a6",
            "InstanceType": "t3.micro",
            "AutoScalingGroupName": "my-asg",
            "HealthStatus": "HEALTHY",
            "LifecycleState": "InService"
        },
        {
            "ProtectedFromScaleIn": false,
            "AvailabilityZone": "us-west-2a",
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1",
                "LaunchTemplateId": "lt-050555ad16a3f9c7f"
            },
            "InstanceId": "i-0c20ac468fa3049e8",
            "InstanceType": "t3.micro",
            "AutoScalingGroupName": "my-asg",
            "HealthStatus": "HEALTHY",
            "LifecycleState": "InService"
        },
        {
            "ProtectedFromScaleIn": false,
            "AvailabilityZone": "us-west-2a",
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1",
                "LaunchTemplateId": "lt-050555ad16a3f9c7f"
            },
            "InstanceId": "i-0787762faf1c28619",
            "InstanceType": "t3.micro",
            "AutoScalingGroupName": "my-asg",
            "HealthStatus": "HEALTHY",
            "LifecycleState": "InService"
        },
        {
            "ProtectedFromScaleIn": false,
            "AvailabilityZone": "us-west-2a",
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1",
```

```
                "LaunchTemplateId": "lt-050555ad16a3f9c7f"
            },
            "InstanceId": "i-0f280a4c58d319a8a",
            "InstanceType": "t3.micro",
            "AutoScalingGroupName": "my-asg",
            "HealthStatus": "HEALTHY",
            "LifecycleState": "InService"
        }
    ]
}
```

2. 必要な容量を減らしずにインスタンスをデタッチするには、次の detach-instances コマンドを使用します。

```
aws autoscaling detach-instances --instance-ids i-05b4f7d5be44822a6 \
  --auto-scaling-group-name my-asg
```

インスタンスをデタッチして希望する容量を減らすには、 --should-decrement-desired-capacityオプションを含めます。

```
aws autoscaling detach-instances --instance-ids i-05b4f7d5be44822a6 \
  --auto-scaling-group-name my-asg --should-decrement-desired-capacity
```

インスタンスを別の Auto Scaling グループにアタッチできるようになりました。

Auto Scaling グループにインスタンスをアタッチするには

1. インスタンスを別の Auto Scaling グループにアタッチするには、次の attach-instances コマンドを使用します。

```
aws autoscaling attach-instances --instance-ids i-05b4f7d5be44822a6 --auto-
scaling-group-name my-asg-for-testing
```

2. インスタンスをアタッチした後に Auto Scaling グループのサイズを確認するには、次の describe-auto-scaling-groups コマンドを使用します。

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-names my-asg-
for-testing
```

次のレスポンスの例は、グループに実行中のインスタンスが 2 つあり、そのうちの 1 つがアタッチしたインスタンスであることを示しています。


```
{
    "AutoScalingGroups": [
        {
            "AutoScalingGroupName": "my-asg-for-testing",
            "AutoScalingGroupARN": "arn",
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "2",
                "LaunchTemplateId": "lt-050555ad16a3f9c7f"
            },
            "MinSize": 1,
            "MaxSize": 5,
            "DesiredCapacity": 2,
            ...
            "Instances": [
                {
                    "ProtectedFromScaleIn": false,
                    "AvailabilityZone": "us-west-2a",
                    "LaunchTemplate": {
                        "LaunchTemplateName": "my-launch-template",
                        "Version": "1",
                        "LaunchTemplateId": "lt-050555ad16a3f9c7f"
                    },
                    "InstanceId": "i-05b4f7d5be44822a6",
                    "InstanceType": "t3.micro",
                    "HealthStatus": "Healthy",
                    "LifecycleState": "InService"
                },
                {
                    "ProtectedFromScaleIn": false,
                    "AvailabilityZone": "us-west-2a",
                    "LaunchTemplate": {
                        "LaunchTemplateName": "my-launch-template",
                        "Version": "2",
                        "LaunchTemplateId": "lt-050555ad16a3f9c7f"
                    },
                    "InstanceId": "i-00dcdfffdf5175890",
                    "InstanceType": "t3.micro",
                    "HealthStatus": "Healthy",
```

```
                    "LifecycleState": "InService"
                }
            ],
            ...
        }
    ]
}
```

Auto Scaling グループからインスタンスを一時的に削除する

インスタンスを InService 状態から Standby 状態に移行でき、インスタンスを更新またはトラブルシューティングして、インスタンスをサービスに返すことができます。スタンバイ状態のインスタンスはまだ Auto Scaling グループの一部ですが、ロードバランサートラフィックをアクティブに処理しません。

この機能を使用すると、Amazon EC2 Auto Scaling がヘルスチェックの一部として、またはスケールインイベント中にインスタンスを終了することを心配することはなく、インスタンスを停止して起動したり、再起動したりできます。

例えば、起動テンプレートまたは起動設定を変更することで、Auto Scaling グループの Amazon マシンイメージ (AMI) をいつでも変更できます。Auto Scaling グループが起動する後続のインスタンスには、この AMI が使用されます。ただし、Auto Scaling グループは現在稼働中のインスタンスを更新しません。これらのインスタンスを終了し、Amazon EC2 Auto Scaling で置き換えるか、インスタンスの更新機能を使用してインスタンスを終了して置き換えることができます。または、インスタンスをスタンバイ状態にしてソフトウェアを更新し、次にインスタンスをサービスに戻すことができます。

Auto Scaling グループからインスタンスをデタッチすることは、インスタンスをスタンバイ状態にすることと似ています。インスタンスを別のグループにアタッチしたり、スタンドアロン EC2 インスタンスなどのインスタンスを管理して終了させたりする場合は、インスタンスをデタッチすると便利です。詳細については、「インスタンスのデタッチまたはアタッチ」を参照してください。

内容

- スタンバイ状態の仕組み
- 考慮事項
- スタンバイ状態のインスタンスのヘルスステータス
- インスタンスをスタンバイに設定して一時的に削除する

スタンバイ状態の仕組み

Auto Scaling グループからインスタンスを一時的に削除できるように、スタンバイ状態は次のように機能します:

1. ユーザーはインスタンスをスタンバイ状態にします。スタンバイ状態を終了するまで、インスタンスはこの状態のままです。
2. Auto Scaling グループにアタッチされたロードバランサーターゲットグループまたは Classic Load Balancer がある場合、インスタンスはロードバランサーから登録解除されます。Connection Draining がロードバランサーに対して有効になっている場合、Elastic Load Balancingは登録解除プロセス完了前にデフォルトで 300 秒待ちます。これは、処理中のリクエストの完了に役立ちます。
3. インスタンスを更新またはトラブルシューティングできます。
4. スタンバイ状態を終了することにより、インスタンスを稼働状態に戻します。
5. Auto Scaling グループにアタッチされたロードバランサーターゲットグループまたは Classic Load Balancer がある場合、インスタンスはロードバランサーに登録されます。

Auto Scaling グループのインスタンスのライフサイクルの詳細については、「Amazon EC2 Auto Scaling インスタンスのライフサイクル」を参照してください。

考慮事項

インスタンスをスタンバイ状態に移行したり、スタンバイ状態から移行したりする際の考慮事項を次に示します。

- インスタンスをスタンバイ状態にするとき、このオペレーションを通じて必要なキャパシティを減らすことも、同じ値を維持することもできます。
  - Auto Scaling グループの必要なキャパシティを減らさないことを選択した場合、Amazon EC2 Auto Scaling はスタンバイ状態のインスタンスを置き換えるインスタンスを起動します。その目的は、1 つ以上のインスタンスがスタンバイ状態である間にアプリケーションのキャパシティーを維持できるようにすることです。
  - Auto Scaling グループの必要なキャパシティを減らすことを選択すると、スタンバイ状態のインスタンスを置き換えるためのインスタンスを起動できなくなります。
- インスタンスを稼働状態に戻すと、Auto Scaling グループ内のインスタンスの数を反映するために必要なキャパシティが増加します。

- 増加 (および減少) を行うには、新しい必要なキャパシティは、最小グループサイズと最大グループサイズの間にある必要があります。それ以外の場合は、このオペレーションは失敗します。
- インスタンスをスタンバイ状態にした後、またはスタンバイ状態を終了してインスタンスをサービスに戻した後、Auto Scaling グループがアベイラビリティーゾーン間で不均衡であることが判明した場合、Amazon EC2 Auto Scaling は、 `AZRebalance` プロセスを一時停止しない限り、アベイラビリティーゾーンのバランスを再調整して補正します。詳細については、「Amazon EC2 Auto Scaling プロセスの一時停止と再開」を参照してください。
- スタンバイ状態のインスタンスに対して課金されます。

スタンバイ状態のインスタンスのヘルスステータス

Amazon EC2 Auto Scaling はスタンバイ状態にあるインスタンスのヘルスチェックを実行しません。インスタンスがスタンバイ状態にあるとき、スタンバイ状態に移行する前の状態がヘルスステータスに反映されます。Amazon EC2 Auto Scaling は、インスタンスを稼働状態に戻すまで、インスタンスのヘルスチェックを実行しません。

例えば、正常なインスタンスをスタンバイ状態に移行して終了する場合、インスタンスは正常であると Amazon EC2 Auto Scaling がレポートし続けます。スタンバイ状態の終了済みインスタンスをサービスに戻そうとすると、Amazon EC2 Auto Scaling はインスタンスのヘルスチェックを実行し、インスタンスが終了していて異常であると判断して、代替インスタンスを起動します。詳細については、「Auto Scaling グループ内のインスタンスのヘルスチェック」を参照してください。

インスタンスをスタンバイに設定して一時的に削除する

インスタンスをスタンバイ状態にして一時的にサービス停止にするには、次のいずれかの手順を使用します。

Console

インスタンスを一時的に削除するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。
3. [Instance management (インスタンス管理)] タブの [Instances (インスタンス)] で、インスタンスを選択します。

4. [Actions]、[Set to Standby] を選択します。
5. [スタンバイに設定] ダイアログボックスで、[インスタンスを置き換える] チェックボックスをオンのままにして、置換インスタンスを起動します。必要なキャパシティを減らすには、チェックボックスをオフにします。
6. 確認を求めるプロンプトが表示されたら、指定したインスタンスを Standby 状態にすることを確認するために **standby** と入力し、[スタンバイに設定] を選択します。
7. 必要に応じてインスタンスを更新、またはトラブルシューティングできます。終了したら、インスタンスを稼働状態に戻すために次のステップに進みます。
8. インスタンスを選択し、アクション 、 に設定 InServiceを選択します。「Set to InService」ダイアログボックスで、「Set to InService」を選択します。

AWS CLI

Auto Scaling グループからインスタンスを一時的に削除するには、次のコマンド例を使用します。各*##############*を独自の情報に置き換えます。

インスタンスを一時的に削除するには

1. 次の describe-auto-scaling-instances コマンドを使用して、更新するインスタンスを確認します。

```
aws autoscaling describe-auto-scaling-instances \
  --query 'AutoScalingInstances[?AutoScalingGroupName==`my-asg`]'
```

次の例は、このコマンドを実行したときに生成される出力を示しています。

グループから削除するインスタンスの ID を書き留めます。この ID は次のステップで必要になります。

```
{
    "AutoScalingInstances": [
        {
            "ProtectedFromScaleIn": false,
            "AvailabilityZone": "us-west-2a",
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1",
                "LaunchTemplateId": "lt-050555ad16a3f9c7f"
            },
```

```
            "InstanceId": "i-05b4f7d5be44822a6",
            "InstanceId": "t3.micro",
            "AutoScalingGroupName": "my-asg",
            "HealthStatus": "HEALTHY",
            "LifecycleState": "InService"
        },
        ...
    ]
}
```

2. インスタンスを次の enter-standby コマンドを使用して、Standby 状態に移行させます。--should-decrement-desired-capacity オプションは、Auto Scaling グループで代わりのインスタンスが起動されないように希望するキャパシティーを減らします。

```
aws autoscaling enter-standby --instance-ids i-05b4f7d5be44822a6 \
  --auto-scaling-group-name my-asg --should-decrement-desired-capacity
```

以下に、応答の例を示します。

```
{
    "Activities": [
        {
            "ActivityId": "3b1839fe-24b0-40d9-80ae-bcd883c2be32",
            "AutoScalingGroupName": "my-asg",
            "Description": "Moving EC2 instance to Standby:
 i-05b4f7d5be44822a6",
            "Cause": "At 2023-12-15T21:31:26Z instance i-05b4f7d5be44822a6 was
 moved to standby
              in response to a user request, shrinking the capacity from 4 to
 3.",
            "StartTime": "2023-12-15T21:31:26.150Z",
            "StatusCode": "InProgress",
            "Progress": 50,
            "Details": "{\"Subnet ID\":\"subnet-c934b782\",\"Availability Zone
\":\"us-west-2a\"}"
        }
    ]
}
```

3. (オプション) インスタンスがStandby以下の describe-auto-scaling-instances コマンド を使用しているか確認します。

```
aws autoscaling describe-auto-scaling-instances --instance-
ids i-05b4f7d5be44822a6
```

以下に、応答の例を示します。インスタンスのステータスが Standby に設定されました。

```
{
    "AutoScalingInstances": [
        {
            "ProtectedFromScaleIn": false,
            "AvailabilityZone": "us-west-2a",
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1",
                "LaunchTemplateId": "lt-050555ad16a3f9c7f"
            },
            "InstanceId": "i-05b4f7d5be44822a6",
            "InstanceType": "t3.micro",
            "AutoScalingGroupName": "my-asg",
            "HealthStatus": "HEALTHY",
            "LifecycleState": "Standby"
        },
        ...
    ]
}
```

4. 必要に応じてインスタンスを更新、またはトラブルシューティングできます。終了したら、インスタンスを稼働状態に戻すために次のステップに進みます。
5. 次の exit-standby コマンドを使用してインスタンスをサービスに戻します。

```
aws autoscaling exit-standby --instance-ids i-05b4f7d5be44822a6 --auto-scaling-
group-name my-asg
```

以下に、応答の例を示します。

```
{
    "Activities": [
        {
            "ActivityId": "db12b166-cdcc-4c54-8aac-08c5935f8389",
            "AutoScalingGroupName": "my-asg",
```

```
            "Description": "Moving EC2 instance out of Standby:
 i-05b4f7d5be44822a6",
            "Cause": "At 2023-12-15T21:46:14Z instance i-05b4f7d5be44822a6 was
 moved out of standby in
               response to a user request, increasing the capacity from 3 to
 4.",
            "StartTime": "2023-12-15T21:46:14.678Z",
            "StatusCode": "PreInService",
            "Progress": 30,
            "Details": "{\"Subnet ID\":\"subnet-c934b782\",\"Availability Zone
\":\"us-west-2a\"}"
        }
    ]
}
```

6. (オプション) 以下の describe-auto-scaling-instances コマンドを使用して、インスタンスが稼働状態に戻っていることを確認します。

```
aws autoscaling describe-auto-scaling-instances --instance-
ids i-05b4f7d5be44822a6
```

以下に、応答の例を示します。インスタンスのステータスが InService に設定されました。

```
{
    "AutoScalingInstances": [
        {
            "ProtectedFromScaleIn": false,
            "AvailabilityZone": "us-west-2a",
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1",
                "LaunchTemplateId": "lt-050555ad16a3f9c7f"
            },
            "InstanceId": "i-05b4f7d5be44822a6",
            "InstanceType": "t3.micro",
            "AutoScalingGroupName": "my-asg",
            "HealthStatus": "HEALTHY",
            "LifecycleState": "InService"
        },
        ...
    ]
```

```
}
```

Auto Scaling インフラストラクチャを削除する

スケーリングインフラストラクチャを完全に削除するには、次のタスクを実行します。

タスク

- Auto Scaling グループの削除
- (オプション) 起動設定の削除
- (オプション) 起動テンプレートの削除
- (オプション) ロードバランサーとターゲットグループの削除
- (オプション) CloudWatch アラームの削除

Auto Scaling グループの削除

Auto Scaling グループを削除すると、目的の値、最小値、および最大値は 0 に設定されます。その結果、インスタンスは削除されます。インスタンスを削除すると、関連するログまたはデータ、およびインスタンスのすべてのボリュームも削除します。1 つ以上のインスタンスを終了しない場合は、Auto Scaling グループを削除する前にこれらをデタッチすることができます。グループにスケーリングポリシーがある場合、グループを削除すると、ポリシー、基盤となるアラームアクション、および関連付けられたアクションがなくなったアラームが削除されます。

Auto Scaling グループを削除するには (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの隣にあるチェックボックスを選択し、[アクション]、[削除] を選択します。
3. 確認を求められたら、**delete** を入力して指定された Auto Scaling グループの削除を確認し、[Delete] (削除) を選択します。

   [Name (名前)] 列のロードアイコンに、Auto Scaling グループが削除されたことが示されます。[Desired] (希望する)、[Min] (最小)、[Max] (最大) 列には、Auto Scaling グループの `0` インスタンスが表示されます。インスタンスを終了し、グループを削除するには数分かかります。リストを更新して、現在の状態を確認します。

Auto Scaling グループを削除するには (AWS CLI)

次の delete-auto-scaling-group コマンドを使用して Auto Scaling グループを削除します。この操作は、グループに EC2 インスタンスがある場合は機能せず、インスタンスがゼロのグループにのみ適用されます。

```
aws autoscaling delete-auto-scaling-group --auto-scaling-group-name my-asg
```

実行中のインスタンスまたはスケーリングアクティビティがグループにある場合は、delete-auto-scaling-group コマンドを `--force-delete` オプションで使用します。これにより、EC2 インスタンスも終了します。Amazon EC2 Auto Scaling コンソールから Auto Scaling グループを削除すると、コンソールはこの操作を使用してすべての EC2 インスタンスを終了すると同時にグループを削除します。

```
aws autoscaling delete-auto-scaling-group --auto-scaling-group-name my-asg --force-
delete
```

(オプション) 起動設定の削除

今後使用できるように起動設定を保存するには、このステップをスキップします。

起動設定を削除するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左のナビゲーションペインの [Auto Scaling] で、[Auto Scaling グループ] を選択します。
3. ページの上部付近にある [起動設定] を選択します。確認を求めるプロンプトが表示されたら、[起動設定を表示] を選択して、[起動設定] ページを表示することを確認します。
4. 起動設定を選択し、[アクション]、[起動設定の削除] の順に選択します。
5. 確認を求めるメッセージが表示されたら、[削除] を選択します。

起動設定を削除するには (AWS CLI)

以下の delete-launch-configuration コマンドを使用します。

```
aws autoscaling delete-launch-configuration --launch-configuration-name my-launch-
config
```

(オプション) 起動テンプレートの削除

起動テンプレートを削除することも、1 つの起動テンプレートバージョンを削除することもできます。起動テンプレートを削除すると、そのすべてのバージョンが削除されます。

このステップをスキップして、後で使用するために起動テンプレートを維持することもできます。

起動テンプレートを削除するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] の [テンプレートの起動] を選択します。
3. 起動テンプレートを選択し、次のいずれかの操作を行います。
   - [アクション]、[テンプレートの削除] の順に選択します。確認を求められたら、**Delete** を入力して指定した起動テンプレートの削除を確認し、[Delete] (削除) を選択します。
   - [アクション]、[Delete template version (テンプレートのバージョンの削除)] の順に選択します。削除するバージョンを選択し、[削除] を選択します。

起動テンプレートを削除するには (AWS CLI)

次の delete-launch-template コマンドを使用して、テンプレートとそのすべてのバージョンを削除します。

```
aws ec2 delete-launch-template --launch-template-id lt-068f72b72934aff71
```

または、 delete-launch-template-versions コマンドを使用して特定の起動テンプレートのバージョンを削除することもできます。

```
aws ec2 delete-launch-template-versions --launch-template-id lt-068f72b72934aff71 --
versions 1
```

(オプション) ロードバランサーとターゲットグループの削除

Auto Scaling グループが Elastic Load Balancing ロードバランサーに関連付けされていない場合、または今後使用できるようにロードバランサーを維持する場合、このステップをスキップします。

ロードバランサーを削除するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインの [ロードバランシング] で [ロードバランサー] を選択します。
3. ロードバランサーを選択してから、[Actions (アクション)]、[Delete (削除)] の順に選択します。
4. 確認を求めるメッセージが表示されたら、[Yes、Delete] を選択します。

ターゲットグループを削除するには (コンソール)

1. ナビゲーションペインの [ロードバランシング] で [ターゲットグループ] を選択します。
2. ターゲットグループを選択し、[Actions (アクション)]、[Delete (削除)] を選択します。
3. 確認を求めるメッセージが表示されたら、[Yes、Delete] を選択します。

Auto Scaling グループに関連付けられているロードバランサーを削除するには (AWS CLI)

Application Load Balancer および Network Load Balancer では、次の Delete-Load Balancing および delete-target-group コマンドを使用します。

```
aws elbv2 delete-load-balancer --load-balancer-arn my-load-balancer-arn
aws elbv2 delete-target-group --target-group-arn my-target-group-arn
```

Classic Load Balancer を削除するには、次の delete-load-balancer コマンドを使用します。

```
aws elb delete-load-balancer --load-balancer-name my-load-balancer
```

(オプション) CloudWatch アラームの削除

Auto Scaling グループに関連付けられている CloudWatch アラームを削除するには、次の手順を実行します。例えば、ステップスケーリングまたはシンプルスケーリングポリシーに関連するアラームがあるかもしれません。

> **Note**
>
> Auto Scaling グループを削除すると、Amazon EC2 Auto Scaling がターゲット追跡スケーリングポリシーに対して管理する CloudWatch アラームが自動的に削除されます。

Auto Scaling グループが CloudWatch アラームに関連付けられていない場合、または今後使用するためにアラームを保持したい場合は、このステップをスキップできます。

CloudWatch アラームを削除するには (コンソール)

1. https://console.aws.amazon.com/cloudwatch/ で CloudWatch コンソールを開きます。
2. ナビゲーションペインで、[アラーム] を選択します。
3. アラームを選び、[Action (アクション)]、[Delete (削除)] を選択します。
4. 確認を求めるメッセージが表示されたら、[削除] を選択します。

CloudWatch アラームを削除するには (AWS CLI)

delete-alarms コマンドを使用します。1 つ以上のアラームを一度に削除することができます。例えば、次のコマンドを使用して `Step-Scaling-AlarmHigh-AddCapacity` アラームおよび `Step-Scaling-AlarmLow-RemoveCapacity` アラームを削除します。

```
aws cloudwatch delete-alarms --alarm-name Step-Scaling-AlarmHigh-AddCapacity Step-Scaling-AlarmLow-RemoveCapacity
```

AWS SDKs を使用した Auto Scaling グループの作成と管理の例

Auto Scaling グループは、、 AWS Management Console、 AWS SDK AWS CLI、および を使用して作成できます AWS CloudFormation。

次のコード例は、 AWS SDKs を使用して、サポートされているお気に入りのプログラミング言語で Auto Scaling グループを作成、更新、説明、削除する方法を示しています。

内容

- AWS SDK を使用して Auto Scaling グループを作成する
- AWS SDK を使用して Auto Scaling グループを更新する
- AWS SDK を使用して Auto Scaling グループを記述する
- AWS SDK を使用して Auto Scaling グループを削除する

AWS SDK を使用して Auto Scaling グループを作成する

以下のコード例は、`CreateAutoScalingGroup` の使用方法を示しています。

.NET

AWS SDK for .NET

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
/// <summary>
/// Create a new Amazon EC2 Auto Scaling group.
/// </summary>
/// <param name="groupName">The name to use for the new Auto Scaling
/// group.</param>
/// <param name="launchTemplateName">The name of the Amazon EC2 Auto Scaling
/// launch template to use to create instances in the group.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> CreateAutoScalingGroupAsync(
    string groupName,
    string launchTemplateName,
    string availabilityZone)
{
    var templateSpecification = new LaunchTemplateSpecification
    {
        LaunchTemplateName = launchTemplateName,
    };

    var zoneList = new List<string>
        {
            availabilityZone,
        };

    var request = new CreateAutoScalingGroupRequest
    {
        AutoScalingGroupName = groupName,
        AvailabilityZones = zoneList,
        LaunchTemplate = templateSpecification,
        MaxSize = 6,
        MinSize = 1
    };
```

```
        var response = await
_amazonAutoScaling.CreateAutoScalingGroupAsync(request);
        Console.WriteLine($"{groupName} Auto Scaling Group created");
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- API の詳細については、「 API リファレンスCreateAutoScalingGroup」の「」を参照してください。 AWS SDK for .NET

C++

SDK for C++

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::AutoScaling::AutoScalingClient autoScalingClient(clientConfig);

        Aws::AutoScaling::Model::CreateAutoScalingGroupRequest request;
        request.SetAutoScalingGroupName(groupName);
        Aws::Vector<Aws::String> availabilityGroupZones;
        availabilityGroupZones.push_back(
                availabilityZones[availabilityZoneChoice - 1].GetZoneName());
        request.SetAvailabilityZones(availabilityGroupZones);
        request.SetMaxSize(1);
        request.SetMinSize(1);

        Aws::AutoScaling::Model::LaunchTemplateSpecification
launchTemplateSpecification;
        launchTemplateSpecification.SetLaunchTemplateName(templateName);
        request.SetLaunchTemplate(launchTemplateSpecification);
```

```
        Aws::AutoScaling::Model::CreateAutoScalingGroupOutcome outcome =
                autoScalingClient.CreateAutoScalingGroup(request);

        if (outcome.IsSuccess()) {
            std::cout << "Created Auto Scaling group '" << groupName << "'..."
                      << std::endl;
        }
        else if (outcome.GetError().GetErrorType() ==
                 Aws::AutoScaling::AutoScalingErrors::ALREADY_EXISTS_FAULT) {
            std::cout << "Auto Scaling group '" << groupName << "' already
 exists."
                      << std::endl;
        }
        else {
            std::cerr << "Error with AutoScaling::CreateAutoScalingGroup. "
                      << outcome.GetError().GetMessage()
                      << std::endl;

        }
```

- API の詳細については、「 API リファレンスCreateAutoScalingGroup」の「」を参照してください。 AWS SDK for C++

CLI

AWS CLI

例 1: Auto Scaling グループを作成するには

次の create-auto-scaling-group の例では、リージョン内の複数のアベイラビリティーゾーンのサブネット内に Auto Scaling グループを作成します。インスタンスは、指定された起動テンプレートのデフォルトバージョンで起動されます。終了ポリシーやヘルスチェック設定など、他のほとんどの設定にはデフォルトが使用されることに注意してください。

```
aws autoscaling create-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --launch-template LaunchTemplateId=lt-1234567890abcde12 \
    --min-size 1 \
    --max-size 5 \
    --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
```

このコマンドでは何も出力されません。

詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling グループ」を参照してください。

例 2: Application Load Balancer、Network Load Balancer、または Gateway Load Balancer をアタッチするには

この例では、予想されるトラフィックをサポートするロードバランサーのターゲットグループの ARN を指定します。ヘルスチェックタイプは、Elastic Load Balancing がインスタンスを異常として報告したときに、Auto Scaling グループがそのインスタンスを置き換えるよう ELB を指定します。このコマンドは、ヘルスチェックの猶予期間 (600 秒) も定義します。猶予期間は、新しく起動したインスタンスが早期に終了するのを防ぐのに役立ちます。

```
aws autoscaling create-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --launch-template LaunchTemplateId=lt-1234567890abcde12 \
    --target-group-arns arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/943f017f100becff \
    --health-check-type ELB \
    --health-check-grace-period 600 \
    --min-size 1 \
    --max-size 5 \
    --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
```

このコマンドでは何も出力されません。

詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Elastic Load Balancing を使用して Auto Scaling グループ内のインスタンス全体にトラフィックを分散させる」を参照してください。

例 3: プレイスメントグループを指定し、起動テンプレートの最新バージョンを使用するには

この例では、単一のアベイラビリティーゾーン内のプレイスメントグループ内でインスタンスを起動します。これは、HPC ワークロードを使用する低レイテンシーのグループに役立ちます。この例では、グループの最小サイズ、最大サイズ、希望する容量も指定しています。

```
aws autoscaling create-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --launch-template LaunchTemplateId=lt-1234567890abcde12,Version='$Latest' \
    --min-size 1 \
    --max-size 5 \
```

```
    --desired-capacity 3 \
    --placement-group my-placement-group \
    --vpc-zone-identifier "subnet-6194ea3b"
```

このコマンドでは何も出力されません。

詳細については、「Linux インスタンス用 Amazon EC2 ユーザーガイド」の「プレイスメントグループ」を参照してください。

例 4: 単一のインスタンスの Auto Scaling グループを指定し、特定のバージョンの起動テンプレートを使用するには

この例では、単一のインスタンスが強制的に実行されるように、最小容量と最大容量を 1 に設定した Auto Scaling グループを作成します。このコマンドは、既存の ENI の ID が指定されている起動テンプレートの v1 も指定します。eth0 の既存の ENI を指定する起動テンプレートを使用する際は、リクエストにサブネット ID を指定せずに、ネットワークインターフェイスと一致する Auto Scaling グループのアベイラビリティーゾーンを指定する必要があります。

```
aws autoscaling create-auto-scaling-group \
    --auto-scaling-group-name my-asg-single-instance \
    --launch-template LaunchTemplateName=my-template-for-auto-scaling,Version='1'
 \
    --min-size 1 \
    --max-size 1 \
    --availability-zones us-west-2a
```

このコマンドでは何も出力されません。

詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling グループ」を参照してください。

例 5: 別の終了ポリシーを指定するには

この例では、起動設定を使用して Auto Scaling グループを作成し、最も古いインスタンスを最初に終了するように終了ポリシーを設定します。またこのコマンドは、`Role` キーと `WebServer` 値を使用して、グループとインスタンスにタグを適用します。

```
aws autoscaling create-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --launch-configuration-name my-lc \
```

```
    --min-size 1 \
    --max-size 5 \
    --termination-policies "OldestInstance" \
    --tags "ResourceId=my-asg,ResourceType=auto-scaling-
group,Key=Role,Value=WebServer,PropagateAtLaunch=true" \
    --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
```

このコマンドでは何も出力されません。

詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling 終了ポリシーを使用する」を参照してください。

例 6: 起動ライフサイクルフックを指定するには

この例では、インスタンス起動時のカスタムアクションをサポートするライフサイクルフックで Auto Scaling グループを設定します。

```
aws autoscaling create-auto-scaling-group \
    --cli-input-json file://~/config.json
```

config.json ファイルの内容。

```
{
    "AutoScalingGroupName": "my-asg",
    "LaunchTemplate": {
        "LaunchTemplateId": "lt-1234567890abcde12"
    },
    "LifecycleHookSpecificationList": [{
        "LifecycleHookName": "my-launch-hook",
        "LifecycleTransition": "autoscaling:EC2_INSTANCE_LAUNCHING",
        "NotificationTargetARN": "arn:aws:sqs:us-west-2:123456789012:my-sqs-
queue",
        "RoleARN": "arn:aws:iam::123456789012:role/my-notification-role",
        "NotificationMetadata": "SQS message metadata",
        "HeartbeatTimeout": 4800,
        "DefaultResult": "ABANDON"
    }],
    "MinSize": 1,
    "MaxSize": 5,
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782",
    "Tags": [{
        "ResourceType": "auto-scaling-group",
        "ResourceId": "my-asg",
```

```
        "PropagateAtLaunch": true,
        "Value": "test",
        "Key": "environment"
    }]
}
```

このコマンドでは何も出力されません。

詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling のライフサイクルフック」を参照してください。

例 7: 終了ライフサイクルフックを指定するには

次の例は、インスタンス終了時のカスタムアクションをサポートするライフサイクルフックで Auto Scaling グループを設定します。

```
aws autoscaling create-auto-scaling-group \
    --cli-input-json file://~/config.json
```

config.json の内容:

```
{
    "AutoScalingGroupName": "my-asg",
    "LaunchTemplate": {
        "LaunchTemplateId": "lt-1234567890abcde12"
    },
    "LifecycleHookSpecificationList": [{
        "LifecycleHookName": "my-termination-hook",
        "LifecycleTransition": "autoscaling:EC2_INSTANCE_TERMINATING",
        "HeartbeatTimeout": 120,
        "DefaultResult": "CONTINUE"
    }],
    "MinSize": 1,
    "MaxSize": 5,
    "TargetGroupARNs": [
        "arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-
targets/73e2d6bc24d8a067"
    ],
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
}
```

このコマンドでは何も出力されません。

詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling のライフサイクルフック」を参照してください。

例 8: カスタム終了ポリシーを指定するには

この例では、スケールイン時にどのインスタンスを安全に終了できるかを Amazon EC2 Auto Scaling に指示するカスタム Lambda 関数終了ポリシーを指定する Auto Scaling グループを作成します。

```
aws autoscaling create-auto-scaling-group \
    --auto-scaling-group-name my-asg-single-instance \
    --launch-template LaunchTemplateName=my-template-for-auto-scaling \
    --min-size 1 \
    --max-size 5 \
    --termination-policies "arn:aws:lambda:us-
west-2:123456789012:function:HelloFunction:prod" \
    --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
```

このコマンドでは何も出力されません。

詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Lambda を使用したカスタム終了ポリシーを作成する」を参照してください。

- API の詳細については、「 コマンドリファレンスCreateAutoScalingGroup」の「」を参照してください。 AWS CLI

Java

SDK for Java 2.x

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
import software.amazon.awssdk.core.waiters.WaiterResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.autoscaling.AutoScalingClient;
import software.amazon.awssdk.services.autoscaling.model.AutoScalingException;
```

```
import
 software.amazon.awssdk.services.autoscaling.model.CreateAutoScalingGroupRequest;
import
 software.amazon.awssdk.services.autoscaling.model.DescribeAutoScalingGroupsRequest;
import
 software.amazon.awssdk.services.autoscaling.model.DescribeAutoScalingGroupsResponse;
import
 software.amazon.awssdk.services.autoscaling.model.LaunchTemplateSpecification;
import software.amazon.awssdk.services.autoscaling.waiters.AutoScalingWaiter;

/**
 * Before running this SDK for Java (v2) code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateAutoScalingGroup {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <groupName> <launchTemplateName> <serviceLinkedRoleARN>
 <vpcZoneId>

                Where:
                    groupName - The name of the Auto Scaling group.
                    launchTemplateName - The name of the launch template.\s
                    vpcZoneId - A subnet Id for a virtual private cloud (VPC)
 where instances in the Auto Scaling group can be created.
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String groupName = args[0];
        String launchTemplateName = args[1];
        String vpcZoneId = args[2];
        AutoScalingClient autoScalingClient = AutoScalingClient.builder()
                .region(Region.US_EAST_1)
```

```
                .build();

        createAutoScalingGroup(autoScalingClient, groupName, launchTemplateName,
vpcZoneId);
        autoScalingClient.close();
    }

    public static void createAutoScalingGroup(AutoScalingClient
autoScalingClient,
            String groupName,
            String launchTemplateName,
            String vpcZoneId) {

        try {
            AutoScalingWaiter waiter = autoScalingClient.waiter();
            LaunchTemplateSpecification templateSpecification =
LaunchTemplateSpecification.builder()
                    .launchTemplateName(launchTemplateName)
                    .build();

            CreateAutoScalingGroupRequest request =
CreateAutoScalingGroupRequest.builder()
                    .autoScalingGroupName(groupName)
                    .availabilityZones("us-east-1a")
                    .launchTemplate(templateSpecification)
                    .maxSize(1)
                    .minSize(1)
                    .vpcZoneIdentifier(vpcZoneId)
                    .build();

            autoScalingClient.createAutoScalingGroup(request);
            DescribeAutoScalingGroupsRequest groupsRequest =
DescribeAutoScalingGroupsRequest.builder()
                    .autoScalingGroupNames(groupName)
                    .build();

            WaiterResponse<DescribeAutoScalingGroupsResponse> waiterResponse =
waiter
                    .waitUntilGroupExists(groupsRequest);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Auto Scaling Group created");

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
```

```
            System.exit(1);
        }
    }
}
```

- API の詳細については、「 API リファレンスCreateAutoScalingGroup」の「」を参照してください。 AWS SDK for Java 2.x

Kotlin

SDK for Kotlin

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
suspend fun createAutoScalingGroup(
    groupName: String,
    launchTemplateNameVal: String,
    serviceLinkedRoleARNVal: String,
    vpcZoneIdVal: String
) {
    val templateSpecification =
        LaunchTemplateSpecification {
            launchTemplateName = launchTemplateNameVal
        }

    val request =
        CreateAutoScalingGroupRequest {
            autoScalingGroupName = groupName
            availabilityZones = listOf("us-east-1a")
            launchTemplate = templateSpecification
            maxSize = 1
            minSize = 1
            vpcZoneIdentifier = vpcZoneIdVal
            serviceLinkedRoleArn = serviceLinkedRoleARNVal
        }
```

```
    // This object is required for the waiter call.
    val groupsRequestWaiter =
        DescribeAutoScalingGroupsRequest {
            autoScalingGroupNames = listOf(groupName)
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        autoScalingClient.createAutoScalingGroup(request)
        autoScalingClient.waitUntilGroupExists(groupsRequestWaiter)
        println("$groupName was created!")
    }
}
```

- API の詳細については、CreateAutoScalingGroupAWS 「 SDK for Kotlin API リファレンス」の「」を参照してください。

PHP

SDK for PHP

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
    public function createAutoScalingGroup(
        $autoScalingGroupName,
        $availabilityZones,
        $minSize,
        $maxSize,
        $launchTemplateId
    ) {
        return $this->autoScalingClient->createAutoScalingGroup([
            'AutoScalingGroupName' => $autoScalingGroupName,
            'AvailabilityZones' => $availabilityZones,
            'MinSize' => $minSize,
            'MaxSize' => $maxSize,
            'LaunchTemplate' => [
                'LaunchTemplateId' => $launchTemplateId,
```

```
            ],
        ]);
    }
```

- API の詳細については、「 API リファレンスCreateAutoScalingGroup」の「」を参照してください。 AWS SDK for PHP

PowerShell

のツール PowerShell

例 1: この例では、指定された名前と属性を持つ Auto Scaling グループを作成します。デフォルトの希望する容量は最小サイズです。したがって、この Auto Scaling グループは、指定された 2 つのアベイラビリティーゾーンのそれぞれに 1 つずつ、2 つのインスタンスを起動します。

```
New-ASAutoScalingGroup -AutoScalingGroupName my-asg -LaunchConfigurationName my-
lc -MinSize 2 -MaxSize 6 -AvailabilityZone @("us-west-2a", "us-west-2b")
```

- API の詳細については、「 コマンドレットリファレンスCreateAutoScalingGroup」の「」を参照してください。 AWS Tools for PowerShell

Python

SDK for Python (Boto3)

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
class AutoScalingWrapper:
    """Encapsulates Amazon EC2 Auto Scaling actions."""

    def __init__(self, autoscaling_client):
        """
        :param autoscaling_client: A Boto3 Amazon EC2 Auto Scaling client.
        """
```

```
        self.autoscaling_client = autoscaling_client


    def create_group(
        self, group_name, group_zones, launch_template_name, min_size, max_size
    ):
        """
        Creates an Auto Scaling group.

        :param group_name: The name to give to the group.
        :param group_zones: The Availability Zones in which instances can be
created.
        :param launch_template_name: The name of an existing Amazon EC2 launch
template.
                                     The launch template specifies the
configuration of
                                     instances that are created by auto scaling
activities.
        :param min_size: The minimum number of active instances in the group.
        :param max_size: The maximum number of active instances in the group.
        """
        try:
            self.autoscaling_client.create_auto_scaling_group(
                AutoScalingGroupName=group_name,
                AvailabilityZones=group_zones,
                LaunchTemplate={
                    "LaunchTemplateName": launch_template_name,
                    "Version": "$Default",
                },
                MinSize=min_size,
                MaxSize=max_size,
            )
        except ClientError as err:
            logger.error(
                "Couldn't create group %s. Here's why: %s: %s",
                group_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- API の詳細については、 CreateAutoScalingGroup AWS SDK for Python (Boto3) API リファレンスの「」を参照してください。

Rust

SDK for Rust

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
async fn create_group(client: &Client, name: &str, id: &str) -> Result<(), Error>
 {
    client
        .create_auto_scaling_group()
        .auto_scaling_group_name(name)
        .instance_id(id)
        .min_size(1)
        .max_size(5)
        .send()
        .await?;

    println!("Created AutoScaling group");

    Ok(())
}
```

- API の詳細については、 CreateAutoScalingGroup AWS SDK for Rust API リファレンスの「」を参照してください。

混合インスタンスグループ を作成するときに使用できる例については、次のリソースを参照してください。

- AWS SDK for .NET
- AWS SDK for Go
- AWS の SDK JavaScript

- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

AWS SDK を使用して Auto Scaling グループを更新する

以下のコード例は、UpdateAutoScalingGroup の使用方法を示しています。

.NET

AWS SDK for .NET

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
    /// <summary>
    /// Update the capacity of an Auto Scaling group.
    /// </summary>
    /// <param name="groupName">The name of the Auto Scaling group.</param>
    /// <param name="launchTemplateName">The name of the EC2 launch template.</
param>
    /// <param name="maxSize">The maximum number of instances that can be
    /// created for the Auto Scaling group.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> UpdateAutoScalingGroupAsync(
        string groupName,
        string launchTemplateName,
        int maxSize)
    {
        var templateSpecification = new LaunchTemplateSpecification
        {
            LaunchTemplateName = launchTemplateName,
        };

        var groupRequest = new UpdateAutoScalingGroupRequest
        {
            MaxSize = maxSize,
            AutoScalingGroupName = groupName,
```

```
            LaunchTemplate = templateSpecification,
        };

        var response = await
_amazonAutoScaling.UpdateAutoScalingGroupAsync(groupRequest);
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            Console.WriteLine($"You successfully updated the Auto Scaling group
{groupName}.");
            return true;
        }
        else
        {
            return false;
        }
    }
```

- API の詳細については、「 API リファレンスUpdateAutoScalingGroup」の「」を参照してください。 AWS SDK for .NET

C++

SDK for C++

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::AutoScaling::AutoScalingClient autoScalingClient(clientConfig);

        Aws::AutoScaling::Model::UpdateAutoScalingGroupRequest request;
        request.SetAutoScalingGroupName(groupName);
        request.SetMaxSize(3);
```

```
        Aws::AutoScaling::Model::UpdateAutoScalingGroupOutcome outcome =
                autoScalingClient.UpdateAutoScalingGroup(request);

        if (!outcome.IsSuccess()) {
            std::cerr << "Error with AutoScaling::UpdateAutoScalingGroup. "
                      << outcome.GetError().GetMessage()
                      << std::endl;

        }
```

- API の詳細については、「 API リファレンスUpdateAutoScalingGroup」の「」を参照してください。 AWS SDK for C++

CLI

AWS CLI

例 1: Auto Scaling グループのサイズ制限を更新するには

この例は、最小サイズが 2、最大サイズが 10 で、指定された Auto Scaling グループを更新します。

```
aws autoscaling update-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --min-size 2 \
    --max-size 10
```

このコマンドでは何も出力されません。

詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling グループにキャパシティーの制限を設定する」を参照してください。

例 2: Elastic Load Balancing ヘルスチェックを追加し、使用するアベイラビリティーゾーンとサブネットを指定するには

この例は、指定された Auto Scaling グループを更新して、Elastic Load Balancing のヘルスチェックを追加します。またこのコマンドは、複数のアベイラビリティーゾーンのサブネット ID のリストを使用して、`--vpc-zone-identifier` の値も更新します。

```
aws autoscaling update-auto-scaling-group \
```

```
    --auto-scaling-group-name my-asg \
    --health-check-type ELB \
    --health-check-grace-period 600 \
    --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
```

このコマンドでは何も出力されません。

詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Elastic Load Balancing を使用して Auto Scaling グループ内のインスタンス全体にトラフィックを分散させる」を参照してください。

例 3: プレイスメントグループと終了ポリシーを更新するには

この例は、プレイスメントグループと終了ポリシーを更新します。

```
aws autoscaling update-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --placement-group my-placement-group \
    --termination-policies "OldestInstance"
```

このコマンドでは何も出力されません。

詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling グループ」を参照してください。

例 4: 起動テンプレートの最新バージョンを使用するには

この例は、最新の起動テンプレートバージョンを使用するように、指定された Auto Scaling グループを更新します。

```
aws autoscaling update-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --launch-template LaunchTemplateId=lt-1234567890abcde12,Version='$Latest'
```

このコマンドでは何も出力されません。

詳細については、Amazon EC2 Auto Scaling ユーザーガイドの起動テンプレートを参照してください。

例 5: 特定のバージョンの起動テンプレートを使用するには

この例は、最新バージョンやデフォルトバージョンではなく、指定された起動テンプレートのバージョンを使用するように、指定された Auto Scaling グループを更新します。

```
aws autoscaling update-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --launch-template LaunchTemplateName=my-template-for-auto-scaling,Version='2'
```

このコマンドでは何も出力されません。

詳細については、Amazon EC2 Auto Scaling ユーザーガイドの起動テンプレートを参照してください。

例 6: 混合インスタンスポリシーを定義し、容量のリバランスを有効にするには

この例は、指定された Auto Scaling グループを更新して、混合インスタンスポリシーを使用し、容量のリバランスを有効にします。この構造により、スポット容量とオンデマンド容量でグループを指定し、アーキテクチャごとに異なる起動テンプレートを使用できます。

```
aws autoscaling update-auto-scaling-group \
    --cli-input-json file://~/config.json
```

config.json の内容:

```
{
    "AutoScalingGroupName": "my-asg",
    "CapacityRebalance": true,
    "MixedInstancesPolicy": {
        "LaunchTemplate": {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template-for-x86",
                "Version": "$Latest"
            },
            "Overrides": [
                {
                    "InstanceType": "c6g.large",
                    "LaunchTemplateSpecification": {
                        "LaunchTemplateName": "my-launch-template-for-arm",
                        "Version": "$Latest"
                    }
                },
                {
                    "InstanceType": "c5.large"
                },
                {
                    "InstanceType": "c5a.large"
```

```
                }
            ]
        },
        "InstancesDistribution": {
            "OnDemandPercentageAboveBaseCapacity": 50,
            "SpotAllocationStrategy": "capacity-optimized"
        }
    }
}
```

このコマンドでは何も出力されません。

詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。

- API の詳細については、「 コマンドリファレンスUpdateAutoScalingGroup」の「」を参照してください。 AWS CLI

Java

SDK for Java 2.x

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
   public static void updateAutoScalingGroup(AutoScalingClient
autoScalingClient, String groupName,
           String launchTemplateName) {
       try {
           AutoScalingWaiter waiter = autoScalingClient.waiter();
           LaunchTemplateSpecification templateSpecification =
LaunchTemplateSpecification.builder()
                   .launchTemplateName(launchTemplateName)
                   .build();

           UpdateAutoScalingGroupRequest groupRequest =
UpdateAutoScalingGroupRequest.builder()
                   .maxSize(3)
                   .autoScalingGroupName(groupName)
```

```
                    .launchTemplate(templateSpecification)
                    .build();

            autoScalingClient.updateAutoScalingGroup(groupRequest);
            DescribeAutoScalingGroupsRequest groupsRequest =
 DescribeAutoScalingGroupsRequest.builder()
                    .autoScalingGroupNames(groupName)
                    .build();

            WaiterResponse<DescribeAutoScalingGroupsResponse> waiterResponse =
 waiter
                    .waitUntilGroupInService(groupsRequest);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("You successfully updated the auto scaling group
 " + groupName);

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- API の詳細については、「 API リファレンスUpdateAutoScalingGroup」の「」を参照してください。 AWS SDK for Java 2.x

Kotlin

SDK for Kotlin

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
suspend fun updateAutoScalingGroup(
    groupName: String,
    launchTemplateNameVal: String,
    serviceLinkedRoleARNVal: String
) {
```

```
    val templateSpecification =
        LaunchTemplateSpecification {
            launchTemplateName = launchTemplateNameVal
        }

    val groupRequest =
        UpdateAutoScalingGroupRequest {
            maxSize = 3
            serviceLinkedRoleArn = serviceLinkedRoleARNVal
            autoScalingGroupName = groupName
            launchTemplate = templateSpecification
        }

    val groupsRequestWaiter =
        DescribeAutoScalingGroupsRequest {
            autoScalingGroupNames = listOf(groupName)
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        autoScalingClient.updateAutoScalingGroup(groupRequest)
        autoScalingClient.waitUntilGroupExists(groupsRequestWaiter)
        println("You successfully updated the Auto Scaling group  $groupName")
    }
}
```

- API の詳細については、 AWS SDK for Kotlin API リファレンスUpdateAutoScalingGroupの「」を参照してください。

PHP

SDK for PHP

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
    public function updateAutoScalingGroup($autoScalingGroupName, $args)
    {
```

```
        if (array_key_exists('MaxSize', $args)) {
            $maxSize = ['MaxSize' => $args['MaxSize']];
        } else {
            $maxSize = [];
        }
        if (array_key_exists('MinSize', $args)) {
            $minSize = ['MinSize' => $args['MinSize']];
        } else {
            $minSize = [];
        }
        $parameters = ['AutoScalingGroupName' => $autoScalingGroupName];
        $parameters = array_merge($parameters, $minSize, $maxSize);
        return $this->autoScalingClient->updateAutoScalingGroup($parameters);
    }
```

- API の詳細については、「 API リファレンスUpdateAutoScalingGroup」の「」を参照してください。 AWS SDK for PHP

PowerShell

のツール PowerShell

例 1: この例では、指定された Auto Scaling グループの最小サイズと最大サイズを更新します。

```
Update-ASAutoScalingGroup -AutoScalingGroupName my-asg -MaxSize 5 -MinSize 1
```

例 2: この例では、指定された Auto Scaling グループのデフォルトのクールダウン期間を更新します。

```
Update-ASAutoScalingGroup -AutoScalingGroupName my-asg -DefaultCooldown 10
```

例 3: この例では、指定された Auto Scaling グループのアベイラビリティーゾーンを更新します。

```
Update-ASAutoScalingGroup -AutoScalingGroupName my-asg -AvailabilityZone @("us-
west-2a", "us-west-2b")
```

例 4: この例では、指定された Auto Scaling グループを更新して Elastic Load Balancing ヘルスチェックを使用します。

```
Update-ASAutoScalingGroup -AutoScalingGroupName my-asg -HealthCheckType ELB -
HealthCheckGracePeriod 60
```

- API の詳細については、「 コマンドレットリファレンスUpdateAutoScalingGroup」の「」を参照してください。 AWS Tools for PowerShell

Python

SDK for Python (Boto3)

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
class AutoScalingWrapper:
    """Encapsulates Amazon EC2 Auto Scaling actions."""

    def __init__(self, autoscaling_client):
        """
        :param autoscaling_client: A Boto3 Amazon EC2 Auto Scaling client.
        """
        self.autoscaling_client = autoscaling_client


    def update_group(self, group_name, **kwargs):
        """
        Updates an Auto Scaling group.

        :param group_name: The name of the group to update.
        :param kwargs: Keyword arguments to pass through to the service.
        """
        try:
            self.autoscaling_client.update_auto_scaling_group(
                AutoScalingGroupName=group_name, **kwargs
            )
        except ClientError as err:
            logger.error(
                "Couldn't update group %s. Here's why: %s: %s",
                group_name,
```

```
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- API の詳細については、 UpdateAutoScalingGroup AWS SDK for Python (Boto3) API リファレンスの「」を参照してください。

Rust

SDK for Rust

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
async fn update_group(client: &Client, name: &str, size: i32) -> Result<(),
 Error> {
    client
        .update_auto_scaling_group()
        .auto_scaling_group_name(name)
        .max_size(size)
        .send()
        .await?;

    println!("Updated AutoScaling group");

    Ok(())
}
```

- API の詳細については、 UpdateAutoScalingGroup AWS SDK for Rust API リファレンスの「」を参照してください。

AWS SDK を使用して Auto Scaling グループを記述する

以下のコード例は、DescribeAutoScalingGroups の使用方法を示しています。

.NET

AWS SDK for .NET

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
    /// <summary>
    /// Get data about the instances in an Amazon EC2 Auto Scaling group.
    /// </summary>
    /// <param name="groupName">The name of the Amazon EC2 Auto Scaling group.</
param>
    /// <returns>A list of Amazon EC2 Auto Scaling details.</returns>
    public async Task<List<AutoScalingInstanceDetails>>
 DescribeAutoScalingInstancesAsync(
        string groupName)
    {
        var groups = await DescribeAutoScalingGroupsAsync(groupName);
        var instanceIds = new List<string>();
        groups!.ForEach(group =>
        {
            if (group.AutoScalingGroupName == groupName)
            {
                group.Instances.ForEach(instance =>
                {
                    instanceIds.Add(instance.InstanceId);
                });
            }
        });

        var scalingGroupsRequest = new DescribeAutoScalingInstancesRequest
        {
            MaxRecords = 10,
            InstanceIds = instanceIds,
```

```
        };

        var response = await
_amazonAutoScaling.DescribeAutoScalingInstancesAsync(scalingGroupsRequest);
        var instanceDetails = response.AutoScalingInstances;

        return instanceDetails;
    }
```

- API の詳細については、「 API リファレンスDescribeAutoScalingGroups」の「」を参照してください。 AWS SDK for .NET

C++

SDK for C++

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
    Aws::Client::ClientConfiguration clientConfig;
    // Optional: Set to the AWS Region (overrides config file).
    // clientConfig.region = "us-east-1";

Aws::AutoScaling::AutoScalingClient autoScalingClient(clientConfig);

Aws::AutoScaling::Model::DescribeAutoScalingGroupsRequest request;
Aws::Vector<Aws::String> groupNames;
groupNames.push_back(groupName);
request.SetAutoScalingGroupNames(groupNames);

Aws::AutoScaling::Model::DescribeAutoScalingGroupsOutcome outcome =
        client.DescribeAutoScalingGroups(request);

if (outcome.IsSuccess()) {
    autoScalingGroup = outcome.GetResult().GetAutoScalingGroups();
}
```

```
    else {
        std::cerr << "Error with AutoScaling::DescribeAutoScalingGroups. "
                  << outcome.GetError().GetMessage()
                  << std::endl;
    }
```

- API の詳細については、「 API リファレンスDescribeAutoScalingGroups」の「」を参照してください。 AWS SDK for C++

CLI

AWS CLI

例 1: 指定された Auto Scaling グループを記述するには

この例は、指定された Auto Scaling グループを記述します。

```
aws autoscaling describe-auto-scaling-groups \
    --auto-scaling-group-name my-asg
```

出力:

```
{
    "AutoScalingGroups": [
        {
            "AutoScalingGroupName": "my-asg",
            "AutoScalingGroupARN": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:930d940e-891e-4781-
a11a-7b0acd480f03:autoScalingGroupName/my-asg",
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1",
                "LaunchTemplateId": "lt-1234567890abcde12"
            },
            "MinSize": 0,
            "MaxSize": 1,
            "DesiredCapacity": 1,
            "DefaultCooldown": 300,
            "AvailabilityZones": [
                "us-west-2a",
                "us-west-2b",
```

```
                "us-west-2c"
            ],
            "LoadBalancerNames": [],
            "TargetGroupARNs": [],
            "HealthCheckType": "EC2",
            "HealthCheckGracePeriod": 0,
            "Instances": [
                {
                    "InstanceId": "i-06905f55584de02da",
                    "InstanceType": "t2.micro",
                    "AvailabilityZone": "us-west-2a",
                    "HealthStatus": "Healthy",
                    "LifecycleState": "InService",
                    "ProtectedFromScaleIn": false,
                    "LaunchTemplate": {
                        "LaunchTemplateName": "my-launch-template",
                        "Version": "1",
                        "LaunchTemplateId": "lt-1234567890abcde12"
                    }
                }
            ],
            "CreatedTime": "2023-10-28T02:39:22.152Z",
            "SuspendedProcesses": [],
            "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-
c934b782",
            "EnabledMetrics": [],
            "Tags": [],
            "TerminationPolicies": [
                "Default"
            ],
            "NewInstancesProtectedFromScaleIn": false,
            "ServiceLinkedRoleARN":"arn",
            "TrafficSources": []
        }
    ]
}
```

例 2: 指定された最初の 100 個の Auto Scaling グループを記述するには

この例は、指定された複数の Auto Scaling グループを記述します。最大 100 個のグループ名を指定できます。

```
aws autoscaling describe-auto-scaling-groups \
```

```
    --max-items 100 \
    --auto-scaling-group-name "group1" "group2" "group3" "group4"
```

出力例については、例 1 を参照してください。

例 3: 指定されたリージョンの Auto Scaling グループを記述するには

この例では、指定されたリージョンの Auto Scaling グループを最大 75 グループまで記述します。

```
aws autoscaling describe-auto-scaling-groups \
    --max-items 75 \
    --region us-east-1
```

出力例については、例 1 を参照してください。

例 4: 指定された数の Auto Scaling グループを記述するには

特定の数の Auto Scaling グループを返すには、`--max-items` オプションを使用します。

```
aws autoscaling describe-auto-scaling-groups \
    --max-items 1
```

出力例については、例 1 を参照してください。

出力に `NextToken` フィールドが含まれている場合は、さらに多くのグループがあることを示しています。追加のグループを取得するには、次のように、以降の呼び出しで `--starting-token` オプションを使用してこのフィールドの値を使用します。

```
aws autoscaling describe-auto-scaling-groups \
    --starting-token Z3M3LMPEXAMPLE
```

出力例については、例 1 を参照してください。

例 5: 起動設定を使用する Auto Scaling グループを記述するには

この例では、 `--query`オプションを使用して、起動設定を使用する Auto Scaling グループを記述します。

```
aws autoscaling describe-auto-scaling-groups \
    --query 'AutoScalingGroups[?LaunchConfigurationName!=`null`]'
```

出力:


```
[
    {
        "AutoScalingGroupName": "my-asg",
        "AutoScalingGroupARN": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:930d940e-891e-4781-
a11a-7b0acd480f03:autoScalingGroupName/my-asg",
        "LaunchConfigurationName": "my-lc",
        "MinSize": 0,
        "MaxSize": 1,
        "DesiredCapacity": 1,
        "DefaultCooldown": 300,
        "AvailabilityZones": [
            "us-west-2a",
            "us-west-2b",
            "us-west-2c"
        ],
        "LoadBalancerNames": [],
        "TargetGroupARNs": [],
        "HealthCheckType": "EC2",
        "HealthCheckGracePeriod": 0,
        "Instances": [
            {
                "InstanceId": "i-088c57934a6449037",
                "InstanceType": "t2.micro",
                "AvailabilityZone": "us-west-2c",
                "HealthStatus": "Healthy",
                "LifecycleState": "InService",
                "LaunchConfigurationName": "my-lc",
                "ProtectedFromScaleIn": false
            }
        ],
        "CreatedTime": "2023-10-28T02:39:22.152Z",
        "SuspendedProcesses": [],
        "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782",
        "EnabledMetrics": [],
        "Tags": [],
        "TerminationPolicies": [
            "Default"
        ],
        "NewInstancesProtectedFromScaleIn": false,
        "ServiceLinkedRoleARN":"arn",
        "TrafficSources": []
```

```
    }
]
```

詳細については、「 コマンドラインインターフェイスユーザーガイド」の AWS 「CLI 出力のフィルタリング」を参照してください。 AWS

- API の詳細については、「 コマンドリファレンスDescribeAutoScalingGroups」の「」を参照してください。 AWS CLI

Java

SDK for Java 2.x

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.autoscaling.AutoScalingClient;
import software.amazon.awssdk.services.autoscaling.model.AutoScalingException;
import software.amazon.awssdk.services.autoscaling.model.AutoScalingGroup;
import
 software.amazon.awssdk.services.autoscaling.model.DescribeAutoScalingGroupsResponse;
import
 software.amazon.awssdk.services.autoscaling.model.DescribeAutoScalingGroupsRequest;
import software.amazon.awssdk.services.autoscaling.model.Instance;
import java.util.List;

/**
 * Before running this SDK for Java (v2) code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DescribeAutoScalingInstances {
    public static void main(String[] args) {
        final String usage = """
```

```
                Usage:
                    <groupName>

                Where:
                    groupName - The name of the Auto Scaling group.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String groupName = args[0];
        AutoScalingClient autoScalingClient = AutoScalingClient.builder()
                .region(Region.US_EAST_1)
                .build();

        String instanceId = getAutoScaling(autoScalingClient, groupName);
        System.out.println(instanceId);
        autoScalingClient.close();
    }

    public static String getAutoScaling(AutoScalingClient autoScalingClient,
String groupName) {
        try {
            String instanceId = "";
            DescribeAutoScalingGroupsRequest scalingGroupsRequest =
DescribeAutoScalingGroupsRequest.builder()
                    .autoScalingGroupNames(groupName)
                    .build();

            DescribeAutoScalingGroupsResponse response = autoScalingClient
                    .describeAutoScalingGroups(scalingGroupsRequest);
            List<AutoScalingGroup> groups = response.autoScalingGroups();
            for (AutoScalingGroup group : groups) {
                System.out.println("The group name is " +
group.autoScalingGroupName());
                System.out.println("The group ARN is " +
group.autoScalingGroupARN());

                List<Instance> instances = group.instances();
                for (Instance instance : instances) {
                    instanceId = instance.instanceId();
```

```
                }
            }
            return instanceId;
        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- API の詳細については、「 API リファレンスDescribeAutoScalingGroups」の「」を参照してください。 AWS SDK for Java 2.x

Kotlin

SDK for Kotlin

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
suspend fun getAutoScalingGroups(groupName: String) {
    val scalingGroupsRequest =
        DescribeAutoScalingGroupsRequest {
            autoScalingGroupNames = listOf(groupName)
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        val response =
 autoScalingClient.describeAutoScalingGroups(scalingGroupsRequest)
        response.autoScalingGroups?.forEach { group ->
            println("The group name is ${group.autoScalingGroupName}")
            println("The group ARN is ${group.autoScalingGroupArn}")
            group.instances?.forEach { instance ->
                println("The instance id is ${instance.instanceId}")
                println("The lifecycle state is " + instance.lifecycleState)
            }
```

```
        }
    }
}
```

- API の詳細については、 AWS SDK for Kotlin API リファレンスDescribeAutoScalingGroupsの「」を参照してください。

PHP

SDK for PHP

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
    public function describeAutoScalingGroups($autoScalingGroupNames)
    {
        return $this->autoScalingClient->describeAutoScalingGroups([
            'AutoScalingGroupNames' => $autoScalingGroupNames
        ]);
    }
```

- API の詳細については、「 API リファレンスDescribeAutoScalingGroups」の「」を参照してください。 AWS SDK for PHP

PowerShell

のツール PowerShell

例 1: この例では、Auto Scaling グループの名前を一覧表示します。

```
Get-ASAutoScalingGroup | format-table -property AutoScalingGroupName
```

出力:

```
AutoScalingGroupName
```

```
--------------------
my-asg-1
my-asg-2
my-asg-3
my-asg-4
my-asg-5
my-asg-6
```

例 2: この例では、指定された Auto Scaling グループについて説明します。

```
Get-ASAutoScalingGroup -AutoScalingGroupName my-asg-1
```

出力:

```
AutoScalingGroupARN     : arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:930d940e-891e-4781-a11a-7b0acd480
                          f03:autoScalingGroupName/my-asg-1
AutoScalingGroupName    : my-asg-1
AvailabilityZones       : {us-west-2b, us-west-2a}
CreatedTime             : 3/1/2015 9:05:31 AM
DefaultCooldown         : 300
DesiredCapacity         : 2
EnabledMetrics          : {}
HealthCheckGracePeriod  : 300
HealthCheckType         : EC2
Instances               : {my-lc}
LaunchConfigurationName : my-lc
LoadBalancerNames       : {}
MaxSize                 : 0
MinSize                 : 0
PlacementGroup          :
Status                  :
SuspendedProcesses      : {}
Tags                    : {}
TerminationPolicies     : {Default}
VPCZoneIdentifier       : subnet-e4f33493,subnet-5264e837
```

例 3: この例では、指定された 2 つの Auto Scaling グループについて説明します。

```
Get-ASAutoScalingGroup -AutoScalingGroupName @("my-asg-1", "my-asg-2")
```

例 4: この例では、指定された Auto Scaling グループの Auto Scaling インスタンスについて説明します。

```
(Get-ASAutoScalingGroup -AutoScalingGroupName my-asg-1).Instances
```

例 5: この例では、すべての Auto Scaling グループについて説明します。

```
Get-ASAutoScalingGroup
```

例 6: この例では、すべての Auto Scaling グループを 10 個のバッチで記述します。

```
$nextToken = $null
do {
  Get-ASAutoScalingGroup -NextToken $nextToken -MaxRecord 10
  $nextToken = $AWSHistory.LastServiceResponse.NextToken
} while ($nextToken -ne $null)
```

例 7: この例では LaunchTemplate 、指定された Auto Scaling グループの について説明します。この例では、「インスタンスの購入オプション」が「起動テンプレートに準拠する」に設定されていることを前提としています。このオプションが「購入オプションとインスタンスタイプの組み合わせ」に設定されている場合、 LaunchTemplate MixedInstances「ポリシー LaunchTemplate」プロパティを使用してアクセスできます。

```
(Get-ASAutoScalingGroup -AutoScalingGroupName my-ag-1).LaunchTemplate
```

出力:

```
LaunchTemplateId     LaunchTemplateName   Version
----------------     ------------------   -------
lt-06095fd619cb40371 test-launch-template $Default
```

- API の詳細については、「 コマンドレットリファレンスDescribeAutoScalingGroups」の「」を参照してください。 AWS Tools for PowerShell

Python

SDK for Python (Boto3)

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
class AutoScalingWrapper:
    """Encapsulates Amazon EC2 Auto Scaling actions."""

    def __init__(self, autoscaling_client):
        """
        :param autoscaling_client: A Boto3 Amazon EC2 Auto Scaling client.
        """
        self.autoscaling_client = autoscaling_client


    def describe_group(self, group_name):
        """
        Gets information about an Auto Scaling group.

        :param group_name: The name of the group to look up.
        :return: Information about the group, if found.
        """
        try:
            response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[group_name]
            )
        except ClientError as err:
            logger.error(
                "Couldn't describe group %s. Here's why: %s: %s",
                group_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            groups = response.get("AutoScalingGroups", [])
            return groups[0] if len(groups) > 0 else None
```

- API の詳細については、 DescribeAutoScalingGroups AWS SDK for Python (Boto3) API リファレンスの「」を参照してください。

Rust

SDK for Rust

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
async fn list_groups(client: &Client) -> Result<(), Error> {
    let resp = client.describe_auto_scaling_groups().send().await?;

    println!("Groups:");

    let groups = resp.auto_scaling_groups();

    for group in groups {
        println!(
            "Name:  {}",
            group.auto_scaling_group_name().unwrap_or("Unknown")
        );
        println!(
            "Arn:   {}",
            group.auto_scaling_group_arn().unwrap_or("unknown"),
        );
        println!("Zones: {:?}", group.availability_zones(),);
        println!();
    }

    println!("Found {} group(s)", groups.len());

    Ok(())
}
```

- API の詳細については、 DescribeAutoScalingGroups AWS SDK for Rust API リファレンスの「」を参照してください。

AWS SDK を使用して Auto Scaling グループを削除する

以下のコード例は、DeleteAutoScalingGroup の使用方法を示しています。

.NET

AWS SDK for .NET

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

Auto Scaling グループの最小サイズをゼロに更新し、グループ内のすべてのインスタンスを終了して、グループを削除します。

```
    /// <summary>
    /// Try to terminate an instance by its Id.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to terminate.</param>
    /// <returns>Async task.</returns>
    public async Task TryTerminateInstanceById(string instanceId)
    {
        var stopping = false;
        Console.WriteLine($"Stopping {instanceId}...");
        while (!stopping)
        {
            try
            {
                await
_amazonAutoScaling.TerminateInstanceInAutoScalingGroupAsync(
                    new TerminateInstanceInAutoScalingGroupRequest()
                    {
                        InstanceId = instanceId,
                        ShouldDecrementDesiredCapacity = false
                    });
                stopping = true;
```

```
            }
            catch (ScalingActivityInProgressException)
            {
                Console.WriteLine($"Scaling activity in progress for
{instanceId}. Waiting...");
                Thread.Sleep(10000);
            }
        }
    }

    /// <summary>
    /// Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
    /// waits and retries until the group is successfully deleted.
    /// </summary>
    /// <param name="groupName">The name of the group to try to delete.</param>
    /// <returns>Async task.</returns>
    public async Task TryDeleteGroupByName(string groupName)
    {
        var stopped = false;
        while (!stopped)
        {
            try
            {
                await _amazonAutoScaling.DeleteAutoScalingGroupAsync(
                    new DeleteAutoScalingGroupRequest()
                    {
                        AutoScalingGroupName = groupName
                    });
                stopped = true;
            }
            catch (Exception e)
                when ((e is ScalingActivityInProgressException)
                      || (e is Amazon.AutoScaling.Model.ResourceInUseException))
            {
                Console.WriteLine($"Some instances are still running.
Waiting...");
                Thread.Sleep(10000);
            }
        }
    }

    /// <summary>
    /// Terminate instances and delete the Auto Scaling group by name.
```

```
    /// </summary>
    /// <param name="groupName">The name of the group to delete.</param>
    /// <returns>Async task.</returns>
    public async Task TerminateAndDeleteAutoScalingGroupWithName(string
 groupName)
    {
        var describeGroupsResponse = await
 _amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { groupName }
            });
        if (describeGroupsResponse.AutoScalingGroups.Any())
        {
            // Update the size to 0.
            await _amazonAutoScaling.UpdateAutoScalingGroupAsync(
                new UpdateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    MinSize = 0
                });
            var group = describeGroupsResponse.AutoScalingGroups[0];
            foreach (var instance in group.Instances)
            {
                await TryTerminateInstanceById(instance.InstanceId);
            }

            await TryDeleteGroupByName(groupName);
        }
        else
        {
            Console.WriteLine($"No groups found with name {groupName}.");
        }
    }
```

```
    /// <summary>
    /// Delete an Auto Scaling group.
    /// </summary>
    /// <param name="groupName">The name of the Amazon EC2 Auto Scaling group.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
```

```
    public async Task<bool> DeleteAutoScalingGroupAsync(
        string groupName)
    {
        var deleteAutoScalingGroupRequest = new DeleteAutoScalingGroupRequest
        {
            AutoScalingGroupName = groupName,
            ForceDelete = true,
        };

        var response = await
_amazonAutoScaling.DeleteAutoScalingGroupAsync(deleteAutoScalingGroupRequest);
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            Console.WriteLine($"You successfully deleted {groupName}");
            return true;
        }

        Console.WriteLine($"Couldn't delete {groupName}.");
        return false;
    }
```

- API の詳細については、「 API リファレンスDeleteAutoScalingGroup」の「」を参照してください。 AWS SDK for .NET

C++

SDK for C++

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::AutoScaling::AutoScalingClient autoScalingClient(clientConfig);
```

```
            Aws::AutoScaling::Model::DeleteAutoScalingGroupRequest request;
            request.SetAutoScalingGroupName(groupName);

            Aws::AutoScaling::Model::DeleteAutoScalingGroupOutcome outcome =
                    autoScalingClient.DeleteAutoScalingGroup(request);

            if (outcome.IsSuccess()) {
                std::cout << "Auto Scaling group '" << groupName << "' was
 deleted."
                          << std::endl;
            }
            else {
                std::cerr << "Error with AutoScaling::DeleteAutoScalingGroup. "
                          << outcome.GetError().GetMessage()
                          << std::endl;
                result = false;
            }
        }
```

- API の詳細については、「 API リファレンスDeleteAutoScalingGroup」の「」を参照してください。 AWS SDK for C++

CLI

AWS CLI

例 1: 指定された Auto Scaling グループを削除するには

この例は、指定された Auto Scaling グループを削除します。

```
aws autoscaling delete-auto-scaling-group \
    --auto-scaling-group-name my-asg
```

このコマンドでは何も出力されません。

詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling インフラストラクチャを削除する」を参照してください。

例 2: 指定された Auto Scaling グループを強制的に削除するには

グループ内のインスタンスが終了するのを待たずに Auto Scaling グループを削除するには、`--force-delete` オプションを使用します。

```
aws autoscaling delete-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --force-delete
```

このコマンドでは何も出力されません。

詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling インフラストラクチャを削除する」を参照してください。

- API の詳細については、「 コマンドリファレンスDeleteAutoScalingGroup」の「」を参照してください。 AWS CLI

Java

SDK for Java 2.x

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.autoscaling.AutoScalingClient;
import software.amazon.awssdk.services.autoscaling.model.AutoScalingException;
import
 software.amazon.awssdk.services.autoscaling.model.DeleteAutoScalingGroupRequest;

/**
 * Before running this SDK for Java (v2) code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteAutoScalingGroup {
```

```
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <groupName>

                Where:
                    groupName - The name of the Auto Scaling group.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String groupName = args[0];
        AutoScalingClient autoScalingClient = AutoScalingClient.builder()
                .region(Region.US_EAST_1)
                .build();

        deleteAutoScalingGroup(autoScalingClient, groupName);
        autoScalingClient.close();
    }

    public static void deleteAutoScalingGroup(AutoScalingClient
 autoScalingClient, String groupName) {
        try {
            DeleteAutoScalingGroupRequest deleteAutoScalingGroupRequest =
 DeleteAutoScalingGroupRequest.builder()
                    .autoScalingGroupName(groupName)
                    .forceDelete(true)
                    .build();


 autoScalingClient.deleteAutoScalingGroup(deleteAutoScalingGroupRequest);
            System.out.println("You successfully deleted " + groupName);

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- API の詳細については、「 API リファレンスDeleteAutoScalingGroup」の「」を参照してください。 AWS SDK for Java 2.x

Kotlin

SDK for Kotlin

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
suspend fun deleteSpecificAutoScalingGroup(groupName: String) {
    val deleteAutoScalingGroupRequest =
        DeleteAutoScalingGroupRequest {
            autoScalingGroupName = groupName
            forceDelete = true
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        autoScalingClient.deleteAutoScalingGroup(deleteAutoScalingGroupRequest)
        println("You successfully deleted $groupName")
    }
}
```

- API の詳細については、 AWS SDK for Kotlin API リファレンスDeleteAutoScalingGroupの「」を参照してください。

PHP

SDK for PHP

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
    public function deleteAutoScalingGroup($autoScalingGroupName)
    {
        return $this->autoScalingClient->deleteAutoScalingGroup([
            'AutoScalingGroupName' => $autoScalingGroupName,
            'ForceDelete' => true,
        ]);
    }
```

- API の詳細については、「 API リファレンスDeleteAutoScalingGroup」の「」を参照してください。 AWS SDK for PHP

PowerShell

のツール PowerShell

例 1: この例では、実行中のインスタンスがない場合、指定された Auto Scaling グループを削除します。操作を続行する前に確認画面が表示されます。

```
Remove-ASAutoScalingGroup -AutoScalingGroupName my-asg
```

出力:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASAutoScalingGroup (DeleteAutoScalingGroup)" on
 Target "my-asg".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

例 2: Force パラメータを指定した場合、オペレーションが続行する前に確認を求められません。

```
Remove-ASAutoScalingGroup -AutoScalingGroupName my-asg -Force
```

例 3: この例では、指定された Auto Scaling グループを削除し、そのグループに含まれる実行中のインスタンスをすべて終了します。

```
Remove-ASAutoScalingGroup -AutoScalingGroupName my-asg -ForceDelete $true -Force
```

- API の詳細については、「 コマンドレットリファレンスDeleteAutoScalingGroup」の「」を参照してください。 AWS Tools for PowerShell

Python

SDK for Python (Boto3)

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

Auto Scaling グループの最小サイズをゼロに更新し、グループ内のすべてのインスタンスを終了して、グループを削除します。

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
```

```
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def _try_terminate_instance(self, inst_id):
        stopping = False
        log.info(f"Stopping {inst_id}.")
        while not stopping:
            try:
                self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                    InstanceId=inst_id, ShouldDecrementDesiredCapacity=True
                )
                stopping = True
            except ClientError as err:
                if err.response["Error"]["Code"] == "ScalingActivityInProgress":
                    log.info("Scaling activity in progress for %s. Waiting...",
inst_id)
                    time.sleep(10)
```

```
                else:
                    raise AutoScalerError(f"Couldn't stop instance {inst_id}:
{err}.")

    def _try_delete_group(self):
        """
        Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
        the function waits and retries until the group is successfully deleted.
        """
        stopped = False
        while not stopped:
            try:
                self.autoscaling_client.delete_auto_scaling_group(
                    AutoScalingGroupName=self.group_name
                )
                stopped = True
                log.info("Deleted EC2 Auto Scaling group %s.", self.group_name)
            except ClientError as err:
                if (
                    err.response["Error"]["Code"] == "ResourceInUse"
                    or err.response["Error"]["Code"] ==
"ScalingActivityInProgress"
                ):
                    log.info(
                        "Some instances are still running. Waiting for them to
stop..."
                    )
                    time.sleep(10)
                else:
                    raise AutoScalerError(
                        f"Couldn't delete group {self.group_name}: {err}."
                    )

    def delete_group(self):
        """
        Terminates all instances in the group, deletes the EC2 Auto Scaling
group.
        """
        try:
            response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[self.group_name]
            )
            groups = response.get("AutoScalingGroups", [])
```

```
            if len(groups) > 0:
                self.autoscaling_client.update_auto_scaling_group(
                    AutoScalingGroupName=self.group_name, MinSize=0
                )
                instance_ids = [inst["InstanceId"] for inst in groups[0]
["Instances"]]
                for inst_id in instance_ids:
                    self._try_terminate_instance(inst_id)
                self._try_delete_group()
            else:
                log.info("No groups found named %s, nothing to do.",
 self.group_name)
        except ClientError as err:
            raise AutoScalerError(f"Couldn't delete group {self.group_name}:
 {err}.")
```

- API の詳細については、 DeleteAutoScalingGroup AWS SDK for Python (Boto3) API リファレンスの「」を参照してください。

Rust

SDK for Rust

> **Note**
>
> については、「」を参照してください GitHub。AWS コード例リポジトリ で全く同じ例を見つけて、設定と実行の方法を確認してください。

```
async fn delete_group(client: &Client, name: &str, force: bool) -> Result<(),
 Error> {
    client
        .delete_auto_scaling_group()
        .auto_scaling_group_name(name)
        .set_force_delete(if force { Some(true) } else { None })
        .send()
        .await?;

    println!("Deleted Auto Scaling group");
```

```
    Ok(())
}
```

- API の詳細については、 DeleteAutoScalingGroup AWS SDK for Rust API リファレンスの「」を参照してください。

Auto Scaling グループ内のインスタンスをリサイクルする

Amazon EC2 Auto Scaling には、新しい起動テンプレートを新しい Amazon マシンイメージ (AMI) に追加したり、新しいインスタンスタイプを追加したりするなど、更新を行った後に Auto Scaling グループ内の Amazon EC2 インスタンスを置き換えることができる機能が用意されています。 Auto Scaling また、インスタンスを置き換えるのと同じオペレーションに更新を含めるオプションを提供することで、更新を効率化することもできます。

このセクションには、以下の作業に役立つ情報が記載されています。

- インスタンスの更新をスタートして、Auto Scaling グループのインスタンスを置き換えます。
- 希望の設定を説明する特定の更新を宣言し、Auto Scaling グループを希望の設定に更新します。
- 更新済みのインスタンスの置き換えをスキップします。
- チェックポイントを使用して、インスタンスを段階的に更新し、特定のポイントでインスタンスの検証を実行します。
- チェックポイントに到達すると、電子メールで通知を受信します。
- ロールバックを使用して Auto Scaling グループが以前使用していた設定に戻します。
- 何らかの理由でインスタンスの更新が失敗した場合、または指定した Amazon CloudWatch アラームが `ALARM`状態になった場合に、自動的にロールバックします。
- インスタンスの存続期間を制限し、Auto Scaling グループ全体でソフトウェアバージョンおよびインスタンス設定の一貫性を提供します。

内容

- インスタンスの更新を使用して Auto Scaling グループのインスタンスを更新する
- インスタンスの最大存続期間に基づいて Auto Scaling インスタンスを置き換える

インスタンスの更新を使用して Auto Scaling グループのインスタンスを更新する

インスタンスの更新を使用して、Auto Scaling グループのインスタンスを更新できます。この機能は、設定の変更でインスタンスを置き換える必要がある場合、特に Auto Scaling グループに多数のインスタンスが含まれている場合に便利です。

インスタンスの更新が役立つ状況には、次のようなものがあります。

- Auto Scaling グループに新しい Amazon マシンイメージ (AMI) またはユーザーデータスクリプトをデプロイします。変更を含む新しい起動テンプレートを作成し、インスタンスの更新を使用して更新をすぐにロールアウトできます。
- インスタンスを新しいインスタンスタイプに移行して、最新の改善と最適化を活用します。
- Auto Scaling グループの起動設定の使用から起動テンプレートの使用への切り替え。起動設定を起動テンプレートにコピーし、インスタンスの更新を使用してインスタンスを新しいテンプレートに更新できます。起動テンプレートの移行については、「Auto Scaling グループを起動テンプレートに移行する」を参照してください。

内容

- インスタンスの更新の仕組み
- インスタンスの更新のデフォルト値について説明する
- インスタンスの更新の開始
- インスタンスの更新をモニタリングする
- インスタンスの更新のキャンセル
- ロールバックで変更の取り消し
- スキップマッチングでのインスタンスの更新の使用
- インスタンスの更新にチェックポイントを追加する

インスタンスの更新の仕組み

このトピックでは、インスタンスの更新の仕組みについて説明し、インスタンスを効果的に使用するために理解しておく必要がある重要な概念を紹介します。

内容

- 仕組み
- 重要な概念
- ヘルスチェックの猶予期間
- インスタンスタイプの互換性
- 制限事項

仕組み

Auto Scaling グループのインスタンスを更新するには、アプリケーションの最新バージョンと、その他の更新を含む新しい設定を定義できます。次に、インスタンスの更新を開始し、その設定に基づいて既存のインスタンスを新しいインスタンスに置き換えます。

インスタンスの更新を実行するには：

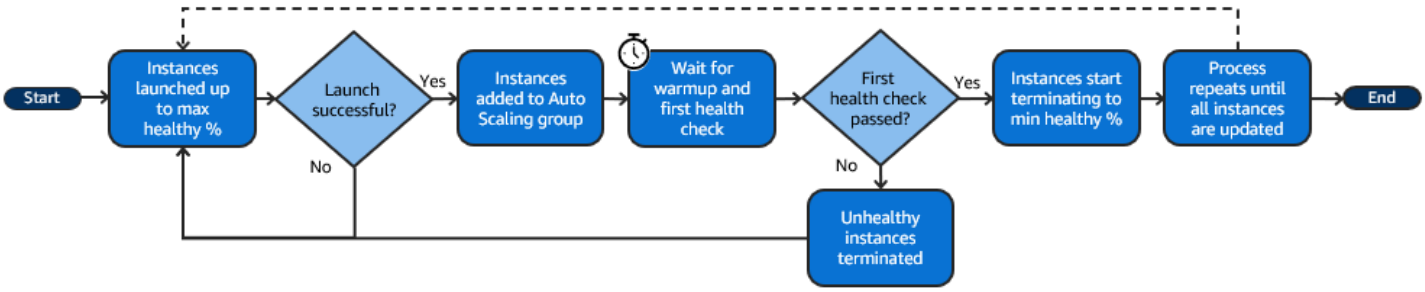
1. 新しい起動テンプレートを作成するか、新しい Amazon マシンイメージ (AMI) などの必要な設定変更で既存のテンプレートを更新します。詳細については、「Auto Scaling グループの起動テンプレートを作成する」を参照してください。
2. Amazon EC2 Auto Scaling コンソール AWS CLI、または SDK を使用してインスタンスの更新を開始します。
   - 作成した新しい起動テンプレートまたは起動テンプレートのバージョンを指定します。これは新しいインスタンスの起動に使用されます。
   - 推奨される正常性の最小パーセンテージと最大パーセンテージを設定します。これにより、同時に置き換えられるインスタンスの数と、古いインスタンスを終了する前に新しいインスタンスを起動するかどうかが制御されます。
   - 次のようなオプションの設定を行います。
     - チェックポイント — 特定の割合の置換後にインスタンスの更新を一時停止して、進行状況を確認します。
     - スキップマッチング — 古いインスタンスを新しい設定と比較し、一致しないインスタンスのみを置き換えます。コンソールからインスタンスの更新を開始すると、スキップマッチングはデフォルトでオンになります。
     - 複数のインスタンスタイプ – 必要な設定の一部として、新規または更新された混合インスタンスポリシーを適用します。

インスタンスの更新が開始されると、Amazon EC2 Auto Scaling は次の処理を行います。

- 最小および最大正常率に基づいて、インスタンスをバッチで置き換えます。
- 最小正常率が 100% に設定されている場合、古いインスタンスを終了する前に新しいインスタンスを起動します。これにより、必要な容量が常に維持されます。
- インスタンスのヘルスステータスをチェックし、より多くのインスタンスが置き換えられる前にウォームアップする時間を確保します。
- 異常が確認されたインスタンスを終了して置き換えます。

- インスタンスの更新が成功した後、新しい設定変更で Auto Scaling グループ設定を自動的に更新します。
- ウォームプール内のInServiceインスタンスの前にインスタンスを置き換えます。

次のフロー図は、最小正常率を 100% に設定した場合の終了前の起動動作を示しています。



> **Note**
>
> インスタンス更新の最小および最大正常率は、インスタンスメンテナンスポリシーを設定していない場合、または既存のポリシーを上書きする必要がある場合にのみ指定する必要があります。詳細については、「インスタンスのメンテナンスポリシー」を参照してください。同様に、インスタンスの更新にインスタンスのウォームアップ期間を指定する必要があるのは、デフォルトのウォームアップを有効にしていない場合、またはデフォルトを上書きする必要がある場合のみです。詳細については、「Auto Scaling グループに対するインスタンスのデフォルトウォームアップを設定する」を参照してください。

重要な概念

開始する前に、次のインスタンス更新のコア概念を理解してください。

最小正常率

最小正常率は、更新を続行できるように、インスタンスの更新中に稼働状態、正常状態、および使用準備が整った状態を維持するのに必要な容量の割合です。例えば、最小正常率が 90% で、最大正常率が 100% の場合、容量の 10% が一度に置き換えられます。新しいインスタンスがヘルスチェックに合格しなかった場合、Amazon EC2 Auto Scaling が終了してインスタンスを置き換えます。インスタンスの更新で正常なインスタンスを起動できない場合、最終的に失敗し、グループの残りの 90% はそのままになります。新しいインスタンスが正常のままでウォームアップ期間を終了すると、Amazon EC2 Auto Scaling は他のインスタンスを引き続き置き換えることができます。

インスタンスの更新では、インスタンスを 1 つずつ、複数ずつ、またはすべてを一度に置き換えることができます。一度に 1 つのインスタンスを置き換えるには、正常性の最小パーセンテージと最大パーセンテージの両方を 100% に設定します。これにより、終了する前にインスタンスの更新の動作が起動するように変更され、グループの容量が希望する容量の 100% を下回ることが防止されます。すべてのインスタンスを一度に置き換えるには、最小正常率を 0% に設定します。

最大正常率

最大正常率は、インスタンスを置き換えるときに Auto Scaling グループが増やすことができる希望する容量の割合です。最小と最大の差は 100 を超えることはできません。範囲を大きくすると、同時に置き換えることができるインスタンスの数が増えます。

インスタンスのウォームアップ

インスタンスのウォームアップとは、新しいインスタンスの状態が `InService` に変更されてから、初期化が完了したとみなされるまでの期間です。インスタンスの更新中、インスタンスがヘルスチェックに合格した場合、新しく起動されたインスタンスが正常であると判断した後、Amazon EC2 Auto Scaling はすぐに次のインスタンスの置き換えに進みません。ウォームアップ期間を待ってから、次のインスタンスの置き換えに進みます。これは、アプリケーションがリクエストに応答するまでにまだ初期化時間が必要な場合に役立ちます。

インスタンスのウォームアップは、デフォルトのインスタンスのウォームアップと同じように機能します。したがって、同じスケーリングに関する考慮事項が適用されます。詳細については、「Auto Scaling グループに対するインスタンスのデフォルトウォームアップを設定する」を参照してください。

必要な設定

必要な設定は、Amazon EC2 Auto Scaling が Auto Scaling グループ全体にデプロイする新しい構成です。例えば、インスタンスに新しい起動テンプレートおよび新しいインスタンスタイプを指定できます。インスタンスの更新中に、Amazon EC2 Auto Scaling は Auto Scaling グループを希望の設定に更新します。インスタンスの更新中にスケールアウト イベントが発生した場合、Amazon EC2 Auto Scaling は、グループの現在の設定ではなく、希望の設定で新しいインスタンスを起動します。インスタンスの更新が成功した後、Amazon EC2 Auto Scaling は Auto Scaling グループの設定を更新し、インスタンスの更新の一環として指定した新しい必要な設定を反映します。

スキップマッチング

スキップマッチングは、既に最新の更新が適用されているインスタンスを無視するように Amazon EC2 Auto Scaling に指示します。これにより、必要以上のインスタンスを置き換える

ことはありません。これは、Auto Scaling グループが特定バージョンの起動テンプレートを使用し、異なるバージョンを使用するインスタンスのみを置き換えることを確認したいときに役立ちます。

チェックポイント

チェックポイントとは、インスタンスの更新が指定した時間のみ一時停止する時点のことです。インスタンスの更新には、複数のチェックポイントを含めることができます。Amazon EC2 Auto Scaling は、チェックポイントごとにイベントを発行します。したがって、チェックポイントに到達したときに通知される Amazon SNS などのターゲットにイベントを送信する EventBridge ルールを追加できます。チェックポイントに到達した後で、デプロイを検証する機会があります。問題が特定された場合、インスタンスの更新をキャンセルまたはロールバックできます。更新プログラムを段階的にデプロイできることは、チェックポイントのキーの利点です。チェックポイントを使用しない場合、ローリング置換は継続的に実行されます。

インスタンスの更新を開始するときに設定できるすべてのデフォルト設定の詳細については、「インスタンスの更新のデフォルト値について説明する」を参照してください。

ヘルスチェックの猶予期間

Amazon EC2 Auto Scaling は、Auto Scaling グループが使用するヘルスチェックのステータスに基づいてインスタンスが正常かどうかを判断します。詳細については、「Auto Scaling グループ内のインスタンスのヘルスチェック」を参照してください。

これらのヘルスチェックをできるだけ早く開始するには、グループのヘルスチェック猶予期間をあまり長く設定しないでください。ただし、Elastic Load Balancing ヘルスチェックがターゲットがリクエストを処理できるかどうかを判断するのに十分な長さに設定します。詳細については、「Auto Scaling グループにヘルスチェックの猶予期間を設定する」を参照してください。

インスタンスタイプの互換性

インスタンスタイプを変更する前に、起動テンプレートで動作することを確認することをお勧めします。これにより、指定した AMI との互換性を確認できます。例えば、準仮想化 (PV) AMI から元のインスタンスを起動したが、ハードウェア仮想マシン (HVM) AMI でのみサポートされている現行世代のインスタンスタイプに変更したいとします。この場合、起動テンプレートで HVM AMI を使用する必要があります。

インスタンスを起動せずにインスタンスタイプの互換性を確認するには、次の例で示すように、「run-instances」コマンドを `--dry-run` オプションとともに使用します。

```
aws ec2 run-instances --launch-template LaunchTemplateName=my-template,Version='1' --
dry-run
```

互換性の決定方法については、「Amazon EC2 ユーザーガイド」の「インスタンスタイプを変更するための互換性」を参照してください。 Amazon EC2

制限事項

- 合計持続時間: インスタンスの更新がインスタンスをアクティブに置き換えることができる最大時間は、14 日間です。
- 加重グループ固有の動作の違い: 混合のインスタンスグループがグループの必要キャパシティ以上のインスタンス分量に設定されている場合、Amazon EC2 Auto Scaling はすべての `InService` インスタンスを一度に置き換えることがあります。このような状況を回避するには、インスタンスの重みを使用するように Auto Scaling グループを設定する トピックの推奨事項に従ってください。Auto Scaling グループで分量を使用するとき、最大の分量よりも大きい必要キャパシティを指定します。
- 1 時間のタイムアウト: インスタンスの更新がスタンバイ状態またはスケールインから保護されているインスタンスを置き換えるために待機しているため、置き換えが続行できない、あるいは新しいインスタンスがヘルスチェックに合格しない場合、Amazon EC2 Auto Scaling は再試行を 1 時間続行します。加えて、問題の解決に役立つステータスメッセージが表示されます。1 時間経っても問題が解決しない場合は、オペレーションは失敗となります。1 時的な問題が発生した場合に、回復する時間を与えることを意図しています。
- ユーザーデータによるコードのデプロイ: スキップマッチングでは、ユーザーデータスクリプトからデプロイされたコードの変更はチェックされません。ユーザーデータを使用して新しいコードをプルし、これらの更新を新しいインスタンスにインストールする場合は、起動テンプレートのバージョンを更新しなくても、スキップマッチングをオフにして、すべてのインスタンスが最新のコードを受け取るようにすることをお勧めします。
- 更新制限 : 目的の設定でインスタンスの更新がアクティブなときに Auto Scaling グループの起動テンプレート、起動設定、または混合インスタンスポリシーを更新しようとすると、リクエストは次の検証エラーで失敗します。 `An active instance refresh with a desired configuration exists. All configuration options derived from the desired configuration are not available for update while the instance refresh is active.`

インスタンスの更新のデフォルト値について説明する

インスタンスの更新を開始する前に、インスタンスの更新に影響するさまざまな設定をカスタマイズできます。一部の設定のデフォルトは、コンソールとコマンドライン (AWS CLI または AWS SDK) のどちらを使用するかによって異なります。

次の表には、インスタンスの更新設定のデフォルト値が一覧表示されています。

設定	AWS CLI または AWS SDK	Amazon EC2 Auto Scaling コンソール
CloudWatch アラーム	無効 (null)	無効
[自動ロールバック]	無効 (`false`)	無効
チェックポイント	無効 (`false`)	無効
[チェックポイントの遅延]	1 時間 (`3600` 秒)	1 時間
インスタンスのウォームアップ	定義されている場合は デフォルトのインスタンスのウォームアップ。定義されていない場合はヘルスチェックの猶予期間。	定義されている場合は デフォルトのインスタンスのウォームアップ。定義されていない場合はヘルスチェックの猶予期間。
最大正常率	インスタンスのメンテナンスポリシーによって異なります。インスタンスのメンテナンスポリシーがない場合、デフォルトで 100% (null) になります。	インスタンスのメンテナンスポリシーによって異なります。インスタンスのメンテナンスポリシーがない場合、デフォルトで 100% (null) になります。
最小正常率	インスタンスのメンテナンスポリシーによって異なります。インスタンスメンテナンスポリシーがない場合、デフォルトで 90% になります。	インスタンスのメンテナンスポリシーによって異なります。インスタンスメンテナンスポリシーがない場合、デフォルトで 90% になります。

設定	AWS CLI または AWS SDK	Amazon EC2 Auto Scaling コンソール
[スケールインで保護されたインスタンス]	待機	Ignore (無視)
スキップマッチング	無効 (`false`)	有効
[スタンバイインスタンス]	待機	Ignore (無視)

各設定の説明は次のとおりです。

CloudWatch アラーム (**`AlarmSpecification`**)

CloudWatch アラーム仕様. CloudWatch alarms を使用すると、問題を特定し、アラームがALARM状態になった場合にオペレーションを失敗させることができます。詳細については、「自動ロールバックでインスタンスの更新を開始」を参照してください。

自動ロールバック (**`AutoRollback`**)

インスタンスの更新が失敗した場合に、Amazon EC2 Auto Scaling が Auto Scaling グループを以前の設定にロールバックするかどうかを制御します。詳細については、「ロールバックで変更の取り消し」を参照してください。

チェックポイント (**`CheckpointPercentages`**)

Amazon EC2 Auto Scaling がインスタンスを段階的に置き換えるかどうかを制御します。これは、すべてのインスタンスを置き換える前にインスタンスの検証を実行する必要がある場合に便利です。詳細については、「インスタンスの更新にチェックポイントを追加する」を参照してください。

チェックポイントの遅延 (**`CheckpointDelay`**)

チェックポイントの後に続行する前に待機する時間 (秒単位)。詳細については、「インスタンスの更新にチェックポイントを追加する」を参照してください。

インスタンスのウォームアップ (**`InstanceWarmup`**)

Amazon EC2 Auto Scaling が新しいインスタンスの初期化が完了したと見なされてから次のインスタンスの置き換えに移行するまでの時間 (秒単位)。Auto Scaling グループのデフォルトのインスタンスウォームアップを既に正しく定義している場合、インスタンスのウォームアップ (デフォルトをオーバーライドする場合を除く) を変更する必要はありません。詳細については、

「Auto Scaling グループに対するインスタンスのデフォルトウォームアップを設定する」を参照してください。

最大正常率 (**`MaxHealthyPercentage`**)

インスタンスを置き換えるときにグループが増やすことができる Auto Scaling グループの希望する容量の割合。

最小正常率 (**`MinHealthyPercentage`**)

オペレーションを続行する前に、稼働中、正常、および使用準備が整っている必要がある Auto Scaling グループの希望する容量の割合。

スケールインで保護されたインスタンス (**`ScaleInProtectedInstances`**)

スケールインから保護されているインスタンスが見つかった場合の Amazon EC2 Auto Scaling の動作を制御します。これらのインスタンスの詳細については、「インスタンスのスケールイン保護を使用する」を参照してください。

Amazon EC2 Auto Scaling は次のオプションを提供します。

- 置き換え (`Refresh`) — スケールインから保護されているインスタンスを置き換えます。
- Ignore (`Ignore`) – スケールインから保護されているインスタンスを無視し、保護されていないインスタンスを引き続き置き換えます。
- Wait (`Wait`) – スケールイン保護を解除するまで 1 時間待ちます。そうしない場合、インスタンスの更新が失敗します。

スキップマッチング (**`SkipMatching`**)

Amazon EC2 Auto Scaling が、目的の設定に一致するインスタンスの置換をスキップするかどうかを制御します。必要な設定が指定されていない場合、インスタンスの更新が開始される前に Auto Scaling グループが使用していた同じ起動テンプレートおよびインスタンスタイプを持つインスタンスの置き換えはスキップされます。詳細については、「スキップマッチングでのインスタンスの更新の使用」を参照してください。

スタンバイインスタンス (**`StandbyInstances`**)

`Standby` 状態のインスタンスが見つかった場合の Amazon EC2 Auto Scaling の動作を制御します。これらのインスタンスの詳細については、「Auto Scaling グループからインスタンスを一時的に削除する」を参照してください。

Amazon EC2 Auto Scaling は次のオプションを提供します。

- 終了 (`Terminate`) – にあるインスタンスを終了しますStandby。

- Ignore (`Ignore`) – にあるインスタンスを無視`Standby`し、 `InService` 状態のインスタンスを置き換え続けます。
- Wait (`Wait`) – インスタンスがサービスに戻るまで 1 時間待ちます。そうしない場合、インスタンスの更新が失敗します。

インスタンスの更新の開始

> **Important**
>
> 進行中のインスタンスの更新をロールバックし、すべての変更を元に戻すことができます。これが実行されるには、インスタンスの更新を開始する前に、Auto Scaling グループがロールバックを使用するための前提条件を満たす必要があります。詳細については、「ロールバックで変更の取り消し」を参照してください。

次の手順は、 AWS Management Console または を使用してインスタンスの更新を開始するのに役立ちます AWS CLI。

インスタンスの更新の開始 (コンソール)

初めてインスタンスの更新を開始する場合は、コンソールにより、利用できる機能とオプションを理解することができます。

コンソールでインスタンスの更新を開始する (基本的な手順)

Auto Scaling グループの混合インスタンスポリシーを事前定義していない場合は、次の手順に従います。混合インスタンスポリシーを事前定義している場合は、コンソールでインスタンスの更新を開始する (混合インスタンスグループ) を参照してインスタンスの更新を開始します。

インスタンスの更新をスタートするには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   Auto Scaling グループページの下部に分割ペインが開きます。
3. [Instance refresh] (インスタンスの更新) タブの [Active instance refresh] (アクティブインスタンスの更新) で、[Start instance refresh] (インスタンスの更新を開始する) を選択します。

4. 可用性設定 では、次の操作を行います。

   a. インスタンス置換方法 の場合：

      - Auto Scaling グループにインスタンスメンテナンスポリシーを設定していない場合、インスタンス置換方法のデフォルト設定は終了と起動です。これは、インスタンスの更新の従来のデフォルト動作です。
      - Auto Scaling グループにインスタンスメンテナンスポリシーを設定すると、インスタンス置換方法 のデフォルト値が提供されます。インスタンスのメンテナンスポリシーを上書きするには、オーバーライド を選択します。上書きは、現在のインスタンスの更新にのみ適用されます。次回インスタンスの更新を開始すると、これらの値はインスタンスメンテナンスポリシーのデフォルトにリセットされます。

      次の手順では、インスタンスの置換方法を更新する方法について説明します。

      i. 次のいずれかのインスタンス置換方法を選択します。

         - 終了前に起動： 既存のインスタンスを終了する前に、新しいインスタンスを最初にプロビジョニングする必要があります。これは、コスト削減よりも可用性を優先するアプリケーションに適しています。
         - の終了と起動: 新しいインスタンスは、既存のインスタンスが終了すると同時にプロビジョニングされます。これは、可用性よりもコスト削減を優先するアプリケーションに適しています。また、現在利用可能な容量よりも多くの容量を起動すべきではないアプリケーションにも適しています。
         - カスタム動作： このオプションを使用すると、インスタンスを置き換えるときに使用可能な容量のカスタム最小範囲と最大範囲を設定できます。これにより、コストと可用性の適切なバランスを実現できます。

      ii. 正常率の設定 では、次のフィールドの一方または両方に値を入力します。有効化フィールドは、インスタンス置換方法 で選択したオプションによって異なります。

         - 最小 : インスタンスの更新を続行するのに必要な最小正常率を設定します。
         - 最大 : インスタンスの更新中に可能な最大正常率を設定します。

      iii. 現在のグループサイズに基づいて置換中の推定一時容量を表示セクションを展開し、最小と最大 の値がグループにどのように適用されるかを確認します。使用される正確な値は、希望する容量値によって異なります。これは、グループがスケールすると変化します。

iv. 無効な置換サイズに対するフォールバック動作の設定セクションを展開し、可用性を優先するために最大正常率を超過するか、最小正常率を超過するかを選択します。

ごく少数のグループでは、デフォルトの Violate min healthy percentage オプションを保持することはお勧めしません。Auto Scaling グループにインスタンスが 1 つしかない場合、インスタンスの更新を開始すると停止する可能性があります。

このステップでは、インスタンスメンテナンスポリシーがまだない Auto Scaling グループを使用している場合のフォールバック動作を設定します。このオプションは使用できず、グループにインスタンスメンテナンスポリシーがある場合に表示されません。このオプションは、Terminate および launch replacement メソッドでのみ使用できます。他の置換方法は、可用性を優先するために最大正常率に違反します。

b. [インスタンスのウォームアップ] には、新しいインスタンスの状態が `InService` に変更されてから初期化が完了するまでの秒数を入力します。Amazon EC2 Auto Scaling は、次のインスタンスを置き換える前にこの時間を待機します。

ウォームアップ中、新しく起動されたインスタンスは Auto Scaling グループの (`CPUUtilization`、`NetworkIn`、`NetworkOut` など) 集計インスタンスのメトリクスにも計上されません。Auto Scaling グループにスケーリング ポリシーを追加した場合、スケーリングアクティビティは並行して実行されます。インスタンスの更新ウォームアップ期間に長い間隔を設定すると、新しく起動されたインスタンスがメトリクスに表示されるまでに時間がかかります。したがって、適切なウォームアップ期間により、Amazon EC2 Auto Scaling が古いメトリクスデータでスケーリングされなくなります。

Auto Scaling グループにデフォルトのインスタンスのウォームアップを既に正しく定義している場合、インスタンスのウォームアップを変更する必要はありません。ただし、デフォルトを上書きする場合は、このオプションの値を設定できます。デフォルトのインスタンスのウォームアップにおける設定の詳細については、「Auto Scaling グループに対するインスタンスのデフォルトウォームアップを設定する」を参照してください。

5. 更新設定 では、次の操作を行います。

a. (オプション) [Checkpoints] (チェックポイント) で [Enable checkpoints] (チェックポイントを有効にする) を選択し、インスタンスの更新に増分または段階的なアプローチを使用するインスタンスを置換します。これにより、置換セット間の検証にさらに時間がかかります。チェックポイントの有効化を選択しない場合、インスタンスはほぼ連続した 1 回のオペレーションで置換されます。

チェックポイントを有効にする場合は、追加ステップ チェックポイントを有効にする (コンソール) を参照してください。

b. スキップマッチングを有効または無効にする:

- 起動テンプレートに既に一致しているインスタンスの置き換えをスキップするには、[スキップマッチングを有効にする] のチェックボックスをオンのままにします。
- このチェックボックスをオフにしてスキップマッチングを無効にすると、すべてのインスタンスを置換することができます。

スキップマッチングを有効にすると、既存の起動テンプレートを使用する代わりに、新しい起動テンプレートまたは起動テンプレートの新しいバージョンを設定できます。これは、[インスタンスの更新を開始] ページの [必要な設定] セクションで行います。

> **Note**
>
> スキップマッチング機能を使用して、現在起動設定を使用している Auto Scaling グループを更新するには、希望する設定で起動テンプレートを選択する必要があります。起動設定でのスキップマッチングはサポートされていません。

c. [スタンバイインスタンス] の場合、[無視]、[終了]、[待機] のいずれかを選択します。これにより、インスタンスが Standby 状態で見つかった場合の処理が決まります。詳細については、「Auto Scaling グループからインスタンスを一時的に削除する」を参照してください。

[待機] を選択する場合、これらのインスタンスをサービスに戻すために追加のステップを実行する必要があります。そうしない場合、インスタンスの更新がすべての InService インスタンスを置き換えて 1 時間待機します。次に、Standby インスタンスが残っていると、インスタンスの更新は失敗します。この状況を防ぐには、代わりにこれらのインスタンスに対して [無視] または [終了] を選択してください。

d. [スケールインで保護されたインスタンス] の場合、[無視]、[置換]、[待機] のいずれかを選択します。これにより、スケールインで保護されたインスタンスが見つかった場合の処理が決まります。詳細については、「インスタンスのスケールイン保護を使用する」を参照してください。

[待機] を選択する場合、これらのインスタンスからスケールイン保護を解除するために追加のステップを実行する必要があります。そうしない場合、インスタンスの更新は保護されていないすべてのインスタンスを置き換えて 1 時間待機します。次に、スケールインで保護

されたインスタンスが残っている場合、インスタンスの更新が失敗します。この状況を防ぐには、代わりにこれらのインスタンスに対して [無視] または [置き換え] を選択してください。

6. (オプション) CloudWatch アラーム では、 CloudWatch アラームを有効にする を選択し、1 つ以上のアラームを選択します。 CloudWatch アラームを使用して問題を特定し、アラームがALARM状態になった場合にオペレーションを失敗させることができます。詳細については、「自動ロールバックでインスタンスの更新を開始」を参照してください。
7. (オプション) [必要な設定] セクションを拡張し、Auto Scaling グループに実行する更新を指定します。

   このステップでは、コンソールインターフェイスで選択する代わりに、JSON または YAML 構文を使用してパラメータ値を編集するように選択できます。このためには、[Use console interface] (コンソールインターフェイスを使用する) の代わりに[Use code editor] (コードエディタを使用する) を選択します。以下の手順では、コンソールインターフェイスを使用して選択する方法について説明します。

   a. 起動テンプレートを更新する場合:

      - Auto Scaling グループの新しい起動テンプレートまたは新しい起動テンプレートバージョンを作成していない場合、このチェックボックスをオンにしないでください。
      - 新しい起動テンプレートまたは新しい起動テンプレートバージョンを作成した場合は、このチェックボックスをオンにします。このオプションを選択すると、Amazon EC2 Auto Scaling が現在の起動テンプレートおよび現在の起動テンプレートバージョンを表示します。他の利用可能なバージョンもすべて確認できます。起動テンプレートを選択し、バージョンを選択します。

        バージョンを選択すると、バージョン情報が表示されます。これは、インスタンスの更新の一部としてインスタンスを置換するときに使用される起動テンプレートのバージョンです。インスタンスの更新に成功すると、グループのスケール時など、新しいインスタンスが起動するたびに起動テンプレートのこのバージョンが使用されます。

   b. インスタンスタイプと購入オプションのセットを選択して、起動テンプレートのインスタンスタイプを上書きする場合:

      - 起動テンプレートで指定したインスタンスタイプおよび購入オプションを使用する場合、このチェックボックスをオンにしないでください。
      - 起動テンプレートのインスタンスタイプをオーバーライドまたはスポットインスタンスを実行する場合、このチェックボックスをオンにします。各インスタンスタイプを手動で追

加するか、プライマリインスタンスタイプおよび一致する追加のインスタンスタイプを取得する推奨オプションを選択できます。スポットインスタンスを起動する予定がある場合は、いくつかの異なるインスタンスタイプを追加することをお勧めします。そうすることで、選択したアベイラビリティーゾーンに十分なインスタンスのキャパシティがない場合、Amazon EC2 Auto Scaling は別のインスタンスタイプを起動できます。詳細については、「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。

> **Warning**
>
> スポットインスタンスの中断を処理できないアプリケーションにはスポットインスタンスを使用しないでください。Amazon EC2 スポットサービスがキャパシティを回収する必要がある場合、中断が発生する可能性があります。

このチェックボックスをオンにした場合は、起動テンプレートがまだスポットインスタンスをリクエストしていないことを確認してください。複数のインスタンスタイプを使用し、スポットインスタンスとオンデマンドインスタンスを起動する Auto Scaling グループを作成するため、スポットインスタンスをリクエストする起動テンプレートを使用することはできません。

> **Note**
>
> 現在起動設定を使用している Auto Scaling グループでこれらのオプションを設定するには、起動テンプレートの更新で起動テンプレートを選択する必要があります。起動設定のインスタンスタイプの上書きはサポートされていません。

8. (オプション) [ロールバック設定] で [自動ロールバックを有効にする] を選択すると、インスタンスの更新が失敗した場合に自動的にロールバックされます。

   この設定は、Auto Scaling グループがロールバックを使用するための前提条件を満たしている場合にのみ有効にできます。

   詳細については、「ロールバックで変更の取り消し」を参照してください。

9. すべての選択内容を見直し、正しく設定されていることを確認します。

   現在の設定と提案された変更の違いが、想定外または望ましくない形でアプリケーションに影響を及ぼさないよう、この時点で確認することをお勧めします。インスタンスタイプが起動テンプ

レートと互換性があることを確認するには、「インスタンスタイプの互換性」を参照してください。

10. インスタンス更新の選択内容が正しい場合は、[インスタンスの更新の開始] を選択します。

コンソールでインスタンスの更新を開始する (混合インスタンスグループ)

混合インスタンスポリシーで Auto Scaling グループを作成した場合は、次の手順に従います。グループに混合インスタンスポリシーをまだ定義していない場合は、コンソールでインスタンスの更新を開始する (基本的な手順) を参照してインスタンスの更新を開始します。

インスタンスの更新をスタートするには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   Auto Scaling グループページの下部に分割ペインが開きます。
3. [Instance refresh] (インスタンスの更新) タブの [Active instance refresh] (アクティブインスタンスの更新) で、[Start instance refresh] (インスタンスの更新を開始する) を選択します。
4. 可用性設定 では、次の操作を行います。

   a. インスタンス置換方法 の場合：

      - Auto Scaling グループにインスタンスメンテナンスポリシーを設定していない場合、インスタンス置換方法のデフォルト設定は Terminate と launch です。これは、インスタンスの更新の従来のデフォルト動作です。
      - Auto Scaling グループにインスタンスメンテナンスポリシーを設定すると、インスタンス置換メソッド のデフォルト値が提供されます。インスタンスのメンテナンスポリシーを上書きするには、オーバーライド を選択します。上書きは、現在のインスタンスの更新にのみ適用されます。次回インスタンスの更新を開始すると、これらの値はインスタンスメンテナンスポリシーのデフォルトにリセットされます。

      次の手順では、インスタンス置換方法を更新する方法について説明します。

      i. 次のいずれかのインスタンス置換方法を選択します。

- 終了前に起動： 既存のインスタンスを終了する前に、新しいインスタンスを最初にプロビジョニングする必要があります。これは、コスト削減よりも可用性を優先するアプリケーションに適しています。
- の終了と起動: 新しいインスタンスは、既存のインスタンスが終了すると同時にプロビジョニングされます。これは、可用性よりもコスト削減を優先するアプリケーションに適しています。また、現在利用可能な容量よりも多くの容量を起動すべきではないアプリケーションにも適しています。
- カスタム動作： このオプションを使用すると、インスタンスを置き換えるときに使用可能な容量のカスタム最小範囲と最大範囲を設定できます。これにより、コストと可用性の適切なバランスを実現できます。

ii. 正常率の設定 では、次のフィールドの一方または両方に値を入力します。有効化フィールドは、インスタンス置換方法 で選択したオプションによって異なります。

- 最小 : インスタンスの更新を続行するのに必要な最小正常率を設定します。
- 最大 : インスタンスの更新中に可能な最大正常率を設定します。

iii. 現在のグループサイズに基づいて置換中の推定一時容量を表示セクションを展開し、最小と最大 の値がグループにどのように適用されるかを確認します。使用される正確な値は、希望する容量値によって異なります。これは、グループがスケールすると変化します。

iv. 無効な置換サイズに対するフォールバック動作の設定セクションを展開し、可用性を優先するために最大正常率を超過するか、最小正常率を超過するかを選択します。

ごく少数のグループでは、デフォルトの Violate min healthy percentage オプションを保持することはお勧めしません。Auto Scaling グループにインスタンスが 1 つしかない場合、インスタンスの更新を開始すると停止する可能性があります。

このステップでは、インスタンスメンテナンスポリシーがまだない Auto Scaling グループを使用している場合のフォールバック動作を設定します。このオプションは使用できず、グループにインスタンスメンテナンスポリシーがある場合に表示されません。このオプションは、Terminate および launch replacement メソッドでのみ使用できます。他の置換方法は、可用性を優先するために最大正常率に違反します。

b. [インスタンスのウォームアップ] には、新しいインスタンスの状態が `InService` に変更されてから初期化が完了するまでの秒数を入力します。Amazon EC2 Auto Scaling は、次のインスタンスを置き換える前にこの時間を待機します。

ウォームアップ中、新しく起動されたインスタンスは Auto Scaling グループの (CPUUtilization、NetworkIn、NetworkOut など) 集計インスタンスのメトリクスにも計上されません。Auto Scaling グループにスケーリング ポリシーを追加した場合、スケーリングアクティビティは並行して実行されます。インスタンスの更新ウォームアップ期間に長い間隔を設定すると、新しく起動されたインスタンスがメトリクスに表示されるまでに時間がかかります。したがって、適切なウォームアップ期間により、Amazon EC2 Auto Scaling が古いメトリクスデータでスケーリングされなくなります。

Auto Scaling グループにデフォルトのインスタンスのウォームアップを既に正しく定義している場合、インスタンスのウォームアップを変更する必要はありません。ただし、デフォルトを上書きする場合は、このオプションの値を設定できます。デフォルトのインスタンスのウォームアップにおける設定の詳細については、「Auto Scaling グループに対するインスタンスのデフォルトウォームアップを設定する」を参照してください。

5. 更新設定 では、次の操作を行います。

   a. (オプション) [Checkpoints] (チェックポイント) で [Enable checkpoints] (チェックポイントを有効にする) を選択し、インスタンスの更新に増分または段階的なアプローチを使用するインスタンスを置換します。これにより、置換セット間の検証にさらに時間がかかります。チェックポイントの有効化を選択しない場合、インスタンスはほぼ連続した 1 回のオペレーションで置換されます。

   チェックポイントを有効にする場合は、追加ステップ チェックポイントを有効にする (コンソール) を参照してください。

   b. スキップマッチングを有効または無効にする:

   - 起動テンプレートと既に一致しているインスタンスの置き換えおよびすべてのインスタンスタイプのオーバーライドをスキップするには、[スキップマッチングを有効にする] チェックボックスをオンのままにします。
   - このチェックボックスをオフにしてスキップマッチングを無効にすると、すべてのインスタンスを置換することができます。

   スキップマッチングを有効にすると、既存の起動テンプレートを使用する代わりに、新しい起動テンプレートまたは起動テンプレートの新しいバージョンを設定できます。これは、[インスタンスの更新を開始] ページの [必要な設定] セクションで行います。[Desired configuration] (希望する設定) でインスタンスタイプの上書きを更新することもできます。

c. [スタンバイインスタンス] の場合、[無視]、[終了]、[待機] のいずれかを選択します。これにより、インスタンスが Standby 状態で見つかった場合の処理が決まります。詳細については、「Auto Scaling グループからインスタンスを一時的に削除する」を参照してください。

[待機] を選択する場合、これらのインスタンスをサービスに戻すために追加のステップを実行する必要があります。そうしない場合、インスタンスの更新はすべての InService インスタンスを置き換えて 1 時間待機します。次に、Standby インスタンスが残っていると、インスタンスの更新は失敗します。この状況を防ぐには、代わりにこれらのインスタンスに対して [無視] または [終了] を選択してください。

d. [スケールインで保護されたインスタンス] の場合、[無視]、[置換]、[待機] のいずれかを選択します。これにより、スケールインで保護されたインスタンスが見つかった場合の処理が決まります。詳細については、「インスタンスのスケールイン保護を使用する」を参照してください。

[待機] を選択する場合、これらのインスタンスからスケールイン保護を解除するために追加のステップを実行する必要があります。そうしない場合、インスタンスの更新は保護されていないすべてのインスタンスを置き換えて 1 時間待機します。次に、スケールインで保護されたインスタンスが残っている場合、インスタンスの更新が失敗します。この状況を防ぐには、代わりにこれらのインスタンスに対して [無視] または [置き換え] を選択してください。

6. (オプション) CloudWatch アラーム では、 CloudWatch アラームを有効にする を選択し、1 つ以上のアラームを選択します。 CloudWatch アラームを使用して問題を特定し、アラームが ALARM状態になった場合にオペレーションを失敗させることができます。詳細については、「自動ロールバックでインスタンスの更新を開始」を参照してください。

7. [Desired configuration] (希望する設定) セクションで以下を実行します。

このステップでは、コンソールインターフェイスで選択する代わりに、JSON または YAML 構文を使用してパラメータ値を編集するように選択できます。このためには、[Use console interface] (コンソールインターフェイスを使用する) の代わりに[Use code editor] (コードエディタを使用する) を選択します。以下の手順では、コンソールインターフェイスを使用して選択する方法について説明します。

a. 起動テンプレートを更新する場合:

- Auto Scaling グループの新しい起動テンプレートまたは新しい起動テンプレートバージョンを作成していない場合、このチェックボックスをオンにしないでください。

- 新しい起動テンプレートまたは新しい起動テンプレートバージョンを作成した場合は、このチェックボックスをオンにします。このオプションを選択すると、Amazon EC2 Auto Scaling が現在の起動テンプレートおよび現在の起動テンプレートバージョンを表示します。他の利用可能なバージョンもすべて確認できます。起動テンプレートを選択し、バージョンを選択します。

  バージョンを選択すると、バージョン情報が表示されます。これは、インスタンスの更新の一部としてインスタンスを置換するときに使用される起動テンプレートのバージョンです。インスタンスの更新に成功すると、グループのスケール時など、新しいインスタンスが起動するたびに起動テンプレートのこのバージョンが使用されます。

b. これらの設定を使用して、起動テンプレートで定義されているインスタンスタイプと購入オプションを上書きする場合:

   デフォルトでは、このチェックボックスはオンになっています。Amazon EC2 Auto Scaling により、現在、Auto Scaling グループの混合インスタンスポリシーで設定されている値が各パラメータに入力されます。変更するパラメータの値のみを更新します。これらの設定に関するガイダンスについては、複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ を参照してください。

   > ⚠ Warning
   >
   > このチェックボックスはオフにしないことをお勧めします。混合インスタンスポリシーの使用を停止する場合にのみオフにします。インスタンスの更新に成功すると、Amazon EC2 Auto Scaling は希望する設定に一致するようグループを更新します。混合インスタンスポリシーが含まれなくなった場合、Amazon EC2 Auto Scaling は現在実行中のスポットインスタンスを徐々に終了し、オンデマンドインスタンスに置換します。または、起動テンプレートがスポットインスタンスをリクエストした場合、Amazon EC2 Auto Scaling は現在実行中のオンデマンドインスタンスを徐々に終了し、スポットインスタンスに置換します。

8. (オプション) [ロールバック設定] で [自動ロールバックを有効にする] を選択すると、何らかの理由でインスタンスの更新が失敗した場合に自動的にロールバックされます。

   この設定は、Auto Scaling グループがロールバックを使用するための前提条件を満たしている場合にのみ有効にできます。

   詳細については、「ロールバックで変更の取り消し」を参照してください。

9. すべての選択内容を見直し、正しく設定されていることを確認します。

現在の設定と提案された変更の違いが、想定外または望ましくない形でアプリケーションに影響を及ぼさないよう、この時点で確認することをお勧めします。インスタンスタイプが起動テンプレートと互換性があることを確認するには、「インスタンスタイプの互換性」を参照してください。

インスタンス更新の選択内容が正しい場合は、[インスタンスの更新の開始] を選択します。

インスタンスの更新 (AWS CLI) の開始

インスタンスの更新をスタートするには

start-instance-refreshコマンドを使用して、 AWS CLIからインスタンスの更新をスタートします。JSON 設定ファイルでは、変更する任意の設定を指定できます。設定ファイルを参照するとき、次の例に示すように、ファイルパスおよび名前を指定します。

```
aws autoscaling start-instance-refresh --cli-input-json file://config.json
```

config.json の内容:

```
{
    "AutoScalingGroupName": "my-asg",
    "Preferences": {
      "InstanceWarmup": 60,
      "MinHealthyPercentage": 50,
      "AutoRollback": true,
      "ScaleInProtectedInstances": Ignore,
      "StandbyInstances": Terminate
    }
}
```

設定が指定されない場合、デフォルト値が使用されます。詳細については、「インスタンスの更新のデフォルト値について説明する」を参照してください。

出力例:

```
{
    "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"
}
```

インスタンスの更新をモニタリングする

AWS Management Console または を使用して、進行中のインスタンスの更新をモニタリングしたり、過去 6 週間の過去のインスタンスの更新のステータスを検索したりできます AWS CLI。

インスタンスの更新のステータスをモニタリングして確認する

インスタンスの更新のステータスをモニタリングして確認するには、次のいずれかの方法を使用します。

Console

> **Tip**
>
> この手順では、名前付き列がすでに表示されているはずです。非表示の列を表示したり、表示される行数を変更したりするには、セクションの右上隅にある歯車アイコンを選択して、設定モーダルを開きます。必要に応じて設定を更新し、[確認] を選択します。

インスタンスの更新のステータスをモニタリングして確認するには (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部に分割されたペインが開きます。
3. [Instance refresh history] (インスタンスの更新履歴) の [Instance refresh] (インスタンスの更新) タブで、[Status] (ステータス) 列を確認し、リクエストのステータスを決定できます。オペレーションは初期化中に `Pending`ステータスになります。その後、ステータスはすぐに `InProgress` に変わります。すべてのインスタンスが更新されると、ステータスが `Successful` に変わります。
4. グループのスケーリングアクティビティを表示することで、進行中のアクティビティの成功または失敗をさらにモニタリングできます。[Activity (アクティビティ)] タブの [Activity history (アクティビティ履歴)] では、インスタンスの更新が開始されると、インスタンスの終了時、インスタンスの起動時に、それぞれ別のエントリが表示されます。スケーリングアクティビティが多数ある場合は、アクティビティ履歴の上部にある > アイコンを選択すると、そのアクティビティの詳細が表示されます。アクティビティが失敗する原因となる問題

のトラブルシューティングについては、「」を参照してくださいAmazon EC2 Auto Scaling をトラブルシューティングする。

5. (オプション) インスタンス管理タブのインスタンス で、必要に応じて特定のインスタンスの進行状況を確認できます。

AWS CLI

インスタンスの更新のステータスをモニタリングして確認するには (AWS CLI)

次の describe-instance-refreshes コマンドを使用します。

```
aws autoscaling describe-instance-refreshes --auto-scaling-group-name my-asg
```

以下は出力例です。

インスタンスの更新は、開始時刻順に並べられます。まだ進行中のインスタンスの更新が最初に記述されます。


```
{
  "InstanceRefreshes":[
    {
      "InstanceRefreshId":"08b91cf7-8fa6-48af-b6a6-d227f40f1b9b",
      "AutoScalingGroupName":"my-asg",
      "Status":"InProgress",
      "StatusReason":"Waiting for instances to warm up before continuing. For
 example: i-0645704820a8e83ff is warming up.",
      "StartTime":"2023-11-24T16:46:52+00:00",
      "PercentageComplete":50,
      "InstancesToUpdate":0,
      "Preferences":{
        "MaxHealthyPercentage":120,
        "MinHealthyPercentage":90,
        "InstanceWarmup":60,
        "SkipMatching":false,
        "AutoRollback":true,
        "ScaleInProtectedInstances":"Ignore",
        "StandbyInstances":"Ignore"
      }
    },
    {
      "InstanceRefreshId":"0e151305-1e57-4a32-a256-1fd14157c5ec",
      "AutoScalingGroupName":"my-asg",
```

```
      "Status":"Successful",
      "StartTime":"2023-11-22T13:53:37+00:00",
      "EndTime":"2023-11-22T13:59:45+00:00",
      "PercentageComplete":100,
      "InstancesToUpdate":0,
      "Preferences":{
        "MaxHealthyPercentage":120,
        "MinHealthyPercentage":90,
        "InstanceWarmup":60,
        "SkipMatching":false,
        "AutoRollback":true,
        "ScaleInProtectedInstances":"Ignore",
        "StandbyInstances":"Ignore"
      }
    }
  ]
}
```

グループのスケーリングアクティビティを表示することで、進行中のアクティビティの成功または失敗をさらにモニタリングできます。スケーリングアクティビティは、インスタンスの更新に関する問題のトラブルシューティングに役立つ詳細についてドリルダウンするのにも役立ちます。詳細については、「Amazon EC2 Auto Scaling をトラブルシューティングする」を参照してください。

インスタンスの更新ステータス

インスタンスの更新を開始するとき、[ペンディング] ステータスになります。成功 、失敗 、キャンセルされた 、、RollbackSuccessfulまたは に達するInProgressまで、保留中 から に渡されますRollbackFailed。

インスタンスの更新には、次のステータスがあります。

ステータス	説明
[保留中]	リクエストは作成されましたが、インスタンスの更新が開始されていません。
InProgress	インスタンスの更新が進行中です。
[成功]	インスタンスの更新が正常に完了しました。

ステータス	説明
[失敗]	インスタンスの更新を完了できませんでした。ステータスの理由とスケーリングアクティビティを使用してトラブルシューティングを行うことができます。
[キャンセル中]	進行中のインスタンスの更新をキャンセルしています。
[キャンセル済]	インスタンスの更新はキャンセルされました。
RollbackIn進行状況	インスタンスの更新がロールバックされています。
RollbackFailed	ロールバックを完了できませんでした。ステータスの理由とスケーリングアクティビティを使用してトラブルシューティングを行うことができます。
RollbackSuccessful	ロールバックが正常に完了しました。

インスタンスの更新のキャンセル

まだ進行中のインスタンスの更新はキャンセルできますが、完了した後にキャンセルすることはできません。

インスタンスの更新をキャンセルしても、既に置き換えられたインスタンスはロールバックされません。インスタンスの変更をロールバックするには、代わりにロールバックを実行してください。詳細については、「ロールバックで変更の取り消し」を参照してください。

トピック

- インスタンスの更新のキャンセル (コンソール)
- インスタンスの更新のキャンセル (AWS CLI)

インスタンスの更新のキャンセル (コンソール)

インスタンスの更新をキャンセルするには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

3. [アクティブインスタンスの更新] の [インスタンスの更新] タブで、[アクション] および [キャンセル] を選択します。
4. 確認を求められたら、[確認] を選択します。

インスタンス更新のステータスは [キャンセル中] に設定されます。キャンセルが完了した後、インスタンスの更新のステータスは [キャンセル済み] に設定されます。

インスタンスの更新のキャンセル (AWS CLI)

インスタンスの更新をキャンセルするには

から cancel-instance-refresh コマンドを使用し AWS CLI 、Auto Scaling グループ名を指定します。

```
aws autoscaling cancel-instance-refresh --auto-scaling-group-name my-asg
```

出力例:

```
{
    "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"
}
```

ロールバックで変更の取り消し

まだ進行中のインスタンスの更新をロールバックできます。終了後はロールバックできません。ただし、新しいインスタンスの更新を開始することによって Auto Scaling グループを再度更新できます。

ロールバックするとき、Amazon EC2 Auto Scaling はそれまでにデプロイされたインスタンスを置き換えます。新しいインスタンスは、インスタンスの更新を開始する前に Auto Scaling グループに最後に保存した設定と一致します。

Amazon EC2 Auto Scaling はロールバック方法として次のものを提供しています。

- 手動ロールバック: ロールバックを手動で開始して、デプロイされた内容をロールバックポイントまで戻します。
- 自動ロールバック: Amazon EC2 Auto Scaling は、何らかの理由でインスタンスの更新が失敗した場合、または指定した CloudWatch アラームが ALARM状態になった場合に、デプロイされた内容を自動的に元に戻します。

内容

- 考慮事項
- ロールバックを手動で開始する
- 自動ロールバックでインスタンスの更新を開始

考慮事項

以下の考慮事項は、ロールバックを使用する場合に適用されます。

- ロールバックオプションは、インスタンスの更新の開始の一部として必要な設定を指定した場合にのみ使用できます。
- 以前のバージョンの起動テンプレートにロールバックできるのは、バージョンが特定の番号の付いたバージョンである場合のみです。Auto Scaling グループが `$Latest` または `$Default` の起動テンプレートバージョンを使用するように設定されている場合、ロールバックオプションは使用できません。
- また、 AWS Systems Manager Parameter Store から AMI エイリアスを使用するように設定された起動テンプレートにロールバックすることはできません。
- Auto Scaling グループに最後に保存した設定は、安定した状態である必要があります。安定していない状態でも、ロールバックのワークフローは実行されますが、最終的には失敗します。問題を解決するまで、Auto Scaling グループは失敗状態になり、インスタンスを正常に起動できなくなる可能性があります。これはサービスまたはアプリケーションの可用性に影響する可能性があります。

ロールバックを手動で開始する

Console

インスタンスの更新のロールバックを手動で開始するには (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。
3. [インスタンスの更新] タブの [アクティブインスタンスの更新] で、[アクション] および [ロールバック開始] を選択します。
4. 確認を求められたら、[確認] を選択します。

AWS CLI

インスタンスの更新のロールバックを手動で開始するには (AWS CLI)

AWS CLI から「rollback-instance-refresh」コマンドを使用し、Auto Scaling グループ名を指定します。

```
aws autoscaling rollback-instance-refresh --auto-scaling-group-name my-asg
```

出力例:

```
{
    "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"
}
```

> **Tip**
>
> このコマンドがエラーをスローする場合は、 を AWS CLI ローカルで最新バージョンに更新していることを確認してください。

自動ロールバックでインスタンスの更新を開始

自動ロールバック機能を使用すると、エラーが発生したときや、指定された Amazon CloudWatch アラームが ALARM状態になったときなど、インスタンスの更新が失敗したときに自動的にロールバックできます。

自動ロールバックを有効にしていて、インスタンスの置き換え中にエラーが発生した場合、インスタンスの更新は失敗してロールバックされるまで 1 時間かけてすべての置換を完了しようとします。これらのエラーは通常、EC2 の起動失敗、ヘルスチェックの設定ミス、Standby 状態にあるインスタンスやスケールインから保護されているインスタンスの終了を無視または許可しないことなどが原因で発生します。

CloudWatch アラームの指定はオプションです。アラームを指定するには、まずアラームを作成する必要があります。メトリクスアラームと複合アラームを作成できます。アラームの作成については、「Amazon ユーザーガイド CloudWatch 」を参照してください。Elastic Load Balancing メトリクスを例にとると、Application Load Balancer を使用する場合は HTTPCode_ELB_5XX_Count メトリクスと HTTPCode_ELB_4XX_Count メトリクスを使用できます。

考慮事項

- CloudWatch アラームを指定しても自動ロールバックを有効にせず、アラームの状態が になるとALARM、インスタンスの更新はロールバックなしで失敗します。
- インスタンスの更新を開始するときに、最大 10 個のアラームを選択できます。
- CloudWatch アラームを選択する場合、アラームは互換性のある状態である必要があります。アラームの状態が INSUFFICIENT_DATA または ALARM の場合、インスタンスの更新を開始しようとすると エラーが発生します。
- Amazon EC2 Auto Scaling が使用するアラームを作成する場合、アラームに欠落データポイントの処理方法を含める必要があります。メトリクスのデータポイントが頻繁に欠落する仕様の場合は、これらの期間中、アラームの状態が INSUFFICIENT_DATA になります。この状態になると、Amazon EC2 Auto Scaling は、新しいデータポイントが見つかるまでインスタンスを置き換えることができません。アラームに以前の ALARM または OK 状態を強制的に維持するには、代わりに欠落データを無視することを選択できます。詳細については、「Amazon ユーザーガイド」の「アラームが欠落データを処理する方法の設定」を参照してください。 CloudWatch

Console

自動ロールバックを使用してインスタンスの更新を開始するには (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。
3. [Instance refresh] (インスタンスの更新) タブの [Active instance refresh] (アクティブインスタンスの更新) で、[Start instance refresh] (インスタンスの更新を開始する) を選択します。
4. インスタンスの更新の開始 (コンソール) の手順に従い、必要に応じてインスタンスの更新設定を行います。
5. (オプション) 更新設定 で、CloudWatch アラーム に対してアラーム を有効にする を選択し、1 CloudWatch つ以上のアラームを選択して問題を特定し、アラームが ALARM状態になった場合に操作を失敗させます。
6. [ロールバックの設定] で、[自動ロールバックを有効にする] を選択して、失敗したインスタンスの更新を、インスタンスの更新を開始する前に最後に Auto Scaling グループに保存した設定に自動的にロールバックします。
7. 選択内容を確認して、[インスタンスの更新を開始する]を選択します。

AWS CLI

自動ロールバック (AWS CLI) でインスタンスの更新を開始するには

start-instance-refresh コマンドを使用し、Preferences の AutoRollback オプションに true を指定します。

次の例は、何かが失敗した場合に自動的にロールバックするインスタンスの更新を開始する方法を示しています。*italicized* パラメータ値をユーザー自身の値に置き換えてください。

```
aws autoscaling start-instance-refresh --cli-input-json file://config.json
```

config.json の内容。

```
{
    "AutoScalingGroupName": "my-asg",
    "DesiredConfiguration": {
      "LaunchTemplate": {
          "LaunchTemplateName": "my-launch-template",
          "Version": "1"
       }
    },
    "Preferences": {
      "AutoRollback": true
    }
}
```

または、インスタンスの更新が失敗したとき、または指定された CloudWatch アラームが ALARM 状態にあるときに自動的にロールバックするには、次の例のように、 で AlarmSpecificationオプションを指定Preferencesし、アラーム名を指定します。*italicized* パラメータ値をユーザー自身の値に置き換えてください。

```
{
    "AutoScalingGroupName": "my-asg",
    "DesiredConfiguration": {
      "LaunchTemplate": {
          "LaunchTemplateName": "my-launch-template",
          "Version": "1"
       }
    },
    "Preferences": {
```

```
      "AutoRollback": true,
      "AlarmSpecification": { "Alarms": [ "my-alarm" ] }
    }
}
```

成功すると、コマンドは以下のような出力を返します。

```
{
  "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"
}
```

> **Tip**
>
> このコマンドがエラーをスローする場合は、 を AWS CLI ローカルで最新バージョンに更新していることを確認してください。

スキップマッチングでのインスタンスの更新の使用

スキップマッチングは、既に最新の更新が適用されているインスタンスを無視するように Amazon EC2 Auto Scaling に指示します。これにより、必要以上のインスタンスを置き換えることはありません。これは Auto Scaling グループが特定のバージョンの起動テンプレートを使用していることを確認し、異なるバージョンを使用するインスタンスのみを置き換えたいときに役立ちます。

スキップマッチングを使用する際は、以下を考慮してください。

- スキップマッチングと必要な設定の両方でインスタンスの更新を開始する場合、Amazon EC2 Auto Scaling は必要な設定と一致するインスタンスがあるかどうかをチェックします。次に、必要な設定に一致しないインスタンスのみを置き換えます。インスタンスの更新に成功した後、Amazon EC2 Auto Scaling はグループを更新して必要な設定を反映させます。
- スキップマッチングでインスタンスの更新を開始しても、必要な設定を指定しない場合、Amazon EC2 Auto Scaling は、Auto Scaling グループに最後に保存した設定と一致するインスタンスがあるかどうかをチェックします。次に、最後に設定した構成と一致しないインスタンスのみが置き換えられます。
- スキップマッチングは、新しい起動テンプレート、起動テンプレートの新しいバージョン、一連のインスタンスタイプにで使用できます。スキップマッチングを有効にしますが、これらのうちどれも変更されていない場合は、インスタンスの更新はインスタンスを置き換えることなく直ちに成功します。必要な設定にその他の変更 (スポット割り当て戦略の変更など) を行った場合、Amazon

EC2 Auto Scaling はインスタンスの更新が成功するまで待機します。次に、新しい必要な設定を反映するように Auto Scaling グループ設定を更新します。

- 新しい起動設定では、スキップマッチングを使用することはできません。
- インスタンスの更新を開始し、必要な設定を指定すると、Amazon EC2 Auto Scaling はすべてのインスタンスが目的の設定を使用するようにします。したがって、インスタンスの更新の進行中に起動テンプレートに必要なバージョン`$Latest`として `$Default`または を指定し、起動テンプレートの新しいバージョンを作成すると、既に置き換えられたインスタンスは再び置き換えられます。
- スキップマッチングでは、起動テンプレートのユーザーデータスクリプトが更新されたコードをプルして新しいインスタンスにインストールするかどうかはわかりません。その結果、スキップマッチングでは、古いコードがインストールされているインスタンスの置き換えがスキップされる可能性があります。この場合、起動テンプレートのバージョンを更新しなくても、スキップマッチングをオフにして、すべてのインスタンスが最新のコードを受信できるようにする必要があります。

このセクションでは、スキップマッチングを有効にしてインスタンスの更新を開始する AWS CLI 手順について説明します。コンソールを使用する手順については、「インスタンスの更新の開始 (コンソール)」を参照してください。

スキップマッチング (基本手順)

このセクションのステップに従って、 を使用して以下 AWS CLI を実行します。

- インスタンスに適用する起動テンプレートを作成します。
- インスタンスの更新を開始して起動テンプレートを Auto Scaling グループに適用します。スキップマッチングを有効にしない場合、すべてのインスタンスが置き換えられます。これは、インスタンスのプロビジョニングに使用した起動テンプレートが、必要な設定に指定した起動テンプレートと同じであっても当てはまります。

新しい起動テンプレートでスキップマッチングを使用するには

1. 「create-launch-template」コマンドを使用し、Auto Scaling グループに新しい起動テンプレートを作成します。Auto Scaling グループに作成されたインスタンスの詳細を定義する `--launch-template-data` オプションおよび JSON 入力を含めます。

   例えば、次のコマンドを使用して AMI ID *`ami-0123456789abcdef0`* および *`t2.micro`* インスタンスタイプを含む基本的な起動テンプレートを作成します。

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling
 --version-description version1 \
  --launch-template-data
 '{"ImageId":"ami-0123456789abcdef0","InstanceType":"t2.micro"}'
```

成功すると、コマンドは以下のような出力を返します。

```
{
   "LaunchTemplate": {
     "LaunchTemplateId": "lt-068f72b729example",
     "LaunchTemplateName": "my-template-for-auto-scaling",
     "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
     "CreateTime": "2023-01-30T18:16:06.000Z",
     "DefaultVersionNumber": 1,
     "LatestVersionNumber": 1
  }
}
```

詳細については、「を使用した起動テンプレートの作成と管理の例 AWS CLI」を参照してください。

2. 「start-instance-refresh」コマンドを使用し、インスタンス置き換えワークフローを開始して ID *lt-068f72b729example* を含む新しい起動テンプレートを適用します。起動テンプレートは新しいため、バージョンは 1 つしかありません。つまり、起動テンプレートの 1 バージョンはこのインスタンスの更新の対象となります。インスタンスの更新中にスケールアウトイベントが発生し、かつ Amazon EC2 Auto Scaling がこの起動テンプレートのバージョン 1 を使用して新しいインスタンスをプロビジョニングする場合は置き換えられません。操作が正常に完了すると、新しい起動テンプレートが Auto Scaling グループに正常に適用されます。

```
aws autoscaling start-instance-refresh --cli-input-json file://config.json
```

config.json の内容。

```
{
    "AutoScalingGroupName": "my-asg",
    "DesiredConfiguration": {
      "LaunchTemplate": {
          "LaunchTemplateId": "lt-068f72b729example",
          "Version": "$Default"
```

```
        }
    },
    "Preferences": {
      "SkipMatching": true
    }
}
```

成功すると、コマンドは以下のような出力を返します。

```
{
  "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"
}
```

スキップマッチング (混合インスタンスグループ)

混合インスタンスポリシー を持つ Auto Scaling グループがある場合は、このセクションの手順に従って、 AWS CLI を使用してスキップマッチングでインスタンスの更新を開始します。次のオプションがあります。

- ポリシーで指定されたすべてのインスタンスタイプに適用する新しい起動テンプレートを指定します。
- ポリシーの起動テンプレートを変更するかどうかを問わず、更新された一連のインスタンスタイプを指定します。例えば、不要なインスタンスタイプから移行できます。起動テンプレートは、置き換えるインスタンスの AMI、セキュリティグループ、その他の詳細を変更せず、そのまま使用します。

ニーズに合ったオプションに応じて、次のいずれかのセクションにある手順に従ってください。

新しい起動テンプレートでスキップマッチングを使用するには

1. 「create-launch-template」コマンドを使用し、Auto Scaling グループに新しい起動テンプレートを作成します。Auto Scaling グループに作成されたインスタンスの詳細を定義する `--launch-template-data` オプションおよび JSON 入力を含めます。

   例えば、次のコマンドを使用して AMI ID *ami-0123456789abcdef0* を含む起動テンプレートを作成します。

   ```
   aws ec2 create-launch-template --launch-template-name my-new-template --version-
   description version1 \
   ```

```
  --launch-template-data '{"ImageId":"ami-0123456789abcdef0"}'
```

成功すると、コマンドは以下のような出力を返します。

```
{
   "LaunchTemplate": {
     "LaunchTemplateId": "lt-04d5cc9b88example",
     "LaunchTemplateName": "my-new-template",
     "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
     "CreateTime": "2023-01-31T15:56:02.000Z",
     "DefaultVersionNumber": 1,
     "LatestVersionNumber": 1
  }
}
```

詳細については、「を使用した起動テンプレートの作成と管理の例 AWS CLI」を参照してください。

2. Auto Scaling グループの既存の混合インスタンスポリシーを表示するには、「describe-auto-scaling-group」コマンドを実行します。この情報は、インスタンスの更新を開始する次のステップで必要になります。

   次のコマンドの例では、*my-asg* という名前の Auto Scaling グループに設定された混合インスタンスポリシーを返します。

   ```
   aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
   ```

   成功すると、コマンドは以下のような出力を返します。

   ```
   {
     "AutoScalingGroups":[
       {
         "AutoScalingGroupName":"my-asg",
         "AutoScalingGroupARN":"arn",
         "MixedInstancesPolicy":{
           "LaunchTemplate":{
             "LaunchTemplateSpecification":{
               "LaunchTemplateId":"lt-073693ed27example",
               "LaunchTemplateName":"my-old-template",
               "Version":"$Default"
             },
   ```

```
          "Overrides":[
            {
              "InstanceType":"c5.large"
            },
            {
              "InstanceType":"c5a.large"
            },
            {
              "InstanceType":"m5.large"
            },
            {
              "InstanceType":"m5a.large"
            }
          ]
        },
        "InstancesDistribution":{
          "OnDemandAllocationStrategy":"prioritized",
          "OnDemandBaseCapacity":1,
          "OnDemandPercentageAboveBaseCapacity":50,
          "SpotAllocationStrategy":"price-capacity-optimized"
        }
      },
      "MinSize":1,
      "MaxSize":5,
      "DesiredCapacity":4,
      ...
    }
  ]
}
```

3. 「start-instance-refresh」コマンドを使用し、インスタンス置き換えワークフローを開始して ID *lt-04d5cc9b88example* を含む新しい起動テンプレートを適用します。起動テンプレートは新しいため、バージョンは 1 つしかありません。つまり、起動テンプレートの 1 バージョンはこのインスタンスの更新の対象となります。インスタンスの更新中にスケールアウトイベントが発生し、かつ Amazon EC2 Auto Scaling がこの起動テンプレートのバージョン 1 を使用して新しいインスタンスをプロビジョニングする場合は置き換えられません。操作が正常に完了すると、更新された混合インスタンスポリシーが Auto Scaling グループに正常に適用されます。

```
aws autoscaling start-instance-refresh --cli-input-json file://config.json
```

config.json の内容。

```
{
  "AutoScalingGroupName":"my-asg",
  "DesiredConfiguration":{
    "MixedInstancesPolicy":{
      "LaunchTemplate":{
        "LaunchTemplateSpecification":{
          "LaunchTemplateId":"lt-04d5cc9b88example",
          "Version":"$Default"
        },
        "Overrides":[
          {
            "InstanceType":"c5.large"
          },
          {
            "InstanceType":"c5a.large"
          },
          {
            "InstanceType":"m5.large"
          },
          {
            "InstanceType":"m5a.large"
          }
        ]
      },
      "InstancesDistribution":{
        "OnDemandAllocationStrategy":"prioritized",
        "OnDemandBaseCapacity":1,
        "OnDemandPercentageAboveBaseCapacity":50,
        "SpotAllocationStrategy":"price-capacity-optimized"
        }
      }
    },
  "Preferences":{
    "SkipMatching":true
  }
}
```

成功すると、コマンドは以下のような出力を返します。

```
{
  "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"
```

```
}
```

次の手順では、起動テンプレートを変更せずに更新された一連のインスタンスタイプを指定します。

更新された一連のインスタンスタイプでスキップマッチングを使用するには

1. Auto Scaling グループの既存の混合インスタンスポリシーを表示するには、「describe-auto-scaling-group」コマンドを実行します。この情報は、インスタンスの更新を開始する次のステップで必要になります。

   次のコマンドの例では、*my-asg* という名前の Auto Scaling グループに設定された混合インスタンスポリシーを返します。

   ```
   aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
   ```

   成功すると、コマンドは以下のような出力を返します。

   ```
   {
     "AutoScalingGroups":[
       {
         "AutoScalingGroupName":"my-asg",
         "AutoScalingGroupARN":"arn",
         "MixedInstancesPolicy":{
           "LaunchTemplate":{
             "LaunchTemplateSpecification":{
               "LaunchTemplateId":"lt-073693ed27example",
               "LaunchTemplateName":"my-template-for-auto-scaling",
               "Version":"$Default"
             },
             "Overrides":[
               {
                 "InstanceType":"c5.large"
               },
               {
                 "InstanceType":"c5a.large"
               },
               {
                 "InstanceType":"m5.large"
               },
               {
                 "InstanceType":"m5a.large"
   ```

```
            }
          ]
        },
        "InstancesDistribution":{
          "OnDemandAllocationStrategy":"prioritized",
          "OnDemandBaseCapacity":1,
          "OnDemandPercentageAboveBaseCapacity":50,
          "SpotAllocationStrategy":"price-capacity-optimized"
        }
      },
      "MinSize":1,
      "MaxSize":5,
      "DesiredCapacity":4,
      ...
    }
  ]
}
```

2. start-instance-refresh コマンドを使用し、インスタンス置き換えワークフローを開始して更新を適用します。特定のインスタンスタイプを使用するインスタンスを置き換える場合、必要な設定で希望するインスタンスタイプのみを含む混合インスタンスポリシーを指定する必要があります。代わりに新しいインスタンスタイプを追加するかどうかを選択できます。

   次のコマンドの例では、不要なインスタンスタイプ *m5a.large* なしでインスタンスの更新を開始します。グループ内のインスタンスタイプが残り 3 つのインスタンスタイプのいずれかと一致しないとき、インスタンスは置き換えられます。(インスタンスの更新は、新しいインスタンスをプロビジョンするインスタンスタイプを選択しませんのでご注意ください。代わりに割り当て戦略によって選択されます) 操作が正常に完了すると、更新された混合インスタンスポリシーが Auto Scaling グループに正常に適用されます。

   ```
   aws autoscaling start-instance-refresh --cli-input-json file://config.json
   ```

   config.json の内容

   ```
   {
     "AutoScalingGroupName":"my-asg",
     "DesiredConfiguration":{
       "MixedInstancesPolicy":{
         "LaunchTemplate":{
           "LaunchTemplateSpecification":{
             "LaunchTemplateId":"lt-073693ed27example",
   ```

```
          "Version":"$Default"
        },
        "Overrides":[
          {
            "InstanceType":"c5.large"
          },
          {
            "InstanceType":"c5a.large"
          },
          {
            "InstanceType":"m5.large"
          }
        ]
      },
      "InstancesDistribution":{
        "OnDemandAllocationStrategy":"prioritized",
        "OnDemandBaseCapacity":1,
        "OnDemandPercentageAboveBaseCapacity":50,
        "SpotAllocationStrategy":"price-capacity-optimized"
        }
      }
    }
  },
  "Preferences":{
    "SkipMatching":true
  }
}
```

インスタンスの更新にチェックポイントを追加する

インスタンスの更新を使用するとき、段階的にインスタンスを置き換え、進行しながらインスタンスに検証を実行できます。段階的な置き換えを行うには、チェックポイントを追加します。チェックポイントは、インスタンスの更新が一時停止する時点です。チェックポイントを使用すると、Auto Scaling グループの更新の選択方法をより詳細に管理できます。これにより、アプリケーションが確実かつ予測可能な方法で機能することを確認できます。

内容

- 仕組み
- 考慮事項
- チェックポイントを有効にする (コンソール)

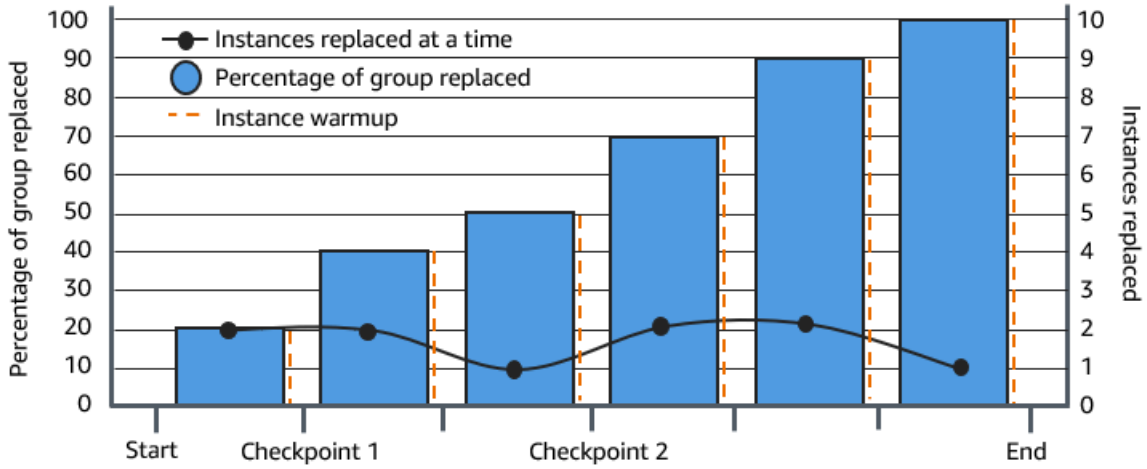
- チェックポイントを有効にする (AWS CLI)

仕組み

インスタンスの更新を開始するときは、Auto Scaling グループ内のインスタンスの合計数に対するチェックポイントの割合を指定します。これらのチェックポイントは、チェックポイントに到達する前に新しいインスタンスである必要がある Auto Scaling グループ内のインスタンスの最小パーセンテージを示します。例えば、チェックポイントが の場合`[20, 50, 100]`、最初のチェックポイントはインスタンスの 20% が新しいときに、2 番目のチェックポイントは 50% が新しいときに、最後のチェックポイントはすべてのインスタンスが新しいときに到達します。

Amazon EC2 Auto Scaling は、グループの最小正常率を維持しながら、指定されたチェックポイントの割合を尊重してインスタンスの置換をペース調整します。チェックポイントのパーセンテージを満たすために、Amazon EC2 Auto Scaling は、最小の正常なパーセンテージで許容されるよりも少ない数を置き換えることがありますが、それを超える数を置き換えることは決してありません。

10 個のインスタンスがある次の Auto Scaling グループについて考えてみましょう。チェックポイントの割合は で`[20,50,100]`、最小正常率は 80%、最大正常率は 100% です。最小の正常なパーセンテージを維持するために、一度に置き換えることができるインスタンスは 2 つだけです。次の図は、チェックポイントに達する前にインスタンスを置き換えるプロセスの概要を示しています。



上記の例では、開始する新しいインスタンスごとにインスタンスのウォームアップ期間があります。インスタンスを待機状態にして、起動または終了時にカスタムアクションを実行するライフサイクルフックを使用することもできます。

Amazon EC2 Auto Scaling は、100% 完全なチェックポイントを除き、チェックポイントごとにイベントを発行します。 EventBridge ルールを追加して、Amazon SNS などのターゲットにイベントを

送信できます。これにより、必要な検証を実行できるタイミングが通知されます。詳細については、「インスタンス更新イベントの EventBridge ルールを作成する」を参照してください。

考慮事項

チェックポイントを使用する際は、次の考慮事項に注意してください。

- チェックポイントは割合に基づいているため、置換されるインスタンスの数はグループサイズに応じて変化します。スケールアウト アクティビティが発生し、グループサイズが大きくなると、進行中のオペレーションがチェックポイントに再び到達する可能性があります。この場合、Amazon EC2 Auto Scaling は別の通知を送信し、チェックポイント間の待機時間を繰り返してから続行します。
- 特定の状況下では、チェックポイントをスキップすることができます。例えば、Auto Scaling グループに 2 個のインスタンスがあり、チェックポイントの割合が `[10,40,100]` だとします。最初のインスタンスが置換された後、Amazon EC2 Auto Scaling は、グループの 50% が置換されたと計算します。50% は最初の 2 つのチェックポイントよりも高いため、最初のチェックポイント (`10`) をスキップし、2 番目のチェックポイント (`40`) の通知を送信します。
- オペレーションをキャンセルすると、それ以降の置換は行われません。オペレーションをキャンセルするか、最後のチェックポイントに到達する前に失敗した場合、すでに置き換えられたインスタンスは前の設定にロールバックされません。
- 部分更新の場合、オペレーションを再実行するとき、Amazon EC2 Auto Scaling は最後のチェックポイントから再開せず、古いインスタンスのみが置換されても停止しません。ただし、新しいインスタンスをターゲットにする前に、まず古いインスタンスを置き換え対象とします。
- チェックポイントのパーセンテージがグループ内のインスタンス数に対して低すぎる場合、実際の完了率はそのチェックポイントのパーセンテージよりも高い可能性があります。例えば、チェックポイントのパーセンテージが 20 % で、グループに 4 つのインスタンスがあるとします。Amazon EC2 Auto Scaling が 4 つのインスタンスのうち 1 つを置き換える場合、実際に置き換えられるパーセンテージ (25 %) はチェックポイントのパーセンテージ (20 %) よりも高くなります。
- チェックポイントに達すると、インスタンスのウォームアップが完了するまで、表示される全体の完了率は更新されません。例えば、チェックポイントの割合は 15 分の`[20,50]`チェックポイント遅延で、最小正常率は 80% です。Auto Scaling グループには 10 個のインスタンスがあり、次の置き換えを行います。
  - 0:00: 2 個の古いインスタンスが新しいインスタンスに置き換えられます。
  - 0:10: 2 個の新しいインスタンスがウォームアップを完了します。
  - 0:25: 2 個の古いインスタンスが新しいインスタンスに置き換えられます。(最小正常率を維持するために、2 個のインスタンスのみが置換されます)。

- 0:35: 2 個の新しいインスタンスがウォームアップを完了します。
- 0:35: 1 個の古いインスタンスが新しいインスタンスに置き換えられます。
- 0:45: 1 個の新しいインスタンスがウォームアップを完了します。

0:35 で、オペレーションは新しいインスタンスの起動を停止します。新しいインスタンスがウォームアップされていないため、完了率は、完了した置換の数 (50%) を正確に反映していません。新しいインスタンスが 0:45 にウォームアップ期間を完了すると、完了率は 50% と表示されます。

チェックポイントを有効にする (コンソール)

インスタンスの更新を開始する前にチェックポイントを有効にして、増分または段階的なアプローチを使用し、インスタンスを置換することができます。これにより、検証にさらに時間がかかります。

チェックポイントを使用するインスタンスの更新をスタートするには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   Auto Scaling グループページの下部に分割ペインが開きます。
3. [Instance refresh] (インスタンスの更新) タブの [Active instance refresh] (アクティブインスタンスの更新) で、[Start instance refresh] (インスタンスの更新を開始する) を選択します。
4. [Start instance refresh] (インスタンスの更新をスタートする) ページで、[Minimum healthy percentage] (最小正常率) および [Instance warmup] (インスタンスのウォームアップ) に値を入力します。
5. [Enable checkpoints] (チェックポイントを有効にする) チェックボックスをオンにします。

   これにより、最初のチェックポイントのしきい値をパーセンテージで定義できるボックスが表示されます。
6. [Proceed until ____ % of the group is refreshed] (グループの ____% が更新されるまで続行する) に数値 (1 ~ 100) を入力します。これにより、最初のチェックポイントの割合が設定されます。
7. 別のチェックポイントを追加するには、[Add checkpoint (チェックポイントの追加)]を選択し、次のチェックポイントの割合を定義します。

8. チェックポイントに達した後、Amazon EC2 Auto Scaling が待機する時間を指定するには、[Wait for **1 hour** between checkpoints (チェックポイント間で待機)]のフィールドを更新します。時間単位は、時、分、秒のいずれかです。
9. インスタンスの更新の選択が終了したら、[インスタンスの更新を開始する] を選択します。

チェックポイントを有効にする (AWS CLI)

を使用してチェックポイントを有効にしてインスタンスの更新を開始するには AWS CLI、次のパラメータを定義する設定ファイルが必要です。

- `CheckpointPercentages`: 置き換えるインスタンスの割合のしきい値を指定します。これらのしきい値は、チェックポイントを提供します。置換およびウォームアップされたインスタンスの割合が指定されたしきい値の 1 つに達すると、オペレーションは指定された期間待機します。待機時間を`CheckpointDelay`秒単位で指定します。指定した期間が経過すると、インスタンスの更新は次のチェックポイント (該当する場合) に到達するまで続行されます。
- `CheckpointDelay`: チェックポイントに到達してから続行するまでに待機する時間 (秒) を規定します。検証の実行に十分な時間を選択します。

`CheckpointPercentages` 配列に表示される最後の値は、正常に置換する必要がある Auto Scaling グループの割合を示します。このパーセンテージが正常に置き換えられ、各インスタンスの初期化が完了したとみなされると、オペレーションは `Successful` になります。

複数のチェックポイントを作成するには

複数のチェックポイントを作成するには、次の例のスタートインスタンスの更新コマンドを使用します。この例では、最初に Auto Scaling グループの 1% を更新する、インスタンスの更新を設定します。10 分の待機後、次の 19% が更新され、さらに 10 分待機します。最後に、オペレーション終了前にグループの残りの部分が更新されます。

```
aws autoscaling start-instance-refresh --cli-input-json file://config.json
```

`config.json` の内容:

```
{
    "AutoScalingGroupName": "my-asg",
    "Preferences": {
      "InstanceWarmup": 60,
      "MinHealthyPercentage": 80,
```

```
      "CheckpointPercentages": [1,20,100],
      "CheckpointDelay": 600
    }
}
```

単一のチェックポイントを作成するには

単一のチェックポイントを作成するには、次の例のスタートインスタンスの更新コマンドを使用します。この例では、最初に Auto Scaling グループの 20% を更新する、インスタンスの更新を設定します。10 分の待機後、オペレーション終了前にグループの残りの部分が更新されます。

```
aws autoscaling start-instance-refresh --cli-input-json file://config.json
```

config.json の内容:

```
{
    "AutoScalingGroupName": "my-asg",
    "Preferences": {
      "InstanceWarmup": 60,
      "MinHealthyPercentage": 80,
      "CheckpointPercentages": [20,100],
      "CheckpointDelay": 600
    }
}
```

Auto Scaling グループを部分的に更新するには

Auto Scaling グループの一部のみを置換して完全に停止させるには、次の例の start-instance-refresh コマンドを使用します。この例では、最初に Auto Scaling グループの 1% を更新する、インスタンスの更新を設定します。10 分の待機後、オペレーション終了前に次の 19% が更新されます。

```
aws autoscaling start-instance-refresh --cli-input-json file://config.json
```

config.json の内容:

```
{
    "AutoScalingGroupName": "my-asg",
    "Preferences": {
      "InstanceWarmup": 60,
      "MinHealthyPercentage": 80,
      "CheckpointPercentages": [1,20],
```

```
      "CheckpointDelay": 600
    }
}
```

インスタンスの最大存続期間に基づいて Auto Scaling インスタンスを置き換える

インスタンスの最大有効期間は、インスタンスが終了し置き換えられるまでに稼働できる最大時間 (秒単位) を指定します。一般的なユースケースでは、内部のセキュリティポリシーや外部のコンプライアンスコントロールにより、スケジュールどおりにインスタンスを置換する要件がある場合があります。

86,400 秒 (1 日) 以上の値を指定する必要があります。以前に設定した値をクリアするには、新しい値 0 を指定します。この設定は、Auto Scaling グループの現在および今後のすべてのインスタンスに適用されます。

内容

- 考慮事項
- 最大インスタンスライフタイムを設定する
- 制限事項

考慮事項

この機能を使用する際の考慮事項は次のとおりです。

- 古いインスタンスが置き換えられて新しいインスタンスが起動するたび、新しいインスタンスは Auto Scaling グループに現在関連付けられている起動テンプレートまたは起動設定を使用します。起動テンプレートまたは起動設定でアプリケーションの別のバージョンの Amazon マシンイメージ (AMI) ID が指定されている場合、このバージョンのアプリケーションは自動的にデプロイされます。
- インスタンスの最大有効期間の設定が低すぎると、インスタンスが希望よりも速く置き換えられる可能性があります。Amazon EC2 Auto Scaling は通常、インスタンスを一度に 1 つずつ置き換え、置き換えの間に一時停止します。ただし、指定した最大インスタンス有効期間で各インスタンスを個別に置き換えるのに十分な時間がない場合、Amazon EC2 Auto Scaling は一度に複数のインスタンスを置き換える必要があります。Auto Scaling グループの現在のキャパシティの最大 10% まで、複数のインスタンスが置換される場合があります。一度に置き換えるインスタンスが

多すぎないようにするには、インスタンスの最大有効期間を長く設定するか、インスタンスのスケールイン保護を使用して、個々のインスタンスが一時的に終了しないようにします。詳細については、「インスタンスのスケールイン保護を使用する」を参照してください。

- デフォルトでは、Amazon EC2 Auto Scaling はインスタンスを終了するための新しいスケーリングアクティビティを作成してから終了します。インスタンスの終了中、別のスケーリングアクティビティが新しいインスタンスを起動します。この動作は、インスタンスメンテナンスポリシーを使用して終了する前に起動するように変更できます。詳細については、「インスタンスのメンテナンスポリシー」を参照してください。

最大インスタンスライフタイムを設定する

コンソールで Auto Scaling グループを作成する場合、インスタンスの最大ライフタイムを設定することはできません。ただし、グループ作成後に、グループを編集してインスタンスの最大ライフタイムを設定できます。

グループの最大インスタンスライフタイムを設定するには (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   [Auto Scaling groups] (Auto Scaling グループ) ページの下部に分割ペインが開き、選択したグループに関する情報が表示されます。
3. [詳細] タブで、[高度な設定]、[編集] の順に選択します。
4. [Maximum instance lifetime] (最大インスタンス有効期間) に、インスタンスが稼働できる最大秒数を入力します。
5. [更新] を選択します。

[Activity history] (アクティビティ履歴) の [Activity] (アクティビティ) タブでは、履歴全体にグループのインスタンスの置換を表示できます。

グループの最大インスタンスライフタイムを設定するには (AWS CLI)

を使用して AWS CLI 、新規または既存の Auto Scaling グループのインスタンスの最大有効期間を設定することもできます。

新しい Auto Scaling グループでは、create Auto Scaling グループコマンドを実行します。

```
aws autoscaling create-auto-scaling-group --cli-input-json file://~/config.json
```

次の例は、インスタンスの最大有効期間を 2592000 秒 (30 日) で示す config.json ファイルです。

```
{
    "AutoScalingGroupName": "my-asg",
    "LaunchTemplate": {
        "LaunchTemplateName": "my-launch-template",
        "Version": "$Default"
    },
    "MinSize": 1,
    "MaxSize": 5,
    "MaxInstanceLifetime": 2592000,
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782",
    "Tags": []
}
```

既存の Auto Scaling グループについては、update Auto Scaling グループコマンドを実行します。

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-existing-asg --
max-instance-lifetime 2592000
```

Auto Scaling グループの最大インスタンス有効期間を確認するには

describe Auto Scaling グループコマンドを実行します。

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
```

制限事項

- すべてのインスタンスで最大ライフタイムが正確であるとは限りません: インスタンスが置き換えられるのは、最大期間が終了したときだけとは限りません。状況によっては、最大インスタンス有効期間パラメータを更新した直後に、Amazon EC2 Auto Scaling によってインスタンスの置き換えをスタートする必要がある場合があります。この動作の理由は、すべてのインスタンスを同時に置き換えることを避けることです。
- インスタンスのスケールイン保護： Amazon EC2 Auto Scaling は、終了できるインスタンスを制御するのに役立つインスタンスのスケールイン保護を提供します。インスタンスでこの保護が有効

になっている場合、Amazon EC2 Auto Scaling は、インスタンスの最大有効期間に達した場合でもインスタンスを終了しません。

- 起動前にインスタンスが終了する: Auto Scaling グループにインスタンスが 1 つしかない場合、Amazon EC2 Auto Scaling はインスタンスを終了してからデフォルトで新しいインスタンスを起動するため、最大インスタンス有効期間機能は停止する可能性があります。この動作を終了前に起動するように変更するには、「」を参照してくださいインスタンスのメンテナンスポリシー。

Auto Scaling グループのサイズをスケールする

スケーリングは、アプリケーションのコンピューティングキャパシティーを増減する機能です。スケーリングは、Auto Scaling グループに Amazon EC2 インスタンスの起動または終了を指示するイベント (スケーリングアクション) で始まります。

Amazon EC2 Auto Scaling では、アプリケーションのニーズを最大限に満たすようにさまざまな方法でスケーリングを調整できます。そのため、アプリケーションを十分に理解していることが重要です。次の考慮事項に注意が必要です。

- どのような役割を Amazon EC2 Auto Scaling がアプリケーションのアーキテクチャで果たすか。自動スケーリングはキャパシティーの増減手段として考えるのが一般的ですが、一定数のサーバーを維持する場合にも便利です。
- どのようなコストの制約がお客様にとって重要か。Amazon EC2 Auto Scaling は EC2 インスタンスを使用するため、使用したリソースに対してのみ料金が発生します。コストの制約を知ることは、アプリケーションをスケーリングするタイミングと量を決定するときに役立ちます。
- アプリケーションにとって重要なメトリクスは何ですか？ Amazon CloudWatch は、Auto Scaling グループで使用できるさまざまなメトリクスをサポートしています。

内容

- スケーリング方法を選択する
- Auto Scaling グループのスケーリング制限を設定する
- Auto Scaling グループに対するインスタンスのデフォルトウォームアップを設定する
- Amazon EC2 Auto Scaling の手動スケーリング
- Amazon EC2 Auto Scaling のスケジュールされたスケーリング
- Amazon EC2 Auto Scaling の動的スケーリング
- Amazon EC2 Auto Scaling の予測スケーリング
- スケールイン中に終了する Auto Scaling インスタンスを制御する
- Amazon EC2 Auto Scaling プロセスの一時停止と再開

スケーリング方法を選択する

Amazon EC2 Auto Scaling では、Auto Scaling グループをスケーリングする方法を用意しています。

固定数のインスタンスを維持する

Auto Scaling グループのデフォルトは、スケーリングポリシーまたはスケジュールされたアクションがアタッチされていないことです。これにより、固定サイズが維持されます。Auto Scaling グループを作成したら、目的の容量を満たすのに十分なインスタンスを起動することから始まります。グループにスケーリング条件がアタッチされていない場合、インスタンスが異常になっても、希望する容量が維持されます。Amazon EC2 Auto Scaling は、Auto Scaling グループ内の各インスタンスの状態をモニタリングします。インスタンスが異常になったことがわかったら、新しいインスタンスに置き換えます。このプロセスの詳細については、「」を参照してくださいAuto Scaling グループ内のインスタンスのヘルスチェック。

手動でスケールする

手動スケーリングは、Auto Scaling グループをスケーリングする最も基本的な方法です。Auto Scaling グループの希望する容量を更新するか、Auto Scaling グループのインスタンスを終了できます。詳細については、「Amazon EC2 Auto Scaling の手動スケーリング」を参照してください。

スケジュールに基づくスケーリング

スケジュールによるスケーリングは、スケーリングアクションが日時の関数として自動的に実行されることを意味します。グループのインスタンスの数を増減しなければならない状況が予測可能なスケジュールで発生するため、いつその数を増減すべきかが正確にわかっている場合に、このスケーリング方法は便利です。詳細については、「Amazon EC2 Auto Scaling のスケジュールされたスケーリング」を参照してください。

需要に基づいて動的にスケーリングする

動的なスケーリングを使用して、リソースをスケーリングする高度な方法では、需要の変化に応じて Auto Scaling グループを動的にサイズ変更するスケーリングポリシーを定義できます。例えば、現在 2 つのインスタンスで実行されているウェブアプリケーションがあり、アプリケーションの負荷が変化しても Auto Scaling グループの CPU 使用率を約 50% に維持する必要があるとします。この方法は、トラフィックがいつ変化するかわからない場合に、トラフィックの変化が発生する際のスケーリングに役立ちます。応答するようにスケーリングポリシーを設定できます。トラフィックの変化に応じてスケーリングするために使用できる複数のポリシータイプ (またはそれらの組み合わせ) があります。詳細については、「Amazon EC2 Auto Scaling の動的スケーリング」を参照してください。

プロアクティブにスケーリングする

また、予測スケーリングと動的スケーリング (それぞれ予防的アプローチと対処的アプローチ) を組み合わせて EC2 のキャパシティを高速スケールできます。予測スケーリングを使用して、トラフィックフローの日次および週次のパターンに先立って Auto Scaling グループ内の EC2 インスタン

スの数を増やします。詳細については、「Amazon EC2 Auto Scaling の予測スケーリング」を参照してください。

Auto Scaling グループのスケーリング制限を設定する

スケーリング制限は、Auto Scaling グループに必要な最小および最大グループサイズを表します。最小サイズと最大サイズに制限を個別に設定します。

グループで希望するキャパシティは、最小および最大サイズの制限内の数値でサイズ変更できます。希望するキャパシティは、グループの最小サイズ以上、最大サイズ以下である必要があります。

- 希望するキャパシティ: Auto Scaling グループ作成時の初期キャパシティを表します。Auto Scaling グループは希望するキャパシティを維持しようとします。まず、希望するキャパシティに指定された数のインスタンスを起動します。Auto Scaling グループにスケーリングポリシーまたはスケジュールされたアクションがアタッチされていない限り、このインスタンス数が維持されます。
- 最小キャパシティ: 最小グループサイズを表します。スケーリングポリシーを設定すると、グループの希望する容量を最小容量よりも小さくすることはできません。
- 最大キャパシティ: 最大グループサイズを表します。スケーリングポリシーを設定すると、グループの希望する容量を最大容量よりも大きくすることはできません。

最小および最大サイズの制限は、次のシナリオにも適用されます。

- 希望するキャパシティを更新してAuto Scaling グループを手動でスケールする場合。
- 希望するキャパシティを更新する、スケジュールされたアクションが実行される場合。グループに新しい最小および最大サイズ制限を指定せずにスケジュールされたアクションを実行すると、グループの現在の最小サイズ制限と最大サイズ制限が適用されます。

Auto Scaling グループは常に、希望するキャパシティを維持しようとします。インスタンスが予期せず終了した場合 (スポットインスタンスの中断、ヘルスチェックの失敗、人為的なアクションなど)、グループは自動的に新しいインスタンスを起動して、希望するキャパシティを維持します。

コンソールでこれらの設定を変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインの Auto Scaling で、[Auto Scaling Groups] (Auto Scaling グループ) を選択します。

3. Auto Scaling グループページで、Auto Scaling グループの横にあるチェックボックスをオンにします。

   ページの下部にスプリットペインが開きます。

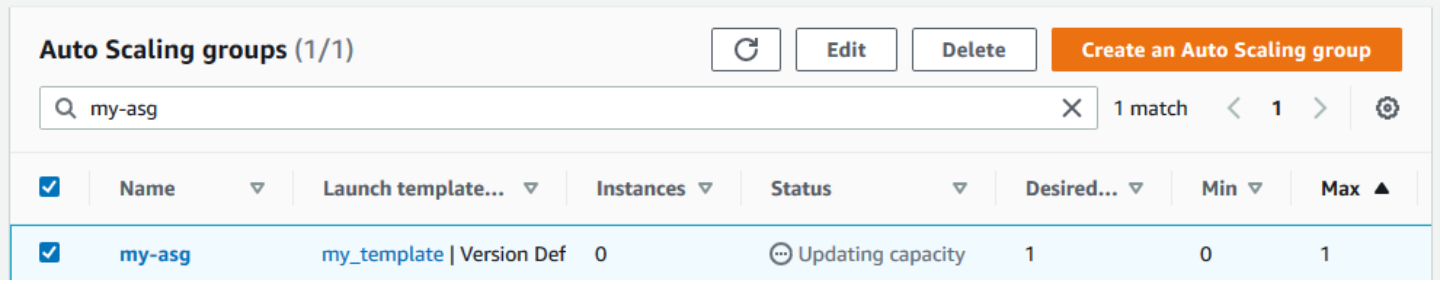
4. 下のペインの詳細タブで、グループの希望する容量、最小容量、最大容量の現在の設定を表示または変更します。詳細については、「既存の Auto Scaling グループの希望する容量を変更する」を参照してください。

詳細ペインの上には、Auto Scaling グループの現在のインスタンス数、希望する容量、最小容量、最大容量、ステータス列などの情報が表示されます。Auto Scaling グループがインスタンスの重みを使用している場合は、希望する容量に寄与した容量ユニットの数も確認できます。

一覧から列を追加または削除するには、ページ上部にある設定アイコンを選択します。次に、[Auto Scaling groups attributes] (Auto Scaling グループの属性) で各列のオンとオフを指定して、[Confirm] (確認) を選択します。

変更後に Auto Scaling グループのサイズを確認するには

[Instances (インスタンス)] 列には、現在実行中のインスタンスの数が表示されます。インスタンスの起動または終了中は、[Status (ステータス)] 列に次のイメージで示すように「Updating capacity (キャパシティーの更新)」というステータスが表示されます。



数分待ってから、ビューを更新して最新のステータスを確認します。スケーリングアクティビティが完了すると、[Instances] (インスタンス) 列に更新された値が表示されます。

[Instances] (インスタンス) の [Instance management] (インスタンス管理) タブから、インスタンス数、および現在実行されているインスタンスのステータスを表示できます。

Auto Scaling グループに対するインスタンスのデフォルトウォームアップを設定する

CloudWatch は、Auto Scaling インスタンス全体で CPU やネットワーク I/O などの使用状況データを収集して集計します。これらのメトリクスを使用して、選択したメトリクスの値の増減に応じて Auto Scaling グループ内にあるインスタンスの数を調整するスケーリングポリシーを作成します。

インスタンスが待機`InService`状態になってから、使用状況データを集約されたメトリクスに提供するまでの時間を指定できます。この指定された時間は、デフォルトのインスタンスウォームアップと呼ばれます。これにより、アプリケーショントラフィックをまだ処理しておらず、コンピューティングリソースの使用率が一時的に高まっている可能性のある個々のインスタンスのメトリクスによる動的スケーリングの影響を受けなくなります。

ターゲット追跡ポリシーとステップスケーリングポリシーのパフォーマンスを最適化するには、デフォルトのインスタンスウォームアップを有効にして設定することを強くお勧めします。デフォルトでは有効または設定されていません。

デフォルトのインスタンスウォームアップを有効にするときは、Auto Scaling グループがインスタンスメンテナンスポリシーを使用するように設定されている場合、またはインスタンスの更新を使用してインスタンスを置き換える場合、初期化が完了する前にインスタンスが最小の正常な割合にカウントされないようにできます。

内容

- パフォーマンスのスケーリングに関する考慮事項
- デフォルトのインスタンスウォームアップ時間を選択する
- グループに対するインスタンスのデフォルトウォームアップを有効にする
- グループに対するインスタンスのデフォルトウォームアップを検証する
- 以前にインスタンスのウォームアップ時間を設定してスケーリングポリシーを検索する
- 以前に設定したスケーリングポリシーのインスタンスウォームアップをクリアする

パフォーマンスのスケーリングに関する考慮事項

ほとんどのアプリケーションでは、機能ごとに異なるウォームアップ時間ではなく、すべての機能に適用されるデフォルトのインスタンスウォームアップ時間を 1 つ持つと便利です。例えば、デフォルトのインスタンスウォームアップを設定しない場合、インスタンスの更新機能はヘルスチェック

の猶予期間をデフォルトのウォームアップ時間として使用します。ターゲット追跡ポリシーとステップスケーリングポリシーがある場合は、デフォルトのクールダウンに設定された値をデフォルトのウォームアップ時間として使用します。予測スケーリングポリシーがある場合、デフォルトのウォームアップ時間はありません。

インスタンスのウォームアップ中、動的スケーリングポリシーは、ウォームアップしていないインスタンスのメトリクス値がポリシーのアラーム上限しきい値 (またはターゲット追跡スケーリングポリシーのターゲット使用率) より大きい場合にのみスケールアウトします。需要が減少すると、動的スケーリングはアプリケーションの可用性を保護するためにより保守的になります。これにより、新しいインスタンスのウォームアップが完了するまで、動的スケーリングのスケールインアクティビティがブロックされます。

スケールアウト中、Amazon EC2 Auto Scaling は、グループに追加するインスタンスの数を決定するときに、ウォームアップ中のインスタンスをグループの容量の一部として考慮します。したがって、同様の容量を追加する必要がある複数のアラーム違反が発生すると、1 つのスケーリングアクティビティが発生します。その目的は、過剰にスケールアウトすることなく、継続的にスケールアウトすることです。

デフォルトのインスタンスウォームアップが有効になっていない場合、メトリクスを に送信 CloudWatch して現在の容量にカウントするまでのインスタンスの待機時間は、インスタンスごとに異なります。したがって、スケーリングポリシーが、実際に発生しているワークロードと比較して、予測不能に実行される可能性があります。

例えば、反復的な on-and-off ワークロードパターンを持つアプリケーションを考えてみましょう。予測スケーリングポリシーを使用して、インスタンス数を増やすかどうかを繰り返し決定します。予測スケーリングポリシーにはデフォルトのウォームアップ時間がないため、インスタンスは集約されたメトリクスにすぐに寄与し始めます。これらのインスタンスの起動時のリソース使用量が多い場合、インスタンスを追加すると、集約されたメトリックスが急増する可能性があります。使用量が安定するまでにかかる時間によっては、これらの指標を使用する動的スケーリングポリシーに影響する可能性があります。動的スケーリングポリシーのアラーム上限しきい値を超えると、グループのサイズは再び大きくなります。新しいインスタンスがウォームアップしている間、スケールインアクティビティはブロックされます。

デフォルトのインスタンスウォームアップ時間を選択する

デフォルトのインスタンスのウォームアップを設定する上で重要なのは、インスタンスが `InService` の状態に達した後、初期化を終了し、リソースの消費が安定するまでに必要な時間を決定することです。インスタンスのウォームアップ時間を選択するときは、正当なトラフィックの使

用状況データの収集と、起動時の一時的な使用量の急増に関連するデータ収集の最小化の間で最適なバランスを取るようにしてください。

Auto Scaling グループが Elastic Load Balancing ロードバランサーにアタッチされているとします。新しいインスタンスが起動を完了すると、`InService` 状態に入る前にロードバランサーに登録されます。インスタンスが `InService` 状態になった後も、リソースの消費は引き続き一時的に急増する場合があり、安定化する時間が必要です。例えば、大量のアセットをダウンロードしてキャッシュする必要があるアプリケーションサーバーのリソース消費が安定するまでにかかる時間は、ダウンロードする大量のアセットがない軽量のウェブサーバーよりも長くなります。インスタンスのウォームアップは、リソース消費の安定化に必要な遅延時間を提供します。

> **Important**
>
> ウォームアップ時間に必要な時間がわからない場合は、300 秒から始めることができます。次に、アプリケーションに最適なスケーリングパフォーマンスが得られるまで、徐々に減らすか、増やします。正しく処理するには、これを数回実行する必要がある場合があります。または、独自のウォームアップ時間 (`EstimatedInstanceWarmup`) を持つスケーリングポリシーがある場合は、この値を使用して開始できます。詳細については、「以前にインスタンスのウォームアップ時間を設定してスケーリングポリシーを検索する」を参照してください。

起動時に実行する設定タスクまたはスクリプトがあるユースケースでは、ライフサイクルフックの使用を検討してください。ライフサイクルフックは、新しいインスタンスが初期化が完了するまで稼働を停止する可能性があります。ライフサイクルフックは特に、完了に時間がかかるブートストラップスクリプトがある場合に便利です。ライフサイクルフックを追加すると、デフォルトのインスタンスウォームアップの値を減らすことができます。ライフサイクルフック使用の詳細については、「Amazon EC2 Auto Scaling のライフサイクルフック」を参照してください。

グループに対するインスタンスのデフォルトウォームアップを有効にする

インスタンスのデフォルトウォームアップは、Auto Scaling グループの作成時に有効化できます。既存のグループに対して有効化することも可能です。

デフォルトのインスタンスウォームアップ機能を有効にすると、次の機能のウォームアップパラメータの値を指定する必要がなくなります。

- インスタンスの更新
- ターゲットトラッキングスケーリング

- ステップスケーリング

Console

新しいグループに対してインスタンスのデフォルトウォームアップを有効にする (コンソール)

Auto Scaling グループを作成するときに、[Configure advanced options] (詳細オプションを設定) ページの [Additional settings] (追加設定)で、[Enable default instance warmup] (インスタンスのデフォルトウォームアップを有効にする) オプションを選択します。アプリケーションに必要なウォームアップ時間を選択します。

AWS CLI

新しいグループに対してインスタンスのデフォルトウォームアップを有効にする (AWS CLI)

Auto Scaling グループに対してインスタンスのデフォルトウォームアップを有効にするには、`--default-instance-warmup` オプションを追加して、0 から 3600 までの値 (秒単位) を指定します。`-1` の値は、有効化後にこの設定をオフにします。

以下の create-auto-scaling-group コマンドは、*my-asg* という名前の Auto Scaling グループを作成し、*120* 秒の値でインスタンスのデフォルトウォームアップを有効にします。

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-asg --
default-instance-warmup 120 ...
```

> **Tip**
>
> このコマンドでエラーが発生した場合は、 を AWS CLI ローカルで最新バージョンに更新していることを確認してください。

Console

既存のグループに対してインスタンスのデフォルトウォームアップを有効にする (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. 画面の上部のナビゲーションバーで、Auto Scaling グループを作した AWS リージョン を選択します。
3. Auto Scaling グループの横にあるチェックボックスを選択します。

ページの下部にスプリットペインが開きます。

4. [詳細] タブで、[高度な設定]、[編集] の順に選択します。
5. デフォルトのインスタンスウォームアップ で、アプリケーションに必要なウォームアップ時間を選択します。
6. [更新] を選択します。

AWS CLI

既存のグループに対してインスタンスのデフォルトウォームアップを有効にする (AWS CLI)

以下の例は、update-auto-scaling-group コマンドを使用して、*my-asg* という名前の既存の Auto Scaling グループに対し、*120* 秒の値でインスタンスのデフォルトウォームアップを有効にします。

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg --
default-instance-warmup 120
```

> **Tip**
>
> このコマンドでエラーが発生した場合は、 を AWS CLI ローカルで最新バージョンに更新していることを確認してください。

グループに対するインスタンスのデフォルトウォームアップを検証する

Auto Scaling グループに対するインスタンスのデフォルトウォームアップを検証する (AWS CLI)

次の describe-auto-scaling-groups コマンドを使用します。*my-asg* を Auto Scaling グループの名前に置き換えます。

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
```

以下に、応答の例を示します。

```
{
    "AutoScalingGroups": [
        {
```

```
            "AutoScalingGroupName": "my-asg",
            "AutoScalingGroupARN": "arn",
            ...
            "DefaultInstanceWarmup": 120
        }
    ]
}
```

以前にインスタンスのウォームアップ時間を設定してスケーリングポリシーを検索する

に独自のウォームアップ時間を持つポリシーがあるかどうかを特定するには`EstimatedInstanceWarmup`、 を使用して次の describe-policies コマンドを実行します AWS CLI。*my-asg* を Auto Scaling グループの名前に置き換えます。

```
aws autoscaling describe-policies --auto-scaling-group-name my-asg
  --query 'ScalingPolicies[?EstimatedInstanceWarmup!=`null`]'
```

以下は出力例です。

```
[
  {
    "AutoScalingGroupName":"my-asg",
    "PolicyName":"cpu50-target-tracking-scaling-policy",
    "PolicyARN":"arn",
    "PolicyType":"TargetTrackingScaling",
    "StepAdjustments":[],
    "EstimatedInstanceWarmup":120,
    "Alarms":[{
        "AlarmARN": "arn:aws:cloudwatch:us-west-2:123456789012:alarm:TargetTracking-my-
asg-AlarmHigh-fc0e4183-23ac-497e-9992-691c9980c38e",
        "AlarmName": "TargetTracking-my-asg-AlarmHigh-
fc0e4183-23ac-497e-9992-691c9980c38e"
      },
      {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmLow-61a39305-ed0c-47af-
bd9e-471a352ee1a2",
            "AlarmName": "TargetTracking-my-asg-AlarmLow-61a39305-ed0c-47af-
bd9e-471a352ee1a2"
    }],
```

```
    "TargetTrackingConfiguration":{
      "PredefinedMetricSpecification":{
        "PredefinedMetricType":"ASGAverageCPUUtilization"
      },
      "TargetValue":50.0,
      "DisableScaleIn":false
    },
    "Enabled":true
  },

    ... additional policies ...

]
```

以前に設定したスケーリングポリシーのインスタンスウォームアップをクリアする

インスタンスのデフォルトウォームアップを有効にした後、まだウォームアップ時間があるスケーリングポリシーを更新して、以前に設定した値をクリアします。そうしないと、デフォルトのインスタンスウォームアップがオーバーライドされます。

スケーリングポリシーは、コンソール、 AWS CLI、または AWS SDKsを使用して更新できます。このセクションでは、コンソールの手順について説明します。 AWS CLI または AWS SDKsを使用する場合は、既存のポリシー設定を保持し、 `EstimatedInstanceWarmup`プロパティを削除してください。既存のスケーリングポリシーを更新すると、ポリシーは、プログラムでポリシー を呼び出すときに指定したポリシーに置き換えられますPutScaling。元の値は保持されません。

以前に設定したスケーリングポリシーのインスタンスウォームアップをクリアする

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。
3. [自動スケーリング] タブの [動的スケーリングポリシー] で、目的のポリシーを選択し、[アクション]、[編集] を選択します。
4. インスタンスウォームアップ では、代わりにデフォルトのインスタンスウォームアップ値を使用するには、インスタンスウォームアップ値をクリアします。
5. [更新] を選択します。

Amazon EC2 Auto Scaling の手動スケーリング

Auto Scaling グループ内の EC2 インスタンスの数は、いつでも手動で調整できます。インスタンス数を手動で変更するこのプロセスは、手動スケーリング と呼ばれます。手動スケーリングは、特に 1 回限りの容量変更を行う場合に、自動スケーリングの代替手段です。

グループを手動でスケーリングすると、Amazon EC2 Auto Scaling は、定義したスケーリングポリシーとスケジュールされたアクションに基づいて、通常の自動スケーリングアクティビティを再開します。インスタンスのデフォルトウォームアップが有効になっているグループの場合、新しいインスタンスは、自動スケーリングに使用されるメトリクスへの寄与を開始する前にウォームアップ期間を経過します。このウォームアップ期間は、新しい容量でグループを安定化するのに役立ちます。詳細については、「Auto Scaling グループに対するインスタンスのデフォルトウォームアップを設定する」を参照してください。

グループを手動でスケーリングする前に、スケーリングポリシーとスケジュールされたアクションを一時的に無効にしたい場合があります。これにより、手動スケーリングアクションと自動スケーリングアクティビティの間に競合が発生するのを防ぐことができます。詳細については、「スケーリングアクティビティをオフにする」を参照してください。

内容

- 既存の Auto Scaling グループの希望する容量を変更する
- Auto Scaling グループのインスタンスを終了する (AWS CLI)

既存の Auto Scaling グループの希望する容量を変更する

Auto Scaling グループの希望する容量を変更すると、Amazon EC2 Auto Scaling はインスタンスの起動と終了のプロセスを管理し、新しい希望するサイズに到達します。

Console

Auto Scaling グループのサイズを変更するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部に分割ペインが表示されます。
3. [詳細] タブで、[グループの詳細]、[編集] の順に選択します。

4. 希望する容量 では、希望する容量を増減します。例えば、グループのサイズを 1 つ増やすには、現在の値が の場合1、 と入力します2。

   希望する容量の新しい値が、希望する最小容量と希望する最大容量 より大きい場合、希望する最大容量は自動的に新しい希望する容量値に増加します。

5. 完了したら、[更新] を選択します。

指定したグループサイズで、同じ量のインスタンスが起動されたことを確認します。例えば、グループのサイズを 1 つ増やした場合は、Auto Scaling グループが追加のインスタンスを 1 つ起動していることを確認します。

Auto Scaling グループのサイズが変更されたことを確認するには

1. アクティビティタブのアクティビティ履歴 で、Auto Scaling グループに関連付けられているアクティビティの進行状況を表示できます。[ステータス] 列には、インスタンスの現在のステータスが表示されます。インスタンスが起動している間、ステータス列には [`Not yet in service`] と表示されます。ステータスは、インスタンスが起動されると `Successful` に変わります。更新アイコンを使用して、インスタンスの現在のステータスを確認することもできます。詳細については、「Auto Scaling グループのスケーリングアクティビティを検証する」を参照してください。
2. インスタンス管理タブのインスタンス で、インスタンスのステータスを表示できます。インスタンスの起動には短時間かかります。

   - [ライフサイクル] 列には、インスタンスの状態が表示されます。最初、インスタンスの状態は `Pending` です。インスタンスがトラフィックを受信できるようになったら、そのステータスは `InService` です。
   - ヘルスステータス列には、インスタンスの Amazon EC2 Auto Scaling ヘルスチェックの結果が表示されます。

AWS CLI

以下の例では、最小サイズが 1 で、最大サイズが 5 である Auto Scaling グループを作成したことを前提としています。したがって、このグループでは現在インスタンスを実行中です。

Auto Scaling グループのサイズを変更するには

以下の例に示すように、set-desired-capacity コマンドを使用して Auto Scaling グループのサイズを変更します。

```
aws autoscaling set-desired-capacity --auto-scaling-group-name my-asg \
  --desired-capacity 2
```

Auto Scaling グループのデフォルトのクールダウン期間を受け入れることを選択した場合は、以下の例に示しているように --honor-cooldown オプションを指定する必要があります。詳細については、「Amazon EC2 Auto Scaling のスケーリングクールダウン」を参照してください。

```
aws autoscaling set-desired-capacity --auto-scaling-group-name my-asg \
  --desired-capacity 2 --honor-cooldown
```

Auto Scaling グループのサイズを確認するには

以下の例のように、describe-auto-scaling-groups コマンドを使用して、Auto Scaling グループのサイズが変更されたことを確認します。

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
```

以下は出力例で、起動されたグループとインスタンスの詳細を示します。

```
{
    "AutoScalingGroups": [
        {
            "AutoScalingGroupName": "my-asg",
            "AutoScalingGroupARN": "arn",
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1",
                "LaunchTemplateId": "lt-050555ad16a3f9c7f"
            },
            "MinSize": 1,
            "MaxSize": 5,
            "DesiredCapacity": 2,
            "DefaultCooldown": 300,
            "AvailabilityZones": [
                "us-west-2a"
            ],
            "LoadBalancerNames": [],
            "TargetGroupARNs": [],
            "HealthCheckType": "EC2",
            "HealthCheckGracePeriod": 300,
            "Instances": [
```

```
                {
                    "ProtectedFromScaleIn": false,
                    "AvailabilityZone": "us-west-2a",
                    "LaunchTemplate": {
                        "LaunchTemplateName": "my-launch-template",
                        "Version": "1",
                        "LaunchTemplateId": "lt-050555ad16a3f9c7f"
                    },
                    "InstanceId": "i-05b4f7d5be44822a6",
                    "InstanceType": "t3.micro",
                    "HealthStatus": "Healthy",
                    "LifecycleState": "Pending"
                },
                {
                    "ProtectedFromScaleIn": false,
                    "AvailabilityZone": "us-west-2a",
                    "LaunchTemplate": {
                        "LaunchTemplateName": "my-launch-template",
                        "Version": "1",
                        "LaunchTemplateId": "lt-050555ad16a3f9c7f"
                    },
                    "InstanceId": "i-0c20ac468fa3049e8",
                    "InstanceType": "t3.micro",
                    "HealthStatus": "Healthy",
                    "LifecycleState": "InService"
                }
            ],
            "CreatedTime": "2019-03-18T23:30:42.611Z",
            "SuspendedProcesses": [],
            "VPCZoneIdentifier": "subnet-c87f2be0",
            "EnabledMetrics": [],
            "Tags": [],
            "TerminationPolicies": [
                "Default"
            ],
            "NewInstancesProtectedFromScaleIn": false,
            "ServiceLinkedRoleARN": "arn",
            "TrafficSources": []
        }
    ]
}
```

DesiredCapacity が新しい値を示していることに注意してください。また、Auto Scaling グループによって追加のインスタンスが起動されています。

Auto Scaling グループのインスタンスを終了する (AWS CLI)

Auto Scaling グループを手動でスケールインしたいが、特定のインスタンスを終了したい場合があります。次の例に示すように、terminate-instance-in-auto-scaling-group コマンドを使用し、終了するインスタンスの ID と --should-decrement-desired-capacity オプションを指定することで、Auto Scaling グループを手動でスケールインできます。

```
aws autoscaling terminate-instance-in-auto-scaling-group \
  --instance-id i-026e4c9f62c3e448c --should-decrement-desired-capacity
```

以下は、スケーリングアクティビティの詳細を提供する出力例です。

```
{
    "Activities": [
        {
            "ActivityId": "b8d62b03-10d8-9df4-7377-e464ab6bd0cb",
            "AutoScalingGroupName": "my-asg",
            "Description": "Terminating EC2 instance: i-026e4c9f62c3e448c",
            "Cause": "At 2023-09-23T06:39:59Z instance i-026e4c9f62c3e448c was taken
 out of service in response to a user request, shrinking the capacity from 1 to 0.",
            "StartTime": "2023-09-23T06:39:59.015000+00:00",
            "StatusCode": "InProgress",
            "Progress": 0,
            "Details": "{\"Subnet ID\":\"subnet-6194ea3b\",\"Availability Zone\":\"us-
west-2c\"}"
        }
    ]
}
```

このオプションはコンソールでは利用できません。ただし、Amazon EC2 コンソールのインスタンスページを使用して、Auto Scaling グループのインスタンスを終了できます。これを行うと、Amazon EC2 Auto Scaling はインスタンスが実行されなくなったことを検出し、ヘルスチェッククプロセスの一環として自動的に置き換えます。新しいインスタンスが起動されるまでに、インスタンスを終了してから 1～2 分かかります。インスタンスを終了する方法については、「Amazon EC2 ユーザーガイド」の「インスタンスの終了」を参照してください。 Amazon EC2

グループ内のインスタンスを終了し、アベイラビリティーゾーン間で分散が不均等になる場合、Amazon EC2 Auto Scaling はAZRebalanceプロセスを停止しない限り、グループのバランスを再調整して均等分散を再確立します。詳細については、「Amazon EC2 Auto Scaling プロセスの一時停止と再開」を参照してください。

Amazon EC2 Auto Scaling のスケジュールされたスケーリング

スケジュールされたスケーリングでは、予測可能な負荷の変化に基づいてアプリケーションの自動スケーリングを設定できます。特定の時間にグループの希望するキャパシティーを増減するスケジュールされたアクションを作成します。

例えば、毎週定期的にトラフィックパターンが発生すると、負荷は週の途中で増加し、週の終わりに向かって減少します。Amazon EC2 Auto Scaling では、このパターンに沿ったスケーリングスケジュールを設定できます。

- 水曜日の午前に、1 つのスケジュールされたアクションがAuto Scaling グループの以前に設定した希望する容量を増やすことで容量を増やします。
- 金曜日の夜、別のスケジュールされたアクションはAuto Scaling グループの以前に設定した希望するキャパシティを減らすことでキャパシティを減らします。

これらのスケジュールされたスケーリングアクションにより、コストとパフォーマンスを最適化できます。アプリケーションには、週半ばのトラフィックのピークを処理するのに十分な容量がありますが、それ以外の場合は不要な容量を過剰にプロビジョニングすることはありません。

スケジュールされたスケーリングポリシーとスケーリングポリシーを一緒に使用して、スケーリングへの両方のアプローチの利点を得ることができます。スケジュールされたスケーリングアクションの実行後、スケーリングポリシーは容量をさらにスケールするかどうかの判断を引き続き行うことができます。これは、アプリケーションの負荷を処理するために十分な容量を確保する上で役立ちます。アプリケーションは需要に合わせてスケールしますが、現行の容量は、スケジュールされたアクションによって設定された最小容量と最大容量内に収まる必要があります。

内容

- スケジュールされたスケーリングのしくみ
- 繰り返しのスケジュール
- Time zone (タイムゾーン)
- 考慮事項

- スケジュールされたアクションの作成
- スケジュールされたアクションの詳細を表示する
- スケーリングアクティビティを検証する
- スケジュールされたアクションの削除
- 制限事項

スケジュールされたスケーリングのしくみ

スケジュールされたスケーリングを使用するには、スケジュールされたアクション を作成します。これにより、Amazon EC2 Auto Scaling に特定の時間にスケーリングアクティビティを実行するように指示します。スケジュールされたアクションを作成するときは、Auto Scaling グループ、スケーリングアクティビティが発生するタイミング、新しい希望する容量、オプションで新しい最小容量と新しい最大容量を指定します。スケジュールされたアクションは、1 度だけスケールする、または定期的なスケジュールに従ってスケールするものを作成できます。

指定された時点で、Amazon EC2 Auto Scaling は、現在の容量を指定された希望する容量と比較することで、新しい容量値に基づいてスケーリングします。

- 現在の容量が指定された希望する容量よりも小さい場合、Amazon EC2 Auto Scaling は指定された希望する容量にインスタンスをスケールアウトまたは追加します。
- 現在の容量が指定された希望する容量より大きい場合、Amazon EC2 Auto Scaling は指定された希望する容量にインスタンスをスケールインまたは削除します。

スケジュールされたアクションは、指定された日時におけるグループの希望容量、最小容量、最大容量を設定します。一度に作成できるスケジュールされたアクションは、希望するキャパシティなど、これらのキャパシティの 1 つだけです。ただし、アクションで指定した希望する容量がこれらの制限を超えないように、最小容量と最大容量を含める必要がある場合があります。

繰り返しのスケジュール

AWS CLI または SDK を使用して定期的なスケジュールを作成するには、cron 式とタイムゾーンを指定して、スケジュールされたアクションがいつ繰り返されるかを記述します。必要に応じて、開始時刻、終了時刻、またはその両方の日付と時刻を指定できます。

を使用して定期的なスケジュールを作成するには AWS Management Console、スケジュールされたアクションの繰り返しパターン、タイムゾーン、開始時刻、およびオプションの終了時刻を指定しま

す。すべての反復パターンオプションは、cron 式に基づいています。または、独自のカスタム cron 式を記述することもできます。

このサポートされた cron 式は、スペースで区切られた 5 つのフィールド ([分] [時間] [日] [月] [曜日]) で構成されます。例えば、cron 式 `30 6 * * 2` は毎週火曜日の午前 6:30 に繰り返されるスケジュールされたアクションを設定します。アスタリスクは、フィールドのすべての値を照合するワイルドカードとして使用されます。cron 式の他の例については、(https://crontab.guru/examples.html) を参照してください。この形式で独自の cron 式を記述する方法については、クロンタブを参照してください。

開始時刻と終了時刻を慎重に選択します。以下に留意してください。

- 開始時刻を指定した場合、Amazon EC2 Auto Scaling はこの時刻にアクションを実行し、指定された反復に基づくアクションを実行します。
- 終了時刻を指定すると、その時刻以降はアクションが反復されなくなります。スケジュールされたアクションは、終了時刻に達するとアカウントに残りません。
- AWS CLI または SDK を使用する場合は、開始時刻と終了時刻を UTC で設定する必要があります。

Time zone (タイムゾーン)

デフォルトでは、設定した定期的なスケジュールは協定世界時 (UTC) です。ローカルタイムゾーンまたはネットワークの他の部分のタイムゾーンに対応するタイムゾーンに変更できます。夏時間 (DST) を遵守するタイムゾーンを指定すると、DST に合わせてアクションが自動的に調整されます。

有効な値は、Internet Assigned Numbers Authority (IANA) Time Zone データベースのタイムゾーンの正規名です。例えば、米国東部時間は正規に として識別されます`America/New_York`。詳細については、https://www.iana.org/time-zones を参照してください。

などの場所ベースのタイムゾーンは、DST に合わせて`America/New_York`自動的に調整されます。ただし、`Etc/UTC` のようなUTC ベースのタイムゾーン は絶対時刻であるような DST に調整されません。

例えば、タイムゾーンが `America/New_York` である繰り返しのスケジュールがります。最初のスケーリングアクションは、DST が始まる前の `America/New_York` でのタイムゾーンで発生します。次のスケーリングアクションは、DST が始まった後の `America/New_York` でのタイムゾーン

で発生します。最初のアクションは現地時間の午前 8:00 UTC-5 から始まり、2 番目のアクションは現地時間の午前 8:00 UTC-4 から始まります。

を使用してスケジュールされたアクションを作成し、DST を監視するタイムゾーンを指定する AWS Management Console と、定期的なスケジュールと開始時刻と終了時刻の両方が DST に合わせて自動的に調整されます。

考慮事項

スケジュールされたアクションを作成する場合、次の点に注意してください。

- スケジュールされたアクションの実行順序は、それらのアクションが同じグループ内で実行される場合は維持されますが、複数のグループ間で実行される場合は必ずしも維持されません。
- スケジュールされたアクションは通常、数秒以内に実行されます。ただし、アクションは、スケジュールされた開始時間から最大 2 分遅れる場合があります。Auto Scaling スケジュールされたグループ内のアクションは指定された順序で実行されるため、開始時間が互いに近い、スケジュールされたアクションの実行には時間がかかる可能性があります。
- Auto Scaling グループのスケジュールされたスケーリングを一時的にオフにするには、`ScheduledActions` プロセスを一時的停止します。これにより、スケジュールされたアクションを削除せずにアクティブになるのを防ぐことができます。スケジュールされたスケーリングを再度使用する場合は、スケジュールされたスケーリングを再開できます。詳細については、「Amazon EC2 Auto Scaling プロセスの一時停止と再開」を参照してください。
- スケジュールされたアクションを作成した後、名前以外のすべての設定を更新できます。

スケジュールされたアクションの作成

Auto Scaling グループにスケジュールされたアクションを作成するには、次のいずれかの方法を使用します。

Console

スケジュールされたアクションを作成するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。

3. [Automatic scaling (自動スケーリング)] タブの [Scheduled actions (スケジュールされたアクション)] で、[Create scheduled action (スケジュールされたアクションの作成)] を選択します。
4. スケジュールされたアクションに [Name (名前)] を入力します。
5. 希望する容量 、最小 、最大 で、グループの新しい希望する容量と、新しい最小サイズ制限と最大サイズ制限を選択します。希望するキャパシティは、グループの最小サイズ以上、最大サイズ以下である必要があります。
6. [Recurrence (反復)] で、使用可能なオプションの 1 つを選択します。

   - 定期的なスケジュールに基づいてスケーリングする場合は、Amazon EC2 Auto Scaling がスケジュールされたアクションを実行する頻度を選択します。
     - [Every (毎)] で始まるオプションを選択した場合、cron 式が作成されます。
     - [Cron] を選択した場合は、いつアクションを実行するかを Cron 式を入力します。
   - スケーリングを 1 回だけ行う場合は、[Once (一度)] を選択します。
7. [Time zone (タイムゾーン)] でタイムゾーンを選択。デフォルト値は `Etc/UTC` です。

   リストされているすべてのタイムゾーンは、IANA タイムゾーンデータベースから取得されます。詳細については、https://en.wikipedia.org/wiki/List_of_tz_database_time_zones を参照してください。
8. 特定の開始時間には、以下の日付と時刻を定義します。

   - 定期的なスケジュールを選択した場合、開始時間によって、定期的なシリーズの最初のスケジュールされたアクションが実行されるタイミングが定義されます。
   - [Once (一度)] を反復として使用する場合、開始時刻は、スケジュールアクションを実行する日付と時刻を定義します。
9. (オプション) 定期的なスケジュールの場合は、[Set End Time (終了時刻の設定)] を選択して終了時間を特定し、[End By (までに終了)] に日付と時刻を選択します。
10. [作成] を選択します。コンソールに Auto Scaling グループのスケジュールされたアクションが表示されます。

AWS CLI

スケジュールされたアクションを作成するには、次のいずれかのサンプルコマンドを使用できます。各###############を独自の情報に置き換えます。

例: 1 回のみスケールするには

`--start-time "YYYY-MM-DDThh:mm:ssZ"` および `--desired-capacity`オプションを指定して、次の put-scheduled-update-group-action コマンドを使用します。

```
aws autoscaling put-scheduled-update-group-action --scheduled-action-name my-one-time-action \
  --auto-scaling-group-name my-asg --start-time "2021-03-31T08:00:00Z" --desired-capacity 3
```

例: 定期的なスケジュールでスケーリングをスケジュールするには

`--recurrence "cron expression"` および `--desired-capacity`オプションを指定して、次の put-scheduled-update-group-action コマンドを使用します。

```
aws autoscaling put-scheduled-update-group-action --scheduled-action-name my-recurring-action \
  --auto-scaling-group-name my-asg --recurrence "0 9 * * *" --desired-capacity 3
```

デフォルトでは、Amazon EC2 Auto Scaling は UTC タイムゾーンに基づいて指定された繰り返しスケジュールを実行します。別のタイムゾーンを指定するには、次の例のように、 `--time-zone`オプションと IANA タイムゾーンの名前を含めます。

```
--time-zone "America/New_York"
```

詳細については、https://en.wikipedia.org/wiki/List_of_tz_database_time_zones を参照してください。

スケジュールされたアクションの詳細を表示する

Auto Scaling グループの今後のスケジュールされたアクションの詳細を表示するには、次のいずれかの方法を使用します。

Console

スケジュールされたアクションの詳細を表示するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling スケーリンググループを選択します。

3. 「自動スケーリング」タブの「スケジュールされたアクション」セクションで、今後のスケジュールされたアクションを表示できます。

コンソールには、ローカル時間の開始時刻と終了時刻の値が表示され、指定した日時で UTC オフセットが有効であることに注意してください。UTC オフセットは、現地時間から UTC までの差を時間と分単位で表したものです。タイムゾーンの値は、例えば America/New_York のようにリクエストされたタイムゾーンが表示されます。

AWS CLI

次の describe-scheduled-actions コマンドを使用します。

```
aws autoscaling describe-scheduled-actions --auto-scaling-group-name my-asg
```

正常に完了した場合、このコマンドは以下のような出力を返します。

```
{
  "ScheduledUpdateGroupActions": [
    {
      "AutoScalingGroupName": "my-asg",
      "ScheduledActionName": "my-recurring-action",
      "Recurrence": "30 0 1 1,6,12 *",
      "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8e86b655-b2e6-4410-8f29-
b4f094d6871c:autoScalingGroupName/my-asg:scheduledActionName/my-recurring-action",
      "StartTime": "2020-12-01T00:30:00Z",
      "Time": "2020-12-01T00:30:00Z",
      "MinSize": 1,
      "MaxSize": 6,
      "DesiredCapacity": 4
    }
  ]
}
```

スケーリングアクティビティを検証する

スケジュールされたスケーリングに関連するスケーリングアクティビティを確認するには、「」を参照してくださいAuto Scaling グループのスケーリングアクティビティを検証する。

スケジュールされたアクションの削除

スケジュールされたアクションを削除するには、次のいずれかの方法を使用します。

Console

スケジュールされたアクションを削除するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling スケーリンググループを選択します。
3. [Automatic scaling (自動スケーリング)] タブの [Scheduled actions (スケジュールされたアクション)] で、スケジュールされたアクションを選択します。
4. [Actions] で、[Delete] を選択します。
5. 確認を求めるメッセージが表示されたら、[Yes、Delete] を選択します。

AWS CLI

次の delete-scheduled-action コマンドを使用します。

```
aws autoscaling delete-scheduled-action --auto-scaling-group-name my-asg \
  --scheduled-action-name my-recurring-action
```

制限事項

- スケジュールされたアクションの名前は、Auto Scaling グループごとに一意である必要があります。
- スケジュールされたアクションには、一意の時間値が必要です。別のスケーリングアクティビティがすでにスケジュールされているときにアクティビティをスケジュールしようとすると、呼び出しは拒否され、このスケジュールされた開始時間を持つスケジュールされたアクションがすでに存在していることを示すエラーが返されます。
- 1 つの Auto Scaling グループあたり最大 125 のスケジュールされたアクションを作成できます。

Amazon EC2 Auto Scaling の動的スケーリング

動的スケーリングは、トラフィックの変化に応じて Auto Scaling グループのキャパシティをスケールします。

Amazon EC2 Auto Scaling は、以下のタイプの動的スケーリングポリシーをサポートしています:

- ターゲット追跡スケーリング — Amazon CloudWatch メトリクスとターゲット値に基づいて、グループの現在の容量を増減します。これはサーモスタットで家の温度を管理する方法と似ています (温度を選択すれば、後はサーモスタットがすべてを実行します)。
- ステップスケーリング - アラーム違反の規模に応じて変動する一連のスケーリング調整値 (ステップ調整値) に基づいて、グループの現在のキャパシティを増減させます。
- シンプルなスケーリング - 1 つのスケーリング調整値に基づいて、各スケーリングアクティビティ間のクールダウン期間により、グループの現在のキャパシティを増減させます。

ターゲット追跡スケーリングポリシーを使用し、Auto Scaling グループの容量の変化に反比例して変化するメトリクスを選択することを強くお勧めします。したがって、Auto Scaling グループのサイズを 2 倍にすると、メトリクスは 50% 減少します。これにより、メトリクスデータが比例スケーリングイベントを正確にトリガーできます。ターゲットあたりの平均 CPU 使用率や平均リクエスト数などのメトリクスが含まれます。

ターゲット追跡では、Auto Scaling グループはアプリケーションの実際の負荷に正比例してスケーリングされます。つまり、ターゲット追跡ポリシーは、負荷の変化に対応するキャパシティの緊急のニーズを満たすだけでなく、季節変動などにより経時的に発生する負荷の変化にも対応できます。

ターゲット追跡ポリシーでは、 CloudWatch アラームとスケーリング調整を手動で定義する必要もなくなります。Amazon EC2 Auto Scaling は、設定したターゲットに基づいてこれを自動的に処理します。

内容

- 動的スケーリングポリシーが機能する方法
- 複数の動的スケーリングポリシー
- Amazon EC2 Auto Scaling のターゲットトラッキングスケーリングポリシー
- Amazon EC2 Auto Scaling のステップおよび簡易スケーリングポリシー
- Amazon EC2 Auto Scaling のスケーリングクールダウン
- Amazon SQS に基づくスケーリング

- Auto Scaling グループのスケーリングアクティビティを検証する
- Auto Scaling グループのスケーリングポリシーを無効化する
- スケーリングポリシーを削除する
- AWS Command Line Interface (AWS CLI) のスケーリングポリシーの例

動的スケーリングポリシーが機能する方法

動的スケーリングポリシーは、特定の CloudWatch メトリクスを追跡するように Amazon EC2 Auto Scaling に指示し、関連付けられた CloudWatch アラームが ALARM にあるときに実行するアクションを定義します。アラームの状態の呼び出しに使用されるメトリクスは、Auto Scaling グループのすべてのインスタンスから取得されるメトリクスの集約です。(例えば、1 つのインスタンスが 60% の CPU で、もう 1 つのインスタンスが 40% の CPU である 2 つのインスタンスを持つ Auto Scaling グループがあるとします。平均して、これらは 50 パーセント CPU です。) ポリシーが有効な場合、アラームのしきい値を超えると、Amazon EC2 Auto Scaling はグループの希望するキャパシティーの増減を調整します。

動的スケーリングポリシーが呼び出されるとき、キャパシティー計算によってグループの最小サイズおよび最大サイズ範囲外の数値が生成される場合、Amazon EC2 Auto Scaling は、新しいキャパシティーが最小サイズおよび最大サイズ制限の範囲外にならないようにします。容量は、インスタンスに関して希望する容量を設定するときに選択したのと同じ単位を使用するか、容量単位を使用するか (インスタンスの重みが適用されている場合) の 2 つの方法のいずれかで測定されます。

- 例 1: Auto Scaling グループの最大キャパシティーは 3、現在のキャパシティーは 2、および 3 つのインスタンスを追加する動的スケーリングポリシーがあります。このスケーリングポリシーを呼び出すとき、Amazon EC2 Auto Scaling は、グループが最大サイズを超えないようにするために、グループに 1 つのインスタンスのみを追加します。
- 例 2: Auto Scaling グループの最小キャパシティーは 2、現在のキャパシティーは 3、および 2 つのインスタンスを削除する動的スケーリングポリシーがあります。このポリシーを呼び出す場合、Amazon EC2 Auto Scaling はグループから 1 つのインスタンスのみを削除して、グループが最小サイズより小さくならないようにします。

希望するキャパシティーが最大サイズ制限に達すると、スケールアウトは停止します。需要が低下し、キャパシティーが減少した場合、Amazon EC2 Auto Scaling は再びスケールアウトできます。

例外は、インスタンスの重みを使用する場合です。この場合、Amazon EC2 Auto Scaling は最大サイズ制限を超えてスケールアウトできますが、最大インスタンスの重みまでスケールアウトできま

す。その意図は、できるだけ新しい希望するキャパシティーに近づけることですが、それでもグループのために指定されている割り当て戦略を遵守することです。割り当て戦略によって、起動するインスタンスタイプを決定します。重みは、インスタンスタイプに基づいて、各インスタンスがグループの希望するキャパシティーに影響するキャパシティーユニット数を決定します。

- 例 3: Auto Scaling グループの最大キャパシティーは 12、現在のキャパシティーは 10、および 5 つのキャパシティーユニットを追加する動的スケーリングポリシーがあります。インスタンスタイプには、1、4、6 の 3 つの重みのいずれかが割り当てられます。スケーリングポリシーを呼び出すとき、Amazon EC2 Auto Scaling は配分戦略に基づいて、重みが 6 のインスタンスタイプを起動することを選択します。このスケールアウトイベントの結果は、希望するキャパシティーが 12 で、現在のキャパシティーが 16 のグループになります。

複数の動的スケーリングポリシー

ほとんどの場合、ターゲット追跡スケーリングポリシーは、自動的にスケールアウトとスケールインするように Auto Scaling グループからインスタンスをデタッチするには、以下の手順を使用します。 グループを設定するのに十分です。ターゲット追跡スケーリングポリシーでは、希望する結果を選択したら、その結果を達成するために、必要に応じて Auto Scaling グループに対してインスタンスを追加および削除できます。

高度なスケーリング設定では、Auto Scaling グループに複数のスケーリングポリシーを設定できます。例えば、1 つ以上のターゲット追跡スケーリングポリシー、1 つ以上のステップスケーリングポリシー、またはその両方を定義できます。これにより、複数のシナリオに対応できる柔軟性が高まります。

複数の動的スケーリングポリシーがどのように連携するかを説明するために、Auto Scaling グループと Amazon SQS キューを使用して 1 つの EC2 インスタンスにリクエストを送信するアプリケーションを考えてみます。アプリケーションのパフォーマンスを最適なレベルに維持するために、Auto Scaling グループをスケールアウトするタイミングをコントロールする 2 つのポリシーがあるとします。1 つは、キュー内の SQS メッセージの数に基づいてキャパシティーを追加および削除するカスタムメトリクスを使用する、ターゲット追跡ポリシーです。もう 1 つのステップスケーリングポリシーは、Amazon CloudWatch `CPUUtilization` メトリクスを使用して、インスタンスが指定された期間に 90% の使用率を超えたときに容量を追加します。

同時に有効なポリシーが複数ある場合は、各ポリシーで、Auto Scaling グループが同時にスケールアウト (またはスケールイン) するように指定している可能性があります。たとえば、SQS カスタム`CPUUtilization`メトリクスがカスタムメトリクス CloudWatch アラームのしきい値をスパイク

および超過すると同時に、メトリクスがアラームのしきい値をスパイクおよび超過する可能性があります。

このような状況になると、Amazon EC2 Auto Scaling はスケールアウトとスケールインの両方に最大のキャパシティーを提供するポリシーを選択します。例えば、CPUUtilization のポリシーが 1 つのインスタンスを起動する一方で、SQS キューのポリシーは 2 つのインスタンスを起動するとします。両方のポリシーのスケールアウト基準が同時に満たされた場合、Amazon EC2 Auto Scaling は SQS キューポリシーを優先します。これにより、Auto Scaling グループは 2 つのインスタンスを起動することになります。

ポリシーがスケールインに異なる基準を使用する場合でも、最大キャパシティーを提供するポリシーを優先するというアプローチが適用されます。例えば、3 つのインスタンスを終了するポリシーと、インスタンス数を 25% 減らすポリシーがあり、スケールイン時にグループに 8 つのインスタンスがあるとします。この場合、Amazon EC2 Auto Scaling は、グループに最大数のインスタンスを提供するポリシーを優先させます。その結果、Auto Scaling グループは 2 つのインスタンスを終了します (8 の 25% = 2)。その目的は、Amazon EC2 Auto Scaling がインスタンスを削除しすぎないようにすることです。

ただし、ターゲット追跡スケーリングポリシーをステップスケーリングポリシーとともに使用する場合、ポリシー間の競合によって望ましくない動作が生じる可能性があるため、注意することをお勧めします。例えば、ターゲット追跡ポリシーがスケールインする準備が整う前に、ステップスケーリングポリシーがスケールインアクティビティを開始した場合、スケールインアクティビティはブロックされません。スケールインアクティビティが完了した後で、ターゲット追跡ポリシーにより、グループに再びスケールアウトするよう指示できます。

Amazon EC2 Auto Scaling のターゲットトラッキングスケーリングポリシー

ターゲット追跡スケーリングポリシーは、ターゲットメトリクス値に基づいて Auto Scaling グループの容量を自動的にスケーリングします。これにより、手動で操作しなくても、アプリケーションは最適なパフォーマンスとコスト効率を維持できます。

ターゲット追跡を使用することで、アプリケーションの理想的な平均使用率またはスループットレベルを表すメトリクスとターゲット値を選択します。Amazon EC2 Auto Scaling は、メトリクスがターゲットから逸脱したときにスケーリングイベントを呼び出す CloudWatch アラームを作成および管理します。例として、これはサーモスタットが目標温度を維持する方法と似ています。

例えば、現在 2 つのインスタンスで実行されているアプリケーションがあり、アプリケーションの負荷が変化しても Auto Scaling グループの CPU 使用率を約 50% に維持する必要があるとします。

これにより、過剰な数のアイドルリソースを維持することなくトラフィックのスパイクを処理するための追加のキャパシティが得られます。

このニーズを満たすには、50% の平均 CPU 使用率をターゲットとする、ターゲット追跡スケーリングポリシーを作成します。その後、CPU が 50% を超えると、Auto Scaling グループはスケールアウトするか、容量を増やして負荷の増加に対応します。CPU が 50% を下回ると、使用率の低い時間帯にコストを最適化するために、容量をスケールインまたは縮小します。

トピック

- 複数のターゲット追跡スケーリングポリシー
- メトリクスを選択する
- ターゲット値の定義
- インスタンスのウォームアップ時間を定義する
- 考慮事項
- ターゲット追跡スケーリングポリシーを作成する
- Metric Math を使用する、Amazon EC2 Auto Scaling のターゲット追跡スケーリングポリシーを作成する

複数のターゲット追跡スケーリングポリシー

スケーリングパフォーマンスを最適化するために複数のターゲット追跡スケーリングポリシーを同時に使用できますが、それぞれが異なるメトリクスを使用する必要があります。例えば、使用率とスループットは互いに影響し合う可能性があります。これは、これらのメトリクスのいずれかが変更されるたびに、通常、他のメトリクスも影響を受けることを意味します。したがって、複数のメトリクスを使用すると、Auto Scaling グループが属する負荷に関する追加情報が提供されます。これにより、Amazon EC2 Auto Scaling は、グループに追加する容量を決定する際に、より多くの情報に基づいた意思決定を行うことができます。

Amazon EC2 Auto Scaling の目的は、常に可用性を優先することです。ターゲット追跡ポリシーのいずれかがスケールアウトする準備ができている場合、Auto Scaling グループがスケールアウトされます。すべてのターゲット追跡ポリシー (スケールイン部分が有効) がスケールインできる場合にのみスケールインされます。

メトリクスを選択する

事前定義されたメトリクスまたはカスタムメトリクスのいずれかを使用して、ターゲット追跡スケーリングポリシーを作成できます。

事前定義されたメトリクスタイプでターゲット追跡スケーリングポリシーを作成する場合、次の事前定義済みメトリクスのリストからメトリクスを選択します。

- `ASGAverageCPUUtilization` - Auto Scaling グループの平均 CPU 使用率。
- `ASGAverageNetworkIn` - すべてのネットワークインターフェイスで単一のインスタンスが受信した平均バイト数。
- `ASGAverageNetworkOut` - すべてのネットワークインターフェイスで単一のインスタンスから送信された平均バイト数。
- `ALBRequestCountPerTarget` - ターゲットあたりの Application Load Balancer の平均リクエスト数。

> **Important**
>
> ターゲットごとの CPU 使用率、ネットワーク I/O、Application Load Balancer リクエスト数のメトリクスに関するその他の貴重な情報は、Amazon EC2 ユーザーガイドのインスタンスで利用可能な CloudWatch メトリクスのリストトピックと、CloudWatch Application Load Balancer ユーザーガイドの Application Load Balancer トピックのメトリクスにそれぞれ記載されています。

カスタム CloudWatch メトリクスを指定 CloudWatch することで、 で使用可能な他のメトリクスまたは独自のメトリクスを選択できます。カスタマイズされたメトリクス仕様でターゲット追跡ポリシーを作成するには、 AWS CLI または SDK を使用する必要があります。を使用してターゲット追跡スケーリングポリシーのカスタマイズされたメトリクス仕様を指定する例については AWS CLI、「」を参照してくださいAWS Command Line Interface (AWS CLI) のスケーリングポリシーの例。

メトリクスを選択するときは、以下の点に注意してください。

- 使用状況の変化に応じてより迅速にスケールできるように、1 分間隔で利用可能なメトリクスのみを使用することをお勧めします。ターゲット追跡では、すべての事前定義済みメトリクスとカスタムメトリクスについて 1 分単位で集計されたメトリクスが評価されますが、基礎となるメトリクスによっては、データが公開される頻度は低くなる可能性があります。たとえば、Amazon EC2 メトリクスはすべてデフォルトで 5 分間隔で送信されますが、1 分に設定できます (詳細モニタリングと呼ばれます)。この選択は個々のサービス次第です。ほとんどのサービスは、可能な限り短い間隔を使用しようとします。詳細モニタリングを有効にする方法については、「Auto Scaling インスタンスのモニタリングを設定する」を参照してください。

- カスタムメトリクスにはターゲット追跡に使用できないものもあります。メトリクスは、有効な使用率メトリクスで、インスタンスの使用頻度を示す必要があります。メトリクス値は Auto Scaling グループのインスタンス数に比例して増減する必要があります。それにより、メトリクスデータを使用して比例的にインスタンス数をスケールアウトまたはスケールインできます。例えば、`CPUUtilization` Auto Scaling グループの負荷がインスタンス間で分散されている場合、Auto Scaling グループの CPU 使用率 (メトリクスディメンション `AutoScalingGroupName` を持つ Amazon EC2 メトリクス) は有効です。
- 以下のメトリクスはターゲットの追跡では機能しません。
  - Auto Scaling グループの前にあるロードバランサーが受信したリクエスト数 (つまり、Elastic Load Balancing メトリクス`RequestCount`)。ロードバランサーによって受信されたリクエストの数は、Auto Scaling グループの使用率に基づいて変化しません。
  - ロードバランサーのリクエストのレイテンシー (Elastic Load Balancing メトリクス`Latency`)。リクエストのレイテンシーは、使用率の増加により増える場合がありますが、必ずしも比例して変化するわけではありません。
  - CloudWatch Amazon SQS キューメトリクス `ApproximateNumberOfMessagesVisible`。キュー内のメッセージ数は、キューからのメッセージを処理する Auto Scaling グループのサイズに比例して変わらない可能性があるためです。ただし、Auto Scaling グループの EC2 インスタンスごとにキュー内のメッセージの数を測定するカスタムメトリクスは機能します。詳しくは、「Amazon SQS に基づくスケーリング」を参照してください。
- `ALBRequestCountPerTarget` メトリクスを使用するには、`ResourceLabel` パラメータを指定して、メトリクスに関連付けられているロードバランサーターゲットグループを識別する必要があります。を使用してターゲット追跡スケーリングポリシーの `ResourceLabel`パラメータを指定する例については AWS CLI、「」を参照してくださいAWS Command Line Interface (AWS CLI) のスケーリングポリシーの例。
- メトリクスが実際の 0 の値を に出力する場合 CloudWatch （ 例: `ALBRequestCountPerTarget` ）、Auto Scaling グループは、アプリケーションへのトラフィックが長期間続かない場合に 0 にスケールインできます。Auto Scaling グループにリクエストがルーティングされないときに Auto Scaling グループを 0 にスケールインするには、グループの最小キャパシティを 0 に設定する必要があります。
- スケーリングポリシーで使用する新しいメトリクスを公開する代わりに、メトリクス数式を使用して既存のメトリクスを組み合わせることができます。詳細については、「Metric Math を使用する、Amazon EC2 Auto Scaling のターゲット追跡スケーリングポリシーを作成する」を参照してください。

ターゲット値の定義

ターゲット追跡スケーリングポリシーを作成するときは、ターゲット値を指定する必要があります。ターゲット値は、Auto Scaling グループの最適な平均使用率またはスループットを表します。優れたコスト効率でリソースを使用するには、予期しないトラフィックの増加に対して適切なバッファを使用し、ターゲット値をできる限り高く設定します。アプリケーションが通常のトラフィックフローに対して最適にスケールアウトされる場合、実際のメトリクス値は、ターゲット値以下である必要があります。

スケーリングポリシーが Application Load Balancer のターゲットごとのリクエスト数、ネットワーク I/O、またはその他のカウントメトリクスなどのスループットに基づいている場合、ターゲット値は単一のインスタンスからの最適な 1 分間の平均スループットを表します。

インスタンスのウォームアップ時間を定義する

オプションで、新しく起動されたインスタンスのウォームアップ時間を秒単位で指定できます。指定されたウォームアップ時間が終了するまで、インスタンスは Auto Scaling グループの集約された EC2 インスタンスメトリクスにカウントされません。

インスタンスがウォームアップ期間中、スケーリングポリシーは、ウォームアップしていないインスタンスのメトリクス値がポリシーのターゲット使用率よりも大きい場合にのみスケールアウトします。

グループが再度スケールアウトする場合、まだウォームアップ中のインスタンスは、次のスケールアウトアクティビティの希望キャパシティーの一部として計上されます。その目的は、スケールアウトを継続的に (ただし過剰になることなく) 行うことです。

スケールアウトアクティビティの進行中は、インスタンスがウォームアップを終了するまで、スケーリングポリシーによって開始されたすべてのスケールインアクティビティがブロックされます。インスタンスのウォームアップが完了し、スケールインイベントが発生した場合、現在終了中のインスタンスは、新しい必要キャパシティーを計算する際にグループの現在のキャパシティーにカウントされます。したがって、Auto Scaling グループから必要以上の数のインスタンスが削除されることがなくなります。

デフォルト値

値が設定されていない場合、スケーリングポリシーはデフォルト値を使用します。これは、グループに定義されたデフォルトのインスタンスウォームアップの値です。デフォルトのインスタンスウォームアップが null の場合、デフォルトのクールダウン の値にフォールバックします。ウォームアップ

時間が変化したときにすべてのスケーリングポリシーを簡単に更新できるように、デフォルトのインスタンスウォームアップを使用することをお勧めします。

考慮事項

ターゲット追跡スケーリングポリシーを使用する場合は、次の考慮事項が適用されます。

- ターゲット追跡スケーリングポリシーで使用される CloudWatch アラームを作成、編集、または削除しないでください。Amazon EC2 Auto Scaling は、 CloudWatchターゲット追跡スケーリングポリシーに関連付けられているアラームを作成および管理し、不要になったら削除します。
- ターゲット追跡スケーリングポリシーは、トラフィック減少時に徐々にスケールインすることにより、トラフィックレベルが変動する期間中の可用性を優先します。ワークロードの終了時に Auto Scaling グループを迅速にスケールインするには、ポリシーのスケールイン部分を無効にします。これにより、使用率が低い場合に、ニーズに最適なスケールイン方法を使用できる柔軟性を得られます。スケールインができる限り迅速に行われるようにするには、クールダウン期間が追加されないように、シンプルなスケーリングポリシーを使用しないことをお勧めします。
- メトリクスにデータポイントがない場合、 CloudWatch アラームの状態は に変わりますINSUFFICIENT_DATA。この状態になると、Amazon EC2 Auto Scaling は、新しいデータポイントが見つかるまでグループをスケールできなくなります。
- メトリクスが設計上まばらに報告される場合は、メトリクス数式が役立つことがあります。例えば、最新の値を使用するには、FILL(m1,REPEAT) という関数を使用します (m1 がメトリクスです)。
- ターゲット値と実際のメトリクスデータポイント間にギャップが発生する場合があります。これは、追加または削除するインスタンス数を決定するときに、その数を切り上げまたは切り捨てて常に控えめに動作するためです。これにより、不十分な数のインスタンスを追加したり、インスタンスを過剰に削除したりすることがなくなります。ただし、インスタンス数が少ない、より小さな Auto Scaling グループでは、グループの使用率はターゲット値からかなり離れているように見える場合があります。例えば、CPU 使用率に 50 パーセントのターゲット値を設定し、Auto Scaling グループがそのターゲットを超過した場合です。1.5 インスタンスを追加することで CPU 使用率が 50 パーセント近くに減少すると判断される場合があります。1.5 インスタンスを追加することはできないので、これを切り上げて、2 インスタンスを追加します。これにより、CPU 使用率は 50 パーセント未満の値に下がる可能性がありますが、アプリケーションをサポートする十分なリソースが確保されます。同様に、1.5 インスタンスを削除することで CPU 使用率が 50 パーセントを超えると判断した場合、1 インスタンスのみを削除します。

より多くのインスタンスのある大規模な Auto Scaling グループの場合、使用率はより多くのインスタンス間で分散されます。その場合、インスタンスの追加または削除により、ターゲット値と実際のメトリクスデータポイントとの差が小さくなります。

- ターゲットの追跡スケーリングポリシーでは、指定されたメトリクスがターゲット値を超えている場合、Auto Scaling グループをスケールアウトする必要があるとみなされます。指定されたメトリクスがターゲット値を下回っている場合、ターゲット追跡スケーリングポリシーを使用して Auto Scaling グループをスケールアウトすることはできません。

ターゲット追跡スケーリングポリシーを作成する

Auto Scaling グループのターゲット追跡スケーリングポリシーを作成するには、次のいずれかの方法を使用します。

開始する前に、(Amazon EC2 メトリクスのデフォルトが 5 分間隔であることを踏まえて) 希望するメトリクスが 1 分間隔で利用可能であることを確認してください。

Console

新しい Auto Scaling グループのターゲット追跡スケーリングポリシーを作成するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. [Auto Scaling グループの作成] を選択します。
3. ステップ 1、2、および 3 で、必要に応じてオプションを選択し、「ステップ 4: グループサイズとスケーリングポリシーを設定する」に進みます。
4. スケーリング で、希望する最小容量 と希望する最大容量 を更新して、スケーリングする範囲を指定します。これら 2 つの設定により、Auto Scaling グループは動的にスケーリングできます。詳細については、「Auto Scaling グループのスケーリング制限を設定する」を参照してください。
5. 自動スケーリング で、ターゲット追跡スケーリングポリシー を選択します。
6. ポリシーを定義するには、以下の作業を行います。

   a. ポリシーの名前を指定します。

   b. [Metric type (メトリクスタイプ)] でメトリクスを選択します。

[Application Load Balancer request count per target (ターゲットあたりのApplication Load Balancer リクエスト数)] を選択し、[Target group (ターゲットグループ)] のターゲットグループを選択します。

c. メトリクスの [Target value] を指定します。

d. (オプション) インスタンスのウォームアップ で、必要に応じてインスタンスのウォームアップ値を更新します。

e. (オプション) [Disable scale in to create only a scale-out policy (スケールインを無効にしてスケールアウトポリシーのみを作成する)] を選択します。これにより、必要に応じて異なるタイプの別のスケールインポリシーを作成できます。

7. Auto Scaling グループの作成に進みます。スケーリングポリシーは、Auto Scaling グループの作成後に作成されます。

既存の Auto Scaling グループにターゲット追跡スケーリングポリシーを作成するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。
3. スケーリング制限が適切に設定されていることを確認します。例えば、グループの希望する容量がすでに最大値に達している場合は、スケールアウトするために新しい最大値を指定する必要があります。詳細については、「Auto Scaling グループのスケーリング制限を設定する」を参照してください。
4. [Automatic scaling] (オートスケーリング) タブの [Dynamic scaling policies] (動的スケーリングポリシー) で、[Create dynamic scaling] (動的スケーリングポリシーの作成) を選択します。
5. ポリシーを定義するには、以下の作業を行います。

   a. [ポリシータイプ] で、[ターゲット追跡スケーリング] のデフォルト設定を維持します。

   b. ポリシーの名前を指定します。

   c. [Metric type (メトリクスタイプ)] でメトリクスを選択します。選択できるメトリクスタイプは 1 つだけです。複数のメトリクスを使用するには、複数のポリシーを作成します。

[Application Load Balancer request count per target (ターゲットあたりのApplication Load Balancer リクエスト数)] を選択し、[Target group (ターゲットグループ)] のターゲットグループを選択します。

d. メトリクスの [Target value] を指定します。

e. (オプション) インスタンスのウォームアップ で、必要に応じてインスタンスのウォームアップ値を更新します。

f. (オプション) [Disable scale in to create only a scale-out policy (スケールインを無効にしてスケールアウトポリシーのみを作成する)] を選択します。これにより、必要に応じて異なるタイプの別のスケールインポリシーを作成できます。

6. [作成] を選択します。

AWS CLI

ターゲット追跡スケーリングポリシーを作成するには、次の例を使用して開始できます。各##########を独自の情報に置き換えます。

> **Note**
>
> その他の例については、「AWS Command Line Interface (AWS CLI) のスケーリングポリシーの例」を参照してください。

ターゲット追跡スケーリングポリシーを作成するには (AWS CLI)

1. 次のcatコマンドを使用して、スケーリングポリシーのターゲット値と事前定義されたメトリクス仕様をホームディレクトリconfig.jsonの という名前の JSON ファイルに保存します。以下は、平均 CPU 使用率を 50% に維持するターゲット追跡設定の例です。

```
$ cat ~/config.json
{
  "TargetValue": 50.0,
  "PredefinedMetricSpecification":
    {
      "PredefinedMetricType": "ASGAverageCPUUtilization"
    }
}
```

詳細については、Amazon EC2 Auto Scaling API リファレンス」のPredefinedMetric「仕様」を参照してください。

2. ut-scaling-policy コマンドを使用して、前のステップで作成した config.json と共に、スケーリングポリシーを作成します。

```
aws autoscaling put-scaling-policy --policy-name cpu50-target-tracking-scaling-policy \
  --auto-scaling-group-name my-asg --policy-type TargetTrackingScaling \
  --target-tracking-configuration file://config.json
```

成功すると、このコマンドはユーザーに代わって作成された 2 つの CloudWatch アラームの ARNs と名前を返します。

```
{
    "PolicyARN": "arn:aws:autoscaling:us-
west-2:123456789012:scalingPolicy:228f02c2-c665-4bfd-
aaac-8b04080bea3c:autoScalingGroupName/my-asg:policyName/cpu50-target-tracking-
scaling-policy",
    "Alarms": [
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmHigh-
fc0e4183-23ac-497e-9992-691c9980c38e",
            "AlarmName": "TargetTracking-my-asg-AlarmHigh-
fc0e4183-23ac-497e-9992-691c9980c38e"
        },
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmLow-61a39305-ed0c-47af-
bd9e-471a352ee1a2",
            "AlarmName": "TargetTracking-my-asg-AlarmLow-61a39305-ed0c-47af-
bd9e-471a352ee1a2"
        }
    ]
}
```

Metric Math を使用する、Amazon EC2 Auto Scaling のターゲット追跡スケーリングポリシーを作成する

Metric Math を使用すると、複数の CloudWatch メトリクスをクエリし、数式を使用して、これらのメトリクスに基づいて新しい時系列を作成できます。作成された時系列を CloudWatch コンソールで視覚化し、ダッシュボードに追加できます。Metric Math の詳細については、「Amazon ユーザーガイド」の「Metric Math の使用」を参照してください。 CloudWatch

Metric Math の数式には、次の考慮事項が適用されます。

- 使用可能なメトリクスをクエリできます CloudWatch 。各メトリクスは、メトリクス名、名前空間、0 以上のディメンションの一意の組み合わせです。
- 算術演算子 (+ - * / ^)、統計関数 (AVG や SUM など) 、または が CloudWatch サポートするその他の関数を使用できます。
- 数式の関係式では、メトリクスと他の数式の結果の両方を使用できます。
- メトリクスの指定で使用される数式はすべて、最終的に単一の時系列を返す必要があります。
- CloudWatch コンソールまたは CloudWatch GetMetricData API を使用して、メトリクスの数式が有効であることを確認できます。

> **Note**
>
> 、、または SDK を使用する場合にのみ AWS CLI AWS CloudFormation、Metric Math を使用してターゲット追跡スケーリングポリシーを作成できます。この機能はコンソールではまだ使用できません。

例: インスタンスごとの Amazon SQS キューバックログ

インスタンスごとの Amazon SQS キューバックログを計算するには、キューからの取得に使用できるメッセージの概数を取得し、その数を Auto Scaling グループの実行キャパシティーで除算します。つまり、`InService` 状態のインスタンスの数になります。詳細については、「Amazon SQS に基づくスケーリング」を参照してください。

この数式の口ジックは次のとおりです。

```
sum of (number of messages in the queue)/(number of InService instances)
```

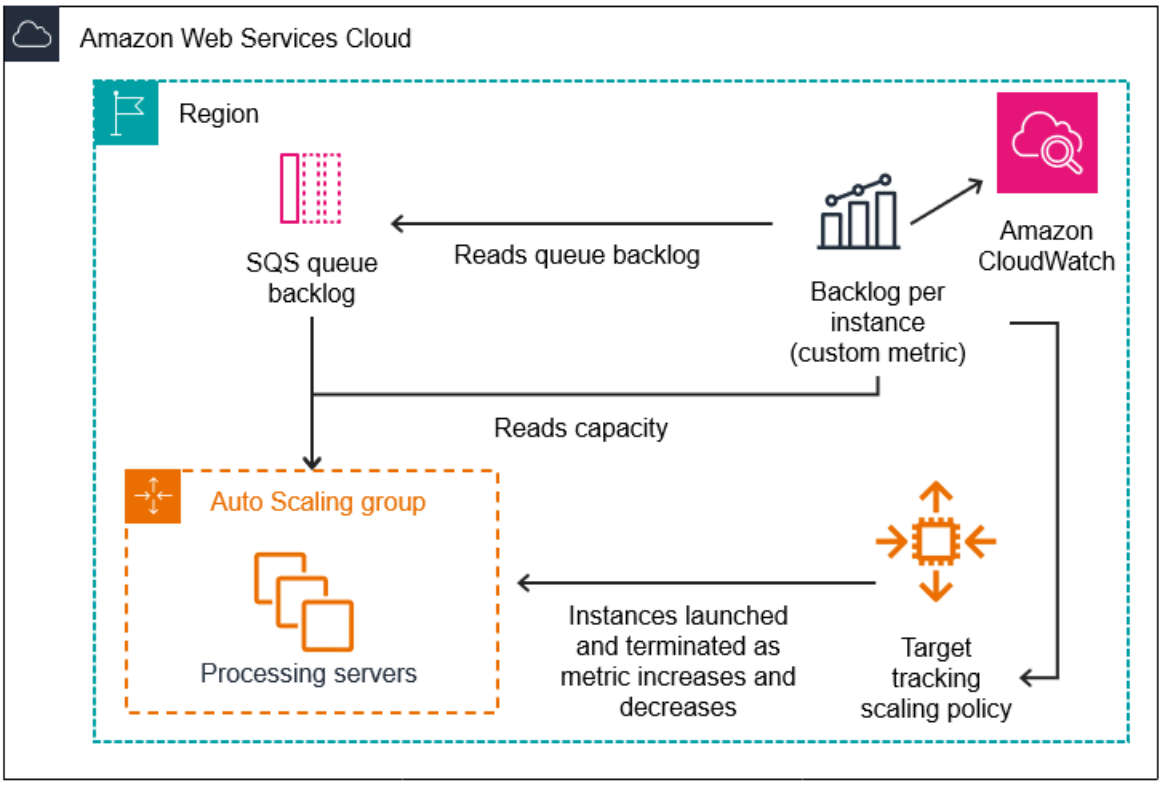
次に、 CloudWatch メトリクス情報は次のとおりです。

ID	CloudWatch メトリクス	統計	間隔
m1	ApproximateNumberOfMessages表示可能	合計	1 分
m2	GroupInServiceInstances	[Average] (平均)	1 分

メトリクス数学 ID と表現は次のとおりです。

ID	表現
e1	`(m1)/(m2)`

次の図は、このメトリクスのアーキテクチャを示しています。



この Metric Math を使用してターゲット追跡スケーリングポリシーを作成するには (AWS CLI)

1. Metric Math の数式を、カスタマイズされたメトリクス仕様の一部として、config.json という名前の JSON ファイルに保存します。

   次の例を参考にして開始してください。各##############を独自の情報に置き換えます。

```
{
    "CustomizedMetricSpecification": {
        "Metrics": [
            {
                "Label": "Get the queue size (the number of messages waiting to be
 processed)",
                "Id": "m1",
                "MetricStat": {
                    "Metric": {
                        "MetricName": "ApproximateNumberOfMessagesVisible",
                        "Namespace": "AWS/SQS",
                        "Dimensions": [
                            {
                                "Name": "QueueName",
                                "Value": "my-queue"
                            }
                        ]
                    },
                    "Stat": "Sum"
                },
                "ReturnData": false
            },
            {
                "Label": "Get the group size (the number of InService instances)",
                "Id": "m2",
                "MetricStat": {
                    "Metric": {
                        "MetricName": "GroupInServiceInstances",
                        "Namespace": "AWS/AutoScaling",
                        "Dimensions": [
                            {
                                "Name": "AutoScalingGroupName",
                                "Value": "my-asg"
                            }
                        ]
                    },
```

```
                    "Stat": "Average"
                },
                "ReturnData": false
            },
            {
                "Label": "Calculate the backlog per instance",
                "Id": "e1",
                "Expression": "m1 / m2",
                "ReturnData": true
            }
        ]
    },
    "TargetValue": 100
}
```

詳細については、Amazon EC2 Auto Scaling API リファレンス」のTargetTracking「設定」を参照してください。

> **Note**
>
> 以下は、メトリクスの名前、名前空間、ディメンション、統計情報の検索に役立つ追加のリソースです CloudWatch。
>
> - サービスで使用可能なメトリクスの詳細については、「Amazon ユーザーガイド AWS 」の「メトリクスを発行する のサービス」を参照してください。 AWS CloudWatch CloudWatch
> - を使用してメトリクスの正確な CloudWatch メトリクス名、名前空間、ディメンション (該当する場合) を取得するには AWS CLI、「list-metrics」を参照してください。

2. このポリシーを作成するには、以下の例にあるように、JSON ファイルを入力として使用して put-scaling-policy コマンドを実行します。

```
aws autoscaling put-scaling-policy --policy-name sqs-backlog-target-tracking-scaling-policy \
  --auto-scaling-group-name my-asg --policy-type TargetTrackingScaling \
  --target-tracking-configuration file://config.json
```

成功すると、このコマンドはポリシーの Amazon リソースネーム (ARN) と、ユーザーに代わって作成された 2 ARNs を返します。 CloudWatch

```
{
    "PolicyARN": "arn:aws:autoscaling:us-
west-2:123456789012:scalingPolicy:228f02c2-c665-4bfd-
aaac-8b04080bea3c:autoScalingGroupName/my-asg:policyName/sqs-backlog-target-
tracking-scaling-policy",
    "Alarms": [
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmHigh-
fc0e4183-23ac-497e-9992-691c9980c38e",
            "AlarmName": "TargetTracking-my-asg-AlarmHigh-
fc0e4183-23ac-497e-9992-691c9980c38e"
        },
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmLow-61a39305-ed0c-47af-
bd9e-471a352ee1a2",
            "AlarmName": "TargetTracking-my-asg-AlarmLow-61a39305-ed0c-47af-
bd9e-471a352ee1a2"
        }
    ]
}
```

> **Note**
>
> このコマンドがエラーをスローする場合は、 を AWS CLI ローカルで最新バージョンに更新していることを確認してください。

Amazon EC2 Auto Scaling のステップおよび簡易スケーリングポリシー

ステップスケーリングポリシーと簡易スケーリングポリシーは、 CloudWatch アラームに基づいて Auto Scaling グループの容量を事前に定義された増分でスケーリングします。アラームのしきい値を超えると、スケールアウト (容量の増加) とスケールイン (キャパシティの減少) を処理するスケーリングポリシーを個別に定義できます。

ステップスケーリングと簡易スケーリングでは、スケーリングプロセスを呼び出す CloudWatch アラームを作成および管理します。アラームに違反すると、Amazon EC2 Auto Scaling はそのアラームに関連付けられたスケーリングポリシーを開始します。

ターゲット追跡スケーリング ポリシーを使用して、ターゲットごとの平均 CPU 使用率や平均リクエスト数などのメトリクスに基づいてスケールすることを強くお勧めします。キャパシティーが増加すると減少し、キャパシティーが減少すると増加するメトリクスを使用すると、ターゲット追跡を使用してインスタンス数を比例的にスケールアウトしたり、インスタンス数を増やすことができます。これにより、Amazon EC2 Auto Scaling がアプリケーションの需要曲線に厳密に従うことが保証されます。詳細については、「ターゲット追跡スケーリングポリシー」を参照してください。

内容

- ステップスケーリングポリシーのしくみ
- ステップスケーリングのステップ調整
- スケーリング調整タイプ
- インスタンスのウォームアップ
- 考慮事項
- スケールアウト用のステップスケーリングポリシーを作成する
- スケールイン用のステップスケーリングポリシーを作成する
- 簡易スケーリングポリシー

ステップスケーリングポリシーのしくみ

ステップスケーリングを使用するには、まず Auto Scaling グループのメトリクスをモニタリングする CloudWatch アラームを作成します。アラーム違反を判断するメトリクス、しきい値、評価期間の数を定義します。次に、アラームのしきい値を超えたときにグループをスケーリングする方法を定義するステップスケーリングポリシーを作成します。

ポリシーにステップ調整値を追加します。アラームの違反規模に基づいて、さまざまなステップ調整値を定義できます。例:

- アラームメトリクスが 60% に達した場合、10 インスタンスずつスケールアウトする
- アラームメトリクスが 75% に達した場合、30 インスタンスずつスケールアウトする
- アラームメトリクスが 85% に達した場合、40 インスタンスずつスケールアウトする

アラームのしきい値が指定された評価期間数を超えると、Amazon EC2 Auto Scaling はポリシーで定義されたステップ調整を適用します。アラームの状態が OK に戻るまで、さらなるアラーム違反が発生した場合に備えて、調整を続けることができます。

各インスタンスにはウォームアップ期間があり、短期間に発生した変更に対してスケーリングアクティビティが過度に反応しすぎないようにします。オプションで、スケーリングポリシーのウォームアップ期間を設定できます。ただし、ウォームアップ時間が変化したときにすべてのスケーリングポリシーを簡単に更新できるように、デフォルトのインスタンスウォームアップを使用することをお勧めします。詳細については、「Auto Scaling グループに対するインスタンスのデフォルトウォームアップを設定する」を参照してください。

シンプルなスケーリングポリシーは、ステップスケーリングポリシーに似ていますが、1 つのスケーリング調整に基づいており、各スケーリングアクティビティ間にクールダウン期間がある点が異なります。詳細については、「簡易スケーリングポリシー」を参照してください。

ステップスケーリングのステップ調整

ステップスケーリングポリシーを作成するときは、アラーム超過のサイズに基づいてインスタンス数を動的にスケーリングする 1 つ以上のステップ調整値を指定します。各ステップ調整値は、次のように指定します。

- メトリクス値の下限
- メトリクス値の上限
- スケーリング調整タイプに基づいてスケールする量

CloudWatch は、 CloudWatch アラームに関連付けられているメトリクスの統計に基づいてメトリクスデータポイントを集計します。アラームに違反すると、適切なスケーリングポリシーが呼び出されます。Amazon EC2 Auto Scaling は、 CloudWatch (raw メトリクスデータではなく) からの最新のメトリクスデータポイントに集約タイプを適用します。ステップ調整によって定義された上限と下限に対して、この集約メトリクス値を比較することにより、実行するステップ調整が決定されます。

違反しきい値に比例して上限と下限を指定します。例えば、メトリクスが 50% を超えた場合に CloudWatch アラームとスケールアウトポリシーを作成したとします。次に、メトリクスが 50% を下回ったときの 2 つ目のアラームとスケールインポリシーを作成しました。各ポリシーの調整タイプ `PercentChangeInCapacity` (またはコンソールのグループの割合) を使用して、一連のステップ調整を行いました。

例: スケールアウトポリシーのステップ調整値

下限	上限	調整
0	10	0

下限	上限	調整
10	20	10
20	null	30

例: スケールインポリシーのステップ調整値

下限	上限	調整
-10	0	0
-20	-10	-10
null	-20	-30

これにより、次のスケーリング設定が作成されます。

```
Metric value

-infinity          30%    40%          60%     70%             infinity
-------------------------------------------------------------------------
          -30%      | -10% | Unchanged  | +10%  |       +30%
-------------------------------------------------------------------------
```

次に、現在の容量と希望する容量の両方が 10 の Auto Scaling グループでこのスケーリング設定を使用するとします。以下の点は、グループの希望キャパシティーと現在のキャパシティーに関連してスケーリング設定の動作をまとめたものです。

- 集合メトリクス値が 40 より大きく 60 未満である間は、現在のキャパシティーと必要なキャパシティーが維持されます。
- メトリクス値が 60 になった場合、スケールアウトのポリシーの 2 番目のステップ調整値 (10 インスタンスの 10% を追加) に基づいて、グループの必要なキャパシティーを 1 インスタンスだけ増やして、11 インスタンスにします。新しいインスタンスが実行され、指定されたウォームアップ時間が終了すると、グループの現在の容量は 11 インスタンスに増加します。このキャパシティーの増加後も、メトリクス値が 70 に上昇すると、グループの希望するキャパシティーはさらに 3 インスタンス増加し、14 インスタンスになります。これは、スケールアウトポリシーの 3 番目のス

テップ調整に基づきます (11 インスタンスの 30% である 3.3 インスタンスを 3 インスタンスに切り捨て、追加します)。

- メトリクス値が 40 になった場合、スケールインポリシーの 2 番目のステップ調整値 (14 インスタンスの 10%、つまり 1.4 インスタンスを丸めた 1 インスタンスを削除) に基づいて、グループの必要なキャパシティーを 1 インスタンスだけ減らして、13 インスタンスにします。このキャパシティーが減少した後も、メトリクス値が 30 に下がると、グループの希望するキャパシティーはさらに 3 インスタンス減少し、10 インスタンスになります。これは、スケールインポリシーの 3 番目のステップ調整に基づきます (13 インスタンスの 30% である 3.9 インスタンスを 3 インスタンスに切り捨て、減算します)。

スケーリングポリシーのステップ調整を指定するときは、次の点に注意してください。

- を使用する場合は AWS Management Console、上限と下限を絶対値として指定します。 AWS CLI または SDK を使用する場合は、違反しきい値に対する上限と下限を指定します。
- ステップ調整値の範囲に重複や間隔があってはなりません。
- 1 つのステップ調整値のみ、下限を null (負の無限大) にすることができます。下限が負のステップ調整値がある場合は、下限が null のステップ調整値が必要です。
- 1 つのステップ調整値のみ、上限を null (正の無限大) にすることができます。上限が正のステップ調整値がある場合は、上限が null のステップ調整値が必要です。
- 同じステップ調整値で上限と下限を null にすることはできません。
- メトリクス値が超過しきい値を上回っている場合、下限にその値を含み、上限には含みません。メトリクス値が超過しきい値を下回っている場合、下限にその値を含まず、上限に含みます。

スケーリング調整タイプ

選択したスケーリング調整タイプに基づいて、最適なスケーリングアクションを実行するスケーリングポリシーを定義できます。調整タイプは、Auto Scaling グループの現在のキャパシティーに対する割合、またはキャパシティーユニットで指定できます。通常、キャパシティユニットとは、インスタンスの重み付け機能を使用していない限り、1 つのインスタンスを意味します。

Amazon EC2 Auto Scaling は、ステップスケーリングと簡易スケーリングに対して次の調整タイプをサポートしています。

- `ChangeInCapacity` - 指定した値だけ現在のキャパシティーを増減させます。正の調整値は現在のキャパシティーを増やし、負の調整値は現在のキャパシティーを減らします。例えば、グループの現在のキャパシティーが 3 で、調整値が 5 の場合、このポリシーが実行されると、キャパシ

ティーユニットが 5 個キャパシティーに追加され、合計 8 個のキャパシティーユニットになります。

- `ExactCapacity` - グループの現在のキャパシティーを指定された値に変更します。この調整タイプには負ではない値を指定します。例えば、グループの現在のキャパシティーが 3 で、調整値が 5 の場合、このポリシーが実行されると、キャパシティーは 5 キャパシティーユニットに変更されます。
- `PercentChangeInCapacity` - 指定したパーセンテージだけ現在のキャパシティーを増減させます。正の値はキャパシティーを増やし、負の値はキャパシティーを減らします。例えば、現在のキャパシティーが 10 で、調整値が 10% の場合、このポリシーが実行されると、キャパシティーユニットが 1 個追加され、合計 11 個のキャパシティーユニットになります。

> **Note**
>
> 結果の値が整数でない場合、その値を以下のように四捨五入します。
>
> - 1 より大きい値は小数点以下が切り捨てられます。例えば、`12.7` は `12` に丸められます。
> - 0 と 1 の間の値は 1 に丸められます。例えば、`.67` は `1` に丸められます。
> - 0 と -1 の間の値は -1 に丸められます。例えば、`-.58` は `-1` に丸められます。
> - -1 未満の値は小数点以下が切り捨てられます。例えば、`-6.67` は `-6` に丸められます。

`PercentChangeInCapacity` では、`MinAdjustmentMagnitude` パラメータを使用して、スケールするインスタンスの最小数を指定することもできます。例えば、25 パーセント追加するポリシーを作成し、インスタンスの最小増減値を 2 に指定するとします。4 つのインスタンスを持つ Auto Scaling グループがあり、スケーリングポリシーを実行する場合、25 パーセントは 1 インスタンスです。ただし、最小増減値が 2 に指定されているため、2 つのインスタンスが追加されます。

インスタンスの重み を使用すると、 `MinAdjustmentMagnitude`パラメータをゼロ以外の値に設定する効果が変化します。値はキャパシティーユニットで指定します。スケールするインスタンスの最小数を設定するには、このパラメータを、最大インスタンスの重みと同じ以上の値に設定します。

インスタンスの重みを使用する場合は、Auto Scaling グループの現在の容量が必要に応じて希望する容量を超える可能性があることに注意してください。デクリメントする絶対数、またはデクリメントする割合が現在のキャパシティーと希望するキャパシティーの差よりも小さい場合、スケーリングアクションは実行されません。しきい値アラームに違反している場合にスケーリングポリシーの結果を確認する際は、このような動作を考慮に入れる必要があります。例えば、希望するキャパシティー

が 30 で、現在のキャパシティーが 32 であるとします。アラームに違反している場合、スケーリングポリシーで希望するキャパシティーを 1 ずつ減らすと、スケーリングアクションは実行されません。

インスタンスのウォームアップ

ステップスケーリングの場合、オプションで、新しく起動されたインスタンスのウォームアップ時間を秒単位で指定できます。指定されたウォームアップ時間が終了するまで、インスタンスは Auto Scaling グループの集約された EC2 インスタンスメトリクスにカウントされません。

インスタンスがウォームアップ期間中、スケーリングポリシーは、ウォームアップしていないインスタンスのメトリクス値がポリシーのアラーム上限しきい値より大きい場合にのみスケールアウトします。

グループが再度スケールアウトする場合、まだウォームアップ中のインスタンスは、次のスケールアウトアクティビティの希望キャパシティーの一部として計上されます。したがって、複数のアラーム超過によってもステップ調整値の範囲は変わらず、1 つのスケーリングアクティビティという結果になります。その目的は、スケールアウトを継続的に (ただし過剰になることなく) 行うことです。

例えば、2 つのステップからなるポリシーを作成するとします。最初のステップでは、メトリクスが 60 になったら 10% を追加し、2 番目のステップではメトリクスが 70 になったら 30% を追加します。Auto Scaling グループの目的キャパシティーは 10 で、現在のキャパシティは 10 です。集約されたメトリクス値が 60 未満の間は、目的キャパシティーと現在のキャパシティーは変わりません。メトリクスが 60 になり、1 つのインスタンス (10 インスタンスの 10 パーセント) が追加されたとします。その後、新しいインスタンスがまだウォームアップしている間に、メトリクスは 62 になります。スケーリングポリシーは、現在のキャパシティー (まだ 10) に基づいて新しく必要なキャパシティーを計算します。ただし、グループの必要キャパシティーはすでに 11 インスタンスに増えているため、スケーリングポリシーはこれ以上必要キャパシティーは増やしません。新しいインスタンスがまだウォームアップ中にメトリックスが 70 になった場合は、3 インスタンス (10 インスタンスの 30%) を追加する必要があります。ただし、グループの必要なキャパシティーはすでに 11 であるため、2 つのインスタンスのみを追加し、新しい必要なキャパシティーは 13 インスタンスになります。

スケールアウトアクティビティの進行中は、インスタンスがウォームアップを終了するまで、スケーリングポリシーによって開始されたすべてのスケールインアクティビティがブロックされます。インスタンスのウォームアップが完了し、スケールインイベントが発生した場合、現在終了中のインスタンスは、新しい必要キャパシティーを計算する際にグループの現在のキャパシティーにカウントされます。したがって、Auto Scaling グループから必要以上の数のインスタンスが削除されることがなくなります。例えば、インスタンスがすでに終了しているときに、必要なキャパシティーを 1 つデク

リメントした同じステップ調整の範囲内でアラームに違反した場合、スケーリングアクションは実行されません。

デフォルト値

値が設定されていない場合、スケーリングポリシーはデフォルト値を使用します。これは、グループに定義されたデフォルトのインスタンスウォームアップの値です。デフォルトのインスタンスウォームアップが null の場合、デフォルトのクールダウン の値にフォールバックします。

考慮事項

ステップおよびシンプルなスケーリングポリシーを使用する場合は、次の考慮事項が適用されます。

- ステップスケーリングを使用できるほど正確にアプリケーションのステップ調整を予測できるかどうかを検討してください。スケーリングメトリクスがスケーラブルターゲットの容量に比例して増減する場合は、代わりにターゲット追跡スケーリングポリシーを使用することをお勧めします。より高度な設定には、追加ポリシーとしてステップスケーリングを使用するオプションがあります。例えば、必要に応じて、使用率が一定のレベルに達したときにより積極的なレスポンスを設定できます。
- フラッピングを防ぐために、スケールアウトとスケールインのしきい値の間には適切なマージンを選択してください。フラッピングは、スケールインとスケールアウトの無限ループです。つまり、スケーリングアクションが実行されると、メトリクス値が変化して、逆方向に別のスケーリングアクションが開始されます。

スケールアウト用のステップスケーリングポリシーを作成する

Auto Scaling グループのスケールアウト用のステップスケーリングポリシーを作成するには、次のいずれかの方法を使用します。

Console

ステップ 1: メトリクス上限しきい値の CloudWatch アラームを作成する

1. https://console.aws.amazon.com/cloudwatch/ で CloudWatch コンソールを開きます。
2. 必要に応じて、リージョンを変更します。ナビゲーションバーから、Auto Scaling グループがあるリージョンを選択します。
3. ナビゲーションペインで、[Alarms]、[All alarms] (アラーム、すべてのアラーム) の順に選択してから、[Create alarm] (アラームの作成) を選択します。

4. [メトリクスの選択] を選択します。
5. [All metrics (すべてのメトリクス)] タブで、[EC2]、[By Auto Scaling Group (Auto Scaling グループ別)] を選択し、検索フィールドに Auto Scaling グループの名前を入力します。次に、`CPUUtilization` を選択し、[メトリクスの選択] を選択します。[Specify metric and conditions] ページに、メトリクスに関するグラフや他の情報が表示されます。
6. [期間] でアラームの評価期間 (1 分など) を選択します。アラームを評価する場合、各期間は 1 つのデータポイントに集約されます。

   > **Note**
   >
   > 期間が短いほど、作成されるアラームの感度が高くなります。

7. [条件] で、次の操作を行います。

   - [Threshold type] で [静的] を選択します。
   - Whenever **`CPUUtilization`**が の場合、メトリクスの値がしきい値以上になるようにアラームを超過するかどうかを指定します。次に、[than] (次よりも:) にアラーム違反のしきい値を入力します。

   > **Important**
   >
   > スケールアウトポリシー (メトリクス高) で使用するアラームについては、しきい値に対して [less than] (次未満:)、または [less than or equal to] (次以下:) を選択しないようにします。

8. [追加設定] で、次の操作を行います。

   - [Datapoints to alarm] (アラームへのデータポイント)に、メトリクス値がしきい値を満たす必要があるデータポイント数 (評価期間) を入力します。例えば、5 分の期間が 2 回連続すると、アラームの状態が呼び出されるまで 10 分かかります。
   - [Missing data treatment (欠落データの処理)] では、[Treat missing data as bad (breaching threshold) (欠落データを問題として処理する (しきい値を超過))] を選択します。詳細については、「Amazon ユーザーガイド」の CloudWatch 「アラームが欠落データを処理する方法の設定」を参照してください。 CloudWatch

9. [次へ] をクリックします。

   [Configure actions] (アクションの設定) ページが表示されます。

10. [通知] で、アラームが ALARM 状態、OK 状態、または INSUFFICIENT_DATA 状態のときに通知するための Amazon SNS トピックを選択します。

    同じアラーム状態または複数の異なるアラーム状態について複数の通知を送信するには、[Add notification (通知の追加)] を選択します。

    アラームの通知を送信しない場合は、 [削除] を選択します。
11. Configure actions (アクションの設定) ページの他のセクションは空にしたままにすることができます。他のセクションを空のままにすると、スケーリングポリシーへの関連付けなしでアラームが作成されます。その後、Amazon EC2 Auto Scaling コンソールから、アラームをスケーリングポリシーに関連付けることができます。
12. [次へ] をクリックします。
13. 名前 (Step-Scaling-AlarmHigh-AddCapacity など) を入力し、必要に応じてアラームの説明を入力して [次へ] を選択します。
14. [アラームを作成] を選択します。

CloudWatch アラームの作成後に中断した場所を続行するには、次の手順に従います。

ステップ 2: スケールアウト用のステップスケーリングポリシーを作成する

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

    ページの下部にスプリットペインが開きます。
3. スケーリング制限が適切に設定されていることを確認します。例えば、グループの希望する容量がすでに最大値に達している場合は、スケールアウトするために新しい最大値を指定する必要があります。詳細については、「Auto Scaling グループのスケーリング制限を設定する」を参照してください。
4. [Automatic scaling] (オートスケーリング) タブの [Dynamic scaling policies] (動的スケーリングポリシー) で、[Create dynamic scaling] (動的スケーリングポリシーの作成) を選択します。
5. ポリシータイプ で、ステップスケーリング を選択し、ポリシーの名前を指定します。
6. CloudWatch アラーム で、アラームを選択します。アラームをまだ作成していない場合は、CloudWatch アラームの作成を選択し、前の手順のステップ 4 からステップ 14 を完了してアラームを作成します。

7. [Take the action (このアクションを実行)] を使用して、このポリシーの実行時に行う現在のグループサイズの変更を指定します。特定の数のインスタンスまたは既存のグループサイズに対する割合を追加したり、グループを正確なサイズに設定したりできます。

   例えば、グループの容量を 30% 増やすスケールアウトポリシーを作成するには、 を選択しAdd、30次のフィールドに と入力してから、 を選択しますpercent of group。デフォルトでは、このステップ調整値の下限はアラームしきい値であり、上限は正 (+) の無限大です。
8. 別のステップを追加するには、[Add step (ステップの追加)] を選択し、スケールする量と、アラームしきい値に対するステップの下限と上限を定義します。
9. 最小数のインスタンスを設定してスケールするには、[Add capacity units in increments of at least (最小数の増分でキャパシティーユニットを追加する)] 1 [capacity units (キャパシティーユニット)] の数値フィールドをアップデートします。
10. (オプション) インスタンスのウォームアップ で、必要に応じてインスタンスのウォームアップ値を更新します。
11. [作成] を選択します。

AWS CLI

スケールアウト (容量を増やす) のステップスケーリングポリシーを作成するには、次のコマンド例を使用できます。各##############を独自の情報に置き換えます。

を使用するときは AWS CLI、まず、メトリクスの値が増えたときにスケールアウトする方法を Amazon EC2 Auto Scaling に指示するステップスケーリングポリシーを作成します。次に、監視するメトリクスを特定し、アラームのメトリクス上限しきい値やその他の詳細を定義し、アラームをスケーリングポリシーに関連付けることでアラームを作成します。

ステップ 1: スケールアウト用のポリシーを作成する

次の put-scaling-policy コマンドを使用して、 という名前のステップスケーリングポリシーを作成します。調整タイプPercentChangeInCapacityは でmy-step-scale-out-policy、次のステップの調整 (CloudWatch アラームしきい値を 60% と仮定) に基づいてグループの容量を増やします。

- メトリクスの値が 60% 以上、75% 未満の場合は、インスタンス数を 10% 増やします。
- メトリクスの値が 75% 以上、85% 未満の場合は、インスタンス数を 20% 増やします。
- メトリクスの値が 85% 以上の場合は、インスタンス数を 30% 増やします。

```
aws autoscaling put-scaling-policy \
  --auto-scaling-group-name my-asg  \
  --policy-name my-step-scale-out-policy \
  --policy-type StepScaling \
  --adjustment-type PercentChangeInCapacity \
  --metric-aggregation-type Average \
  --step-adjustments
 MetricIntervalLowerBound=0.0,MetricIntervalUpperBound=15.0,ScalingAdjustment=10 \

 MetricIntervalLowerBound=15.0,MetricIntervalUpperBound=25.0,ScalingAdjustment=20 \
                     MetricIntervalLowerBound=25.0,ScalingAdjustment=30 \
  --min-adjustment-magnitude 1
```

ポリシーの Amazon リソースネーム (ARN) を記録します。ポリシーの CloudWatch アラームを作成するには、これが必要です。

```
{
    "PolicyARN":
 "arn:aws:autoscaling:region:123456789012:scalingPolicy:4ee9e543-86b5-4121-b53b-
aa4c23b5bbcc:autoScalingGroupName/my-asg:policyName/my-step-scale-in-policy
}
```

ステップ 2: メトリクスの上限しきい値の CloudWatch アラームを作成する

次の CloudWatch put-metric-alarm コマンドを使用して、2 分間の連続した 2 つの評価期間の平均 CPU しきい値 60% に基づいて Auto Scaling グループのサイズを増やすアラームを作成します。独自のカスタムメトリクスを使用するには、名前を `--metric-name` で指定し、その名前空間を `--namespace` で指定します。

```
aws cloudwatch put-metric-alarm --alarm-name Step-Scaling-AlarmHigh-AddCapacity \
  --metric-name CPUUtilization --namespace AWS/EC2 --statistic Average \
  --period 120 --evaluation-periods 2 --threshold 60 \
  --comparison-operator GreaterThanOrEqualToThreshold \
  --dimensions "Name=AutoScalingGroupName,Value=my-asg" \
  --alarm-actions PolicyARN
```

スケールイン用のステップスケーリングポリシーを作成する

Auto Scaling グループのスケールイン用のステップスケーリングポリシーを作成するには、次のいずれかの方法を使用します。

Console

ステップ 1: メトリクスのしきい値が低い CloudWatch アラームを作成する

1. https://console.aws.amazon.com/cloudwatch/ で CloudWatch コンソールを開きます。
2. 必要に応じて、リージョンを変更します。ナビゲーションバーから、Auto Scaling グループがあるリージョンを選択します。
3. ナビゲーションペインで、[Alarms]、[All alarms] (アラーム、すべてのアラーム) の順に選択してから、[Create alarm] (アラームの作成) を選択します。
4. [メトリクスの選択] を選択します。
5. [All metrics (すべてのメトリクス)] タブで、[EC2]、[By Auto Scaling Group (Auto Scaling グループ別)] を選択し、検索フィールドに Auto Scaling グループの名前を入力します。次に、`CPUUtilization` を選択し、[メトリクスの選択] を選択します。[Specify metric and conditions] ページに、メトリクスに関するグラフや他の情報が表示されます。
6. [期間] でアラームの評価期間 (1 分など) を選択します。アラームを評価する場合、各期間は 1 つのデータポイントに集約されます。

   > **Note**
   >
   > 期間が短いほど、作成されるアラームの感度が高くなります。

7. [条件] で、次の操作を行います。

   - [Threshold type] で [静的] を選択します。
   - Whenever **`CPUUtilization`**が の場合、メトリクスの値をしきい値以下に設定するか、しきい値以下に設定するかを指定します。次に、[than] (次よりも:) にアラーム違反のしきい値を入力します。

   > **Important**
   >
   > スケールインポリシー (メトリクス低) で使用するアラームについては、しきい値に対して [greater than] (次より大きい:)、または [greater than or equal to] (次以上:) を選択しないようにします。

8. [追加設定] で、次の操作を行います。

- [Datapoints to alarm] (アラームへのデータポイント)に、メトリクス値がしきい値を満たす必要があるデータポイント数 (評価期間) を入力します。例えば、5 分の期間が 2 回連続すると、アラームの状態が呼び出されるまで 10 分かかります。
- [Missing data treatment (欠落データの処理)] では、[Treat missing data as bad (breaching threshold) (欠落データを問題として処理する (しきい値を超過))] を選択します。詳細については、「Amazon ユーザーガイド」の CloudWatch 「アラームが欠落データを処理する方法の設定」を参照してください。 CloudWatch

9. [次へ] をクリックします。

   [Configure actions] (アクションの設定) ページが表示されます。

10. [通知] で、アラームが ALARM 状態、OK 状態、または INSUFFICIENT_DATA 状態のときに通知するための Amazon SNS トピックを選択します。

    同じアラーム状態または複数の異なるアラーム状態について複数の通知を送信するには、[Add notification (通知の追加)] を選択します。

    アラームの通知を送信しない場合は、 [削除] を選択します。

11. Configure actions (アクションの設定) ページの他のセクションは空にしたままにすることができます。他のセクションを空のままにすると、スケーリングポリシーへの関連付けなしでアラームが作成されます。その後、Amazon EC2 Auto Scaling コンソールから、アラームをスケーリングポリシーに関連付けることができます。
12. [次へ] をクリックします。
13. 名前 (Step-Scaling-AlarmLow-RemoveCapacity など) を入力し、必要に応じてアラームの説明を入力して [次へ] を選択します。
14. [アラームを作成] を選択します。

CloudWatch アラームの作成後に中断した場所を続行するには、次の手順に従います。

ステップ 2: スケールイン用のステップスケーリングポリシーを作成する

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。

3. スケーリング制限が適切に設定されていることを確認します。例えば、グループの希望する容量がすでに最小である場合、スケールインするには新しい最小容量を指定する必要があります。詳細については、「Auto Scaling グループのスケーリング制限を設定する」を参照してください。
4. [Automatic scaling] (オートスケーリング) タブの [Dynamic scaling policies] (動的スケーリングポリシー) で、[Create dynamic scaling] (動的スケーリングポリシーの作成) を選択します。
5. ポリシータイプ で、ステップスケーリング を選択し、ポリシーの名前を指定します。
6. CloudWatch アラーム で、アラームを選択します。アラームをまだ作成していない場合は、CloudWatch アラームの作成を選択し、前の手順のステップ 4 からステップ 14 を完了してアラームを作成します。
7. [Take the action (このアクションを実行)] を使用して、このポリシーの実行時に行う現在のグループサイズの変更を指定します。特定の数のインスタンスまたは既存のグループサイズに対する割合を削除したり、グループを正確なサイズに設定したりできます。

   例えば、グループの容量を 2 つのインスタンスずつ減らすスケールインポリシーを作成するには、 を選択しRemove、2次のフィールドに と入力してから、 を選択しますcapacity units。デフォルトでは、このステップ調整値の上限はアラームしきい値であり、下限は負 (-) の無限大です。
8. 別のステップを追加するには、[Add step (ステップの追加)] を選択し、スケールする量と、アラームしきい値に対するステップの下限と上限を定義します。
9. [作成] を選択します。

AWS CLI

スケールイン (容量を減らす) 用のステップスケーリングポリシーを作成するには、次のコマンド例を使用できます。各##############を独自の情報に置き換えます。

を使用するときは AWS CLI、まず、メトリクスの値が減少した場合にスケールインする方法を Amazon EC2 Auto Scaling に指示するステップスケーリングポリシーを作成します。次に、監視するメトリクスを識別し、アラームのメトリクスのしきい値の下限やその他の詳細を定義し、アラームをスケーリングポリシーに関連付けることでアラームを作成します。

ステップ 1: スケールイン用のポリシーを作成する

次の put-scaling-policy コマンドを使用して、 という名前のステップスケーリングポリシーを作成します。このポリシーの調整タイプはmy-step-scale-in-policy、関連付けられた

CloudWatch アラームChangeInCapacityがメトリクスのしきい値を下回った場合に、グループの容量を 2 インスタンス減少させます。

```
aws autoscaling put-scaling-policy \
  --auto-scaling-group-name my-asg  \
  --policy-name my-step-scale-in-policy \
  --policy-type StepScaling \
  --adjustment-type ChangeInCapacity \
  --step-adjustments MetricIntervalUpperBound=0.0,ScalingAdjustment=-2
```

ポリシーの Amazon リソースネーム (ARN) を記録します。ポリシーの CloudWatch アラームを作成するには、これが必要です。

```
{
    "PolicyARN": "arn:aws:autoscaling:region:123456789012:scalingPolicy:ac542982-cbeb-4294-891c-a5a941dfa787:autoScalingGroupName/my-asg:policyName/my-step-scale-out-policy
}
```

ステップ 2: メトリクスのしきい値が低い CloudWatch アラームを作成する

次の CloudWatch put-metric-alarm コマンドを使用して、2 分間の連続した 2 つの評価期間の平均 CPU しきい値 40% に基づいて Auto Scaling グループのサイズを小さくするアラームを作成します。独自のカスタムメトリクスを使用するには、名前を `--metric-name` で指定し、その名前空間を `--namespace` で指定します。

```
aws cloudwatch put-metric-alarm --alarm-name Step-Scaling-AlarmLow-RemoveCapacity \
  --metric-name CPUUtilization --namespace AWS/EC2 --statistic Average \
  --period 120 --evaluation-periods 2 --threshold 40 \
  --comparison-operator LessThanOrEqualToThreshold \
  --dimensions "Name=AutoScalingGroupName,Value=my-asg" \
  --alarm-actions PolicyARN
```

簡易スケーリングポリシー

次の例は、 CLI コマンドを使用してシンプルなスケーリングポリシーを作成する方法を示しています。これらは、使用したいすべてのお客様のリファレンスとしてこのドキュメントに残りますが、代わりにターゲット追跡ポリシーまたはステップスケーリングポリシーを使用することをお勧めします。

ステップスケーリングポリシーと同様に、シンプルなスケーリングポリシーでは、スケーリングポリシーの CloudWatch アラームを作成する必要があります。作成するポリシーでは、インスタンスを追加または削除するかどうか、およびインスタンスの数を定義するか、グループを正確なサイズに設定する必要があります。

ステップスケーリングポリシーと簡易スケーリングポリシーの主な違いの 1 つは、ステップスケーリングポリシーで得られるステップ調整です。ステップスケーリングを使用すると、指定したステップ調整に基づいて、グループのサイズを大きくまたは小さく変更できます。

簡易スケーリングポリシーは、進行中のスケーリングアクティビティまたはヘルスチェックの置き換えが完了し、クールダウン期間が終了するのを待ってから、追加のアラームに応答する必要があります。対照的に、ステップスケーリングでは、スケーリングアクティビティまたはヘルスチェックの置き換えが進行中であっても、ポリシーは追加のアラームに応答し続けます。つまり、Amazon EC2 Auto Scaling は、アラームメッセージを受信すると、すべてのアラーム違反を評価します。このため、スケーリング調整が 1 つだけであっても、代わりにステップスケーリングポリシーを使用することをお勧めします。

Amazon EC2 Auto Scaling は元々、簡易スケーリングポリシーのみをサポートしていました。ターゲット追跡ポリシーとステップスケーリングポリシーが導入される前にスケーリングポリシーを作成した場合、ポリシーは単純なスケーリングポリシーとして扱われます。

スケールアウト用のシンプルなスケーリングポリシーを作成する

次の put-scaling-policy コマンドを使用して、 という名前のシンプルなスケーリングポリシーを作成します。調整タイプはmy-simple-scale-out-policy、関連付けられた CloudWatch アラームPercentChangeInCapacityがメトリクスのしきい値を超えたときに、グループの容量を 30% 増やします。

```
aws autoscaling put-scaling-policy --policy-name my-simple-scale-out-policy \
  --auto-scaling-group-name my-asg --scaling-adjustment 30 \
  --adjustment-type PercentChangeInCapacity
```

ポリシーの Amazon リソースネーム (ARN) を記録します。ポリシーの CloudWatch アラームを作成するには、これが必要です。

スケールイン用のシンプルなスケーリングポリシーを作成する

次の put-scaling-policy コマンドを使用して、 という名前のシンプルなスケーリングポリシーを作成します。このポリシーの調整タイプはmy-simple-scale-in-policy、関連付けられた

CloudWatch アラームChangeInCapacityがメトリクスの低いしきい値を超えた場合に、グループの容量を 1 つのインスタンス減少させます。

```
aws autoscaling put-scaling-policy --policy-name my-simple-scale-in-policy \
  --auto-scaling-group-name my-asg --scaling-adjustment -1 \
  --adjustment-type ChangeInCapacity --cooldown 180
```

ポリシーの Amazon リソースネーム (ARN) を記録します。ポリシーの CloudWatch アラームを作成するには、これが必要です。

Amazon EC2 Auto Scaling のスケーリングクールダウン

> **Important**
>
> ベストプラクティスとして、シンプルスケーリングポリシーとスケーリングクールダウンを使用しないことをお勧めします。ターゲット追跡スケーリングポリシーまたはステップスケーリングポリシーは、スケーリングパフォーマンスに適しています。スケーリングメトリクスの値が減少または増加するにつれて Auto Scaling グループのサイズを比例して変更するスケーリングポリシーの場合は、簡易スケーリングまたはステップスケーリングのいずれかでターゲットを追跡することをお勧めします。

Auto Scaling グループに対してシンプルなスケーリングポリシーを作成する場合は、スケーリングのクールダウンを同時に設定することをお勧めします。

Auto Scaling グループは、インスタンスの起動または終了後、クールダウン期間が終了するのを待ってから、シンプルスケーリングポリシーによって開始される追加のスケーリングアクティビティを開始します。クールダウン期間の目的は、Auto Scaling グループを安定させ、以前のスケーリングアクティビティの影響が表示される前に追加のインスタンスを起動または終了しないようにすることです。

例えば、CPU 使用率のシンプルスケーリングポリシーがインスタンスを 2 個起動することを推奨するとします。Amazon EC2 Auto Scaling はインスタンスを 2 個起動してから、クールダウン期間が終了するまでスケーリングアクティビティを一時停止します。シンプルスケーリングポリシーによって開始されたスケーリングアクティビティを再開できるのは、このクールダウン期間の終了後になります。CPU 使用率がアラーム上限しきい値を再度超過すると、Auto Scaling グループが再度スケールアウトし、クールダウン期間も再度有効になります。ただし、メトリクス値を低減させるためには 2 個のインスタンスで十分であった場合、グループは現行のサイズのままになります。

内容

- 考慮事項
- 追加の遅延を発生させる可能性のあるライフサイクルフック
- デフォルトのクールダウン期間を変更する
- 特定のシンプルスケーリングポリシーにクールダウン期間を設定する

考慮事項

以下の考慮事項は、シンプルスケーリングポリシーとスケーリングクールダウンの使用時に適用されます。

- ターゲット追跡ポリシーとステップスケーリングポリシーは、クールダウン期間の終了を待つことなく、ただちにスケールアウトアクティビティを開始できます。代わりに、Auto Scaling グループがインスタンスを起動するたびに、個々のインスタンスにウォームアップ期間があります。詳細については、「Auto Scaling グループに対するインスタンスのデフォルトウォームアップを設定する」を参照してください。
- スケジュールされたアクションがスケジュールされた時刻に開始されるときも、クールダウン期間の終了を待つことなく、ただちにスケーリングアクティビティを開始できます。
- インスタンスが異常になった場合、Amazon EC2 Auto Scaling はクールダウン期間の終了を待たずに異常なインスタンスを置き換えます。
- 複数のインスタンスが起動または終了される場合は、最後のインスタンスの起動または終了が完了した時点でクールダウン期間 (デフォルトのクールダウンまたはスケーリングポリシー固有のクールダウン) が実施されます。
- Auto Scaling グループを手動でスケールするときは、クールダウンの終了を待たないことがデフォルトになります。ただし、 AWS CLI または SDK を使用して手動でスケーリングする場合は、この動作を上書きしてデフォルトのクールダウンを適用できます。
- デフォルトで、Elastic Load Balancing は登録解除 (Connection Draining) プロセスが完了するまで 300 秒間待機します。グループが Elastic Load Balancing ロードバランサーの背後にある場合は、クールダウン期間を開始する前に、終了されるインスタンスが登録解除されるのを待ちます。

追加の遅延を発生させる可能性のあるライフサイクルフック

ライフサイクルフックが呼び出される場合、クールダウン期間はライフサイクルアクションの完了後、またはタイムアウト期間の終了後に開始されます。例えば、インスタンス起動用のライフサイ

クルフックがある Auto Scaling グループについて考えてみましょう。アプリケーションで需要の増加が生じると、グループはキャパシティーを追加するためにインスタンスを起動します。ライフサイクルフックが存在することから、インスタンスは待機状態になり、シンプルスケーリングポリシーによるスケーリングアクティビティは一時停止されます。インスタンスが `InService` 状態になると、クールダウン期間が開始します。クールダウン期間が終了すると、シンプルなスケーリングポリシーのアクティビティが再開されます。

Elastic Load Balancing を有効にすると、スケールインの目的で、終了用に選択されたインスタンスが Connection Draining (登録解除の遅延) を開始したときにクールダウン期間が開始されます。クールダウン期間は、Connection Draining が完了したり、ライフサイクルフックがそのアクションを完了するのを待つことはありません。つまり、シンプルスケーリングポリシーによるスケーリングアクティビティは、スケールインイベントの結果がグループのキャパシティーに反映され次第再開できることになります。これ以外の場合は、3 つのアクティビティ (Connection Draining、ライフサイクルフック、およびクールダウン期間) のすべてが完了するまで待機することになり、Auto Scaling グループがスケーリングを一時停止しなければならない時間が大幅に長くなります。

デフォルトのクールダウン期間を変更する

Amazon EC2 Auto Scaling コンソールで Auto Scaling グループを初めて作成するときにデフォルトクールダウンを設定することはできません。デフォルトで、このクールダウン期間は 300 秒 (5 分) に設定されています。これは、グループの作成後に必要に応じて更新できます。

デフォルトのクールダウン期間を変更する (コンソール)

Auto Scaling グループの作成後、[Details] (詳細) タブの [Advanced configurations] (高度な設定) で [Edit] (編集) を選択します。[Default cooldown] (デフォルトのクールダウン) で、インスタンスの起動時間やその他のアプリケーションニーズに基づいて、希望する時間を選択します。

デフォルトのクールダウン期間を変更する (AWS CLI)

以下のコマンドを使用して、新しい、または既存の Auto Scaling グループのデフォルトのクールダウンを変更します。デフォルトのクールダウンが定義されていない場合は、デフォルト値の 300 秒が使用されます。

- create-auto-scaling-group
- update-auto-scaling-group

デフォルトのクールダウンの値を確認するには、describe-auto-scaling-groups コマンドを使用します。

特定のシンプルスケーリングポリシーにクールダウン期間を設定する

デフォルトで、すべてのシンプルスケーリングポリシーは Auto Scaling グループに定義されているデフォルトのクールダウン期間を使用します。特定のシンプルスケーリングポリシーにクールダウン期間を設定するには、ポリシーを作成または更新するときに、オプションのクールダウンパラメータを使用します。ポリシーにクールダウン期間を指定すると、デフォルトのクールダウンが上書きされます。

スケーリングポリシー固有のクールダウン期間の一般的な用途の 1 つに、スケールインポリシーとの使用があります。このポリシーはインスタンスを削除するため、Amazon EC2 Auto Scaling が追加のインスタンスを削除するかどうかを判断するために必要な時間が減ります。インスタンスを終了するオペレーションは、インスタンスを起動するよりもはるかに高速です。したがって、デフォルトのクールダウン期間である 300 秒は長すぎます。この場合、スケールインポリシーの値をより短くしたスケーリングポリシー固有のクールダウン期間は、グループがより迅速にスケールインできるようにすることで、コストの削減に役立ちます。

コンソールでシンプルスケーリングポリシーを作成または更新するには、グループの作成後に [Automatic scaling] (自動スケーリング) タブを選択します。を使用してシンプルなスケーリングポリシーを作成または更新するには AWS CLI、put-scaling-policy コマンドを使用します。詳細については、「ステップスケーリングポリシーおよび簡易スケーリングポリシー」を参照してください。

Amazon SQS に基づくスケーリング

> **Important**
>
> 以下の情報とステップは、カスタムメトリクスとして に発行する前に、キュー属性を使用してインスタンスごとに Amazon SQS `ApproximateNumberOfMessages`キューバックログを計算する方法を示しています CloudWatch。ただし、Metric Math を使用することで、独自のメトリクスを公開するコストと労力を節約できるようになりました。詳細については、「Metric Math を使用する、Amazon EC2 Auto Scaling のターゲット追跡スケーリングポリシーを作成する」を参照してください。

このセクションでは、Amazon Simple Queue Service (Amazon SQS) キュー内のシステムロードの変化に応じて Auto Scaling グループをスケーリングする方法について説明します。Amazon SQS の使用方法の詳細については、「Amazon Simple Queue Service 開発者ガイド」をご参照ください。

Amazon SQS キューのアクティビティに応じたスケーリングを検討するシナリオはいくつかあります。例えば、ユーザーがイメージをアップロードしてオンラインで使用できるウェブアプリがある

とします。このシナリオでは、各イメージはサイズ変更されエンコードされてから公開されます。このアプリケーションは、標準的なアップロード速度を処理するように設定された Auto Scaling グループ内の EC2 インスタンスで実行されます。異常なインスタンスは終了され置き換えられて、常に現在のインスタンスレベルが維持されます。このアプリは処理のためイメージの raw ビットマップデータを SQS キューに配置します。イメージが処理され、処理されたイメージはユーザーから確認できる場所に発行されます。このシナリオのアーキテクチャは、イメージのアップロード数が時間の経過とともに変化しない場合に有効です。ただし、アップロード数が時間の経過とともに変化する場合は、動的スケーリングを使用して Auto Scaling グループのキャパシティーを拡張することを検討してください。

内容

- 適切なメトリクスを用いたターゲット追跡を使用する
- 制限事項と前提条件
- Amazon SQS に基づくスケーリングを設定
- Amazon SQS とインスタンスのスケールイン保護

適切なメトリクスを用いたターゲット追跡を使用する

カスタム Amazon SQS キューメトリクスに基づくターゲット追跡スケーリングポリシーを使用する場合、動的スケーリングはアプリケーションの需要曲線に合わせてより効果的に調整できます。ターゲット追跡のメトリクスの選択の詳細については、「メトリクスを選択する」を参照してください。

ターゲット追跡`ApproximateNumberOfMessagesVisible`に などの CloudWatch Amazon SQS メトリクスを使用する場合の問題は、キュー内のメッセージ数が、キューからのメッセージを処理する Auto Scaling グループのサイズに比例して変化しない可能性があることです。これは SQS キューにあるメッセージの数のみでは、必要なインスタンスの数は定義されないためです。Auto Scaling グループ内のインスタンスの数は複数の要因によって決まります。例えば、メッセージを処理するためにかかる時間や、許容されるレイテンシー (キュー遅延) の量などです。

解決策は、インスタンスあたりのバックログのメトリクスを使用して、ターゲット値を維持するインスタンスあたりの適正バックログにすることです。これらの数は以下のように計算できます。

- インスタンスあたりのバックログ: インスタンスあたりのバックログを計算するには、まず `ApproximateNumberOfMessages` キュー属性を使用して SQS キューの長さ (このキューから取得できるメッセージ数) を決定します。その数をフリートの実行キャパシティーで割ります。Auto Scaling グループの場合、これは `InService` 状態にあるインスタンスの数です。これでインスタンスあたりのバックログを得ることができます。

- インスタンスあたりの適正バックログ: ターゲット値を計算するには、まずレイテンシーの点でアプリケーションが受け付けることができる数を決定します。次に、適切なレイテンシー値を取ってそれを EC2 インスタンスがメッセージを処理する平均所要時間で割ります。

例として、現在、10 個のインスタンスを持つ Auto Scaling グループがあり、キュー (`ApproximateNumberOfMessages`) にある可視メッセージの数が 1,500 であるとします。メッセージあたりの平均処理時間が 0.1 秒で、最長許容レイテンシーが 10 秒の場合、インスタンスあたりの許容バックログは 10/0.1、つまり 100 メッセージとなります。つまり、100 がターゲット追跡ポリシーのターゲット値です。インスタンスあたりのバックログがターゲット値に達すると、スケールアウトイベントが発生します。インスタンスあたりのバックログが既に 150 メッセージ (1,500 メッセージ/10 インスタンス) あるため、グループは、ターゲット値との比例を維持するために 5 インスタンス分スケールアウトします。

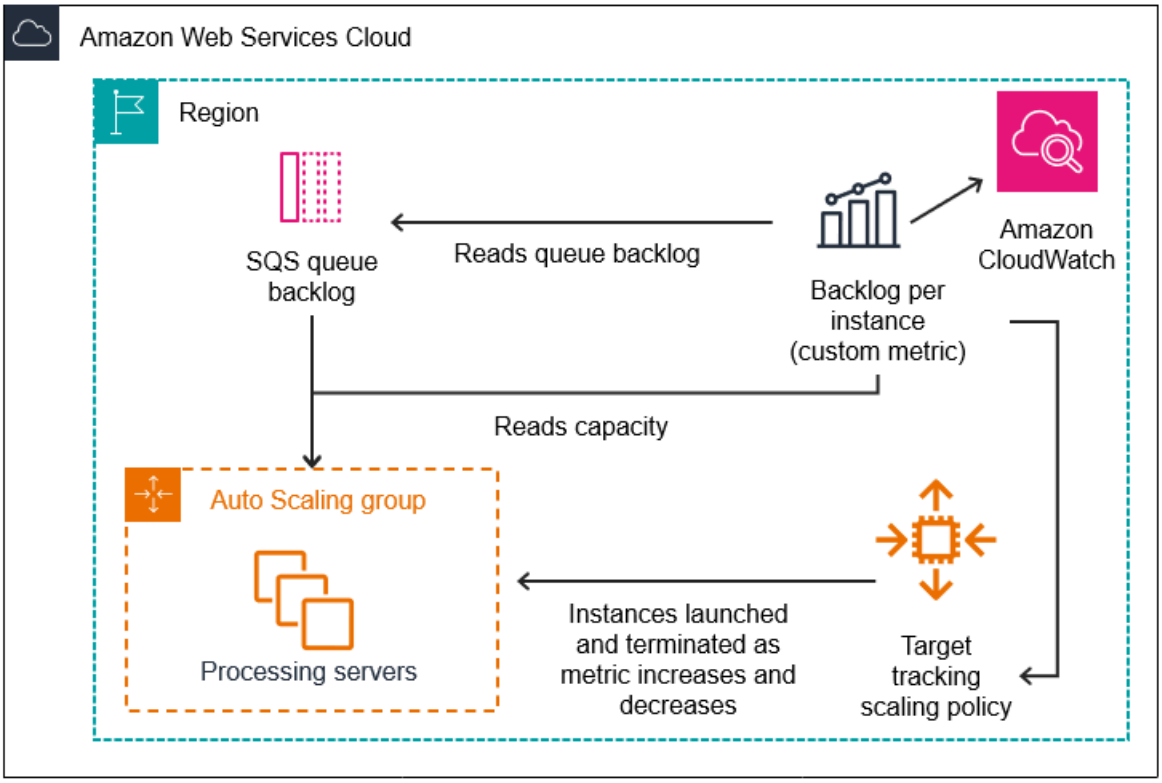
次の手順は、カスタムメトリクスを公開し、これらの計算に基づいてスケーリングするように Auto Scaling グループを設定するターゲット追跡スケーリングポリシーを作成する方法を示しています。

> **Important**
>
> コストを削減するには、必ず代わりに Metric Math を使用してください。詳細については、「Metric Math を使用する、Amazon EC2 Auto Scaling のターゲット追跡スケーリングポリシーを作成する」を参照してください。

この設定は 3 つの主要な部分で構成されます。

- SQS キューからのメッセージを処理するために EC2 インスタンスを管理する Auto Scaling グループ。
- Auto Scaling グループの EC2 インスタンスごとにキュー内のメッセージ数 CloudWatch を測定する Amazon に送信するカスタムメトリクス。
- カスタムメトリクスと設定されたターゲット値. CloudWatch alarms に基づいてスケーリングするように Auto Scaling グループを設定するターゲット追跡ポリシーは、スケーリングポリシーを呼び出します。

次の図は、この設定のアーキテクチャを示しています。



制限事項と前提条件

この設定を使用するには、次の制限事項に注意する必要があります。

- カスタムメトリクスを に発行するには、 AWS CLI または SDK を使用する必要があります CloudWatch。その後、 を使用してメトリクスをモニタリングできます AWS Management Console。
- Amazon EC2 Auto Scaling コンソールには、カスタムメトリックスを使用するターゲット追跡スケーリングポリシーはありません。スケーリングポリシーのカスタムメトリクスを指定するには、AWS CLI または SDK を使用する必要があります。

以下のセクションでは、実行する必要のあるタスク AWS CLI に を使用する方法について説明します。例えば、キューの現在の使用状況を反映するメトリックスデータを取得するには、SQS get-queue-attributes コマンドを使用します。CLI がインストールされ、設定されていることを確認します。

開始する前に、使用する Amazon SQS キューが必要です。以下のセクションは、キュー (標準または FIFO)、Auto Scaling グループ、キューを使用するアプリケーションを実行している EC2 インス

タンスがあることを前提としています。Amazon SQS の詳細については、「Amazon Simple Queue Service Developer Guide」を参照してください。

Amazon SQS に基づくスケーリングを設定

タスク

- ステップ 1: CloudWatch カスタムメトリクスを作成する
- ステップ 2: ターゲット追跡スケーリングポリシーを作成する
- ステップ 3: スケーリングポリシーをテストする

ステップ 1: CloudWatch カスタムメトリクスを作成する

カスタムメトリックスは、選択したメトリックス名と名前空間を使用して定義されます。カスタムメトリクスの名前空間を AWS/ で始めることはできません。カスタムメトリクスの発行の詳細については、「Amazon CloudWatch ユーザーガイド」の「カスタムメトリクスの発行」トピックを参照してください。

この手順に従って、まず AWS アカウントから情報を読み取ることでカスタムメトリクスを作成します。次に、前のセクションで推奨されたようにインスタンスメトリクスごとにバックログを計算します。最後に、この番号を 1 分単位で CloudWatch に発行します。可能な限り、システム負荷の変化に迅速に対応できるように、メトリクスを 1 分単位でスケーリングすることを強くお勧めします。

CloudWatch カスタムメトリクスを作成するには (AWS CLI)

1. SQS get-queue-attributes コマンドを使用して、キューで待機しているメッセージ数を取得します (ApproximateNumberOfMessages)。

```
aws sqs get-queue-attributes --queue-url https://
sqs.region.amazonaws.com/123456789/MyQueue \
  --attribute-names ApproximateNumberOfMessages
```

2. describe-auto-scaling-groups コマンドを使用して、グループの実行キャパシティーを取得します。これは InService ライフサイクル状態にあるインスタンスの数です。このコマンドは、Auto Scaling グループのインスタンスとそのライフサイクル状態を返します。

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-names my-asg
```

3. キューからの取得に使用できるメッセージの概数をグループの実行キャパシティで除算して、インスタンスあたりのバックログを算出します。

4. 1 分ごとに実行されるスクリプトを作成して、インスタンス値ごとにバックログを取得し、CloudWatch カスタムメトリクスに発行します。カスタムメトリクスを公開する際は、そのメトリクスの名前、名前空間、単位、値、および 0 以上のディメンションを指定します。ディメンションには、そのディメンションの名前と値を含みます。

   カスタムメトリクスを公開するには、*##*で示されたプレースホルダーの値を好みのメトリクス名、メトリクスの値、名前空間 (「AWS」で開始することはできません)、ディメンション (オプション) に置き換えてから、次の put-metric-data コマンドを実行します。

   ```
   aws cloudwatch put-metric-data --metric-name MyBacklogPerInstance --
   namespace MyNamespace \
     --unit None --value 20 --
   dimensions MyOptionalMetricDimensionName=MyOptionalMetricDimensionValue
   ```

アプリケーションが目的のメトリクスを出力すると、データは に送信されます CloudWatch。メトリクスは CloudWatch コンソールに表示されます。にログイン AWS Management Console し、ページに移動することでアクセスできます CloudWatch 。その後、メトリクスを表示するには、メトリクスページに移動するか、検索ボックスを使用してメトリクスを検索します。メトリクスの表示の詳細については、「Amazon ユーザーガイド」の「使用可能なメトリクスの表示」を参照してください。CloudWatch

ステップ 2: ターゲット追跡スケーリングポリシーを作成する

この時点で、作成したメトリクスをターゲット追跡スケーリングポリシーに追加できるようになっています。

ターゲット追跡スケーリングポリシーを作成するには (AWS CLI)

1. 以下の cat コマンドを使用して、スケーリングポリシーのターゲット値と、カスタムメトリクスの仕様を指定する JSON ファイル (名前: config.json) をホームディレクトリに保存します。各*##############*を独自の情報に置き換えます。TargetValue には、インスタンスあたりの適正バックログメトリックスを計算して、それを入力します。この数を計算するには、上記のセクションで説明しているとおり、標準のレイテンシー値を決定し、その値をメッセージの処理にかかる平均時間で割ります。

   ステップ 1 で作成したメトリクスにディメンションを指定しなかった場合は、カスタムメトリクスの仕様にディメンションを含めないでください。

   ```
   $ cat ~/config.json
   ```

```
{
   "TargetValue":100,
   "CustomizedMetricSpecification":{
      "MetricName":"MyBacklogPerInstance",
      "Namespace":"MyNamespace",
      "Dimensions":[
         {
            "Name":"MyOptionalMetricDimensionName",
            "Value":"MyOptionalMetricDimensionValue"
         }
      ],
      "Statistic":"Average",
      "Unit":"None"
   }
}
```

2. ut-scaling-policy コマンドを使用して、前のステップで作成した config.json と共に、スケーリングポリシーを作成します。

```
aws autoscaling put-scaling-policy --policy-name sqs100-target-tracking-scaling-policy \
  --auto-scaling-group-name my-asg --policy-type TargetTrackingScaling \
  --target-tracking-configuration file://~/config.json
```

これにより、2 つのアラーム (スケールアウトとスケールイン) が作成されます。また、 に登録されているポリシーの Amazon リソースネーム (ARN) も返します。このポリシーは CloudWatch、 CloudWatch を使用してメトリクスのしきい値を超えるたびにスケーリングを呼び出します。

ステップ 3: スケーリングポリシーをテストする

設定が完了したら、スケーリングポリシーが機能していることを確認します。SQS キュー内のメッセージ数を増やし、Auto Scalingグループが追加の EC2 インスタンスを起動したことを確認することによってテストできます。SQS キュー内のメッセージ数を減らし、Auto Scaling グループが EC2 インスタンスを終了したことを確認することによってもテストできます。

スケールアウト機能をテストするには

1. Amazon SQS標準キューを作成してメッセージを送信する」またはAmazon SQS FIFO キューを作成してキューにメッセージを追加するメッセージを送信する」のステップに従います。インス

タンスあたりのバックログメトリックスがターゲット値を超えるようにキュー内のメッセージ数を増やしたことを確認します。

この変更により、アラームの呼び出しに数分かかる場合があります。

2. describe-auto-scaling-groups コマンドを使用して、グループがインスタンスを起動したことを確認します。

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
```

スケールイン機能をテストするには

1. 「メッセージの受信と削除 (コンソール)」の手順に従って、キューからメッセージを削除します。インスタンスあたりのバックログメトリックスがターゲット値を下回るようにキュー内のメッセージ数を減らしたことを確認します。

   この変更により、アラームの呼び出しに数分かかる場合があります。

2. describe-auto-scaling-groups コマンドを使用して、グループがインスタンスを終了したことを確認します。

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
```

Amazon SQS とインスタンスのスケールイン保護

インスタンスの削除時に処理されていなかったメッセージは、SQS キューに戻され、まだ実行中である別のインスタンスで処理されます。長時間実行されるタスクが実行されるアプリケーションでは、オプションでインスタンススケールイン保護を使用して、Auto Scaling グループのスケールイン時に終了するキューワーカーを制御できます。

次の擬似コードは、長時間実行されるキュー駆動のワーカープロセスをスケールイン終了から保護する方法の 1 つを示しています。

```
while (true)
{
  SetInstanceProtection(False);
  Work = GetNextWorkUnit();
  SetInstanceProtection(True);
  ProcessWorkUnit(Work);
  SetInstanceProtection(False);
```

```
}
```

詳細については、「インスタンスの終了を正常に処理するために、Amazon EC2 Auto Scaling でアプリケーションを設計する」を参照してください。

Auto Scaling グループのスケーリングアクティビティを検証する

Amazon EC2 コンソールの Amazon EC2 Auto Scaling セクションでは、Auto Scaling グループの [Activity history] (アクティビティ履歴) で、現在進行中のスケーリングアクティビティの最新のステータスを表示できます。スケーリングが終了すると、それが成功したか失敗したかを確認できます。これは、Auto Scaling グループを作成している場合や、既存のグループにスケーリング条件を追加している場合に特に便利です。

ターゲット追跡、ステップ、または単純なスケーリングポリシーを Auto Scaling グループに追加すると、Amazon EC2 Auto Scaling は直ちにメトリクスに対するポリシーの評価を開始します。指定した評価期間数にわたってメトリクスがしきい値を超えると、メトリクスアラームが ALARM 状態に移行します。これは、スケーリングポリシーは、それが作成された直後から、スケーリング処理に反映されることを意味します。Amazon EC2 Auto Scaling がスケーリングポリシーに応じてキャパシティーを調整した後、アカウントでのスケーリングが確認できるようになります。スケーリングの処理に関する E メール通知を Amazon EC2 Auto Scaling から受信する場合は、「Amazon EC2 Auto Scaling の Amazon SNS 通知オプション Amazon EC2 Auto Scaling」の手順に従います。

> **Tip**
>
> 以下の手順では、Auto Scaling グループについて [Activity history] (アクティビティ履歴) と [Instances] (インスタンス) の各セクションを調べます。どちらのセクションにも、名前付きの列がすでに表示されているはずです。非表示の列を表示する、または表示される行の数を変更するには、各セクションの右上隅にある歯車アイコンを選択して設定モーダルを開き、必要に応じて設定を更新してから、[Confirm] (確認) を選択します。

Auto Scaling グループのスケーリングアクティビティを表示するには (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. 画面の上部のナビゲーションバーで、Auto Scaling グループが配置されているリージョンを選択します。
3. Auto Scaling グループの横にあるチェックボックスを選択します。

ページの下部にスプリットペインが開きます。

4. [Activity] (アクティビティ) タブで、[Activity history] (アクティビティ履歴) の下の [Status] (ステータス) 列に、Auto Scaling グループがインスタンスを正常に起動あるいは終了したか、または、依然としてスケーリングが進行中なのかが表示されます。
5. (オプション) スケーリングアクティビティが多数存在する場合は、アクティビティ履歴の上部端にある [>] アイコンを選択して、スケーリングアクティビティの次ページを表示します。
6. [Instance management (インスタンス管理)] タブにある、[Instances (インスタンス)] の [Lifecycle (ライフサイクル)] 列にインスタンスの状態が含まれます。インスタンスが起動し、ライフサイクルフックが終了すると、そのライフサイクル状態は `InService` に変わります。[Health Status (ヘルスステータス)] 列には、インスタンスの EC2 インスタンスのヘルスチェックの結果が表示されます。

Auto Scaling グループのスケーリングアクティビティを表示するには (AWS CLI)

以下の describe-scaling-activities コマンドを実行します。

```
aws autoscaling describe-scaling-activities --auto-scaling-group-name my-asg
```

以下は出力例です。

スケーリングアクティビティは、開始時刻順に並べられます。まだ進行中のアクティビティを最初に説明します。

```
{
  "Activities": [
    {
      "ActivityId": "5e3a1f47-2309-415c-bfd8-35aa06300799",
      "AutoScalingGroupName": "my-asg",
      "Description": "Terminating EC2 instance: i-06c4794c2499af1df",
      "Cause": "At 2020-02-11T18:34:10Z a monitor alarm TargetTracking-my-asg-AlarmLow-
b9376cab-18a7-4385-920c-dfa3f7783f82 in state ALARM triggered policy my-target-
tracking-policy changing the desired capacity from 3 to 2.  At 2020-02-11T18:34:31Z
 an instance was taken out of service in response to a difference between desired and
 actual capacity, shrinking the capacity from 3 to 2.  At 2020-02-11T18:34:31Z instance
 i-06c4794c2499af1df was selected for termination.",
      "StartTime": "2020-02-11T18:34:31.268Z",
      "EndTime": "2020-02-11T18:34:53Z",
      "StatusCode": "Successful",
      "Progress": 100,
```

```
      "Details": "{\"Subnet ID\":\"subnet-5ea0c127\",\"Availability Zone\":\"us-west-2a
\"...}",
      "AutoScalingGroupARN": "arn"
    },
...
  ]
}
```

出力の各フィールドについては、「Amazon EC2 Auto Scaling API Reference」(Amazon EC2 Auto Scaling API リファレンス) の「Activity」(アクティビティ) を参照してください。

削除されたグループのスケーリングアクティビティの取得方法と、発生する可能性のあるエラーの種類と処理方法の詳細については、「Amazon EC2 Auto Scaling をトラブルシューティングする」を参照してください。

Auto Scaling グループのスケーリングポリシーを無効化する

このトピックでは、Auto Scaling グループに含まれるインスタンス数の変更を開始しないように、スケーリングポリシーを一時的に無効にする方法について説明します。スケーリングポリシーを無効にすると、設定の詳細が保持されるため、ポリシーをすばやく再度有効にできます。これは、不要なポリシーを一時的に削除し、後で再作成するよりも簡単です。

スケーリングポリシーが無効になっている場合、スケーリングポリシーが無効になっている間は、違反したメトリクスアラームに対して Auto Scaling グループはスケールアウトまたはスケールインしません。ただし、進行中のスケーリングアクティビティは停止しません。

無効にしたスケーリングポリシーは、Auto Scaling グループに追加できるスケーリングポリシーの数のクォータにカウントされることに注意してください。

スケーリングポリシーを無効にするには (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。
3. [Automatic scaling] (オートスケーリング) タブの、[Dynamic scaling policies] (動的スケーリングポリシー) で、目的のスケーリングポリシーの右上隅にあるチェックボックスをオンにします。
4. 画面の上部までスクロールし、[Dynamic scaling policies] (動的スケーリングポリシー) セクションを選択した後、[Actions] (アクション)、[Disable] (無効) の順にクリックします。

スケーリングポリシーを再度有効にする準備ができたら、これらのステップを繰り返し、[Actions (アクション)]、[Enable (有効にする)] の順に選択します。スケーリングポリシーを再度有効にすると、現在 ALARM 状態のアラームがある場合、Auto Scaling グループは直ちにスケーリングアクションを開始できます。

スケーリングポリシーを無効にするには (AWS CLI)

`--no-enabled` オプションを次のように使用して、put-scolicy コマンドを使用します。ポリシーの作成時に指定するオプションと同様に、コマンド内のすべてのオプションを指定します。

```
aws autoscaling put-scaling-policy --auto-scaling-group-name my-asg \
   --policy-name my-scaling-policy --policy-type TargetTrackingScaling \
   --estimated-instance-warmup 360 \
   --target-tracking-configuration '{ "TargetValue": 70,
 "PredefinedMetricSpecification": { "PredefinedMetricType":
 "ASGAverageCPUUtilization" } }' \
   --no-enabled
```

スケーリングポリシーを再度有効にするには (AWS CLI)

`--enabled` オプションを次のように使用して、put-scolicy コマンドを使用します。ポリシーの作成時に指定するオプションと同様に、コマンド内のすべてのオプションを指定します。

```
aws autoscaling put-scaling-policy --auto-scaling-group-name my-asg \
   --policy-name my-scaling-policy --policy-type TargetTrackingScaling \
   --estimated-instance-warmup 360 \
   --target-tracking-configuration '{ "TargetValue": 70,
 "PredefinedMetricSpecification": { "PredefinedMetricType":
 "ASGAverageCPUUtilization" } }' \
   --enabled
```

スケーリングポリシーを説明するには (AWS CLI)

describe-policies コマンドを使用して、スケーリングポリシーの有効なステータスを確認します。

```
aws autoscaling describe-policies --auto-scaling-group-name my-asg \
   --policy-names my-scaling-policy
```

以下は出力例です。

```
{
```

```
    "ScalingPolicies": [
        {
            "AutoScalingGroupName": "my-asg",
            "PolicyName": "my-scaling-policy",
            "PolicyARN": "arn:aws:autoscaling:us-
west-2:123456789012:scalingPolicy:1d52783a-b03b-4710-
bb0e-549fd64378cc:autoScalingGroupName/my-asg:policyName/my-scaling-policy",
            "PolicyType": "TargetTrackingScaling",
            "StepAdjustments": [],
            "Alarms": [
                {
                    "AlarmName": "TargetTracking-my-asg-
AlarmHigh-9ca53fdd-7cf5-4223-938a-ae1199204502",
                    "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmHigh-9ca53fdd-7cf5-4223-938a-
ae1199204502"
                },
                {
                    "AlarmName": "TargetTracking-my-asg-AlarmLow-7010c83d-d55a-4a7a-
abe0-1cf8b9de6d6c",
                    "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmLow-7010c83d-d55a-4a7a-
abe0-1cf8b9de6d6c"
                }
            ],
            "TargetTrackingConfiguration": {
                "PredefinedMetricSpecification": {
                    "PredefinedMetricType": "ASGAverageCPUUtilization"
                },
                "TargetValue": 70.0,
                "DisableScaleIn": false
            },
            "Enabled": true
        }
    ]
}
```

スケーリングポリシーを削除する

不要になったスケーリングポリシーは削除できます。スケーリングポリシーのタイプによっては、CloudWatch アラームを削除する必要がある場合もあります。ターゲット追跡スケーリングポリシーを削除すると、関連する CloudWatch アラームもすべて削除されます。ステップスケーリングポリ

シーまたは簡易スケーリングポリシーを削除すると、基盤となるアラームアクションは削除されますが、アクションが関連付けられていなくても CloudWatch アラームは削除されません。

スケーリングポリシーを削除するには (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。
3. [Automatic scaling] (オートスケーリング) タブの、[Dynamic scaling policies] (動的スケーリングポリシー) で、目的のスケーリングポリシーの右上隅にあるチェックボックスをオンにします。
4. 画面の上部までスクロールし、[Dynamic scaling policies] (動的スケーリングポリシー) セクションを選択した後、[Actions] (アクション)、[Delete] (削除) の順にクリックします。
5. 確認を求めるメッセージが表示されたら、[Yes、Delete] を選択します。
6. (オプション) ステップスケーリングポリシーまたは簡易スケーリングポリシーを削除した場合は、次の手順を実行して、ポリシーに関連付けられた CloudWatch アラームを削除します。今後使用できるように、このサブステップをスキップしてアラームを保持することもできます。

   a. https://console.aws.amazon.com/cloudwatch/ で CloudWatch コンソールを開きます。

   b. ナビゲーションペインで、[アラーム] を選択します。

   c. アラームを選び (`Step-Scaling-AlarmHigh-AddCapacity` など)、[Action (アクション)]、[Delete (削除)] を選択します。

   d. 確認を求めるメッセージが表示されたら、[削除] を選択します。

Auto Scaling グループのスケーリングポリシーを取得するには (AWS CLI)

スケーリングポリシーを削除する前に、次の describe-policies を使用して、Auto Scaling グループに対して作成されたスケーリングポリシーを確認します。出力は、ポリシーと CloudWatchアラームを削除するときに使用できます。

```
aws autoscaling describe-policies --auto-scaling-group-name my-asg
```

`--query` パラメータを使用して、スケーリングポリシーのタイプで結果をフィルタリングできます。この `query` の構文は Linux または macOS で動作します。Windows では、一重引用符を二重引用符に変更します。

```
aws autoscaling describe-policies --auto-scaling-group-name my-asg
  --query 'ScalingPolicies[?PolicyType==`TargetTrackingScaling`]'
```

以下は出力例です。

```
[
    {
        "AutoScalingGroupName": "my-asg",
        "PolicyName": "cpu50-target-tracking-scaling-policy",
        "PolicyARN": "PolicyARN",
        "PolicyType": "TargetTrackingScaling",
        "StepAdjustments": [],
        "Alarms": [
            {
                "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmHigh-
fc0e4183-23ac-497e-9992-691c9980c38e",
                "AlarmName": "TargetTracking-my-asg-AlarmHigh-
fc0e4183-23ac-497e-9992-691c9980c38e"
            },
            {
                "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmLow-61a39305-ed0c-47af-
bd9e-471a352ee1a2",
                "AlarmName": "TargetTracking-my-asg-AlarmLow-61a39305-ed0c-47af-
bd9e-471a352ee1a2"
            }
        ],
        "TargetTrackingConfiguration": {
            "PredefinedMetricSpecification": {
                "PredefinedMetricType": "ASGAverageCPUUtilization"
            },
            "TargetValue": 50.0,
            "DisableScaleIn": false
        },
        "Enabled": true
    }
]
```

スケーリングポリシーを削除するには (AWS CLI)

次の [delete-policy] コマンドを使用します。

```
aws autoscaling delete-policy --auto-scaling-group-name my-asg \
  --policy-name cpu50-target-tracking-scaling-policy
```

CloudWatch アラームを削除するには (AWS CLI)

ステップポリシーと簡易スケーリングポリシーでは、delete-alarms コマンドを使用して、ポリシーに関連付けられた CloudWatch アラームを削除します。今後使用できるように、このステップをスキップしてアラームを保持することもできます。1 つ以上のアラームを一度に削除することができます。例えば、次のコマンドを使用して Step-Scaling-AlarmHigh-AddCapacity アラームおよび Step-Scaling-AlarmLow-RemoveCapacity アラームを削除します。

```
aws cloudwatch delete-alarms --alarm-name Step-Scaling-AlarmHigh-AddCapacity Step-
Scaling-AlarmLow-RemoveCapacity
```

AWS Command Line Interface (AWS CLI) のスケーリングポリシーの例

Amazon EC2 Auto Scaling のスケーリングポリシーは AWS Management Console、、 AWS CLI、または SDKsを使用して作成できます。

次の例は、put-scaling-policy コマンドを使用して Amazon EC2 Auto Scaling のスケーリングポリシーを作成する方法を示しています。 Auto Scaling AWS CLI https://docs.aws.amazon.com/cli/latest/reference/autoscaling/put-scaling-policy.html 各##############を独自の情報に置き換えます。

を使用してスケーリングポリシーの作成を開始するには AWS CLI、ターゲット追跡スケーリングポリシー「」および「」の入門演習を参照してくださいステップスケーリングポリシーおよび簡易スケーリングポリシー。

例 1: 事前定義されたメトリクス指定を使用してターゲット追跡スケーリングポリシーを適用するには

```
aws autoscaling put-scaling-policy --policy-name cpu50-target-tracking-scaling-policy \
  --auto-scaling-group-name my-asg --policy-type TargetTrackingScaling \
  --target-tracking-configuration file://config.json
{
  "TargetValue": 50.0,
  "PredefinedMetricSpecification": {
    "PredefinedMetricType": "ASGAverageCPUUtilization"
  }
}
```

詳細については、Amazon EC2 Auto Scaling API リファレンス」のPredefinedMetric「仕様」を参照してください。

> **Note**
>
> ファイルが現在のディレクトリにない場合は、ファイルへのフルパスを入力します。ファイルから AWS CLI パラメータ値を読み取る方法の詳細については、「 ユーザーガイド」の「ファイルからのパラメータのロード AWS CLI AWS Command Line Interface 」を参照してください。

例 2: カスタマイズされたメトリクス仕様を使用してターゲット追跡スケーリングポリシーを適用するには

```
aws autoscaling put-scaling-policy --policy-name sqs100-target-tracking-scaling-policy
 \
  --auto-scaling-group-name my-asg --policy-type TargetTrackingScaling \
  --target-tracking-configuration file://config.json
{
  "TargetValue": 100.0,
  "CustomizedMetricSpecification": {
    "MetricName": "MyBacklogPerInstance",
    "Namespace": "MyNamespace",
    "Dimensions": [{
      "Name": "MyOptionalMetricDimensionName",
      "Value": "MyOptionalMetricDimensionValue"
    }],
    "Statistic": "Average",
    "Unit": "None"
  }
}
```

詳細については、Amazon EC2 Auto Scaling API リファレンス」のCustomizedMetric「仕様」を参照してください。

例 3: スケールアウトにのみターゲット追跡スケーリングポリシーを適用するには

```
aws autoscaling put-scaling-policy --policy-name alb1000-target-tracking-scaling-policy
 \
  --auto-scaling-group-name my-asg --policy-type TargetTrackingScaling \
  --target-tracking-configuration file://config.json
```

```
{
  "TargetValue": 1000.0,
  "PredefinedMetricSpecification": {
    "PredefinedMetricType": "ALBRequestCountPerTarget",
    "ResourceLabel": "app/my-alb/778d41231b141a0f/targetgroup/my-alb-target-
group/943f017f100becff"
  },
  "DisableScaleIn": true
}
```

例 4: スケールアウトにステップスケーリングポリシーを適用するには

```
aws autoscaling put-scaling-policy \
  --auto-scaling-group-name my-asg  \
  --policy-name my-step-scale-out-policy \
  --policy-type StepScaling \
  --adjustment-type PercentChangeInCapacity \
  --metric-aggregation-type Average \
  --step-adjustments
 MetricIntervalLowerBound=10.0,MetricIntervalUpperBound=20.0,ScalingAdjustment=10 \

 MetricIntervalLowerBound=20.0,MetricIntervalUpperBound=30.0,ScalingAdjustment=20 \
                     MetricIntervalLowerBound=30.0,ScalingAdjustment=30 \
  --min-adjustment-magnitude 1
```

ポリシーの Amazon リソースネーム (ARN) を記録します。 CloudWatch アラームの作成時に ARN が必要です。

例 5: スケールインにステップスケーリングポリシーを適用するには

```
aws autoscaling put-scaling-policy \
  --auto-scaling-group-name my-asg  \
  --policy-name my-step-scale-in-policy \
  --policy-type StepScaling \
  --adjustment-type ChangeInCapacity \
  --step-adjustments MetricIntervalUpperBound=0.0,ScalingAdjustment=-2
```

ポリシーの Amazon リソースネーム (ARN) を記録します。 CloudWatch アラームの作成時に ARN が必要です。

例 6: スケールアウトに単純なスケーリングポリシーを適用するには

```
aws autoscaling put-scaling-policy --policy-name my-simple-scale-out-policy \
  --auto-scaling-group-name my-asg --scaling-adjustment 30 \
  --adjustment-type PercentChangeInCapacity --min-adjustment-magnitude 2
```

ポリシーの Amazon リソースネーム (ARN) を記録します。 CloudWatch アラームの作成時に ARN が必要です。

例 7: スケールに簡易スケーリングポリシーを適用するには

```
aws autoscaling put-scaling-policy --policy-name my-simple-scale-in-policy \
  --auto-scaling-group-name my-asg --scaling-adjustment -1 \
  --adjustment-type ChangeInCapacity --cooldown 180
```

ポリシーの Amazon リソースネーム (ARN) を記録します。 CloudWatch アラームの作成時に ARN が必要です。

Amazon EC2 Auto Scaling の予測スケーリング

予測スケーリングは、過去のロードデータを分析して、トラフィックフローの日次または週次のパターンを検出します。この情報を使用して将来の容量ニーズを予測し、Amazon EC2 Auto Scaling が予想される負荷に合わせて Auto Scaling グループの容量を事前に増やすことができます。

予測スケーリングは、次のような状況に適しています。

- 通常の営業時間にはリソースの使用率が高く、夜間や週末はリソースの使用率が低いといったサイクルがあるトラフィック
- バッチ処理、テスト、定期的なデータ分析などの反復的な on-and-off ワークロードパターン
- 初期化に時間がかかり、スケールアウトイベント中のアプリケーションのパフォーマンスにレイテンシーが顕著な影響を与えるアプリケーション

一般に、トラフィックが増加する規則的なパターンがあり、アプリケーションの初期化に長い時間がかかる場合は、予測スケーリングの使用を検討する必要があります。反応的な性質を持つ動的スケーリングのみを使用する場合と比較して、予測スケーリングを使用すると、予測される負荷に先立ってキャパシティーを起動することで、より迅速にスケーリングできます。予測スケーリングは、容量を過剰にプロビジョニングする必要がなくなるため、EC2 請求書のコストを節約できる可能性もあります。

例えば、営業時間中の使用率が高く、夜間の使用量が少ないアプリケーションを考えてみましょう。各営業日の開始時に、予測スケーリングにより、トラフィックが最初に流入する前にキャパシティーを追加できます。これにより、使用率の低い期間から高い使用率の期間に移行するときに、アプリケーションの高可用性とパフォーマンスを維持するのに役立ちます。トラフィックの変化に動的スケーリングが反応するのを待つ必要はありません。また、アプリケーションの負荷パターンを確認し、スケジュールされたスケーリングを使用して適切なキャパシティーをスケジュールしようと時間を費やす必要はありません。

トピック

- 予測スケーリングの仕組み
- 予測スケーリングポリシーを作成する
- 予測スケーリングポリシーの評価
- 予定されたアクションを使用して予測値を上書きする
- カスタムメトリクスを使用した高度な予測スケーリングポリシー設定

予測スケーリングの仕組み

このトピックでは、予測スケーリングの仕組みと、予測スケーリングポリシーを作成するときに考慮すべき点について説明します。

トピック

- 仕組み
- 最大容量制限
- 考慮事項
- サポートされるリージョン

仕組み

予測スケーリングを使用するには、モニタリングおよび分析する CloudWatch メトリクスを指定する予測スケーリングポリシーを作成します。予測スケーリングが将来の値の予測を開始するには、このメトリクスに少なくとも 24 時間のデータが必要です。

ポリシーを作成すると、予測スケーリングは過去 14 日間のメトリクスデータの分析を開始し、パターンを識別します。この分析を使用して、今後 48 時間の容量要件の時間ごとの予測を生成します。予測は、最新の CloudWatch データを使用して 6 時間ごとに更新されます。新しいデータが入ってくると、予測スケーリングは将来の予測の精度を継続的に改善できます。

最初に予測スケーリングを有効にすると、予測専用モードで実行されます。このモードでは、キャパシティ予測が生成されますが、それらの予測に基づいて Auto Scaling グループを実際にスケーリングすることはありません。これにより、予測の精度と適合性を評価できます。`GetPredictiveScalingForecast` API オペレーションまたは を使用して予測データを表示できます AWS Management Console。

予測データを確認し、そのデータに基づいてスケーリングを開始することを決定したら、スケーリングポリシーを予測モードとスケーリングモードに切り替えます。このモードでは、次のようになります。

- 予測で負荷の増加が予想される場合、Amazon EC2 Auto Scaling はスケールアウトによって容量を増やします。
- 予測で負荷の減少が予想される場合、容量を削除するためにスケールインされることはありません。不要になった容量を削除する場合は、動的スケーリングポリシーを作成する必要があります。

デフォルトでは、Amazon EC2 Auto Scaling は、その時間の予測に基づいて、各時間の開始時に Auto Scaling グループをスケーリングします。オプションで、`PutScalingPolicy`API オペレーションの `SchedulingBufferTime`プロパティまたは の起動前インスタンス設定を使用して、より早い開始時間を指定できます AWS Management Console。これにより、Amazon EC2 Auto Scaling は予測された需要よりも先に新しいインスタンスを起動し、起動してトラフィックを処理する準備が整います。

予測された需要より前に新しいインスタンスを起動できるように、Auto Scaling グループのデフォルトのインスタンスウォームアップを有効にすることを強くお勧めします。これは、動的スケーリングポリシーが容量を減らす必要があることを示している場合でも、Amazon EC2 Auto Scaling がスケールインしないスケールアウトアクティビティの後の期間を指定します。これにより、スケールインオペレーションの対象となる前に、新しく起動したインスタンスが増加したトラフィックの処理を開始するのに十分な時間を確保できます。詳細については、「Auto Scaling グループに対するインスタンスのデフォルトウォームアップを設定する」を参照してください。

最大容量制限

Auto Scaling グループには、グループに対して起動できる EC2 インスタンスの最大数を制限する最大容量設定があります。デフォルトでは、スケーリングポリシーが設定されている場合、最大容量を超える容量を増やすことはできません。

または、予測容量が Auto Scaling グループの最大容量に近づいたり、超えたりした場合に、グループの最大容量を自動的に増やすことを許可することもできます。この動作を有効にするに

は、 PutScalingPolicy API オペレーションで プロパティMaxCapacityBreachBehaviorとMaxCapacityBufferプロパティを使用するか、 で最大容量動作設定を使用します AWS Management Console。

> **Warning**
>
> 最大容量を自動的に増やす場合は注意してください。これにより、最大容量の増加がモニタリングおよび管理されていない場合、意図したよりも多くのインスタンスが起動される可能性があります。増加した最大容量は、手動で更新するまで Auto Scaling グループの新しい通常の最大容量になります。最大容量は自動的に元の最大容量まで減少しません。

考慮事項

- 予測スケーリングがワークロードに適しているかどうかを確認します。ワークロードは、曜日または時刻に固有の定期的な負荷パターンを示す場合に、予測スケーリングに適しています。これを確認するには、[予測のみ] モードで予測スケーリングポリシーを設定し、コンソールの推奨事項を参照してください。Amazon EC2 Auto Scaling は、潜在的なポリシーのパフォーマンスに関する観察内容に基づいた推奨事項を提供します。予測スケーリングがアプリケーションをアクティブにスケーリングできるようにする前に、予測および推奨事項を評価します。
- 予測スケーリングでは、予測を開始するには 24 時間以上の履歴データが必要です。ただし、履歴データが 2 週間あれば予測がより効果的です。新しい Auto Scaling グループを作成し、古いグループを削除してアプリケーションを更新する場合、予測スケーリングが再び予測の生成を開始するには、新しい Auto Scaling グループに 24 時間の履歴負荷データが必要です。カスタムメトリクスを使用して新旧の Auto Scaling グループ全体のメトリクスを集計できます。そうでない場合、より正確な予測を得るために数日かかる場合があります。
- アプリケーションのフルロードを正確に表し、スケールオンが最も重要なアプリケーションの側面であるロードメトリクスを選択します。
- 予測スケーリングで動的スケーリングを使用すると、アプリケーションの需要曲線に厳密に従い、トラフィックが少ない時間帯にスケールインし、トラフィックが予想よりも多いときにスケールアウトできます。複数のスケーリングポリシーがアクティブな場合、各ポリシーによって希望するキャパシティーが個別に決定され、希望するキャパシティーはそれらの最大値に設定されます。例えば、ターゲット追跡スケーリングポリシーでターゲット使用率を維持するために 10 インスタンスが必要で、予測スケーリングポリシーでターゲット使用率を維持するために 8 つのインスタンスが必要である場合、グループの希望するキャパシティーは 10 に設定されます。動的スケーリングを初めて使用する場合は、ターゲット追跡スケーリングポリシーを使用することをお勧めします。詳細については、「Amazon EC2 Auto Scaling の動的スケーリング」を参照してください。

- 予測スケーリングの中核的な前提は、Auto Scaling グループが同種構成であり、すべてのインスタンスの容量が同じであるということです。これがグループに当てはまらない場合、予測された容量が正確ではない可能性が生じます。したがって、異なるタイプのインスタンスを不均等な容量でプロビジョニングできるため、混合インスタンスグループの予測スケーリングポリシーを作成するときは注意が必要です。以下は、予測された容量が不正確になる場合の例です。
  - 予測スケーリングポリシーが CPU 使用率に基づいているのに、各 Auto Scaling インスタンスの vCPU の数がインスタンスタイプに応じて異なる。
  - 予測スケーリングポリシーがネットワークインまたはネットワークアウトに基づいているのに、各 Auto Scaling インスタンスのネットワーク帯域幅のスループットがインスタンスタイプに応じて異なる。例えば、M5 と M5n インスタンスタイプは類似していますが、M5n インスタンスタイプの方が大幅に高いネットワークスループットを提供します。

サポートされるリージョン

Amazon EC2 Auto Scaling は、次の で予測スケーリングポリシーをサポートしています AWS リージョン。 米国東部 (バージニア北部)、 米国東部 (オハイオ)、 米国西部 (オレゴン)、 米国西部 (北カリフォルニア)、 アフリカ (ケープタウン)、 カナダ (中部)、 欧州 (フランクフルト)、 欧州 (アイルランド)、 欧州 (ロンドン)、 欧州 (ミラノ)、 欧州 (パリ)、 欧州 (ストックホルム)、 アジアパシフィック (香港)、 アジアパシフィック (ジャカルタ)、 アジアパシフィック (ムンバイ)、 アジアパシフィック (大阪)、 アジアパシフィック (東京)、 アジアパシフィック (シンガポール)、 アジアパシフィック (ソウル)、 アジアパシフィック (シドニー)、 中東 (バーレーン)、 中東 (アラブ首長国連邦)、 南米 (サンパウロ)、 中国 (北京)、 中国 (寧夏)、 AWS GovCloud （ 米国東部)、 および AWS GovCloud （ 米国西部)。

予測スケーリングポリシーを作成する

次の手順は、 AWS Management Console または を使用して予測スケーリングポリシーを作成するのに役立ちます AWS CLI。

Auto Scaling グループが新しい場合に Amazon EC2 Auto Scaling がグループの予測を生成するには、そのグループが少なくとも 24 時間分のデータを提供する必要があります。

内容

- 予測スケーリングポリシーを作成する (コンソール)
- 予測スケーリングポリシーの作成 (AWS CLI)

予測スケーリングポリシーを作成する (コンソール)

予測スケーリングポリシーを初めて作成する場合は、 コンソールを使用して、予測のみモードで複数の予測スケーリングポリシーを作成することをお勧めします。これにより、さまざまなメトリクスとターゲット値による潜在的な影響をテストできます。Auto Scaling グループごとに複数の予測スケーリングポリシーを作成できますが、アクティブなスケーリングに使用できるポリシーは 1 つだけです。

コンソールで予測スケーリングポリシーを作成する (事前定義されたメトリクス)

事前定義されたメトリクス (CPU、ネットワーク I/O、またはターゲットあたりの Application Load Balancer リクエスト数) を使用して予測スケーリングポリシーを作成するには、以下の手順を実行します。予測スケーリングポリシーを作成する最も簡単な方法は、事前定義されたメトリクスを使用することです。その代わりにカスタムメトリクスを使用する場合は、「コンソールで予測スケーリングポリシーを作成する (カスタムメトリクス)」を参照してください。

予測スケーリングポリシーを作成する

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部に分割されたペインが開きます。
3. [Automatic scaling] (自動スケーリング) タブの [Scaling policies] (スケーリングポリシー) で、[Create predictive scaling policy] (予測スケーリングポリシーの作成) を選択します。
4. ポリシーの名前を入力します。
5. Amazon EC2 Auto Scaling にすぐにスケーリングを開始させるには、[Scale based on forecast] (予測に基づくスケーリング) をオンにします。

   ポリシーを予測のみモードにしておくには、[Scale based on forecast] (予測に基づくスケーリング) をオフのままにします。
6. [Metrics] (メトリクス) で、オプションのリストからメトリクスを選択します。オプションには、[CPU]、[Network In] (ネットワーク入力)、[Network Out] (ネットワーク出力)、[Application Load Balancer request count] (Application Load Balancer リクエスト数)、および [Custom metric pair] (カスタムメトリクスペア) が含まれます。

   [Application Load Balancer request count per target] (ターゲットあたりの Application Load Balancer リクエスト数) を選択した場合、[Target group] (ターゲットグループ) のターゲットグ

ループを選択します。[Application Load Balancer request count per target] (ターゲットあたりの Application Load Balancer リクエスト数) は、Application Load Balancer ターゲットグループを Auto Scaling グループにアタッチしている場合にのみサポートされます。

[Custom metric pair] (カスタムメトリクスペア) を選択した場合、[Load metric] (負荷のメトリクス) と [Scaling metric] (スケーリングのメトリクス) のドロップダウンリストから個々のメトリクスを選択します。

7. [Target utilization] (ターゲット使用率) に、Amazon EC2 Auto Scaling が維持する必要があるターゲット値を入力します。Amazon EC2 Auto Scaling は、平均使用率がターゲット使用率になるまで、または指定したインスタンスの最大数に達するまで、キャパシティーをスケールアウトします。

スケーリングメトリクスが以下である場合..。	ターゲット使用率は以下の内容になります。
CPU	各インスタンスが使用する CPU の理想的な割合。
ネットワーク入力	各インスタンスが受信する理想的な 1 分あたりの平均バイト数。
ネットワーク出力	各インスタンスが送信する理想的な 1 分あたりの平均バイト数。
ターゲットあたりの Application Load Balancer リクエスト数	各インスタンスが受信する理想的な 1 分あたりの平均リクエスト数。

8. (オプション) [Pre-launch instances] (インスタンスの事前起動) で、予測によって負荷の増加が必要とされる際、どれくらい前にインスタンスを起動しておくかを選択します。
9. (オプション) [Max capacity behavior] (最大容量の動作) で、予測されたキャパシティーが定義された最大値を超える場合に、Amazon EC2 Auto Scaling がグループの最大容量を超えてスケールアウトできるようにするかどうかを選択します。この設定をオンにすると、トラフィックが最高になると予測される期間中にスケールアウトが実行されます。
10. (オプション) [Buffer maximum capacity above the forecasted capacity] (予測キャパシティーを超える最大キャパシティーのバッファ) で、予測キャパシティーが最大キャパシティーに近づいたか、それを超えたときに使用する追加キャパシティーを選択します。この値は予測キャパシティーに対する割合 (%) として指定されます。たとえば、バッファが 10 の場合、バッファは

10% です。従って、予測キャパシティーが 50 で最大キャパシティーが 40 の場合、有効な最大キャパシティーは 55 です。

これを 0 に設定すると、Amazon EC2 Auto Scaling は最大キャパシティーを超えてスケールすることはできても、予測キャパシティーまでとなり、それを超えることはできません。

11. [Create predictive scaling policy] (予測スケーリングポリシーを作成) を選択します。

コンソールで予測スケーリングポリシーを作成する (カスタムメトリクス)

カスタムメトリクスを使用して予測スケーリングポリシーを作成するには、以下の手順を実行します。カスタムメトリクスには、 が提供する他のメトリクス CloudWatch や、 に発行するメトリクスを含めることができます CloudWatch。CPU、ネットワーク I/O、またはターゲットあたりの Application Load Balancer リクエスト数を使用するには、「コンソールで予測スケーリングポリシーを作成する (事前定義されたメトリクス)」を参照してください。

カスタムメトリクスを使用して予測スケーリングポリシーを作成するには、以下を実行する必要があります。

- Amazon EC2 Auto Scaling が のメトリクスとやり取りできるようにする raw クエリを指定する必要があります CloudWatch。詳細については、「カスタムメトリクスを使用した高度な予測スケーリングポリシー設定」を参照してください。Amazon EC2 Auto Scaling が からメトリクスデータを抽出できるようにするには CloudWatch、各クエリがデータポイントを返すことを確認します。コンソール CloudWatch または API オペレーションを使用してこれを確認します CloudWatch GetMetricData。

  > **Note**
  >
  > Amazon EC2 Auto Scaling コンソールの JSON エディタには、サンプル JSON ペイロードが提供されています。これらの例では、 が提供する他の CloudWatch メトリクス AWS や、以前に に公開したメトリクスを追加するために必要なキーと値のペアのリファレンスを提供します CloudWatch。これらを開始点として使用してから、必要に応じてカスタマイズすることができます。

- メトリクス計算を使用する場合は、独自のシナリオに適した JSON を手動で構築する必要があります。詳細については、「Metric Math 式を使用する」を参照してください。ポリシーでメトリクス計算を使用する前に、メトリクス数式に基づくメトリクスクエリが有効であり、単一の時系列を返すことを確認してください。コンソール CloudWatch または API オペレーションを使用してこれを確認します CloudWatch GetMetricData。

誤ったデータ (間違った Auto Scaling グループ名など) を提供することによってクエリでエラーが発生した場合、予測にはデータがありません。カスタムメトリクス問題のトラブルシューティングについては、「考慮事項とトラブルシューティング」を参照してください。

予測スケーリングポリシーを作成する

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部に分割されたペインが開きます。
3. [Automatic scaling] (自動スケーリング) タブの [Scaling policies] (スケーリングポリシー) で、[Create predictive scaling policy] (予測スケーリングポリシーの作成) を選択します。
4. ポリシーの名前を入力します。
5. Amazon EC2 Auto Scaling にすぐにスケーリングを開始させるには、[Scale based on forecast] (予測に基づくスケーリング) をオンにします。

   ポリシーを予測のみモードにしておくには、[Scale based on forecast] (予測に基づくスケーリング) をオフのままにします。
6. [Metrics] (メトリクス) では、[Custom metric pair] (カスタムメトリクスのペア) を選択します。

   a. ロードメトリクス で、カスタム CloudWatch メトリクスを使用するカスタムメトリクスを選択します。ポリシーのロードメトリクス定義が含まれる JSON ペイロードを構築し、それを JSON エディタボックスに貼り付けて、ボックス内にすでに入力されているペイロードを置き換えます。

   b. スケーリングメトリクス で、カスタム CloudWatch メトリクスを使用するカスタムメトリクスを選択します。ポリシーのスケーリングメトリクス定義が含まれる JSON ペイロードを構築し、それを JSON エディタボックスに貼り付けて、ボックス内にすでに入力されているペイロードを置き換えます。

   c. (オプション) カスタム容量メトリクスを追加するには、[Add custom capacity metric] (カスタム容量メトリクスを追加する) のチェックボックスをオンにします。ポリシーの容量メトリクス定義が含まれる JSON ペイロードを構築し、それを JSON エディタボックスに貼り付けて、ボックス内にすでに入力されているペイロードを置き換えます。

      このオプションを有効にする必要があるのは、容量メトリクスデータが複数の Auto Scaling グループにまたがる場合に容量の新しい時系列を作成するためのみです。この場合は、メトリクス計算を使用してデータを単一の時系列に集計する必要があります。

7. [Target utilization] (ターゲット使用率) に、Amazon EC2 Auto Scaling が維持する必要があるターゲット値を入力します。Amazon EC2 Auto Scaling は、平均使用率がターゲット使用率になるまで、または指定したインスタンスの最大数に達するまで、キャパシティーをスケールアウトします。
8. (オプション) [Pre-launch instances] (インスタンスの事前起動) で、予測によって負荷の増加が必要とされる際、どれくらい前にインスタンスを起動しておくかを選択します。
9. (オプション) [Max capacity behavior] (最大容量の動作) で、予測されたキャパシティーが定義された最大値を超える場合に、Amazon EC2 Auto Scaling がグループの最大容量を超えてスケールアウトできるようにするかどうかを選択します。この設定をオンにすると、トラフィックが最高になると予測される期間中にスケールアウトが実行されます。
10. (オプション) [Buffer maximum capacity above the forecasted capacity] (予測キャパシティーを超える最大キャパシティーのバッファ) で、予測キャパシティーが最大キャパシティーに近づいたか、それを超えたときに使用する追加キャパシティーを選択します。この値は予測キャパシティーに対する割合 (%) として指定されます。たとえば、バッファが 10 の場合、バッファは 10% です。従って、予測キャパシティーが 50 で最大キャパシティーが 40 の場合、有効な最大キャパシティーは 55 です。

    これを 0 に設定すると、Amazon EC2 Auto Scaling は最大キャパシティーを超えてスケールすることはできても、予測キャパシティーまでとなり、それを超えることはできません。
11. [Create predictive scaling policy] (予測スケーリングポリシーを作成) を選択します。

予測スケーリングポリシーの作成 (AWS CLI)

AWS CLI 次のように を使用して、Auto Scaling グループの予測スケーリングポリシーを設定します。各##############を独自の情報に置き換えます。

指定できる CloudWatch メトリクスの詳細については、Amazon EC2 Auto Scaling API リファレンスPredictiveScalingMetricSpecification」の「」を参照してください。

例 1: 予測を作成するが、スケーリングしない予測スケーリングポリシー

次のポリシー例では、予測スケーリングに CPU 使用率メトリクスを使用し、ターゲット使用率が 40 である完全なポリシー設定を示しています。使用するモードを明示的に指定しない限り、デフォルトで ForecastOnly モードが使用されます。この設定を config.json という名前のファイルに保存してください。

```
{
    "MetricSpecifications": [
```

```
        {
            "TargetValue": 40,
            "PredefinedMetricPairSpecification": {
                "PredefinedMetricType": "ASGCPUUtilization"
            }
        }
    ]
}
```

コマンドラインからポリシーを作成するには、次の例に示すように、 設定ファイルを指定して put-scaling-policy コマンドを実行します。

```
aws autoscaling put-scaling-policy --policy-name cpu40-predictive-scaling-policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
```

成功した場合、このコマンドはポリシーの Amazon リソースネーム (ARN) を返します。

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/cpu40-predictive-scaling-
policy",
  "Alarms": []
}
```

例 2: 予測とスケーリングを行う予測スケーリングポリシー

Amazon EC2 Auto Scaling の予測およびスケーリングを許可するポリシーには、Mode プロパティを ForecastAndScale の値で追加します。次の例は、Application Load Balancer リクエスト数メトリクスを使用するポリシー設定を示しています。ターゲット使用率は 1000 で、予測スケーリングは ForecastAndScale モードに設定されています。

```
{
    "MetricSpecifications": [
        {
            "TargetValue": 1000,
            "PredefinedMetricPairSpecification": {
                "PredefinedMetricType": "ALBRequestCount",
                "ResourceLabel": "app/my-alb/778d41231b141a0f/targetgroup/my-alb-
target-group/943f017f100becff"
            }
        }
```

```
    ],
    "Mode": "ForecastAndScale"
}
```

このポリシーを作成するには、次の例に示すように、 設定ファイルを指定して put-scaling-policy コマンドを実行します。

```
aws autoscaling put-scaling-policy --policy-name alb1000-predictive-scaling-policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
```

成功した場合、このコマンドはポリシーの Amazon リソースネーム (ARN) を返します。

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-
id:scalingPolicy:19556d63-7914-4997-8c81-d27ca5241386:autoScalingGroupName/my-
asg:policyName/alb1000-predictive-scaling-policy",
  "Alarms": []
}
```

例 3: 最大キャパシティーを超えてスケーリングできる予測スケーリングポリシー

次の例は、通常よりも高い負荷を処理する必要がある場合に、グループの最大サイズ制限を超えてスケーリングできるポリシーを作成する方法を示しています。デフォルトでは、Amazon EC2 Auto Scaling は、定義した最大キャパシティーを超えて EC2 のキャパシティーをスケーリングしません。しかし、パフォーマンスや可用性の問題を回避するために、キャパシティーを少し増やしてスケーリングすると便利な場合があります。

キャパシティーがグループの最大サイズに達している、または非常に近いと予測されるときに、Amazon EC2 Auto Scaling が追加のキャパシティーをプロビジョニングする余地を提供するには、次の例に示すように、MaxCapacityBreachBehavior および MaxCapacityBuffer プロパティを指定します。MaxCapacityBreachBehavior に値 IncreaseMaxCapacity を指定する必要があります。グループに含めることができるインスタンスの最大数は、MaxCapacityBuffer の値によって異なります。

```
{
    "MetricSpecifications": [
        {
            "TargetValue": 70,
            "PredefinedMetricPairSpecification": {
                "PredefinedMetricType": "ASGCPUUtilization"
```

```
            }
        }
    ],
    "MaxCapacityBreachBehavior": "IncreaseMaxCapacity",
    "MaxCapacityBuffer": 10
}
```

この例では、10% のバッファ ("MaxCapacityBuffer": 10) を使用するようにポリシーが設定されています。したがって、予測キャパシティーが 50、最大キャパシティーが 40 の場合、有効な最大キャパシティーは 55 です。キャパシティーを最大キャパシティーを超えてスケーリングし、予測キャパシティーに等しくするが、予測キャパシティーを超えないようにするポリシーでは、バッファは 0 です ("MaxCapacityBuffer": 0)。

このポリシーを作成するには、次の例に示すように、 設定ファイルを指定して put-scaling-policy コマンドを実行します。

```
aws autoscaling put-scaling-policy --policy-name cpu70-predictive-scaling-policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
```

成功した場合、このコマンドはポリシーの Amazon リソースネーム (ARN) を返します。

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:d02ef525-8651-4314-
bf14-888331ebd04f:autoScalingGroupName/my-asg:policyName/cpu70-predictive-scaling-
policy",
  "Alarms": []
}
```

予測スケーリングポリシーの評価

予測スケーリングポリシーを使用して Auto Scaling グループをスケーリングする前に、Amazon EC2 Auto Scaling コンソールでポリシーの推奨事項やその他のデータを確認します。予測が正確であることがわかるまで、予測スケーリングポリシーが実際のキャパシティをスケーリングすることが好ましくないため、これは重要です。

Auto Scaling グループが新しい場合、Amazon EC2 Auto Scaling が最初の予測を作成するために 24 時間かかります。

Amazon EC2 Auto Scaling が予測を作成するとき、履歴データを使用します。Auto Scaling グループにまだ最新の履歴データがない場合、Amazon EC2 Auto Scaling は現在利用可能な履歴集計から作

成された集計で予測を一時的にバックフィルすることがあります。予測は、ポリシーの作成日より最大 2 週間前にバックフィルされます。

内容

- 予測スケーリングの推奨事項の表示
- 予測スケーリングのモニタリンググラフの確認
- による予測スケーリングメトリクスのモニタリング CloudWatch

予測スケーリングの推奨事項の表示

分析を効果的に行うには、Amazon EC2 Auto Scaling に比較対象となる予測スケーリングポリシーが少なくとも 2 つ必要です。(ただし、単一ポリシーの結果を確認することはできます) 複数のポリシーを作成するとき、1 つのメトリクスを使用するポリシーを異なるメトリクスを使用するポリシーと比較して評価できます。異なる目標値およびメトリクスの組み合わせによる影響を評価することもできます。予測スケーリングポリシーが作成された後、Amazon EC2 Auto Scaling は、どのポリシーがグループのスケーリングに適しているかについて、すぐに評価を開始します。

Amazon EC2 Auto Scaling コンソールで推奨事項を表示するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。
3. [予測スケーリングポリシー] の [Auto Scaling] タブで、ポリシーの詳細および推奨事項を表示できます。推奨事項は、予測スケーリングポリシーを使用しない場合よりも優れた性能を発揮するかどうかについて説明します。

   予測スケーリングポリシーがグループに適切かどうかが不明な場合、[可用性への影響] 列および [コストへの影響] 列を確認し、適切なポリシーを選択してください。各列の情報は、ポリシーの影響について説明します。

   - [可用性への影響]: ポリシーを使用しない場合と比較し、ワークロードを処理するために十分なインスタンスをプロビジョニングすることにより、ポリシーが可用性への悪影響を回避できるかどうかについて説明します。
   - [コストへの影響]: ポリシーを使用しない場合と比較し、インスタンスをオーバープロビジョニングしないことにより、ポリシーがコストへの悪影響を回避できるかどうかについて説明

します。過度にオーバープロビジョニングすることにより、インスタンスが十分に活用されない、あるいはアイドル状態になり、コストへの影響が増す一方です。

複数のポリシーがある場合、より低コストで最も可用性のメリットが高いポリシーの名前の横に [最良予測] タグが表示されます。可用性への影響がより重視されます。

4. (オプション) 推奨結果の必要な期間を選択するには、[評価期間] のドロップダウンから [2 日]、[1 週間]、[2 週間]、[4 週間]、[6 週間]、[8 週間] のいずれから希望する値を選択します。デフォルトでは、評価期間は過去の 2 週間です。評価期間が長いほど、推奨結果のデータポイントが増えます。ただし、需要が非常に高い時期の後など、負荷パターンが変化した場合、データポイントを追加しても結果が改善されない可能性があります。この場合、最新のデータを見ることでより焦点を絞った推奨事項を得ることができます。

> **Note**
>
> 推奨事項は [予測のみ] モードのポリシーに対してのみ生成されます。推奨機能は、評価期間中にポリシーが [予測のみ] モードのときにより効果的に機能します。ポリシーを [予測とスケーリング] モードで開始し、後で [予測のみ] モードに切り替える場合、そのポリシーの結果に偏りが生じる可能性があります。これは、ポリシーが既に実際のキャパシティに関与しているためです。

予測スケーリングのモニタリンググラフの確認

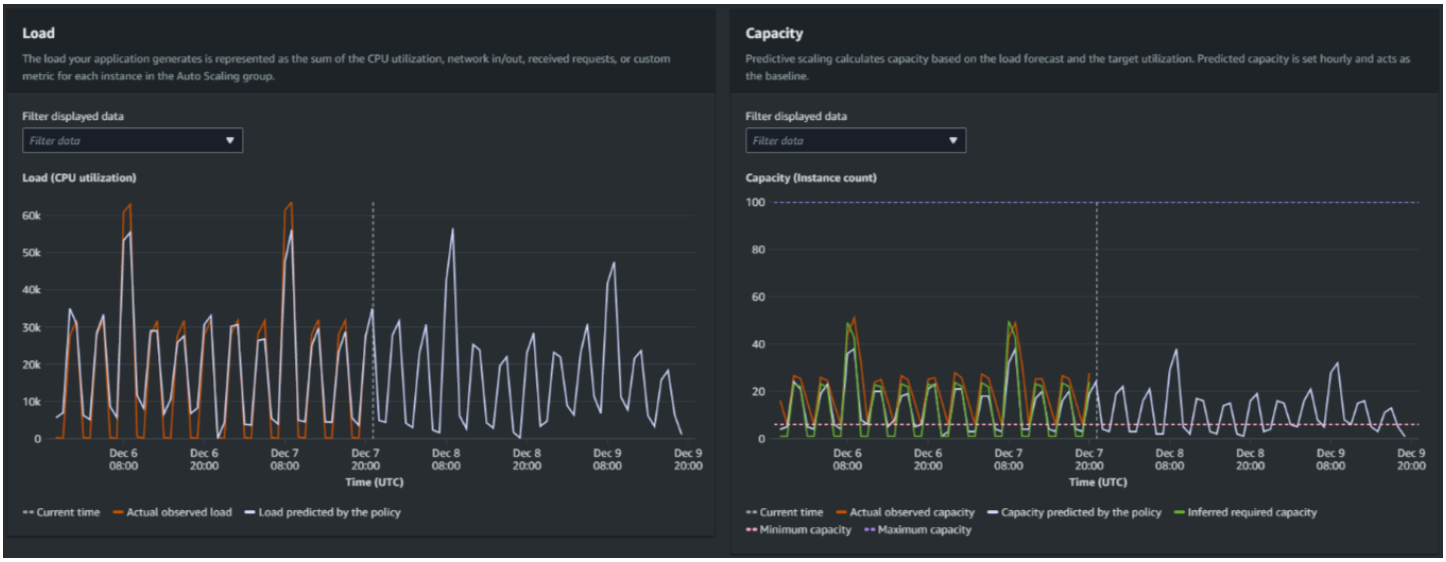
Amazon EC2 Auto Scaling コンソールでは、以前の日、週間、月の予測を確認し、時間の経過とともにポリシーがどの程度機能しているか視覚化できます。ポリシーが実際のキャパシティをスケーリングするかどうかを決定するとき、この情報を使用して予測の精度を評価できます。

Amazon EC2 Auto Scaling コンソールで予測スケーリングのモニタリンググラフを表示するには

1. [予測スケーリングポリシー] リストからポリシーを選択します。
2. [モニタリング] セクションでは、ポリシーの負荷およびキャパシティに関する過去および今後の予測を実際の値と比較できます。[負荷] グラフには、選択した負荷メトリクスの負荷予測および実際の値が表示されます。[キャパシティ] グラフには、ポリシーによって予測されたインスタンスの数が表示されます。実際に起動されたインスタンスの数も含まれます。縦線は履歴の値と今後の予測を区切っています。これらのグラフは、ポリシーの作成後にすぐ利用できます。

3. (オプション) グラフに表示される履歴データの量を変更するには、ページ上部の [評価期間] のドロップダウンから希望する値を選択します。評価期間はこのページのデータをはいかなる方法で変換することはありません。表示される履歴データの量のみを変更します。

次の画像は、予測が複数回適用された場合の [負荷] グラフおよび [キャパシティ] グラフを表示しています。予測スケーリングは、履歴の負荷データに基づいて負荷を予測します。アプリケーションが生成する負荷は、Auto Scaling グループの各インスタンスの CPU 使用率、ネットワークの入出力、受信したリクエスト、カスタムメトリクスのいずれかの合計として表されます。予測スケーリングは、負荷予測およびスケーリングメトリクスで達成したい目標の使用率に基き、今後のキャパシティのニーズを計算します。



[負荷] グラフのデータを比較

各水平線は、1 時間間隔で報告される異なる一連のデータポイントを表しています。

1. [実際に観測された負荷] は、選択した負荷メトリクスの SUM 統計を使用し、過去の 1 時間ごとの合計負荷を表示します。
2. [ポリシーによって予測される負荷] は、1 時間ごとの負荷予測を表示します。この予測は過去 2 週間分の実際の負荷観測に基づいています。

[キャパシティ] グラフのデータの比較

各水平線は、1 時間間隔で報告される異なる一連のデータポイントを表しています。

1. [実際に観測されたキャパシティ] には、Auto Scaling グループの過去の実際のキャパシティが表示されます。これは、他のスケーリングポリシーや、選択した期間に有効な最小グループサイズによって異なります。
2. [ポリシーによって予測されるキャパシティ] には、ポリシーが [予測とスケーリング] モードになっているときに各時間の開始時に予想されるベースラインキャパシティが表示されます。
3. [推定必要容量は] には、スケーリング指標を選択した目標値に維持するための理想的な容量を表示します。
4. [最小容量] には、Auto Scaling グループの最小容量を表示します。
5. [最大容量] には、Auto Scaling グループの最大容量を表示します。

推定される必要キャパシティを計算するため、最初は各インスタンスが指定された目標値で均等に使用されていると仮定します。実際には、インスタンスは均等に使用されません。ただし、使用率がインスタンス間で均等に分散されていると仮定することにより、必要なキャパシティの量を推定できます。次に、キャパシティ要件は、予測スケーリングポリシーに使用したスケーリングメトリクスに反比例するように計算されます。つまり、キャパシティが増加するにつれ、スケーリングメトリクスは同じ割合で減少します。例えば、キャパシティが 2 倍になった場合、スケーリングメトリクスは半減します。

推定された必要キャパシティの計算式は、次のとおりです。

```
sum of (actualCapacityUnits*scalingMetricValue)/(targetUtilization)
```

例えば、特定の時間の `actualCapacityUnits` (`10`) および `scalingMetricValue` (`30`) を算出します。その後、予測スケーリングポリシー (`60`) で指定した `targetUtilization` を使用し、同じ時間に推定される必要キャパシティを計算します。これは `5` の値を返します。これは、スケーリングメトリクスの目標値とは正反比例し、キャパシティを維持するために必要なキャパシティの推定量は 5 であることを意味します。

> **Note**
>
> コスト削減およびアプリケーションの可用性を調整および改善するため、さまざまな手段が用意されています。
>
> - ベースラインキャパシティに予測スケーリングを使用し、追加のキャパシティに動的スケーリングを使用します。動的スケーリングは予測スケーリングとは独立して動作し、現在の使用率に基づいてスケールインおよびスケールアウトを行います。まず、Amazon EC2 Auto Scaling は、動的スケーリングポリシーごとに推奨されるインスタンスの数を計

算します。次に、最も多くのインスタンスを提供するポリシーに基づいてスケーリングします。

- 負荷が減少したときにスケールインを実行できるようにするには、Auto Scaling グループには、スケールイン部分を有効にした動的スケーリングポリシーが常に少なくとも 1 つ必要です。
- 最小キャパシティおよび最大キャパシティを制限しすぎないようにすることにより、スケーリングパフォーマンスを向上させることができます。推奨されるインスタンスの数が最小キャパシティおよび最大キャパシティの範囲に収まらないポリシーは、スケールインおよびスケールアウトができなくなります。

による予測スケーリングメトリクスのモニタリング CloudWatch

必要に応じて、Amazon EC2 Auto Scaling コンソール CloudWatch ではなく、Amazon から予測スケーリングのモニタリングデータにアクセスすることもできます。 Amazon EC2 予測スケーリングポリシーを作成すると、ポリシーは、今後の負荷とキャパシティを予測するために使用するデータを収集します。このデータが収集されると、定期的に に自動的に保存されます CloudWatch 。その後、 を使用して CloudWatch 、ポリシーが時間の経過とともにどの程度うまく機能するかを視覚化できます。また、パフォーマンス指標が で定義した制限を超えて変化したときに通知する CloudWatch アラームを作成することもできます CloudWatch。

トピック

- 履歴予測データの視覚化
- Metric Math を使用して精度メトリクスを作成する

履歴予測データの視覚化

で予測スケーリングポリシーの負荷と容量の予測データを表示できます CloudWatch。これは、1 つのグラフで他の CloudWatchメトリクスに対して予測を視覚化する場合に役立ちます。また、経時的な傾向を確認するために、より長い期間を表示することもできます。最大 15 か月間の履歴メトリクスにアクセスして、ポリシーの動作をより的確に把握できます。

詳細については、「予測スケーリングのメトリクスとディメンション」を参照してください。

CloudWatch コンソールを使用して過去の予測データを表示するには

1. https://console.aws.amazon.com/cloudwatch/ で CloudWatch コンソールを開きます。

2. ナビゲーションペインで、[Metrics] (メトリクス)、[All metrics] (すべてのメトリクス) の順に選択します。
3. [Auto Scaling] メトリクス名前空間を選択します。
4. 以下のオプションのいずれかを選択して、負荷予測またはキャパシティ予測メトリクスのいずれかを表示します。

   - 予測スケーリングの負荷予測
   - 予測スケーリングのキャパシティ予測

5. 検索フィールドに、予測スケーリングポリシー名または Auto Scaling グループ名を入力し、Enter キーを押して結果をフィルタリングします。
6. メトリクスをグラフ表示するには、メトリクスの横にあるチェックボックスを選択します。グラフの名前を変更するには、鉛筆アイコンを選択します。時間範囲を変更するには、事前定義済みの値を選択するか、[custom] を選択します。詳細については、「Amazon CloudWatch ユーザーガイド」の「メトリクスのグラフ化」を参照してください。
7. 統計を変更するには、[Graphed metrics] タブを選択します。列見出しまたは個々の値を選択し、続いて各種統計を選択します。メトリクスごとに任意の統計を選択できますが、すべての統計が PredictiveScalingLoadForecastおよび PredictiveScalingCapacityForecastメトリクスに役立つわけではありません。例えば、平均、最小、最大統計は有用ですが、合計統計は有用ではありません。
8. グラフに別のメトリクスを追加するには、[Browse] (参照) で [All] (すべて) を選択し、追加したいメトリクスを見つけて、その横にあるチェックボックスをオンにします。最大 10 個のメトリクスを追加できます。

   例えば、CPU 使用率の実際の値をグラフに追加するには、[EC2] 名前空間、[By Auto Scaling Group] (Auto Scaling グループ別) の順に選択します。次に、[CPUUtilization] メトリクス、および対象とする Auto Scaling グループのチェックボックスをオンにします。
9. (オプション) グラフを CloudWatch ダッシュボードに追加するには、アクション 、ダッシュボードに追加する を選択します。

Metric Math を使用して精度メトリクスを作成する

Metric Math を使用すると、複数の CloudWatch メトリクスをクエリし、数式を使用して、これらのメトリクスに基づいて新しい時系列を作成できます。作成された時系列を CloudWatch コンソールで視覚化し、ダッシュボードに追加できます。Metric Math の詳細については、「Amazon ユーザーガイド」の「Metric Math の使用」を参照してください。 CloudWatch

Metric Math を使用して、Amazon EC2 Auto Scaling が予測スケーリングのために生成するデータを各種の方法でグラフ化できます。これにより、ポリシーのパフォーマンスを経時的にモニタリングし、メトリクスの組み合わせを改善できるかどうかを把握することができます。

例えば、Metric Math 式を使用して、平均絶対パーセント誤差 (MAPE) をモニタリングできます。MAPE メトリクスは、予測値と、特定の予測期間中に観測された実際の値の差をモニタリングするのに役立ちます。MAPE の値の変化は、アプリケーションの性質が変化するにつれて、ポリシーのパフォーマンスが経時的に低下しているかどうかを示します。MAPE の増加は、予測値と実際の値の差が大きいことを示します。

例: Metric Math 式

このタイプのグラフを使用するには、次の例に示すような Metric Math 式を作成します。

```
{
  "MetricDataQueries": [
    {
      "Expression": "TIME_SERIES(AVG(ABS(m1-m2)/m1))",
      "Id": "e1",
      "Period": 3600,
      "Label": "MeanAbsolutePercentageError",
      "ReturnData": true
    },
    {
      "Id": "m1",
      "Label": "ActualLoadValues",
      "MetricStat": {
        "Metric": {
          "Namespace": "AWS/EC2",
          "MetricName": "CPUUtilization",
          "Dimensions": [
            {
              "Name": "AutoScalingGroupName",
              "Value": "my-asg"
            }
          ]
        },
        "Period": 3600,
        "Stat": "Sum"
      },
      "ReturnData": false
    },
    {
```

```
      "Id": "m2",
      "Label": "ForecastedLoadValues",
      "MetricStat": {
        "Metric": {
          "Namespace": "AWS/AutoScaling",
          "MetricName": "PredictiveScalingLoadForecast",
          "Dimensions": [
            {
              "Name": "AutoScalingGroupName",
              "Value": "my-asg"
            },
            {
              "Name": "PolicyName",
              "Value": "my-predictive-scaling-policy"
            },
            {
              "Name": "PairIndex",
              "Value": "0"
            }
          ]
        },
        "Period": 3600,
        "Stat": "Average"
      },
      "ReturnData": false
    }
  ]
}
```

単一のメトリクスではなく、MetricDataQueries 用のメトリクスデータクエリ構造の配列があります。MetricDataQueries の各項目は、メトリクスを取得するか、数式を実行します。最初の項目は、数式である e1 です。指定された式は、ReturnData パラメータを true に設定し、最終的に単一の時系列を生成します。他のすべてのメトリクスで、ReturnData 値は false です。

この例では、指定された式は実際の値と予測値を入力として使用し、新しいメトリクス (MAPE) を返します。 m1は CloudWatch実際の負荷値を含むメトリクスです (CPU 使用率は、 という名前のポリシーで最初に指定された負荷メトリクスであると仮定しますmy-predictive-scaling-policy) 。 m2は CloudWatch、予測された負荷値を含むメトリクスです。MAPE メトリクスの計算構文は次のとおりです。

(絶対値 ((実際の値 - 予測値)/(実際の値))) の平均

精度メトリクスを視覚化してアラームを設定する

精度メトリクスデータを視覚化するには、 CloudWatch コンソールでメトリクスタブを選択します。そこからデータをグラフ化できます。詳細については、「Amazon ユーザーガイド」の CloudWatch「グラフへの数式の追加」を参照してください。 CloudWatch

[Metrics] (メトリクス) セクションから、モニタリングしているメトリクスにアラームを設定することもできます。[Graphed metrics] (グラフ化したメトリクス) タブで、[Actions] (アクション) 列にある [Create alarm] (アラームを作成) アイコンをクリックします。[Create alarm] (アラームを作成) アイコンは小さなベルです。詳細と通知オプションについては、「Amazon CloudWatch ユーザーガイド」の「メトリクスの数式に基づく CloudWatch アラームの作成」および「アラームの変更に関するユーザーへの通知」を参照してください。

または、 GetMetricDataと PutMetricAlarmを使用して Metric Math を使用して計算を実行し、出力に基づいてアラームを作成することもできます。

予定されたアクションを使用して予測値を上書きする

予測計算では考慮できない将来のアプリケーション要件に関する追加情報がある場合があります。例えば、予測の計算では、今後のマーケティングイベントに必要なキャパシティーが過小評価される可能性があります。スケジュールされたアクションを使用して、将来の期間中の予測を一時的に上書きできます。スケジュールされたアクションは、繰り返し実行することも、1 回限りの需要変動がある特定の日時に実行することもできます。

例えば、予測されるキャパシティーを超える最小キャパシティーでスケジュールされたアクションを作成できます。実行中に、Amazon EC2 Auto Scaling は Auto Scaling グループの最小キャパシティーを更新します。予測スケーリングはキャパシティーを最適化するので、予測値を超える最小キャパシティーでスケジュールされたアクションが適用されます。これにより、キャパシティーが想定より少なくなるのを防ぎます。予測の上書きを停止するには、2 番目のスケジュールされたアクションを使用して、最小キャパシティーを元の設定に戻します。

次の手順では、将来の期間中の予測を上書きするステップを示します。

トピック

- ステップ 1: (オプション) 時系列データを分析する
- ステップ 2: 2 つのスケジュールされたアクションを作成する

> **Important**
>
> このトピックでは、予測を上書きして、予測よりも大きい容量にスケーリングしようとしていることを前提としています。予測スケーリングポリシーの干渉なしに一時的に容量を減らす必要がある場合は、代わりに予測専用モードを使用します。予測専用モードでは、予測スケーリングは引き続き予測を生成しますが、キャパシティーは自動的に増加しません。その後、リソース使用率をモニタリングし、必要に応じてグループのサイズを手動で減らすことができます。手動でのスケーリングの詳細については、「」を参照してくださいAmazon EC2 Auto Scaling の手動スケーリング。

ステップ 1: (オプション) 時系列データを分析する

まず、予測時系列データを分析します。これはオプションのステップですが、予測の詳細を理解したい場合に役立ちます。

1. 予測を取得する

   予測が作成されたら、予測の特定の期間をクエリできます。このクエリの目的は、特定の期間の時系列データの完全なビューを取得することです。

   クエリには、将来の予測データを最大 2 日間含めることができます。予測スケーリングをしばらく使用している場合は、過去の予測データにアクセスすることもできます。ただし、開始時刻と終了時刻の間の最大期間は 30 日間です。

   get-predictive-scaling-forecast AWS CLI コマンドを使用して予測を取得するには、 コマンドで次のパラメータを指定します。

   - Auto Scaling グループの名前を `--auto-scaling-group-name` パラメータに入力します。
   - ポリシーの名前を `--policy-name` パラメータに入力します。
   - 開始時刻を `--start-time` パラメータに入力して、指定した時刻以降の予測データのみが返されるようにします。
   - 終了時刻を `--end-time` パラメータに入力して、指定された時刻より前の予測データのみが返されるようにします。

```
aws autoscaling get-predictive-scaling-forecast --auto-scaling-group-name my-asg \
    --policy-name cpu40-predictive-scaling-policy \
    --start-time "2021-05-19T17:00:00Z" \
```

```
    --end-time "2021-05-19T23:00:00Z"
```

成功すると、コマンドは次の例のようなデータを返します。

```
{
    "LoadForecast": [
        {
            "Timestamps": [
                "2021-05-19T17:00:00+00:00",
                "2021-05-19T18:00:00+00:00",
                "2021-05-19T19:00:00+00:00",
                "2021-05-19T20:00:00+00:00",
                "2021-05-19T21:00:00+00:00",
                "2021-05-19T22:00:00+00:00",
                "2021-05-19T23:00:00+00:00"
            ],
            "Values": [
                153.0655799339254,
                128.8288551285919,
                107.1179447150675,
                197.3601844551528,
                626.4039934516954,
                596.9441277518481,
                677.9675713779869
            ],
            "MetricSpecification": {
                "TargetValue": 40.0,
                "PredefinedMetricPairSpecification": {
                    "PredefinedMetricType": "ASGCPUUtilization"
                }
            }
        }
    ],
    "CapacityForecast": {
        "Timestamps": [
            "2021-05-19T17:00:00+00:00",
            "2021-05-19T18:00:00+00:00",
            "2021-05-19T19:00:00+00:00",
            "2021-05-19T20:00:00+00:00",
            "2021-05-19T21:00:00+00:00",
            "2021-05-19T22:00:00+00:00",
            "2021-05-19T23:00:00+00:00"
        ],
```

```
        "Values": [
            2.0,
            2.0,
            2.0,
            2.0,
            4.0,
            4.0,
            4.0
        ]
    },
    "UpdateTime": "2021-05-19T01:52:50.118000+00:00"
}
```

応答には LoadForecast と CapacityForecast の 2 つの予測が含まれています。LoadForecast は、時間ごとの負荷予測を示します。CapacityForecast は、40.0 の TargetValue (平均 CPU 使用率 40%) を維持しながら予測された負荷を処理するために時間単位で必要なキャパシティーの予測値を示します。

2. ターゲット期間を特定する

   1 回限りの需要変動が発生する時間または時間範囲を特定します。予測に表示される日付と時刻は UTC であることに注意してください。

ステップ 2: 2 つのスケジュールされたアクションを作成する

次に、アプリケーションの負荷が予測を上回る特定の期間に、2 つのスケジュールされたアクションを作成します。例えば、マーケティングイベントで一時的にトラフィックがサイトに流入する場合は、1 回限りのアクションをスケジュールして、開始時に最小キャパシティーを更新できます。次に、イベント終了時に最小キャパシティーを元の設定に戻す別のアクションをスケジュールします。

1 回限りのイベントに対して 2 つのスケジュールされたアクションを作成するには (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。
3. [Automatic scaling (自動スケーリング)] タブの [Scheduled actions (スケジュールされたアクション)] で、[Create scheduled action (スケジュールされたアクションの作成)] を選択します。
4. 次のスケジュールされたアクション設定を入力します。

a. スケジュールされたアクションに [Name (名前)] を入力します。
b. [Min] (最小) に、Auto Scaling グループの新しい最小キャパシティーを入力します。[Min] (最小) は、グループの最大サイズ以下である必要があります。[Min] (最小) の新しい値が、グループの最大サイズよりも大きい場合、[Max] (最大) を更新する必要があります。
c. [繰り返し] で、[1 回] を選択します。
d. [Time zone (タイムゾーン)] でタイムゾーンを選択。タイムゾーンが選択されていない場合は、デフォルトでは、ETC/UTC が使用されます。
e. [Specific start time] (特定の開始時刻) を定義します。

5. [作成] を選択します。

   コンソールに Auto Scaling グループのスケジュールされたアクションが表示されます。

6. イベントの終了時に、最小キャパシティーを元の設定に戻すように、2 番目のスケジュールされたアクションを設定します。予測スケーリングでは、[Min] (最小) に設定した値が予測値未満の場合のみ、キャパシティーをスケーリングできます。

1 回限りのイベントに対して 2 つのスケジュールされたアクションを作成するには (AWS CLI)

を使用してスケジュールされたアクション AWS CLI を作成するには、put-scheduled-update-group-action コマンドを使用します。

例えば、5 月 19 日の午後 5 時から 8 時間、最小キャパシティーを 3 インスタンスに維持するスケジュールを定義しましょう。以下のコマンドは、このシナリオを実装する方法を示しています。

最初の put-scheduled-update-group-action コマンドは、2021 年 5 月 19 日の UTC 午後 5 時に指定された Auto Scaling グループの最小容量を更新するように Amazon EC2 Auto Scaling に指示します。 Auto Scaling

```
aws autoscaling put-scheduled-update-group-action --scheduled-action-name my-event-start \
  --auto-scaling-group-name my-asg --start-time "2021-05-19T17:00:00Z" --minimum-capacity 3
```

2 番目のコマンドは、2021 年 5 月 20 日の午前 1 時 (UTC) にグループの最小キャパシティーを 1 に設定するように Amazon EC2 Auto Scaling に指示します。

```
aws autoscaling put-scheduled-update-group-action --scheduled-action-name my-event-end \
```

```
  --auto-scaling-group-name my-asg --start-time "2021-05-20T01:00:00Z" --minimum-
capacity 1
```

これらのスケジュールされたアクションを Auto Scaling グループに追加すると、Amazon EC2 Auto Scaling は次の処理を実行します。

- 2021 年 5 月 19 日の午後 5 時 (UTC) に、最初にスケジュールされたアクションが実行されます。グループのインスタンスが 3 未満である場合、グループは 3 インスタンスにスケールアウトされます。この時刻以降の 8 時間の間、予測キャパシティーが実際のキャパシティーよりも大きい場合、または動的スケーリングポリシーが有効な場合、Amazon EC2 Auto Scaling は引き続きスケールアウトできます。
- 2021 年 5 月 20 日の午前 1 時 (UTC) に、2 番目のスケジュールされたアクションが実行されます。これにより、イベントの終了時に最小キャパシティーが元の設定に戻ります。

繰り返し起こるスケジュールに基づくスケーリング

毎週同じ期間の予測を上書きするには、2 つのスケジュールされたアクションを作成し、cron 式を使用して日時のロジックを指定します。

この cron 式のフォーマットは、スペースで区切られた 5 つのフィールド ([分] [時間] [日] [月] [曜日]) で構成されます。フィールドには、特殊文字を含む任意の許容される値を含めることができます。

例えば、次の cron 式は、毎週火曜日の午前 6:30 にアクションを実行します。アスタリスクは、フィールドのすべての値を照合するワイルドカードとして使用されます。

```
30 6 * * 2
```

以下も参照してください。

スケジュールされたアクションを作成、一覧表示、編集、および削除する方法の詳細については、「Amazon EC2 Auto Scaling のスケジュールされたスケーリング」を参照してください。

カスタムメトリクスを使用した高度な予測スケーリングポリシー設定

予測スケーリングポリシーでは、事前定義されたメトリクスまたはカスタムメトリクスを使用できます。カスタムメトリクスは、事前定義されたメトリクス (CPU、ネットワーク I/O、および Application Load Balancer のリクエスト数) ではアプリケーションの負荷が十分に反映できない場合に役立ちます。

カスタムメトリクスを使用して予測スケーリングポリシーを作成するときは、 が提供する他の CloudWatch メトリクスを指定するか AWS、自分で定義して公開するメトリクスを指定できます。Metric Math を使用して既存のメトリクスを集計し、自動的に追跡 AWS されない新しい時系列に変換することもできます。新しい合計や平均の計算など、データの値を組み合わせることを、集計すると言います。結果のデータは集計と言います。

以下のセクションには、ポリシー用の JSON 構造を構築する方法のベストプラクティスと例が記載されています。

トピック

- ベストプラクティス
- 前提条件
- カスタムメトリクス用の JSON の構築
- 考慮事項とトラブルシューティング
- 制限事項

ベストプラクティス

次のベストプラクティスは、カスタムメトリクスをより効果的に使用するのに役立ちます。

- 負荷メトリクスの指定では、グループのキャパシティーに関係なく Auto Scaling グループ全体の負荷を表すメトリクスが最も有用です。
- スケーリングメトリクスの指定では、インスタンスあたりの平均スループットまたは使用率のメトリクスでスケールするのが最も有用です。
- スケーリングメトリクスは、キャパシティーに反比例する必要があります。つまり、Auto Scaling グループのインスタンス数が増加すると、スケーリングメトリクスはほぼ同じ割合で減少するようにします。予測スケーリングが期待どおりに動作するようにするには、負荷メトリクスとスケーリングメトリクスに強い相関がある必要もあります。
- ターゲット使用率は、スケーリングメトリクスのタイプと一致する必要があります。CPU 使用率を使用するポリシー設定の場合、これはパーセンテージのターゲットです。リクエスト数やメッセージ数など、スループットを使用するポリシー設定の場合、これは任意の 1 分間のインスタンスあたりのリクエスト数やメッセージ数のターゲットです。
- これらの推奨事項に従わない場合、予測される将来の時系列の値は、多くの場合、誤りになります。データが正しいことを確認するために、Amazon EC2 Auto Scaling コンソールで予測値を表示できます。または、予測スケーリングポリシーを作成した後、 GetPredictiveScalingForecast

API の呼び出しによって返される LoadForecastおよび CapacityForecast オブジェクトを検査します。

- 予測スケーリングがキャパシティーのアクティブスケーリングを開始する前に予測を評価できるように、予測のみモードで予測スケーリングを設定することを強くお勧めします。

前提条件

予測スケーリングポリシーにカスタムメトリクスを追加するには、cloudwatch:GetMetricData 許可が必要です。

AWS が提供するメトリクスの代わりに独自のメトリクスを指定するには、まずメトリクスを に公開する必要があります CloudWatch。詳細については、「Amazon CloudWatch ユーザーガイド」の「カスタムメトリクスの発行」を参照してください。

独自のメトリクスを発行するときは、少なくとも 5 分間隔の頻度でデータポイントを発行するようにしてください。Amazon EC2 Auto Scaling は、必要な期間の長さ CloudWatch に基づいて からデータポイントを取得します。例えば、ロードメトリクスの仕様では、時間単位のメトリクスを使用してアプリケーションの負荷を測定します。 は、発行されたメトリクスデータ CloudWatch を使用して、各 1 時間内のタイムスタンプを持つすべてのデータポイントを集約することで、任意の 1 時間にわたって単一のデータ値を提供します。

カスタムメトリクス用の JSON の構築

次のセクションでは、 からデータをクエリするように予測スケーリングを設定する方法の例を示します CloudWatch。このオプションの設定には 2 つの異なる手法あり、予測スケーリングポリシーの JSON を構築するために使用する形式は、選択される手法の影響を受けます。メトリクス計算を使用する場合は、実行されるメトリクス計算に基づいて JSON の形式がさらに多様化します。

1. が提供する他の CloudWatch メトリクス AWS または に発行するメトリクスから直接データを取得するポリシーを作成するには CloudWatch、「」を参照してくださいカスタムロードメトリクスとスケーリングメトリクスを使用する予測スケーリングポリシーの例 (AWS CLI)。
2. 複数の CloudWatch メトリクスをクエリし、数式を使用してこれらのメトリクスに基づいて新しい時系列を作成できるポリシーを作成するには、「」を参照してくださいMetric Math 式を使用する。

カスタムロードメトリクスとスケーリングメトリクスを使用する予測スケーリングポリシーの例 (AWS CLI)

を使用してカスタムロードとスケーリングメトリクスを含む予測スケーリングポリシーを作成するには AWS CLI、 という名前の JSON ファイルに `--predictive-scaling-configuration` の引数を保存します`config.json`。

カスタムメトリクスの追加は、以下の例にある置き換え可能な値を独自のメトリクスとターゲット使用率に置き換えることによって開始します。

```
{
  "MetricSpecifications": [
    {
      "TargetValue": 50,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "scaling_metric",
            "MetricStat": {
              "Metric": {
                "MetricName": "MyUtilizationMetric",
                "Namespace": "MyNameSpace",
                "Dimensions": [
                  {
                    "Name": "MyOptionalMetricDimensionName",
                    "Value": "MyOptionalMetricDimensionValue"
                  }
                ]
              },
              "Stat": "Average"
            }
          }
        ]
      },
      "CustomizedLoadMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "load_metric",
            "MetricStat": {
              "Metric": {
                "MetricName": "MyLoadMetric",
                "Namespace": "MyNameSpace",
                "Dimensions": [
```

```
                {
                  "Name": "MyOptionalMetricDimensionName",
                  "Value": "MyOptionalMetricDimensionValue"
                }
              ]
            },
            "Stat": "Sum"
          }
        }
      ]
    }
  }
]
}
```

詳細については、MetricDataQueryAmazon EC2 Auto Scaling API リファレンス」の「」を参照してください。

> **Note**
>
> 以下は、メトリクスの名前、名前空間、ディメンション、統計情報の検索に役立つ追加のリソースです CloudWatch 。
>
> - AWS サービスで使用可能なメトリクスの詳細については、「Amazon AWS ユーザーガイド」の CloudWatch 「メトリクスを発行する のサービス」を参照してください。CloudWatch
> - を使用してメトリクスの正確な CloudWatch メトリクス名、名前空間、ディメンション (該当する場合) を取得するには AWS CLI、「list-metrics」を参照してください。

このポリシーを作成するには、次の例に示すように、JSON ファイルを入力として使用して put-scaling-policy コマンドを実行します。

```
aws autoscaling put-scaling-policy --policy-name my-predictive-scaling-policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
```

成功した場合、このコマンドはポリシーの Amazon リソースネーム (ARN) を返します。

```
{
```

```
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-predictive-scaling-policy",
  "Alarms": []
}
```

Metric Math 式を使用する

以下のセクションには、ポリシーでメトリクス計算を使用する方法を説明する予測スケーリングポリシーの情報と例が記載されています。

トピック

- Metric Math について
- メトリクス計算を使用してメトリクスを組み合わせる予測スケーリングポリシーの例 (AWS CLI)
- ブルー/グリーンデプロイシナリオで使用する予測スケーリングのポリシーの例 (AWS CLI)

Metric Math について

既存のメトリクスデータを集計するだけで、 CloudWatch メトリクス計算によって別のメトリクスを に発行する労力とコストを節約できます CloudWatch。 AWS が提供する任意のメトリクスを使用でき、アプリケーションの一部として定義したメトリクスを使用することもできます。例えば、インスタンスごとに Amazon SQS キューバックログを計算したい場合があります。これを行うには、キューから取得可能なメッセージのおおよその数を取得し、その数を Auto Scaling グループの実行中のキャパシティーで割ります。

詳細については、「Amazon ユーザーガイド」の「Metric Math の使用」を参照してください。 CloudWatch

予測スケーリングポリシーで Metric Math の数式を使用する場合は、次の点を考慮してください。

- Metric Math 演算では、メトリクスのメトリクス名、名前空間、ディメンションのキーと値のペアの一意の組み合わせのデータポイントを使用します。
- 算術演算子 (+ - * / ^)、統計関数 (AVG や SUM など) 、または が CloudWatch サポートするその他の関数を使用できます。
- 数式の関係式では、メトリクスと他の数式の結果の両方を使用できます。
- Metric Math の数式は、さまざまな集計で構成できます。ただし、最終的な集計結果として、`Average` をスケーリングメトリクスに使用し、`Sum` を負荷メトリクスに使用するのがベストプラクティスです。
- メトリクスの指定で使用される数式はすべて、最終的に単一の時系列を返す必要があります。

Metric Math を使用するには、次の操作を実行します。

- 1 つ以上の CloudWatch メトリクスを選択します。次に、数式を作成します。詳細については、「Amazon ユーザーガイド」の「Metric Math の使用」を参照してください。 CloudWatch
- コンソールまたは CloudWatch GetMetricData API を使用して、 CloudWatchメトリクスの数式が有効であることを確認します。

メトリクス計算を使用してメトリクスを組み合わせる予測スケーリングポリシーの例 (AWS CLI)

場合によっては、メトリクスを直接指定するのではなく、まず何らかの方法でそのデータを処理する必要がある場合があります。例えば、Amazon SQS キューから作業を取り出すアプリケーションがあり、キュー内の項目数を予測スケーリングの基準として使用したいとします。キューにあるメッセージの数だけでは、必要なインスタンスの数は定義されません。インスタンスごとのバックログを計算するために使用できるメトリクスを作成するには、さらに多くの作業が必要です。詳細については、「Amazon SQS に基づくスケーリング」を参照してください。

このシナリオの予測スケーリングポリシー例を次に示します。Amazon SQS `ApproximateNumberOfMessagesVisible` メトリクス (キューから取得可能なメッセージの数) に基づくスケーリングメトリクスおよび負荷メトリクスを指定します。Amazon EC2 Auto Scaling `GroupInServiceInstances` メトリクスと、スケーリングメトリクスのインスタンスごとのバックログを計算するための数式も使用します。

```
aws autoscaling put-scaling-policy --policy-name my-sqs-custom-metrics-policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
{
  "MetricSpecifications": [
    {
      "TargetValue": 100,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
          {
            "Label": "Get the queue size (the number of messages waiting to be
 processed)",
            "Id": "queue_size",
            "MetricStat": {
              "Metric": {
                "MetricName": "ApproximateNumberOfMessagesVisible",
                "Namespace": "AWS/SQS",
                "Dimensions": [
```

```
                {
                  "Name": "QueueName",
                  "Value": "my-queue"
                }
              ]
            },
            "Stat": "Sum"
          },
          "ReturnData": false
        },
        {
          "Label": "Get the group size (the number of running instances)",
          "Id": "running_capacity",
          "MetricStat": {
            "Metric": {
              "MetricName": "GroupInServiceInstances",
              "Namespace": "AWS/AutoScaling",
              "Dimensions": [
                {
                  "Name": "AutoScalingGroupName",
                  "Value": "my-asg"
                }
              ]
            },
            "Stat": "Sum"
          },
          "ReturnData": false
        },
        {
          "Label": "Calculate the backlog per instance",
          "Id": "scaling_metric",
          "Expression": "queue_size / running_capacity",
          "ReturnData": true
        }
      ]
    },
    "CustomizedLoadMetricSpecification": {
      "MetricDataQueries": [
        {
          "Id": "load_metric",
          "MetricStat": {
            "Metric": {
              "MetricName": "ApproximateNumberOfMessagesVisible",
              "Namespace": "AWS/SQS",
```

```
                "Dimensions": [
                  {
                    "Name": "QueueName",
                    "Value": "my-queue"
                  }
                ],
              },
              "Stat": "Sum"
            },
            "ReturnData": true
          }
        ]
      }
    }
  ]
}
```

この例では、ポリシーの ARN が返されます。

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-sqs-custom-metrics-policy",
  "Alarms": []
}
```

ブルー/グリーンデプロイシナリオで使用する予測スケーリングのポリシーの例 (AWS CLI)

検索式には、複数の Auto Scaling グループからメトリクスをクエリし、それらに対して数式を実行できる高度なオプションが用意されています。これは、Blue/Green デプロイで特に有用です。

> **Note**
>
> Blue/Green デプロイ とは、同一の Auto Scaling グループを 2 つ別々に作成するデプロイ方法です。本番トラフィックを受信するグループは 1 つだけです。ユーザートラフィックは、最初は以前の (「青」の) Auto Scaling グループに送信され、新しいグループ (「緑」) はアプリケーションまたはサービスの新しいバージョンのテストと評価に使用されます。新しいデプロイがテストされ、合格すると、ユーザートラフィックは緑の Auto Scaling グループに送信されるようになります。デプロイが成功したら、青のグループを削除できます。

Blue/Green デプロイの一部として新しい Auto Scaling グループが作成されると、メトリクスの指定を変更する必要なく、各グループのメトリクス履歴を自動的に予測スケーリングポリシーに含めることができます。詳細については、 コンピューティングブログの「ブルー/グリーンデプロイでの EC2 Auto Scaling 予測スケーリングポリシーの使用」を参照してください。 AWS

次のポリシー例で、これをどのように実行できるかを示します。この例では、ポリシーで、Amazon EC2 が出力する CPUUtilization メトリクスを使用します。Amazon EC2 Auto Scaling GroupInServiceInstances メトリクスとインスタンスごとのスケーリングメトリクスの値を計算するための数式を使用します。また、キャパシティーメトリック仕様を指定して、GroupInServiceInstances メトリクスを取得します。

検索式は、指定した検索条件に基づいて、複数の Auto Scaling グループ内のインスタンスの CPUUtilization を検索します。後で同じ検索条件に一致する新しい Auto Scaling グループを作成すると、新しい Auto Scaling グループのインスタンスの CPUUtilization が、自動的に含まれます。

```
aws autoscaling put-scaling-policy --policy-name my-blue-green-predictive-scaling-
policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
{
  "MetricSpecifications": [
    {
      "TargetValue": 25,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "load_sum",
            "Expression": "SUM(SEARCH('{AWS/EC2,AutoScalingGroupName} MetricName=
\"CPUUtilization\" ASG-myapp', 'Sum', 300))",
            "ReturnData": false
          },
          {
            "Id": "capacity_sum",
            "Expression": "SUM(SEARCH('{AWS/AutoScaling,AutoScalingGroupName}
 MetricName=\"GroupInServiceInstances\" ASG-myapp', 'Average', 300))",
            "ReturnData": false
          },
          {
            "Id": "weighted_average",
            "Expression": "load_sum / capacity_sum",
            "ReturnData": true
```

```
          }
        ]
      },
      "CustomizedLoadMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "load_sum",
            "Expression": "SUM(SEARCH('{AWS/EC2,AutoScalingGroupName} MetricName=
\"CPUUtilization\" ASG-myapp', 'Sum', 3600))"
          }
        ]
      },
      "CustomizedCapacityMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "capacity_sum",
            "Expression": "SUM(SEARCH('{AWS/AutoScaling,AutoScalingGroupName}
 MetricName=\"GroupInServiceInstances\" ASG-myapp', 'Average', 300))"
          }
        ]
      }
    }
  ]
}
```

この例では、ポリシーの ARN が返されます。

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-blue-green-predictive-
scaling-policy",
  "Alarms": []
}
```

考慮事項とトラブルシューティング

カスタムメトリクスの使用中に問題が発生した場合は、次の操作を実行することをお勧めします。

- エラーメッセージが表示された場合は、メッセージを読み、可能な場合は報告されている問題を解決します。
- Blue/Green デプロイシナリオで検索式を使用しようとしているときに問題が発生した場合は、まず、完全一致ではなく部分一致を検索する検索式の作成方法を理解していることを確認して

ください。また、クエリが、特定のアプリケーションを実行している Auto Scaling グループのみを見つけることを確認します。検索式の構文の詳細については、「Amazon ユーザーガイド」のCloudWatch 「検索式の構文」を参照してください。 CloudWatch

- 式を事前に検証しなかった場合、スケーリングポリシーを作成するときにput-scaling-policyコマンドによって検証されます。ただし、このコマンドでは、検出されたエラーの正確な原因を特定できない可能性があります。問題を解決するには、 get-metric-data コマンドへのリクエストからのレスポンスで受け取るエラーをトラブルシューティングします。 CloudWatch コンソールから式をトラブルシューティングすることもできます。
- コンソールで [Load] (負荷) と [Capacity] (キャパシティー) のグラフを表示すると、[Capacity] (キャパシティー) グラフにはデータがまったく表示されない場合があります。グラフに完全なデータが含まれるようにするには、Auto Scaling グループのグループメトリクスを常に有効にしてください。詳細については、「Auto Scaling グループのメトリクスを有効にする (コンソール)」を参照してください。
- キャパシティーメトリクスの指定は、有効期間にわたって異なる Auto Scaling グループで実行されるアプリケーションがある場合にのみ、Blue/Green デプロイで役立ちます。このカスタムメトリクスでは、複数の Auto Scaling グループの総キャパシティーを指定できます。予測スケーリングは、これを使用して、履歴データをコンソールの [Capacity] (キャパシティー) グラフに表示します。
- `MetricDataQueries` で SUM() のような数学関数を使用せずに、独自の SEARCH() 関数を指定する場合、`ReturnData` に `false` を指定する必要があります。これは、検索式が複数の時系列を返す可能性がある一方、数式に基づくメトリクス指定は 1 つの時系列しか返すことができないためです。
- 検索式に含まれるすべてのメトリクスは、同じ解像度である必要があります。

制限事項

- 1 つのメトリクス指定で最大 10 個のメトリクスのデータポイントをクエリできます。
- この制限に関しては、1 つの式は 1 つのメトリクスとしてカウントされます。

スケールイン中に終了する Auto Scaling インスタンスを制御する

Amazon EC2 Auto Scaling は、終了ポリシーを使用して、インスタンスを終了する順序を決定します。事前定義されたポリシーを使用するか、特定の要件を満たすカスタムポリシーを作成できます。カスタムポリシーまたはインスタンスのスケールイン保護を使用することで、Auto Scaling グループが、まだ終了する準備ができていないインスタンスを終了しないようにすることもできます。

内容

- Amazon EC2 Auto Scaling が終了ポリシーを使用する場合
- Amazon EC2 Auto Scaling の終了ポリシーを設定する
- Lambda を使用したカスタム終了ポリシーを作成する
- インスタンスのスケールイン保護を使用する
- インスタンスの終了を正常に処理するために、Amazon EC2 Auto Scaling でアプリケーションを設計する

Amazon EC2 Auto Scaling が終了ポリシーを使用する場合

以下のセクションでは、Amazon EC2 Auto Scaling が終了ポリシーを使用するシナリオについて説明します。

内容

- スケールインイベント
- インスタンスの更新
- アベイラビリティーゾーンの再調整

スケールインイベント

スケールインイベントは、グループの現在のキャパシティーよりも低い Auto Scaling グループ の希望するキャパシティーの新しい値がある場合にも発生します。

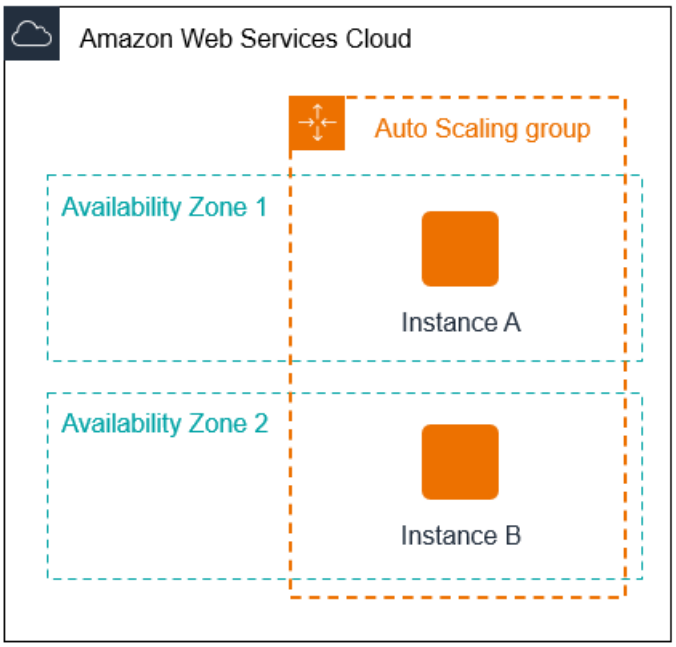
スケールインイベントは、次のシナリオで発生します。

- 動的スケーリングポリシーを使用し、メトリクス値の変更の結果としてグループのサイズが小さくなる場合
- スケジュールされたスケーリングを使用し、スケジュールされたアクションの結果としてグループのサイズが小さくなる場合
- 手動でグループのサイズを縮小します。

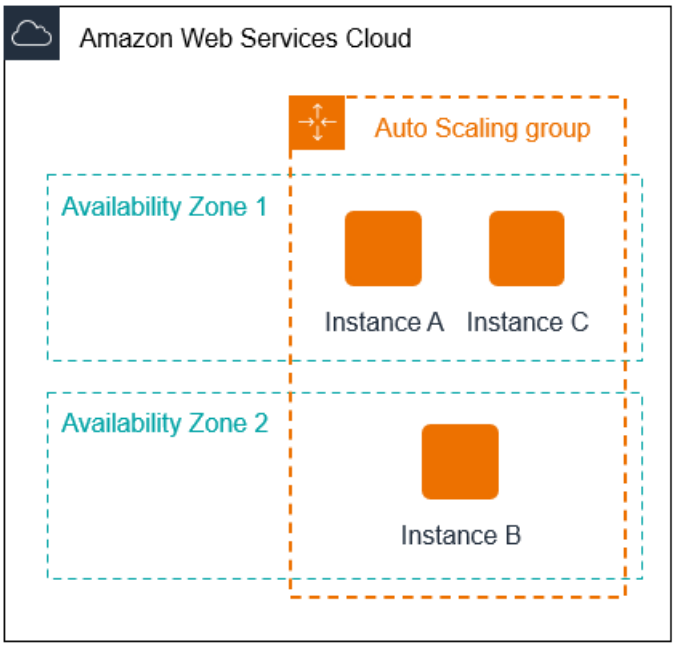
次に、スケールインイベントがある場合に終了ポリシーを使用する方法の例を示します。

1. この例の Auto Scaling グループには、1 つのインスタンスタイプ、2 つのアベイラビリティーゾーン、2 つのインスタンスの希望するキャパシティーがあるとします。また、リソース使用率

の増減時にインスタンスを追加および削除する、動的スケーリングポリシーもあります。このグループの 2 つのインスタンスは、次の図に示すように 2 つのアベイラビリティーゾーンに分散されます。

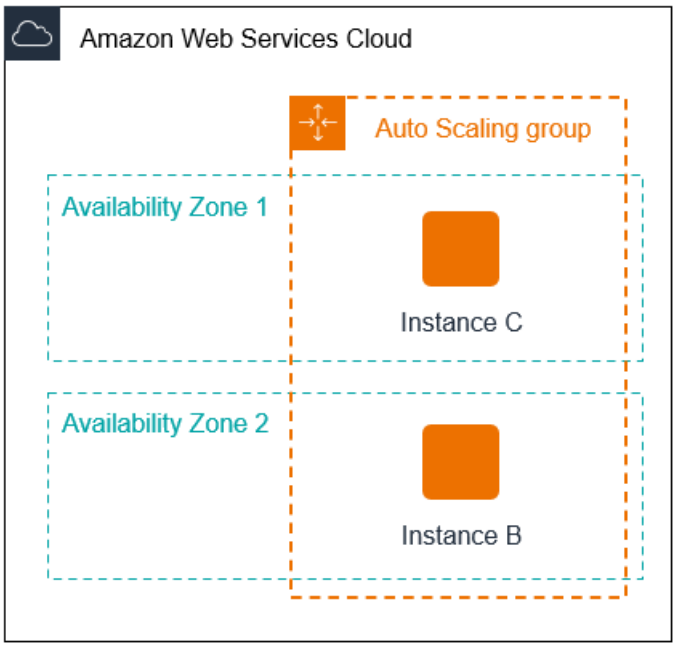


2. Auto Scaling グループがスケールアウトすると、Amazon EC2 Auto Scaling が新しいインスタンスを起動します。Auto Scaling グループには、次の図に示すように、2 つのアベイラビリティーゾーンに分散された 3 つのインスタンスがあります。



3. Auto Scaling グループがスケールインすると、Amazon EC2 Auto Scaling はインスタンスのいずれかを終了します。

4. グループに特定の終了ポリシーが割り当てられていない場合、Amazon EC2 Auto Scaling はデフォルトの終了ポリシーを使用します。2 つのインスタンスを持つアベイラビリティーゾーンを選択し、起動設定、別の起動テンプレート、または現在の起動テンプレートの最も古いバージョンから起動されたインスタンスを終了します。インスタンスが同じ起動テンプレートとバージョンから起動された場合、Amazon EC2 Auto Scaling は次の請求時間に最も近いインスタンスを選択し、終了します。



インスタンスの更新

インスタンスの更新を開始して、Auto Scaling グループのインスタンスを更新できます。インスタンスの更新中、Amazon EC2 Auto Scaling はグループのインスタンスを終了し、終了したインスタンスの置換インスタンスを起動します。Auto Scaling グループの終了ポリシーは、どのインスタンスを最初に置き換えるかを制御します。

アベイラビリティーゾーンの再調整

Amazon EC2 Auto Scaling は、Auto Scaling グループに対して有効になっているアベイラビリティーゾーン全体でキャパシティーを均等に分散させます。これにより、アベイラビリティーゾーンの停止による影響を軽減できます。アベイラビリティーゾーン間でのキャパシティーの分散のバランスが崩れた場合、Amazon EC2 Auto Scaling は、有効なアベイラビリティーゾーン内でインスタンスが最も少ないインスタンスを起動し、他の場所でインスタンスを終了することによって、Auto Scaling グループを再分散します。終了ポリシーは、最初に終了を優先するインスタンスを制御します。

アベイラビリティーゾーン間でのインスタンスの分散のバランスが崩れる原因はいくつかあります。

インスタンスの削除

Auto Scaling グループからインスタンスをデタッチ、スタンドバイにインスタンスを置くか、インスタンスを明示的に終了して希望するキャパシティーを減らし、代替インスタンスが起動しないようにすると、グループのバランスが崩れる可能性があります。この場合、 Amazon EC2 Auto Scaling は、アベイラビリティーゾーン間のバランスを再度取って不均衡を補います。

最初に指定したアベイラビリティーゾーンとは異なるアベイラビリティーゾーンの使用

Auto Scaling グループを拡張して追加のアベイラビリティーゾーンを含めるか、使用するアベイラビリティーゾーンを変更した場合、Amazon EC2 Auto Scaling は新しいアベイラビリティーゾーンでインスタンスを起動し、他のゾーンでインスタンスを終了します。これにより、インスタンスがアベイラビリティーゾーンに均等に広がります。

可用性の停止

可用性の停止はまれにしか発生しません。ただし、1 つのアベイラビリティーゾーンが使用できなくなってから回復すると、Auto Scaling グループでアベイラビリティーゾーン間で不均衡になる可能性があります。Amazon EC2 Auto Scaling はグループを徐々に再分散しようとしますが、再分散すると他のゾーンのインスタンスが終了する可能性があります。

例えば、1 つのインスタンスタイプ、2 つのアベイラビリティーゾーン、2 つのインスタンスの希望するキャパシティーのある Auto Scaling グループがあるとします。1 つのアベイラビリティーゾーンで障害が発生した場合、Amazon EC2 Auto Scaling は正常なアベイラビリティーゾーンで新しいインスタンスを自動的に起動し、異常のあるアベイラビリティーゾーン内のインスタンスを置き換えます。異常なアベイラビリティーゾーンが正常な状態にその後戻ると、Amazon EC2 Auto Scaling はこのゾーンで新しいインスタンスを自動的に起動し、影響を受けないゾーンのインスタンスを終了します。

> **Note**
>
> 分散バランスを再調整する場合、Amazon EC2 Auto Scaling は再調整によってアプリケーションのパフォーマンスや可用性が低下しないように、古いインスタンスを終了する前に新しいインスタンスを起動します。
> Amazon EC2 Auto Scaling は、古いインスタンスを終了する前に新しいインスタンスの起動を試みるため、指定した最大キャパシティーまたはそれに近い状態になると、再分散アクティビティの処理が遅くなったり、完全に停止する可能性があります。この問題を回避するため、再分散アクティビティの間、グループに対して指定されている最大キャパシティーが一時的に 10% のマージン（または 1 インスタンスのマージンのどちらか大きい方）で増えます。このマージンは、グループが最大キャパシティーに達しているか、それに近い状

態であり、ユーザーがゾーンの再設定をリクエストしたため、またはゾーンの可用性の問題を補正するために、グループの再分散が必要な場合にのみ追加されます。この追加キャパシティーは、グループの再分散に要する時間にわたってのみ提供されます。

Amazon EC2 Auto Scaling の終了ポリシーを設定する

終了ポリシーは、Amazon EC2 Auto Scaling が特定の順序でインスタンスを終了するために従う基準を提供します。

デフォルトでは、Amazon EC2 Auto Scaling は古い設定を使用しているインスタンスを最初に終了するように設計された終了ポリシーを使用します。終了ポリシーを変更して、最初に終了することが最も重要なインスタンスを制御できます。

Amazon EC2 Auto Scaling がインスタンスを終了すると、Auto Scaling グループで有効になっているアベイラビリティーゾーン間でバランスを維持しようとします。ゾーンバランスの維持は、終了ポリシーよりも優先されます。1 つのアベイラビリティーゾーンに他のアベイラビリティーゾーンよりも多くのインスタンスがある場合、Amazon EC2 Auto Scaling はまず不均衡なゾーンに終了ポリシーを適用します。アベイラビリティーゾーンのバランスが取れている場合は、すべてのゾーンに終了ポリシーが適用されます。

トピック

- デフォルトの終了ポリシーの仕組み
- デフォルトの終了ポリシーと混合インスタンスグループ
- 定義済みの終了ポリシー
- Auto Scaling グループの終了ポリシーを変更する

デフォルトの終了ポリシーの仕組み

Amazon EC2 Auto Scaling がインスタンスを終了する必要がある場合、まず、どのアベイラビリティーゾーン (またはゾーン) が最も多くのインスタンスと、スケールインから保護されていないインスタンスを少なくとも 1 つ特定します。次に、次のように、識別されたアベイラビリティーゾーン内の保護されていないインスタンスを評価します。

古い設定を使用するインスタンス

- 起動テンプレートを使用するグループの場合 – いずれかのインスタンスが古い設定を使用しているかどうかを判断し、次の順序で優先順位を付けます。

1. まず、起動設定で起動されたインスタンスを確認します。
2. 次に、現在の起動テンプレートではなく、別の起動テンプレートを使用して起動されたインスタンスを確認します。
3. 最後に、現在の起動テンプレートの最も古いバージョンを使用しているインスタンスを確認します。

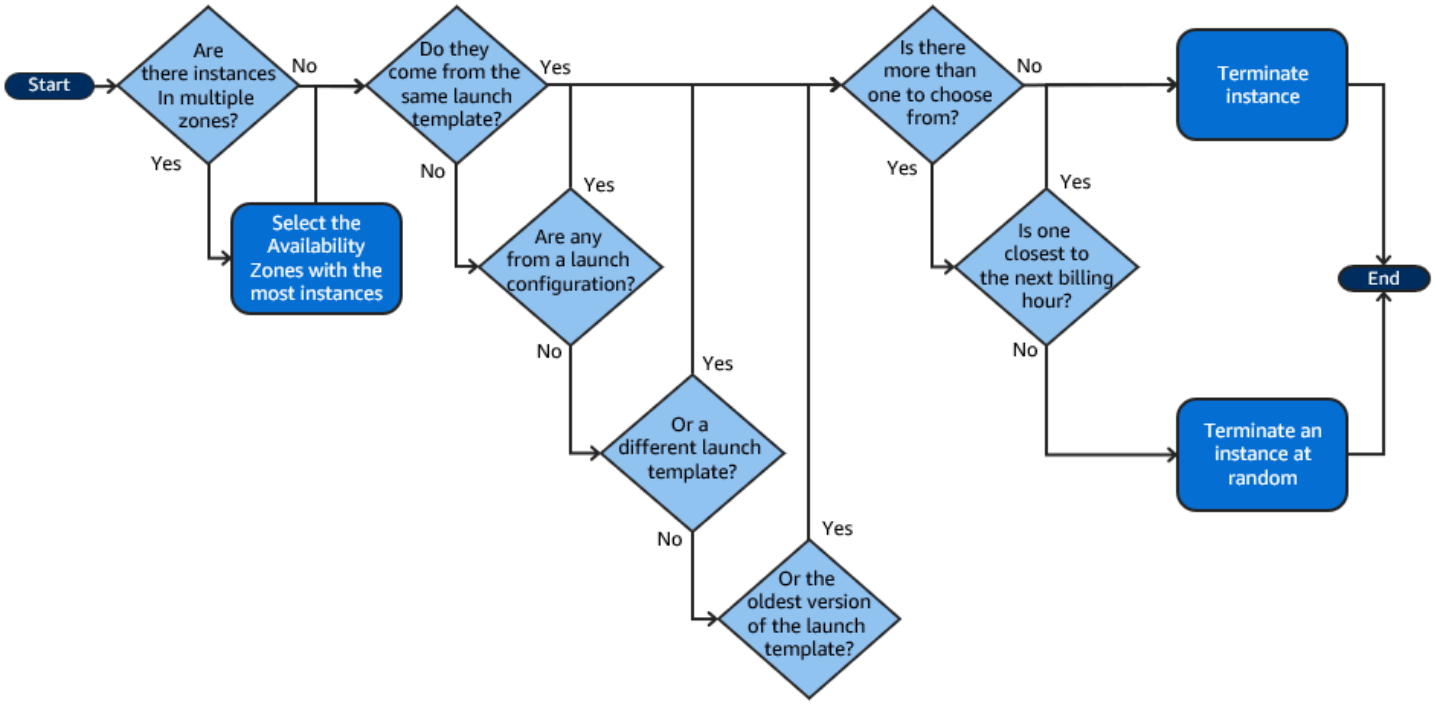
- 起動設定を使用するグループの場合 – いずれかのインスタンスが最も古い起動設定を使用しているかどうかを確認します。

古い設定のインスタンスが見つからない場合、または複数のインスタンスから選択できる場合、Amazon EC2 Auto Scaling は次の請求時間に近いインスタンスの次の基準を考慮します。

次の請求時間に近いインスタンス

前の条件を満たすインスタンスのいずれかが、次の請求時間に最も近いかどうかを確認します。複数のインスタンスが等しく近い場合は、1 つをランダムに終了します。これにより、時間単位で請求されるインスタンスの使用を最大化できます。ただし、EC2 のほとんどの使用量は 1 秒あたりに請求されるようになったため、この最適化によるメリットは少なくなります。詳細については、「Amazon EC2 料金」を参照してください。

次のフロー図は、起動テンプレートを使用するグループに対するデフォルトの終了ポリシーの仕組みを示しています。



デフォルトの終了ポリシーと混合インスタンスグループ

Amazon EC2 Auto Scaling は、混合インスタンスグループ のインスタンスを終了するときに追加の基準を適用します。

Amazon EC2 Auto Scaling がインスタンスを終了する必要がある場合、まずグループの設定に基づいて終了する購入オプション (スポットまたはオンデマンド) を特定します。これにより、グループが時間の経過とともにスポットインスタンスとオンデマンドインスタンスの指定された比率に向かうようになります。

次に、各アベイラビリティーゾーン内で終了ポリシーを個別に適用します。アベイラビリティーゾーンのバランスを維持するために終了するスポットインスタンスまたはオンデマンドインスタンスを決定します。同じロジックが、インスタンスタイプに定義された重みを持つ混合インスタンスグループに適用されます。

各ゾーン内では、デフォルトの終了ポリシーは次のように動作し、識別された購入オプション内でどの保護されていないインスタンスを終了できるかを決定します。

1. Auto Scaling グループの指定された割り当て戦略との整合性を向上させるために、いずれかのインスタンスを終了できるかどうかを決定します。最適化するインスタンスが特定されない場合、または複数のインスタンスから選択できる場合、評価は続行されます。
2. いずれかのインスタンスが古い設定を使用しているかどうかを確認し、次の順序で優先順位を付けます。
   a. まず、起動設定で起動されたインスタンスを確認します。
   b. 次に、現在の起動テンプレートではなく、別の起動テンプレートを使用して起動されたインスタンスを確認します。
   c. 最後に、現在の起動テンプレートの最も古いバージョンを使用してインスタンスを確認します。

   古い設定のインスタンスが見つからない場合、または選択できるインスタンスが複数ある場合、評価は続行されます。
3. いずれかのインスタンスが次の請求時間に最も近いかどうかを確認します。複数のインスタンスが等しく近い場合は、ランダムに 1 つ選択します。

定義済みの終了ポリシー

次の事前定義された終了ポリシーから選択します。

- **`Default`** – デフォルトの終了ポリシーに従ってインスタンスを終了します。
- **`AllocationStrategy`** – Auto Scaling グループのインスタンスを終了して、残りのインスタンスを、終了するインスタンスのタイプ (スポットインスタンスまたはオンデマンドインスタンス) の割り当て戦略に合わせます。このポリシーは、優先するインスタンスタイプが変更されたときに便利です。スポット配分戦略が `lowest-price` の場合、N 個の最低価格のスポットプール間で、スポットインスタンスの分散バランスを徐々に再調整できます。スポット配分戦略が `capacity-optimized` の場合、使用可能なスポットキャパシティーがより多いスポットプール間で、スポットインスタンスの分散バランスを徐々に再調整できます。優先度の低いタイプのオンデマンドインスタンスを優先度の高いタイプのオンデマンドインスタンスに徐々に置き換えることもできます。
- **`OldestLaunchTemplate`** – 最も古い起動テンプレートを持つインスタンスを終了します。このポリシーでは、現在の起動テンプレートを使用していないインスタンスが最初に終了され、その後、現在の起動テンプレートのうち最も古いバージョンを使用しているインスタンスが終了されます。このポリシーは、グループを更新し、以前の設定を使用しているインスタンスを廃止する場合に便利です。
- **`OldestLaunchConfiguration`** – 最も古い起動設定を持つインスタンスを終了します。このポリシーは、グループを更新し、以前の設定を使用しているインスタンスを廃止する場合に便利です。このポリシーでは、最新以外の起動設定を使用するインスタンスが最初に終了されます。
- **`ClosestToNextInstanceHour`** – 次の請求時間に最も近いインスタンスを終了します。これにより、時間単価のインスタンスを最大限に活用できます。
- **`NewestInstance`** – グループ内の最新のインスタンスを終了します。このポリシーは、新しい起動設定をテストするが、新しい設定は本稼働環境には保持しない場合に便利です。
- **`OldestInstance`** – グループ内の最も古いインスタンスを終了します。このオプションは、Auto Scaling グループ内のインスタンスを新しい EC2 インスタンスタイプにアップグレードする場合に役立ちます。より古いタイプのインスタンスをより新しいタイプのインスタンスに徐々に置き換えることができます。

> **Note**
>
> Amazon EC2 Auto Scaling は、どの終了ポリシーが使用されているかに関係なく、まずアベイラビリティーゾーン間でインスタンスのバランスをとります。その結果、いくつかの新しいインスタンスが古いインスタンスの前に終了される状況が発生する可能性があります。例えば、最近追加されたアベイラビリティーゾーンがある場合、グループに使用されている他のアベイラビリティーゾーンより多くのインスタンスが含まれるアベイラビリティーゾーンがある場合。

Auto Scaling グループの終了ポリシーを変更する

Auto Scaling グループの終了ポリシーを変更するには、次のいずれかの方法を使用します。

Console

Amazon EC2 Auto Scaling Auto Scaling コンソールで Auto Scaling グループを最初に作成するときに、終了ポリシーを変更することはできません。デフォルトの終了ポリシーが自動的に使用されます。Auto Scaling グループを作成したら、デフォルトポリシーを、適用する順序でリストされている別の終了ポリシーまたは複数の終了ポリシーに置き換えることができます。

Auto Scaling グループの終了ポリシーを変更するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。
3. [詳細] タブで、[高度な設定]、[編集] の順に選択します。
4. [終了ポリシー] で、1 つまたは複数の終了ポリシーを選択します。複数のポリシーを選択する場合は、適用する順に合わせて選択していきます。

   オプションで、[Custom termination policy] (カスタム終了ポリシー) を選択した後、ニーズを満たす Lambda 関数を選択することもできます。Lambda 関数のために作成したバージョンとエイリアスがある場合は、バージョン/エイリアス ドロップダウンリストから、いずれかのバージョンとエイリアスを選択します。Lambda 関数の未公開バージョンを使用する場合には、[Version/Alias] (バージョン/エイリアス) の設定はデフォルトのままにします。詳細については、「Lambda を使用したカスタム終了ポリシーを作成する」を参照してください。

   > **Note**
   >
   > 複数のポリシーを使用する場合は、その順序を正しく設定する必要があります。
   >
   > - [Default] (デフォルト) のポリシーを使用する場合は、リストの末尾にあるポリシーを選択する必要があります。
   > - [Custom termination policy] (カスタム終了ポリシー) を使用する場合には、リストの最初にあるポリシーを選択します。
5. [更新] を選択します。

AWS CLI

別のポリシーが指定されていない限り、デフォルトの終了ポリシーが自動的に使用されます。

Auto Scaling グループの終了ポリシーを変更するには

以下のいずれかのコマンドを使用します。

- create-auto-scaling-group
- update-auto-scaling-group

終了ポリシーを個別に使用することも、ポリシーのリストに組み合わせることもできます。例えば、次のコマンドを使用して、最初に OldestLaunchConfiguration ポリシーを使用し、その後で ClosestToNextInstanceHour ポリシーを使用するように Auto Scaling グループを更新します。

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg --
termination-policies "OldestLaunchConfiguration" "ClosestToNextInstanceHour"
```

Default の終了ポリシーを使用する場合、終了ポリシーのリストでは最後のポリシーになるように指定します。例えば --termination-policies "OldestLaunchConfiguration" "Default" です。

カスタム終了ポリシーを使用するには、まず を使用して終了ポリシーを作成する必要があります AWS Lambda。終了ポリシーとして使用する Lambda 関数を指定するには、終了ポリシーのリストで最初の関数を使用します。例えば、--termination-policies "arn:aws:lambda:us-west-2:123456789012:function:HelloFunction:prod" "OldestLaunchConfiguration" です。詳細については、「Lambda を使用したカスタム終了ポリシーを作成する」を参照してください。

Lambda を使用したカスタム終了ポリシーを作成する

Amazon EC2 Auto Scaling は、終了ポリシーを使用して、Auto Scaling グループのサイズを小さくするときに、最初に終了するインスタンスを優先順位付けします (スケールインと言います)。Auto Scaling グループではデフォルトの終了ポリシーを使用しますが、独自の終了ポリシーを選択または作成することもできます。定義済みの終了ポリシーを選択する方法の詳細については、「Amazon EC2 Auto Scaling の終了ポリシーを設定する」を参照してください。

このトピックでは、カスタム終了ポリシーを作成する方法について説明し、Amazon EC2 Auto Scaling が特定のイベントに応答して呼び出す AWS Lambda 関数を使用します。作成した Lambda 関数は、Amazon EC2 Auto Scaling から送信された入力データの情報を処理し、終了する準備ができているインスタンスのリストを返します。

カスタム終了ポリシーを使用すると、終了するインスタンスとタイミングをより適切に制御できます。例えば、Auto Scaling グループがスケールインする場合、Amazon EC2 Auto Scaling は中断すべきではない実行中のワークロードがあるかどうかを判断できません。Lambda 関数を使用すると、終了リクエストを検証し、ワークロードが完了するまで待機してから、終了するためにインスタンス ID を Amazon EC2 Auto Scaling に返して終了することができます。

内容

- 入力データ
- レスポンスデータ
- 考慮事項
- Lambda 関数を作成する
- 制限事項

入力データ

Amazon EC2 Auto Scaling は、スケールインイベントの JSON ペイロードを生成します。また、インスタンスの最大有効期間またはインスタンスの更新機能の結果としてインスタンスが終了されるときにもこれを行います。また、アベイラビリティーゾーン間でグループを再分散するときに開始できるスケールインイベントの JSON ペイロードも生成されます。

このペイロードには、Amazon EC2 Auto Scaling の終了に必要なキャパシティー、終了を提案するインスタンスのリスト、終了を開始したイベントに関する情報が含まれます。

次にペイロードの例を示します。

```
{
  "AutoScalingGroupARN": "arn:aws:autoscaling:us-east-1:<account-
id>:autoScalingGroup:d4738357-2d40-4038-ae7e-b00ae0227003:autoScalingGroupName/my-asg",
  "AutoScalingGroupName": "my-asg",
  "CapacityToTerminate": [
    {
      "AvailabilityZone": "us-east-1b",
```

```
      "Capacity": 2,
      "InstanceMarketOption": "on-demand"
    },
    {
      "AvailabilityZone": "us-east-1b",
      "Capacity": 1,
      "InstanceMarketOption": "spot"
    },
    {
      "AvailabilityZone": "us-east-1c",
      "Capacity": 3,
      "InstanceMarketOption": "on-demand"
    }
  ],
  "Instances": [
    {
      "AvailabilityZone": "us-east-1b",
      "InstanceId": "i-0056faf8da3e1f75d",
      "InstanceType": "t2.nano",
      "InstanceMarketOption": "on-demand"
    },
    {
      "AvailabilityZone": "us-east-1c",
      "InstanceId": "i-02e1c69383a3ed501",
      "InstanceType": "t2.nano",
      "InstanceMarketOption": "on-demand"
    },
    {
      "AvailabilityZone": "us-east-1c",
      "InstanceId": "i-036bc44b6092c01c7",
      "InstanceType": "t2.nano",
      "InstanceMarketOption": "on-demand"
    },
    ...
  ],
  "Cause": "SCALE_IN"
}
```


ペイロードには、Auto Scaling グループの名前、その Amazon リソースネーム (ARN)、および次の要素が含まれます:

- `CapacityToTerminate` は、特定のアベイラビリティーゾーンで終了するように設定されたスポットまたはオンデマンドのキャパシティーを示します。

- `Instances` は、Amazon EC2 Auto Scaling が「`CapacityToTerminate`」の情報に基づいて、終了を提案するインスタンスを表します。
- `Cause` は、終了の原因となったイベントである `SCALE_IN`、`INSTANCE_REFRESH`、`MAX_INSTANCE_LIFETIME`、`REBALANCE` を示します。

以下の情報は、Amazon EC2 Auto Scaling が入力データ `Instances` を生成する方法において最も重要な要因の概略を説明する

- スケールインイベントおよびインスタンスのリフレッシュベースの終了によってインスタンスが終了する場合は、アベイラビリティーゾーン間のバランスを維持することが優先されます。そのため、グループに使用されている他のアベイラビリティーゾーンより多くのインスタンスが含まれるアベイラビリティーゾーンがある場合、入力データのインスタンスはそのバランスのとれていないアベイラビリティーゾーンのみからインスタンスに適用されます。グループに使用されているアベイラビリティーゾーンのバランスがとれている場合、入力データにはグループのすべてのアベイラビリティーゾーンのインスタンスが含まれます。
- 混合インスタンスポリシーを使用する場合、各購入オプションの希望する割合に基づいて、スポットおよびオンデマンドのキャパシティーをバランスよく維持することも優先されます。まず、2 つのタイプ (スポットまたはオンデマンド) のどちらを終了すべきかを識別します。次に、アベイラビリティーゾーンのバランスが最も高い結果となるアベイラビリティーゾーンを終了できるインスタンス (特定された購入オプション内) を特定します。

レスポンスデータ

入力データと応答データが連携して、終了するインスタンスのリストを絞り込みます。

指定された入力では、Lambda 関数からの応答は次の例のようになります。

```
{
  "InstanceIDs": [
    "i-02e1c69383a3ed501",
    "i-036bc44b6092c01c7",
    ...
  ]
}
```

`InstanceIDs` は、終了する準備ができているインスタンスを表します。

または、終了する準備ができている別のインスタンスのセットを返すこともできます。これにより、入力データのインスタンスが上書きされます。Lambda 関数が呼び出されたときに終了する準備ができていない場合は、インスタンスを返さないように選択することもできます。

終了する準備ができているインスタンスがない場合、Lambda 関数からの応答は次の例のようになります。

```
{
  "InstanceIDs": [ ]
}
```

考慮事項

カスタム終了ポリシーを使用する場合、次の点を考慮してください。

- レスポンスデータで最初にインスタンスを返しても、その終了は保証されません。Lambda 関数が呼び出されたときに必要な数を超えるインスタンスが返された場合、Amazon EC2 Auto Scaling は、Auto Scaling グループに対して指定した他の終了ポリシーに対して各インスタンスを評価します。複数の終了ポリシーがある場合、リスト内の次の終了ポリシーを適用しようとします。終了に必要な数を超えるインスタンスがある場合は、次の終了ポリシーに移ります。他の終了ポリシーが指定されていない場合は、デフォルトの終了ポリシーを使用して、終了するインスタンスを決定します。
- インスタンスが返されない場合、または Lambda 関数がタイムアウトした場合、Amazon EC2 Auto Scaling は関数を再度呼び出す前に少し待機します。スケールインイベントでは、グループの希望するキャパシティーが現在のキャパシティーよりも小さい限り、試行を続けます。例えば、リフレッシュベースの終了の場合、1 時間試行し続けます。その後、インスタンスの終了に失敗し続けると、インスタンスの更新操作は失敗します。インスタンスの最大有効期間では、Amazon EC2 Auto Scaling は、その最大有効期間を超えていると識別されたインスタンスを終了しようとします。
- 関数は繰り返し再試行されるため、Lambda 関数をカスタム終了ポリシーとして使用する前に、コード内の永続的なエラーをテストして修正してください。
- 終了するインスタンスの独自のリストで入力データを上書きし、これらのインスタンスを終了してアベイラビリティーゾーンのバランスが崩れると、Amazon EC2 Auto Scaling はアベイラビリティーゾーン間のキャパシティーの分散を徐々に再調整します。まず、Lambda 関数を呼び出して、リバランシングを開始するかどうかを判断できるように、終了する準備ができているインスタンスがあるかどうかを確認します。終了する準備ができているインスタンスがある場合、最初に新しいインスタンスを起動します。インスタンスの起動が完了すると、グループの現在のキャパシ

ティーが希望するキャパシティーよりも大きいことが検出され、スケールインイベントが開始されます。

- 特定のインスタンスにおいて、スケールイン保護を使用して終了からの保護を行っている場合、その機能にカスタム終了ポリシーが影響をおよぼすことはありません。詳細については、「インスタンスのスケールイン保護を使用する」を参照してください。

Lambda 関数を作成する

まず Lambda 関数を作成し、Auto Scaling グループの終了ポリシーで Amazon リソースネーム (ARN) を指定できるようにします。

Lambda 関数を作成するには (コンソール)

1. Lambda コンソールで [Functions (関数)] ページを開きます。
2. 画面の上部のナビゲーションバーで、Auto Scaling グループの作成時に使用したのと同じリージョンを選択します。
3. [Create function (関数の作成)] を選択し、[Author from scratch (一から作成)] を選択します。
4. [基本的な情報] の [関数名] に、関数の名前を入力します。
5. [機能の作成]を選択します。関数のコードと設定に戻ります。
6. 関数をまだコンソールで開いている状態で、関数コードの下にエディタに貼り付けます。
7. [デプロイ] を選択します。
8. 必要に応じて、Lambda 関数の公開バージョンを作成するには、[Versions (バージョン)] タブをクリックし、次に新しいバージョンを発行します。Lambda でのバージョニングの詳細については、AWS Lambda デベロッパーガイドの「Lambda 関数のバージョン」を参照してください。
9. バージョンを公開することを選択した場合、このバージョンの Lambda 関数と関連付けるには [Aliases (エイリアス)] タブを選択します。Lambda のエイリアスの詳細については、AWS Lambda デベロッパーガイドの「Lambda 関数のエイリアス」を参照してください。
10. 次に [Configuration (設定)] タブと [Permissions (アクセス許可)] を選択します。
11. [Resource-based policy (リソースベースのポリシー)] にスクロールダウンして [アクセス許可の追加] を選択します。リソースベースのポリシーを使用して、関数を呼び出すアクセス許可を、ポリシーで指定されているプリンシパルに付与します。この場合、プリンシパルは Auto Scaling グループに関連付けられている Amazon EC2 Auto Scaling service-linked role です。
12. [Policy statement (ポリシーステートメント)] セクションで、権限を設定します。

    a. AWS アカウント を選択します。

b. Principal (プリンシパル) で、例えば **arn:aws:iam::<aws-account-id>:role/aws-service-role/autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling** といった呼び出しサービスにリンクされたロールの ARN を入力します。

c. アクション で、Lambda InvokeFunctionを選択します。

d. [Statement ID (ステートメント ID)] に **AllowInvokeByAutoScaling** といった一意のステートメント ID を入力します。

e. [保存] を選択します。

13. これらの指示に従った後、次のステップとして Auto Scaling グループの終了ポリシーの ARN を指定し続けます。詳細については、「Auto Scaling グループの終了ポリシーを変更する」を参照してください。

> **Note**
>
> Lambda 関数を開発するためのリファレンスとして使用できる例については、Amazon EC2 Auto Scaling のGitHub リポジトリ」を参照してください。

制限事項

- Auto Scaling グループの終了ポリシーで指定できる Lambda 関数は 1 つだけです。複数の終了ポリシーが指定されている場合は、最初に Lambda 関数を指定する必要があります。
- Lambda 関数を参照するには、修飾されていない ARN (サフィックスなし)、バージョンまたはエイリアスをサフィックスとして持つ修飾された ARN を使用します。修飾されていない ARN が使用されている場合 (例えば、`function:my-function`)、リソースベースのポリシーは、関数の未公開バージョンで作成する必要があります。修飾された ARN が使用されている場合 (例えば、`function:my-function:1` または `function:my-function:prod`)、リソースベースのポリシーは、その公開バージョンの関数に対して作成する必要があります。
- $LATEST サフィックスで修飾された ARN を使用できません。$LATEST サフィックスで修飾された ARN を参照するカスタム終了ポリシーを追加すると、エラーが発生します。
- 入力データで提供されるインスタンスの数は、30,000 インスタンスまでに制限されています。終了できるインスタンスが 30,000 個を超える場合、入力データには インスタンスの最大数が戻されることを示す `"HasMoreInstances": true` を示します。
- Lambda 関数の最大実行時間は 2 秒 (2000 ミリ秒) です。ベストプラクティスとして、予想される実行時間に基づいて Lambda 関数のタイムアウト値を設定する必要があります。Lambda 関数のデフォルトのタイムアウトは 3 秒ですが、これを減らすことができます。

- ランタイムが 2 秒の制限を超えると、ランタイムがこのしきい値を下回るまでスケールインアクションは保留されます。ランタイムが一貫して長い Lambda 関数の場合は、後続の Lambda 呼び出し中に取得できる結果をキャッシュするなど、ランタイムを短縮する方法を見つけます。

インスタンスのスケールイン保護を使用する

インスタンスのスケールイン保護により、Amazon EC2 Auto Scaling がどのインスタンスを終了できるかを制御できます。この機能の一般的なユースケースは、コンテナベースのワークロードのスケーリングです。詳細については、「インスタンスの終了を正常に処理するために、Amazon EC2 Auto Scaling でアプリケーションを設計する」を参照してください。

デフォルトでは、Auto Scaling グループを作成すると、インスタンスのスケールイン保護は無効になります。つまり、Amazon EC2 Auto Scaling はグループ内の任意のインスタンスを終了できます。

Auto Scaling グループでスケールイン保護設定を有効化しインスタンスを起動すると、その直後からインスタンスの保護が開始されます。インスタンスのスケールイン保護は、インスタンスの状態が `InService` の場合に開始されます。その後、終了できるインスタンスを制御するには、Auto Scaling グループ内で個別インスタンスのスケールイン保護設定を無効にします。そうすることで、引き続き特定のインスタンスを望ましくない終了から保護できます。

トピック

- 考慮事項
- Auto Scaling グループのスケールイン保護を変更する
- インスタンスのスケールイン保護を変更する

考慮事項

インスタンスのスケールイン保護を使用する際の考慮事項を次に示します。

- スケールインイベントが発生した際に、Auto Scaling グループのすべてのインスタンスがスケールインから保護されていると、必要とされるキャパシティーが減少します。ただし、Auto Scaling グループはインスタンスのスケールイン保護の設定が無効になるまで、必要な数のインスタンスを終了することはできません。では AWS Management Console、Auto Scaling グループ内のすべてのインスタンスがスケールインイベントが発生したときにスケールインから保護されている場合、Auto Scaling グループのアクティビティ履歴に次のメッセージが含まれます。`Could not scale to desired capacity because all remaining instances are protected from scale-in.`

- スケールインされない設定のインスタンスをデタッチすると、インスタンスのスケールイン保護の設定は失われます。インスタンスをグループに再度アタッチすると、グループの現在のインスタンスのスケールイン保護を受け継ぎます。Amazon EC2 Auto Scaling が新しいインスタンスを起動する、あるいはウォームプールから Auto Scaling グループにインスタンスを移動すると、そのインスタンスには、Auto Scaling グループでのインスタンスのスケールイン保護に関する設定が受け継がれます。
- インスタンスのスケールイン保護は、次の状況から Auto Scaling インスタンスを保護することはできません。
  - インスタンスがヘルスチェックに失敗した場合のヘルスチェックの置換。詳細については、「Auto Scaling グループ内のインスタンスのヘルスチェック」を参照してください。
  - スポットインスタンスの中断。キャパシティーが使用できなくなった場合、またはスポット料金が上限価格を超えた場合、スポットインスタンスは終了されます。
  - キャパシティブロックの予約は終了します。Amazon EC2 は、キャパシティブロックインスタンスがスケールインから保護されていても、それらを再利用します。
  - `terminate-instance-in-auto-scaling-group` コマンドによる手動終了。詳細については、「Auto Scaling グループのインスタンスを終了する (AWS CLI)」を参照してください。
  - Amazon EC2 コンソール、CLI コマンド、および API オペレーションによる手動終了。Auto Scaling インスタンスを手動の終了から保護するには、Amazon EC2 の終了保護を有効にします。(これにより、Amazon EC2 Auto Scaling が `terminate-instance-in-auto-scaling-group` コマンドを使用してインスタンスを終了したり、手動で終了したりするのを防ぐことはできません。) 起動テンプレートで Amazon EC2 終了保護を有効にする方法については、「」を参照してください詳細設定を使用して起動テンプレートを作成する。

Auto Scaling グループのスケールイン保護を変更する

Auto Scaling グループのインスタンスのスケールイン保護の設定は、有効または無効にすることができます。有効にすると、グループによって起動されるすべての新しいインスタンスで、インスタンスのスケールイン保護が有効になります。

Auto Scaling グループのこの設定を有効または無効にしても、既存のインスタンスには影響しません。

Console

新しい Auto Scaling グループのスケールイン保護を有効にするには

Auto Scaling グループを作成するときに、「グループサイズとスケーリングポリシーの設定」ページの「インスタンスのスケールイン保護」で、「インスタンスのスケールイン保護を有効にする」チェックボックスを選択します。

既存のグループのスケールイン保護を有効または無効にするには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループのチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。
3. [詳細] タブで、[高度な設定]、[編集] の順に選択します。
4. インスタンスのスケールイン保護 で、インスタンスのスケールイン保護を有効にするチェックボックスを選択またはオフにして、必要に応じてこのオプションを有効または無効にします。
5. [更新] を選択します。

AWS CLI

新しい Auto Scaling グループのスケールイン保護を有効にするには

次の create-auto-scaling-group コマンドを使用してインスタンスのスケールイン保護を有効にします。

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-asg --new-instances-protected-from-scale-in ...
```

既存のグループのスケールイン保護を有効にするには

次の update-auto-scaling-group コマンドを使用して、指定した Auto Scaling グループにインスタンスのスケールインの保護を有効にします。

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg --new-instances-protected-from-scale-in
```

既存のグループのスケールイン保護を無効にするには

次のコマンドを使用して、指定したグループのインスタンスのスケールイン保護を無効にします。

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg --no-new-instances-protected-from-scale-in
```

インスタンスのスケールイン保護を変更する

デフォルトで、インスタンスは所属する Auto Scaling グループからインスタンスのスケールイン保護の設定を取得します。ただし、起動後に個々のインスタンスのインスタンススケールイン保護を有効または無効にすることはできます。

Console

インスタンスのスケールイン保護を有効または無効にするには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。
3. [Instance management (インスタンス管理)] タブの [Instances (インスタンス)] で、インスタンスを選択します。
4. インスタンスのスケールイン保護を有効にするには、[Actions (アクション)]、[Set scale-in protection (スケールイン保護を設定)] の順に選択します。プロンプトが表示されると、[Set scale-in protection (スケールイン保護を設定)] を選択します。
5. インスタンスのスケールイン保護を無効にするには、[Actions (アクション)]、[Remove scale-in protection (スケールイン保護の削除)] の順に選択します。プロンプトが表示されたら、[Remove scale-in protection (スケールイン保護の削除)] を選択します。

AWS CLI

インスタンスのスケールイン保護を有効にするには

次の set-instance-protection コマンドを使用して、指定したインスタンスにおけるインスタンスのスケールインの保護を有効にします。

```
aws autoscaling set-instance-protection --instance-ids i-5f2e8a0d --auto-scaling-group-name my-asg --protected-from-scale-in
```

インスタンスのスケールイン保護を無効にするには

次のコマンドを使用して、指定したインスタンスにおけるインスタンスのスケールイン保護を無効にします。

```
aws autoscaling set-instance-protection --instance-ids i-5f2e8a0d --auto-scaling-group-name my-asg --no-protected-from-scale-in
```

> **Note**
>
> インスタンスのスケールイン保護は、Amazon EC2 コンソールまたは を使用してインスタンスを手動で終了するなど、人為的なエラーが発生した場合にインスタンスが終了しないことを保証しません AWS CLI。インスタンスが誤って終了されないように保護するために、Amazon EC2 終了保護を使用します。ただし、終了保護とインスタンスのスケールイン保護が有効になっている場合でも、ヘルスチェックでインスタンスが異常であると判断された場合、またはグループ自体が誤って削除された場合、インスタンスストレージに保存されたデータが失われる可能性があります。あらゆる環境と同様に、ベスト・プラクティスは、データのバックアップを頻繁に行うこと、またはビジネス継続性要件に適している場合にいつでもバックアップすることです。

インスタンスの終了を正常に処理するために、Amazon EC2 Auto Scaling でアプリケーションを設計する

このトピックでは、Amazon EC2 Auto Scaling がスケールインイベントに応答するときに、予期せず終了しないことが理想的であるインスタンス上でアプリケーションを実行している場合に採用できるさまざまなアプローチについて説明します。

例えば、長時間実行されるジョブの着信メッセージを収集する Amazon SQS キューがあるとします。新しいメッセージが到達すると、Auto Scaling グループのインスタンスがメッセージを取得し、処理を開始します。各メッセージの処理には 3 時間かかります。メッセージの数が増えると、新しいインスタンスが Auto Scaling グループに自動的に追加されます。メッセージの数が少なくなると、既存のインスタンスは自動的に終了します。この場合、Amazon EC2 Auto Scaling はどのインスタンスを終了するかを決定する必要があります。デフォルトでは、Amazon EC2 Auto Scaling は、現在アイドル状態のインスタンスではなく、3 時間かかるジョブの処理を開始してから 2.9 時間が経過したインスタンスを終了する可能性があります。Amazon EC2 Auto Scaling を使用する際の

予期せぬ終了の問題を回避するには、このシナリオに対応するようにアプリケーションを設計する必要があります。

次の機能を使用すると、Auto Scaling グループは、まだ終了の準備ができていないインスタンスを終了したり、割り当てられたジョブが未完了である状態でインスタンスを終了したりしなくなります。これらの 3 つの機能はすべて、組み合わせて使用することも、個別に使用することもできます。

内容

- インスタンスのスケールイン保護
- カスタム終了ポリシー
- 終了ライフサイクルフック

> **Important**
>
> インスタンスの終了を正常に処理するように Amazon EC2 Auto Scaling でアプリケーションを設計する際には、次の点に留意してください。
>
> - インスタンスに異常がある場合、(`ReplaceUnhealthy` プロセスを一時停止しない限り) どの機能を使用しているかにかかわらず、Amazon EC2 Auto Scaling はインスタンスを置き換えます。ライフサイクルフックを使用すると、アプリケーションは、正常にシャットダウンしたり、インスタンスの終了前に回復する必要があるデータをコピーしたりできます。
> - 終了ライフサイクルフックは、インスタンスの終了前に実行または完了することが保証されていません。何らかのエラーが発生した場合でも、Amazon EC2 Auto Scaling はインスタンスを終了します。

インスタンスのスケールイン保護

インスタンスのスケールイン保護は、インスタンスの終了が、デフォルトで拒否され、かつ、特定のインスタンスについてのみ明示的に許可されるべき重要なアクションである多くの状況で使用できます。例えば、コンテナ化されたワークロードを実行する場合、すべてのインスタンスを保護し、現在のタスクやスケジュールされたタスクがないインスタンスについてのみ保護を解除したいと考えるのが一般的です。Amazon ECS などのサービスは、インスタンスのスケールイン保護との統合を製品に組み込んでいます。

Auto Scaling グループでスケールイン保護を有効にして、インスタンスの作成時にスケールイン保護を適用したり、既存のインスタンスのためにそれを有効にしたりできます。インスタンスで実行する作業がなくなった場合、保護をオフに切り替えることができます。インスタンスは新しいジョブのためにポーリングを続行し、新しいジョブが割り当てられたら保護を再度有効にすることができます。

アプリケーションは、インスタンスが終了可能かどうかを管理する一元的なコントロールプレーンから、またはインスタンス自体から保護を設定できます。ただし、多数のインスタンスがスケールイン保護を継続的に切り替えている場合、大規模なフリートでスロットリングの問題が発生する可能性があります。

詳細については、「インスタンスのスケールイン保護を使用する」を参照してください。

カスタム終了ポリシー

インスタンスのスケールイン保護と同様に、カスタム終了ポリシーは、Auto Scaling グループが特定のインスタンスを終了しないようにするのに役立ちます。

デフォルトでは、Auto Scaling グループはデフォルトの終了ポリシーを使用して、最初に終了するインスタンスを決定します。どのインスタンスが最初に終了するかをさらに制御したい場合は、Lambda 関数を使用して独自のカスタム終了ポリシーを実装できます。Amazon EC2 Auto Scaling は、どのインスタンスを終了するかを決定する必要がある場合は常に関数を呼び出します。関数によって返されたインスタンスのみが終了します。関数がエラーを返し、タイムアウトし、または空のリストを生成した場合、Amazon EC2 Auto Scaling はインスタンスを終了しません。

カスタム終了ポリシーは、インスタンスが十分に冗長であるか、または十分に活用されていないためにインスタンスを終了できることがわかっている場合に役立ちます。これをサポートするには、グループ全体のワークロードをモニタリングするコントロールプレーンを備えたアプリケーションを実装する必要があります。これにより、インスタンスがまだジョブを処理している場合、Lambda 関数はそのインスタンスを含めてはならないことを認識できます。

詳細については、「Lambda を使用したカスタム終了ポリシーを作成する」を参照してください。

終了ライフサイクルフック

終了ライフサイクルフックは、既に終了対象として選択されているインスタンスの寿命を延長します。これにより、現在インスタンスに割り当てられているすべてのメッセージまたはリクエストを完了するため、または進行状況を保存して作業を別のインスタンスに転送するための追加の時間を確保できます。

多くのワークロードでは、終了対象として選択されたインスタンス上のアプリケーションを正常にシャットダウンするには、ライフサイクルフックで十分な場合があります。これはベストエフォート型のアプローチであり、エラーが発生した場合の終了を防ぐために使用することはできません。

ライフサイクルフックを使用するには、インスタンスの終了がいつ選択されるかを知る必要があります。これを知るには 2 つの方法があります。

オプション	説明	最適なケース	ドキュメントへのリンク
インスタンス内	インスタンスメタデータサービス (IMDS) は、インスタンスのステータスをインスタンスから直接ポーリングできる安全なエンドポイントです。メタデータが `Terminated` で返された場合、インスタンスの終了がスケジュールされています。	インスタンスを終了する前に、インスタンスに対してアクションを実行する必要があるアプリケーション。	ターゲットライフサイクル状態を取得する
インスタンス外	インスタンスが終了する際に、イベント通知が生成されます。Amazon EventBridge、Amazon SQS、または Amazon SNS を使用してルールを作成し、これらのイベントをキャプチャし、Lambda 関数で などのレスポンスを呼び出すことができます。	インスタンスの外部でアクションを実行する必要があるアプリケーション。	通知ターゲットを作成する

ライフサイクルフックを使用するには、インスタンスの終了準備が完全に整うタイミングを知る必要もあります。Amazon EC2 Auto Scaling は、CompleteLifecycleアクション呼び出しを受信するか、タイムアウトが経過するかのいずれか早い方まで、インスタンスを終了するように Amazon EC2 に指示しません。

デフォルトでは、インスタンスは終了ライフサイクルフックにより、引き続き 1 時間にわたって実行できます (ハートビートタイムアウト)。ライフサイクルアクションを完了するのに 1 時間では足

りない場合は、デフォルトのタイムアウトを設定できます。ライフサイクルアクションが実際に進行中の場合は、RecordLifecycleActionHeartbeatAPI コールを使用してタイムアウトを延長できます。

詳細については、「Amazon EC2 Auto Scaling のライフサイクルフック」を参照してください。

Amazon EC2 Auto Scaling プロセスの一時停止と再開

このトピックでは、Auto Scaling グループの 1 つ以上のプロセスを一時停止して再開し、特定のオペレーションを一時的に無効にする方法について説明します。

プロセスの停止は、スケーリングポリシーやスケジュールされたアクションの干渉なしに問題を調査またはトラブルシューティングする必要がある場合に役立ちます。また、Auto Scaling グループを変更している間にAmazon EC2 Auto Scaling がインスタンスを異常とマークして置き換えるのを防ぐこともできます。

トピック

- プロセスのタイプ
- 考慮事項
- プロセスを中断する
- プロセスを再開する
- 中断されたプロセスが他のプロセスに与える影響

> **Note**
>
> Auto Scaling グループのプロセスは、お客様が中断するだけでなく、インスタンスの起動に繰り返し失敗するという理由で Amazon EC2 Auto Scaling によって中断されることもあります。これは、管理上の中断と呼ばれます。管理上の中断は一般に、24 時間以上インスタンスの起動を試みているが、インスタンスの起動に成功しない Auto Scaling グループに適用されます。管理上の理由で Amazon EC2 Auto Scaling によって中断されたプロセスは、お客様が再開できます。

プロセスのタイプ

中断/再開機能は、以下のプロセスをサポートします。

- `Launch` – グループがスケールアウトしたとき、または Amazon EC2 Auto Scaling がウォームプールにインスタンスを追加するときなど、他の理由でインスタンスを起動することを選択した場合に、Auto Scaling グループにインスタンスを追加します。 Auto Scaling
- `Terminate` – グループがスケールインするとき、または Amazon EC2 Auto Scaling が最大存続期間を超えたためにインスタンスが終了したり、ヘルスチェックに失敗したりするなどの理由でインスタンスを終了することを選択した場合に、Auto Scaling グループからインスタンスを削除します。 Auto Scaling
- `AddToLoadBalancer` – インスタンスが起動されたときに、アタッチされたロードバランサーターゲットグループまたは Classic Load Balancer にインスタンスを追加します。詳細については、「Elastic Load Balancing を使用して Auto Scaling グループ内のインスタンス全体にトラフィックを分散させる」を参照してください。
- `AlarmNotification` – 動的スケーリングポリシーに関連付けられているアラームからの CloudWatch通知を受け入れます。詳細については、「Amazon EC2 Auto Scaling の動的スケーリング」を参照してください。
- `AZRebalance` – 以前に使用できなかったアベイラビリティーゾーンが正常な状態に戻った場合など、グループが不均衡になったときに、指定されたすべてのアベイラビリティーゾーンにわたってグループ内の EC2 インスタンスの数を均等に分散します。詳細については、「アクティビティの再分散」を参照してください。
- `HealthCheck` — Amazon EC2 または Elastic Load Balancing がインスタンスに異常があることを Amazon EC2 Auto Scaling に指示した場合、インスタンスの状態をチェックし、インスタンスを異常としてマークします。このプロセスは、手動で設定したインスタンスのヘルス ステータスをオーバーライドできます。詳細については、「Auto Scaling グループ内のインスタンスのヘルスチェック」を参照してください。
- `InstanceRefresh` – インスタンスの更新機能を使用してインスタンスを終了して置き換えます。詳細については、「インスタンスの更新を使用して Auto Scaling グループのインスタンスを更新する」を参照してください。
- `ReplaceUnhealthy` – 異常とマークされたインスタンスを終了し、新しいインスタンスを作成して置き換えます。詳細については、「Auto Scaling グループ内のインスタンスのヘルスチェック」を参照してください。
- `ScheduledActions` – スケーリングプランを作成し、予測スケーリングを有効にするときに、作成した、または自動的に作成されたスケジュールされた AWS Auto Scaling スケーリングアクションを実行します。詳細については、「Amazon EC2 Auto Scaling のスケジュールされたスケーリング」を参照してください。

考慮事項

プロセスを中断する前に、以下を考慮してください。

- 一時停止AlarmNotificationすると、スケーリングポリシーまたは関連する CloudWatch アラームを削除せずに、グループのターゲット追跡、ステップ、および簡易スケーリングポリシーを一時的に停止できます。その代わりに個々のスケーリングポリシーを一時的に停止するには、「Auto Scaling グループのスケーリングポリシーを無効化する」を参照してください。
- Amazon EC2 Auto Scaling がヘルスチェックに基づいてインスタンスを終了せずに、インスタンスを再起動する HealthCheckおよび ReplaceUnhealthyプロセスを停止することもできます。ただし、Amazon EC2 Auto Scaling で残りのインスタンスのヘルスチェックを引き続き実行する必要がある場合は、代わりにスタンバイ機能を使用してください。詳細については、「Auto Scaling グループからインスタンスを一時的に削除する」を参照してください。
- Launch プロセスと Terminate プロセス、または AZRebalance を中断してから、インスタンスのデタッチ、または指定されたアベイラビリティーゾーンの変更などで Auto Scaling グループを変更すると、アベイラビリティーゾーン間でのグループのバランスが悪くなる可能性があります。その場合は、中断されたプロセスの再開後、Amazon EC2 Auto Scaling が徐々に、インスタンスをアベイラビリティーゾーン間で均等に再分散します。
- Terminate プロセスを停止しても、force delete オプションを指定して delete-auto-scaling-group コマンドを使用することで、インスタンスを強制的に終了できます。
- Terminate プロセスの停止は、現在 InService状態にあるインスタンスにのみ適用されます。など、他の状態のインスタンスPendingや、スタンバイから適切に再開できないインスタンスの終了を防ぐことはできません。
- AWS CLI または SDKs を使用して Auto Scaling グループを記述する呼び出しに存在する場合、RemoveFromLoadBalancerLowPriorityこのプロセスは無視できます。このプロセスは非推奨で後方互換性のためにのみ保持されています。

プロセスを中断する

Auto Scaling グループのプロセスを停止するには、次のいずれかの方法を使用します。

Console

プロセスを停止するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。

2. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。
3. [詳細] タブで、[高度な設定]、[編集] の順に選択します。
4. [Suspended processes (中断したプロセス)] で、停止するプロセスを選択します。
5. [更新] を選択します。

AWS CLI

以下の suspend-processes コマンドを使用して、個々のプロセスを中断します。

```
aws autoscaling suspend-processes --auto-scaling-group-name my-asg --scaling-
processes HealthCheck ReplaceUnhealthy
```

すべてのプロセスを中断するには、以下のように `--scaling-processes` オプションを削除します。

```
aws autoscaling suspend-processes --auto-scaling-group-name my-asg
```

プロセスを再開する

Auto Scaling グループの中断されたプロセスを再開するには、次のいずれかの方法を使用します。

Console

停止されたプロセスを再開するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。
3. [詳細] タブで、[高度な設定]、[編集] の順に選択します。
4. [Suspended processes] (中断されたプロセス) で、中断されたプロセスを削除します。
5. [更新] を選択します。

AWS CLI

中断されたプロセスを再開するには、次の resume-processes コマンドを使用します。

```
aws autoscaling resume-processes --auto-scaling-group-name my-asg --scaling-
processes HealthCheck
```

中断されたすべてのプロセスを再開するには、以下のように --scaling-processes オプションを削除します。

```
aws autoscaling resume-processes --auto-scaling-group-name my-asg
```

中断されたプロセスが他のプロセスに与える影響

以下のセクションでは、さまざまなプロセスが個別に中断された場合に何が起こるかについて説明します。

トピック

- Launch が中断されている
- Terminate が中断されている
- AddToLoadBalancer が中断されている
- AlarmNotification が中断されている
- AZRebalance が停止されている
- HealthCheck が中断されている
- InstanceRefresh が中断されている
- ReplaceUnhealthy が中断されている
- ScheduledActions が中断されている
- 追加の考慮事項

Launch が中断されている

- AlarmNotification は引き続きアクティブですが、Auto Scaling グループは、しきい値を超過した状態のアラームに対してスケールアウトアクティビティを開始できません。
- ScheduledActions はアクティブですが、Auto Scaling グループは、実行されるスケジュールされたアクションに対してスケールアウトアクティビティを開始できません。

- AZRebalance は、グループの再配分を停止します。
- ReplaceUnhealthy は引き続き異常なインスタンスを終了しますが、置き換えは開始しません。Launch プロセスを再開すると、Amazon EC2 Auto Scaling は、Launch 停止中に終了されたインスタンスを直ちに置き換えます。
- InstanceRefresh はインスタンスを置き換えません。

Terminate が中断されている

- AlarmNotification は引き続きアクティブですが、Auto Scaling グループは、しきい値を超過した状態のアラームに対してスケールインアクティビティを開始できません。
- ScheduledActions はアクティブですが、Auto Scaling グループは、実行されるスケジュールされたアクションに対してスケールインアクティビティを開始できません。
- AZRebalance はまだアクティブですが、正しく機能していません。古いインスタンスを終了せずに新しいインスタンスを起動することがあります。これにより、Auto Scaling グループがその最大サイズより 10% まで大きくなることあります。バランスの再調整アクティビティ中にこの状態が一時的に許可されるためです。Terminate プロセスを再開するまで、Auto Scaling グループは最大サイズを超えることがあります。
- ReplaceUnhealthy は非アクティブですが、HealthCheck はアクティブです。Terminate が再開されると、ReplaceUnhealthy プロセスはすぐに実行を開始します。Terminate が中断されている間に異常とマークされたインスタンスがある場合、それらのインスタンスはすぐに置き換えられます。
- InstanceRefresh はインスタンスを置き換えません。

AddToLoadBalancer が中断されている

- Amazon EC2 Auto Scaling はインスタンスを起動しますが、ロードバランサーターゲットグループまたは Classic Load Balancer に追加しません。AddToLoadBalancer プロセスを再開すると、インスタンスが起動されるときのロードバランサーへの追加が再開されます。ただし、このプロセスが中断されている間に起動されたインスタンスは追加されません。これらのインスタンスを手動で登録する必要があります。

`AlarmNotification` が中断されている

- Amazon EC2 Auto Scaling は、 CloudWatch アラームのしきい値を超えた場合、スケーリングポリシーを呼び出しません。`AlarmNotification` を再開すると、Amazon EC2 Auto Scaling はアラームしきい値に現在違反しているポリシーを処理します。

`AZRebalance` が停止されている

- Amazon EC2 Auto Scaling は、特定イベントの発生後にインスタンスの再分散を試みません。ただし、スケールアウトまたはスケールインのイベントが発生した場合でも、スケーリングプロセスはアベイラビリティーゾーン間のバランスを調整します。例えば、スケールアウト中に、インスタンスが最も少ないアベイラビリティーゾーンでインスタンスを起動します。`AZRebalance` が中断されている間にグループのバランスがとれなくなった場合、そのプロセスを再開しても、Amazon EC2 Auto Scaling はグループのバランスを再調整しようとしません。最初に `Launch` を呼び出してから `Terminate` を呼び出します。

`HealthCheck` が中断されている

- Amazon EC2 Auto Scaling は、EC2 および Elastic Load Balancing のヘルスチェックの結果として、インスタンスに異常とマークしなくなります。カスタムヘルスチェックは引き続き正常に機能します。`HealthCheck` を中断した後、必要に応じて、グループ内のインスタンスのヘルス状態を手動で設定し、`ReplaceUnhealthy` がそれらのインスタンスを置き換えるようにできます。

`InstanceRefresh` が中断されている

- Amazon EC2 Auto Scaling は、インスタンス更新の結果としてのインスタンスの置き換えを停止します。進行中のインスタンス更新がある場合、操作はキャンセルされず、一時停止されます。

`ReplaceUnhealthy` が中断されている

- Amazon EC2 Auto Scaling は、異常とマークされたインスタンスを置き換えなくなります。EC2 または Elastic Load Balancing のヘルスチェックに失敗したインスタンスも異常とマークされます。`ReplaceUnhealthy` プロセスを再開するとすぐに、Amazon EC2 Auto Scaling はこのプロセスが中断されている間に異常とマークされたインスタンスを置き換えます。`ReplaceUnhealthy` プロセスは最初に `Terminate` を呼び出し、次に `Launch` を呼び出します。

`ScheduledActions` が中断されている

- Amazon EC2 Auto Scaling は、中断期間中に実行されるようにスケジュールされたアクションを実行しません。`ScheduledActions` を再開すると、Amazon EC2 Auto Scaling は、スケジュールされたアクションのうち、実行時間が過ぎていないもののみを考慮します。

追加の考慮事項

さらに、`Launch` または `Terminate` が中断される場合は、以下の機能が正しく機能しない可能性があります。

- インスタンスの最大有効期間 – `Launch`または `Terminate`が中断されている場合、インスタンスの最大有効期間機能はインスタンスを置き換えることができません。
- スポットインスタンスの中断 – `Terminate`が中断され、Auto Scaling グループにスポットインスタンスがある場合、スポットキャパシティーが利用できなくなっても、スポットインスタンスを終了できます。`Launch` が中断されている間、Amazon EC2 Auto Scaling は、別のスポットインスタンスプール、または同じスポットインスタンスプール (再度利用可能になったとき) から代替インスタンスを起動できません。
- 容量の再調整 — `Terminate`が中断され、容量の再調整を使用してスポットインスタンスの中断を処理する場合、Amazon EC2 スポットサービスは、スポット容量が使用できなくなってもインスタンスを終了できます。`Launch` が中断される場合、Amazon EC2 Auto Scaling は、別のスポットインスタンスプール、または同じスポットインスタンスプール (再度利用可能になったとき) から代替インスタンスを起動できません。
- インスタンスのアタッチとデタッチ — `Launch`と `Terminate`が中断されている場合、Auto Scaling グループにアタッチされているインスタンスをデタッチできますが、 が中断されている間`Launch`は、グループに新しいインスタンスをアタッチすることはできません。
- スタンバイインスタンス - `Launch`と `Terminate`が中断されている場合、インスタンスを`Standby`状態にすることはできますが、 `Launch` が中断されている間は、 `Standby`状態のインスタンスをサービスに戻すことはできません。

Amazon EC2 Auto Scaling グループのモニタリング

モニタリングは、Amazon EC2 Auto Scaling および AWS クラウド ソリューションの信頼性、可用性、パフォーマンスを維持する上で重要な部分です。 AWS は、Amazon EC2 Auto Scaling をモニタリングし、問題が発生したときに報告し、必要に応じて自動アクションを実行するための以下のモニタリングツールを提供します。

ヘルスチェック

Amazon EC2 Auto Scaling は、Auto Scaling グループのインスタンスに対して定期的にヘルスチェックを実行します。インスタンスがヘルスチェックに合格しない場合、そのインスタンスは異常とマークされ、Amazon EC2 Auto Scaling がインスタンスを置き換えるために新しいインスタンスを起動する間に終了します。詳細については、「Auto Scaling グループ内のインスタンスのヘルスチェック」を参照してください。

AWS Health Dashboard

は情報 AWS Health Dashboard を表示し、 AWS リソースの正常性の変化によって呼び出される通知も提供します。情報は 2 つの方法で表示されます。ダッシュボードには、最近のイベントおよび予定されているイベントがカテゴリ別に分類されて表示されます。詳細なイベントログには、過去 90 日間のすべてのイベントが表示されます。詳細については、「AWS Health Dashboard Amazon EC2 Auto Scaling に関する通知」を参照してください。

CloudTrail

を使用すると AWS CloudTrail、 によって、または に代わって Amazon EC2 Auto Scaling API に対して行われた呼び出しを追跡できます AWS アカウント。 は、指定した Amazon S3 バケットのログファイルに情報 CloudTrail を保存します。これらのログファイルを使用して、Auto Scaling グループの動作をモニタリングできます。ログには、実行されたリクエスト、そのリクエストの作成元のソース IP アドレス、リクエストの実行者、リクエストの実行日時などが含まれています。詳細については、「Amazon EC2 Auto Scaling API 呼び出しを口グに記録する AWS CloudTrail」を参照してください。

> **Amazon EC2 インスタンスのログ収集**
>
> を使用して CloudWatch 、EC2 インスタンスのオペレーティングシステムからログを収集できます。詳細については、「Amazon CloudWatch ユーザーガイド」の CloudWatch 「 エージェントを使用して Amazon EC2 インスタンスとオンプレミスサーバーからメトリクスと口グを収集する」および CloudWatch 「ログに送信された口グデータを表示する」を参照してください。

ワークロードに関するデータのログ記録と収集に役立つ他の AWS のサービスについては、「 規範ガイダンス」の「アプリケーション所有者向けのログ記録とモニタリングガイド」を参照してください。 AWS

Amazon CloudWatch

Amazon CloudWatch は、ログを分析し、 AWS リソースとホストされたアプリケーションのメトリクスをリアルタイムでモニタリングするのに役立ちます。メトリクスを収集および追跡し、カスタマイズされたダッシュボードを作成し、指定されたメトリックが指定したしきい値に達したときに通知またはアクションを実行するアラームを設定できます。たとえば、ネットワークアクティビティがメトリクスの期待値よりも急激に高くなった、または低くなったときに、通知を受け取ることができます。このサービスを使用して Auto Scaling グループとインスタンスのメトリックスをモニターリングする方法の詳細については、「Auto Scaling グループとインスタンスの CloudWatch メトリクスをモニタリングする」を参照してください。

CloudWatch は、Amazon EC2 Auto Scaling の AWS API 使用状況メトリクスも追跡します。これらのメトリクスを使用して、API 呼び出し量が定義したしきい値を超えたときに警告するアラームを設定できます。詳細については、「Amazon ユーザーガイド」のAWS 「使用状況メトリクス」を参照してください。 CloudWatch

AWS Compute Optimizer

Compute Optimizer は、新しいインスタンスタイプに移行するかどうかの判断に役立つ Amazon EC2 インスタンス推奨を提供します。Auto Scaling グループのインスタンスタイプが最適かどうかを分析し、コストを削減してワークロードのパフォーマンスを向上させるための推奨事項を生成します。詳細については、「AWS Compute Optimizer を使用して Auto Scaling グループのインスタンスタイプのレコメンデーションを取得する」を参照してください。

Amazon EventBridge

Amazon EventBridge は、アプリケーションをさまざまなソースのデータに簡単に接続できるサーバーレスイベントバスサービスです。 は、独自のアプリケーション、Software-as-a-Service (SaaS) アプリケーション、および AWS のサービスからリアルタイムデータのストリームを EventBridge 配信し、そのデータを Lambda などのターゲットにルーティングします。これにより、サービスで発生したイベントをモニタリングし、イベント駆動型アーキテクチャを構築できます。詳細については、「 EventBridge を使用して Auto Scaling イベントを処理する」を参照してください。

AWS Security Hub

AWS Security Hub を使用して、セキュリティのベストプラクティスに関連して Amazon EC2 Auto Scaling の使用状況をモニタリングできます。Security Hub は、検出セキュリティコントロールを使用してリソース設定とセキュリティ標準を評価し、お客様がさまざまなコンプライアンスフレームワークに準拠できるようサポートします。Security Hub を使用して Amazon EC2 Auto Scaling リソースを評価する方法の詳細については、「AWS Security Hub ユーザーガイド」の「Amazon EC2 Auto Scaling コントロール」を参照してください。

Amazon Simple Notification Service

Amazon EC2 Auto Scaling がインスタンスを起動または終了するときに Amazon SNS 通知を送信するように、Auto Scaling グループを設定できます。詳しくは、「Amazon EC2 Auto Scaling の Amazon SNS 通知オプション Amazon EC2 Auto Scaling」を参照してください。

Auto Scaling グループ内のインスタンスのヘルスチェック

Amazon EC2 Auto Scaling は、Auto Scaling グループ内のインスタンスのヘルスステータスを継続的にモニタリングして、必要な容量を維持します。

Auto Scaling グループ内のすべてのインスタンスは、 `Healthy`ステータスで始まります。インスタンスに異常があるという通知を Amazon EC2 Auto Scaling が受け取らない限り、インスタンスは正常であると見なされます。インスタンスが異常になり、置き換える必要がある場合、さまざまなソースから通知を受け取ることができます。こうしたソースには、以下が含まれます。

- Amazon EC2
- Elastic Load Balancing
- VPC Lattice
- 定義するカスタムヘルスチェック

Amazon EC2 Auto Scaling は、`InService`インスタンスが異常であると判断すると、そのインスタンスを新しいインスタンスに置き換えて、グループの希望する容量を維持します。新しいインスタンスは、Auto Scaling グループの現在の設定、およびそれに関連する起動テンプレートまたは起動設定を使用して起動します。

異常なインスタンスは、スポットインスタンスの中断やユーザーによる手動終了など、インスタンスが予期せず終了した場合にも発生する可能性があります。この場合も、Amazon EC2 Auto Scaling は希望する容量を維持するために、代替インスタンスを自動的に起動します。

内容

- Auto Scaling グループのヘルスチェックについて
- Auto Scaling グループにヘルスチェックの猶予期間を設定する
- ヘルスチェックが失敗した理由を表示する
- Amazon EC2 Auto Scaling の異常なインスタンスのトラブルシューティング

Auto Scaling グループのヘルスチェックについて

このトピックでは、利用可能なヘルスチェックタイプの概要と、Amazon EC2 Auto Scaling ヘルスチェックをアプリケーションに統合する際の主な考慮事項について説明します。

内容

- ヘルスチェックタイプ
- Amazon EC2 ヘルスチェック
- Elastic Load Balancing のヘルスチェック
- VPC Lattice ヘルスチェック
- Amazon EC2 Auto Scaling によってダウンタイムが最小限に抑えられる仕組み
- ウォームプール内のインスタンスのヘルスチェック
- ヘルスチェックの考慮事項
- カスタムヘルスチェック

ヘルスチェックタイプ

Amazon EC2 Auto Scaling は、次のヘルスチェックを 1 つ以上使用して、`InService`インスタンスのヘルスステータスを判断できます。

ヘルスチェックタイプ	チェックする事柄
Amazon EC2 ステータスチェックと予定されているイベント	• インスタンスが実行中であることを確認します。 • インスタンスを損なう可能性のある基盤となるハードウェアまたはソフトウェアの問題をチェックします。

ヘルスチェックタイプ	チェックする事柄
	これは、Auto Scaling グループに対するデフォルトのヘルスチェックタイプです。
Elastic Load Balancing のヘルスチェック	• ロードバランサーがインスタンスを正常と報告しているかどうかをチェックし、インスタンスがリクエストを処理できるかどうかを確認します。 このヘルスチェックタイプを実行するには、Auto Scaling グループに対して有効にする必要があります。
VPC Lattice ヘルスチェック	• VPC Lattice がインスタンスを正常と報告しているかどうかをチェックし、インスタンスがリクエストを処理できるかどうかを確認します。 このヘルスチェックタイプを実行するには、Auto Scaling グループに対して有効にする必要があります。
カスタムヘルスチェック	• カスタムヘルスチェックに従って、インスタンスのヘルス問題を示す可能性のあるその他の問題をチェックします。

Amazon EC2 ヘルスチェック

インスタンスが起動されると、インスタンスは Auto Scaling グループにアタッチされ、`InService` 状態になります。Auto Scaling グループ内のインスタンスの異なるライフサイクル状態に関する詳細については、「Amazon EC2 Auto Scaling インスタンスのライフサイクル」を参照してください。

Amazon EC2 Auto Scaling は、Auto Scaling グループ内のすべてのインスタンスのヘルスステータスを定期的にチェックすることで、それらが実行中で良好な状態であることを確認します。

ステータスチェック

Amazon EC2 Auto Scaling は、Amazon EC2 インスタンスのステータスチェックとシステムステータスチェックの結果を使用して、インスタンスのヘルスステータスを判断します。インスタンスが `running` 以外の Amazon EC2 状態である場合、またはステータスチェックのステータスが `impaired` になった場合、Amazon EC2 Auto Scaling はインスタンスを異常であると見なし、その

インスタンスを置き換えます。これには、インスタンスが以下のいずれかの状態にある場合が含まれます。

- `stopping`
- `stopped`
- `shutting-down`
- `terminated`

Amazon EC2 ステータスチェックに特別な設定は必要なく、常に有効になっています。詳細については、「Amazon EC2 ユーザーガイド」の「ステータスチェックのタイプ」を参照してください。 Amazon EC2

> **Important**
>
> Amazon EC2 Auto Scaling は、何のアクションも実行せずにステータスチェックを不合格にすることがあります。ステータスチェックが失敗すると、Amazon EC2 Auto Scaling は が問題を解決 AWS するまで数分待ちます。ステータスチェックのステータスが `impaired` になっても、インスタンスは直ちに `Unhealthy` としてマークされません。
> ただし、インスタンスが `running` 状態ではなくなったことを Amazon EC2 Auto Scaling が検出すると、この状況は即時不合格として扱われます。この場合、インスタンスをすぐに としてマークUnhealthyし、置き換えます。

予定されているイベント

Amazon EC2 は、インスタンスのイベントを、特定のタイムスタンプ後に実行されるようにスケジュールすることがあります。詳細については、「Amazon EC2 ユーザーガイド」の「インスタンスのスケジュールされたイベント」を参照してください。 Amazon EC2

インスタンスのいずれかが予定されているイベントの影響を受ける場合、Amazon EC2 Auto Scaling は、そのインスタンスを異常と見なして置き換えます。タイムスタンプで指定された日時に達するまで、インスタンスはシャットダウンを開始しません。

Elastic Load Balancing のヘルスチェック

Auto Scaling グループの Elastic Load Balancing ヘルスチェックを有効にすると、Amazon EC2 Auto Scaling はそれらのヘルスチェックの結果を使用してインスタンスのヘルスステータスを判断できます。

Auto Scaling グループの Elastic Load Balancing ヘルスチェックを有効にする前に、Elastic Load Balancing ロードバランサーを設定し、そのヘルスチェックを設定して、インスタンスが正常かどうかを判断する必要があります。詳細については、「Elastic Load Balancing ロードバランサーを Auto Scaling グループにアタッチする準備をします。」を参照してください。

ロードバランサーを Auto Scaling グループにアタッチすると、次のようになります。

- Amazon EC2 Auto Scaling が、Auto Scaling グループ内のインスタンスをロードバランサーに登録します。
- インスタンスの登録が終了すると、インスタンスは `InService` 状態になり、ロードバランサーで使用できるようになります。

デフォルトで、Amazon EC2 Auto Scaling は Elastic Load Balancing ヘルスチェックの結果を無視しますが、Auto Scaling グループのこれらのヘルスチェックを有効にすると、Elastic Load Balancing が登録済みインスタンスを としてレポートすると`Unhealthy`、Amazon EC2 Auto Scaling は次の定期的なヘルスチェック`Unhealthy`でインスタンスをマークし、置き換えます。

ロードバランサーに対して Connection Draining が有効になっている場合、Amazon EC2 Auto Scaling は、処理中のリクエストが完了するまで、または最大タイムアウト時間が終了するまで待機してから、異常なインスタンスを終了します。

> **Note**
>
> ロードバランサーをアタッチし、Auto Scaling グループの Elastic Load Balancing ヘルスチェックを有効にする方法については、「」を参照してくださいAuto Scaling グループに Elastic Load Balancing ロードバランサーをアタッチする Auto Scaling 。 Auto Scaling グループの Elastic Load Balancing ヘルスチェックを有効にすると、Amazon EC2 Auto Scaling は、ロードバランサーが `InService`状態になった後にのみ、Elastic Load Balancing が異常として報告するインスタンスを置き換えることができます。詳細については、「ロードバランサーのアタッチメントステータスを確認する」を参照してください。

VPC Lattice ヘルスチェック

デフォルトでは、Amazon EC2 Auto Scaling は、VPC Lattice ヘルスチェックの結果を無視します。オプションで、Auto Scaling グループのこれらのヘルスチェックを有効にできます。これらのヘルスチェックを有効化した後、VPC Lattice が登録されたインスタンスを `Unhealthy` として報告すると、Amazon EC2 Auto Scaling は、次回の定期ヘルスチェックでそのインスタンスを

Unhealthy としてマークし、置き換えます。インスタンスを登録してからその状態をチェックする処理は、Elastic Load Balancing ヘルスチェックの仕組みと同じです。

> **Note**
>
> VPC Lattice ターゲットグループをアタッチし、Auto Scaling グループの VPC Lattice ヘルスチェックを有効にする方法については、「」を参照してくださいVPC Lattice ターゲットグループを Auto Scaling グループにアタッチする。
> グループの VPC Lattice ヘルスチェックを有効にすると、Amazon EC2 Auto Scaling は、VPC Lattice が異常と報告するインスタンスを置き換えることができますが、ターゲットグループが InService状態になった後にのみ置き換えることができます。詳細については、「VPC Lattice ターゲットグループのアタッチメントステータスを確認する」を参照してください。

Amazon EC2 Auto Scaling によってダウンタイムが最小限に抑えられる仕組み

デフォルトでは、新しいインスタンスは既存のインスタンスが終了すると同時にプロビジョニングされるため、新しいインスタンスが完全に動作するまで新しいリクエストが受け入れられない可能性があります。

Amazon EC2 Auto Scaling は、インスタンスが実行されなくなった (または set-instance-health コマンドUnhealthyでマークされた) と判断した場合、それらを直ちに置き換えます。ただし、他のインスタンスが異常である場合、Amazon EC2 Auto Scaling は不合格状態からの回復に以下のアプローチを使用します。このアプローチは、一時的な問題、または誤設定されたヘルスチェックが原因で発生する可能性があるダウンタイムを最小限に抑えます。

- スケーリングアクティビティが進行中で、Auto Scaling グループが希望する容量を 10% 以上下回っている場合、Amazon EC2 Auto Scaling は進行中のスケーリングアクティビティを待ってから、異常なインスタンスを置き換えます。
- スケールアウト時、Amazon EC2 Auto Scaling はインスタンスが最初のヘルスチェックに合格するのを待ちます。また、デフォルトのインスタンスウォームアップが完了するのを待って、新しいインスタンスの準備が整っていることも確実にします。
- インスタンスのウォームアップが完了し、グループが希望する容量の 90% を超えると、Amazon EC2 Auto Scaling は異常のあるインスタンスを次のように置き換えます。
  - Amazon EC2 Auto Scaling が一度に置き換えるインスタンスはグループの希望容量の最大 10% のみで、異常なインスタンスのすべてが置き換えられるまで続行されます。

- インスタンスを置き換えるときは、新しいインスタンスが最初のヘルスチェックに合格するのを待ちます。また、デフォルトのインスタンスウォームアップが完了するのも待ち、完了後に続行します。

> **Note**
>
> Auto Scaling グループのサイズが 10% の結果値が 1 未満になるほど小さい場合、Amazon EC2 Auto Scaling は異常のあるインスタンスを一度に 1 つずつ置き換えます。これは、グループのダウンタイムの原因になる可能性があります。
> また、Auto Scaling グループ内のすべてのインスタンスが異常であると Elastic Load Balancing ヘルスチェックが報告し、ロードバランサーが `InService` 状態である場合、Amazon EC2 Auto Scaling は一度に異常としてマークするインスタンス数を減らすことがあります。そうすることで、他のシナリオに適用される 10% よりも、一度に置き換えられるインスタンスの数を大幅に削減することができます。これにより、Amazon EC2 Auto Scaling がグループ全体を自動的に終了することなく、問題を解決する時間を確保できます。

ウォームプール内のインスタンスのヘルスチェック

Amazon EC2 Auto Scaling は、ウォームプール内のインスタンスのヘルスチェックも実行します。詳細については、「ヘルスチェックのステータスとヘルスチェックの失敗理由を表示する」を参照してください。

ヘルスチェックの考慮事項

Amazon EC2 Auto Scaling ヘルスチェックを使用する際の考慮事項を次に示します。

- 終了処理中のインスタンス、または起動中のインスタンスで何かを実行する必要がある場合は、ライフサイクルフックを使用することができます。これらのフックを使用すると、Amazon EC2 Auto Scaling がインスタンスを起動または終了する時点で、カスタムアクションを実行できます。詳細については、「Amazon EC2 Auto Scaling のライフサイクルフック」を参照してください。
- Amazon EC2 Auto Scaling は、そのヘルスチェックから Amazon EC2 ステータスチェックと予定されているイベントを削除する方法を提供しません。インスタンスが置き換えられないようにしたい場合は、個々の Auto Scaling グループについて `ReplaceUnhealthy` プロセスと `HealthCheck` プロセスを停止することをお勧めします。詳細については、「Amazon EC2 Auto Scaling プロセスの一時停止と再開」を参照してください。

- 異常なインスタンスのヘルスステータスを手動で Healthy に戻す場合は、set-instance-health コマンドの使用を試みることができます。エラーが発生する場合、その原因はインスタンスが既に終了中であるためだと考えられます。通常、set-instance-health コマンドを使用してインスタンスのヘルスステータスを Healthy に戻すことは、ReplaceUnhealthy プロセスまたは Terminate プロセスが停止している場合にのみ有効です。
- ヘルスチェックの干渉なしにインスタンスのトラブルシューティングが必要な場合は、インスタンスを Standby状態にすることができます。Amazon EC2 Auto Scaling は、インスタンスを稼働状態に戻すまで、 Standby 状態にあるインスタンスのヘルスチェックを実行しません。詳細については、「Auto Scaling グループからインスタンスを一時的に削除する」を参照してください。
- インスタンスを削除すると、関連付けられたすべての Elastic IP アドレスは関連付けを解除され、新しいインスタンスと自動的に関連付けられることはありません。これらの Elastic IP アドレスと新しいインスタンスは、手動で関連付けるか、ライフサイクルフックベースのソリューションを使用して自動的に関連付ける必要があります。詳細については、「Amazon EC2 ユーザーガイド」の「Elastic IP アドレス」を参照してください。
- 同様に、インスタンスが終了されると、それにアタッチされている EBS ボリュームがデタッチ (または、ボリュームの DeleteOnTermination 属性に応じて削除) されます。これらの EBS ボリュームは、新しいインスタンスに手動でアタッチするか、ライフサイクルフックベースのソリューションを使用して自動的にアタッチする必要があります。詳細については、「Amazon EBS ユーザーガイド」の「インスタンスへの Amazon EBS ボリュームのアタッチ」を参照してください。

カスタムヘルスチェック

オプションで、Auto Scaling グループのインスタンスでカスタムヘルス検出タスクを実行し、タスクが失敗Unhealthyした場合にインスタンスのヘルスステータスを に設定することができます。これは、カスタムヘルスチェック、Amazon EC2 ステータスチェック、および Elastic Load Balancing ヘルスチェック (有効になっている場合) の組み合わせを使用することで、ヘルスチェックを拡張します。

インスタンスのヘルス情報は、 AWS CLI または SDK を使用して Amazon EC2 Auto Scaling に直接送信することができます。次の例は、 を使用してインスタンスのヘルスステータス AWS CLI を設定し、インスタンスのヘルスステータスを確認する方法を示しています。

次の set-instance-health コマンドを使用して、指定したインスタンスのヘルスステータスを **Unhealthy** に設定します。

```
aws autoscaling set-instance-health --instance-id i-1234567890abcdef0 --health-
status Unhealthy
```

デフォルトで、このコマンドはヘルスチェックの猶予期間を守りますが、--no-should-respect-grace-period オプションを含めることでこの動作を上書きし、猶予期間を守らないようにすることも可能です。

次の describe-auto-scaling-groups コマンドを使用して、インスタンスのヘルスステータスが Unhealthy であることを確認します。

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-names my-asg
```

次に示すのは、インスタンスのヘルスステータスが Unhealthy であり、インスタンスが終了中であることを示す応答の例です。

```
{
    "AutoScalingGroups": [
        {
            ....
            "Instances": [
                {
                    "ProtectedFromScaleIn": false,
                    "AvailabilityZone": "us-west-2a",
                    "LaunchTemplate": {
                        "LaunchTemplateName": "my-launch-template",
                        "Version": "1",
                        "LaunchTemplateId": "lt-1234567890abcdef0"
                    },
                    "InstanceId": "i-1234567890abcdef0",
                    "InstanceType": "t2.micro",
                    "HealthStatus": "Unhealthy",
                    "LifecycleState": "Terminating"
                },
                ...
            ]
        }
    ]
}
```

Auto Scaling グループにヘルスチェックの猶予期間を設定する

Amazon EC2 Auto Scaling ヘルスチェックで`InService`インスタンスが異常であると判断された場合、インスタンスは新しいインスタンスに置き換えられます。ヘルスチェックの猶予期間は、新しいインスタンスに異常がある場合、そのインスタンスを終了するまでに稼働する最小時間 (秒単位) を指定します。

ユースケース例としては、Elastic Load Balancing のヘルスチェックが失敗し、インスタンスがまだ初期化していることが原因である場合、Amazon EC2 Auto Scaling が実行を回避する要件があります。Elastic Load Balancing のヘルスチェックは、インスタンスがロードバランサーに登録されたときに開始して並行で実行されます。猶予期間により、Amazon EC2 Auto Scaling は、新しく起動したインスタンスをマーク`Unhealthy`し、`InService`状態になった後にこれらのヘルスチェックにすぐに合格しなかった場合に、不必要に終了することを防ぎます。

コンソールでは、Auto Scaling グループを作成するときのヘルスチェックの猶予期間がデフォルトで 300 秒になっています。 AWS CLI または SDK を使用して Auto Scaling グループを作成する場合、デフォルト値は 0 秒です。値が 0 の場合、ヘルスチェックの猶予期間は無効になります。

この値を高く設定しすぎると、Amazon EC2 Auto Scaling ヘルスチェックの効果が低減します。インスタンスの起動にライフサイクルフックを使用する場合は、ヘルスチェックの猶予期間の値を 0 に設定できます。ライフサイクルフックを使用すると、Amazon EC2 Auto Scaling は、インスタンスが常に初期化されてから `InService` 状態になることを確実にするための手段を提供します。詳細については、「Amazon EC2 Auto Scaling のライフサイクルフック」を参照してください。

猶予期間は以下のインスタンスに適用されます。

- 新しく起動されたインスタンス
- スタンバイ状態になった後で実行状態に戻されるインスタンス
- グループに手動でアタッチされるインスタンス

> **Important**
>
> ヘルスチェックの猶予期間中に Amazon EC2 Auto Scaling が Amazon EC2 `running` 状態ではなくなったインスタンスを検出した場合は、直ちにそのインスタンスを `Unhealthy` としてマークし、置き換えます。例えば、Auto Scaling グループ内のインスタンスを停止すると、そのインスタンスは `Unhealthy` とマークされ、置き換えられます。

グループにヘルスチェックの猶予期間を設定する

ヘルスチェックの猶予期間は、新規または既存の Auto Scaling グループに設定できます。

Console

新しいグループのヘルスチェックの猶予期間を変更するには

Auto Scaling グループを作成するときは、「詳細オプションの設定」ページ、「ヘルスチェック」、「ヘルスチェック猶予期間」に時間 (秒単位) を入力します。これは、Amazon EC2 Auto Scaling が `InService`状態になった後にインスタンスのヘルスステータスをチェックするまで待機する必要がある時間です。

AWS CLI

新しいグループのヘルスチェックの猶予期間を変更するには

create-auto-scaling-group コマンドに `--health-check-grace-period` オプションを追加します。以下の例は、*`my-asg`* という名前の新しい Auto Scaling グループに対するヘルスチェック猶予期間を *`60`* 秒の値で設定します。

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-asg \
  --health-check-grace-period 60 ...
```

Console

既存のグループのヘルスチェックの猶予期間を変更するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. 画面の上部のナビゲーションバーで、Auto Scaling グループを作した AWS リージョン を選択します。
3. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。
4. [詳細] タブで、[ヘルスチェック]、[編集] の順に選択します。
5. [Health check grace period] (ヘルスチェックの猶予期間) に、秒単位で時間を入力します。これは、Amazon EC2 Auto Scaling が `InService`状態になった後にインスタンスのヘルスステータスをチェックするまで待機する必要がある時間です。

6. [更新] を選択します。

AWS CLI

既存のグループのヘルスチェックの猶予期間を変更するには

update-auto-scaling-group コマンドに `--health-check-grace-period` オプションを追加します。以下の例は、*my-asg* という名前の既存の Auto Scaling グループに対するヘルスチェック猶予期間を *120* 秒の値で設定します。

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --health-check-grace-period 120
```

> **Note**
>
> Auto Scaling グループに対してデフォルトのインスタンスウォームアップ時間を設定することも強く推奨されます。詳細については、「Auto Scaling グループに対するインスタンスのデフォルトウォームアップを設定する」を参照してください。

ヘルスチェックが失敗した理由を表示する

次の手順を使用して、ヘルスチェックによって置き換えられたインスタンスに関する情報を表示できます。

デフォルトでは、Amazon EC2 Auto Scaling は異常のあるインスタンスを終了するための新しいスケーリングアクティビティを作成し、終了します。インスタンスの終了中、別のスケーリングアクティビティが新しいインスタンスを起動します。この動作を変更して、インスタンスのメンテナンスポリシーを使用して、できるだけ早く新しいインスタンスの起動を開始できます。詳細については、「インスタンスのメンテナンスポリシー」を参照してください。

Console

ヘルスチェックが失敗した理由の表示

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

[Auto Scaling groups] (Auto Scaling グループ) ページの下部にスプリットペインが開きます。

3. [Activity (アクティビティ)] タブの [Activity history (アクティビティ履歴)] の下の [Status (ステータス)] 列に、Auto Scaling グループがインスタンスを正常に起動したか、終了したかが表示されます。

   正常でないインスタンスを終了した場合、原因列には、終了の日時、およびヘルスチェックが失敗した理由が表示されます。例えば `At 2022-05-14T20:11:53Z an instance was taken out of service in response to a user health-check` です。このメッセージは、カスタムヘルスチェックによってインスタンスに異常がマークされたことを示します。

   ヘルスチェックの失敗については、「」を参照してくださいAmazon EC2 Auto Scaling の異常なインスタンスのトラブルシューティング。

AWS CLI

ヘルスチェックが失敗した理由の表示

以下の describe-scaling-activities コマンドを実行します。

```
aws autoscaling describe-scaling-activities --auto-scaling-group-name my-asg
```

以下はレスポンスの例です。 にはヘルスチェックが失敗した理由Causeが含まれています。

```
{
  "Activities": [
    {
      "ActivityId": "4c65e23d-a35a-4e7d-b6e4-2eaa8753dc12",
      "AutoScalingGroupName": "my-asg",
      "Description": "Terminating EC2 instance: i-04925c838b6438f14",
      "Cause": "At 2021-04-01T21:48:35Z an instance was taken out of service in
 response to a user health-check.",
      "StartTime": "2021-04-01T21:48:35.859Z",
      "EndTime": "2021-04-01T21:49:18Z",
      "StatusCode": "Successful",
      "Progress": 100,
      "Details": "{\"Subnet ID\":\"subnet-5ea0c127\",\"Availability Zone\":\"us-
west-2a\"...}",
```

```
      "AutoScalingGroupARN": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:283179a2-
f3ce-423d-93f6-66bb518232f7:autoScalingGroupName/my-asg"
    },
...
  ]
}
```

出力の各フィールドについては、「Amazon EC2 Auto Scaling API Reference」(Amazon EC2 Auto Scaling API リファレンス) の「Activity」(アクティビティ) を参照してください。

Auto Scaling グループが削除された後のスケーリングアクティビティを記述するには、describe-scaling-activities コマンドに `--include-deleted-groups`オプションを追加します。

Amazon EC2 Auto Scaling の異常なインスタンスのトラブルシューティング

以下は、Amazon EC2 Auto Scaling によって返されるエラーメッセージ、考えられる原因、および問題を解決するために実行できる手順です。

エラーメッセージを取得するには、「ヘルスチェックが失敗した理由を表示する」を参照してください。

エラーメッセージ

- EC2 インスタンスのステータスチェックが失敗したことにより、インスタンスのサービスは停止されました。
- EC2 ヘルスチェックでインスタンスが終了または停止済みであることが検出され、インスタンスのサービスが停止されてしまう
- ELB システムのヘルスチェックが障害を検出してインスタンスのサービスが停止される
- 追加リソース

EC2 インスタンスのステータスチェックが失敗したことにより、インスタンスのサービスは停止されました。

問題: Auto Scaling インスタンスで Amazon EC2 のステータスチェックが失敗します。

原因 1: Amazon EC2 が Auto Scaling グループ内のインスタンスに障害が発生したと見なす問題がある場合、Amazon EC2 Auto Scaling はヘルスチェックの一部としてインスタンスを自動的に置き換えます。

解決策 1: インスタンスのステータスチェックが失敗した場合、通常、アプリケーションに問題がなくなるまでインスタンス設定を変更して、自分で問題に対処する必要があります。この問題を解決するには、以下の手順を実行します。

1. Auto Scaling グループには含まれない Amazon EC2 インスタンスを手動で作成して、問題を調査します。障害が発生したインスタンスの調査に関する一般的なヘルプについては、Amazon EC2 ユーザーガイド」の「ステータスチェックに失敗したインスタンスのトラブルシューティング」およびAmazon EC2 ユーザーガイド」の「Windows インスタンスのトラブルシューティング」を参照してください。
2. インスタンスが起動に成功し、正常に機能していることを確認したら、エラーのない新しいインスタンス設定を Auto Scaling グループにデプロイします。
3. AWS アカウントへの課金が継続されないよう、作成したインスタンスを削除します。

EC2 ヘルスチェックでインスタンスが終了または停止済みであることが検出され、インスタンスのサービスが停止されてしまう

問題: 停止、再起動、または終了した Auto Scaling インスタンスは置き換えられます。

原因 1: ユーザーが手動で、インスタンスを停止、再起動、または終了しています。

解決策 1: Auto Scaling グループのインスタンスを停止または再起動する必要がある場合は、まずインスタンスをスタンバイ状態にすることをお勧めします。詳細については、「Auto Scaling グループからインスタンスを一時的に削除する」を参照してください。

原因 2: Amazon EC2 Auto Scaling は、Amazon EC2 スポットサービスがインスタンスを中断した後、スポットインスタンスの置き換えを試みます。これは、スポット料金がお客様の上限価格を超えるか、またはキャパシティーが使用できなくなるためです。

解決策 2: 特定の時点で要求を満たせるだけのスポットインスタンスが存在するという保証はありません。ただし、以下を試すことができます。

- より高いスポット上限価格を設定します (オンデマンド料金が利用できます)。上限価格を高く設定することで、要求されたキャパシティーを Amazon EC2 スポットサービスが確保し、それを維持できる可能性を高められます。

- 複数のアベイラビリティーゾーンで複数のインスタンスタイプを実行することで、より多くの異なるキャパシティプールからインスタンスを起動できるようにします。詳細については、「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。
- 複数のインスタンスタイプを使用する場合は、キャパシティーの再調整機能を有効にすることも考慮してください。この機能は、実行中のインスタンスが終了する前に Amazon EC2 スポットサービスで、新しいスポットインスタンスを起動させたい場合に便利です。詳細については、「キャパシティの再調整を使用して Amazon EC2 スポットの中断に対処する」を参照してください。

原因 3: キャパシティブロックでは、Amazon EC2 はキャパシティブロックの終了時刻の 30 分前にまだ実行中のインスタンスをすべて終了します。この突然の終了により、Auto Scaling グループは、キャパシティブロックが終了している場合でも、希望するキャパシティを維持するために新しいインスタンスを起動しようとします。

解決策 3: この問題を解決するには、以下を試してください。

- Auto Scaling グループの希望する容量を減らして、新しいインスタンスを起動しようとしないようにします。詳細については、「Amazon EC2 Auto Scaling の手動スケーリング」を参照してください。
- このエラーが頻繁に発生しないように、キャパシティブロックの終了時刻の 30 分前に Auto Scaling グループをスケールインしてください。キャパシティブロックの終了時刻の 30 分前にライフサイクルフックが完了していることを確認します。詳細については、「機械学習ワークロード Capacity Blocksに を使用する」を参照してください。

ELB システムのヘルスチェックが障害を検出してインスタンスのサービスが停止される

問題: Auto Scaling インスタンスが EC2 ステータスチェックに合格する場合があります。ただし、Auto Scaling グループが登録されている、ターゲットグループや Classic Load Balancer に対するElastic Load Balancing のヘルスチェックで失敗する可能性があります。

原因 1: Auto Scaling グループが Elastic Load Balancing によって提供されるヘルスチェックに依存している場合、Amazon EC2 Auto Scaling は EC2 ステータスチェックと Elastic Load Balancing ヘルスチェックの両方の結果をチェックして、インスタンスのヘルスステータスを判断します。ロードバランサーは、各インスタンスにリクエストを送信して正しい応答を待つか、インスタンスとの接続の確立を試みることで、ヘルスチェックを実行します。インスタンスで実行するアプリケーションに問題があり、ロードバランサーがそのインスタンスを停止中と判断する場合、そのインスタンスは Elastic Load Balancing のヘルスチェックに失敗する場合があります。

解決策 1Elastic Load Balancing のヘルスチェックに合格するには:

- ターゲットグループのヘルスチェックにおいて、適切な設定が行われていることを確認します。ロードバランサーのヘルスチェック設定は、ターゲットグループごとに定義します。詳細については、「ターゲットのヘルスチェックを設定する」を参照してください。
- ロードバランサーで予期される成功コードを記録し、成功時にアプリケーションがそれらのコードを返すかを見て、適切な設定が行われていることを確認します。
- ロードバランサーと Auto Scaling グループのセキュリティグループでの設定が適切であることを確認します。
- ロードバランサーが Auto Scaling グループと同じアベイラビリティーゾーンに構成されていることを確認します。

解決策 2: Auto Scaling グループを更新して、Elastic Load Balancing のヘルスチェックを無効にします。これらのヘルスチェックを無効にする方法については、「」を参照してくださいAuto Scaling グループに Elastic Load Balancing ロードバランサーをアタッチする Auto Scaling 。

原因 2: ヘルスチェックの猶予期間とインスタンスのスタートアップ時間の間に不一致があります。

解決策 3: Auto Scaling グループのヘルスチェックの猶予期間を編集します。猶予期間を、Elastic Load Balancing が新しく起動されたインスタンスが正常であると判断するまでに必要なヘルスチェックの連続成功回数をサポートするのに十分な期間に設定します。詳細については、「Auto Scaling グループにヘルスチェックの猶予期間を設定する」を参照してください。

追加リソース

別の問題が発生した場合は、次の AWS re:Post 記事でトラブルシューティングに関するその他のヘルプを参照してください。

- インスタンスが Amazon EC2 Auto Scaling により終了された理由は何ですか?
- 異常なインスタンスが Amazon EC2 Auto Scaling により終了されない理由は何ですか?

AWS Health Dashboard Amazon EC2 Auto Scaling に関する通知

AWS Health Dashboard は Amazon EC2 Auto Scaling から送信される通知をサポートしています。これらの通知は、アプリケーションに影響を与える可能性のあるリソースのパフォーマンスや可用性の問題の把握と修復のガイダンスを提供します。現在使用できるのは、存在しないセキュリティグループおよび起動テンプレートに固有のイベントのみです。

AWS Health Dashboard AWS Health はサービスの一部です。セットアップは一切必要なく、アカウントで認証されるすべてのユーザーが表示できます。詳細については、「AWS Health ダッシュボード入門」を参照してください。

次のようなメッセージを受信した場合は、アクションを実行するためのアラームとして処理する必要があります。

例: 欠落したセキュリティグループが原因で Auto Scaling グループがスケールアウトされていない

```
 Hello,

 At 2020-01-11 04:00 UTC, we detected an issue with your Auto Scaling group [ARN] in
AWS ##### 123456789012.

 A security group associated with this Auto Scaling group cannot be found. Each time
a
 scale out operation is performed, it will be prevented until you make a change that
 fixes the issue.

 We recommend that you review and update your Auto Scaling group configuration to
change
 the launch template or launch configuration that depends on the unavailable security
 group.

 Sincerely,
 Amazon Web Services
```

例: 欠落した起動テンプレートが原因で Auto Scaling グループがスケールアウトされていない

```
 Hello,

 At 2021-05-11 04:00 UTC, we detected an issue with your Auto Scaling group [ARN] in
AWS ##### 123456789012.

 The launch template associated with this Auto Scaling group cannot be found. Each
time
 a scale out operation is performed, it will be prevented until you make a change
that
 fixes the issue.

 We recommend that you review and update your Auto Scaling group configuration and
 specify an existing launch template to use.
```

```
Sincerely,
Amazon Web Services
```

Auto Scaling グループとインスタンスの CloudWatch メトリクスをモニタリングする

メトリクスは Amazon の基本的な概念です CloudWatch。メトリクスは、 に発行される時系列のデータポイントのセットを表します CloudWatch。メトリクスはモニターリング対象の変数と考え、データポイントは時間の経過と共に変数の値を表します。これらのメトリクスを使用して、システムが正常に実行されていることを確認できます。

Auto Scaling グループに関する情報を収集する Amazon EC2 Auto Scaling メトリクスは、`AWS/AutoScaling` 名前空間にあります。Auto Scaling インスタンスから CPU やその他の使用状況データを収集する Amazon EC2 インスタンスメトリクスは、`AWS/EC2` 名前空間にあります。

Amazon EC2 Auto Scaling コンソールには、グループメトリクスとグループの集計インスタンスメトリクスの一連のグラフが表示されます。必要に応じて、Amazon EC2 Auto Scaling コンソール CloudWatch ではなく、Amazon から Auto Scaling グループとインスタンスのデータにアクセスすることもできます。 Amazon EC2

詳細については、「Amazon ユーザーガイド CloudWatch 」を参照してください。

内容

- Amazon EC2 Auto Scaling コンソールでモニタリンググラフを表示する
- Amazon EC2 Auto Scaling の Amazon CloudWatch メトリクス Auto Scaling
- Auto Scaling インスタンスのモニタリングを設定する

Amazon EC2 Auto Scaling コンソールでモニタリンググラフを表示する

Amazon EC2 コンソールの Amazon EC2 Auto Scaling セクションでは、 CloudWatch メトリクスを使用して個々の Auto Scaling グループの進行状況をモニタリング minute-by-minuteできます。

次のタイプのメトリクスをモニタリングできます。

- Auto Scaling メトリクス – Auto Scaling メトリクスは、これを有効にした場合にのみオンになります。詳細については、「Auto Scaling グループのメトリクスを有効にする (コンソール)」を参照

してください。Auto Scaling メトリクスが有効になっている場合、モニタリンググラフには Auto Scaling メトリクスについて 1 分単位で発行されたデータが表示されます。

- EC2 メトリクス – Amazon EC2 インスタンスメトリクスは常に有効になっています。詳細モニタリングが有効になっている場合、モニタリンググラフには、インスタンスメトリクスについて 1 分単位で発行されたデータが表示されます。詳細については、「Auto Scaling インスタンスのモニタリングを設定する」を参照してください。

Amazon EC2 Auto Scaling コンソールを使用してモニタリンググラフを表示するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. メトリクスを表示する Auto Scaling グループの横にあるチェックボックスをオンにします。

   [Auto Scaling グループ] ページの下部に分割ペインが開き、グループに関する情報が表示されます。
3. モニタリングタブを選択します。

   Amazon EC2 Auto Scaling は、[Auto Scaling] メトリクスのモニタリンググラフを表示します。
4. グループの集計インスタンスメトリクスのモニタリンググラフを表示するには、[EC2] を選択します。

グラフアクション

- データポイントにカーソルを合わせると、特定の UTC 時刻のデータがポップアップ表示されます。
- グラフを拡大するには、グラフの右上にあるメニューツール (縦の 3 ドット) から [Enlarge] (拡大する) 選択します。または、グラフの上部にある最大化アイコンをクリックします。
- 定義済みの期間の値を 1 つ選択して、グラフに表示されるデータの期間を調整します。グラフが拡大されている場合は、[Custom] (カスタム) を選択して独自の期間を定義できます。
- メニューツールから [Refresh] (更新) を選択して、グラフのデータを更新します。
- グラフデータの上でカーソルをドラッグし、特定の範囲を選択します。次に、メニューツールで [Apply time range] (時間範囲の適用) を選択します。
- メニューツールからログを表示 を選択して、関連するログストリーム (存在する場合) を CloudWatch コンソールで表示します。

- でグラフを表示するには CloudWatch、メニューツールからメトリクスで表示を選択します。これにより、そのグラフの CloudWatch ページが表示されます。ここでは、Auto Scaling グループが長期にわたってどのように変化したかをより深く理解するために、詳細情報を表示したり、履歴情報にアクセスしたりできます。

Auto Scaling グループのメトリクスをグラフ表示する

Auto Scaling グループを作成したら、Amazon EC2 Auto Scaling コンソールを開き、[Monitoring] (モニタリング) タブにグループのモニタリンググラフを表示できます。

[Auto Scaling] セクションのグラフメトリクスには、次のメトリクスが含まれます。これらのメトリクスは、終了インスタンス数や保留中のインスタンスの数など、潜在的な問題の指標となる測定値を提供します。これらのメトリクスの定義については、「Amazon EC2 Auto Scaling の Amazon CloudWatch メトリクス Auto Scaling」を参照してください。

Display name (表示名)	CloudWatch メトリクス名
最小グループサイズ	`GroupMinSize`
最大グループサイズ	`GroupMaxSize`
希望するキャパシティー	`GroupDesiredCapacity`
稼働中のインスタンス	`GroupInServiceInstances`
保留中のインスタンス	`GroupPendingInstances`
スタンバイインスタンス	`GroupStandbyInstances`
終了処理中のインスタンス	`GroupTerminatingInstances`
インスタンスの合計	`GroupTotalInstances`

[EC2] セクションで、Amazon EC2 インスタンスの主要なパフォーマンスメトリクスに基づく、次のグラフメトリクスを確認できます。これらの EC2 メトリクスは、グループ内のすべてのインスタンスのメトリクスの集計です。これらのメトリクスの定義は、Amazon EC2 ユーザーガイド」の「インスタンスで使用できる CloudWatch メトリクスのリスト」を参照してください。

Display name (表示名)	CloudWatch メトリクス名
CPU 使用率	CPUUtilization
ディスク読み取り数	DiskReadBytes
ディスク読み取りオペレーション数	DiskReadOps
ディスク書き込み数	DiskWriteBytes
ディスク書き込みオペレーション数	DiskWriteOps
ネットワーク入力	NetworkIn
ネットワーク出力	NetworkOut
ステータスチェックに失敗 (すべて)	StatusCheckFailed
ステータスチェックに失敗 (インスタンス)	StatusCheckFailed_Instance
ステータスチェックに失敗 (システム)	StatusCheckFailed_System

さらに、Auto Scaling グラフメトリクスには、特定のユースケースに使用できるメトリクスがいくつかあります。

以下のメトリクスは、各インスタンスがグループの希望するキャパシティに寄与するユニット数を定義する重みをグループで使用する場合に役立ちます。これらのメトリクスの定義については、「Amazon EC2 Auto Scaling の Amazon CloudWatch メトリクス Auto Scaling」を参照してください。

Display name (表示名)	CloudWatch メトリクス名
稼働中キャパシティのユニット数	GroupInServiceCapacity
保留中キャパシティのユニット数	GroupPendingCapacity
スタンバイ中キャパシティのユニット数	GroupStandbyCapacity
終了処理中キャパシティのユニット数	GroupTerminatingCapacity
キャパシティのユニット数合計	GroupTotalCapacity

次のメトリクスは、グループでウォームプール機能を使用する場合に役立ちます。これらのメトリクスの定義については、「Amazon EC2 Auto Scaling の Amazon CloudWatch メトリクス Auto Scaling」を参照してください。

Display name (表示名)	CloudWatch メトリクス名
ウォームプールの最小サイズ	WarmPoolMinSize
ウォームプールの希望するキャパシティー	WarmPoolDesiredCapacity
ウォームプールの保留中キャパシティのユニット数	WarmPoolPendingCapacity
ウォームプールの終了処理中キャパシティのユニット数	WarmPoolTerminatingCapacity
ウォームプールのウォームアップされたキャパシティのユニット数	WarmPoolWarmedCapacity

Display name (表示名)	CloudWatch メトリクス名
ウォームプールの起動したキャパシティのユニット数合計	`WarmPoolTotalCapacity`
グループおよびウォームプールの希望するキャパシティー	`GroupAndWarmPoolDesiredCapacity`
グループおよびウォームプールの起動したキャパシティのユニット数合計	`GroupAndWarmPoolTotalCapacity`

関連リソース

- インスタンスごとのメトリクスをモニタリングするには、「Amazon EC2 ユーザーガイド」の「インスタンスのメトリクスをグラフ化する」を参照してください。 Amazon EC2
- CloudWatch ダッシュボードは、 CloudWatch コンソールでカスタマイズ可能なホームページです。これらのページを使用して、異なるリージョンにまたがるリソースも含めて、単一のビューでリソースをモニタリングできます。 CloudWatch ダッシュボードを使用して、 AWS リソースのメトリクスとアラームのカスタマイズされたビューを作成できます。詳細については、「Amazon ユーザーガイド CloudWatch 」を参照してください。

Amazon EC2 Auto Scaling の Amazon CloudWatch メトリクス Auto Scaling

Amazon EC2 Auto Scaling は `AWS/AutoScaling` 名前空間に以下のメトリクスを公開します。実際に使用できる Auto Scaling グループメトリクスは、グループメトリクスを有効にしているかどうか、およびどのグループメトリクスを有効にしたかによって異なります。グループメトリクスは、追加料金なしで 1 分単位で利用できますが、有効にする必要があります。

Auto Scaling グループメトリクスを有効にすると、Amazon EC2 Auto Scaling はベストエフォートベースで 1 分 CloudWatch ごとにサンプリングされたデータを に送信します。まれに、 でサービスの中断 CloudWatch が発生した場合、グループメトリクス履歴のギャップを埋めるためにデータがバックフィルされません。

内容

- Auto Scaling グループメトリクス
- Auto Scaling グループメトリクスのディメンション
- 予測スケーリングのメトリクスとディメンション
- Auto Scaling グループのメトリクスを有効にする (コンソール)
- Auto Scaling グループのメトリクスを有効にする (AWS CLI)

Auto Scaling グループメトリクス

これらのメトリクスを使用して、グループサイズの経時変化など、Auto Scaling グループの履歴をほぼ継続的に把握することができます。

メトリクス	説明
`GroupMinSize`	Auto Scaling グループの最小サイズ。 レポート基準: メトリクス収集が有効になっている場合に報告されます。
`GroupMaxSize`	Auto Scaling グループの最大サイズ。 レポート基準: メトリクス収集が有効になっている場合に報告されます。
`GroupDesiredCapacity`	Auto Scaling グループが保持しようとするインスタンスの数。 レポート基準: メトリクス収集が有効になっている場合に報告されます。
`GroupInServiceInstances`	Auto Scaling グループの一部として実行するインスタンスの数。このメトリクスには保留中もしくは終了処理中のインスタンスは含まれません。 レポート基準: メトリクス収集が有効になっている場合に報告されます。
`GroupPendingInstances`	保留中のインスタンスの数。保留中のインスタンスは、稼働状態ではありません。このメトリクスには稼働中もしくは終了処理中のインスタンスは含まれません。

メトリクス	説明
	レポート基準: メトリクス収集が有効になっている場合に報告されます。
GroupStandbyInstances	Standby 状態にあるインスタンスの数。この状態のインスタンスはまだ実行中ですが、実際には使用されていません。 レポート基準: メトリクス収集が有効になっている場合に報告されます。
GroupTerminatingInstances	終了処理中のインスタンスの数。このメトリクスには稼働中もしくは保留中のインスタンスは含まれません。 レポート基準: メトリクス収集が有効になっている場合に報告されます。
GroupTotalInstances	Auto Scaling グループに含まれるインスタンスの合計数。このメトリクスは稼働中、保留中、および終了処理中のインスタンスの数を特定します。 レポート基準: メトリクス収集が有効になっている場合に報告されます。

各インスタンスタイプの vCPU 数に基づいて重みを割り当てるなど、さまざまなユニットで希望するキャパシティを測定するように混合インスタンスグループを設定すると、次のメトリクスは Auto Scaling グループによって使用されるユニット数をカウントします。希望するキャパシティをさまざまなユニットで測定するように混合インスタンスグループを設定しなかった場合、次のメトリクスが事前入力されますが、これらは前の表で定義されているメトリクスと同じです。詳細については、「設定の概要」を参照してください。

メトリクス	説明
GroupInServiceCapacity	Auto Scaling グループの一部として実行されているキャパシティユニットの数。 レポート基準: メトリクス収集が有効になっている場合に報告されます。

メトリクス	説明
GroupPendingCapacity	保留中のキャパシティユニットの数。 レポート基準: メトリクス収集が有効になっている場合に報告されます。
GroupStandbyCapacity	Standby 状態にあるキャパシティユニットの数。 レポート基準: メトリクス収集が有効になっている場合に報告されます。
GroupTerminatingCapacity	終了処理中のキャパシティユニットの数。 レポート基準: メトリクス収集が有効になっている場合に報告されます。
GroupTotalCapacity	Auto Scaling グループ内のキャパシティユニットの合計数。 レポート基準: メトリクス収集が有効になっている場合に報告されます。

Amazon EC2 Auto Scaling は、ウォームプールを持つ Auto Scaling グループの以下のメトリクスもレポートします。詳細については、「Amazon EC2 Auto Scaling のウォームプール」を参照してください。

メトリクス	説明
WarmPoolMinSize	ウォームプールの最小サイズ。 レポート基準: メトリクス収集が有効になっている場合に報告されます。
WarmPoolDesiredCapacity	Amazon EC2 Auto Scaling がウォームプールで維持しようとするキャパシティの量。 これは、Auto Scaling グループの最大サイズから希望するキャパシティを引いた値に相当します。また、設定されている場合

メトリクス	説明
	は、Auto Scaling グループの準備されている最大キャパシティから希望するキャパシティを引いた値に相当します。 ただし、ウォームプールの最小サイズが、最大サイズ (または設定されている場合は、準備された最大キャパシティ) と Auto Scaling グループの希望するキャパシティの差以上である場合、希望するウォームプールのキャパシティは `WarmPoolMinSize` に等しくなります。 レポート基準: メトリクス収集が有効になっている場合に報告されます。
`WarmPoolPendingCapacity`	保留中のウォームプール内のキャパシティ量。このメトリクスには実行中、停止中、または終了処理中のインスタンスは含まれません。 レポート基準: メトリクス収集が有効になっている場合に報告されます。
`WarmPoolTerminatingCapacity`	終了処理中のウォームプールのキャパシティの量。このメトリクスには実行中、停止中、または保留中のインスタンスは含まれません。 レポート基準: メトリクス収集が有効になっている場合に報告されます。
`WarmPoolWarmedCapacity`	スケールアウト中に Auto Scaling グループに入れることができるキャパシティの量。このメトリクスには保留中もしくは終了処理中のインスタンスは含まれません。 レポート基準: メトリクス収集が有効になっている場合に報告されます。
`WarmPoolTotalCapacity`	実行中、停止中、保留中、または終了処理中のインスタンスを含む、ウォームプールの合計キャパシティ。 レポート基準: メトリクス収集が有効になっている場合に報告されます。

メトリクス	説明
GroupAndWarmPoolDesiredCapacity	Auto Scaling グループとウォームプールを合わせた希望するキャパシティ。 レポート基準: メトリクス収集が有効になっている場合に報告されます。
GroupAndWarmPoolTotalCapacity	Auto Scaling グループとウォームプールを合わせた合計キャパシティ。これには、実行中、停止中、保留中、終了処理中、または稼働中のインスタンスが含まれます。 レポート基準: メトリクス収集が有効になっている場合に報告されます。

Auto Scaling グループメトリクスのディメンション

以下のディメンションを使用して、前の表に示したメトリクスを絞り込むことができます。

ディメンション	説明
AutoScalingGroupName	Auto Scaling グループの名前をフィルターします。

予測スケーリングのメトリクスとディメンション

AWS/AutoScaling 名前空間には、予測スケーリングに関する以下のメトリクスが含まれます。

メトリクスは、1 時間の解像度で使用できます。

予測値を実際の値と比較することで、予測精度を評価できます。これらのメトリクスを使用した予測精度の評価についての詳細は、「による予測スケーリングメトリクスのモニタリング CloudWatch」(CloudWatch で予測スケーリングメトリクスをモニタリングする) を参照してください。

メトリクス	説明	ディメンション
PredictiveScalingLoadForecast	アプリケーションによって生成されることが予想される負荷の量。	AutoScalingGroupName ,

メトリクス	説明	ディメンション
	Average、Minimum、Maximum の統計は有用ですが、Sum の統計は有用ではありません。 レポート基準: 初期予測の作成後にレポートされます。	PolicyName , PairIndex
PredictiveScalingCapacityForecast	アプリケーションの需要を満たすために必要であると予想されるキャパシティ。これは、Auto Scaling インスタンスを維持するための負荷予測と目標使用率レベルに基づいています。 Average、Minimum、Maximum の統計は有用ですが、Sum の統計は有用ではありません。 レポート基準: 初期予測の作成後にレポートされます。	AutoScalingGroupName , PolicyName
PredictiveScalingMetricPairCorrelation	スケーリング指標と負荷指標のインスタンスごとの平均の間における相関関係。予測スケーリングは高い相関関係を前提とします。したがって、この指標の値が小さい場合、指標ペアを使用しない方がよいでしょう。 Average、Minimum、Maximum の統計は有用ですが、Sum の統計は有用ではありません。 レポート基準: 初期予測の作成後にレポートされます。	AutoScalingGroupName , PolicyName , PairIndex

> **Note**
>
> PairIndex ディメンションは、Amazon EC2 Auto Scaling によって割り当てられた際に、負荷スケーリングメトリクスペアのインデックスに関連付けられた情報を返します。現在、有効な値は 0 のみです。

Auto Scaling グループのメトリクスを有効にする (コンソール)

グループメトリクスを有効にするには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。
3. [Monitoring] (モニタリング) タブで、ページ上部の [Auto Scaling] の下にある [Auto Scaling group metrics collection] (Auto Scaling グループのメトリクスのコレクション) を選択し、[Enable] (有効化) チェックボックスをオンにします。

グループメトリクスを無効にするには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling スケーリンググループを選択します。
3. [モニタリング] タブで、[Auto Scaling group metrics collection (Auto Scaling グループメトリクスの収集)] の [Enable (有効)] チェックボックスをオフにします。

Auto Scaling グループのメトリクスを有効にする (AWS CLI)

Auto Scaling グループメトリクスを有効にするには

enable-metrics-collection コマンドを使用して、1 つ以上のグループメトリクスを有効にします。例えば、次のコマンドは、指定した Auto Scaling グループの単一のメトリクスを有効にします。

```
aws autoscaling enable-metrics-collection --auto-scaling-group-name my-asg \
  --metrics GroupDesiredCapacity --granularity "1Minute"
```

`--metrics` メトリクスを省略した場合、すべてのメトリクスが有効になります。

```
aws autoscaling enable-metrics-collection --auto-scaling-group-name my-asg \
  --granularity "1Minute"
```

Auto Scaling グループメトリクスを無効にするには

disable-metrics-collection コマンドを使用して、すべてのグループメトリクスを無効にします。

```
aws autoscaling disable-metrics-collection --auto-scaling-group-name my-asg
```

Auto Scaling インスタンスのモニタリングを設定する

Amazon EC2 は、インスタンスから未加工データを収集して、Auto Scaling グループの CPU やその他の使用率データを記述する、読み取り可能なほぼリアルタイムのメトリクスに加工します。これらのメトリクスをモニタリングする間隔を、1 分または 5 分のいずれかに設定できます。

インスタンスモニタリングは、インスタンスが起動されるたびに、基本モニタリング (5 分ごと) または詳細モニタリング (1 分ごと) で有効になります。詳細モニタリングでは追加料金が適用されます。詳細については、「Amazon EC2 ユーザーガイド」の「Amazon の CloudWatch 料金」および「 を使用したインスタンスのモニタリング」を参照してください。 CloudWatch Amazon EC2

Auto Scaling グループを作成する前に、アプリケーションに適したモニタリングタイプを許可する起動テンプレートまたは起動設定を作成する必要があります。グループにスケーリングポリシーを追加する場合は、負荷の変動に迅速に対応するために、詳細モニタリングを使用して EC2 インスタンスのメトリクスデータを 1 分単位で取得することを強くお勧めします。

内容

- 詳細モニタリングを有効にするには (コンソール)
- 詳細モニタリングを有効にする (AWS CLI)
- 基本モニタリングと詳細モニタリングを切り替える
- CloudWatch エージェントを使用して追加のメトリクスを収集する

詳細モニタリングを有効にするには (コンソール)

デフォルトでは、 を使用して起動テンプレートまたは起動設定を作成すると AWS Management Console 、基本モニタリングが有効になります。

起動テンプレートの詳細モニタリングを有効にするには

を使用して起動テンプレートを作成する場合 AWS Management Console、詳細モニタリング の詳細 CloudWatchセクションで、 を有効にするを選択します。それ以外の場合は、基本モニタリングが有効です。詳細については、「詳細設定を使用して起動テンプレートを作成する」を参照してください。

起動設定で詳細モニタリングを有効にするには

を使用して起動設定を作成するときは AWS Management Console、「追加設定」セクションで、「内で EC2 インスタンスの詳細モニタリングを有効にする CloudWatch」を選択します。それ以外の場合は、基本モニタリングが有効です。詳細については、「「起動設定を作成する」」を参照してください。

詳細モニタリングを有効にする (AWS CLI)

デフォルトでは、 AWS CLIを使用して起動テンプレートを作成すると、基本モニタリングが有効になります。詳細モニタリングは、 AWS CLIを使用して起動設定を作成する場合に有効になります。

起動テンプレートの詳細モニタリングを有効にするには

起動テンプレートの場合は、create-launch-template コマンドを使用して、起動テンプレートを作成するための情報が含まれる JSON ファイルを渡します。モニタリング属性を `"Monitoring": {"Enabled":true}` に設定して詳細モニタリングを有効にするか、または `"Monitoring": {"Enabled":false}` に設定して基本モニタリングを有効にします。

起動設定で詳細モニタリングを有効にするには

起動設定の場合は、create-launch-configuration コマンドを `--instance-monitoring` オプションで使用します。このオプションを `true` に設定して詳細モニタリングを有効に、または `false` に設定して基本モニタリングを有効にします。

```
--instance-monitoring Enabled=true
```

基本モニタリングと詳細モニタリングを切り替える

新しい EC2 インスタンスで有効になるモニタリングのタイプを変更するには、起動テンプレートを更新するか、Auto Scaling グループを更新して新しい起動テンプレートまたは起動設定を使用します。既存のインスタンスは、以前に有効化されたモニタリングタイプを使用し続けます。すべてのインスタンスを更新するには、すべてのインスタンスを終了して Auto Scaling グループに置き換えます。インスタンスを個々に更新するには、monitor-instances および unmonitor-instances を使用します。

> **Note**
>
> インスタンスの更新機能と最大有効期間の機能を使用すると、Auto Scaling グループ内のすべてのインスタンスを置き換えて、新しい設定を使用する新しいインスタンスを起動するこ

ともできます。詳細については、「Auto Scaling グループ内のインスタンスをリサイクルする」を参照してください。

基本モニタリングと詳細モニタリングを切り替えたときは、以下を実行します。

Auto Scaling グループのステップスケーリングポリシーまたは簡易スケーリングポリシーに関連付けられた CloudWatch アラームがある場合は、put-metric-alarm コマンドを使用して各アラームを更新します。各間隔をモニタリングタイプに合わせます (基本モニタリングでは 300 秒、詳細モニタリングでは 60 秒)。詳細モニタリングから基本モニタリングに変更しても、5 分間アラームを更新しないと、1 分ごとに統計を確認し続けます。5 回のうち最大 4 回は利用可能なデータが検出されない可能性があります。

CloudWatch エージェントを使用して追加のメトリクスを収集する

使用可能なメモリや使用済みメモリなどのオペレーティングシステムレベルのメトリクスを収集するには、 CloudWatch エージェントをインストールする必要があります。追加料金が発生する場合があります。 CloudWatch エージェントを使用して、Amazon EC2 インスタンスからシステムメトリクスとログファイルの両方を収集できます。詳細については、「Amazon ユーザーガイド」の CloudWatch 「エージェントによって収集されるメトリクス」を参照してください。 CloudWatch

Amazon EC2 Auto Scaling API 呼び出しを口グに記録する AWS CloudTrail

Amazon EC2 Auto Scaling は AWS CloudTrail、Amazon EC2 Auto Scaling を使用するユーザー、ロール、またはサービスによって実行されたアクションの記録を提供するサービスと統合されています。 CloudTrail Amazon EC2 Auto Scaling すべての API 呼び出しをイベントとしてキャプチャします。キャプチャされた呼び出しには、Amazon EC2 Auto Scaling コンソールからの呼び出しと、Amazon EC2 Auto Scaling API へのコード呼び出しが含まれます。

証跡を作成すると、Amazon EC2 Auto Scaling CloudTrail のイベントを含むイベントを Amazon S3 バケットに継続的に配信できるようになります。トレイルを設定しなくても、 CloudTrail コンソールの [イベント履歴] に最新のイベントが表示されます。によって収集された情報を使用して CloudTrail、Amazon EC2 Auto Scaling に対して行われたリクエスト、リクエストが行われた IP アドレス、リクエストの実行者、実行日時、その他の詳細を判断できます。

詳細については CloudTrail、AWS CloudTrail ユーザーガイドを参照してください。

の Amazon EC2 Auto Scaling 情報 CloudTrail

CloudTrail アカウントを作成すると、Amazon Web Services アカウントで有効になります。Amazon EC2 Auto Scaling でアクティビティが発生すると、 CloudTrail そのアクティビティはイベント履歴の他のAmazon Web Services イベントとともにイベントに記録されます。AWS アカウントでの最近のイベントを表示、検索、ダウンロードできます。詳細については、「CloudTrail イベントとイベント履歴の表示」を参照してください。

Amazon EC2 Auto Scaling のイベントなど、AWS アカウントのイベントの継続的な記録については、証跡を作成します。トレイルを使用すると CloudTrail 、Amazon S3 バケットにログファイルを配信できます。デフォルトでは、コンソールで証跡を作成すると、すべての リージョンに証跡が適用されます。証跡は、AWS パーティションのすべてのリージョンからのイベントをログに記録し、指定した Amazon S3 バケットにログファイルを配信します。さらに、 CloudTrail ログに収集されたイベントデータをさらに分析して処理するように他のAmazon Web Services 設定できます。詳細については、次を参照してください:

- 「証跡作成の概要」
- CloudTrail サポート対象のサービスとインテグレーション
- の Amazon SNS 通知の設定 CloudTrail
- CloudTrail 複数のリージョンからのログファイルの受信、 CloudTrail および複数のアカウントからのログファイルの受信

Amazon EC2 Auto Scaling CloudTrail アクションはすべてログに記録され、Amazon EC2 Auto Scaling API リファレンスに記載されています。たとえば、、UpdateAutoScalingGroupおよびアクションを呼び出すとCreateLaunchConfigurationDescribeAutoScalingGroup、 CloudTrail ログファイルにエントリが生成されます。

各イベントまたはログエントリには、誰がリクエストを生成したかという情報が含まれます。アイデンティティ情報は、以下を判別するのに役立ちます:

- リクエストが root ユーザー認証情報または AWS Identity and Access Management (IAM) ユーザー認証情報を使用して行われたかどうか。
- リクエストがロールまたはフェデレーションユーザーのテンポラリなセキュリティ認証情報を使用して行われたかどうか。
- リクエストが別の サービスによって行われたかどうか。

詳細については、CloudTrailuserIdentity要素を参照してください。

Amazon EC2 Auto Scaling のログファイルエントリについて

トレイルは、指定した Amazon S3 バケットにイベントをログファイルとして配信できるようにする設定です。 CloudTrail ログファイルには 1 つ以上のログエントリが含まれます。イベントはあらゆるソースからの単一のリクエストを表し、リクエストされたアクションに関する情報、アクションの日時、リクエストパラメータなどが含まれます。 CloudTrail ログファイルはパブリック API 呼び出しの順序付けられたスタックトレースではないため、特定の順序で表示されることはありません。

次の例は、 CloudTrail CreateLaunchConfigurationアクションを示すログエントリを示しています。

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "Root",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:root",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-08-21T17:05:42Z"
            }
        }
    },
    "eventTime": "2018-08-21T17:07:49Z",
    "eventSource": "autoscaling.amazonaws.com",
    "eventName": "CreateLaunchConfiguration",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Coral/Jakarta",
    "requestParameters": {
        "ebsOptimized": false,
        "instanceMonitoring": {
            "enabled": false
        },
        "instanceType": "t2.micro",
        "keyName": "EC2-key-pair-oregon",
        "blockDeviceMappings": [
            {
                "deviceName": "/dev/xvda",
                "ebs": {
                    "deleteOnTermination": true,
```

```
                    "volumeSize": 8,
                    "snapshotId": "snap-01676e0a2c3c7de9e",
                    "volumeType": "gp2"
                }
            }
        ],
        "launchConfigurationName": "launch_configuration_1",
        "imageId": "ami-6cd6f714d79675a5",
        "securityGroups": [
            "sg-00c429965fd921483"
        ]
    },
    "responseElements": null,
    "requestID": "0737e2ea-fb2d-11e3-bfd8-99133058e7bb",
    "eventID": "3fcfb182-98f8-4744-bd45-b38835ab61cb",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```


関連リソース

CloudWatch Logs を使用すると、 CloudTrailによってキャプチャされた特定のイベントを監視し、アラートを受信できます。 CloudWatch ログに送信されるイベントは、トレイルによって記録されるように設定されているイベントなので、監視したいイベントタイプを記録するように 1 つまたは複数のトレイルを設定していることを確認してください。 CloudWatch ログはログファイル内の情報を監視し、特定のしきい値に達すると通知してくれます。高い耐久性を備えたストレージにログデータをアーカイブすることもできます。詳細については、『Amazon CloudWatch Logs ユーザーガイド』および『ユーザーガイド』の「Amazon CloudWatch Logs CloudTrail による口グファイルの監視」トピックを参照してください。AWS CloudTrail

Amazon EC2 Auto Scaling の Amazon SNS 通知オプション Amazon EC2 Auto Scaling

アプリケーションに影響する重要なイベントを通知するように Auto Scaling グループを設定できます。通知を使用すると、ポーリングを排除することもできます。ポーリングによって発生することがあるRequestLimitExceeded工ラーは発生しません。

Amazon EC2 Auto Scaling に関する通知を受け取るには、2 つの方法があります。

- Amazon Simple Notification Service – Amazon SNS は、Auto Scaling グループがインスタンスを起動または終了したときに通知できます。Amazon SNS 通知は、オン/オフの切り替えのみが可能です。詳細については、「Amazon SNS と Amazon EC2 Auto Scaling」を参照してください。
- Amazon EventBridge – 指定された基準に一致し、Amazon SNS を含むさまざまなターゲットに送信される、より高度なイベント駆動型通知 EventBridge を提供します。 EventBridge は、より正確なモニタリングのために、幅広い Auto Scaling イベントをモニタリングすることもできます。詳細については、「 EventBridge を使用して Auto Scaling イベントを処理する」を参照してください。

、Amazon SNS EventBridge、Amazon SQS などのライフサイクルフックやサービスを使用して、インスタンスが起動時または終了時に保留状態になったときにカスタムアクションを実行することもできます。ライフサイクルフックは、Amazon EC2 Auto Scaling がインスタンスをグループに追加する前に、新しいインスタンスがユーザーデータで指定されたスクリプトを完了するまでさらに時間がかかることもあります。詳細については、「Amazon EC2 Auto Scaling のライフサイクルフック」を参照してください。

Amazon SNS と Amazon EC2 Auto Scaling

このセクションでは、Amazon SNS を使用して、Auto Scaling グループがインスタンスを起動または終了するタイミングをモニタリングする方法を示します。

例えば、`autoscaling: EC2_INSTANCE_TERMINATE` 通知タイプを使用するように Auto Scaling グループを設定する場合、Auto Scaling グループがインスタンスを終了すると、E メール通知が送信されます。この E メールには、インスタンス ID やインスタンスを終了した理由など、終了したインスタンスの詳細が含まれます。

Amazon EC2 Auto Scaling がグループに対してインスタンスを追加または削除すると、これらの変更に関する通知が送信され、インスタンスごとに 1 つの通知が送信されます。ただし、これらの通知の配信はベストエフォートベースであり、例えば、後のヘルスチェックが失敗した場合など、最初の通知後もインスタンスが失敗する可能性があります。そのため、Amazon EC2 Auto Scaling が最初に通知しても、インスタンスは後で失敗する可能性があります。最初のヘルスチェックを実行する前に、インスタンスの起動後 Amazon EC2 Auto Scaling が待機する時間を設定できます。詳細については、「Auto Scaling グループにヘルスチェックの猶予期間を設定する」を参照してください。

Amazon SNS 全般の詳細については、「Amazon Simple Notification Service デベロッパーガイド」を参照してください。

コンテンツ

- SNS 通知
- Amazon EC2 Auto Scaling の Amazon SNS 通知を設定する
  - Amazon SNS トピックを作成します。
  - Amazon SNS トピックを購読します。
  - Amazon SNS サブスクリプションを確認する
  - 通知を送信するように Auto Scaling グループを設定する
  - 通知をテストする
  - 通知設定を削除する
- 暗号化された Amazon SNS トピックのキーポリシー

SNS 通知

Amazon EC2 Auto Scaling は、以下のイベントが発生したときの Amazon SNS 通知の送信をサポートしています。

イベント	説明
`autoscaling:EC2_INSTANCE_LAUNCH`	インスタンスの起動成功
`autoscaling:EC2_INSTANCE_LAUNCH_ERROR`	インスタンスの起動失敗
`autoscaling:EC2_INSTANCE_TERMINATE`	インスタンスの削除成功
`autoscaling:EC2_INSTANCE_TERMINATE_ERROR`	インスタンスの削除失敗

メッセージには、次に示す情報が含まれます。

- `Event` – イベント。
- `AccountId` - アマゾン ウェブ サービスアカウント ID
- `AutoScalingGroupName` - Auto Scaling グループの名前。
- `AutoScalingGroupARN` – Auto Scaling グループの ARN。
- `EC2InstanceId` - EC2 インスタンスの ID。

例えば、以下のようになります。

```
Service: AWS Auto Scaling
Time: 2016-09-30T19:00:36.414Z
RequestId: 4e6156f4-a9e2-4bda-a7fd-33f2ae528958
Event: autoscaling:EC2_INSTANCE_LAUNCH
AccountId: 123456789012
AutoScalingGroupName: my-asg
AutoScalingGroupARN: arn:aws:autoscaling:region:123456789012:autoScalingGroup...
ActivityId: 4e6156f4-a9e2-4bda-a7fd-33f2ae528958
Description: Launching a new EC2 instance: i-0598c7d356eba48d7
Cause: At 2016-09-30T18:59:38Z a user request update of AutoScalingGroup constraints
 to ...
StartTime: 2016-09-30T19:00:04.445Z
EndTime: 2016-09-30T19:00:36.414Z
StatusCode: InProgress
StatusMessage:
Progress: 50
EC2InstanceId: i-0598c7d356eba48d7
Details: {"Subnet ID":"subnet-id","Availability Zone":"zone"}
Origin: AutoScalingGroup
Destination: EC2
```

Amazon EC2 Auto Scaling の Amazon SNS 通知を設定する

Amazon SNS を使用して E メール通知を送信するには、最初にトピックを作成してから、そのトピックと共に E メールアドレスを登録する必要があります。

Amazon SNS トピックを作成します。

SNS トピックとは、論理アクセスポイント、つまり Auto Scaling グループが通知を送信するために使用する通信チャネルです。トピックの名前を指定することにより、トピックを作成します。

トピック名を作成する際には、名前が次の要件を満たしている必要があります。

- 1 ~ 256 文字
- 大文字および小文字の ASCII 文字、数字、アンダースコア、またはハイフンが含まれている

詳細については、Amazon Simple 通知サービス デベロッパーガイドの「Amazon SNS トピックの作成」を参照してください。

Amazon SNS トピックを購読します。

Auto Scaling グループがトピックに送信した通知を受信するには、そのトピックにエンドポイントを登録する必要があります。この手順では、エンドポイント に、Amazon EC2 Auto Scaling からの通知を受信する E メールアドレスを指定します。

詳細については、Amazon Simple 通知サービス デベロッパーガイドの「Amazon SNS トピックへのサブスクライブ」を参照してください。

Amazon SNS サブスクリプションを確認する

Amazon SNS は、前のステップで指定した E メールアドレスに確認メールを送信します。

次のステップに進む前に、 AWS Notifications からの E メールを開き、リンクを選択してサブスクリプションを確認するようにしてください。

から確認メッセージが表示されます AWS。Amazon SNS は、通知を受信し、通知を E メールとして指定された E メールアドレスに送信するように設定されました。

通知を送信するように Auto Scaling グループを設定する

Auto Scaling グループは、インスタンスの起動または終了などのスケーリングイベントが発生したときに Amazon SNS に通知を送信するように設定することができます。Amazon SNS は、インスタンスに関する情報が含まれた通知を、ユーザーが指定する E メールアドレスに送信します。

Auto Scaling グループの Amazon SNS 通知を設定する (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの隣にあるチェックボックスを選択します。

   ページ下部に分割ウィンドウが開き、選択したグループの情報が表示されます。
3. [Activity] (アクティビティ) タブで、[Activity notifications] (アクティビティ通知)、[Create notification] (通知の作成) の順に選択します。
4. [Create notifications] ペインで、以下の作業を行います。

   a. [SNS Topic] (SNS トピック) で、SNS トピックを選択します。

   b. [Event types] (イベントタイプ) で、通知を送信するイベントを選択します。

   c. [作成] を選択します。

Auto Scaling グループの Amazon SNS 通知を設定する (AWS CLI)

次の put-notification-configuration コマンドを使用します。

```
aws autoscaling put-notification-configuration --auto-scaling-group-name my-
asg --topic-arn arn --notification-types "autoscaling:EC2_INSTANCE_LAUNCH"
 "autoscaling:EC2_INSTANCE_TERMINATE"
```

通知をテストする

起動イベントの通知を生成するには、Auto Scaling グループの希望容量を 1 増やして、Auto Scaling グループを更新します。インスタンスを起動してから数分以内に通知を受け取ります。

希望する容量を変更するには (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   [Auto Scaling グループ] ページの下部に分割ペインが開き、選択したグループに関する情報が表示されます。
3. [詳細] タブで、[グループの詳細]、[編集] の順に選択します。
4. [Desired capacity (希望するキャパシティ)] の場合は、現在の値を 1 ずつ増やします。この値が [Maximum capacity (最大容量)] を超える場合は、[Maximum capacity (最大容量)] の値を 1 ずつ増やす必要があります。
5. [更新] を選択します。
6. 数分後、イベントの通知が届きます。このテスト用に起動した追加のインスタンスが必要なくなった場合は、[Desired capacity (希望する容量)] を 1 減らすことができます。数分後、イベントの通知が届きます。

通知設定を削除する

Amazon EC2 Auto Scaling 通知設定が使用されなくなった場合は、設定を削除できます。

Amazon EC2 Auto Scaling 通知設定を削除する (コンソール)

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。

2. Auto Scaling スケーリンググループを選択します。
3. [Activity] (アクティビティ) タブで、削除する通知の横にあるチェックボックスをオンにしてから、[Actions] (アクション)、[Delete] (削除) の順に選択します。

Amazon EC2 Auto Scaling 通知設定を削除する (AWS CLI)

次の delete-notification-configuration コマンドを使用します。

```
aws autoscaling delete-notification-configuration --auto-scaling-group-name my-asg --
topic-arn arn
```

Auto Scaling グループに関連付けられている Amazon SNS トピックとすべてのサブスクリプションの削除については、「Amazon Simple Notification Service デベロッパーガイド」の「Amazon SNS サブスクリプションおよびトピックを削除する」を参照してください。

暗号化された Amazon SNS トピックのキーポリシー

指定した Amazon SNS トピックは、 AWS Key Management Service で作成されたカスタマーマネージドキーで暗号化される場合があります。暗号化されたトピックに発行するための許可を Amazon EC2 Auto Scaling に付与するには、まず KMS キーを作成してから、次のステートメントを KMS キーのポリシーに追加する必要があります。サンプルの ARN を、キーへのアクセスが許可されている適切なサービスにリンクされたロールの ARN に置き換えます。詳細については、「Amazon Simple Notification Service デベロッパーガイド」の「AWS KMS 許可を設定する」を参照してください。

この例では、ポリシーステートメントは、 という名前のサービスにリンクされたロールに、カスタマーマネージドキーを使用するためのAWSServiceRoleForAutoScalingアクセス許可を付与します。Amazon EC2 Auto Scaling のサービスにリンクされたロールの詳細については、「Amazon EC2 Auto Scaling のサービスにリンクされたロール」を参照してください。

```
{
  "Sid": "Allow service-linked role use of the customer managed key",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::123456789012:role/aws-service-role/autoscaling.amazonaws.com/
AWSServiceRoleForAutoScaling"
  },
  "Action": [
    "kms:GenerateDataKey*",
```

```
    "kms:Decrypt"
  ],
  "Resource": "*"
}
```

Amazon EC2 Auto Scaling が暗号化されたトピックにパブリッシュできるようにするキーポリシーでは、aws:SourceArn および aws:SourceAccount 条件キーはサポートされていません。

AWS Amazon EC2 Auto Scaling と統合された のサービス

Amazon EC2 Auto Scaling は他の AWS サービスと統合できます。各サービスが Amazon EC2 Auto Scaling と連動する方法については、以下の統合オプションをご確認ください。

トピック

- キャパシティの再調整を使用して Amazon EC2 スポットの中断に対処する
- オンデマンドキャパシティ予約を使用して特定のアベイラビリティーゾーンのキャパシティを予約する
- を使用してコマンドラインから Auto Scaling グループを作成する AWS CloudShell
- AWS CloudFormationで Auto Scaling グループを作成する
- AWS Compute Optimizer を使用して Auto Scaling グループのインスタンスタイプのレコメンデーションを取得する
- Elastic Load Balancing を使用して Auto Scaling グループ内のインスタンス全体にトラフィックを分散させる
- VPC Lattice ターゲットグループを使用して、トラフィックを Auto Scaling グループにルーティングする
- EventBridge を使用して Auto Scaling イベントを処理する
- Amazon VPC を使用して Auto Scaling インスタンスにネットワーク接続を提供する

キャパシティの再調整を使用して Amazon EC2 スポットの中断に対処する

スポットインスタンスの可用性に影響する変更をモニタリングし、自動的に応答するように Amazon EC2 Auto Scaling を設定できます。キャパシティの再調整は、実行中のインスタンスが Amazon EC2 により中断される前に、新しいスポットインスタンスでフリートを事前に拡張することにより、ワークロードの可用性を維持するのに役立ちます。

キャパシティの再調整の目標は、ワークロードの処理を中断されることなく継続することです。スポットインスタンスの中断リスクが高い場合、Amazon EC2 スポットサービスは、Amazon EC2 Auto Scaling に EC2 インスタンスの再調整レコメンデーションを通知します。

Auto Scaling グループに対してキャパシティの再調整を有効にすると、Amazon EC2 Auto Scaling は、再調整レコメンデーションを受け取ったグループ内のスポットインスタンスを積極的に置き換

えようとします。これは、ワークロードを中断リスクが低い新しいスポットインスタンスに再調整する機会を提供します。ワークロードは、既存のインスタンスが中断される前に Amazon EC2 Auto Scaling が新しいスポットインスタンスを起動している間も、作業の処理を続行できます。

キャパシティの再調整が無効化されていると、Amazon EC2 Auto Scaling は、Amazon EC2 スポットサービスがインスタンスを中断し、それらのヘルスチェックが失敗するまでスポットインスタンスを置き換えません。Amazon EC2 は常に、インスタンスを中断する前に EC2 インスタンスの再調整レコメンデーションと、スポットインスタンスの中断 2 分前の通知の両方を提供します。

内容

- 概要
- キャパシティーの再調整の動作
- 考慮事項
- キャパシティの再調整のを有効にする (コンソール)
- キャパシティの再調整を有効にする (AWS CLI)
- 関連リソース
- 制限事項

概要

Auto Scaling グループでキャパシティの再調整を使用するための基本的な手順は、以下のとおりです。

1. 複数のインスタンスタイプとアベイラビリティーゾーンを使用するように Auto Scaling グループを設定します。そうすることで、Amazon EC2 Auto Scaling は各アベイラビリティーゾーンでスポットインスタンスの利用可能なキャパシティを検討できるようになります。詳細については、「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。
2. 必要に応じてライフサイクルフックを追加して、再調整通知を受け取るインスタンス内でアプリケーションを正常にシャットダウンします。詳細については、「Amazon EC2 Auto Scaling のライフサイクルフック」を参照してください。

   以下は、ライフサイクルフックを使用するいくつかの理由です。

   - Amazon SQS ワーカーの正常なシャットダウンのために
   - ドメインネームシステム (DNS) からの登録解除を完了するには

- システムログまたはアプリケーションログをプルし、Amazon Simple Storage Service (Amazon S3) にアップロードするには

3. ライフサイクルフックのカスタムアクションを開発します。カスタムアクションを呼び出すには、インスタンスの終了の準備が整ったタイミングを知る必要があります。インスタンスのライフサイクル状態を検出することでタイミングを確認します。
   - インスタンスの外部でアクションを呼び出すには、 EventBridge ルールを記述し、イベントパターンがルールに一致するときに実行するアクションを自動化します。
   - インスタンス内でアクションを呼び出すには、シャットダウンスクリプトを実行し、インスタンスのメタデータを介してライフサイクルスの状態を取得するようにインスタンスを設定します。

   カスタムアクションが 2 分以内に終了するように設計することが重要です。これにより、インスタンス終了前にタスクを完了するのに十分な時間が確保されます。

これらの手順を完了すると、キャパシティの再調整を使用できるようになります。

キャパシティーの再調整の動作

キャパシティの再調整では、インスタンスが再調整レコメンデーションを受け取ると、Amazon EC2 Auto Scaling は次のように動作します。

- 新しいスポットインスタンスの起動時、Amazon EC2 Auto Scaling は、新しいインスタンスがヘルスチェックに合格するまで待機してから、以前のインスタンスを終了させます。複数のインスタンスを置き換える場合、以前の各インスタンスの終了は、新しいインスタンスが起動され、ヘルスチェックに合格してから開始されます。
- Amazon EC2 Auto Scaling は以前のインスタンスを終了する前に新しいインスタンスの起動を試みるため、指定された最大キャパシティに到達している、またはそれに近い状態は、再調整アクティビティを妨げたり、それらを完全に停止させる可能性があります。この問題を回避するために、Amazon EC2 Auto Scaling は一時的にグループの最大サイズを必要なキャパシティの最大 10% まで超過できます。
- Auto Scaling グループにライフサイクルフックを追加しなかった場合、Amazon EC2 Auto Scaling は、新しいインスタンスがヘルスチェックに合格すると同時に以前のインスタンスの終了を開始します。
- ライフサイクルフックを追加した場合は、前のインスタンスの終了を開始するまでの時間が、ライフサイクルフックに指定したタイムアウト値だけ延長されます。

- スケーリングポリシーまたはスケジュールされたスケーリングを使用している場合、スケーリングアクティビティは並行して実行されます。スケーリングアクティビティが進行中で、Auto Scaling グループが新しい希望キャパシティを下回っている場合、Amazon EC2 Auto Scaling はまずスケールアウトを行ってから、以前のインスタンスを終了します。

あるアベイラビリティーゾーンに該当するインスタンスタイプのキャパシティがない場合、Amazon EC2 Auto Scaling は他の有効なアベイラビリティーゾーンで成功するまでスポットインスタンスの起動を試行し続けます。

最悪のシナリオでは、新しいインスタンスの起動に失敗するか、ヘルスチェックに失敗すると、Amazon EC2 Auto Scaling はインスタンスの再起動を試行し続けます。　新しいインスタンスの起動試行中、以前のインスタンスは最終的に中断され、2 分間の中断通知により強制的に終了されます。

考慮事項

キャパシティの再調整を使用する場合は、次の点を考慮してください。

スポットの中断に耐えられるようにアプリケーションを設計する

アプリケーションはインスタンス数の動的な変化と、スポット インスタンスが早期に中断される可能性に対処する必要があります。例えば、Auto Scaling グループが Elastic Load Balancing ロードバランサーの背後にある場合、Amazon EC2 Auto Scaling は、インスタンスがロードバランサーから登録解除されるまで待機してから、終了ライフサイクルフックを呼び出します。インスタンスの登録解除とライフサイクルアクションの完了にかかる時間が長すぎる場合、Amazon EC2 Auto Scaling がインスタンスを終了させる前にライフサイクルアクションの完了を待機する間に、インスタンスが中断される可能性があります。

2 分間のスポットインスタンス中断通知の前に、Amazon EC2 が再調整のレコメンデーションシグナルを送信できるとは限りません。場合によっては、再調整のレコメンデーションシグナルが、2 分間の中断通知と同時に到着する可能性があります。この場合、Amazon EC2 Auto Scaling はライフサイクルフックを呼び出し、新しいスポットインスタンスをすぐに起動しようとします。

代替スポットインスタンスが中断されるリスクの増大を回避する

`lowest-price` 割り当て戦略を使用している場合、代替スポットインスタンスが中断されるリスクが高くなることがあります。これは、代替スポットインスタンスが起動後すぐに中断される可能性が高い場合も、その時点で利用可能なキャパシティを持つ最低料金のプールでインスタンス

を起動することが原因です。中断のリスクが高まるのを避けるため、lowest-price の割り当て戦略を使用しないことを強くお勧めします。代わりに、price-capacity-optimized の使用をお勧めします。この戦略では、中断される可能性が最も低く、可能な限り低価格のスポットプールで代替スポットインスタンスを起動します。したがって、近い将来に中断される可能性は低くなります。

Amazon EC2 Auto Scaling は、可用性が同じかそれ以上の場合にのみ、新しいインスタンスを起動します

キャパシティ再調整の目的の 1 つは、スポットインスタンスの可用性を改善することです。既存のスポットインスタンスが再調整のレコメンデーションを受け取った場合、Amazon EC2 Auto Scaling は、新しいインスタンスが既存のインスタンスと同等かそれ以上の可用性を提供する場合にのみ新しいインスタンスを起動します。新しいインスタンスの中断のリスクが既存のインスタンスよりもひどい場合、Amazon EC2 Auto Scaling は新しいインスタンスを起動しません。ただし、Amazon EC2 Auto Scaling は引き続き Amazon EC2 スポットサービスから提供された情報に基づいてスポットキャパシティプールを評価し、可用性が向上した場合は新しいインスタンスを起動します。

Amazon EC2 Auto Scaling が新しいインスタンスをプロアクティブに起動しないと、既存のインスタンスが中断する可能性があります。中断が発生すると、Amazon EC2 Auto Scaling は、スポットインスタンスの中断通知を受け取ると直ちに新しいインスタンスの起動を試みます。これは、新しいインスタンスが中断されるリスクが高いかどうかに関係なく起こります。

キャパシティーの再調整は、スポットインスタンスの中断率を増加させるものではありません

キャパシティの再調整を有効にしても、スポットインスタンスの中断率 (Amazon EC2 がキャパシティを取り戻す必要があるときに再利用されるスポットインスタンスの数) は増加しません。ただし、インスタンスに中断のリスクがあることをキャパシティの再調整が検出した場合、Amazon EC2 Auto Scaling は直ちに新しいインスタンスの起動を試みます。したがって、リスクのあるインスタンスが中断された後に Amazon EC2 Auto Scaling が新しいインスタンスを起動するのを待つ場合よりも多くのインスタンスが置き換えられる可能性があります。

キャパシティの再調整が有効になっているインスタンスをさらに置き換える可能性がありますが、事後対応ではなくプロアクティブに対応できるというメリットがあります。これにより、インスタンスが中断される前にアクションを実行できる時間が増えます。スポットインスタンスの中断通知では、通常、インスタンスを正常にシャットダウンするための猶予期間が最大 2 分しかありません。キャパシティの再調整で新しいインスタンスが事前に起動されるため、リスクのあるインスタンスで既存のプロセスを完了できる可能性が高まります。また、インスタンスのシャットダウン手順を開始し、リスクのあるインスタンスで新しい作業がスケジュールされないようにしたり、新しく起動したインスタンスがアプリケーションを引き継ぐように準備すること

もできます。キャパシティの再調整のプロアクティブな置き換えにより、正常な継続性の恩恵を受けることができます。

次の理論的な例は、キャパシティの再調整を使用するリスクとメリットを示しています。

- 午後 2 時 – インスタンス A に対する再調整のレコメンデーションを受け取りました。Amazon EC2 Auto Scaling が直ちに代替インスタンス B の起動の試行を開始するため、ユーザーはシャットダウン手順を開始する時間を確保できます。
- 午後 2 時 30 分 – インスタンス B の再調整のレコメンデーションを受け取り、インスタンス C に置き換えられました。これにより、ユーザーはシャットダウン手順を開始する時間を確保できます。
- 午後 2 時 32 分 – キャパシティの再調整が有効になっておらず、インスタンス A のスポットインスタンスの中断通知が午後 2 時 32 分に受信されていたとすれば、アクションを実行するための猶予時間は最大でも 2 分だけでした。ただし、インスタンス A はこの時点まで実行され続けていたはずです。

キャパシティの再調整のを有効にする (コンソール)

Auto Scaling グループを作成または更新するときに、キャパシティの再調整を有効または無効にできます。

新しい Auto Scaling グループのキャパシティの再調整を有効にするには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. [Auto Scaling グループの作成] を選択します。
3. ステップ 1 で、起動テンプレートまたは構成を選択し、Auto Scaling グループの名前を入力し、起動テンプレートを選択してから、[Next] (次へ) を選択して次のステップに進みます。
4. [ステップ 2: インスタンスの起動オプションを選択] の [インスタンスタイプの要件] で、混合インスタンスグループを作成するための設定を選択します。この混合インスタンスグループには、起動できるインスタンスタイプ、インスタンス購入オプション、スポットインスタンスとオンデマンドインスタンスの配分戦略が含まれます。これらの設定はデフォルトで設定されていません。これらを設定するには、[Override launch template] (起動テンプレートを上書きする) を選択する必要があります。混合インスタンスグループの作成に関する詳細については、「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。
5. [ネットワーク] で、必要なオプションを選択します。使用するサブネットが、異なるアベイラビリティーゾーンにあることを確認します。

6. [配分戦略] セクションで、スポット配分戦略を選択します。[容量の再分散] で、チェックボックスをオンまたはオフにして、容量の再分散を有効または無効にします。このオプションは、[インスタンスの購入オプション] セクションで Auto Scaling グループのスポットインスタンスとして起動するようにリクエストした場合にのみ表示されます。
7. Auto Scaling グループを作成します。
8. (オプション) 必要に応じてライフサイクルフックを追加します。詳細については、「ライフサイクルフックを追加する」を参照してください。

既存の Auto Scaling グループのキャパシティの再調整を有効または無効にするには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。ページの下部にスプリットペインが開きます。
3. [Details] (詳細) タブで、[Allocation strategies] (配分戦略)、[Edit] (編集) の順に選択します。
4. [配分戦略] セクションで、[容量の再分散] チェックボックスをオンまたはオフにして、容量の再分散を有効または無効にします。
5. [更新] を選択します。

キャパシティの再調整を有効にする (AWS CLI)

次の例は、 を使用してキャパシティーの再調整 AWS CLI を有効または無効にする方法を示しています。

[create-auto-scaling-group] または [update-auto-scaling-group] コマンドに次のパラメータを指定して使用します。

- `--capacity-rebalance` / `--no-capacity-rebalance` – キャパシティの再調整が有効になっているかどうかを示すブール値。

[create-auto-scaling-group] コマンドを呼び出す前に、Auto Scaling グループで使用するように設定されている起動テンプレートの名前が必要です。詳細については、「Auto Scaling グループの起動テンプレートを作成する」を参照してください。

> **Note**
>
> 次の手順は、JSON または YAML でフォーマットされた設定ファイルの使用方法を示しています。 AWS CLI バージョン 1 を使用する場合は、JSON 形式の設定ファイルを指定する必要があります。 AWS CLI バージョン 2 を使用する場合は、YAML または JSON のいずれかでフォーマットされた設定ファイルを指定できます。

JSON

新しい Auto Scaling グループを作成して設定するには

- 次の create-auto-scaling-group コマンドを使用して、新しい Auto Scaling グループを作成し、キャパシティの再調整を有効にします。このコマンドは、Auto Scaling グループの唯一のパラメーターとして JSON ファイルを参照します。

```
aws autoscaling create-auto-scaling-group --cli-input-json file://~/config.json
```

[混合インスタンス・ポリシー] を指定する CLI 設定ファイルがまだない場合は、作成します。

次の行を設定ファイルのトップレベルの JSON オブジェクトに追加します。

```
{
    "CapacityRebalance": true
}
```

次は、config.json ファイルの例です。

```
{
    "AutoScalingGroupName": "my-asg",
    "DesiredCapacity": 12,
    "MinSize": 12,
    "MaxSize": 15,
    "CapacityRebalance": true,
    "MixedInstancesPolicy": {
        "InstancesDistribution": {
            "OnDemandBaseCapacity": 0,
            "OnDemandPercentageAboveBaseCapacity": 25,
            "SpotAllocationStrategy": "price-capacity-optimized"
        },
```

```
        "LaunchTemplate": {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "$Default"
            },
            "Overrides": [
                {
                    "InstanceType": "c5.large"
                },
                {
                    "InstanceType": "c5a.large"
                },
                {
                    "InstanceType": "m5.large"
                },
                {
                    "InstanceType": "m5a.large"
                },
                {
                    "InstanceType": "c4.large"
                },
                {
                    "InstanceType": "m4.large"
                },
                {
                    "InstanceType": "c3.large"
                },
                {
                    "InstanceType": "m3.large"
                }
            ]
        }
    },
    "TargetGroupARNs": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-alb-target-group/943f017f100becff",
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
}
```

YAML

新しい Auto Scaling グループを作成して設定するには

- 次の create-auto-scaling-group コマンドを使用して、新しい Auto Scaling グループを作成し、キャパシティの再調整を有効にします。このコマンドは、Auto Scaling グループの唯一のパラメーターとして YAML ファイルを参照します。

```
aws autoscaling create-auto-scaling-group --cli-input-yaml file://~/config.yaml
```

YAML でフォーマットされた設定ファイルに以下の行を追加します。

```
CapacityRebalance: true
```

次は、config.yaml ファイルの例です。

```
---
AutoScalingGroupName: my-asg
DesiredCapacity: 12
MinSize: 12
MaxSize: 15
CapacityRebalance: true
MixedInstancesPolicy:
  InstancesDistribution:
    OnDemandBaseCapacity: 0
    OnDemandPercentageAboveBaseCapacity: 25
    SpotAllocationStrategy: price-capacity-optimized
  LaunchTemplate:
    LaunchTemplateSpecification:
      LaunchTemplateName: my-launch-template
      Version: $Default
    Overrides:
    - InstanceType: c5.large
    - InstanceType: c5a.large
    - InstanceType: m5.large
    - InstanceType: m5a.large
    - InstanceType: c4.large
    - InstanceType: m4.large
    - InstanceType: c3.large
    - InstanceType: m3.large
TargetGroupARNs:
```

```
- arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-alb-target-
group/943f017f100becff
VPCZoneIdentifier: subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782
```

既存の Auto Scaling グループのキャパシティの再調整を有効にするには

- 以下の update-auto-scaling-group コマンドを使用して、キャパシティの再調整を有効にします。

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --capacity-rebalance
```

Auto Scaling グループのキャパシティの再調整を有効になっていることを確認するには

- 以下の describe-auto-scaling-group コマンドを使用して、キャパシティの再調整が有効になっていることを確認し、詳細を表示します。

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
```

以下に、応答の例を示します。

```
{
    "AutoScalingGroups": [
        {
            "AutoScalingGroupName": "my-asg",
            "AutoScalingGroupARN": "arn",
            ...
            "CapacityRebalance": true
        }
    ]
}
```

キャパシティの再調整を無効にするには

キャパシティの再調整を無効にするには、[update-auto-scaling-group] コマンドに--no-capacity-rebalanceオプションを付けて使用します。

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
```

```
  --no-capacity-rebalance
```

関連リソース

容量の再調整の詳細については、 AWS コンピューティングブログのEC2 Auto Scaling の新しい容量再調整機能を使用してスポットインスタンスのライフサイクルをプロアクティブに管理する」を参照してください。

EC2 インスタンスの再調整に関する推奨事項の詳細については、「Amazon EC2 ユーザーガイド」の「EC2 インスタンスの再調整に関する推奨事項」を参照してください。 Amazon EC2

ライフスタイルフックの詳細については、以下のリソースを参照してください。

- チュートリアル:Lambda 関数を呼び出すライフサイクルフックの設定 (を使用 EventBridge)
- チュートリアル: インスタンスメタデータを使用してターゲットライフサイクル状態を取得するようにユーザーデータを設定する

制限事項

- Amazon EC2 Auto Scaling は、インスタンスがスケールインから保護されていない場合のみ、再調整通知を受信したインスタンスを置き換えることができます。ただし、スケールイン保護はスポットの中断による終了を防ぐことはできません。詳細については、「インスタンスのスケールイン保護を使用する」を参照してください。
- キャパシティの再調整のサポートは、中東 (アラブ首長国連邦) リージョンを除くAmazon EC2 Auto Scaling が使用可能なすべての商用 AWS リージョン で利用できます。

オンデマンドキャパシティ予約を使用して特定のアベイラビリティーゾーンのキャパシティを予約する

Amazon EC2 オンデマンド キャパシティ予約は、特定のアベイラビリティーゾーンでコンピューティング性能を予約するのに役立ちます。キャパシティ予約の使用を開始するには、特定のアベイラビリティーゾーンにキャパシティ予約を作成します。次に、インスタンスをリザーブドキャパシティに起動し、そのキャパシティの使用率をリアルタイムで表示して、必要に応じてキャパシティを増減することができます。

キャパシティ予約は、open または `targeted` のいずれかで設定されます。キャパシティ予約が open の場合、一致する属性を持つすべての新規および既存のインスタンスは、キャパシティ予約

のキャパシティ内で自動的に実行されます。キャパシティ予約 が `targeted` の場合、インスタンスはそれがリザーブドキャパシティで実行されるように具体的に設定する必要があります。

このトピックでは、`targeted` キャパシティ予約にオンデマンドインスタンスを起動する Auto Scaling グループを作成する方法を説明します。これにより、特定のキャパシティ予約をいつ使用するかをより細かく制御できます。

基本的なステップは次のとおりです。

1. 同じインスタンスタイプ、プラットフォーム、インスタンス数を持つ複数のアベイラビリティーゾーンでキャパシティ予約を作成します。
2. AWS Resource Groups を使用してキャパシティ予約をグループ化します。
3. キャパシティ予約と同じアベイラビリティーゾーンを使用して、リソースグループを対象とする起動テンプレートを使用して Auto Scaling グループを作成します。

内容

- ステップ 1: キャパシティ予約を作成する
- ステップ 2: キャパシティ予約グループを作成する
- ステップ 3: 起動テンプレートを作成する
- ステップ 4: Auto Scaling グループを作成する
- 関連リソース

ステップ 1: キャパシティ予約を作成する

最初のステップは、Auto Scaling グループがデプロイされる各アベイラビリティーゾーンにキャパシティ予約を作成することです。

> **Note**
>
> 最初にキャパシティ予約を作成するときは、`targeted` 予約のみを作成できます。

Console

キャパシティ予約を作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. [キャパシティ予約]、[作成キャパシティ予約] の順に選択します。
3. 「キャパシティ予約を作成」のページで、[インスタンスの詳細] セクションで、以下の設定を指定します。起動するインスタンスのインスタンスタイプ、プラットフォーム、アベイラビリティーゾーンは、ここで指定するインスタンスタイプ、プラットフォーム、アベイラビリティーゾーンと一致する必要があります。一致しない場合、キャパシティ予約 は適用されません。

   a. [インスタンスのタイプ] では、リザーブドキャパシティーに起動するインスタンスのタイプを選択します。
   b. [プラットフォーム] では、インスタンスのオペレーティングシステムを選択します。
   c. [アベイラビリティーゾーン]で、キャパシティを予約したい最初のアベイラビリティーゾーンを選択します。
   d. 合計容量 で、必要なインスタンスの数を選択します。Auto Scaling グループに必要なインスタンスの総数を、使用する予定のアベイラビリティーゾーンの数で割って計算します。
4. [キャパシティ予約の詳細] で、[キャパシティ予約の終了] について、次のオプションのいずれかを選択します。

   - 特定の時刻 – 指定された日時に自動的にキャパシティーの予約をキャンセルします。
   - 手動 – 明示的にキャンセルするまで容量を予約します。
5. [インスタンスの適格性] については、[ターゲット: キャパシティ予約を対象とするインスタンスのみ] を選択します。
6. (オプション) [タグ] には、キャパシティ予約に関連付けるタグを指定します。
7. [作成] を選択します。
8. 新しく作成したキャパシティ予約の ID をメモしておきます。キャパシティ予約グループを設定するために必要です。

Auto Scaling グループに対して有効にする各アベイラビリティーゾーンに対してこの手順を繰り返し、[アベイラビリティーゾーン] オプションの値のみを変更します。

AWS CLI

キャパシティ予約を作成するには

次の create-capacity-reservation コマンドを使用してキャパシティリザーブを作成します。`--availability-zone`、`--instance-type`、`--instance-platform`、および `--instance-count` のサンプルの値を置き換えます。

```
aws ec2 create-capacity-reservation \
    --availability-zone us-east-1a \
    --instance-type c5.xlarge \
    --instance-platform Linux/UNIX \
    --instance-count 3 \
    --instance-match-criteria targeted
```

キャパシティ予約 ID の結果の例

```
{
    "CapacityReservation": {
        "CapacityReservationId": "cr-1234567890abcdef1",
        "OwnerId": "123456789012",
        "CapacityReservationArn": "arn:aws:ec2:us-east-1:123456789012:capacity-
reservation/cr-1234567890abcdef1",
        "InstanceType": "c5.xlarge",
        "InstancePlatform": "Linux/UNIX",
        "AvailabilityZone": "us-east-1a",
        "Tenancy": "default",
        "TotalInstanceCount": 3,
        "AvailableInstanceCount": 3,
        "EbsOptimized": false,
        "EphemeralStorage": false,
        "State": "active",
        "StartDate": "2023-07-26T21:36:14+00:00",
        "EndDateType": "unlimited",
        "InstanceMatchCriteria": "targeted",
        "CreateDate": "2023-07-26T21:36:14+00:00"
    }
}
```

新しく作成したキャパシティ予約の ID をメモしておきます。キャパシティ予約グループを設定するために必要です。

Auto Scaling グループに対して有効にする各アベイラビリティーゾーンに対してこのコマンドを繰り返し、`--availability-zone` オプションの値のみを変更します。

ステップ 2: キャパシティ予約グループを作成する

キャパシティ予約の作成が完了したら、 AWS Resource Groups サービスを使用してそれらをグループ化できます。 AWS Resource Groups は、さまざまな用途で複数の異なるタイプのグループをサ

ポートします。Amazon EC2 は、サービスにリンクされたリソースグループと呼ばれる特別な目的のグループを使用して、キャパシティ予約グループをターゲットにします。このサービスにリンクされたリソースグループを操作するには、 AWS CLI または SDK ですが、コンソールではありません。サービスにリンクされたリソース グループの詳細については、「AWS Resource Groups User Guide」の 「Service configurations for resource groups」を参照してください。

を使用してキャパシティ予約グループを作成するには AWS CLI

create-group コマンドを使用して、キャパシティ予約のみを含めることができるリソースグループを作成します。この例では、プレイスメントグループ名は*my-cr-group*です。

```
aws resource-groups create-group \
    --name my-cr-group \
    --configuration '{"Type":"AWS::EC2::CapacityReservationPool"}'
 '{"Type":"AWS::ResourceGroups::Generic", "Parameters": [{"Name": "allowed-resource-
types", "Values": ["AWS::EC2::CapacityReservation"]}]}'
```

以下に、応答の例を示します。

```
{
    "Group": {
        "GroupArn": "arn:aws:resource-groups:us-east-1:123456789012:group/my-cr-group",
        "Name": "my-cr-group"
    },
    "GroupConfiguration": {
        "Configuration": [
            {
                "Type": "AWS::EC2::CapacityReservationPool"
            },
            {
                "Type": "AWS::ResourceGroups::Generic",
                "Parameters": [
                    {
                        "Name": "allowed-resource-types",
                        "Values": [
                            "AWS::EC2::CapacityReservation"
                        ]
                    }
                ]
            }
        ],
        "Status": "UPDATE_COMPLETE"
```

```
    }
}
```

リソースグループの ARN をメモしておきます。Auto Scaling グループの起動テンプレートを設定するために必要です。

AWS CLI を使用してキャパシティ予約を新しく作成したグループに関連付けるには

次の group-resources コマンドを使用して、キャパシティ予約を新しく作成したキャパシティ予約グループに関連付けます。--resource-arns オプションの場合、ARN を使用してキャパシティ予約を指定します。関連するリージョン、アカウント ID、および前にメモした予約 ID を使用して ARN を作成します。この例では、ID *cr-1234567890abcdef1* と *cr-54321abcdef567890* の予約が*my-cr-group* という名前のグループとしてグループ化されます。

```
aws resource-groups group-resources \
    --group my-cr-group \
    --resource-arns \
      arn:aws:ec2:region:account-id:capacity-reservation/cr-1234567890abcdef1 \
      arn:aws:ec2:region:account-id:capacity-reservation/cr-54321abcdef567890
```

以下に、応答の例を示します。

```
{
   "Succeeded": [
   "arn:aws:ec2:us-east-1:123456789012:capacity-reservation/cr-1234567890abcdef1",
   "arn:aws:ec2:us-east-1:123456789012:capacity-reservation/cr-54321abcdef567890"
   ],
   "Failed": [],
   "Pending": []
}
```

リソースグループの変更または削除については、「AWS Resource Groups API リファレンス」を参照してください。

ステップ 3: 起動テンプレートを作成する

Console

起動テンプレートを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインで、[インスタンス] の [テンプレートの起動] を選択します。
3. [起動テンプレートの作成] を選択します。名前を入力し、起動テンプレートの最初のバージョンの説明を加えます。
4. [Auto Scaling ガイダンス] で、[チェックボックス] を選択します。
5. 起動テンプレートを作成します。使用する予定のキャパシティ予約と一致する AMI とインスタンスタイプを選択し、オプションでキーペア、1 つ以上のセキュリティグループ、インスタンス用の追加の EBS ボリュームまたはインスタンスストアボリュームを選択します。
6. [高度な設定]を展開し、以下の操作を行います。

   a. [キャパシティ予約]で、[グループ別のターゲット] を選択します。

   b. [キャパシティ予約 - グループ別のターゲット] で、前のセクションで作成したキャパシティ予約グループを選択し、[保存] を選択します。
7. [起動テンプレートの作成] を選択します。
8. 確認ページで、[Auto Scaling グループの作成] を選択します。

AWS CLI

起動テンプレートを作成するには

次の create-launch-template コマンドを使用して、キャパシティ予約が特定のリソースグループをターゲットにすることを指定する起動テンプレートを作成します。--launch-template-name のサンプル値を置き換えます。*c5.xlarge* をキャパシティ予約で使用したインスタンス タイプに置き換え、*ami-0123456789EXAMPLE* を使用する AMI の ID に置き換えます。*arn:aws:resource-groups:region:account-id:group/my-cr-group* を、前のセクションのセクションで作成したリソースグループの ARN に置き換えます。

```
aws ec2 create-launch-template \
    --launch-template-name my-launch-template \
    --launch-template-data \
        '{"InstanceType": "c5.xlarge",
          "ImageId": "ami-0123456789EXAMPLE",
          "CapacityReservationSpecification":
            {"CapacityReservationTarget":
                { "CapacityReservationResourceGroupArn": "arn:aws:resource-
groups:region:account-id:group/my-cr-group" }
            }
        }'
```

以下に、応答の例を示します。

```
{
    "LaunchTemplate": {
        "LaunchTemplateId": "lt-0dd77bd41dEXAMPLE",
        "LaunchTemplateName": "my-launch-template",
        "CreateTime": "2023-07-26T21:42:48+00:00",
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "DefaultVersionNumber": 1,
        "LatestVersionNumber": 1
    }
}
```

ステップ 4: Auto Scaling グループを作成する

Console

通常どおり Auto Scaling グループを作成しますが、VPC サブネットを選択するときは、作成した `targeted` のキャパシティ予約に一致する各アベイラビリティーゾーンからサブネットを選択します。その後、Auto Scaling グループがこれらのアベイラビリティーゾーンのいずれかでオンデマンドインスタンスを起動すると、そのインスタンスがそのアベイラビリティーゾーンのリザーブドキャパシティで実行されます。希望するキャパシティが満たされる前にリソースグループがキャパシティ予約を使い果たした場合、リザーブドキャパシティを超えるものは通常のオンデマンドキャパシティとして起動されます。

Auto Scaling グループを作成するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. 画面上部のナビゲーションバーで、起動テンプレートの作成時に使用した AWS リージョンものと同じ を選択します。
3. [Auto Scaling グループの作成] を選択します。
4. [起動テンプレートまたは起動設定を選択する] ページで [Auto Scaling グループ名] にAuto Scaling グループの名前を入力します。
5. [起動テンプレート] で、既存の起動テンプレートを選択します。
6. [起動テンプレートのバージョン] で、スケールアウト時に Auto Scaling グループで使用する起動テンプレートのバージョン (デフォルト、最新、または特定のバージョン) を選択します。

7. [インスタンス起動オプションを選択] ページで、[インスタンスタイプの要件] セクションをスキップし、起動テンプレートで指定されている EC2 インスタンスタイプを使用します。
8. [Network] (ネットワーク) の下にある [VPC] で、VPC を選択します。Auto Scaling グループは、起動テンプレートで指定したセキュリティグループと、同じ VPC 内に作成する必要があります。起動テンプレートにセキュリティグループを指定しなかった場合、キャパシティ予約と同じアベイラビリティーゾーンにサブネットがある任意の VPC を選択できます。
9. [アベイラビリティーゾーンとサブネット]では、キャパシティ予約がどのアベイラビリティーゾーンにあるかに基づいて、含めたいサブネットを各アベイラビリティーゾーンから選択します。
10. [次へ] を 2 回選択します。
11. [グループサイズとスケーリングポリシーを設定] ページの [必要なキャパシティ] に、起動するインスタンスの初期数を入力します。この数値を最小キャパシティまたは最大キャパシティ制限の範囲外の値に変更する場合は、[最小キャパシティ] または [最大キャパシティ] の値を更新する必要があります。詳細については、「Auto Scaling グループのスケーリング制限を設定する」を参照してください。
12. [Skip to review] を選択します。
13. [Review (レビュー)]ページで、[Create Auto Scaling group (Auto Scaling グループを作成)] を選択します。

AWS CLI

Auto Scaling グループを作成するには

次の create-auto-scaling-group コマンドを使用し、起動テンプレートの名前とバージョンを `--launch-template` オプションの値として指定します。`--auto-scaling-group-name`、`--min-size`、`--max-size`、および `--vpc-zone-identifier` のサンプルの値を置き換えます。

`--availability-zones` オプションには、キャパシティ予約を作成したアベイラビリティーゾーンを指定します。例えば、キャパシティ予約で `us-east-1a` と `us-east-1b` のアベイラビリティーゾーンの場合は、同じゾーンに Auto Scaling グループを作成する必要があります。その後、Auto Scaling グループがこれらのアベイラビリティーゾーンのいずれかでオンデマンドインスタンスを起動すると、そのインスタンスがそのアベイラビリティーゾーンのリザーブドキャパシティで実行されます。希望するキャパシティが満たされる前にリソースグループがキャパシティ予約を使い果たした場合、リザーブドキャパシティを超えるものは通常のオンデマンドキャパシティとして起動されます。

```
aws autoscaling create-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --launch-template LaunchTemplateName=my-launch-template,Version='1' \
    --min-size 6 \
    --max-size 6 \
    --vpc-zone-identifier "subnet-5f46ec3b,subnet-0ecac448" \
    --availability-zones us-east-1a us-east-1b
```

関連リソース

実装例については、次の AWS サンプル GitHub リポジトリの AWS CloudFormation テンプレートを参照してください: https://github.com/aws-samples/aws-auto-scaling-backed-by-on-demand-capacity-reservations/。

キャパシティ予約の詳細について調べる際は、以下の関連トピックが役に立ちます。

- On-Demand Capacity Reservations
  - 「Amazon EC2 ユーザーガイド」の「キャパシティーの予約」を作成する Amazon EC2
  - 「Amazon EC2 ユーザーガイド」の「オンデマンドキャパシティ予約Amazon EC2」
  - AWS クラウド運用と移行ブログのAmazon EC2 オンデマンドキャパシティ予約」のグループをターゲットにする
- キャパシティブロック (期間が定義されたキャパシティ予約)
  - 「Amazon EC2 ユーザーガイド」の「ML のキャパシティブロックAmazon EC2」
  - 機械学習ワークロードCapacity Blocksに を使用する

を使用してコマンドラインから Auto Scaling グループを作成する AWS CloudShell

サポートされている では AWS リージョン、 から直接起動するブラウザベースの事前認証済みシェル AWS CloudShell に対して を使用して AWS CLI コマンドを実行できます AWS Management Console。任意のシェル (Bash、または Z シェル) を使用して PowerShell、 サービスに対して AWS CLI コマンドを実行できます。

次の 2 つの方法のいずれか AWS Management Console を使用して、 AWS CloudShell から を起動できます。

- コンソールのナビゲーションバー AWS CloudShell のアイコンを選択します。これは検索ボックスの右側にあります。
- コンソールナビゲーションバーの検索ボックスを使用して を検索しCloudShell、CloudShellオプションを選択します。

が新しいブラウザウィンドウで初めて AWS CloudShell 起動すると、ウェルカムパネルが表示され、主要な機能が一覧表示されます。このパネルを閉じると、シェルがコンソール認証情報を構成および転送する間、ステータスのアップロードが提供されます。コマンドプロンプトが表示されたら、シェルは対話的な操作の準備ができています。

このサービスの詳細については、「AWS CloudShell ユーザーガイド」を参照してください。

AWS CloudFormationで Auto Scaling グループを作成する

Amazon EC2 Auto Scaling は AWS CloudFormation、リソースのモデル化と設定を支援するサービスと統合されているため、 AWS リソースとインフラストラクチャの作成と管理に費やす時間を短縮できます。 AWS 必要なすべてのリソース (Auto Scaling グループなど) を記述したテンプレートを作成し、 AWS CloudFormation それらのリソースを自動的にプロビジョニングして設定します。

を使用すると AWS CloudFormation、テンプレートを再利用して Amazon EC2 Auto Scaling リソースを一貫して繰り返し設定できます。リソースを一度記述すれば、 AWS アカウント 同じリソースを複数のリージョンやリージョンに何度もプロビジョニングできます。

Amazon EC2 Auto Scaling AWS CloudFormation とテンプレート

Amazon EC2 Auto Scaling と関連サービスのリソースをプロビジョニングして設定するには、AWS CloudFormation テンプレートを理解しておく必要があります。テンプレートは、JSON や YAML でフォーマットされたテキストファイルです。これらのテンプレートには、 AWS CloudFormation スタックにプロビジョニングするリソースが記述されています。JSON や YAML に慣れていない場合は、 AWS CloudFormation Designer を使用してテンプレートを使い始めることができます。 AWS CloudFormation 詳細については、「Designer とは AWS CloudFormation」を参照してください。『AWS CloudFormation ユーザーガイド』の。

Amazon EC2 Auto Scaling に独自のスタックテンプレートの作成を開始するには、次のタスクを完了してください。

- を使用して起動テンプレートを作成しますAWS::EC2::LaunchTemplate。
- グループグループを使用して Auto Scaling AWS::AutoScaling::AutoScaling。

Auto Scaling グループを Application Load Balancer の背後にデプロイする方法を表示するウォークスルーについては、「AWS CloudFormation ユーザーガイド」の「チュートリアル: スケーラブルなロードバランシングウェブサーバーの作成」を参照してください。

Auto Scaling グループと関連リソースを作成するテンプレートスニペットのその他の便利な例は、AWS CloudFormation ユーザーガイドの以下のセクションにあります。

- Amazon EC2 Auto Scaling リソースタイプリファレンス
- Amazon EC2 Auto Scaling リソースを次のように設定します AWS CloudFormation

以下についてさらに詳しく AWS CloudFormation

詳細については AWS CloudFormation、以下のリソースを参照してください。

- AWS CloudFormation
- AWS CloudFormation ユーザーガイド
- AWS CloudFormation API リファレンス
- AWS CloudFormation コマンド・ライン・インターフェース・ユーザー・ガイド

AWS Compute Optimizer を使用して Auto Scaling グループのインスタンスタイプのレコメンデーションを取得する

AWS は、 の機能を使用して、パフォーマンスの向上、コストの削減、またはその両方に役立つ Amazon EC2 インスタンスのレコメンデーションを提供します AWS Compute Optimizer。これらの推奨事項を使用して、新しいインスタンスタイプに移行するかどうかを判断できます。

推奨事項を作成するために、Compute Optimizer は既存インスタンスの仕様と最近のメトリックス履歴を分析します。次に、コンパイルされたデータを使用して、既存のパフォーマンスワークロードを処理するために最適な Amazon EC2 インスタンスタイプを推奨します。推奨事項は、時間あたりのインスタンス料金とともに返されます。

> **Note**
>
> Compute Optimizer から推奨事項を取得するには、まず Compute Optimizer にオプトインする必要があります。詳細については、「AWS Compute Optimizer ユーザーガイド」の「AWS Compute Optimizerの使用開始」を参照してください。

内容

- 制限事項
- 結果
- 推奨事項の表示
- 推奨事項の評価に関する考慮事項

制限事項

Compute Optimizer は、M、C、R、T、X のインスタンスタイプを起動して実行するように設定された Auto Scaling グループ内のインスタンスの推奨事項を生成します。ただし、 AWS Graviton2 プロセッサを搭載した -g インスタンスタイプ (C6g など) と、ネットワーク帯域幅のパフォーマンスが高い -n インスタンスタイプ (M5n など) のレコメンデーションは生成されません。

また、Auto Scaling グループは、単一のインスタンスタイプを実行するように設定する必要があります (つまり、インスタンスタイプが混在しない) 。また、スケーリングポリシーがアタッチされておらず、希望キャパシティ、最小キャパシティ、最大キャパシティ (インスタンス数が固定されている Auto Scaling グループ) に対して同じ値を持つ必要があります。Compute Optimizer は、これらの構成要件のすべてを満たす Auto Scaling グループのインスタンス推奨事項を生成します。

結果

Compute Optimizer は、Auto Scaling グループの調査結果を次のように分類します。

- 最適化されていない – Compute Optimizer がワークロードのパフォーマンスを向上できる推奨事項を特定した場合、Auto Scaling グループは、最適化されていないとみなされます。
- 最適化 – 選択したインスタンスタイプに基づいて、ワークロードを実行するためにグループが正しくプロビジョニングされていると Compute Optimizer が判断した場合、Auto Scaling グループは、最適化されていると見なされます。最適化されたリソースについては、Compute Optimizer が新世代のインスタンスタイプを推奨することがあります。
- なし – Auto Scaling グループの推奨事項はありません。これは、Compute Optimizer を 12 時間未満にオプトインした場合、または Auto Scaling グループの実行が 30 時間未満の場合、または Auto Scaling グループまたはインスタンスタイプが Compute Optimizer でサポートされていない場合に発生する可能性があります。詳細については、「制限事項」セクションを参照してください。

推奨事項の表示

Compute Optimizer にオプトインすると、Auto Scaling グループに対して生成された結果と推奨事項を表示できます。最近オプトインした場合、推奨事項が最大 12 時間、反映されないことがあります。

Auto Scaling グループに対して生成された推奨事項を表示する

1. https://console.aws.amazon.com/compute-optimizer/ で、Compute Optimizer コンソールを開きます。

   ダッシュボードページが開きます。
2. [View recommendations for all Auto Scaling groups (すべての Auto Scaling グループの推奨事項を表示する)] を選択します。
3. Auto Scaling スケーリンググループを選択します。
4. [View detail (詳細を表示)] を選択します。

   デフォルトのテーブル設定に基づいて、事前構成されたビューに最大 3 つの異なるインスタンスの推奨事項が表示されるように、ビューが変更されます。また、Auto Scaling グループの最近の CloudWatch メトリクスデータ (平均 CPU 使用率、平均ネットワーク入力、平均ネットワーク出力) も提供します。

推奨事項の 1 つを使用するかどうかを決定します。パフォーマンスの向上のために最適化するか、コスト削減のために最適化するか、これら 2 つの組み合わせのために最適化するかを決定します。

Auto Scaling グループのインスタンスタイプを変更するには、起動テンプレートを更新するか、Auto Scaling group を更新して新しい起動設定を使用します。既存のインスタンスでは、引き続き以前の設定を使用します。既存のインスタンスを更新するには、これらのインスタンスを終了して Auto Scaling グループに置き換えるようにするか、オートスケーリングにより終了ポリシーに基づいて古いインスタンスを新しいインスタンスに徐々に置き換えるようにします。

> **Note**
>
> インスタンスの最大有効期間とインスタンスの更新機能を使用すると、Auto Scaling グループ内の既存のインスタンスを置き換えて、新しい起動テンプレートや起動設定を使用する新しいインスタンスを起動することもできます。詳細については、「インスタンスの最大存続

期間に基づいて Auto Scaling インスタンスを置き換える」および「インスタンスの更新を使用して Auto Scaling グループのインスタンスを更新する」を参照してください。

推奨事項の評価に関する考慮事項

新しいインスタンスタイプに移行する前に、次の点を考慮してください。

- 推奨情報は使用状況を予測するものではありません。推奨事項は、直近の 14 日間の使用履歴に基づいています。将来の使用ニーズを満たすことが予想されるインスタンスタイプをかならず選択してください。
- グラフ化されたメトリクスを参考にして、実際の使用量がインスタンスのキャパシティーよりも低いかどうかを判断します。メトリクスデータ (平均、ピーク、パーセンタイル) を で表示 CloudWatch して、EC2 インスタンスのレコメンデーションをさらに評価することもできます。例えば、一日の CPU パーセンテージメトリクスがどのように変化するか、ピークに対応する必要があるかどうかに注目します。詳細については、「Amazon CloudWatch ユーザーガイド」の「使用可能なメトリクスの表示」を参照してください。
- Compute Optimizer は、バーストパフォーマンスインスタンス (T3、T3a、および T2 インスタンス) の推奨事項を提供する場合があります。ベースラインを定期的に上回る場合は、新しいインスタンスタイプの vCPU に基づいて引き続きバーストを実行できることを確認します。詳細については、「Amazon EC2 ユーザーガイド」の「バーストパフォーマンスインスタンスの CPU クレジットとベースラインパフォーマンスAmazon EC2」を参照してください。
- リザーブドインスタンスを購入した場合、オンデマンドインスタンスはリザーブドインスタンスとして請求される場合があります。現在のインスタンスタイプを変更する前に、まずリザーブドインスタンスの使用率と適用範囲に対する影響を評価します。
- 可能であれば、新世代のインスタンスへの交換を検討します。
- 別のインスタンスファミリーに移行する場合は、仮想化、アーキテクチャー、ネットワークタイプなどの点で、現在のインスタンスタイプと新しいインスタンスタイプに互換性があることを確認してください。詳細については、「Amazon EC2 ユーザーガイド」の「インスタンスのサイズ変更の互換性Amazon EC2」を参照してください。
- 最後に、推奨事項ごとに提供されるパフォーマンスリスク評価を検討します。パフォーマンスリスクは、推奨されるインスタンスタイプがワークロードのパフォーマンス要件を満たすかどうかを検証するために費やす必要のある作業量を示します。また、変更前と変更後に厳格な負荷テストおよびパフォーマンステストを行うことをお勧めします。

追加リソース

このページのトピックに加えて、次のリソースも参照してください。

- Amazon EC2 インスタンスタイプ
- AWS Compute Optimizer ユーザーガイド

Elastic Load Balancing を使用して Auto Scaling グループ内のインスタンス全体にトラフィックを分散させる

Elastic Load Balancing は、実行しているすべての EC2 インスタンス間で、受信したアプリケーションのトラフィックを自動的に分散させます。Elastic Load Balancing は、どのインスタンスにも負荷がかからないように、トラフィックを最適にルーティングすることで受信したリクエストを管理します。

Auto Scaling グループで Elastic Load Balancing を使用するには、Auto Scaling グループにロードバランサーをアタッチする。これにより、グループがロードバランサーに登録され、ロードバランサーは、Auto Scaling グループへのすべての受信ウェブトラフィックの 1 つのお問合せポイントとして機能します。

Auto Scaling グループで Elastic Load Balancing を使用する場合、ロードバランサーまたはターゲットグループに個々の EC2 インスタンスをロードバランサーに登録する必要はありません。Auto Scaling グループによって起動されたインスタンスは、自動的にロードバランサーのメンバーとなります。同様に、Auto Scaling グループによって終了されたインスタンスは、ロードバランサーから自動的に登録解除されます。

ロードバランサーを Auto Scaling グループにアタッチした後、Elastic Load Balancing メトリクス (ターゲットあたりの Application Load Balancer のリクエスト数など)を使用して、需要の変化に応じてグループ内のインスタンス数をスケールするように Auto Scaling グループを設定できます。

必要に応じて、Auto Scaling グループに Elastic Load Balancing ヘルスチェックを追加できます。これにより、Amazon EC2 Auto Scaling はこれらの追加のヘルスチェックに基づいて異常なインスタンスを識別して置き換えることができます。それ以外の場合は、ターゲットグループの正常なホスト数が許可を下回った場合に通知する CloudWatch アラームを作成できます。

内容

- Elastic Load Balancing のタイプ

- Elastic Load Balancing ロードバランサーを Auto Scaling グループにアタッチする準備をします。
- Auto Scaling グループに Elastic Load Balancing ロードバランサーをアタッチする Auto Scaling
- Amazon EC2 Auto Scaling コンソールから Application Load Balancer または Network Load Balancer を設定する
- ロードバランサーのアタッチメントステータスを確認する
- アベイラビリティーゾーンを追加および削除する
- での Elastic Load Balancing の使用例 AWS Command Line Interface

Elastic Load Balancing のタイプ

Elastic Load Balancing は、Auto Scaling グループで使用できる四つのタイプのロードバランサーを提供します:それらは、Application Load Balancer、Network Load Balancer、Gateway Load Balancer、Classic Load Balancer です。

ロードバランサーの設定方法は、種類によって大きく異なります。Application Load Balancer、Network Load Balancer、Gateway Load Balancer で、インスタンスはターゲットグループにターゲットとしてメンバーとされ、トラフィックをターゲットグループに送信します。Classic Load Balancer で、インスタンスはロードバランサーに直接メンバーとされます。

Application Load Balancer

ルーティングと負荷分散をアプリケーションレイヤー (HTTP/HTTPS) で行い、パスベースのルーティングをサポートしています。Application Load Balancer は、仮想プライベートクラウド (VPC) の EC2 instancesのように、1 つまたは複数のメンバーとなったターゲット上のポート にリクエストを送信することができます。

Network Load Balancer

レイヤー 4 ヘッダーから抽出されたアドレス情報に基づいて、トランスポートレイヤー (TCP/UDP レイヤー 4) でルーティングとロードバランシングを行います。Network Load Balancer は、ロードバランサーの有効期間中、トラフィックバーストを処理し、クライアントの出典 IP を保持して、固定 IP を使用します。

Gateway Load Balancer

アプライアンス· インスタンスのフリートにトラフィックを分散します。ファイアウォール、侵入検知および防止システム、その他のアプライアンスなど、サードパーティー製の仮想アプライアンスのスケール、可用性、およびシンプルさを提供します。Gateway Load Balancer

は、GENEVEプロトコルをサポートする仮想アプライアンスと連携します。追加の技術統合が必要なため、Gateway Load Balancer を選択する前に、必ずユーザーガイドを参照してください。

Classic Load Balancer

トランスポートレイヤー (TCL/SSL) あるいはアプリケーションレイヤー (HTTP/HTTPS) のいずれかで行うルーティングあるいはロードバランサー。

利用可能なさまざまなタイプのロードバランサーについてより深く理解するには、次のリソースを参照してください。

- Elastic Load Balancing とは?
- Application Load Balancer とは?
- Network Load Balancer とは?
- Gateway Load Balancer とは ?
- Classic Load Balancer とは ?

Elastic Load Balancing ロードバランサーを Auto Scaling グループにアタッチする準備をします。

Elastic Load Balancing ロードバランサーを Auto Scaling グループにアタッチする前に、次の前提条件を満たす必要があります。

- Auto Scaling グループにトラフィックをルーティングするために使用されるロードバランサーとターゲットグループが既に作成されている必要があります。

  ロードバランサーとターゲットグループを作成するには、次の 2 つの方法があります。

  - Elastic Load Balancing の使用 – Auto Scaling グループを作成する前に、Elastic Load Balancing ドキュメントの手順に従ってロードバランサーとターゲットグループを作成して設定します。Amazon EC2 インスタンスを登録するステップは省略します。ターゲットグループを Auto Scaling グループにアタッチすると、Amazon EC2 Auto Scaling がインスタンスの登録 (および登録解除) を自動的に処理します。詳細については、Elastic Load Balancing ユーザーガイドの Elastic Load Balancing で使用開始 を参照してください。
  - Amazon EC2 Auto Scaling の使用 — Amazon EC2 Auto Scaling コンソールから基本的な設定を使用して、ロードバランサーとターゲットグループを作成、設定、アタッチします。詳細については、「Amazon EC2 Auto Scaling コンソールから Application Load Balancer または Network Load Balancer を設定する」を参照してください。

- ロードバランサーを作成する前に、必要なロードバランサーのタイプを確認してください。詳細については、「Elastic Load Balancing のタイプ」を参照してください。
- ロードバランサーとそのターゲットグループは AWS アカウント、Auto Scaling グループと同じ 、VPC、リージョンに存在する必要があります。
- ターゲットグループは、`instance` のターゲットタイプを指定する必要があります。Auto Scaling グループを使用する場合、`ip` のターゲットタイプを指定することはできません。
- Auto Scaling グループの起動テンプレートに、ロードバランサーからの必要なインバウンドトラフィックを許可する正しいセキュリティグループが含まれていない場合は、起動テンプレートを更新する必要があります。推奨されるルールは、ロードバランサーのタイプと、ロードバランサーが使用するバックエンドのタイプによって異なります。例えば、トラフィックをウェブ サーバーに送信するには、ロードバランサーからポート 80 でインバウンド HTTP アクセスを許可します。起動テンプレートが変更されても、既存のインスタンスは新しい設定に更新されません。既存のインスタンスを更新するには、インスタンスの更新を開始してインスタンスを置き換えます。詳細については、「インスタンスの更新を使用して Auto Scaling グループのインスタンスを更新する」を参照してください。
- 起動テンプレートのセキュリティグループでは、Elastic Load Balancing がヘルスチェックを実行するために、正しいポートのロードバランサーからのアクセスも許可する必要があります。
- Gateway Load Balancer の背後に仮想アプライアンスをデプロイする場合、起動テンプレートの Amazon マシンイメージ (AMI) は GENEVE プロトコルをサポートする AMI の ID を指定して、Auto Scaling グループが Gateway Load Balancer とトラフィックを交換できるようにする必要があります。また、起動テンプレートのセキュリティグループは、ポート 6081 で UDP トラフィックを許可する必要があります。

> **Tip**
>
> 完了に時間がかかるブートストラップスクリプトがある場合、必要に応じて起動ライフサイクルフックを Auto Scaling グループに追加して、ブートストラップスクリプトが正常に完了し、インスタンス上のアプリケーションでトラフィックを受け入れる準備ができるまで、ロードバランサーの背後でのインスタンス登録を遅延させることができます。Amazon EC2 Auto Scaling コンソールで Auto Scaling グループを初めて作成するときにライフサイクルフックを追加することはできません。ただし、グループの作成後にライフサイクルフックを追加できます。詳細については、「Amazon EC2 Auto Scaling のライフサイクルフック」を参照してください。

ターゲットのヘルスチェックを設定する

Elastic Load Balancing ロードバランサーに登録されているターゲットのヘルスチェックを設定して、ターゲットがトラフィックを適切に処理できることを確認できます。具体的な手順は、使用しているロードバランサーのタイプによって異なります。詳細については、以下のリソースを参照してください。

- Application Load Balancer – Application Load Balancer のユーザーガイドの「ターゲットグループのヘルスチェック」を参照してください。
- Network Load Balancer – Network Load Balancer のユーザーガイドの「ターゲットグループのヘルスチェック」を参照してください。
- Gateway Load Balancer – Gateway Load Balancer のユーザーガイドの「ターゲットグループのヘルスチェック」を参照してください。
- Classic Load Balancer – Classic Load Balancer のユーザーガイドの「Classic Load Balancer のヘルスチェックを設定する」を参照してください。

デフォルトでは、Amazon EC2 Auto Scaling はインスタンスを異常と見なさず、Elastic Load Balancing のヘルスチェックに失敗した場合に置き換えます。Auto Scaling グループのデフォルトのヘルスチェックは EC2 ヘルスチェックのみです。詳細については、「Auto Scaling グループ内のインスタンスのヘルスチェック」を参照してください。

Amazon EC2 Auto Scaling が Elastic Load Balancing によって異常と報告されたインスタンスを置き換えるようにするには、Elastic Load Balancing ヘルスチェックを使用するように Auto Scaling グループを設定できます。これにより、Amazon EC2 Auto Scaling は、EC2 ヘルスチェックまたは Elastic Load Balancing ヘルスチェックのいずれかに失敗した場合、インスタンスを異常と見なします。複数のロードバランサー ターゲットグループまたは Classic Load Balancer をグループにアタッチする場合、インスタンスが正常と見なされるためには、すべてのロードバランサーが、インスタンスは正常であるとして報告する必要があります。ロードバランサーの 1 つがインスタンスを異常として報告した場合は、他のロードバランサーがこれを正常として報告した場合でも、Auto Scaling グループ はそのインスタンスを置き換えます。

Auto Scaling グループでこれらのヘルスチェックを有効にする方法については、「」を参照してくださいAuto Scaling グループに Elastic Load Balancing ロードバランサーをアタッチする Auto Scaling。

> **Note**
>
> これらのヘルスチェックができるだけ早く開始されるようにするには、グループのヘルスチェックの猶予期間が高すぎず、Elastic Load Balancing のヘルスチェックがターゲットがリクエストを処理できるかどうかを判断できるほど高いことを確認してください。詳細については、「Auto Scaling グループにヘルスチェックの猶予期間を設定する」を参照してください。

Auto Scaling グループに Elastic Load Balancing ロードバランサーをアタッチする Auto Scaling

このトピックでは、Elastic Load Balancing ロードバランサーを Auto Scaling グループにアタッチする方法について説明します。また、Elastic Load Balancing ヘルスチェックを有効にして、Elastic Load Balancing が異常とレポートするインスタンスを Amazon EC2 Auto Scaling で置き換える方法についても説明します。

デフォルトでは、Amazon EC2 Auto Scaling は、Amazon EC2 ヘルスチェックに基づいて、異常なインスタンスまたはアクセスできないインスタンスのみを置き換えます。Elastic Load Balancing ヘルスチェックを有効にすると、Auto Scaling グループにアタッチした Elastic Load Balancing ロードバランサーのいずれかが異常と報告した場合、Amazon EC2 Auto Scaling は実行中のインスタンスを置き換えることができます。

Application Load Balancer を Auto Scaling グループにアタッチするチュートリアルについては、「」を参照してくださいチュートリアル: スケーリングとロードバランシングを使用するアプリケーションのセットアップ。

> **Important**
>
> 先に進む前に、前のセクションのすべての前提条件を満たしてください。

内容

- ターゲットグループまたは Classic Load Balancer をアタッチする
- ターゲットグループまたは Classic Load Balancer をデタッチする

ターゲットグループまたは Classic Load Balancer をアタッチする

Auto Scaling グループを作成または更新するときに、1 つ以上のターゲットグループまたは Classic Load Balancer をアタッチできます。Application Load Balancer 、Network Load Balancer 、または Gateway Load Balancer をアタッチするときは、ロードバランサー自体ではなくターゲットグループをアタッチします。

このセクションの手順に従い、コンソールを使用して次の操作を実行します。

- ターゲットグループまたは Classic Load Balancer を Auto Scaling グループにアタッチする
- Elastic Load Balancing のヘルスチェックを有効にする

新しい Auto Scaling グループを作成しているときに、既存のロードバランサーをアタッチするには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. 画面上部のナビゲーションバーで、ロードバランサーを作成した AWS リージョン を選択します。
3. [Auto Scaling グループの作成] を選択します。
4. ステップ 1 と 2 では、必要に応じてオプションを選択し、「ステップ 3: アドバンストオプションを設定する」へ進みます。
5. [ロードバランシング] では、[既存のロードバランサーにアタッチ] を選択します。
6. [既存のロードバランサーにアタッチ] で、次のいずれかの操作を行います。

   a. Application Load Balancerでは、Network Load Balancers、および Gateway Load Balancer:

      [ロードバランサーのターゲットグループから選択] を選択して、[Existing load balancer target groups] (既存のロードバランサーターゲットグループ) でターゲットグループを選択します。

   b. Classic Load Balancerでは:

      [Classic Load Balancerから選択] を選択して、[Classic Load Balancer] でロードバランサーを選択します。
7. (オプション) [ヘルスチェック] の [追加のヘルスチェックタイプ] で、[Elastic Load Balancing のヘルスチェックをオンにする] を選択します。
8. (オプション) [ヘルスチェックの猶予期間] に秒単位で時間を入力します。これは、インスタンスが `InService` 状態になった後で、Amazon EC2 Auto Scaling がインスタンスのヘルスステー

タスチェックの実行を待つ必要がある時間です。詳細については、「Auto Scaling グループにヘルスチェックの猶予期間を設定する」を参照してください。

9. Auto Scaling グループの作成に進みます。Auto Scaling グループの作成後、インスタンスは自動的にロードバランサーにメンバーとされます。

Auto Scaling グループの作成後に既存のロードバランサーをアタッチするには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   [Auto Scaling groups] (Auto Scaling グループ) ページの下部にスプリットペインが開きます。
3. [詳細] タブで、[ロードバランシング]、[編集] の順に選択します。
4. [Load balancing] (ロードバランシング) で、次のいずれかの操作を行います。

   a. [Application, Network または Gateway Load Balancer のターゲットグループ] で、そのチェックボックスを選択してターゲットグループを選択します。

   b. [Classic Load Balancer] で、そのチェックボックスを選択してロードバランサーを選択します。
5. [更新] を選択します。

ロードバランサーのアタッチが完了したら、ロードバランサーを使用するヘルスチェックをオプションでオンにできます。

Elastic Load Balancing ヘルスチェックを有効にするには

1. [詳細] タブで、[ヘルスチェック]、[編集] の順に選択します。
2. [ヘルスチェック] の [追加のヘルスチェックタイプ] で、[Elastic Load Balancing のヘルスチェックをオンにする] を選択します。
3. [ヘルスチェックの猶予期間] に、秒単位で時間を入力します。これは、インスタンスが `InService` 状態になった後で、Amazon EC2 Auto Scaling がインスタンスのヘルスステータスチェックの実行を待つ必要がある時間です。詳細については、「Auto Scaling グループにヘルスチェックの猶予期間を設定する」を参照してください。
4. [更新] を選択します。

> **Note**
>
> ロードバランサーがアタッチされている間、 AWS CLIを使用してロードバランサーのステータスを監視できます。Amazon EC2 Auto Scaling がインスタンスを正常に登録し、少なくとも 1 つの登録済みインスタンスがヘルスチェックに合格すると、`InService` のステータスが得られます。詳細については、「ロードバランサーのアタッチメントステータスを確認する」を参照してください。

ターゲットグループまたは Classic Load Balancer をデタッチする

ロードバランサーが不要になった場合、以下の手順に従って、Auto Scaling グループからデタッチします。

グループからロードバランサーをデタッチするには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. 既存のグループの横にあるチェックボックスをオンにします。

   [Auto Scaling groups] (Auto Scaling グループ) ページの下部にスプリットペインが開きます。
3. [詳細] タブで、[ロードバランシング]、[編集] の順に選択します。
4. [ロードバランシング] で、次のいずれかの操作を行います。

   a. [Application, Network または Gateway Load Balancer のターゲットグループ] で、ターゲットグループの横にある削除 (X) アイコンを選択します。

   b. [Classic Load Balancer] で、ロードバランサーの横にある削除 (X) アイコンを選択します。
5. [更新] を選択します。

ターゲットグループのデタッチが完了したら、Elastic Load Balancing ヘルスチェックをオフにできます。

Elastic Load Balancing ヘルスチェックをオフにするには

1. [詳細] タブで、[ヘルスチェック]、[編集] の順に選択します。
2. ヘルスチェック 、その他のヘルスチェックタイプ では、Elastic Load Balancing ヘルスチェックをオンにする の選択を解除します。

3. [更新] を選択します。

Amazon EC2 Auto Scaling コンソールから Application Load Balancer または Network Load Balancer を設定する

以下の手順に従い、Auto Scaling グループを作成するときに、Application Load Balancer または Network Load Balancer を作成してアタッチします。

新しい Auto Scaling グループの作成時に、新しいロードバランサーを作成してアタッチするには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. [Auto Scaling グループの作成] を選択します。
3. ステップ 1 と 2 では、必要に応じてオプションを選択し、「ステップ 3: アドバンストオプションを設定する」へ進みます。
4. [ロードバランシング] で、[新しいロードバランサーにアタッチ] を選択します。
   a. [新しいロードバランサーにアタッチ] で、[ロードバランサーのタイプ] に、Application Load Balancer または Network Load Balancer のいずれを作成するかを選択します。
   b. [ロードバランサーの名前] は、ロードバランサーの名前を入力するか、デフォルトの名前のままにします。
   c. [ロードバランサーのスキーム] で、インターネット向け 公開ロードバランサーを作成するか、内部向けロードバランサーのデフォルトのままにするかを選択します。
   d. [アベイラビリティーゾーンとサブネット] で、EC2 インスタンスを起動するために選択した各アベイラビリティーゾーンに対して、パブリックサブネットを選択します。(これらは、ステップ 2 で事前設定されます)。
   e. [リスナーとルーティング] をクリックし、リスナーのポート番号を更新し (必要な場合)、[デフォルトルーティング] で、[ターゲットグループの作成] を選択します。または、ドロップダウンリストから既存のターゲットグループを選択することができます。
   f. 最後のステップで、[ターゲットグループの作成] を選択した場合、[新しいターゲットグループ名] にターゲットグループの名前を入力するか、デフォルトの名前のままにします。
   g. ロードバランサーにタグを追加するには、[タグの追加] を選択して、タグごとにタグのキーと値を指定します。
5. (オプション) [ヘルスチェック] の [追加のヘルスチェックタイプ] で、[Elastic Load Balancing のヘルスチェックをオンにする] を選択します。

6. (オプション) [ヘルスチェックの猶予期間] に秒単位で時間を入力します。これは、インスタンスが InService 状態になった後で、Amazon EC2 Auto Scaling がインスタンスのヘルスステータスチェックの実行を待つ必要がある時間です。詳細については、「Auto Scaling グループにヘルスチェックの猶予期間を設定する」を参照してください。
7. Auto Scaling グループの作成に進みます。Auto Scaling グループの作成後、インスタンスは自動的にロードバランサーにメンバーとされます。

> **Note**
>
> Auto Scaling グループを作成したら、Elastic Load Balancing コンソールを使用して追加のリスナーを作成できます。これは、HTTPS などの安全なプロトコルや UDP リスナーを使用してリスナーを作成する必要がある場合に便利です。異なるポートを使用していれば、既存のロードバランサーにもっとリスナーを追加できます。

ロードバランサーのアタッチメントステータスを確認する

ロードバランサーは、アタッチされた後、グループのインスタンスを登録している間は、Adding 状態になります。グループのすべてのインスタンスが登録済みになると、Added 状態になります。最低 1 つの登録されたインスタンスがヘルスチェックを通過した後、InService 状態になります。ロードバランサーが InService 状態にある場合、Amazon EC2 Auto Scaling は異常と報告されたインスタンスを 終了し、置き換えることができます。メンバーとされたインスタンスがヘルスチェックに合格しない場合 (例えば、誤って設定されたヘルスチェックが原因)、ロードバランサーは InService 状態に入力しません。Amazon EC2 Auto Scaling はインスタンスを終了し置き換えをすることはありません。

ロードバランサーをデタッチすると、グループのインスタンスの登録解除中、Removing 状態になります。登録解除してもインスタンスは引き続き実行されます。Application Load Balancer、Network Load Balancer、および Gateway Load Balancer では、Connection Draining がデフォルトで有効になっています。Connection Draining が有効になっている場合、Elastic Load Balancing は、処理中のリクエストが完了するまで、または最大タイムアウトの期限が切れるまで (どちらか早い方) 待機してから、インスタンスの登録を解除します。

アタッチメントのステータスは、 AWS Command Line Interface （ AWS CLI) または AWS SDKsを使用して確認できます。アタッチメントステータスはコンソールから確認できません。

を使用してアタッチメントのステータス AWS CLI を確認するには

次の describe-traffic-sources コマンドは、指定した Auto Scaling グループのすべてのトラフィックソースのアタッチメントステータスを返します。

```
aws autoscaling describe-traffic-sources --auto-scaling-group-name my-asg
```

この例では、Auto Scaling グループにアタッチされている Elastic Load Balancing ターゲットグループの ARN と、State エレメント内のターゲットグループのアタッチメントステータスを返します。

```
{
    "TrafficSources": [
        {
            "Identifier": "arn:aws:elasticloadbalancing:region:account-
id:targetgroup/my-targets/1234567890123456",
            "State": "InService",
            "Type": "elbv2"
        }
    ]
}
```

アベイラビリティーゾーンを追加および削除する

理的な冗長性による安全性と信頼性を活用するには、作業しているリージョンの複数のアベイラビリティーゾーンにまたがって Auto Scaling グループを配置し、ロードバランサーをアタッチしてそれらのアベイラビリティーゾーンに受信トラフィックを分散させます。

1 つのアベイラビリティーゾーンが異常または使用不可になったとき、Amazon EC2 Auto Scaling は、影響を受けていないアベイラビリティーゾーンで新しいインスタンスを起動します。異常な状態のアベイラビリティーゾーンが正常な状態に戻ったら、Amazon EC2 Auto ScalingはAuto Scaling グループのすべてのアベイラビリティーゾーンにわたって均等にアプリケーションインスタンスを自動的に再分配します。Amazon EC2 Auto Scaling は、インスタンス数が最も少ないアベイラビリティーゾーンで新しいインスタンスの起動を試みることによって、これを実行します。この試みが失敗すると、Amazon EC2 Auto Scaling は成功するまで他のアベイラビリティーゾーンでの起動を試みます。

Elastic Load Balancing は、ロードバランサーに有効な各アベイラビリティーゾーンにロードバランサー ノードを作成します。ロードバランサーのクロスゾーン ロードバランシングを有効にすると、各ロードバランサー ノードは、有効化されたすべてのアベイラビリティーゾーンのメンバーとされたインスタンスに均等にトラフィックを分配します。クロスゾーン負荷分散が無効の場合は、各ロードバランサーノードは、そのアベイラビリティーゾーンの登録されたインスタンスにのみリクエストを均等に分散します。

Auto Scaling グループを作成しているときは、少なくとも 1 つのアベイラビリティーゾーンを指定する必要があります。その後、Auto Scaling グループにアベイラビリティーゾーンを追加し、そのアベイラビリティーゾーンをロードバランサーに対して有効にすることで (ロードバランサーがサポートしている場合)、アプリケーションの可用性を拡張できます。

内容

- アベイラビリティーゾーンを追加する
- アベイラビリティーゾーンの削除
- 関連リソース
- 制限事項

アベイラビリティーゾーンを追加する

次の手順に従い、追加のアベイラビリティーゾーンのサブネットに Auto Scaling グループとロードバランサーを拡張します。

アベイラビリティーゾーンを追加するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. 既存のグループの横にあるチェックボックスをオンにします。

   [Auto Scaling グループ ページの下部にスプリットペインが開きます。
3. [詳細] タブで、[ネットワーク]、[編集] の順に選択します。
4. [サブネット] で、Auto Scaling グループに追加したいアベイラビリティーゾーンに対応するサブネットの削除 (X) アイコンを選択します。
5. [更新] を選択します。
6. ロードバランサーのアベイラビリティーゾーンを更新して、Auto Scaling グループと同じゾーンを共有するには、以下のステップを完了します:

   a. ナビゲーションペインの [ロードバランシング] で [ロードバランサー] を選択します。

   b. ロードバランサーを選択します。

   c. 次のいずれかを行います。

      - Application Load Balancer および Network Load Balancer の場合:

1. [説明] タブで、[アベイラビリティーゾーン] で、[サブネットを編集する] を選択します。
2. [サブネットを編集する] ページの [アベイラビリティーゾーン] で、追加するアベイラビリティーゾーンのチェックボックスを選択します。そのゾーンにサブネットが 1 つしかない場合は、そのサブネットが選択されています。そのゾーンに複数のサブネットがある場合は、いずれかのサブネットを選択します。

- VPC 内の Classic Load Balancer の場合:

  1. [インスタンス] タブで、[アベイラビリティーゾーンの編集] を選択します。
  2. [サブネットを追加および削除する] ページの [使用可能なサブネット] で、追加 (+) アイコンを使用してサブネットを選択します。サブネットが [選択済みサブネット] に移動します。

d. [保存] を選択します。

アベイラビリティーゾーンの削除

Auto Scaling グループおよびロードバランサーからアベイラビリティーゾーンを削除するには、以下の手順に従います。

アベイラビリティーゾーンを削除するには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. 既存のグループの横にあるチェックボックスをオンにします。

   [Auto Scaling グループ ページの下部にスプリットペインが開きます。
3. [詳細] タブで、[ネットワーク]、[編集] の順に選択します。
4. [サブネット] で、Auto Scaling グループから削除したいアベイラビリティーゾーンに対応するサブネットの削除 (X) アイコンを選択します。そのゾーンに複数のサブネットがある場合は、それぞれの 削除 (X) アイコンを選択します。
5. [更新] を選択します。
6. ロードバランサーのアベイラビリティーゾーンを更新して、Auto Scaling グループと同じゾーンを共有するには、以下のステップを完了します:

   a. ナビゲーションペインの [ロードバランシング] で [ロードバランサー] を選択します。

b. ロードバランサーを選択します。

c. 次のいずれかを行います。

- Application Load Balancer および Network Load Balancer の場合:

  1. [説明] タブで、[アベイラビリティーゾーン] で、[サブネットを編集する] を選択します。
  2. [サブネットの編集] ページの [アベイラビリティーゾーン] で、チェックボックスをオフにするとアベイラビリティーゾーンのサブネットが削除されます。

- VPC 内の Classic Load Balancer の場合:

  1. [インスタンス] タブで、[アベイラビリティーゾーンの編集] を選択します。
  2. [サブネットをの追加と削除] ページの[利用可能なサブネット] で、その削除 (-) アイコンを使用してサブネットを削除します。サブネットが [利用可能なサブネット] の下に移動します。

d. [Save] を選択します。

関連リソース

アベイラビリティーゾーンを変更すると、Amazon EC2 Auto Scaling によってグループが再調整されます。これは、一部のインスタンスを置き換えて再配布することを意味します。詳細については、「例: 複数のアベイラビリティーゾーン全体にインスタンスを分散させる」を参照してください。

ロードバランサーのために有効になっていないアベイラビリティーゾーンにターゲットを登録している場合、そのロードバランサーはそれらのターゲットにトラフィックをルーティングしません。詳細については、Elastic Load Balancing ユーザーガイドの How Elastic Load Balancing works を参照してください。

制限事項

ロードバランサーで有効になっているアベイラビリティーゾーンを更新するには、次の制限事項に注意する必要があります。

- ロードバランサー用のアベイラビリティーゾーンを有効にすると、そのアベイラビリティーゾーンからサブネットを 1 つ指定します。ロードバランサーでは、アベイラビリティーゾーンごとに最大で 1 つのサブネットを有効にできることに注意してください。
- インターネット向けロードバランサーの場合、ロードバランサーに指定するサブネットには最低八個の利用可能な IP アドレスが必要です。

- Application Load Balancer の場合、少なくても 二つのアベイラビリティーゾーンを有効にする必要があります。
- Network Load Balancers の場合、有効になっているアベイラビリティーゾーンを無効にすることはできませんが、追加のアベイラビリティーゾーンを有効にすることはできます。
- Gateway Load Balancer では、有効なアベイラビリティーゾーンを無効にすることはできませんが、追加のアベイラビリティーゾーンを有効にすることはできます。

での Elastic Load Balancing の使用例 AWS Command Line Interface

AWS Command Line Interface （ AWS CLI) を使用して、ロードバランサーとターゲットグループをアタッチ、デタッチ、記述し、Elastic Load Balancing ヘルスチェックを追加および削除し、どのアベイラビリティーゾーンが有効になっているかを変更します。

このトピックでは、Amazon EC2 Auto Scaling の一般的なタスクを実行する AWS CLI コマンドの例を示します。

> **Important**
>
> その他の例については、「AWS CLI コマンドリファレンス」の「aws elbv2」と「aws elb」を参照してください。

内容

- ターゲットグループまたは Classic Load Balancer をアタッチする
- ターゲットグループまたは Classic Load Balancer の説明を表示する
- Elastic Load Balancing のヘルスチェックを追加する
- アベイラビリティーゾーンを変更する
- ターゲットグループまたは Classic Load Balancer をデタッチする
- Elastic Load Balancing のヘルスチェックを削除する
- レガシーコマンド

ターゲットグループまたは Classic Load Balancer をアタッチする

次の create-auto-scaling-group コマンドを使用して Auto Scaling グループを作成し、同時にその Amazon リソースネーム (ARN) を指定してターゲットグループをアタッチします。ターゲットグ

ループは、Application Load Balancer、Network Load Balancer、または Gateway Load Balancer に関連付けることができます。

`--auto-scaling-group-name`、`--vpc-zone-identifier`、`--min-size`、および `--max-size` のサンプルの値を置き換えます。`--launch-template` オプションの場合、*my-launch-template* と *1* を Auto Scaling グループの起動テンプレートの名前とバージョンに置き換えます。`--traffic-sources` オプションの場合、サンプルの ARN を Application Load Balancer、Network Load Balancer、または Gateway Load Balancer のターゲットグループの ARN に置き換えます。

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-asg \
  --launch-template LaunchTemplateName=my-launch-template,Version='1' \
  --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782" \
  --min-size 1 --max-size 5 \
  --traffic-sources "Identifier=arn:aws:elasticloadbalancing:region:account-
id:targetgroup/my-targets/12345678EXAMPLE1"
```

Auto Scaling グループを作成したら、attach-traffic-sources コマンドを使用して、追加のターゲットグループを Auto Scaling グループにアタッチします。

次のコマンドは、同じグループに別のターゲットグループを追加します。

```
aws autoscaling attach-traffic-sources --auto-scaling-group-name my-asg \
  --traffic-sources "Identifier=arn:aws:elasticloadbalancing:region:account-
id:targetgroup/my-targets/12345678EXAMPLE2"
```

別の方法として、Classic Load Balancer をグループにアタッチするには、次の例のように、create-auto-scaling-group または attach-traffic-sources を使用するときに `--traffic-sources` オプションおよび `--type` オプションを指定します。*my-classic-load-balancer* を Classic Load Balancer の名前に置き換えます。`--type` オプションの場合、**elb** の値を指定します。

```
--traffic-sources "Identifier=my-classic-load-balancer" --type elb
```

ターゲットグループまたは Classic Load Balancer の説明を表示する

Auto Scaling グループにアタッチされているロードバランサーまたはターゲットグループの説明を表示するには、次の describe-traffic-sources コマンドを使用します。*my-asg* をグループの名前に置き換えます。

```
aws autoscaling describe-traffic-sources --auto-scaling-group-name my-asg
```

この例では、Auto Scaling グループにアタッチした Elastic Load Balancing ターゲットグループの ARN を返します。

```
{
    "TrafficSources": [
        {
            "Identifier": "arn:aws:elasticloadbalancing:region:account-
id:targetgroup/my-targets/12345678EXAMPLE1",
            "State": "InService",
            "Type": "elbv2"
        },
        {
            "Identifier": "arn:aws:elasticloadbalancing:region:account-
id:targetgroup/my-targets/12345678EXAMPLE2",
            "State": "InService",
            "Type": "elbv2"
        }
    ]
}
```

出力された State フィールドの説明については、「ロードバランサーのアタッチメントステータスを確認する」を参照してください。

Elastic Load Balancing のヘルスチェックを追加する

Auto Scaling グループがインスタンスに対して実行するヘルスチェックに Elastic Load Balancing のヘルスチェックを追加するには、次の update-auto-scaling-group コマンドを実行し、--health-check-type オプションの値として **ELB** を指定します。*my-asg* をグループの名前に置き換えます。

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --health-check-type "ELB"
```

新しいインスタンスは、多くの場合、ヘルスチェックに合格する前に短いウォームアップの時間を必要とします。猶予期間に十分なウォームアップ時間がない場合、インスタンスはトラフィックを処理する準備ができていないように見える可能性があります。Amazon EC2 Auto Scaling は、これらのインスタンスを異常と見なして置き換えることがあります。

ヘルスチェックの猶予期間を更新するには、次の例のように、update-auto-scaling-group を使用するときに --health-check-grace-period オプションを使用します。*300* は、新しいインスタンスに異常が見つかった場合、終了させるまでに稼働させておく秒数に置き換えてください。

```
--health-check-grace-period 300
```

詳細については、「Auto Scaling グループ内のインスタンスのヘルスチェック」を参照してください。

アベイラビリティーゾーンを変更する

アベイラビリティーゾーンの変更には、注意が必要な制限がいくつかあります。詳細については、「制限事項」を参照してください。

Application Load Balancer または Network Load Balancer のアベイラビリティーゾーンを変更するには

1. ロードバランサーのアベイラビリティーゾーンを変更する前に、まず Auto Scaling グループのアベイラビリティーゾーンを更新して、使用しているインスタンスタイプが指定したゾーンで使用できることを確認することをお勧めします。

   Auto Scaling グループのアベイラビリティーゾーンを更新するには、次の update-auto-scaling-group コマンドを使用します。サンプルのサブネット ID を、有効にするアベイラビリティーゾーンのサブネットの ID に置き換えます。指定したサブネットは、以前に有効であったサブネットに置き換わります。*my-asg* をグループの名前に置き換えます。

   ```
   aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
     --vpc-zone-identifier "subnet-41767929,subnet-cb663da2,subnet-8360a9e7"
   ```

2. 次の describe-auto-scaling-groups コマンドを使用して、新しいサブネットのインスタンスが起動されたことを確認します。インスタンスが起動した場合は、インスタンスとそのステータスのリストが表示されます。*my-asg* をグループの名前に置き換えます。

   ```
   aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
   ```

3. 次の set-subnets コマンドを使用して、ロードバランサーのサブネットを指定します。サンプルのサブネット ID を、有効にするアベイラビリティーゾーンのサブネットの ID に置き換えます。アベイラビリティーゾーンごとに 1 つだけサブネットを指定できます。指定したサブネットは、以前に有効であったサブネットに置き換わります。*my-lb-arn* を、使用しているロードバランサーの ARN に置き換えます。

```
aws elbv2 set-subnets --load-balancer-arn my-lb-arn \
  --subnets subnet-41767929 subnet-cb663da2 subnet-8360a9e7
```

Classic Load Balancer のアベイラビリティーゾーンを変更するには

1. ロードバランサーのアベイラビリティーゾーンを変更する前に、まず Auto Scaling グループのアベイラビリティーゾーンを更新して、使用しているインスタンスタイプが指定したゾーンで使用できることを確認することをお勧めします。

   Auto Scaling グループのアベイラビリティーゾーンを更新するには、次の update-auto-scaling-group コマンドを使用します。サンプルのサブネット ID を、有効にするアベイラビリティーゾーンのサブネットの ID に置き換えます。指定したサブネットは、以前に有効であったサブネットに置き換わります。*my-asg* をグループの名前に置き換えます。

   ```
   aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
     --vpc-zone-identifier "subnet-41767929,subnet-cb663da2"
   ```

2. 次の describe-auto-scaling-groups コマンドを使用して、新しいサブネットのインスタンスが起動されたことを確認します。インスタンスが起動した場合は、インスタンスとそのステータスのリストが表示されます。*my-asg* をグループの名前に置き換えます。

   ```
   aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
   ```

3. 次の attach-load-balancer-to-subnets コマンドを使用して Classic Load Balancer の新しいアベイラビリティーゾーンを有効にします。サンプルのサブネット ID を、有効にするアベイラビリティーゾーンのサブネットの ID に置き換えます。*my-lb* を、使用しているロードバランサーの名前に置き換えます。

   ```
   aws elb attach-load-balancer-to-subnets --load-balancer-name my-lb \
     --subnets subnet-cb663da2
   ```

   アベイラビリティーゾーンを無効にするには、次の detach-load-balancer-from-subnets コマンドを使用します。サンプルのサブネット ID を、無効にするアベイラビリティーゾーンのサブネットの ID に置き換えます。*my-lb* を、使用しているロードバランサーの名前に置き換えます。

   ```
   aws elb detach-load-balancer-from-subnets --load-balancer-name my-lb \
   ```

```
  --subnets subnet-8360a9e7
```

ターゲットグループまたは Classic Load Balancer をデタッチする

ターゲットグループが不要になった場合、次の detach-traffic-sources コマンドを使用して、Auto Scaling グループからターゲットグループをデタッチします。

`--auto-scaling-group-name` オプションの場合、*my-asg* を使用しているグループの名前に置き換えます。`--traffic-sources` オプションの場合、サンプルの ARN を Application Load Balancer、Network Load Balancer、または Gateway Load Balancer のターゲットグループの ARN に置き換えます。

```
aws autoscaling detach-traffic-sources --auto-scaling-group-name my-asg \
  --traffic-sources "Identifier=arn:aws:elasticloadbalancing:region:account-
id:targetgroup/my-targets/1234567890123456"
```

Classic Load Balancer をグループからデタッチするには、次の例のように `--traffic-sources` オプションと `--type` オプションを指定します。*my-classic-load-balancer* を Classic Load Balancer の名前に置き換えます。`--type` オプションの場合、**elb** の値を指定します。

```
--traffic-sources "Identifier=my-classic-load-balancer" --type elb
```

Elastic Load Balancing のヘルスチェックを削除する

Elastic Load Balancing のヘルスチェックを Auto Scaling グループから削除するには、次の update-auto-scaling-group コマンドを使用し、`--health-check-type` オプションの値として **EC2** を指定します。*my-asg* をグループの名前に置き換えます。

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --health-check-type "EC2"
```

詳細については、「Auto Scaling グループ内のインスタンスのヘルスチェック」を参照してください。

レガシーコマンド

以下の例は、レガシー CLI コマンドを使用してロードバランサーとターゲットグループをアタッチおよびデタッチする方法と、それらの説明を表示する方法を示しています。これらは、お客様が使

用する際の参照用として、このドキュメントに残してあります。レガシー CLI コマンドは引き続きサポートされますが、新しい「トラフィックソース」CLI コマンドは複数のトラフィックソースタイプをアタッチおよびデタッチできるので、こちらの使用をお勧めします。レガシー CLI コマンドと「トラフィックソース」CLI コマンドの両方を同じ Auto Scaling グループで使用できます。

ターゲットグループまたは Classic Load Balancer (レガシー) をアタッチする

ターゲットグループをアタッチするには

次の create-auto-scaling-group コマンドは、アタッチされたターゲットグループを使用して、Auto Scaling グループを作成します。Application Load Balancer、Network Load Balancer、または Gateway Load Balancer のターゲットグループの Amazon リソースネーム (ARN) を指定します。

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-asg \
  --launch-template LaunchTemplateName=my-launch-template,Version='1' \
  --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782" \
  --target-group-arns "arn:aws:elasticloadbalancing:region:account-id:targetgroup/my-targets/1234567890123456" \
  --min-size 1 --max-size 5
```

以下の attach-load-balancer-target-groups コマンドは、既存の Auto Scaling グループにターゲットグループをアタッチします。

```
aws autoscaling attach-load-balancer-target-groups --auto-scaling-group-name my-asg \
  --target-group-arns "arn:aws:elasticloadbalancing:region:account-id:targetgroup/my-targets/1234567890123456"
```

Classic Load Balancer をアタッチするには

以下の create-auto-scaling-group コマンドは、Classic Load Balancer がアタッチされた Auto Scaling グループを作成します。

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-asg \
  --launch-configuration-name my-launch-config \
  --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782" \
  --load-balancer-names "my-load-balancer" \
  --min-size 1 --max-size 5
```

以下の attach-load-balancers コマンドは、指定された Classic Load Balancer を既存の Auto Scaling グループにアタッチします。

```
aws autoscaling attach-load-balancers --auto-scaling-group-name my-asg \
  --load-balancer-names my-lb
```

ターゲットグループまたは Classic Load Balancer (レガシー) の説明を表示する

ターゲットグループの説明を表示するには

Auto Scaling グループに関連付けられているターゲットグループの説明を表示するには、describe-load-balancer-target-groups コマンドを使用します。次の例では、*my-asg*のターゲットグループの一覧を表示します。

```
aws autoscaling describe-load-balancer-target-groups --auto-scaling-group-name my-asg
```

Classic Load Balancer の説明を表示するには

Auto Scaling グループに関連付けられている Classic Load Balancer の説明を表示するには、describe-load-balancers コマンドを使用します。次の例では、*my-asg* の Classic Load Balancer を一覧表にしています。

```
aws autoscaling describe-load-balancers --auto-scaling-group-name my-asg
```

ターゲットグループまたは Classic Load Balancer (レガシー) をデタッチする

ターゲットグループをデタッチするには

ターゲットグループが不要になった場合、以下の detach-load-balancer-target-groups コマンドを使用して、Auto Scaling グループからターゲットグループをデタッチします。

```
aws autoscaling detach-load-balancer-target-groups --auto-scaling-group-name my-asg \
  --target-group-arns "arn:aws:elasticloadbalancing:region:account-id:targetgroup/my-targets/1234567890123456"
```

Classic Load Balancer をデタッチするには

ロードバランサーが不要になったら、以下の detach-load-balancers コマンドが、Auto Scaling グループから Classic Load Balancer をデタッチします。

```
aws autoscaling detach-load-balancers --auto-scaling-group-name my-asg \
```

```
  --load-balancer-names my-lb
```

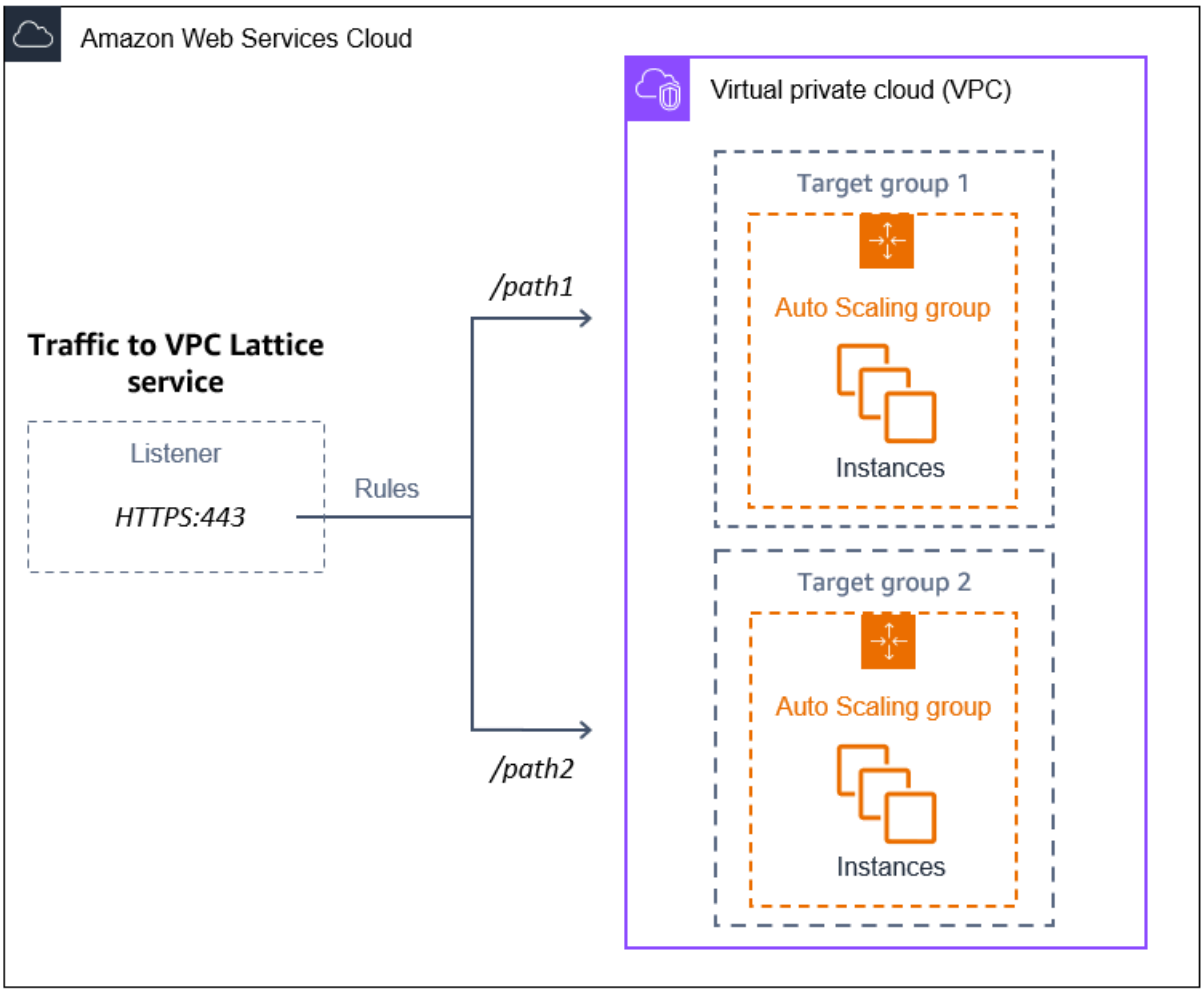
VPC Lattice ターゲットグループを使用して、トラフィックを Auto Scaling グループにルーティングする

Amazon VPC Lattice を使用すると、Auto Scaling グループや Lambda 関数など、個別のリソースで実行されるアプリケーションとサービス間のトラフィックおよび API コールのフローを管理できます。VPC Lattice は、複数のアカウントおよび仮想プライベートクラウド (VPC) にわたって、すべてのサービスを接続、保護、およびモニタリングできるアプリケーションネットワーキングサービスです。VPC Lattice の詳細については、「VPC Lattice とは?」を参照してください。

VPC Lattice の使用を開始するには、まず、サービスネットワークに関連付けられた VPC 内のリソースが互いに接続できるようにするために必要な VPC Lattice リソースを作成します。これらのリソースには、サービス、リスナー、リスナールール、およびターゲットグループが含まれます。

Auto Scaling グループを VPC Lattice サービスに関連付けるには、インスタンス ID で登録したインスタンスへのリクエストをルーティングするサービスのターゲットグループを作成し、リクエストをターゲットグループに送信するリスナーを追加します。次に、ターゲットグループを Auto Scaling グループにアタッチします。Amazon EC2 Auto Scaling は、EC2 インスタンスをターゲットとしてターゲットグループに自動的に登録します。その後、Amazon EC2 Auto Scaling がインスタンスを終了する必要がある場合、終了前にターゲットグループからインスタンスを自動的に登録解除します。

ターゲットグループをアタッチすると、そのグループが Auto Scaling グループへのすべての受信リクエストのエントリポイントになります。続いて、下図の例が示すように、VPC Lattice サービスで指定したリスナールールを使用して、受信リクエストを適切なターゲットグループにルーティングします。



トラフィックが VPC Lattice 経由で Auto Scaling グループにルーティングされると、VPC Lattice はラウンドロビン負荷分散を使用してグループ内のインスタンス間でリクエストの負荷を分散させます。また、VPC Lattice は登録済みのインスタンスの正常性をモニタリングし、トラフィックを正常なインスタンスにのみルーティングすることができます。

受信リクエストに対してインスタンスを使用できるようにしておくために、必要に応じて VPC Lattice ヘルスチェックを Auto Scaling グループに追加します。このようにしておくと、いずれかの EC2 インスタンスに障害が発生した場合でも、Auto Scaling グループで新しいインスタンスが起動され、自動的に置き換えられます。VPC Lattice ヘルスチェックの動作は、Elastic Load Balancing ヘルスチェックの動作と似ています。Auto Scaling グループのデフォルトのヘルスチェックは EC2 ヘルスチェックのみです。

VPC Lattice の詳細については、 AWS ブログの「Simplify Service-to-Service Connectivity, Security, and Monitoring with Amazon VPC Lattice – Now General Available」を参照してください。

内容

- Auto Scaling グループに VPC Lattice ターゲットグループをアタッチする準備を行う
- VPC Lattice ターゲットグループを Auto Scaling グループにアタッチする
- VPC Lattice ターゲットグループのアタッチメントステータスを確認する

Auto Scaling グループに VPC Lattice ターゲットグループをアタッチする準備を行う

Auto Scaling グループに VPC Lattice ターゲットグループをアタッチする前に、以下の前提条件を満たす必要があります。

- VPC Lattice サービスネットワーク、サービス、リスナー、およびターゲットグループの作成を既に行っている必要があります。詳細については、「Amazon Lattice ユーザーガイド」の次のトピックを参照してください。
  - サービスネットワーク
  - サービス
  - リスナー
  - ターゲットグループ
- ターゲットグループは AWS アカウント、Auto Scaling グループと同じ 、VPC、リージョンに存在する必要があります。
- ターゲットグループは、`instance` のターゲットタイプを指定する必要があります。Auto Scaling グループを使用する場合、`ip` のターゲットタイプを指定することはできません。
- ターゲットグループを Auto Scaling グループにアタッチするには、十分な IAM アクセス許可が付与されている必要があります。次のポリシー例は、ターゲットグループをアタッチおよびデタッチするために必要な最小限のアクセス許可を示しています。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "autoscaling:AttachTrafficSources",
                "autoscaling:DetachTrafficSources",
                "autoscaling:DescribeTrafficSources",
                "vpc-lattice:RegisterTargets",
                "vpc-lattice:DeregisterTargets"
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

- Auto Scaling グループの起動テンプレートに VPC Lattice の正しい設定 (互換性のあるセキュリティグループなど) が含まれていない場合は、起動テンプレートを更新する必要があります。起動テンプレートが変更されても、既存のインスタンスは新しい設定に更新されません。既存のインスタンスを更新するには、インスタンスの更新を開始してインスタンスを置き換えます。詳細については、「インスタンスの更新を使用して Auto Scaling グループのインスタンスを更新する」を参照してください。
- Auto Scaling グループで VPC Lattice ヘルスチェックを有効にする前に、アプリケーションベースのヘルスチェックを設定して、アプリケーションが期待どおりに応答していることを確認できます。詳細については、「VPC Lattice ユーザーガイド」の「ターゲットグループのヘルスチェック」を参照してください。

セキュリティグループ: インバウンドルールとアウトバウンドルール

セキュリティグループは、関連付けられた EC2 インスタンスのファイアウォールとして機能し、インバウンドトラフィックとアウトバウンドトラフィックの両方をインスタンスレベルでコントロールします。

> **Note**
>
> ネットワーク設定は非常に複雑であるため、VPC Lattice で使用するセキュリティグループを新たに作成することを強くお勧めします。また、 AWS Support に連絡する必要がある場合に、 がサポートしやすくなります。次のセクションは、ユーザーがこの推奨事項に従うことを前提とした内容になっています。
>
> Auto Scaling グループで使用できる VPC Lattice のセキュリティグループを作成する方法の詳細については、「VPC Lattice ユーザーガイド」の「セキュリティグループを使用してトラフィックを制御する」を参照してください。トラフィックフローの問題をトラブルシューティングするには、「VPC Lattice ユーザーガイド」で詳細を確認してください。

セキュリティグループの作成方法については、Amazon EC2 ユーザーガイド」の「セキュリティグループの作成」および次の表を使用して、選択するオプションを決定します。

オプション	値
名前	覚えやすい名前。
説明	セキュリティグループの識別に役立つ説明。
VPC	Auto Scaling グループと同じ VPC。

インバウンドルール

セキュリティグループを作成するときには、インバウンドルールはありません。インバウンドルールをセキュリティグループに追加するまで、VPC Lattice サービスネットワーク内のクライアントからインスタンスに送信されるインバウンドトラフィックは許可されません。

VPC Lattice サービスネットワーク内のクライアントが Auto Scaling グループ内のインスタンスに接続できるようにするには、Auto Scaling グループのセキュリティグループを正しく設定する必要があります。この場合、特定の IP アドレスではなく、VPC Lattice の AWS マネージドプレフィックスリストの名前からのトラフィックを許可するインバウンドルールを指定します。VPC Lattice のプレフィックスリストは、VPC Lattice が使用する IP アドレスの範囲を CIDR 表記で表したものです。詳細については、「Amazon VPC ユーザーガイド」の AWS「 マネージドプレフィックスリストの操作」を参照してください。

セキュリティグループにルールを追加する方法の詳細については、「Amazon VPC ユーザーガイド」の「ルールをセキュリティグループに追加する」を参照し、次の表を使用して、選択するオプションを決定してください。

オプション	値
HTTP ルール	Type: HTTP ソース: com.amazonaws.*region*.vpc-lattice
HTTPS ルール	タイプ: HTTPS

オプション	値	
	ソース: com.amazonaws.*region*.vpc-lattice	

セキュリティグループはステートフルです。VPC Lattice サービスネットワーク内のクライアントから Auto Scaling グループ内のインスタンスへのトラフィックを許可し、以前に離れたクライアントに応答を返します。

アウトバウンドルール

デフォルトでは、セキュリティグループにはすべてのアウトバウンドトラフィックを許可するアウトバウンドルールが含まれています。必要に応じて、このデフォルトルールを削除し、特定のセキュリティニーズに対応するアウトバウンドルールを追加できます。

制限事項

- 混合インスタンスグループはサポートされていません。混合インスタンスポリシーが適用されている Auto Scaling グループに VPC Lattice ターゲットグループをアタッチしようとすると、エラーメッセージ「現在、混合インスタンスを含む Auto Scaling グループは VPC Lattice サービスと統合できません」が表示されます。これは、負荷分散アルゴリズムにより、利用可能なすべてのリソースに負荷が均等に分散されたことで、インスタンス同士が同じ負荷を処理できるほどに類似していると想定されたためです。

VPC Lattice ターゲットグループを Auto Scaling グループにアタッチする

このトピックでは、VPC Lattice ターゲットグループを Auto Scaling グループにアタッチする方法について説明します。また、VPC Lattice ヘルスチェックを有効にして、VPC Lattice から異常として報告されたインスタンスを Amazon EC2 Auto Scaling で置き換える方法についても説明します。

デフォルトでは、Amazon EC2 Auto Scaling は、Amazon EC2 ヘルスチェックに基づいて、異常なインスタンスまたはアクセスできないインスタンスのみを置き換えます。VPC Lattice ヘルスチェックを有効にすると、Auto Scaling グループにアタッチした VPC Lattice ターゲットグループのいずれかから、実行中のインスタンスが異常として報告された場合、Amazon EC2 Auto Scaling は、そのインスタンスを置き換えることができます。詳細については、「Auto Scaling グループ内のインスタンスのヘルスチェック」を参照してください。

> **Important**
>
> 先に進む前に、前のセクションのすべての前提条件を満たしてください。

VPC Lattice ターゲットグループをアタッチする

グループを作成または更新するときに、1 つ以上のターゲットグループを Auto Scaling グループにアタッチできます。

Console

このセクションの手順に従い、コンソールを使用して次の操作を実行します。

- VPC Lattice ターゲットグループを Auto Scaling グループにアタッチする
- VPC Lattice のヘルスチェックを有効にする

VPC Lattice ターゲットグループを新しい Auto Scaling グループにアタッチするには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. 画面の上部のナビゲーションバーで、ターゲットグループを作成した AWS リージョン を選択します。
3. [Auto Scaling グループの作成] を選択します。
4. ステップ 1 と 2 で、希望するオプションを選択し、「ステップ 3: 詳細オプションを設定する」へ進みます。
5. [VPC Lattice 統合オプション] で、[VPC Lattice サービスへアタッチ] を選択します。
6. [VPC Lattice ターゲットグループを選択] で、ターゲットグループを選択します。
7. (オプション) [ヘルスチェック] の [追加のヘルスチェックタイプ] で、[VPC Lattice ヘルスチェックを有効にする] を選択します。
8. (オプション) [ヘルスチェックの猶予期間] に秒単位で時間を入力します。これは、インスタンスが `InService` 状態になった後で、Amazon EC2 Auto Scaling がインスタンスのヘルスステータスチェックの実行を待つ必要がある時間です。詳細については、「Auto Scaling グループにヘルスチェックの猶予期間を設定する」を参照してください。
9. Auto Scaling グループの作成に進みます。Auto Scaling グループの作成後、インスタンスは自動的に VPC Lattice ターゲットグループに登録されます。

VPC Lattice ターゲットグループを既存の Auto Scaling グループにアタッチするには

次の手順に従って、ターゲットグループを既存の Auto Scaling グループにアタッチします。

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. Auto Scaling グループの横にあるチェックボックスを選択します。

   ページの下部にスプリットペインが開きます。
3. [詳細] タブで、[VPC Lattice 統合オプション]、[編集] を順に選択します。
4. [VPC Lattice 統合オプション] で、[VPC Lattice サービスへアタッチ] を選択します。
5. [VPC Lattice ターゲットグループを選択] で、ターゲットグループを選択します。
6. [更新] を選択します。

ターゲットグループのアタッチが完了したら、必要に応じて、そのターゲットグループで使用するヘルスチェックを有効にできます。

VPC Lattice ヘルスチェックを有効にするには

1. [詳細] タブで、[ヘルスチェック]、[編集] の順に選択します。
2. [ヘルスチェック] の [追加のヘルスチェックタイプ] で、[VPC Lattice ヘルスチェックを有効にする] を選択します。
3. [ヘルスチェックの猶予期間] に、秒単位で時間を入力します。これは、インスタンスが `InService` 状態になった後で、Amazon EC2 Auto Scaling がインスタンスのヘルスステータスチェックの実行を待つ必要がある時間です。詳細については、「Auto Scaling グループにヘルスチェックの猶予期間を設定する」を参照してください。
4. [更新] を選択します。

AWS CLI

このセクションのステップに従って、 を使用して以下 AWS CLI を行います。

- VPC Lattice ターゲットグループを Auto Scaling グループにアタッチする
- VPC Lattice のヘルスチェックを有効にする

VPC Lattice ターゲットグループを Auto Scaling グループにアタッチするには

次の create-auto-scaling-group コマンドを使用して Auto Scaling グループを作成し、同時に、そのグループの Amazon リソースネーム (ARN) を指定して VPC Lattice ターゲットグループをアタッチします。

--auto-scaling-group-name、--vpc-zone-identifier、--min-size、および --max-size のサンプルの値を置き換えます。--launch-template オプションの場合、*my-launch-template* と *1* を、VPC Lattice ターゲットグループに登録したインスタンス用に作成した起動テンプレートの名前とバージョンに置き換えます。--traffic-sources オプションの場合、サンプル ARN を VPC Lattice ターゲットグループの ARN に置き換えます。

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-asg \
  --launch-template LaunchTemplateName=my-launch-template,Version='1' \
  --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782" \
  --min-size 1 --max-size 5 \
  --traffic-sources "Identifier=arn:aws:vpc-lattice:region:account-id:targetgroup/
tg-0e2f2665eEXAMPLE"
```

VPC Lattice ターゲットグループを既に作成していたら、次の attach-traffic-sources コマンドを使用して、そのターゲットグループを Auto Scaling グループにアタッチします。

```
aws autoscaling attach-traffic-sources --auto-scaling-group-name my-asg \
  --traffic-sources "Identifier=arn:aws:vpc-lattice:region:account-id:targetgroup/
tg-0e2f2665eEXAMPLE"
```

VPC Lattice のヘルスチェックを有効にするには

VPC Lattice ターゲットグループにアプリケーションベースのヘルスチェックを設定している場合は、これらのヘルスチェックを有効にできます。--health-check-type オプションと **VPC_LATTICE** の値を指定した create-auto-scaling-group または update-auto-scaling-group コマンドを使用します。Auto Scaling グループで実行するヘルスチェックの猶予期間を指定する場合は、--health-check-grace-period オプションを含め、その値を秒単位で指定します。

```
--health-check-type "VPC_LATTICE" --health-check-grace-period 60
```

VPC Lattice ターゲットグループをデタッチする

VPC Lattice が不要になった場合、次の手順に従って、Auto Scaling グループからターゲットグループをデタッチします。

Console

このセクションの手順に従い、コンソールを使用して次の操作を実行します。

- Auto Scaling グループから VPC Lattice ターゲットグループをデタッチする
- VPC Lattice のヘルスチェックを無効にする

Auto Scaling グループから VPC Lattice ターゲットグループをデタッチするには

1. https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Auto Scaling グループ] を選択します。
2. 既存のグループの横にあるチェックボックスをオンにします。

   ページの下部にスプリットペインが開きます。
3. [詳細] タブで、[VPC Lattice 統合オプション]、[編集] を順に選択します。
4. [VPC Lattice 統合オプション] で、ターゲットグループの横にある [削除] アイコン (X) を選択します。
5. [更新] を選択します。

ターゲットグループのデタッチが完了したら、VPC Lattice ヘルスチェックを無効にできます。

VPC Lattice ヘルスチェックを無効にするには

1. [詳細] タブで、[ヘルスチェック]、[編集] の順に選択します。
2. [ヘルスチェック] の [追加のヘルスチェックタイプ] で、[VPC Lattice ヘルスチェックを有効にする] の選択を解除します。
3. [更新] を選択します。

AWS CLI

このセクションのステップに従って、 を使用して以下 AWS CLI を行います。

- Auto Scaling グループから VPC Lattice ターゲットグループをデタッチする
- VPC Lattice のヘルスチェックを無効にする

ターゲットグループが不要になったら、detach-traffic-sources コマンドを使用して、Auto Scaling グループからそのターゲットグループをデタッチします。

```
aws autoscaling detach-traffic-sources --auto-scaling-group-name my-asg \
  --traffic-sources "Identifier=arn:aws:vpc-lattice:region:account-id:targetgroup/
tg-0e2f2665eEXAMPLE"
```

Auto Scaling グループのヘルスチェックを更新して VPC Lattice ヘルスチェックを使用しないようにするには、update-auto-scaling-group コマンドを使用します。`--health-check-type` オプションと **EC2** の値を含めます。

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --health-check-type "EC2"
```

VPC Lattice ターゲットグループのアタッチメントステータスを確認する

VPC Lattice ターゲットグループを Auto Scaling グループにアタッチすると、グループ内のインスタンスの登録中、`Adding` 状態になります。グループのすべてのインスタンスが登録済みになると、`Added` 状態になります。最低 1 つの登録されたインスタンスがヘルスチェックを通過した後、`InService` 状態になります。ターゲットグループが `InService` 状態にある場合、Amazon EC2 Auto Scaling は異常として報告されたインスタンスを終了し、置き換えることができます。登録したインスタンスがどれもヘルスチェックを通過しなかった場合 (ヘルスチェックの設定ミスなどによる)、ターゲットグループは `InService` 状態になりません。Amazon EC2 Auto Scaling はインスタンスを終了し置き換えをすることはありません。

サービスのターゲットグループをデタッチすると、グループ内のインスタンスの登録解除中、`Removing` 状態になります。登録解除してもインスタンスは引き続き実行されます。デフォルトでは、Connection Draining (登録解除の遅延) が有効になっています。Connection Draining が有効になっている場合、VPC Lattice は、処理中のリクエストが完了するまで、または最大タイムアウト時間が終了するまで (どちらか早い方) 待機してから、インスタンスの登録を解除します。

アタッチメントのステータスは、 AWS Command Line Interface （ AWS CLI) または AWS SDKsを使用して確認できます。アタッチメントステータスはコンソールから確認できません。

を使用してアタッチメントのステータス AWS CLI を確認するには

次の describe-traffic-sources コマンドは、指定した Auto Scaling グループのすべてのトラフィックソースのアタッチメントステータスを返します。

```
aws autoscaling describe-traffic-sources --auto-scaling-group-name my-asg
```

この例では、Auto Scaling グループにアタッチされている VPC Lattice ターゲットグループの ARN と、State 要素内のターゲットグループのアタッチメントステータスを返します。

```
{
    "TrafficSources": [
        {
            "Identifier": "arn:aws:vpc-lattice:region:account-
id:targetgroup/tg-0e2f2665eEXAMPLE",
            "State": "InService",
            "Type": "vpc-lattice"
        }
    ]
}
```

EventBridge を使用して Auto Scaling イベントを処理する

以前は CloudWatch Events と呼ばれていた Amazon は、 リソースをモニタリングし EventBridge、他の AWS のサービスを使用するターゲットアクションを開始するイベント駆動型ルールを設定するのに役立ちます。

Amazon EC2 Auto Scaling からのイベントは、ほぼリアルタイムで EventBridge に配信されます。さまざまなイベントに応じてプログラムによるアクションや通知を呼び出す EventBridge ルールを設定できます。例えば、インスタンスの起動中または終了中に、 AWS Lambda 関数を呼び出して事前設定されたタスクを実行できます。

EventBridge ルールのターゲットには、 AWS Lambda 関数、Amazon SNS トピック、API 送信先、他の のイベントバス AWS アカウントなどが含まれます。サポートされているターゲットの詳細については、「Amazon ユーザーガイド」の「Amazon EventBridge ターゲット」を参照してください。 EventBridge

Amazon SNS トピックと EventBridge ルールを使用して、例を使用して EventBridge ルールを作成します。その後、ユーザーがインスタンスの更新を開始すると、チェックポイントに到達するたびに Amazon SNS が E メールで通知します。詳細については、「インスタンス更新イベントの EventBridge ルールを作成する」を参照してください。

内容

- Amazon EC2 Auto Scaling イベントリファレンス

- ウォームプールのイベントとパターンの例
- EventBridge ルールの作成

Amazon EC2 Auto Scaling イベントリファレンス

Amazon を使用すると EventBridge、受信イベントを照合し、処理のためにターゲットにルーティングするルールを作成できます。

内容

- ライフサイクルアクションイベント
- 成功したスケーリングイベント
- 失敗したスケーリングイベント
- インスタンス更新イベント

ライフサイクルアクションイベント

Auto Scaling グループにライフサイクルフックを追加すると、インスタンスが待機状態に移行すると EventBridge、Amazon EC2 Auto Scaling はイベントを に送信します。イベントは、ベストエフォートベースで生成されます。

イベントタイプ

- スケールアウトライフサイクルアクション
- スケールインライフサイクルアクション

スケールアウトライフサイクルアクション

次のイベント例は、起動ライフサイクルフックを理由として、Amazon EC2 Auto Scaling がインスタンスを `Pending:Wait` 状態に移行したことを示しています。

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Instance-launch Lifecycle Action",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
```

```
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "LifecycleActionToken": "87654321-4321-4321-4321-210987654321",
    "AutoScalingGroupName": "my-asg",
    "LifecycleHookName": "my-lifecycle-hook",
    "EC2InstanceId": "i-1234567890abcdef0",
    "LifecycleTransition": "autoscaling:EC2_INSTANCE_LAUNCHING",
    "NotificationMetadata": "additional-info",
    "Origin": "EC2",
    "Destination": "AutoScalingGroup"
  }
}
```

スケールインライフサイクルアクション

次のイベント例は、終了ライフサイクルフックを理由として、Amazon EC2 Auto Scaling がインスタンスを `Terminating:Wait` 状態に移行したことを示しています。

> **Important**
>
> Auto Scaling グループがスケールイン時にインスタンスをウォームプールに返す場合、インスタンスをウォームプールに返すことによって `EC2 Instance-terminate Lifecycle Action` イベントが生成される場合もあります。インスタンスがスケールインで待機状態に移行したときに配信されるイベントには、`Destination` の値として `WarmPool` が含まれます。詳細については、「Instance reuse policy」を参照してください。

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Instance-terminate Lifecycle Action",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
```

```
    "LifecycleActionToken":"87654321-4321-4321-4321-210987654321",
    "AutoScalingGroupName":"my-asg",
    "LifecycleHookName":"my-lifecycle-hook",
    "EC2InstanceId":"i-1234567890abcdef0",
    "LifecycleTransition":"autoscaling:EC2_INSTANCE_TERMINATING",
    "NotificationMetadata":"additional-info",
    "Origin": "AutoScalingGroup",
    "Destination": "EC2"
  }
}
```

成功したスケーリングイベント

次の例は、成功したスケーリングイベントのイベントタイプを示しています。イベントは、ベストエフォートベースで生成されます。

イベントタイプ

- 成功したスケールアウトイベント
- 成功したスケールインイベント

成功したスケールアウトイベント

次のイベント例は、Amazon EC2 Auto Scaling がインスタンスを正常に起動したことを示しています。

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Instance Launch Successful",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn",
    "instance-arn"
  ],
  "detail": {
    "StatusCode": "InProgress",
    "Description": "Launching a new EC2 instance: i-12345678",
    "AutoScalingGroupName": "my-asg",
```

```
    "ActivityId": "87654321-4321-4321-4321-210987654321",
    "Details": {
      "Availability Zone": "us-west-2b",
      "Subnet ID": "subnet-12345678"
    },
    "RequestId": "12345678-1234-1234-1234-123456789012",
    "StatusMessage": "",
    "EndTime": "yyyy-mm-ddThh:mm:ssZ",
    "EC2InstanceId": "i-1234567890abcdef0",
    "StartTime": "yyyy-mm-ddThh:mm:ssZ",
    "Cause": "description-text",
    "Origin": "EC2",
    "Destination": "AutoScalingGroup"
  }
}
```

成功したスケールインイベント

次のイベント例は、Amazon EC2 Auto Scaling がインスタンスを正常に終了したことを示しています。

> **Important**
>
> Auto Scaling グループがスケールイン時にインスタンスをウォームプールに返す場合、インスタンスをウォームプールに返すことによって EC2 Instance Terminate Successful イベントが生成される場合もあります。インスタンスがウォームプールに正常に戻ったときに配信されるイベントには、Destination の値として WarmPool が含まれます。詳細については、「Instance reuse policy」を参照してください。

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Instance Terminate Successful",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn",
    "instance-arn"
```

```
  ],
  "detail": {
    "StatusCode": "InProgress",
    "Description": "Terminating EC2 instance: i-12345678",
    "AutoScalingGroupName": "my-asg",
    "ActivityId": "87654321-4321-4321-4321-210987654321",
    "Details": {
      "Availability Zone": "us-west-2b",
      "Subnet ID": "subnet-12345678"
    },
    "RequestId": "12345678-1234-1234-1234-123456789012",
    "StatusMessage": "",
    "EndTime": "yyyy-mm-ddThh:mm:ssZ",
    "EC2InstanceId": "i-1234567890abcdef0",
    "StartTime": "yyyy-mm-ddThh:mm:ssZ",
    "Cause": "description-text",
    "Origin": "AutoScalingGroup",
    "Destination": "EC2"
  }
}
```

失敗したスケーリングイベント

次の例は、失敗したスケーリングイベントのイベントタイプを示しています。イベントは、ベストエフォートベースで生成されます。

イベントタイプ

- 失敗したスケールアウトイベント
- 失敗したスケールインイベント

失敗したスケールアウトイベント

次のイベント例は、Amazon EC2 Auto Scaling がインスタンスの起動に失敗したことを示しています。

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Instance Launch Unsuccessful",
  "source": "aws.autoscaling",
  "account": "123456789012",
```

```
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn",
    "instance-arn"
  ],
  "detail": {
    "StatusCode": "Failed",
    "AutoScalingGroupName": "my-asg",
    "ActivityId": "87654321-4321-4321-4321-210987654321",
    "Details": {
      "Availability Zone": "us-west-2b",
      "Subnet ID": "subnet-12345678"
    },
    "RequestId": "12345678-1234-1234-1234-123456789012",
    "StatusMessage": "message-text",
    "EndTime": "yyyy-mm-ddThh:mm:ssZ",
    "EC2InstanceId": "i-1234567890abcdef0",
    "StartTime": "yyyy-mm-ddThh:mm:ssZ",
    "Cause": "description-text",
    "Origin": "EC2",
    "Destination": "AutoScalingGroup"
  }
}
```

失敗したスケールインイベント

次のイベント例は、Amazon EC2 Auto Scaling がインスタンスの終了に失敗したことを示しています。

> **Important**
>
> Auto Scaling グループがスケールイン時にインスタンスをウォームプールに返すときに、インスタンスをウォームプールに返すことに失敗すると、EC2 Instance Terminate Unsuccessful イベントが生成される場合もあります。インスタンスがウォームプールに戻るのに失敗した場合に配信されるイベントには、Destination の値として WarmPool が含まれます。詳細については、「Instance reuse policy」を参照してください。

```
{
  "version": "0",
```

```
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Instance Terminate Unsuccessful",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn",
    "instance-arn"
  ],
  "detail": {
    "StatusCode": "Failed",
    "AutoScalingGroupName": "my-asg",
    "ActivityId": "87654321-4321-4321-4321-210987654321",
    "Details": {
      "Availability Zone": "us-west-2b",
      "Subnet ID": "subnet-12345678"
    },
    "RequestId": "12345678-1234-1234-1234-123456789012",
    "StatusMessage": "message-text",
    "EndTime": "yyyy-mm-ddThh:mm:ssZ",
    "EC2InstanceId": "i-1234567890abcdef0",
    "StartTime": "yyyy-mm-ddThh:mm:ssZ",
    "Cause": "description-text",
    "Origin": "AutoScalingGroup",
    "Destination": "EC2"
  }
}
```

インスタンス更新イベント

次の例は、インスタンス更新機能のイベントを示しています。イベントは、ベストエフォートベースで生成されます。

イベントタイプ

- チェックポイントに達しました
- インスタンスの更新が開始されました
- インスタンスの更新が成功しました
- インスタンスの更新に失敗しました
- インスタンスの更新がキャンセルされました
- インスタンスの更新ロールバックが開始されました

- インスタンスの更新のロールバックに成功しました
- インスタンスの更新のロールバックに失敗しました

チェックポイントに達しました

置き換えられたインスタンスの数が、チェックポイントに定義された割合のしきい値に達すると、Amazon EC2 Auto Scaling は次のイベントを送信します。

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Auto Scaling Instance Refresh Checkpoint Reached",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "InstanceRefreshId": "ab00cf8f-9126-4f3c-8010-dbb8cad6fb86",
    "AutoScalingGroupName": "my-asg",
    "CheckpointPercentage": "50",
    "CheckpointDelay": "300"
  }
}
```

インスタンスの更新が開始されました

インスタンスの更新のステータスが `InProgress` に変わると、Amazon EC2 Auto Scaling は次のイベントを送信します。

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Auto Scaling Instance Refresh Started",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
```

```
  ],
  "detail": {
    "InstanceRefreshId": "c613620e-07e2-4ed2-a9e2-ef8258911ade",
    "AutoScalingGroupName": "my-asg"
  }
}
```

インスタンスの更新が成功しました

インスタンスの更新のステータスが Successful に変わると、Amazon EC2 Auto Scaling は次のイベントを送信します。

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Auto Scaling Instance Refresh Succeeded",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "InstanceRefreshId": "c613620e-07e2-4ed2-a9e2-ef8258911ade",
    "AutoScalingGroupName": "my-asg"
  }
}
```

インスタンスの更新に失敗しました

インスタンスの更新のステータスが Failed に変わると、Amazon EC2 Auto Scaling は次のイベントを送信します。

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Auto Scaling Instance Refresh Failed",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
```

```
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "InstanceRefreshId": "c613620e-07e2-4ed2-a9e2-ef8258911ade",
    "AutoScalingGroupName": "my-asg"
  }
}
```

インスタンスの更新がキャンセルされました

インスタンスの更新のステータスが `Cancelled` に変わると、Amazon EC2 Auto Scaling は次のイベントを送信します。

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Auto Scaling Instance Refresh Cancelled",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "InstanceRefreshId": "c613620e-07e2-4ed2-a9e2-ef8258911ade",
    "AutoScalingGroupName": "my-asg"
  }
}
```

インスタンスの更新ロールバックが開始されました

インスタンスの更新のステータスが `RollbackInProgress` に変わると、Amazon EC2 Auto Scaling は次のイベントを送信します。

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Auto Scaling Instance Refresh Rollback Started",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
```

```
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "InstanceRefreshId": "c613620e-07e2-4ed2-a9e2-ef8258911ade",
    "AutoScalingGroupName": "my-asg"
  }
}
```

インスタンスの更新のロールバックに成功しました

インスタンスの更新のステータスが RollbackSuccessful に変わると、Amazon EC2 Auto Scaling は次のイベントを送信します。

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Auto Scaling Instance Refresh Rollback Succeeded",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "InstanceRefreshId": "c613620e-07e2-4ed2-a9e2-ef8258911ade",
    "AutoScalingGroupName": "my-asg"
  }
}
```

インスタンスの更新のロールバックに失敗しました

インスタンスの更新のステータスが Failed に変わると、Amazon EC2 Auto Scaling は次のイベントを送信します。

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Auto Scaling Instance Refresh Rollback Failed",
  "source": "aws.autoscaling",
  "account": "123456789012",
```

```
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "InstanceRefreshId": "c613620e-07e2-4ed2-a9e2-ef8258911ade",
    "AutoScalingGroupName": "my-asg"
  }
}
```

ウォームプールのイベントとパターンの例

Amazon EC2 Auto Scaling は、Amazon でいくつかの事前定義されたパターンをサポートしています EventBridge。これは、イベントパターンが作成される方法を簡素化します。フォームでフィールド値を選択すると、 によってパターン EventBridge が生成されます。現在、Amazon EC2 Auto Scaling は、ウォームプールがある Auto Scaling グループによって発行されるイベントに対して事前定義されたパターンをサポートしていません。パターンは、JSON オブジェクトとして入力する必要があります。このセクションと「ウォームプールイベントの EventBridge ルールを作成する」トピックでは、イベントパターンを使用してイベントを選択し、それらをターゲットに送信する方法を説明します。

Amazon EC2 Auto Scaling が に送信するウォームプール関連のイベントをフィルタリングする EventBridge ルールを作成するには EventBridge、イベントの `detail`セクションの `Origin`および `Destination`フィールドを含めます。

`Origin`および`Destination` の値には以下を指定できます。

`EC2` | `AutoScalingGroup` | `WarmPool`

内容

- イベントの例
- イベントパターンの例

イベントの例

Auto Scaling グループにライフサイクルフックを追加すると、インスタンスが待機状態に移行すると EventBridge、Amazon EC2 Auto Scaling はイベントを に送信します。詳細については、「ウォームプールでライフサイクルフックを使用する」を参照してください。

このセクションには、Auto Scaling グループにウォームプールがある場合のこれらのイベントの例が含まれています。イベントは、ベストエフォートベースで出力されます。

> **Note**
>
> スケーリングが成功 EventBridge したときに Amazon EC2 Auto Scaling が に送信するイベントについては、「」を参照してください成功したスケーリングイベント。スケーリングが失敗した場合のイベントについては、「失敗したスケーリングイベント」を参照してください。

イベント例

- スケールアウトライフサイクルアクション
- スケールインライフサイクルアクション

スケールアウトライフサイクルアクション

インスタンスがスケールアウトイベントの待機状態に移行するときに配信されるイベントには、`detail-type` の値として `EC2 Instance-launch Lifecycle Action` が含まれます。`detail` オブジェクト内の `Origin` および `Destination` 属性の値は、インスタンスがどこから来てどこへ行くのかを示します。

このスケールアウトイベントの例では、新しいインスタンスが起動し、ウォームプールに追加されるため、その状態が `Warmed:Pending:Wait` に変わります。詳細については、「ウォームプール内のインスタンスのライフサイクル状態の移行」を参照してください。

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Instance-launch Lifecycle Action",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "2021-01-13T00:12:37.214Z",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "LifecycleActionToken": "71514b9d-6a40-4b26-8523-05e7eEXAMPLE",
    "AutoScalingGroupName": "my-asg",
```

```
    "LifecycleHookName": "my-launch-lifecycle-hook",
    "EC2InstanceId": "i-1234567890abcdef0",
    "LifecycleTransition": "autoscaling:EC2_INSTANCE_LAUNCHING",
    "NotificationMetadata": "additional-info",
    "Origin": "EC2",
    "Destination": "WarmPool"
  }
}
```

このスケールアウトイベントの例では、インスタンスがウォームプールから Auto Scaling グループに追加されるため、インスタンスの状態は Pending:Wait に変わります。詳細については、「ウォームプール内のインスタンスのライフサイクル状態の移行」を参照してください。

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Instance-launch Lifecycle Action",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "2021-01-19T00:35:52.359Z",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "LifecycleActionToken": "19cc4d4a-e450-4d1c-b448-0de67EXAMPLE",
    "AutoScalingGroupName": "my-asg",
    "LifecycleHookName": "my-launch-lifecycle-hook",
    "EC2InstanceId": "i-1234567890abcdef0",
    "LifecycleTransition": "autoscaling:EC2_INSTANCE_LAUNCHING",
    "NotificationMetadata": "additional-info",
    "Origin": "WarmPool",
    "Destination": "AutoScalingGroup"
  }
}
```

スケールインライフサイクルアクション

インスタンスがスケールインイベントの待機状態に移行するときに配信されるイベントには、detail-type の値として EC2 Instance-terminate Lifecycle Action が含まれます。detail オブジェクト内の Origin および Destination 属性の値は、インスタンスがどこから来てどこへ行くのかを示します。

このスケールインイベントの例では、インスタンスがウォームプールに戻されるため、インスタンスの状態は Warmed:Pending:Wait に変わります。詳細については、「ウォームプール内のインスタンスのライフサイクル状態の移行」を参照してください。

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Instance-terminate Lifecycle Action",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "2022-03-28T00:12:37.214Z",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "LifecycleActionToken": "42694b3d-4b70-6a62-8523-09a1eEXAMPLE",
    "AutoScalingGroupName": "my-asg",
    "LifecycleHookName": "my-termination-lifecycle-hook",
    "EC2InstanceId": "i-1234567890abcdef0",
    "LifecycleTransition": "autoscaling:EC2_INSTANCE_TERMINATING",
    "NotificationMetadata": "additional-info",
    "Origin": "AutoScalingGroup",
    "Destination": "WarmPool"
  }
}
```

イベントパターンの例

前のセクションでは、Amazon EC2 Auto Scaling によって生成されるイベント例を説明します。

EventBridge イベントパターンは、一致するイベントと同じ構造です。イベントのパターンでは、照合する対象のフィールドを引用符で囲み、検出したい値を指定します。

イベント内の次のフィールドは、ルールで定義され、アクションをトリガーするイベントパターンを形成します。

`"source": "aws.autoscaling"`

イベントが Amazon EC2 Auto Scaling Group からのものであることを特定します。

`"detail-type": "EC2 Instance-launch Lifecycle Action"`

イベントタイプを特定します。

`"Origin": "EC2"`

インスタンスがどこから来ているかを識別します。

`"Destination": "WarmPool"`

インスタンスがどこに行くかを識別します。

次のサンプルイベントパターンを使用して、ウォームプールに入るインスタンスに関連付けられているすべての EC2 Instance-launch Lifecycle Action イベントをキャプチャします。

```
{
  "source": [ "aws.autoscaling" ],
  "detail-type": [ "EC2 Instance-launch Lifecycle Action" ],
  "detail": {
      "Origin": [ "EC2" ],
      "Destination": [ "WarmPool" ]
   }
}
```

次のサンプルイベントパターンを使用して、スケールアウトイベントのためにウォームプールから取り出されるインスタンスに関連付けられたすべての EC2 Instance-launch Lifecycle Action イベントをキャプチャします。

```
{
  "source": [ "aws.autoscaling" ],
  "detail-type": [ "EC2 Instance-launch Lifecycle Action" ],
  "detail": {
      "Origin": [ "WarmPool" ],
      "Destination": [ "AutoScalingGroup" ]
   }
}
```

次のサンプルイベントパターンを使用して、Auto Scaling グループに直接起動するインスタンスに関連付けられているすべての EC2 Instance-launch Lifecycle Action イベントをキャプチャします。

```
{
  "source": [ "aws.autoscaling" ],
  "detail-type": [ "EC2 Instance-launch Lifecycle Action" ],
  "detail": {
```

```
      "Origin": [ "EC2" ],
      "Destination": [ "AutoScalingGroup" ]
   }
}
```

次のサンプルイベントパターンを使用して、スケールインでウォームプールに戻されるインスタンスに関連付けられているすべての EC2 Instance-terminate Lifecycle Action イベントをキャプチャします。

```
{
  "source": [ "aws.autoscaling" ],
  "detail-type": [ "EC2 Instance-terminate Lifecycle Action" ],
  "detail": {
      "Origin": [ "AutoScalingGroup" ],
      "Destination": [ "WarmPool" ]
   }
}
```

次のイベントパターンのサンプルを使用して、送信元や送信先に関わらず、EC2 Instance-launch Lifecycle Actionに関連付けられているすべてのイベントをキャプチャします。

```
{
  "source": [ "aws.autoscaling" ],
  "detail-type": [ "EC2 Instance-launch Lifecycle Action" ]
}
```

EventBridge ルールの作成

イベントが Amazon EC2 Auto Scaling によって発行されると、イベント通知が JSON ファイル EventBridge として Amazon に送信されます。イベントパターンが EventBridge ルールに一致するときに実行するアクションを自動化する ルールを作成できます。がルールで定義されたパターンに一致するイベントパターン EventBridge を検出すると、 はルールで指定されたターゲット (またはターゲット) を EventBridge 呼び出します。

このセクションの手順例を開始点として使用できます。

以下のドキュメントも役に立つかもしれません。

- Lambda 関数を使用して、インスタンスの起動中、またはインスタンスの終了前にそれらでカスタムアクションを実行するには、「チュートリアル:Lambda 関数を呼び出すライフサイクルフックの設定」を参照してください。

- で ログ記録された API コールで Lambda 関数を呼び出すには CloudTrail、「Amazon ユーザーガイド」の「チュートリアル: を使用した AWS API コールのログ EventBridge記録」を参照してください。 EventBridge
- イベントルールの作成方法の詳細については、「Amazon ユーザーガイド」の「イベントに対応する Amazon EventBridge ルールの作成」を参照してください。 EventBridge

トピック

- インスタンス更新イベントの EventBridge ルールを作成する
- ウォームプールイベントの EventBridge ルールを作成する

インスタンス更新イベントの EventBridge ルールを作成する

次の例では、E メール通知を送信する EventBridge ルールを作成します。これは、インスタンス更新中、チェックポイントに到達した時に Auto Scaling グループがイベントを発行するたびに実行されます。Amazon SNS を使用して E メール通知を設定する手順も含まれています。Amazon SNS を使用して E メール通知を送信するには、最初にトピックを作成してから、そのトピックと共に E メールアドレスを登録する必要があります。

インスタンスの更新機能に関する詳細については、「インスタンスの更新を使用して Auto Scaling グループのインスタンスを更新する」を参照してください。

Amazon SNS トピックを作成します。

SNS トピックは論理アクセスポイント、つまり Auto Scaling グループ が通知を送信するために使用する通信チャネルです。トピックの名前を指定することにより、トピックを作成します。

トピック名は、以下の要件を満たしている必要があります。

- 1 ~ 256 文字
- 大文字および小文字の ASCII 文字、数字、アンダースコア、またはハイフンが含まれている

詳細については、Amazon Simple 通知サービス デベロッパーガイドの「Amazon SNS トピックの作成」を参照してください。

Amazon SNS トピックを購読します。

Auto Scaling グループがトピックに送信した通知を受信するには、そのトピックにエンドポイントを登録する必要があります。この手順では、エンドポイント に、Amazon EC2 Auto Scaling からの通知を受信する E メールアドレスを指定します。

詳細については、Amazon Simple 通知サービス デベロッパーガイドの「Amazon SNS トピックへのサブスクライブ」を参照してください。

Amazon SNS サブスクリプションを確認する

Amazon SNS は、前のステップで指定した E メールアドレスに確認メールを送信します。

次のステップに進む前に、 AWS 通知から E メールを開き、リンクを選択してサブスクリプションを確認します。

から確認メッセージが表示されます AWS。Amazon SNS は、通知を受信し、通知を E メールとして指定された E メールアドレスに送信するように設定されました。

Amazon SNS トピックにイベントをルートする

選択したイベントに一致するルールを作成し、それらを Amazon SNS トピックにルーティングして、購読済みの E メールアドレスに通知します。

Amazon SNS トピックに通知を送信するルールを作成する

1. https://console.aws.amazon.com/events/ で Amazon EventBridge コンソールを開きます。
2. ナビゲーションペインで Rules] (ルール) を選択します。
3. ルールの作成 を選択します。
4. [Define rule detail] (詳細の定義) で、次の操作を行います。
   a. ルールの [Name (名前)] を入力し、必要に応じて説明を入力します。

      ルールには、同じリージョン内および同じイベントバス上の別のルールと同じ名前を付けることはできません。
   b. [イベントバス] として、[デフォルト] を選択します。アカウントの AWS サービスがイベントを生成すると、常にアカウントのデフォルトのイベントバスに送られます。
   c. ルールタイプ では、イベントパターンを持つルール] を選択します。
   d. 次へ をクリックします。

5. [Build event pattern] (イベントパターンの作成) で、次の操作を行います。

   a. イベントソース で、AWS イベント または EventBridge パートナーイベント を選択します。

   b. [Event pattern] (イベントパターン) の場合は、次のいずれかを実行します。

      i. イベントソース で AWS のサービス を選択します。

      ii. [AWS のサービス] には、[Auto Scaling] を選択します。

      iii. [Event type (イベントタイプ)] で、[Instance Refresh (インスタンス更新)] を選択します。

      iv. デフォルトでは、ルールは更新イベントのすべてのインスタンスに一致します。インスタンスの更新中、チェックポイントに到達したときに通知するルールを作成するには、[Specific instance event(s)] (特定のインスタンスイベント) を選択し、[EC2 Auto Scaling Instance Refresh Checkpoint Reached] (EC2 Auto Scaling インスタンス更新チェックポイントに到達) を選択します。

      v. デフォルトでは、このルールはリージョン内のすべての Auto Scaling グループと一致します。ルールを特定の Auto Scaling グループに一致させるには、[Specific group name (特定のグループ名)] を選択して 1 つ以上の Auto Scaling グループを選択します。

      vi. [次へ] をクリックします。

6. [Select target(s)] (ターゲットを選択) で、以下の操作を行います。

   a. [Target types] (ターゲットタイプ) には、[AWS のサービス] を選択します。

   b. [Select a target] (ターゲットの選択) には、[SNS topic] (SNS トピック) を選択します。

   c. [Topic] (トピック) には、お使いの Amazon SNS トピックを選択します。

   d. (オプション) [Additional settings] (追加設定) で、その他の設定を行うこともできます。詳細については、「Amazon ユーザーガイド」の「イベントに反応する Amazon EventBridge ルールの作成」を参照してください。 EventBridge

   e. [次へ] をクリックします。

7. (オプション) 必要な場合は、[Tags] (タグ) で 1 つ以上のタグを作成したルールに割り当て、[Next] (次へ) を選択します。

8. [Review and create] (レビューと作成) で、ルールの詳細を確認し、必要に応じて変更します。その後、[Create rule] (ルールの作成) を選択します。

ウォームプールイベントの EventBridge ルールを作成する

次の例では、プログラムによるアクションを呼び出す EventBridge ルールを作成します。これは、ウォームプールへの新しいインスタンスの追加時に Auto Scaling グループがイベントを発行するたびに実行されます。

ルールを作成する前に、ルールをターゲットとして使用する AWS Lambda 関数を作成します。この関数をルールのターゲットとして指定する必要があります。次の手順では、新しいインスタンスがウォームプールに入ったときに機能する EventBridge ルールを作成する手順のみを示します。着信イベントがルールに一致するときに呼び出すシンプルな Lambda 関数の作成方法を説明する初歩的なチュートリアルについては、「チュートリアル:Lambda 関数を呼び出すライフサイクルフックの設定」を参照してください。

ウォームプールの作成と使用に関する詳細については、「Amazon EC2 Auto Scaling のウォームプール」を参照してください。

Lambda 関数を呼び出すイベントルールを作成する

1. https://console.aws.amazon.com/events/ で Amazon EventBridge コンソールを開きます。
2. ナビゲーションペインで Rules] (ルール) を選択します。
3. ルールの作成 を選択します。
4. [Define rule detail] (詳細の定義) で、次の操作を行います。
   a. ルールの [Name (名前)] を入力し、必要に応じて説明を入力します。

      ルールには、同じリージョン内および同じイベントバス上の別のルールと同じ名前を付けることはできません。
   b. [イベントバス] として、[デフォルト] を選択します。 AWS のサービス アカウントの がイベントを生成すると、常にアカウントのデフォルトのイベントバスに送られます。
   c. ルールタイプ では、イベントパターンを持つルール] を選択します。
   d. 次へ をクリックします。
5. [Build event pattern] (イベントパターンの作成) で、次の操作を行います。
   a. イベントソース で、AWS イベント または EventBridge パートナーイベント を選択します。
   b. [Event pattern] では、[Custom pattern (JSON editor)] (カスタムパターン (JSON エディタ)) を選択し、以下のパターンを [Event pattern] (イベントパターン) ボックスに貼り付けて、#####のテキストを Auto Scaling グループの名前に置き換えます。

```
{
  "source": [ "aws.autoscaling" ],
  "detail-type": [ "EC2 Instance-launch Lifecycle Action" ],
  "detail": {
      "AutoScalingGroupName": [ "my-asg" ],
      "Origin": [ "EC2" ],
      "Destination": [ "WarmPool" ]
   }
}
```

他のイベントに一致するルールを作成するには、イベントパターンを変更します。詳細については、「イベントパターンの例」を参照してください。

c. [Next] (次へ) を選択します。

6. [Select target(s)] (ターゲットを選択) で、以下の操作を行います。

   a. [Target types] (ターゲットタイプ) には、[AWS のサービス] を選択します。

   b. [Select a target] (ターゲットを選択) では、[Lambda function] (Lambda 関数) を選択します。

   c. [Function] (機能) には、イベントの送信先となる関数を選択します。

   d. (オプション) [Configure version/alias] (バージョン/エイリアスを設定) で、ターゲットの Lambda 関数のバージョンとエイリアスの設定を入力します。

   e. (オプション)[Additional settings] (追加設定) で、アプリケーションに適切な追加設定を入力します。詳細については、「Amazon ユーザーガイド」の「イベントに反応する Amazon EventBridge ルールの作成」を参照してください。 EventBridge

   f. [次へ] をクリックします。

7. (オプション) 必要な場合は、[Tags] (タグ) で 1 つ以上のタグを作成したルールに割り当て、[Next] (次へ) を選択します。

8. [Review and create] (レビューと作成) で、ルールの詳細を確認し、必要に応じて変更します。その後、[Create rule] (ルールの作成) を選択します。

Amazon VPC を使用して Auto Scaling インスタンスにネットワーク接続を提供する

Amazon Virtual Private Cloud (Amazon VPC) は、定義した論理的に分離された仮想ネットワークで Auto Scaling グループなどの AWS リソースを起動できるサービスです。

Amazon VPC のサブネットとは、アベイラビリティーゾーンを VPC の IP アドレス範囲の一部で分割したものです。サブネットを使うと、インスタンスをセキュリティや運用上の必要に応じてグループ化することができます。サブネットは作成されたアベイラビリティーゾーン内に完全に包含されるものです。サブネットで Auto Scaling インスタンスを起動します。

サブネット内のインスタンスがインターネットと通信できるようにするには、インターネットゲートウェイを作成して、それを VPC にアタッチする必要があります。インターネットゲートウェイを使用すると、サブネット内のリソースが Amazon EC2 ネットワークエッジを介してインターネットに接続できるようになります。サブネットのトラフィックがインターネットゲートウェイにルーティングされる場合、そのサブネットはパブリックサブネットと呼ばれます。サブネットのトラフィックがインターネットゲートウェイにルーティングされない場合、そのサブネットはプライベートサブネットと呼ばれます。インターネットに接続する必要があるリソースにはパブリックサブネットを、インターネットに接続する必要がないリソースにはプライベートサブネットを使用してください。VPC 内のインスタンスへのインターネットアクセスを許可する方法の詳細については、「Amazon VPC ユーザーガイド」の「インターネットへのアクセス」を参照してください。

内容

- デフォルト VPC
- デフォルトではない VPC
- VPC サブネットを選択する場合の考慮事項
- VPC での IP アドレス指定
- VPC 内のネットワークインターフェイス
- インスタンスのプレイスメントテナンシー
- AWS Outposts
- VPC に関するその他のリソース

デフォルト VPC

2013 年 12 月 4 AWS アカウント 日以降に を作成した場合、または新しい で Auto Scaling グループを作成する場合は AWS リージョン、デフォルトの VPC が作成されます。各アベイラビリティーゾーンのデフォルトサブネットにはデフォルト VPC が用意されています。デフォルト VPC がある場合、デフォルトではそのデフォルト VPC で Auto Scaling グループが作成されます。

VPC は、Amazon VPC コンソールのお使いの VPC ページで表示できます。

デフォルト VPC の詳細については、「Amazon VPCs」の「デフォルト VPC」を参照してください。

デフォルトではない VPC

AWS Management Console の VPC ダッシュボードページにアクセスして、[Create VPC] (VPC を作成) を選択することで、追加の VPC を作成できます。

詳細については、Amazon VPC ユーザーガイドを参照してください。

> **Note**
>
> VPC は、その AWS リージョン内のすべてのアベイラビリティーゾーンを対象としています。VPC にサブネットを追加するときは、複数のアベイラビリティーゾーンを選択して、これらのサブネットでホストされているアプリケーションの高可用性を確保します。アベイラビリティーゾーンは、 AWS リージョンの冗長電源、ネットワーク、および接続を備えた 1 つ以上の個別のデータセンターです。アベイラビリティーゾーンは、本番環境アプリケーションの可用性、耐障害性、およびスケーラビリティを向上させるために役立ちます。

VPC サブネットを選択する場合の考慮事項

Auto Scaling グループの VPC サブネットを選択するときは、以下の点に留意してください。

- Elastic Load Balancing ロードバランサーを Auto Scaling グループにアタッチする場合、インスタンスはパブリックサブネットまたはプライベートサブネットで起動できます。ただし、DNS 解決をサポートするには、ロードバランサーをパブリックサブネットに作成する必要があります。
- SSH を介して Auto Scaling インスタンスに直接アクセスする場合、インスタンスはパブリックサブネットでのみ起動できます。

- AWS Systems Manager Session Manager を使用して no-ingress Auto Scaling インスタンスにアクセスする場合、インスタンスはパブリックサブネットまたはプライベートサブネットのいずれかで起動できます。
- プライベートサブネットを使用している場合は、パブリック NAT ゲートウェイを使用して Auto Scaling インスタンスがインターネットへのアクセスを許可します。
- デフォルトでは、デフォルト VPC のデフォルトサブネットはパブリックサブネットです。

VPC での IP アドレス指定

VPC で Auto Scaling インスタンスを起動すると、インスタンスには、インスタンスが起動されたサブネットの CIDR 範囲のプライベート IP アドレスが自動的に割り当てられます。これにより、インスタンスは VPC の他のインスタンスと通信できるようになります。

インスタンスにパブリック IPv4 アドレスを割り当てるように、起動設定または起動テンプレートを設定することができます。インスタンスにパブリック IP アドレスを割り当てると、インスタンスはインターネットや他の AWS サービスと通信できます。

IPv6 アドレスをインスタンスに自動的に割り当てるように設定されたサブネットでインスタンスを起動すると、インスタンスは IPv4 と IPv6 アドレスの両方を受け取ります。それ以外の場合は、IPv4 アドレスのみを受け取ります。詳細については、Amazon EC2 ユーザーガイドの IPv6 アドレスを参照してください。

VPC またはサブネットの CIDR 範囲の指定の詳細については、「Amazon VPC ユーザーガイド」を参照してください。

追加のネットワークインターフェイスを指定する起動テンプレートを使用すると、Amazon EC2 Auto Scaling はインスタンスの起動時に追加のプライベート IP アドレスを自動的に割り当てることができます。各ネットワークインターフェイスには、インスタンスが起動されるサブネットの CIDR 範囲から単一のプライベート IP アドレスが割り当てられます。この場合、システムはプライマリネットワークインターフェイスにパブリック IPv4 アドレスを自動的に割り当てることができなくなります。利用可能な Elastic IP アドレスを Auto Scaling インスタンスに関連付ける場合を除き、パブリック IPv4 アドレス経由でインスタンスに接続することはできません。

VPC 内のネットワークインターフェイス

VPC の各インスタンスには、デフォルトのネットワークインターフェイスがあります (プライマリネットワークインターフェイス)。プライマリネットワークインターフェイスをインスタンスからデ

タッチすることはできません。追加の ネットワークインスタンスを作成して、ユーザーの VPC 内の任意のインスタンスにアタッチできます。使用できる Elastic Network Interface の最大数はインスタンスタイプによって異なります。

起動テンプレートを使用してインスタンスを起動するときは、追加のネットワークインターフェイスを指定できます。ただし、複数のネットワークインターフェイスを使用して Auto Scaling インスタンスを起動すると、インスタンスと同じサブネットに各インターフェイスが自動的に作成されます。これは、Amazon EC2 Auto Scaling では、起動テンプレートで定義されたサブネットが無視され、Auto Scaling グループに指定されたものが優先されるためです。詳細については、「Auto Scaling グループの起動テンプレートの作成」を参照してください。

同じサブネットから複数のネットワークインターフェイスを作成または、インスタンスにアタッチすると、非対称ルーティングなどのネットワーク問題が発生する場合があります。特に Amazon Linux 以外のバリアントを使用しているインスタンスに発生する場合があります。このタイプの設定が必要な場合は、OS 内でセカンダリネットワークインターフェイスを設定する必要があります。例については、 AWS ナレッジセンターの「Ubuntu EC2 インスタンスでセカンダリネットワークインターフェイスを機能させるにはどうすればよいですか？」を参照してください。

インスタンスのプレイスメントテナンシー

デフォルトでは、VPC のすべてのインスタンスはインスタンス共有テナンシーとして実行されます。Amazon EC2 Auto Scaling では、Dedicated Instances (ハードウェア専有インスタンス) および Dedicated Hosts (ハードウェア専有ホスト) をサポートしています。詳細については、「詳細設定を使用して起動テンプレートを作成する」を参照してください。

AWS Outposts

AWS Outposts は、Amazon VPC を AWS リージョンから Outpost に拡張します。これには、インターネットゲートウェイ、仮想プライベートゲートウェイ、Amazon VPC トランジットゲートウェイ、VPC エンドポイントなど、リージョンでアクセス可能な VPC コンポーネントが含まれます。Outpost はリージョン内のアベイラビリティーゾーンに設置されており、そのアベイラビリティーゾーンの耐障害性のために使用できる拡張機能です。

詳細については、『AWS Outposts ユーザーガイド』を参照してください。

Outpost 内の Application Load Balancer からのトラフィックに対応する Auto Scaling グループをデプロイする方法の例については、「AWS Outpostsで Application Load Balancer を構成」というブログ記事を参照してください。

VPC に関するその他のリソース

以下のトピックでは、VPC およびサブネットの詳細について説明します。

- VPC のプライベートサブネット
  - 例: プライベートサブネットと NAT にサーバーがある VPC
  - NAT ゲートウェイ
- VPC のパブリックサブネット
  - 例: テスト環境の VPC
  - 例: ウェブサーバーとデータベースサーバー用の VPC
- Application Load Balancer のサブネット
  - ロードバランサーのサブネット
- 一般的な VPC 情報
  - Amazon VPC User Guide
  - VPCs ピアリングを使用して VPC を接続する
  - Elastic Network Interface
  - プライベート接続のための VPC エンドポイントを使用する

Amazon EC2 Auto Scaling のセキュリティ

のクラウドセキュリティが最優先事項 AWS です。 AWS のお客様は、セキュリティを最も重視する組織の要件を満たすように構築されたデータセンターとネットワークアーキテクチャから利点を得られます。

セキュリティは、 AWS とお客様間で共有される責任です。責任共有モデル では、これをクラウドのセキュリティおよびクラウド内のセキュリティとして説明しています。

- クラウドのセキュリティ — クラウドで AWS サービスを実行するインフラストラクチャを保護する責任 AWS を担います AWS 。また、 は、ユーザーが安全に使用できるサービス AWS も提供します。AWS コンプライアンスプログラム コンプライアンスプログラム の一環として、サードパーティーの監査者が定期的にセキュリティの有効性をテストおよび検証しています。Amazon EC2 Auto Scaling に適用されるコンプライアンスプログラムの詳細については、AWS 「コンプライアンスプログラムによる対象範囲内の のサービスコンプライアンスプログラム」を参照してください。
- クラウド内のセキュリティ – お客様の責任は、使用する AWS サービスによって決まります。また、お客様は、お客様のデータの機密性、企業の要件、および適用可能な法律および規制などの他の要因についても責任を担います。

このドキュメントは、Amazon EC2 Auto Scaling 使用時における責任共有モデルの適用法を理解するのに役立ちます。以下のトピックで、セキュリティおよびコンプライアンスの目的を満たすように、Amazon EC2 Auto Scaling を設定する方法について説明します。また、Amazon EC2 Auto Scaling リソースのモニタリングや保護に役立つ他の AWS のサービスの使用方法についても説明します。

トピック

- Amazon EC2 Auto Scaling のインフラストラクチャセキュリティ
- Amazon EC2 Auto Scaling のレジリエンス
- Amazon EC2 Auto Scaling でのデータ保護
- Amazon EC2 Auto Scaling の Identity and Access Management
- Amazon EC2 Auto Scaling のコンプライアンス検証
- Amazon EC2 Auto Scaling エンドポイントとインターフェイス VPC エンドポイント

Amazon EC2 Auto Scaling のインフラストラクチャセキュリティ

マネージドサービスである Amazon EC2 Auto Scaling は AWS グローバルネットワークセキュリティで保護されています。 AWS セキュリティサービスと がインフラストラクチャ AWS を保護する方法については、AWS 「 クラウドセキュリティ」を参照してください。インフラストラクチャセキュリティのベストプラクティスを使用して AWS 環境を設計するには、「 Security Pillar AWS Well-Architected Framework」の「Infrastructure Protection」を参照してください。

が AWS 公開した API コールを使用して、ネットワーク経由で Amazon EC2 Auto Scaling にアクセスします。クライアントは以下をサポートする必要があります:

- Transport Layer Security (TLS)。TLS 1.2 は必須で TLS 1.3 がお勧めです。
- DHE (楕円ディフィー・ヘルマン鍵共有) や ECDHE (楕円曲線ディフィー・ヘルマン鍵共有) などの完全前方秘匿性 (PFS) による暗号スイート。これらのモードは、Java 7 以降など、ほとんどの最新システムでサポートされています。

また、リクエストには、アクセスキー ID と、IAM プリンシパルに関連付けられているシークレットアクセスキーを使用して署名する必要があります。または、AWS Security Token Service (AWS STS) を使用して、一時的なセキュリティ認証情報を生成し、リクエストに署名することもできます。

Amazon EC2 Auto Scaling に仮想プライベートクラウド (VPC) エンドポイントを使用することもできます。インターフェイス VPC エンドポイントは、パブリックインターネットにさらされることなく、Amazon VPC リソースがプライベート IP アドレスを使用して Amazon EC2 Auto Scaling にアクセスできるようになります。詳細については、「Amazon EC2 Auto Scaling エンドポイントとインターフェイス VPC エンドポイント」を参照してください。

関連リソース

Amazon EC2 によって提供されるサービストラフィックを分離する機能については、「Amazon Amazon EC2 ユーザーガイド」の「Amazon EC2 のインフラストラクチャセキュリティ」を参照してください。 Amazon EC2

Amazon EC2 Auto Scaling のレジリエンス

AWS グローバルインフラストラクチャは、 AWS リージョン とアベイラビリティーゾーンを中心に構築されています。 は、低レイテンシー、高スループット、および高度の冗長ネットワークで接続

されている複数の物理的に独立および隔離されたアベイラビリティーゾーン AWS リージョン を提供します。アベイラビリティーゾーンでは、ゾーン間で中断することなく自動的にフェイルオーバーするアプリケーションとデータベースを設計および運用することができます。アベイラビリティーゾーンは、従来の単一または複数のデータセンターインフラストラクチャよりも可用性が高く、フォールトトレラントで、スケーラブルです。

AWS リージョン およびアベイラビリティーゾーンの詳細については、AWS 「グローバルインフラストラクチャ」を参照してください。

アベイラビリティーゾーンの地理的な冗長性を活用するには、以下を行います。

- Auto Scaling グループを複数のアベイラビリティーゾーンにわたって配置します。
- 各アベイラビリティーゾーンで少なくとも 1 つのインスタンスを維持します。
- ロードバランサーをアタッチして、受信トラフィックを同じアベイラビリティーゾーンに分散させます。Application Load Balancer を使用する場合は、クロスゾーン負荷分散を有効にして、各 EC2 インスタンスに同程度のトラフィックがかかるようにしてください。これにより、フェイルオーバーイベント時の既存インスタンスへの負荷増加の影響を抑えることができ、クロスゾーン負荷分散を行わない場合よりも耐障害性が向上します。
- Elastic Load Balancing のヘルスチェックが正しく設定されていること、また Auto Scaling グループで有効になっていることを確認してください。その後、インスタンスがヘルスチェックに失敗すると、Elastic Load Balancing はそのインスタンスへのトラフィックの送信を停止して、トラフィックを正常なインスタンスに再ルーティングします。一方、Amazon EC2 Auto Scaling は異常なインスタンスを置き換えます。

Amazon EC2 Auto Scaling は、以下の方法でアプリケーションの耐障害性のニーズをサポートします。

- インスタンスの正常性やアクセス性に問題がないかを確認します。インスタンスに異常があると、そのインスタンスを終了させて新しいインスタンスを起動します。
- 動的スケーリングポリシーが有効な場合は、受信トラフィックに応じてキャパシティーを自動的にスケーリングします。
- スケーリングポリシーをサポートする Amazon CloudWatch メトリクスの信頼性の問題を検出し、データポイントが欠落している場合など、信頼できるメトリクスが利用できない場合にスケールインアクティビティを一時停止します。
- グループのスケーリングに伴い、有効化された各アベイラビリティーゾーンで同等の数のインスタンスを維持しようと試みます。

- 高可用性を維持するためにアベイラビリティーゾーンを使用します。アベイラビリティーゾーンが異常になったとき、Amazon EC2 Auto Scaling は次の処理を実行します。
  - Auto Scaling グループに対して有効になっている別のアベイラビリティーゾーンで新しいインスタンスを起動します。
  - 異常のあるアベイラビリティーゾーンが正常な状態に戻ったときに、有効なすべてのアベイラビリティーゾーンにインスタンスを再配布します。
- 特定のアベイラビリティーゾーンでインスタンスが起動しなかった場合、有効な他のアベイラビリティーゾーンでインスタンスの起動を継続的に試みます。
- Auto Scaling グループに関連付けられたロードバランサーにインスタンスの登録と解除を自動的に行います。そのため、インスタンスを個別に登録したり解除したりする必要はありません。

関連リソース

Amazon EBS が提供するデータの耐障害性のニーズをサポートする機能の詳細については、「Amazon EBS ユーザーガイド」の「Amazon Elastic Block Store の耐障害性」を参照してください。

Amazon EC2 Auto Scaling でのデータ保護

責任 AWS 共有モデル、Amazon EC2 Auto Scaling でのデータ保護に適用されます。このモデルで説明されているように、 AWS はすべての を実行するグローバルインフラストラクチャを保護する責任があります AWS クラウド。お客様は、このインフラストラクチャでホストされているコンテンツに対する管理を維持する責任があります。また、使用する AWS のサービス のセキュリティ設定と管理タスクもユーザーの責任となります。データプライバシーの詳細については、「データプライバシーのよくある質問」を参照してください。欧州でのデータ保護の詳細については、AWS セキュリティブログに投稿された記事「AWS 責任共有モデルおよび GDPR」を参照してください。

データ保護の目的で、 認証情報を保護し AWS アカウント 、 AWS IAM Identity Center または AWS Identity and Access Management (IAM) を使用して個々のユーザーを設定することをお勧めします。この方法により、それぞれのジョブを遂行するために必要な権限のみが各ユーザーに付与されます。また、次の方法でデータを保護することもお勧めします:

- 各アカウントで多要素認証 (MFA) を使用します。
- SSL/TLS を使用して AWS リソースと通信します。TLS 1.2 は必須であり TLS 1.3 がお勧めです。
- で API とユーザーアクティビティのログ記録を設定します AWS CloudTrail。

- AWS 暗号化ソリューションと、 内のすべてのデフォルトのセキュリティコントロールを使用します AWS のサービス。
- Amazon Macie などの高度なマネージドセキュリティサービスを使用します。これらは、Amazon S3 に保存されている機密データの検出と保護を支援します。
- コマンドラインインターフェイスまたは API AWS を介して にアクセスするときに FIPS 140-2 検証済みの暗号化モジュールが必要な場合は、FIPS エンドポイントを使用します。利用可能な FIPS エンドポイントの詳細については、「連邦情報処理規格 (FIPS) 140-2」を参照してください。

お客様の E メールアドレスなどの極秘または機密情報は、タグ、または名前フィールドなどの自由形式のテキストフィールドに配置しないことを強くお勧めします。これは、コンソール、API、または SDK を使用して Amazon EC2 Auto Scaling AWS CLIまたは他の AWS のサービス を使用する場合も同様です。 AWS SDKs 名前に使用する自由記述のテキストフィールドやタグに入力したデータは、課金や診断ログに使用される場合があります。外部サーバーへの URL を提供する場合は、そのサーバーへのリクエストを検証するための認証情報を URL に含めないように強くお勧めします。

Amazon EC2 インスタンスを起動するときに、インスタンスの起動時にユーザーデータをインスタンスに渡すことで、追加の設定を行うことができます。また、インスタンスに渡されるユーザーデータに機密情報を含めないことを強くお勧めします。

AWS KMS keys を使用して Amazon EBS ボリュームを暗号化する

Auto Scaling グループは、 AWS KMS keysを使用してクラウドに保存されている Amazon EBS ボリュームデータを暗号化するように設定できます。Amazon EC2 Auto Scaling は、データを暗号化するための AWS マネージドキーとカスタマーマネージドキーをサポートします。起動設定を使用するときは、カスタマーマネージドキーを指定するための `KmsKeyId` オプションを利用できないことに注意してください。カスタマーマネージドキーを指定するには、その代わりに起動テンプレートを使用してください。詳細については、「Auto Scaling グループの起動テンプレートを作成する」を参照してください。 AWS KMS 暗号化キーを作成、保存、管理する方法については、「 AWS Key Management Service デベロッパーガイド」を参照してください。

起動テンプレートまたは起動設定をセットアップする前に EBS-backed AMI でカスタマーマネージドキーを設定したり、デフォルトで暗号化を使用して、作成する新しい EBS ボリュームとスナップショットコピーの暗号化を実施したりすることも可能です。詳細については、Amazon EC2 ユーザーガイド」の「EBS-backed AMIs による暗号化の使用」および「Amazon EBS ユーザーガイド」の「デフォルトでの暗号化」を参照してください。

> **Note**
>
> 暗号化にカスタマーマネージドキーを使用するときに Auto Scaling インスタンスの起動に必要になるキーポリシーを設定する方法については、「暗号化されたボリュームで使用するために必要な AWS KMS キーポリシー」を参照してください。

関連リソース

Amazon EBS が提供するデータ保護ガイドラインについては、「Amazon EBS ユーザーガイド」の「Amazon Elastic Block Storeでのデータ保護」を参照してください。

暗号化されたボリュームで使用するために必要な AWS KMS キーポリシー

Amazon EC2 Auto Scaling は、サービスにリンクされたロールを使用して、アクセス許可を他の に委任します AWS のサービス。Amazon EC2 Auto Scaling サービスにリンクされたロールは事前定義されており、Amazon EC2 Auto Scaling が AWS のサービス ユーザーに代わって他の を呼び出すために必要なアクセス許可が含まれています。事前定義されたアクセス許可には、 へのアクセスも含まれます AWS マネージドキー。ただし、顧客管理キーへのアクセスは含まれていないため、これらのキーを完全に制御できます。

このトピックでは、Amazon EBS 暗号化のカスタマーマネージドキーを指定するときに Auto Scaling インスタンスを起動するために必要なキーポリシーを設定する方法について説明します。

> **Note**
>
> Amazon EC2 Auto Scaling では、アカウントの暗号化されたボリュームの保護にデフォルトの AWS マネージドキー を使用する場合、追加の承認は不要です。

内容

- 概要
- キーポリシーを設定する
- 例 1: カスタマー管理キーへのアクセスを許可するキーポリシーセクション
- 例 2: カスタマー管理キーへのクロスアカウントアクセスを許可するキーポリシーセクション
- AWS KMS コンソールでキーポリシーを編集する

概要

Amazon EC2 Auto Scaling がインスタンスを起動するときに、Amazon EBS 暗号化に以下 AWS KMS keys を使用できます。 Auto Scaling

- AWS マネージドキー – Amazon EBS が作成、所有、管理するアカウントの暗号化キー。これは、新しいアカウントのデフォルトの暗号化キーです。カスタマーマネージドキーを指定しない限り、AWS マネージドキー は暗号化に使用されます。
- カスタマーマネージドキー – ユーザーが作成、所有、管理するカスタム暗号化キー。詳細については、デベロッパーガイドの「AWS Key Management Service キーの作成」を参照してください。

  注:キーは対称である必要があります。Amazon EBS は非対称カスタマー管理キーをサポートしていません。

カスタマーマネージド型キーは、暗号化されたスナップショットを作成するとき、または暗号化されたボリュームを指定する起動テンプレートを作成するとき、またはデフォルトで暗号化を有効にするときに設定します。

キーポリシーを設定する

KMS キーには、Amazon EC2 Auto Scaling が、カスタマーマネージド型キーで暗号化された Amazon EBS ボリュームを使用してインスタンスを起動できるようにするキーポリシーが必要です。

このページの例を使用して、Amazon EC2 Auto Scaling にカスタマー管理型キーへのアクセスを許可するようにキーポリシーを設定します。カスタマー管理型 キーのキーポリシーは、キーの作成時または後で変更できます。

Amazon EC2 Auto Scaling で CMK のキーポリシーを使用するには、少なくとも、2 つのポリシーステートメントをそのポリシーに追加する必要があります。

- 最初のステートメントでは、`Principal` 要素で指定された IAM アイデンティティに、カスタマー型マネージドキー を直接使用できるようにします。これには、キーに対して AWS KMS `Encrypt`、`Decrypt`、`ReEncrypt*`、`GenerateDataKey*`、および `DescribeKey`オペレーションを実行するアクセス許可が含まれます。
- 2 番目のステートメントでは、 `Principal`要素で指定された IAM ID が `CreateGrant`オペレーションを使用して、独自のアクセス許可のサブセットを、 AWS KMS または別のプリンシパルと

統合されている に委任 AWS のサービス する許可を生成できます。これにより、それらのサービスはお客様に代わって、キーを使用して、暗号化されたリソースを作成できるようになります。

キーポリシーに新しいポリシーステートメントを追加する場合は、ポリシーの既存のステートメントを変更しないでください。

以下の各例では、キー ID やサービスにリンクされたロールの名前など、置き換える必要がある引数は、*###############* として表示されます。ほとんどの場合、サービスにリンクされたロールの名前を Amazon EC2 Auto Scaling サービスにリンクされたロールの名前に置き換えることができます。

詳細については、以下のリソースを参照してください。

- を使用してキーを作成するには AWS CLI、「create-key」を参照してください。
- でキーポリシーを更新するには AWS CLI、「put-key-policy」を参照してください。
- キー ID と Amazon リソースネーム (ARN) を確認するには、AWS Key Management Service デベロッパーガイドの「キー ID と ARN を検索する」を参照してください。
- Amazon EC2 Auto Scaling というサービスにリンクされたロールについては、「Amazon EC2 Auto Scaling のサービスにリンクされたロール」を参照してください。
- Amazon EBS 暗号化と KMS 全般の詳細については、「Amazon EBS ユーザーガイド」および「 デベロッパーガイド」の「Amazon EBS 暗号化」を参照してください。 AWS Key Management Service

例 1: カスタマー管理キーへのアクセスを許可するキーポリシーセクション

以下の 2 つのポリシーステートメントをカスタマー管理型キーのキーポリシーに追加して、例の ARN を、キーへのアクセスが許可されている適切なサービスにリンクされたロールの ARN に置き換えます。この例では、ポリシーセクションは、 という名前のサービスにリンクされたロールに、カスタマーマネージドキーを使用するためのAWSServiceRoleForAutoScalingアクセス許可を付与します。

```
{
   "Sid": "Allow service-linked role use of the customer managed key",
   "Effect": "Allow",
   "Principal": {
       "AWS": [
           "arn:aws:iam::account-id:role/aws-service-role/
autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling"
```

```
        ]
    },
    "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
    ],
    "Resource": "*"
}
```

```
{
    "Sid": "Allow attachment of persistent resources",
    "Effect": "Allow",
    "Principal": {
        "AWS": [
            "arn:aws:iam::account-id:role/aws-service-role/
autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling"
        ]
    },
    "Action": [
        "kms:CreateGrant"
    ],
    "Resource": "*",
    "Condition": {
        "Bool": {
            "kms:GrantIsForAWSResource": true
        }
     }
}
```

例 2: カスタマー管理キーへのクロスアカウントアクセスを許可するキーポリシーセクション

カスタマー管理キーを Auto Scaling グループとは異なるアカウントで作成する場合は、キーへのクロスアカウントアクセスを許可するキーポリシーと組み合わせてグラントを使用する必要があります。

これには 2 つのステップがあり、以下の順序で完了する必要があります。

1. まず、カスタマー管理キーのキーポリシーに以下の 2 つのポリシーステートメントを追加します。サンプル ARN を他のアカウントの ARN に置き換え、*111122223333* を Auto Scaling グループ AWS アカウント を作成する の実際のアカウント ID に置き換えてください。そうすることで、次の CLI コマンドを使用して、指定されたアカウント内の IAM ユーザーまたはロールに、キーに対するグラントを作成する許可を提供できます。ただし、これ自体がキーへのアクセス許可をユーザーに提供するのではありません。

```
{
   "Sid": "Allow external account 111122223333 use of the customer managed key",
   "Effect": "Allow",
   "Principal": {
       "AWS": [
           "arn:aws:iam::111122223333:root"
       ]
   },
   "Action": [
       "kms:Encrypt",
       "kms:Decrypt",
       "kms:ReEncrypt*",
       "kms:GenerateDataKey*",
       "kms:DescribeKey"
   ],
   "Resource": "*"
}
```

```
{
   "Sid": "Allow attachment of persistent resources in external
 account 111122223333",
   "Effect": "Allow",
   "Principal": {
       "AWS": [
           "arn:aws:iam::111122223333:root"
       ]
   },
   "Action": [
       "kms:CreateGrant"
   ],
   "Resource": "*"
}
```

2. 次に、Auto Scaling グループを作成するアカウントから、関連する許可を適切なサービスリンクロールに委任するグラントを作成します。グラントの `Grantee Principal` 要素は、適切なサービスリンクロールの ARN です。`key-id` は、キーの ARN です。

   以下は、アカウント *111122223333* の という名前のサービスにリンクされたロールに、アカウント AWSServiceRoleForAutoScalingのカスタマーマネージドキーを使用するアクセス許可を付与する create-grant CLI コマンドの例です*444455556666*。

```
aws kms create-grant \
  --region us-west-2 \
  --key-id arn:aws:kms:us-
west-2:444455556666:key/1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d \
  --grantee-principal arn:aws:iam::111122223333:role/aws-service-role/
autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling \
  --operations "Encrypt" "Decrypt" "ReEncryptFrom" "ReEncryptTo" "GenerateDataKey"
 "GenerateDataKeyWithoutPlaintext" "DescribeKey" "CreateGrant"
```

   このコマンドが成功するには、リクエストを行うユーザーが `CreateGrant` アクションに対する許可を持っている必要があります。

   次の IAM ポリシーの例では、アカウント *111122223333* 内の IAM アイデンティティ (ユーザーまたはロール) がアカウント *444455556666* 内でカスタマーマネージド型キーの付与を作成できるようにします。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCreationOfGrantForTheKMSKeyinExternalAccount444455556666",
      "Effect": "Allow",
      "Action": "kms:CreateGrant",
      "Resource": "arn:aws:kms:us-
west-2:444455556666:key/1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d"
    }
  ]
}
```

   異なる AWS アカウント内での KMS キーに対するグラントの作成に関する詳細については、「 デベロッパーガイド」の「 AWS KMSでの権限」を参照してください。

> **Important**
>
> 被付与者のプリンシパルとして指定されたサービスにリンクされたロール名は、既存のロールの名前でなければなりません。グラントを作成した後は、Amazon EC2 Auto Scaling が指定された KMS キーを確実に使用できるようにするために、サービスにリンクされたロールを削除して再作成しないでください。

AWS KMS コンソールでキーポリシーを編集する

以前のセクションの例では、キーポリシーにステートメントを追加する方法のみを示しています。これは、キーポリシーを変更する 1 つの方法にすぎません。キーポリシーを変更する最も簡単な方法は、キーポリシーに AWS KMS コンソールのデフォルトビューを使用し、IAM ID (ユーザーまたはロール) を適切なキーポリシーのキーユーザーの 1 人にすることです。詳細については、「AWS Key Management Service デベロッパーガイド」の AWS Management Console 「デフォルトビュー」の使用」を参照してください。

> **Important**
>
> 以下の点に注意してください。コンソールのデフォルトのビューポリシーステートメントには、カスタマーマネージドキーでオペレーションを実行する AWS KMS Revokeアクセス許可が含まれています。アカウント内のカスタマーマネージドキー AWS アカウント へのアクセスを許可し、このアクセス許可を付与した権限を誤って取り消した場合、外部ユーザーは暗号化されたデータやデータの暗号化に使用されたキーにアクセスできなくなります。

Amazon EC2 Auto Scaling の Identity and Access Management

AWS Identity and Access Management (IAM) は、管理者が AWS リソースへのアクセスを安全に制御 AWS のサービス するのに役立つ です。IAM 管理者は、誰を認証 (サインインを許可) し、誰に Amazon EC2 Auto Scaling リソースの使用を承認する (アクセス許可を付与する) かを制御します。IAM は、追加料金なしで AWS のサービス 使用できる です。

Amazon EC2 Auto Scaling を使用するには、アカウントにサインインするための AWS アカウントとセキュリティ認証情報が必要です。詳細については、「IAM ユーザーガイド」のAWS 「セキュリティ認証情報」を参照してください。

完全な IAM ドキュメントについては、「IAM ユーザーガイド」を参照してください。

アクセスコントロール

リクエストを認証するために有効な認証情報を持つことはできますが、アクセス許可を持っていなければ Amazon EC2 Auto Scaling リソースを作成またはアクセスすることはできません。例えば、Auto Scaling グループの作成、起動テンプレートを使用したインスタンスの起動などを実行するための許可が必要です。

以下のセクションでは、IAM 管理者が IAM を使用して、Amazon EC2 Auto Scaling アクションを実行できるユーザーを制御することで、Amazon EC2 Auto Scaling リソースをセキュリティで保護する方法について詳しく説明します。

最初に Amazon EC2 トピックを読むことをお勧めします。Amazon EC2 ユーザーガイド」の「Amazon EC2 の Identity and Access ManagementAmazon EC2」を参照してください。このセクションのトピックを読むことで、アクセスコントロールのアクセス許可 Amazon EC2 のサービス内容と、Amazon EC2 Auto Scaling のリソースのアクセス許可への利用方法についてご理解いただけます。

トピック

- Amazon EC2 Auto Scaling と IAM の連携
- Amazon EC2 Auto Scaling API アクセス許可
- AWS Amazon EC2 Auto Scaling の マネージドポリシー
- Amazon EC2 Auto Scaling のサービスにリンクされたロール
- Amazon EC2 Auto Scaling アイデンティティベースのポリシーの例
- サービス間の混乱した代理の防止
- 起動テンプレートのサポート
- Amazon EC2 インスタンスで実行中のアプリケーション用の IAM ロール

Amazon EC2 Auto Scaling と IAM の連携

IAM を使用して Amazon EC2 Auto Scaling へのアクセスを管理する前に、Amazon EC2 Auto Scaling で使用できる IAM 機能について理解しておく必要があります。

Amazon EC2 Auto Scaling で使用できる IAM 機能

IAM 機能	Amazon EC2 Auto Scaling サポート
アイデンティティベースのポリシー	あり
リソースベースのポリシー	いいえ
ポリシーアクション	あり
ポリシーリソース	はい
ポリシー条件キー (サービス固有)	はい
ACL	なし
ABAC (ポリシー内のタグ)	部分的
一時的な認証情報	はい
サービスロール	あり
サービスリンクロール	はい

Amazon EC2 Auto Scaling およびその他の がほとんどの IAM 機能と AWS のサービス 連携する方法の概要を把握するには、「IAM ユーザーガイド」の「IAM AWS のサービス と連携する 」を参照してください。

Amazon EC2 Auto Scaling のアイデンティティベースのポリシー

アイデンティティベースポリシーをサポートする	あり

アイデンティティベースポリシーは、IAM ユーザー、ユーザーグループ、ロールなど、アイデンティティにアタッチできる JSON 権限ポリシードキュメントです。これらのポリシーは、ユーザーとロールが実行できるアクション、リソース、および条件を制御します。アイデンティティベースのポリシーを作成する方法については、「IAM ユーザーガイド」の「IAM ポリシーの作成」を参照してください。

IAM アイデンティティベースのポリシーでは、許可または拒否するアクションとリソース、およびアクションを許可または拒否する条件を指定できます。プリンシパルは、それが添付されているユーザーまたはロールに適用されるため、アイデンティティベースのポリシーでは指定できません。JSON ポリシーで使用できるすべての要素について学ぶには、「IAM ユーザーガイド」の「IAM JSON ポリシーの要素のリファレンス」を参照してください。

Amazon EC2 Auto Scaling 内のリソースベースのポリシー

リソースベースのポリシーのサポート	なし

リソースベースのポリシーは、リソースに添付する JSON ポリシードキュメントです。リソースベースのポリシーには例として、IAM ロールの信頼ポリシーや Amazon S3 バケットポリシーがあげられます。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスを制御できます。ポリシーが添付されているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーでは、プリンシパルを指定する必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーティッドユーザー、または を含めることができます AWS のサービス。

クロスアカウントアクセスを有効にするには、アカウント全体、または別のアカウントの IAM エンティティをリソースベースのポリシーのプリンシパルとして指定します。リソースベースのポリシーにクロスアカウントのプリンシパルを追加しても、信頼関係は半分しか確立されない点に注意してください。プリンシパルとリソースが異なる にある場合 AWS アカウント、信頼されたアカウントの IAM 管理者は、リソースへのアクセス許可をプリンシパルエンティティ (ユーザーまたはロール) に付与する必要もあります。IAM 管理者は、アイデンティティベースのポリシーをエンティティにアタッチすることで権限を付与します。ただし、リソースベースのポリシーで、同じアカウントのプリンシパルへのアクセス権が付与されている場合は、アイデンティティベースのポリシーを追加する必要はありません。詳細については、「IAM ユーザーガイド」の「IAM ロールとリソースベースのポリシーとの相違点」を参照してください。

Amazon EC2 Auto Scaling のポリシーアクション

ポリシーアクションに対するサポート	はい

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどんなリソースにどんな条件でアクションを実行できるかということです。

JSON ポリシーのAction要素には、ポリシー内のアクセスを許可または拒否するために使用できるアクションが記述されます。ポリシーアクションの名前は通常、関連する AWS API オペレーションと同じです。一致する API オペレーションのない許可のみのアクションなど、いくつかの例外があります。また、ポリシーに複数アクションが必要なオペレーションもあります。これらの追加アクションは、依存アクションと呼ばれます。

このアクションは、関連付けられたオペレーションを実行するためのアクセス許可を付与するポリシーで使用されます。

Amazon EC2 Auto Scaling アクションのリストを確認するには、「サービス認証リファレンス」の「Amazon EC2 Auto Scaling で定義されるアクション」を参照してください。

Amazon EC2 Auto Scaling のポリシーアクションは、アクションの前に以下のプレフィックスを使用します。

```
autoscaling
```

単一のステートメントで複数のアクションを指定するには、アクションをカンマで区切ります。

```
"Action": [
  "autoscaling:action1",
  "autoscaling:action2"
]
```

ワイルドカード (*) を使用して複数のアクションを指定できます。例えば、Describe という単語で始まるすべてのアクションを指定するには、次のアクションを含めます。

```
"Action": "autoscaling:Describe*"
```

Amazon EC2 Auto Scaling のポリシーリソース

ポリシーリソースに対するサポート	はい

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

Resource JSON ポリシーの要素は、オブジェクトあるいはアクションが適用されるオブジェクトを指定します。ステートメントには、Resource または NotResource 要素を含める必要がありま

す。ベストプラクティスとしては、Amazon リソースネーム (ARN) を使用してリソースを指定します。これは、リソースレベルの権限と呼ばれる特定のリソースタイプをサポートするアクションに対して実行できます。

オペレーションのリスト化など、リソースレベルのアクセス許可をサポートしないアクションの場合は、ステートメントがすべてのリソースに適用されることを示すために、ワイルドカード (*) を使用します。

```
"Resource": "*"
```

ARN を使用して、IAM ポリシーを適用する Auto Scaling グループと起動設定を特定できます。

Auto Scaling グループには、次の ARN があります。

```
"Resource": "arn:aws:autoscaling:region:account-
id:autoScalingGroup:uuid:autoScalingGroupName/asg-name"
```

起動設定には次の ARN があります。

```
"Resource": "arn:aws:autoscaling:region:account-
id:launchConfiguration:uuid:launchConfigurationName/lc-name"
```

`CreateAutoScalingGroup` アクションを使用して Auto Scaling グループを指定するには、次の例に示すように UUID をワイルドカード (*) に置き換える必要があります。

```
"Resource": "arn:aws:autoscaling:region:account-
id:autoScalingGroup:*:autoScalingGroupName/asg-name"
```

`CreateLaunchConfiguration` アクションを使用して起動設定を指定するには、次の例に示すように UUID をワイルドカード (*) に置き換える必要があります。

```
"Resource": "arn:aws:autoscaling:region:account-
id:launchConfiguration:*:launchConfigurationName/lc-name"
```

Amazon EC2 Auto Scaling リソースタイプ、およびその ARN の詳細については、「サービス認証リファレンス」の「Amazon EC2 Auto Scaling で定義されるリソースタイプ」を参照してください。各リソースの ARN を指定できるアクションについては、「Amazon EC2 Auto Scaling で定義されるアクション」を参照してください。

> **Note**
>
> ARN を使用して Auto Scaling グループへのアクセスを制御する IAM ポリシーの例については、「削除できる Auto Scaling グループを制御する」を参照してください。。

現在、すべての Amazon EC2 Auto Scaling アクションがリソースレベルのアクセス許可をサポートしているわけではありません。リソースレベルの許可をサポートしていないアクションの場合、ワイルドカード (*) をリソースとして使用する必要があります。

次の Amazon EC2 Auto Scaling アクションは、リソースレベルのアクセス許可をサポートしていません。

- `DescribeAccountLimits`
- `DescribeAdjustmentTypes`
- `DescribeAutoScalingGroups`
- `DescribeAutoScalingInstances`
- `DescribeAutoScalingNotificationTypes`
- `DescribeInstanceRefreshes`
- `DescribeLaunchConfigurations`
- `DescribeLifecycleHooks`
- `DescribeLifecycleHookTypes`
- `DescribeLoadBalancers`
- `DescribeLoadBalancerTargetGroups`
- `DescribeMetricCollectionTypes`
- `DescribeNotificationConfigurations`
- `DescribePolicies`
- `DescribeScalingActivities`
- `DescribeScalingProcessTypes`
- `DescribeScheduledActions`
- `DescribeTags`
- `DescribeTerminationPolicyTypes`
- `DescribeWarmPool`

Amazon EC2 Auto Scaling のポリシー条件キー

サービス固有のポリシー条件キーのサポート	はい

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどんなリソースにどんな条件でアクションを実行できるかということです。

Condition 要素 (または Condition ブロック) を使用すると、ステートメントが有効な条件を指定できます。Condition 要素はオプションです。equal や less than などの条件演算子を使用して条件式を作成することによって、ポリシーの条件とリクエスト内の値を一致させることができます。

1 つのステートメントに複数の Condition 要素を指定する場合、または 1 つの Condition 要素に複数のキーを指定する場合、 AWS は AND 論理演算子を使用してそれらを評価します。1 つの条件キーに複数の値を指定すると、 は論理OR演算を使用して条件 AWS を評価します。ステートメントの権限が付与される前にすべての条件が満たされる必要があります。

条件を指定する際にプレースホルダー変数も使用できます。例えば IAM ユーザーに、IAM ユーザー名がタグ付けされている場合のみリソースにアクセスできる権限を付与することができます。詳細については、「IAM ユーザーガイド」の「IAM ポリシーの要素: 変数およびタグ」を参照してください。

AWS は、 グローバル条件キーとサービス固有の条件キーをサポートします。すべての AWS グローバル条件キーを確認するには、IAM ユーザーガイドのAWS 「 グローバル条件コンテキストキー」を参照してください。

Amazon EC2 Auto Scaling が次の条件キーをサポートすることで、サポートしているアクションへのアクセスを制御でき、Auto Scaling グループの設定を強制することができます。

- `autoscaling:InstanceTypes`
- `autoscaling:LaunchConfigurationName`
- `autoscaling:LaunchTemplateVersionSpecified`
- `autoscaling:LoadBalancerNames`
- `autoscaling:MaxSize`
- `autoscaling:MinSize`
- `autoscaling:ResourceTag/key-name: tag-value`
- `autoscaling:TargetGroupARNs`

- autoscaling:VPCZoneIdentifiers

次の条件キーは、起動構成リクエストの作成に固有のものです。

- autoscaling:ImageId
- autoscaling:InstanceType
- autoscaling:MetadataHttpEndpoint
- autoscaling:MetadataHttpPutResponseHopLimit
- autoscaling:MetadataHttpTokens
- autoscaling:SpotPrice

Amazon EC2 Auto Scaling は、リクエスト内のタグまたは Auto Scaling グループに存在するタグに基づいてアクセス許可を定義するために使用できる次のグローバル条件キーもサポートしています。詳しくは、「Auto Scaling グループとインスタンスにタグを付ける」を参照してください。

- aws:RequestTag/*key-name*: *tag-value*
- aws:ResourceTag/*key-name*: *tag-value*
- aws:TagKeys: *[tag-key, ...]*

条件キーを使用できる Amazon EC2 Auto Scaling API アクションについて理解するには、「サービス認証リファレンス」の「Amazon EC2 Auto Scaling で定義されるアクション」を参照してください。Amazon EC2 Auto Scaling 条件キーに関する詳細については、「Amazon EC2 Auto Scaling の条件キー」を参照してください。

> **Note**
>
> 条件キーを使用して、サポートしているアクションへのアクセスを制御し、Auto Scaling グループの設定を強制する IAM ポリシーの例については、次のリソースを参照してください。
>
> - 起動テンプレートとバージョン番号を要求する – この例では、Auto Scaling グループを作成または更新するときに、起動テンプレートと起動テンプレートのバージョン番号を指定する必要があります。
> - 作成できる Auto Scaling グループのサイズを制御する — この例では、特定のタグを持つ Auto Scaling グループを作成または更新するときに、 プロパティMinSizeと MaxSizeプロパティに指定できる値に制約を適用します。

- 削除できるスケーリングポリシーを制御する – この例では、特定のタグを持たない Auto Scaling グループに対してのみスケーリングポリシーの削除が許可されるように強制しています。

Amazon EC2 Auto Scaling の ACL

ACL のサポート	なし

アクセスコントロールリスト (ACL) は、どのプリンシパル (アカウントメンバー、ユーザー、またはロール) がリソースにアクセスするための権限を持つかを制御します。ACL はリソースベースのポリシーに似ていますが、JSON ポリシードキュメント形式は使用しません。

Amazon EC2 Auto Scaling による ABAC

ABAC (ポリシー内のタグ) のサポート	部分的

属性ベースのアクセスコントロール (ABAC) は、属性に基づいて権限を定義する認可戦略です。では AWS、これらの属性はタグ と呼ばれます。タグは、IAM エンティティ (ユーザーまたはロール) および多くの AWS リソースにアタッチできます。エンティティとリソースのタグ付けは、ABAC の最初の手順です。次に、プリンシパルのタグがアクセスを試行するリソースのタグと一致したときにオペレーションを許可するよう、ABAC ポリシーを設計します。

ABAC は、急成長する環境やポリシー管理が煩雑になる状況で役立ちます。

タグに基づいてアクセスを制御するには、`aws:ResourceTag/key-name`、`aws:RequestTag/key-name`、または `aws:TagKeys` の条件キーを使用して、ポリシーの条件要素 でタグ情報を提供します。

サービスがすべてのリソースタイプに対して 3 つの条件キーすべてをサポートする場合、そのサービスの値ははいです。サービスが一部のリソースタイプに対してのみ 3 つの条件キーすべてをサポートする場合、値は Partial です。

ABAC の詳細については、「IAM ユーザーガイド」の [ABAC とは?」を参照してください。ABAC をセットアップする手順を説明するチュートリアルについては、「IAM ユーザーガイド」の「属性ベースのアクセス制御 (ABAC) を使用する」を参照してください。

ABAC はタグをサポートするリソースでは可能ですが、すべてのリソースがタグをサポートしているわけではありません。起動設定とスケーリングポリシーはタグをサポートしていませんが、Auto Scaling グループはタグをサポートしています。

詳細については、「Auto Scaling グループとインスタンスにタグを付ける」を参照してください。

Amazon EC2 Auto Scaling での一時認証情報の使用

一時的な認証情報のサポート	はい

一部の AWS のサービス は、一時的な認証情報を使用してサインインすると機能しません。一時的な認証情報を AWS のサービス 使用できる などの詳細については、AWS のサービス IAM ユーザーガイドの「IAM と連携する 」を参照してください。

ユーザー名とパスワード以外の AWS Management Console 方法で にサインインする場合は、一時的な認証情報を使用しています。例えば、会社の Single Sign-On (SSO) リンク AWS を使用して にアクセスすると、そのプロセスは自動的に一時的な認証情報を作成します。また、ユーザーとしてコンソールにサインインしてからロールを切り替える場合も、一時的な認証情報が自動的に作成されます。ロールの切り替えに関する詳細については、「IAM ユーザーガイド」の「ロールへの切り替え (コンソール)」を参照してください。

一時的な認証情報は、 AWS CLI または AWS API を使用して手動で作成できます。その後、これらの一時的な認証情報を使用して、長期的なアクセスキーを使用する代わりに、動的に一時的な認証情報を生成する AWS. AWS recommends にアクセスできます。詳細については、「IAM の一時的セキュリティ認証情報」を参照してください。

Amazon EC2 Auto Scaling のサービスロール

サービスロールに対するサポート	あり

サービスロールとは、サービスがユーザーに代わってアクションを実行するために引き受ける IAM ロールです。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除できます。詳細については、「IAM ユーザーガイド」の「AWS のサービスにアクセス許可を委任するロールの作成」を参照してください。

Amazon SNS トピックまたは Amazon SQS キューを通知するライフサイクルフックを作成する場合は、ロールを指定して、Amazon EC2 Auto Scaling がユーザーに代わって Amazon SNS または

Amazon SQS にアクセスすることを許可する必要があります。IAM コンソールを使用して、ライフサイクルフックのサービスロールを設定します。コンソールは、マネージドポリシーを使用して十分なアクセス許可セットを持つロールを作成するのに役立ちます。詳細については、Amazon SNS を使用した通知の受信およびAmazon SQS を使用した通知の受信を参照してください。

Auto Scaling グループを作成するときに、オプションでサービスロールを渡して、Amazon EC2 インスタンスが AWS のサービス ユーザーに代わって他の にアクセスすることを許可できます。Amazon EC2 インスタンスのサービスロール (起動テンプレート用または起動設定用の Amazon EC2 インスタンスプロファイル) は、インスタンス起動時に Auto Scaling グループ内のすべての EC2 インスタンスに割り当てられる特殊なサービスロールです。IAM コンソールと を使用して AWS CLI 、このサービスロールを作成または編集できます。詳細については、「Amazon EC2 インスタンスで実行中のアプリケーション用の IAM ロール」を参照してください。

> **Warning**
>
> サービスロールの許可を変更すると、Amazon EC2 Auto Scaling の機能を損なうおそれがあります。Amazon EC2 Auto Scaling が指示する場合以外はサービスロールを編集しないでください。

Amazon EC2 Auto Scaling のサービスにリンクされたロール

サービスリンクロールのサポート	はい

サービスにリンクされたロールは、 にリンクされたサービスロールの一種です AWS のサービス。サービスは、ユーザーに代わってアクションを実行するロールを引き受けることができます。サービスにリンクされたロールは に表示され AWS アカウント 、サービスによって所有されます。IAM 管理者は、サービスリンクロールの許可を表示できますが、編集することはできません。

Amazon EC2 Auto Scaling でのサービスにリンクされたロールの作成または管理の詳細については、「Amazon EC2 Auto Scaling のサービスにリンクされたロール」を参照してください。

Amazon EC2 Auto Scaling API アクセス許可

「Amazon EC2 Auto Scaling のポリシーアクション」で説明されているように、必要な Amazon EC2 Auto Scaling API アクションを呼び出す許可をユーザーに付与する必要があります。さらに、一部の Amazon EC2 Auto Scaling アクションでは、他の AWS APIs から特定のアクションを呼び出すアクセス許可をユーザーに付与する必要があります。

他の AWS API からの必要なアクセス許可

Amazon EC2 Auto Scaling API のアクセス許可に加えて、ユーザーは、関連するアクションを正常に実行するために、他の AWS APIsから次のアクセス許可を持っている必要があります。

Auto Scaling グループの作成 (autoscaling:CreateAutoScalingGroup)

- iam:CreateServiceLinkedRole – その口ールがまだ存在しない場合に、デフォルトのサービスにリンクされた口ールを作成します。
- iam:PassRole – 起動時に IAM 口ールをサービスまたは EC2 インスタンスに渡す。デフォルト以外のサービスにリンクされた口ール、ライフサイクルフックの IAM 口ール、またはインスタンスプロファイル (IAM 口ールのコンテナ) を指定する起動テンプレートが提供されている場合に必要です。
- ec2:RunInstances - 起動テンプレートが提供されたときにインスタンスを起動します。
- ec2:CreateTags – タグ仕様の起動テンプレートが提供されているときに、起動時にインスタンスとボリュームにタグを付ける。

ライフサイクルフックの作成 (autoscaling:PutLifecycleHook)

- iam:PassRole – IAM 口ールをサービスに渡す。IAM 口ールが指定されている場合に必要です。

VPC Lattice ターゲットグループをアタッチする (autoscaling:AttachTrafficSources)

- vpc-lattice:RegisterTargets – インスタンスをターゲットグループに自動的に登録します。

VPC Lattice ターゲットグループをデタッチする (autoscaling:DetachTrafficSources)

- vpc-lattice:DeregisterTargets – ターゲットグループへのインスタンスの登録を自動的に解除します。

起動設定を作成する (autoscaling:CreateLaunchConfiguration)

- ec2:DescribeImages
- ec2:DescribeInstances
- ec2:DescribeInstanceAttribute
- ec2:DescribeKeyPairs
- ec2:DescribeSecurityGroups
- ec2:DescribeSpotInstanceRequests
- ec2:DescribeVpcClassicLink

- `iam:PassRole` – 起動時に IAM ロールを EC2 インスタンスに渡す。起動設定でインスタンスプロファイル (IAM ロールのコンテナ) が指定されている場合に必要です。

AWS Amazon EC2 Auto Scaling の マネージドポリシー

AWS 管理ポリシーは、 によって作成および管理されるスタンドアロンポリシーです AWS。 AWS 管理ポリシーは、多くの一般的なユースケースにアクセス許可を付与するように設計されているため、ユーザー、グループ、ロールにアクセス許可の割り当てを開始できます。

AWS 管理ポリシーは、すべての AWS お客様が使用できるため、特定のユースケースに対して最小特権のアクセス許可を付与しない場合があることに注意してください。ユースケース別にカスタマーマネージドポリシーを定義して、マネージドポリシーを絞り込むことをお勧めします。

AWS 管理ポリシーで定義されているアクセス許可は変更できません。が AWS 管理ポリシーで定義されたアクセス許可 AWS を更新すると、ポリシーがアタッチされているすべてのプリンシパル ID (ユーザー、グループ、ロール) が更新されます。 は、新しい AWS のサービス が起動されたとき、または既存のサービスで新しい API AWS オペレーションが使用可能になったときに、 AWS 管理ポリシーを更新する可能性が最も高くなります。

詳細については、「IAM ユーザーガイド」の「AWS マネージドポリシー」を参照してください。

Amazon EC2 Auto Scaling 管理ポリシー

(IAM) ID AWS Identity and Access Management （ ユーザーまたはロール) には、次の管理ポリシーをアタッチできます。各ポリシーは Amazon EC2 Auto Scaling のすべてまたは一部の API アクションへのアクセスを提供します。

- AutoScalingConsoleFullアクセス — を使用して Amazon EC2 Auto Scaling へのフルアクセスを許可します AWS Management Console。このポリシーは、起動設定を使用しているときは機能しますが、起動テンプレートを使用しているときは機能しません。
- AutoScalingConsoleReadOnlyAccess – を使用して Amazon EC2 Auto Scaling への読み取り専用アクセスを許可します AWS Management Console。このポリシーは、起動設定を使用しているときは機能しますが、起動テンプレートを使用しているときは機能しません。
- AutoScalingFullAccess – AWS CLI または SDKs からの完全な Amazon EC2 Auto Scaling アクセスを必要とするが、 AWS Management Console アクセスを必要としない IAM ID の Amazon EC2 Auto Scaling へのフルアクセスを許可します。
- AutoScalingReadOnlyアクセス — AWS CLI または SDKs のみを呼び出す IAM ID に対して Amazon EC2 Auto Scaling への読み取り専用アクセスを許可します。

コンソールから起動テンプレートを使用する場合は、起動テンプレートに固有の追加のアクセス許可を付与する必要があります。詳細については、「起動テンプレートのサポート」で詳しく説明します。Amazon EC2 Auto Scaling コンソールは、ec2アクションに、起動テンプレートに関する情報を表示したり、起動テンプレートを使用してインスタンスを起動したりできるアクセス許可が必要です。

AutoScalingServiceRoleポリシー AWS 管理ポリシー

このポリシーは、Amazon EC2 Auto Scaling がユーザーに代わってアクションを実行できるようにするサービスにリンクされたロールにアタッチされます。詳細については、「Amazon EC2 Auto Scaling のサービスにリンクされたロール」を参照してください。

このポリシーのアクセス許可を確認するには、「 管理AutoScalingServiceRoleポリシーリファレンス」の「 ポリシー」を参照してください。 AWS

AWS マネージドポリシーに対する Amazon EC2 Auto Scaling の更新

Amazon EC2 Auto Scaling の AWS マネージドポリシーの更新に関する詳細を、このサービスがこれらの変更の追跡を開始した以降の分について表示します。このページの変更に関する自動通知を有効にするには、Amazon EC2 Auto Scaling ドキュメントの履歴ページから RSS フィードをサブスクライブしてください。

変更	説明	日付
Amazon EC2 Auto Scaling は、サービスにリンクされたロールにアクセス許可を追加します。	このAutoScalingServiceRolePolicy ポリシーは、検証を改善するために VPC のすべてのセキュリティグループを取得するための Amazon EC2 GetSecurityGroupsForVpc API アクションを呼び出すアクセス許可と、特定の一連のインスタンス要件を満たすインスタンスタイプに関する情報を取得するための Amazon EC2 GetInstanceTypesFromInstanceRequirements API	2024 年 2 月 29 日

変更	説明	日付
	アクションを付与するようになりました。詳細については、「Amazon EC2 Auto Scaling のサービスにリンクされたロール」を参照してください。	

変更	説明	日付
Amazon EC2 Auto Scaling は、サービスにリンクされたロールにアクセス許可を追加します。	`AutoScalingServiceRolePolicy` ポリシーにより、VPC Lattice との統合に必要な API アクションにアクセスする権限がサービスに付与されるようになりました。 • `GetTargetGroup` および `ListTargetGroup` アクション。VPC Lattice ターゲットグループに関する情報を取得するために必要です。 • `RegisterTargets` および `DeregisterTargets` アクション。VPC Lattice ターゲットグループへのインスタンスの登録と登録解除に必要です。 • `ListTargets` 。Amazon EC2 Auto Scaling が VPC Lattice ターゲットグループに登録されたインスタンスの正常性の情報を取得できるようにします。 詳細については、「Amazon EC2 Auto Scaling のサービスにリンクされたロール」を参照してください。	2022 年 12 月 6 日

変更	説明	日付
Amazon EC2 Auto Scaling は、サービスにリンクされたロールにアクセス許可を追加します。	起動テンプレートの作成時に AWS Systems Manager パラメータを AMI ID のエイリアスとして使用できるように、`AutoScalingServiceRolePolicy` ポリシーは API アクションを呼び出すアクセス許可を付与する AWS Systems Manager GetParameters ようになりました。詳細については、「Amazon EC2 Auto Scaling のサービスにリンクされたロール」を参照してください。	2022 年 3 月 28 日
Amazon EC2 Auto Scaling は、サービスにリンクされたロールにアクセス許可を追加します。	予測スケーリングをサポートするために、`AutoScalingServiceRolePolicy` ポリシーに CloudWatch GetMetricData API アクションを呼び出すアクセス許可が含まれるようになりました。詳細については、「Amazon EC2 Auto Scaling のサービスにリンクされたロール」を参照してください。	2021 年 5 月 19 日
Amazon EC2 Auto Scaling が変更の追跡を開始しました	Amazon EC2 Auto Scaling が AWS マネージドポリシーの変更の追跡を開始しました。	2021 年 5 月 19 日

Amazon EC2 Auto Scaling のサービスにリンクされたロール

Amazon EC2 Auto Scaling は、ユーザーに代わって他の AWS のサービス を呼び出すために必要な、アクセス許可用の「サービスにリンクされたロール」を使用します。サービスにリンクされたロールは、 に直接リンクされた一意のタイプの IAM ロールです AWS のサービス。

サービスにリンクされたロールは、他の AWS のサービス にアクセス許可を委任するためのセキュアな方法を提供します。これは、リンクされたサービスのみが、サービスにリンクされたロールを引き受けることができるためです。詳細については、「 IAM ユーザーガイド」の「サービスにリンクされたロールの使用」を参照してください。サービスにリンクされたロールを使用すると、すべての API コールを から表示することもできます AWS CloudTrail。これがモニタリングと監査の要件を満たすのに役立つのは、Amazon EC2 Auto Scaling によってお客様に代わって実行されるすべてのアクションを追跡できるためです。詳細については、「Amazon EC2 Auto Scaling API 呼び出しを口グに記録する AWS CloudTrail」を参照してください。

以下のセクションでは、Amazon EC2 Auto Scaling サービスにリンクされたロールを作成および管理する方法について説明します。まず、IAM アイデンティティ (ユーザーまたはロールなど) がサービスにリンクされたロールを作成、編集、削除を行うための許可を設定します。詳細については、「 IAM ユーザーガイド」の「サービスにリンクされたロールの使用」を参照してください。

内容

- 概要
- サービスにリンクされたロールによって付与されるアクセス許可
- サービスにリンクされたロールを作成する (自動)
- サービスにリンクされたロールを作成する (マニュアル)
- サービスにリンクされたロールを編集する
- サービスにリンクされたロールを削除する
- Amazon EC2 Auto Scaling サービスリンクロールがサポートされるリージョン

概要

Amazon EC2 Auto Scaling サービスにリンクされたロールには 2 つのタイプがあります。

- という名前のアカウントのデフォルトのサービスにリンクされたロールAWSServiceRoleForAutoScaling。このロールは、自動的に Auto Scaling グループに割り当てられます。ただし、別のサービスにリンクされたロールを指定している場合を除きます。

- ロールの作成時に指定するカスタムサフィックスを持つサービスにリンクされたロール。例: AWSServiceRoleForAutoScaling_*mysuffix* 。

カスタムサフィックス付きのサービスにリンクされたロールのアクセス許可は、デフォルトのサービスにリンクされたロールのアクセス許可と同じです。いずれの場合も、ロールを編集することはできません。また、Auto Scaling グループが使用中の場合は削除することもできません。唯一の違いは、ロール名サフィックスです。

AWS Key Management Service キーポリシーを編集するときにどちらのロールも指定して、Amazon EC2 Auto Scaling によって起動されるインスタンスをカスタマーマネージドキーで暗号化できます。ただし、特定のカスタマー管理キーへのきめ細かなアクセスを許可する場合は、サービスにリンクされたロールカスタムサフィックスを使用する必要があります。カスタムサフィックス付きのサービスにリンクされたロールを使用すると、以下のことが可能です。

- カスタマー管理キーをより詳細にコントロールする
- ログでどの Auto Scaling グループが API コールを行ったかを追跡する CloudTrail機能

一部のユーザーにのみアクセスを許可するカスタマー管理キー を作成する場合は、以下のステップに従って、カスタムサフィックス付きのサービスにリンクされたロールを使用できます。

1. カスタムサフィックス付きのサービスにリンクされたロールを作成します。詳細については、「サービスにリンクされたロールを作成する (マニュアル)」を参照してください。
2. サービスにリンクされたロールにカスタマー管理キー へのアクセスを許可します。サービスにリンクされたロールにキーの使用を許可するキーポリシーの詳細については、「暗号化されたボリュームで使用するために必要な AWS KMS キーポリシー」を参照してください。
3. ユーザーに、作成したサービスにリンクされたロールへのアクセスを許可します。IAM ポリシーの作成の詳細については、「どのサービスにリンクされたロールを渡すことができるかを制御する (を使用 PassRole)」を参照してください。ユーザーが、サービスにリンクされたロールを渡すためのアクセス許可なしでそのロールを指定しようとすると、エラーが表示されます。

サービスにリンクされたロールによって付与されるアクセス許可

Amazon EC2 Auto Scaling は、 という名前のサービスにリンクされたロールAWSServiceRoleForAutoScalingまたはカスタムサフィックスのサービスにリンクされたロールを使用します。

サービスにリンクされたロールはその引き受け時に、以下のサービスを信頼します。

- autoscaling.amazonaws.com

ロールのアクセス許可ポリシー によりAutoScalingServiceRolePolicy、Amazon EC2 Auto Scaling は次のアクションを実行できます。

- ec2 – EC2 インスタンスを作成、説明、変更、開始/停止、および終了します。
- iam – IAM ロールを EC2 インスタンスに渡して、インスタンスで実行されているアプリケーションがロールの一時的な認証情報にアクセスできるようにします。
- iam – AWSServiceRoleForEC2Spotサービスにリンクされたロールを作成して、Amazon EC2 Auto Scaling がユーザーに代わってスポットインスタンスを起動できるようにします。
- elasticloadbalancing – Elastic Load Balancing でインスタンスを登録および登録解除し、登録されたターゲットの状態を確認します。
- cloudwatch – スケーリングポリシーのアラームを作成、説明、変更、削除 CloudWatchし、予測スケーリングに使用されるメトリクスを取得します。
- sns — インスタンスの起動または終了時に Amazon SNS に通知を発行します。
- events – ユーザーに代わってルールを作成、説明、更新、削除 EventBridgeします。
- ssm – 起動テンプレートで Systems Manager パラメータを AMI ID のエイリアスとして使用する場合は、Parameter Store からパラメータを読み取ります。
- vpc-lattice – VPC Lattice でインスタンスを登録および登録解除し、登録されたターゲットの状態を確認します。

サービスにリンクされたロールを作成する (自動)

Amazon EC2 Auto Scaling は、Auto Scaling グループを初めて作成するときにAWSServiceRoleForAutoScaling、サービスにリンクされたロールを作成します。ただし、カスタムサフィックスサービスにリンクされたロールを手動で作成して、グループの作成時に指定する場合は除きます。

> **Important**
>
> サービスにリンクされたロールを作成するための IAM アクセス許可が必要です。それ以外の場合、自動作成は失敗します。詳細については、IAM ユーザーガイドおよびサービスにリンクされたロールの作成このガイドの「サービスにリンクされたロールのアクセス許可」を参照してください。

Amazon EC2 Auto Scaling は、サービスにリンクされたロールのサポートを 2018 年 3 月に開始しました。それ以前に Auto Scaling グループを作成した場合、Amazon EC2 Auto Scaling はアカウントにAWSServiceRoleForAutoScalingロールを作成しました。詳細については、IAM ユーザーガイドの「私の AWS アカウントに新しいロールが表示される」を参照してください。

サービスにリンクされたロールを作成する (マニュアル)

サービスにリンクされたロールを作成するには (コンソール)

1. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
2. ナビゲーションペインで [ロール]、[ロールの作成] の順に選択します。
3. [Select trusted entity] (信頼されたエンティティの選択) で、[AWS のサービス] を選択します。
4. [このロールを使用するサービスを選択] で、[EC2 Auto Scaling]、[EC2 Auto Scaling] ユースケースの順に選択します。
5. [Next: Permissions (次へ: アクセス許可)]、[Next: Tags (次へ: タグ)]、[Next: Review (次へ: レビュー)] の順に選択します。注意: サービスにリンクされたロールの作成時にタグ付けを行うことはできません。
6. レビューページで、ロール名を空白のままにして、 という名前のサービスにリンクされたロールを作成するかAWSServiceRoleForAutoScaling、サフィックスを入力してAWSServiceRoleForAutoScaling_***suffix*** という名前のサービスにリンクされたロールを作成します。
7. (オプション) [ロールの説明] で、サービスにリンクされたロールの説明を編集します。
8. [ロールの作成] を選択します。

サービスリンクロールの作成 (AWS CLI)

次の create-service-linked-role CLI コマンドを使用して、AWSServiceRoleForAutoScaling_***suffix*** という名前の Amazon EC2 Auto Scaling のサービスにリンクされたロールを作成します。

```
aws iam create-service-linked-role --aws-service-name autoscaling.amazonaws.com --
custom-suffix suffix
```

このコマンドの出力には、サービスにリンクされたロールの ARN が含まれており、これを使用してサービスにリンクされたロールにカスタマー管理キーへのアクセスを許可できます。

```
{
```

```
    "Role": {
        "RoleId": "ABCDEF0123456789ABCDEF",
        "CreateDate": "2018-08-30T21:59:18Z",
        "RoleName": "AWSServiceRoleForAutoScaling_suffix",
        "Arn": "arn:aws:iam::123456789012:role/aws-service-role/
autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling_suffix",
        "Path": "/aws-service-role/autoscaling.amazonaws.com/",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Action": [
                        "sts:AssumeRole"
                    ],
                    "Principal": {
                        "Service": [
                            "autoscaling.amazonaws.com"
                        ]
                    },
                    "Effect": "Allow"
                }
            ]
        }
    }
}
```

詳細については、『IAM ユーザーガイド』の「サービスにリンクされたロールの作成」を参照してください。

サービスにリンクされたロールを編集する

Amazon EC2 Auto Scaling 用に作成されたサービスにリンクされたロールは編集できません。サービスにリンクされたロールを作成した後、ロールの名前またはアクセス許可を変更することはできません。ただし、ロールの説明は編集できます。詳細については、IAM ユーザーガイドの「サービスにリンクされたロールの編集」を参照してください。

サービスにリンクされたロールを削除する

Auto Scaling グループを使用していない場合、そのサービスにリンクされたロールを削除することをお勧めします。ロールを削除すると、使用されていないエンティティやアクティブにモニタリングおよび維持されていないエンティティがなくなります。

サービスにリンクされたロールを削除するには、まずその関連依存リソースを削除します。これにより、リソースに対する Amazon EC2 Auto Scaling アクセス許可を誤って取り消すことがなくなります。サービスにリンクされたロールが複数の Auto Scaling グループで使用されている場合、サービスにリンクされたロールを削除する前に、そのロールを使用するすべての Auto Scaling グループを削除する必要があります。詳細については、「Auto Scaling インフラストラクチャを削除する」を参照してください。

IAM を使用して、サービスにリンクされたロールを削除できます。詳細については、IAM ユーザーガイド の「サービスにリンクされたロールの削除」を参照してください。

AWSServiceRoleForAutoScaling サービスにリンクされたロールを削除すると、Auto Scaling グループを作成し、別のサービスにリンクされたロールを指定しないときに、Amazon EC2 Auto Scaling によってロールが再度作成されます。

Amazon EC2 Auto Scaling サービスリンクロールがサポートされるリージョン

Amazon EC2 Auto Scaling は、サービス AWS リージョン が利用可能なすべての でサービスにリンクされたロールの使用をサポートします。

Amazon EC2 Auto Scaling アイデンティティベースのポリシーの例

デフォルトでは、 内のまったく新しいユーザー AWS アカウント には、何もするアクセス許可がありません。IAM 管理者は、IAM アイデンティティ (ユーザーやロールなど) に Amazon EC2 Auto Scaling API アクションを実行する許可を与える IAM ポリシーを作成して割り当てる必要があります。

これらの JSON ポリシードキュメント例を使用して IAM ポリシーを作成する方法については、IAM ユーザーガイドの「JSON タブでのポリシーの作成」を参照してください。

以下に示しているのは、アクセス許可ポリシーの例です。

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
          "autoscaling:CreateAutoScalingGroup",
          "autoscaling:UpdateAutoScalingGroup",
          "autoscaling:DeleteAutoScalingGroup"
      ],
      "Resource": "*",
      "Condition": {
```

```
            "StringEquals": { "autoscaling:ResourceTag/purpose": "testing" }
        }
    },
    {
        "Effect": "Allow",
        "Action": "autoscaling:Describe*",
        "Resource": "*"
    }]
}
```

このサンプルポリシーは、グループが **purpose=testing** タグを使用している場合に限り、Auto Scaling グループを作成、更新、削除する許可を付与します。Describe アクションはリソースレベルの許可をサポートしないため、条件のない別のステートメントで指定する必要があります。起動テンプレートを使用してインスタンスを起動するには、ユーザーに ec2:RunInstances アクセス許可も必要です。詳細については、「起動テンプレートのサポート」を参照してください。

> **Note**
>
> 独自のカスタム IAM ポリシーを作成し、IAM アイデンティティ (ユーザーまたはロール) に Amazon EC2 Auto Scaling アクションを実行するための許可または拒否することができます。これらのカスタムポリシーは、指定された許可が必要な IAM アイデンティティにアタッチできます。次の例では、いくつかの一般的ユースケースの許可を示します。
> Amazon EC2 Auto Scaling API アクションの中には、アクションによって作成/変更できるポリシー内に特定の Auto Scaling グループを持つことを許可するものもあります。これらのアクションの対象となるリソースを制限するには、Auto Scaling グループの ARN を個別に指定します。ただし、ベストプラクティスとして、特定のタグを持つ Auto Scaling グループに対するアクションを許可 (または拒否) するタグベースのポリシーを使用することをお勧めします。

例

- 作成できる Auto Scaling グループのサイズを制御する
- 使用できるタグキーとタグ値を制御する
- 削除できる Auto Scaling グループを制御する
- 削除できるスケーリングポリシーを制御する
- インスタンスの更新アクションへのアクセスを制御する
- サービスにリンクされたロールの作成

- どのサービスにリンクされたロールを渡すことができるかを制御する (を使用 PassRole)

作成できる Auto Scaling グループのサイズを制御する

次のポリシーでは、リクエスタが最小サイズとして **1** 未満または最大サイズとして **10** より大きい値を指定しない限り、タグ **environment=development** を持つすべての Auto Scaling グループを作成および更新する許可が付与されます。可能な限りタグを使用して、アカウント内の Auto Scaling グループへのアクセスを制御できるようにします。

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
          "autoscaling:CreateAutoScalingGroup",
          "autoscaling:UpdateAutoScalingGroup"
      ],
      "Resource": "*",
      "Condition": {
          "StringEquals": { "autoscaling:ResourceTag/environment": "development" },
          "NumericGreaterThanEqualsIfExists": { "autoscaling:MinSize": 1 },
          "NumericLessThanEqualsIfExists": { "autoscaling:MaxSize": 10 }
      }
   }]
}
```

Auto Scaling グループへのアクセス制御にタグを使用していない場合は、ARN を使用して IAM ポリシーが適用される Auto Scaling グループを識別できます。

Auto Scaling グループには、次の ARN があります。

```
"Resource": "arn:aws:autoscaling:region:account-
id:autoScalingGroup:*:autoScalingGroupName/my-asg"
```

複数の ARN をリストに含めて指定することもできます。Resource 要素での Amazon EC2 Auto Scaling リソースの ARN の指定の詳細については、「Amazon EC2 Auto Scaling のポリシーリソース」を参照してください。

使用できるタグキーとタグ値を制御する

また、IAM ポリシーの条件を使用して、Auto Scaling グループに適用できるタグキーとタグ値を制御することもできます。リクエスタが特定のタグを指定する場合に限り、Auto Scaling グループを作成またはタグ付けする許可を付与するには、`aws:RequestTag` 条件キーを使用します。特定のタグキーのみ許可するには、`aws:TagKeys` 修飾子とともに `ForAllValues` 条件キーを使用します。

次のポリシーでは、リクエストでキー **environment** にタグを指定することをリクエスタに要求されます。**"?*"** 値は、タグキーに何らかの値を含めることを強制します。ワイルドカードを含める場合は、`StringLike` 条件演算子を使用する必要があります。

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
          "autoscaling:CreateAutoScalingGroup",
          "autoscaling:CreateOrUpdateTags"
      ],
      "Resource": "*",
      "Condition": {
          "StringLike": { "aws:RequestTag/environment": "?*" }
      }
   }]
}
```

次のポリシーでは、リクエスタが Auto Scaling グループにタグ付けできるタグは **purpose=webserver** および **cost-center=cc123** であることを指定し、**purpose** タグおよび **cost-center** タグのみが許可されます (他のタグは指定できません)。

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
          "autoscaling:CreateAutoScalingGroup",
          "autoscaling:CreateOrUpdateTags"
      ],
      "Resource": "*",
      "Condition": {
          "StringEquals": {
              "aws:RequestTag/purpose": "webserver",
```

```
                "aws:RequestTag/cost-center": "cc123"
            },
            "ForAllValues:StringEquals": { "aws:TagKeys": ["purpose", "cost-center"] }
        }
    }]
}
```

次のポリシーでは、リクエスタがリクエストで少なくとも 1 つのタグを指定することを要求し、**cost-center** および **owner** キーのみ使用できます。

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "autoscaling:CreateAutoScalingGroup",
            "autoscaling:CreateOrUpdateTags"
        ],
        "Resource": "*",
        "Condition": {
            "ForAnyValue:StringEquals": { "aws:TagKeys": ["cost-center", "owner"] }
        }
    }]
}
```

> **Note**
>
> 条件においては、条件キーでは大文字と小文字が区別されず、条件値では大文字と小文字が区別されます。したがって、タグキーの大文字と小文字を区別するには、条件の値としてタグキーが指定される aws:TagKeys 条件キーを使用します。

削除できる Auto Scaling グループを制御する

次のポリシーは、グループにタグ **environment=development** が付けられている場合にのみ、Auto Scaling グループの削除を許可します。

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
```

```
      "Action": "autoscaling:DeleteAutoScalingGroup",
      "Resource": "*",
      "Condition": {
          "StringEquals": { "aws:ResourceTag/environment": "development" }
      }
   }]
}
```

Auto Scaling グループへのアクセス制御で条件キーを使用していない場合は、代わりに `Resource` 要素内のリソースの ARN を指定してアクセスを制御できます。

次のポリシーは、名前が **devteam-** で始まる Auto Scaling グループについてのみ、`DeleteAutoScalingGroup` API アクションを使用するための許可をユーザーに付与します。

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": "autoscaling:DeleteAutoScalingGroup",
      "Resource": "arn:aws:autoscaling:region:account-
id:autoScalingGroup:*:autoScalingGroupName/devteam-*"
   }]
}
```

複数の ARN をリストに含めて指定することもできます。UUID を含めることで、特定の Auto Scaling グループに確実にアクセス許可が付与されます。新しいグループの UUID は、削除された同じ名前のグループの UUID とは異なります。

```
"Resource": [
    "arn:aws:autoscaling:region:account-
id:autoScalingGroup:uuid:autoScalingGroupName/devteam-1",
    "arn:aws:autoscaling:region:account-
id:autoScalingGroup:uuid:autoScalingGroupName/devteam-2",
    "arn:aws:autoscaling:region:account-
id:autoScalingGroup:uuid:autoScalingGroupName/devteam-3"
]
```

削除できるスケーリングポリシーを制御する

次のポリシーでは、`DeletePolicy` アクションを使用してスケーリングポリシーを削除する許可が付与されます。ただし、処理対象の Auto Scaling グループに **environment=production** タグが

ある場合、そのアクションを拒否します。可能な限りタグを使用して、アカウント内の Auto Scaling グループへのアクセスを制御できるようにします。

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": "autoscaling:DeletePolicy",
      "Resource": "*"
   },
   {
      "Effect": "Deny",
      "Action": "autoscaling:DeletePolicy",
      "Resource": "*",
      "Condition": {
          "StringEquals": { "autoscaling:ResourceTag/environment": "production" }
      }
   }]
}
```

インスタンスの更新アクションへのアクセスを制御する

次のポリシーは、処理対象の Auto Scaling グループにタグ **environment=testing** が付けられている場合にのみ、インスタンスの更新を開始、ロールバック、キャンセルするアクセス許可を付与します。Describe アクションはリソースレベルの許可をサポートしないため、条件のない別のステートメントで指定する必要があります。

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
          "autoscaling:StartInstanceRefresh",
          "autoscaling:CancelInstanceRefresh",
          "autoscaling:RollbackInstanceRefresh"
      ],
      "Resource": "*",
      "Condition": {
          "StringEquals": { "autoscaling:ResourceTag/environment": "testing" }
      }
   },
   {
```

```
      "Effect": "Allow",
      "Action": "autoscaling:DescribeInstanceRefreshes",
      "Resource": "*"
   }]
}
```

StartInstanceRefresh 呼び出しで希望する設定を指定するには、次のような関連するアクセス許可が必要になる場合があります。

- ec2:RunInstances – 起動テンプレートを使用して EC2 インスタンスを起動するには、ユーザーは IAM ポリシーで アクセスec2:RunInstances許可を持っている必要があります。詳細については、「起動テンプレートのサポート」を参照してください。
- ec2:CreateTags – 作成時にインスタンスとボリュームにタグを追加する起動テンプレートから EC2 インスタンスを起動するには、ユーザーに IAM ポリシーの アクセスec2:CreateTags許可が必要です。詳細については、「インスタンスおよびボリュームにタグ付けするために必要なアクセス許可」を参照してください。
- iam:PassRole – インスタンスプロファイル (IAM ロールのコンテナ) を含む起動テンプレートから EC2 インスタンスを起動するには、IAM ポリシーの アクセスiam:PassRole許可も必要です。詳細および IAM ポリシーの例については、「Amazon EC2 インスタンスで実行中のアプリケーション用の IAM ロール」を参照してください。
- ssm:GetParameters – AWS Systems Manager パラメータを使用する起動テンプレートから EC2 インスタンスを起動するには、ユーザーは IAM ポリシーの アクセスssm:GetParameters許可も持っている必要があります。詳細については、「起動テンプレートで AMI IDs の代わりに AWS Systems Manager パラメータを使用する」を参照してください。

サービスにリンクされたロールの作成

Amazon EC2 Auto Scaling では、 のユーザーが Amazon EC2 Auto Scaling API アクションを初めて AWS アカウント 呼び出すときに、サービスにリンクされたロールを作成するためのアクセス許可が必要です。サービスにリンクされたロールがまだ存在しない場合は、Amazon EC2 Auto Scaling によってアカウント内に作成されます。サービスにリンクされたロールは、Amazon EC2 Auto Scaling がユーザーに代わって他の を呼び出すことができるように、Amazon EC2 Auto Scaling AWS のサービス にアクセス許可を付与します。

この自動ロール作成を成功させるには、ユーザーには iam:CreateServiceLinkedRole アクションへのアクセス許可が必要です。

```
"Action": "iam:CreateServiceLinkedRole"
```

以下は、Amazon EC2 Auto Scaling に対する Amazon EC2 Auto Scaling のサービスにリンクされたロールの作成をユーザーに許可するアクセス許可ポリシーの例です。

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": "iam:CreateServiceLinkedRole",
      "Resource": "arn:aws:iam::*:role/aws-service-role/
autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling",
      "Condition": {
          "StringLike": { "iam:AWSServiceName":"autoscaling.amazonaws.com" }
      }
   }]
}
```

どのサービスにリンクされたロールを渡すことができるかを制御する (を使用 PassRole)

Auto Scaling グループを作成または更新し、リクエストでカスタムサフィックスサービスにリンクされたロールを指定するユーザーは、`iam:PassRole` 許可が必要です。

異なるサービスにリンクされたロールに異なるキーへのアクセスを許可する場合は、 アクセス`iam:PassRole`許可を使用して AWS KMS カスタマーマネージドキーのセキュリティを保護できます。組織の必要に応じて、開発チームに 1 つの キー、QA チームにもう 1 つの キー、そして財務チームにもう 1 つの キー を持つことができます。まず、 という名前のサービスにリンクされたロールなど、必要なキーにアクセスできるサービスにリンクされたロールを作成しますAWSServiceRoleForAutoScaling_devteamkeyaccess。次に、ポリシーをユーザーまたはロールなどの IAM アイデンティティにアタッチします。

次のポリシーは、名前が **`devteam-`** で始まる Auto Scaling グループに **`AWSServiceRoleForAutoScaling_devteamkeyaccess`** ロールを渡すための許可を付与します。Auto Scaling グループを作成する IAM アイデンティティがサービスにリンクされた別のロールを指定しようとすると、エラーが表示されます。サービスにリンクされたロールを指定しない場合、代わりにデフォルトのAWSServiceRoleForAutoScalingロールが使用されます。

```
{
```

```
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::account-id:role/aws-service-role/
autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling_devteamkeyaccess",
      "Condition": {
          "StringEquals": { "iam:PassedToService": [ "autoscaling.amazonaws.com" ] },
          "StringLike": { "iam:AssociatedResourceARN":
 [ "arn:aws:autoscaling:region:account-
id:autoScalingGroup:*:autoScalingGroupName/devteam-*" ] }
      }
   }]
}
```

カスタムサフィックス付きのサービスにリンクされたロールの詳細については、「Amazon EC2 Auto Scaling のサービスにリンクされたロール」を参照してください。

サービス間の混乱した代理の防止

混乱した代理問題は、アクションを実行するためのアクセス許可を持たないエンティティが、より特権のあるエンティティにアクションの実行を強制できてしまう場合に生じる、セキュリティ上の問題です。

では AWS、サービス間のなりすましにより、混乱した代理問題が発生する可能性があります。サービス間でのなりすましは、1 つのサービス (呼び出し元サービス) が、別のサービス (呼び出し対象サービス) を呼び出すときに発生する可能性があります。呼び出し元サービスは、本来ならアクセスすることが許可されるべきではない方法でその許可を使用して、別のお客様のリソースに対する処理を実行するように操作される場合があります。

これを防ぐために、 は、アカウント内のリソースへのアクセスが許可されているサービスプリンシパルを持つすべてのサービスのデータを保護するのに役立つツール AWS を提供します。Amazon EC2 Auto Scaling サービスロールの信頼ポリシーでは、`aws:SourceArn` および `aws:SourceAccount` グローバル条件コンテキストキーを使用することをお勧めします。これらのキーは、Amazon EC2 Auto Scaling が他のサービスに付与するそのリソースへのアクセス許可を制限します。

`SourceArn` および `SourceAccount`フィールドの値は、Amazon EC2 Auto Scaling が AWS Security Token Service （ AWS STS) を使用してユーザーに代わってロールを引き受ける場合に設定されます。

aws:SourceArn または aws:SourceAccount グローバル条件キーを使用する場合、値を Amazon リソースネーム (ARN) または Amazon EC2 Auto Scaling が保存するリソースのアカウントに設定します。可能な限り、より具体的な aws:SourceArn を使用してください。ARN の不明な部分の値を ARN またはワイルドカード (*) を含む ARN パターンに設定します。リソースの ARN が不明の場合は、代わりに aws:SourceAccount を使用してください。

次の例では、Amazon EC2 Auto Scaling で aws:SourceArn および aws:SourceAccount グローバル条件コンテキストキーを使用して、「混乱した代理」問題を回避する方法を示します。

例: **aws:SourceArn** 条件キーおよび **aws:SourceAccount** 条件キー

サービスがお客様に代わってアクションを実行するために引き受けるロールは、サービスロールと呼ばれます。Amazon 以外の場所に通知を送信するライフサイクルフックを作成する場合は EventBridge、Amazon EC2 Auto Scaling がユーザーに代わって Amazon SNS トピックまたは Amazon SQS キューに通知を送信できるようにするサービスロールを作成する必要があります。クロスサービスアクセスに関連付ける Auto Scaling グループを 1 つだけにする場合は、サービスロールの信頼ポリシーを次のように指定できます。

この信頼ポリシーの例では、条件文を使用して、サービスロールの AssumeRole 機能を、指定されたアカウントの指定された Auto Scaling グループに影響を与えるアクションのみに制限します。aws:SourceArn および aws:SourceAccount の条件は個別に評価されます。サービスロールを使用するリクエストでは、両方の条件が満たされている必要があります。

このポリシーを使用する前に、リージョン、アカウント ID、UUID、およびグループ名をアカウントの有効な値に置き換えてください。

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
      "Service": "autoscaling.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "ArnLike": {
        "aws:SourceArn":
 "arn:aws:autoscaling:region:account_id:autoScalingGroup:uuid:autoScalingGroupName/my-
asg"
      },
```

```
      "StringEquals": {
        "aws:SourceAccount": "account_id"
      }
    }
  }
}
```

前の例では、以下のようになっています。

- `Principal` 要素は、サービス(`autoscaling.amazonaws.com`) のサービスプリンシパルを指定します。
- `Action` 要素は、`sts:AssumeRole` アクションを指定します。
- `Condition` 要素は、`aws:SourceArn` および `aws:SourceAccount` グローバル条件キーを指定します。ソースの ARN にはアカウント ID が含まれているため、`aws:SourceArn` で `aws:SourceAccount` を使用する必要はありません 。

追加情報

詳細については、「IAM ユーザーガイド」の「AWS グローバル条件コンテキストキー」、「混乱した使節の問題」、および「ロールの信頼ポリシーの変更 (コンソール)」を参照してください。

起動テンプレートのサポート

Amazon EC2 Auto Scaling は、Auto Scaling グループでの Amazon EC2 起動テンプレートの使用をサポートしています。起動テンプレートから Auto Scaling グループを作成することをユーザーに許可することをお勧めします。これにより、ユーザーは Amazon EC2 Auto Scaling と Amazon EC2 の最新機能を使用できます。例えば、ユーザーは混合インスタンスポリシーを使用するための起動テンプレートを指定する必要があります。

`AmazonEC2FullAccess` ポリシーを使用すると、アカウント内の Amazon EC2 Auto Scaling リソース、起動テンプレート、およびその他の EC2 リソースを使用するための完全なアクセス許可をユーザーに付与できます。または、このトピックで説明するように、独自のカスタム IAM ポリシーを作成して、起動テンプレートを使用するきめ細かなアクセス許可をユーザーに付与することもできます。

独自の用途に合わせてカスタマイズできるサンプルポリシー

次に、独自の用途に合わせてカスタマイズできる基本アクセス許可ポリシーの例を表示します。ポリシーは、グループが **`purpose=testing`** タグを使用している場合に限り、すべての Auto Scaling グ

ループを作成、更新、削除することを許可します。その後、すべての Describe アクションに許可を与えます。Describe アクションはリソースレベルの許可をサポートしないため、条件のない別のステートメントで指定する必要があります。

このポリシーを持つ IAM アイデンティティ (ユーザーまたはロール) は、ec2:RunInstances アクションを使用する許可も与えられているため、起動テンプレートを使用して Auto Scaling グループを作成または更新する許可があります。

```
{
    "Version": "2012-10-17",
    "Statement": [
          {
            "Effect": "Allow",
            "Action": [
                "autoscaling:CreateAutoScalingGroup",
                "autoscaling:UpdateAutoScalingGroup",
                "autoscaling:DeleteAutoScalingGroup"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": { "autoscaling:ResourceTag/purpose": "testing" }
            }
         },
         {
            "Effect": "Allow",
            "Action": [
                "autoscaling:Describe*",
                "ec2:RunInstances"
            ],
            "Resource": "*"
        }
    ]
}
```

Auto Scaling グループを作成または更新するユーザーには、関連する次のような権限が必要になる場合があります。

- ec2:CreateTags – 作成時にインスタンスとボリュームにタグを追加するには、ユーザーに IAM ポリシーの アクセスec2:CreateTags許可が必要です。詳細については、「インスタンスおよびボリュームにタグ付けするために必要なアクセス許可」を参照してください。

- iam:PassRole – インスタンスプロファイル (IAM ロールのコンテナ) を含む起動テンプレートから EC2 インスタンスを起動するには、IAM ポリシーの アクセス`iam:PassRole`許可も必要です。詳細および IAM ポリシーの例については、「Amazon EC2 インスタンスで実行中のアプリケーション用の IAM ロール」を参照してください。
- ssm:GetParameters – AWS Systems Manager パラメータを使用する起動テンプレートから EC2 インスタンスを起動するには、ユーザーは IAM ポリシーの アクセス`ssm:GetParameters`許可も持っている必要があります。詳細については、「起動テンプレートで AMI IDs の代わりに AWS Systems Manager パラメータを使用する」を参照してください。

インスタンス起動時に完了するアクションに対するこれらのアクセス許可は、ユーザーが Auto Scaling グループを操作するときにチェックされます。詳細については、「`ec2:RunInstances` と `iam:PassRole` の許可の検証」を参照してください。

次の例では、IAM ユーザーが起動テンプレートを使用するためのアクセスを制御できる、ポリシーステートメントを示しています。

トピック

- 特定のタグを持つ起動テンプレートを要求する
- 起動テンプレートとバージョン番号を要求する
- インスタンスメタデータサービスバージョン 2 (IMDSv2) の使用を要求する
- Amazon EC2 リソースへのアクセスを制限する
- インスタンスおよびボリュームにタグ付けするために必要なアクセス許可
- 起動テンプレートの追加アクセス許可
- ec2:RunInstances と iam:PassRole の許可の検証
- 関連リソース

特定のタグを持つ起動テンプレートを要求する

`ec2:RunInstances` アクセス許可を付与する場合、ユーザーが特定のタグまたは特定の ID を持つ起動テンプレートのみを使用できるように指定すると、起動テンプレートを使用してインスタンスを起動する際にアクセス許可を制限できます。また、`RunInstances` 呼び出しに対するリソースレベルの追加のアクセス許可を指定することで、起動テンプレートを使用するすべてのユーザーがインスタンスの起動時に参照および使用できる AMI やその他のリソースを制御できます。

次の例では、指定されたリージョンにある起動テンプレートを持ち、タグ **`purpose=testing`** を持つ `ec2:RunInstances` アクションへのアクセス許可を制限します。また、ユーザーは起動テンプ

レートで指定されているリソース (AMI、インスタンスタイプ、ボリューム、キーペア、ネットワークインターフェイス、セキュリティグループ) にもアクセスできるようになります。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": "arn:aws:ec2:region:account-id:launch-template/*",
            "Condition": {
                "StringEquals": { "aws:ResourceTag/purpose": "testing" }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:region::image/ami-*",
                "arn:aws:ec2:region:account-id:instance/*",
                "arn:aws:ec2:region:account-id:subnet/*",
                "arn:aws:ec2:region:account-id:volume/*",
                "arn:aws:ec2:region:account-id:key-pair/*",
                "arn:aws:ec2:region:account-id:network-interface/*",
                "arn:aws:ec2:region:account-id:security-group/*"
            ]
        }
    ]
}
```

起動テンプレートでタグベースのポリシーを使用する方法の詳細については、「Amazon EC2 ユーザーガイド」の「IAM アクセス許可を使用して起動テンプレートへのアクセスを制御する」を参照してください。 Amazon EC2

起動テンプレートとバージョン番号を要求する

IAM のアクセス許可を使用して、Auto Scaling グループを作成または更新するときに、起動テンプレートと、起動テンプレートのバージョン番号を指定するように強制することもできます。

次の例では、起動テンプレートおよび起動テンプレートのバージョン番号が指定されている場合にのみ、ユーザーが Auto Scaling グループを作成および更新できるようにします。このポリシーを持つユーザーが、`$Latest` または `$Default` の起動テンプレートのバージョンを指定するためのバー

ジョン番号を省略したり、代わりに起動設定を使用しようとしたりすると、アクションは失敗します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "autoscaling:CreateAutoScalingGroup",
                "autoscaling:UpdateAutoScalingGroup"
            ],
            "Resource": "*",
            "Condition": {
                "Bool": { "autoscaling:LaunchTemplateVersionSpecified": "true" }
            }
        },
        {
            "Effect": "Deny",
            "Action": [
                "autoscaling:CreateAutoScalingGroup",
                "autoscaling:UpdateAutoScalingGroup"
            ],
            "Resource": "*",
            "Condition": {
                "Null": { "autoscaling:LaunchConfigurationName": "false" }
            }
        }
    ]
}
```

インスタンスメタデータサービスバージョン 2 (IMDSv2) の使用を要求する

セキュリティを強化するために、IMDSv2 を必要とする起動テンプレートの使用を要求するようにユーザーのアクセス許可を設定できます。詳細については、「Amazon EC2 ユーザーガイド」の「インスタンスメタデータサービスの設定」を参照してください。 Amazon EC2

次のポリシーでは、IMDSv2 の使用を要求するようにインスタンスもオプトインされていない限り (`"ec2:MetadataHttpTokens":"required"` で指定)、ユーザーは `ec2:RunInstances` アクションを呼び出せないことを示しています。

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RequireImdsV2",
            "Effect": "Deny",
            "Action": "ec2:RunInstances",
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringNotEquals": { "ec2:MetadataHttpTokens": "required" }
            }
        }
    ]
}
```

> **Tip**
>
> インスタンスのメタデータオプションが設定された、新しい起動テンプレートまたは新しいバージョンの起動テンプレートを使用している、代替の Auto Scaling インスタンスを強制的に起動することで、インスタンスの更新を開始できます。詳細については、「Auto Scaling インスタンスの更新」を参照してください。

Amazon EC2 リソースへのアクセスを制限する

次の例では、Amazon EC2 リソースへのアクセスを制限してユーザーが起動できるインスタンスの設定を制御します。起動テンプレートで指定されたリソースにリソースレベルのアクセス許可を指定するには、RunInstances アクションステートメントにそのリソースを含める必要があります

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:region:account-id:launch-template/*",
                "arn:aws:ec2:region::image/ami-04d5cc9b88example",
                "arn:aws:ec2:region:account-id:subnet/subnet-1a2b3c4d",
                "arn:aws:ec2:region:account-id:volume/*",
                "arn:aws:ec2:region:account-id:key-pair/*",
                "arn:aws:ec2:region:account-id:network-interface/*",
```

```
                "arn:aws:ec2:region:account-id:security-group/sg-903004f88example"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": "arn:aws:ec2:region:account-id:instance/*",
            "Condition": {
                "StringEquals": { "ec2:InstanceType": ["t2.micro", "t2.small"] }
            }
        }
    ]
}
```

この例には、 2 つのステートメントがあります。

- 1 番目のステートメントは、ユーザーが、特定のセキュリティグループ (**sg-903004f88example**) と、特定の AMI (**ami-04d5cc9b88example**) を使用して、特定のサブネット (**subnet-1a2b3c4d**) でインスタンスを起動することを要求します。また、ユーザーは起動テンプレートで指定されているリソース (ネットワークインターフェイス、キーペア、ボリューム) にもアクセスできるようになります。
- 2 番目のステートメントでは、ユーザーは **t2.micro** または **t2.small** インスタンスタイプのみを使用してインスタンスを起動でき、コストを管理できます。

  ただし、起動テンプレートを使用してインスタンスを起動するアクセス許可を持つユーザーが他のインスタンスタイプを起動することを完全に防ぐ有効な方法は、現在ないことに注意してください。これは、起動テンプレートで指定されたインスタンスタイプを上書きして、属性ベースのインスタンスタイプの選択を使用して定義されたインスタンスタイプを使用できるためです。

ユーザーが起動するインスタンスの設定の制御に使用できるリソースレベルのアクセス許可の完全なリストについては、「サービス認可リファレンス」の「Amazon EC2 のアクション、リソース、および条件キー」を参照してください。

インスタンスおよびボリュームにタグ付けするために必要なアクセス許可

次の例では、インスタンスとボリュームの作成時に、以下のタグを付けることをユーザーに許可します。このポリシーは、起動テンプレートにタグが指定されている場合に必要です。詳細については、「Amazon EC2 ユーザーガイド」の「作成時にリソースにタグを付けるアクセス許可を付与する」を参照してください。 Amazon EC2

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:region:account-id:*/*",
            "Condition": {
                "StringEquals": { "ec2:CreateAction": "RunInstances" }
            }
        }
    ]
}
```

起動テンプレートの追加アクセス許可

コンソールユーザーには、ec2:DescribeLaunchTemplatesおよびec2:DescribeLaunchTemplateVersionsアクションへのアクセス許可を与える必要があります。これらのアクセス許可なしでは、起動テンプレートとネットワークオプションが Auto Scaling グループウィザードに読み込まれないため、ユーザーはウィザードをステップ実行できず、起動テンプレートを使用してインスタンスを起動することができません。これらの追加のアクションは、IAM ポリシーステートメントの Action 要素で指定できます。

ec2:RunInstances と iam:PassRole の許可の検証

ユーザーは、Auto Scaling グループが使用する起動テンプレートのバージョンを指定できます。許可に応じて、これは特定の番号付きバージョン、または起動テンプレートの $Latest もしくは $Default バージョンになります。後者の場合は特に注意してください。これにより、制限することを意図していた ec2:RunInstances および iam:PassRole の許可がオーバーライドされる可能性があります。

このセクションでは、Auto Scaling グループで最新またはデフォルトバージョンの起動テンプレートを使用するシナリオについて説明します。

ユーザーが CreateAutoScalingGroup、UpdateAutoScalingGroup、または StartInstanceRefresh API を呼び出すと、Amazon EC2 Auto Scaling は、リクエストを続行する前に、その時点での最新またはデフォルトのバージョンである起動テンプレートのバージョンに照らしてユーザーの許可をチェックします。これにより、インスタンスの起動時に完了するアクション(ec2:RunInstances や iam:PassRole アクションなど) の許可が検証されます。これを実現するために、Amazon EC2 のRunInstancesドライランコールを発行して、ユーザーが実際にリクエス

トを行わずに、アクションに必要なアクセス許可を持っているかどうかを検証します。レスポンスが返されると、Amazon EC2 Auto Scaling がそのレスポンスを読み取ります。ユーザーの許可が所定のアクションが許可しない場合、Amazon EC2 Auto Scaling はリクエストを失敗させ、欠落している許可に関する情報が含まれたエラーをユーザーに返します。

初期検証とリクエストが完了すると、インスタンスが起動するたびに、Amazon EC2 Auto Scaling は、サービスにリンクされたロールの許可を使用して、変更されている場合でも最新またはデフォルトのバージョンを使用してインスタンスを起動します。つまり、起動テンプレートを使用しているユーザは、`iam:PassRole` アクセス許可を持っていなくても、インスタンスに IAM ロールを渡すように更新できる可能性があります。

`$Latest` または `$Default` バージョンを使用するようにグループを設定するアクセス許可を持つ人を制限したい場合は、`autoscaling:LaunchTemplateVersionSpecified` 条件キーを使用します。これにより、ユーザーが `CreateAutoScalingGroup` および `UpdateAutoScalingGroup` API を呼び出す際に、Auto Scaling グループが特定の番号付きバージョンのみを受け入れることが保証されます。この条件キーを IAM ポリシーに追加する方法を示す例については、「起動テンプレートとバージョン番号を要求する」を参照してください。

`$Latest` または `$Default` 起動テンプレートのバージョンを使用するように構成されているAuto Scalingグループについては、デフォルトの起動テンプレートのバージョンをユーザーが指定できる `ec2:ModifyLaunchTemplate` アクションを含め、起動テンプレートのバージョンを作成および管理できるユーザーを制限することを検討してください。詳細については、Amazon EC2 ユーザーガイド」の「バージョニング許可の制御」を参照してください。

関連リソース

起動テンプレートと起動テンプレートのバージョンを表示、作成、削除するアクセス許可の詳細については、「Amazon EC2 ユーザーガイド」の「IAM アクセス許可を使用して起動テンプレートへのアクセスを制御する」を参照してください。 Amazon EC2

`RunInstances` 呼び出しへのアクセスを制御するために使用できるリソースレベルのアクセス許可の詳細については、「サービス認可リファレンス」の「Amazon EC2 のアクション、リソース、および条件キー」を参照してください。

Amazon EC2 インスタンスで実行中のアプリケーション用の IAM ロール

Amazon EC2 インスタンスで実行されるアプリケーションには、他の AWS のサービスにアクセスするための認証情報が必要です。これらの認証情報を安全な方法で提供するには、IAM ロールを使用します。このロールは、アプリケーションが他の AWS リソースにアクセスするときに使用できる一

時的なアクセス許可を付与します。アプリケーションに許可される操作は、ロールのアクセス許可で決定されます。

Auto Scaling グループ内のインスタンス用に、起動テンプレートまたは起動設定を作成し、インスタンスに関連付けるインスタンスプロファイルを選択する必要があります。インスタンスプロファイルは IAM ロールのコンテナであり、インスタンスの起動時に Amazon EC2 インスタンスに IAM ロール情報を渡すために使用できます。まず、 AWS リソースへのアクセスに必要なすべてのアクセス許可を持つ IAM ロールを作成します。次に、インスタンスプロファイルを作成し、そのプロファイルにロールを割り当てます。

> **Note**
>
> ベストプラクティスとして、アプリケーション AWS のサービス が必要とする他の に対する最小限のアクセス許可を持つように、ロールを作成することを強くお勧めします。

内容

- 前提条件
- 起動テンプレートの作成
- 以下も参照してください。

前提条件

Amazon EC2 で実行されているアプリケーションが引き受けることができる IAM ロールを作成します。結果としてロールを引き受けたアプリケーションが必要な特定の API 呼び出しを実行できるように、適切なアクセス許可を選択します。

AWS CLI または AWS SDKsコンソールを使用する場合、コンソールは自動的にインスタンスプロファイルを作成し、対応するロールと同じ名前を付けます。

IAM ロールを作成するには (コンソール)

1. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
2. 左側のナビゲーションペインで、[Roles] を選択します。
3. [ロールの作成] を選択します。
4. [Select trusted entity] (信頼されたエンティティの選択) で、[AWS のサービス] を選択します。
5. ユースケースに [EC2] を選択してから、[Next] (次へ) を選択します。

6. 可能な場合は、アクセス許可ポリシーとして使用するポリシーを選択するか、[ポリシーの作成] を選択して新しいブラウザタブを開き、新しいポリシーをゼロから作成します。詳細については、『IAM ユーザーガイド』の「IAM ポリシーの作成」を参照してください。ポリシーを作成したら、そのタブを閉じて元のタブに戻ります。サービスに割り当てるアクセス許可ポリシーの横のチェックボックスをオンにします。
7. (オプション) アクセス許可の境界を設定します。これは、サービスロールに利用できる高度な機能です。詳細については、「IAM ユーザーガイド」の「IAM エンティティのアクセス許可境界」を参照してください。
8. [次へ] をクリックします。
9. [Name, review, and create] (名前、確認、および作成) ページの [Role name] (ロール名) に、このロールの目的を識別するために役立つロール名を入力します。この名前は AWS アカウント内で一意である必要があります。他の AWS リソースはロールを参照する可能性があるため、ロールの作成後にロールの名前を編集することはできません。
10. ロールを確認したら、[Create role] (ロールを作成) を選択します。

IAM アクセス許可

IAM アイデンティティーベースのポリシーを使用し、新しい IAM ロールへのアクセスを制御します。`iam:PassRole` 許可は、インスタンスプロファイルを指定する起動テンプレートを使用して Auto Scaling グループを作成または更新する IAM ID (ユーザーまたはロール) で必要です。

次のポリシーの例では、名前が **`qateam-`** で始まる IAM ロールのみを渡す許可を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::account-id:role/qateam-*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com",
                        "ec2.amazonaws.com.cn"
                    ]
                }
            }
        }
```

```
    ]
}
```

> **Important**
>
> 起動テンプレートを使用する Auto Scaling グループの `iam:PassRole` アクションの許可を Amazon EC2 Auto Scaling が検証する方法については、「ec2:RunInstances と iam:PassRole の許可の検証」を参照してください。

起動テンプレートの作成

を使用して起動テンプレートを作成する場合 AWS Management Console、詳細セクションで、IAM インスタンスプロファイル からロールを選択します。詳細については、「詳細設定を使用して起動テンプレートを作成する」を参照してください。

から create-launch-template コマンドを使用して起動テンプレートを作成するときは AWS CLI、次の例に示すように、IAM ロールのインスタンスプロファイル名を指定します。

```
aws ec2 create-launch-template --launch-template-name my-lt-with-instance-profile --
version-description version1 \
--launch-template-data
 '{"ImageId":"ami-04d5cc9b88example","InstanceType":"t2.micro","IamInstanceProfile":
{"Name":"my-instance-profile"}}'
```

以下も参照してください。

Amazon EC2 の IAM ロールについて学習し、使用する方法の詳細については、「」を参照してください。

- Amazon EC2 ユーザーガイド」の「Amazon EC2 の IAM ロールAmazon EC2」
- IAM ユーザーガイドのインスタンスプロファイルの使用およびIAM ロールを使用して、Amazon EC2 インスタンスで実行されるアプリケーションにアクセス許可を付与する

Amazon EC2 Auto Scaling のコンプライアンス検証

AWS のサービス が特定のコンプライアンスプログラムの範囲内にあるかどうかを確認するには、コンプライアンスプログラムAWS のサービス による対象範囲内のコンプライアンスプログラムを参照

し、関心のあるコンプライアンスプログラムを選択します。一般的な情報については、AWS「コンプライアンスプログラム」を参照してください。

を使用して、サードパーティーの監査レポートをダウンロードできます AWS Artifact。詳細については、「 でのレポートのダウンロード AWS Artifact」の」を参照してください。

を使用する際のお客様のコンプライアンス責任 AWS のサービス は、お客様のデータの機密性、貴社のコンプライアンス目的、適用される法律および規制によって決まります。 では、コンプライアンスに役立つ以下のリソース AWS を提供しています。

- セキュリティとコンプライアンスのクイックスタートガイド – これらのデプロイガイドでは、アーキテクチャ上の考慮事項について説明し、セキュリティとコンプライアンスに重点を置いたベースライン環境 AWS を にデプロイする手順について説明します。
- アマゾン ウェブ サービスにおける HIPAA セキュリティとコンプライアンスのアーキテクチャ — このホワイトペーパーでは、企業が AWS を使用して HIPAA 対象アプリケーションを作成する方法について説明します。

> **Note**
>
> すべて AWS のサービス HIPAA の対象となるわけではありません。詳細については、「HIPAA 対応サービスのリファレンス」を参照してください。

- AWS コンプライアンスリソース – このワークブックとガイドのコレクションは、お客様の業界や地域に適用される場合があります。
- AWS カスタマーコンプライアンスガイド — コンプライアンスの観点から責任共有モデルを理解します。このガイドでは、ガイダンスを保護し AWS のサービス 、複数のフレームワーク (米国国立標準技術研究所 (NIST)、Payment Card Industry Security Standards Council (PCI)、国際標準化機構 (ISO) を含む) のセキュリティコントロールにマッピングするためのベストプラクティスをまとめています。
- 「 デベロッパーガイド」の「 ルールによるリソースの評価」 – この AWS Config サービスは、リソース設定が社内プラクティス、業界ガイドライン、および規制にどの程度準拠しているかを評価します。 AWS Config
- AWS Security Hub – これにより AWS のサービス 、 内のセキュリティ状態を包括的に確認できます AWS。Security Hub では、セキュリティコントロールを使用して AWS リソースを評価し、セキュリティ業界標準とベストプラクティスに対するコンプライアンスをチェックします。サポートされているサービスとコントロールのリストについては、「Security Hub のコントロールリファレンス」を参照してください。

- Amazon GuardDuty – これにより AWS アカウント、疑わしいアクティビティや悪意のあるアクティビティがないか環境を監視することで、、、ワークロード、コンテナ、データに対する潜在的な脅威 AWS のサービス を検出します。 GuardDuty は、特定のコンプライアンスフレームワークで義務付けられている侵入検知要件を満たすことで、PCI DSS などのさまざまなコンプライアンス要件への対応に役立ちます。
- AWS Audit Manager – これにより AWS のサービス 、 AWS 使用状況を継続的に監査し、リスクの管理方法と規制や業界標準への準拠を簡素化できます。

PCI DSS コンプライアンス

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

AWS ワークロードの PCI DSS コンプライアンスの達成については、次のコンプライアンスガイドを参照してください。

- での Payment Card Industry Data Security Standard (PCI DSS) 3.2.1 AWS

Amazon EC2 Auto Scaling エンドポイントとインターフェイス VPC エンドポイント

インターフェイス VPC エンドポイントを使用するように Amazon EC2 Auto Scaling を設定することで、VPC のセキュリティ体制を強化できます。インターフェイスエンドポイントは AWS PrivateLink、VPC と Amazon EC2 Auto Scaling 間のネットワークへのすべてのネットワークトラフィックを制限することで、Amazon EC2 Auto Scaling API にプライベートにアクセスできるようにするテクノロジーを利用しています。 AWS インターフェイスエンドポイントでは、インターネットゲートウェイ、NAT デバイス、または仮想プライベートゲートウェイも必要ありません。

設定は必須ではありませんが AWS PrivateLink、設定することをおすすめします。VPC エンドポイントの詳細については AWS PrivateLink 、「What is?」を参照してください。 AWS PrivateLinkガイドのAWS PrivateLink

トピック

- インターフェイス VPC エンドポイントを作成する

- VPC エンドポイントポリシーを作成する

インターフェイス VPC エンドポイントを作成する

以下のサービス名を使用して、Amazon EC2 Auto Scaling のエンドポイントを作成します。

```
com.amazonaws.region.autoscaling
```

詳細については、AWS PrivateLink ガイドの「インターフェイス VPC AWS エンドポイントを使用してサービスにアクセスする」を参照してください。

Amazon EC2 Auto Scaling 設定を変更する必要はありません。Amazon EC2 Auto Scaling は、使用中のサービスエンドポイントまたはプライベートインターフェイス VPC AWS エンドポイントのいずれかを使用して他のサービスを呼び出します。

VPC エンドポイントポリシーを作成する

Amazon EC2 Auto Scaling API へのアクセスをコントロールするために、VPC エンドポイントにポリシーをアタッチすることができます。このポリシーでは以下の内容を指定します。

- アクションを実行できるプリンシパル。
- 実行可能なアクション。
- このアクションを実行できるリソース。

以下の例では、エンドポイントを介してスケーリングポリシーを削除するためのアクセス許可を全員に対して拒否する VPC エンドポイントポリシーを示しています。このポリシー例では、他のすべてのアクションを実行するアクセス許可も全員に付与しています。

```
{
   "Statement": [
        {
            "Action": "*",
            "Effect": "Allow",
            "Resource": "*",
            "Principal": "*"
        },
        {
            "Action": "autoscaling:DeleteScalingPolicy",
            "Effect": "Deny",
```

```
            "Resource": "*",
            "Principal": "*"
        }
    ]
}
```

詳細については、『AWS PrivateLink ガイド』の「Control access to VPC endpoints using endpoint policies (エンドポイントポリシーを使用して VPC エンドポイントへのアクセスをコントロールする)」を参照してください。

Amazon EC2 Auto Scaling をトラブルシューティングする

Amazon EC2 Auto Scaling では、固有かつ説明的なエラー情報を出力するので、問題のトラブルシューティングに役立てられます。エラーメッセージは規模の拡大や縮小の説明から取得できます。

トピック

- スケーリングアクティビティからのエラーメッセージを取得する
- スケーリングアクティビティをオフにする
- その他のトラブルシューティングリソース
- Amazon EC2 Auto Scaling をトラブルシューティングする: EC2 インスタンス起動の失敗
- Amazon EC2 Auto Scaling をトラブルシューティングする: AMI 問題
- Amazon EC2 Auto Scaling をトラブルシューティングする: ロードバランサー問題
- Amazon EC2 Auto Scaling をトラブルシューティングする: 起動テンプレート

スケーリングアクティビティからのエラーメッセージを取得する

スケーリングのアクティビティの説明からエラーメッセージを取得するには、describe-scaling-activities コマンドを使用します。スケーリングについては、6 週間前からの記録が保存されています。このためのリストでは、最新のスケーリングを最上位に、スケーリングが開始時刻順に並べられます。

> **Note**
>
> スケーリングアクティビティは、Amazon EC2 Auto Scaling コンソールの Auto Scaling グループ用の [Activity] (アクティビティ) タブにあるアクティビティ履歴にも表示されています。

特定の Auto Scaling グループのスケーリングを表示するには、次のコマンドを使用します。

```
aws autoscaling describe-scaling-activities --auto-scaling-group-name my-asg
```

以下は、StatusCode で現在のスケーリングのステータスを示し、StatusMessage でエラーメッセージを示している応答の例です。

```
{
```

```
    "Activities": [
        {
            "ActivityId": "3b05dbf6-037c-b92f-133f-38275269dc0f",
            "AutoScalingGroupName": "my-asg",
            "Description": "Launching a new EC2 instance: i-003a5b3ffe1e9358e.  Status
 Reason: Instance failed to complete user's Lifecycle Action: Lifecycle Action with
 token e85eb647-4fe0-4909-b341-a6c42d8aba1f was abandoned: Lifecycle Action Completed
 with ABANDON Result",
            "Cause": "At 2021-01-11T00:35:52Z a user request created an
 AutoScalingGroup changing the desired capacity from 0 to 1.  At 2021-01-11T00:35:53Z
 an instance was started in response to a difference between desired and actual
 capacity, increasing the capacity from 0 to 1.",
            "StartTime": "2021-01-11T00:35:55.542Z",
            "EndTime": "2021-01-11T01:06:31Z",
            "StatusCode": "Cancelled",
            "StatusMessage": "Instance failed to complete user's Lifecycle Action:
 Lifecycle Action with token e85eb647-4fe0-4909-b341-a6c42d8aba1f was abandoned:
 Lifecycle Action Completed with ABANDON Result",
            "Progress": 100,
            "Details": "{\"Subnet ID\":\"subnet-5ea0c127\",\"Availability Zone\":\"us-
west-2b\"...}",
            "AutoScalingGroupARN": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:283179a2-
f3ce-423d-93f6-66bb518232f7:autoScalingGroupName/my-asg"
        },
     ...
    ]
}
```

出力の各フィールドについては、「Amazon EC2 Auto Scaling API Reference」(Amazon EC2 Auto Scaling API リファレンス) の「Activity」(アクティビティ) を参照してください。

削除されたグループのスケーリングアクティビティを表示するには

Auto Scaling グループが削除された後にスケーリングのアクティビティを表示するには、次のように describe-scaling-activities コマンドに `--include-deleted-groups` オプションを追加します。

```
aws autoscaling describe-scaling-activities --auto-scaling-group-name my-asg --include-
deleted-groups
```

下記は、削除されたグループのスケーリングに関する応答の例です。

```
{
```

```
    "Activities": [
        {
            "ActivityId": "e1f5de0e-f93e-1417-34ac-092a76fba220",
            "AutoScalingGroupName": "my-asg",
            "Description": "Launching a new EC2 instance.  Status Reason: Your Spot
 request price of 0.001 is lower than the minimum required Spot request fulfillment
 price of 0.0031. Launching EC2 instance failed.",
            "Cause": "At 2021-01-13T20:47:24Z a user request update of AutoScalingGroup
 constraints to min: 1, max: 5, desired: 3 changing the desired capacity from 0 to 3.
  At 2021-01-13T20:47:27Z an instance was started in response to a difference between
 desired and actual capacity, increasing the capacity from 0 to 3.",
            "StartTime": "2021-01-13T20:47:30.094Z",
            "EndTime": "2021-01-13T20:47:30Z",
            "StatusCode": "Failed",
            "StatusMessage": "Your Spot request price of 0.001 is lower than the
 minimum required Spot request fulfillment price of 0.0031. Launching EC2 instance
 failed.",
            "Progress": 100,
            "Details": "{\"Subnet ID\":\"subnet-5ea0c127\",\"Availability Zone\":\"us-
west-2b\"...}",
            "AutoScalingGroupState": "Deleted",
            "AutoScalingGroupARN": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:283179a2-
f3ce-423d-93f6-66bb518232f7:autoScalingGroupName/my-asg"
        },
     ...
    ]
}
```

スケーリングアクティビティをオフにする

スケーリングポリシーやスケジュールされたアクションによる干渉なしに問題を調査する必要がある場合は、次のオプションがあります。

- および ScheduledActionsプロセスを一時停止することで、すべての動的スケーリングポリシーAlarmNotificationとスケジュールされたアクションがグループの希望する容量を変更しないようにします。詳細については、「Amazon EC2 Auto Scaling プロセスの一時停止と再開」を参照してください。
- 個々の動的スケーリングポリシーを無効にして、負荷の変化に応じてグループの希望する容量を変更しないようにします。詳細については、「Auto Scaling グループのスケーリングポリシーを無効化する」を参照してください。

- 個々のターゲット追跡スケーリングポリシーを更新して、ポリシーのスケールイン部分を無効にしてスケールアウト (容量を追加) のみにします。この方法では、グループの希望する容量が縮小するのを防ぎますが、負荷が増加すると増やすことができます。詳細については、「Amazon EC2 Auto Scaling のターゲットトラッキングスケーリングポリシー」を参照してください。
- 予測スケーリングポリシーを予測専用モードに更新します。予測専用モードでは、予測スケーリングは引き続き予測を生成しますが、キャパシティーは自動的に増加しません。詳細については、「予測スケーリングポリシーを作成する」を参照してください。

その他のトラブルシューティングリソース

以下のページには、Amazon EC2 Auto Scaling での問題のトラブルシューティングに関する追加の情報が記載されています。

- Auto Scaling グループのスケーリングアクティビティを検証する
- Amazon EC2 Auto Scaling コンソールでモニタリンググラフを表示する
- Auto Scaling グループ内のインスタンスのヘルスチェック
- ライフサイクルフックの考慮事項と制限
- ライフサイクルアクションを完了する
- Amazon VPC を使用して Auto Scaling インスタンスにネットワーク接続を提供する
- Auto Scaling グループからインスタンスを一時的に削除する
- Auto Scaling グループのスケーリングポリシーを無効化する
- Amazon EC2 Auto Scaling プロセスの一時停止と再開
- スケールイン中に終了する Auto Scaling インスタンスを制御する
- Auto Scaling インフラストラクチャを削除する
- Auto Scaling リソースとグループのクォータ

以下の AWS リソースも役立ちます。

- AWS ナレッジセンターの Amazon EC2 Auto Scaling トピック
- AWS re:Post に関する Amazon EC2 Auto Scaling の質問
- コンピューティングブログの Amazon EC2 Auto Scaling AWS の投稿
- AWS CloudFormation ユーザーガイド CloudFormation のトラブルシューティング

トラブルシューティングには、エキスパート、またはヘルパーコミュニティによる反復的な照会と検出が必要になることがよくあります。このセクションの提案を試しても問題が解決しない場合は、に問い合わせるか AWS Support (で Support AWS Management Console、 Support Center をクリックします)、Amazon EC2 Auto Scaling タグを使用して AWS re:Post で質問してください。

Amazon EC2 Auto Scaling をトラブルシューティングする: EC2 インスタンス起動の失敗

このページでは、起動に失敗する EC2 インスタンスに関する情報、考えられる原因、問題を解決するために実行できるステップを提供します。

エラーメッセージを取得するには、「スケーリングアクティビティからのエラーメッセージを取得する」を参照してください。

EC2 インスタンスが起動に失敗する場合、以下のエラーメッセージが 1 つ以上表示される可能性があります。

起動に関する問題

- リクエストされた設定は現在サポートされていません。
- セキュリティグループ <セキュリティグループ名> は存在しません。EC2 インスタンスの起動に失敗しました。
- キーペア <お使いの EC2 インスタンスに関連付けられているキーペア> は存在しません。EC2 インスタンスの起動に失敗しました。
- リクエストされたインスタンスタイプ (<インスタンスタイプ>) は、リクエストされたアベイラビリティーゾーン (<インスタンスのアベイラビリティーゾーン>) ではサポートされません。
- 設定したポットリクエスト料金 0.015 は、スポットリクエストに最低限必要な料金 0.0735 を下回っています...。
- デバイス名 <device name> が無効です/無効なデバイス名をアップロードしようとしています。EC2 インスタンスの起動に失敗しました。
- パラメータ virtualName の値 (<インスタンスストレージのデバイスに関連付けられている名前>) は無効です。EC2 インスタンスの起動に失敗しました。
- EBS ブロックデバイスマッピングはインスタンスストア AMI ではサポートされません。
- プレイスメントグループは、「<instance type>」タイプのインスタンスでは使用できません。EC2 インスタンスの起動に失敗しました。

- Client.InternalError: 起動時のクライアントエラー。
- 現在、お客様がリクエストしたアベイラビリティゾーン内には、<instance type> の十分なキャパシティーがありません..。EC2 インスタンスの起動に失敗しました。
- リクエストされた予約には、このリクエストに十分な互換性と使用可能な容量がありません。EC2 インスタンスの起動に失敗しました。
- キャパシティブロック予約 <予約 ID> はまだアクティブではありません。EC2 インスタンスの起動に失敗しました。
- リクエストに適合した、使用可能なスポットキャパシティーはありません。EC2 インスタンスの起動に失敗しました。
- <インスタンス数> 個のインスタンスがすでに実行中です。EC2 インスタンスの起動に失敗しました。

リクエストされた設定は現在サポートされていません。

原因 : 起動テンプレートまたは起動設定の一部のオプションがインスタンスタイプと互換性がない、またはインスタンス設定がリクエストされた AWS リージョンまたはアベイラビリティーゾーンでサポートされていない可能性があります。

解決策 : 別のインスタンス設定を試してください。要件を満たすインスタンスタイプを検索するには、Amazon EC2 ユーザーガイド」の「Amazon EC2 インスタンスタイプの検索」を参照してください。 Amazon EC2

この問題の詳細な解決方法については、下記を確認してください。

- 選択した AMI が、インスタンスタイプでサポートされたものであることを確認します。例えば、インスタンスタイプがインテル Xeon プロセッサではなく Arm ベースの AWS Graviton プロセッサを使用している場合、Arm 互換 AMI が必要です。互換性のあるインスタンスタイプの選択の詳細については、「Amazon EC2 ユーザーガイド」の「インスタンスタイプを変更するための互換性」を参照してください。 Amazon EC2
- リクエストしたリージョンとアベイラビリティーゾーンの中で、インスタンスタイプが利用可能であるかをテストします。最新世代のインスタンスタイプには、特定のリージョンまたはアベイラビリティーゾーンではまだ利用できないものがあります。古い世代のインスタンスタイプについては、新しいリージョンとアベイラビリティーゾーンで利用できない場合があります。場所 (リージョンまたはアベイラビリティーゾーン) ごとに、利用可能なインスタンスタイプを検索するには、describe-instance-type-offerings コマンドを使用します。詳細については、Amazon EC2 ユーザーガイド」の「Amazon EC2 インスタンスタイプの検索」を参照してください。 Amazon EC2

- ハードウェア専有インスタンスまたは Dedicated Hosts を使用する場合は、ハードウェア専有インスタンスまたは Dedicated Hosts としてサポートされているインスタンスタイプを選択する必要があります。

セキュリティグループ <セキュリティグループ名> は存在しません。EC2 インスタンスの起動に失敗しました。

原因: 起動テンプレートもしくは起動設定で指定されたセキュリティグループが、削除されている可能性があります。

解決策:

1. describe-security-groups コマンドを使用して、アカウントに関連付けられているセキュリティグループのリストを取得します。
2. リストから、使用するセキュリティグループを選択します。代わりに、セキュリティグループを作成するには、create-security-group コマンドを使用します。
3. 新しい起動テンプレート、または起動設定を作成します。
4. update-auto-scaling-group コマンドにより、新しい起動テンプレートか起動設定を使用する、Auto Scaling グループを更新します。

キーペア <お使いの EC2 インスタンスに関連付けられているキーペア> は存在しません。EC2 インスタンスの起動に失敗しました。

原因: インスタンスの起動時に使用されたキーペアが削除された可能性があります。

解決策:

1. describe-key-pairs コマンドを使用して、利用可能なキーペアのリストを取得します。
2. リストから、使用するキーペアを選択します。代わりに、キーペアを作成するには、create-key-pair コマンドを使用します。
3. 新しい起動テンプレート、または起動設定を作成します。
4. update-auto-scaling-group コマンドにより、新しい起動テンプレートか起動設定を使用する、Auto Scaling グループを更新します。

リクエストされたインスタンスタイプ (<インスタンスタイプ>) は、リクエストされたアベイラビリティーゾーン (<インスタンスのアベイラビリティーゾーン>) ではサポートされません。

エラーメッセージ: Your requested instance type (<instance type>) is not supported in your requested Availability Zone (<instance Availability Zone>)...Launching EC2 instance failed. (お客様がリクエストしたインスタンスタイプ (<instance type>) は、ご指定のアベイラビリティーゾーン (<instance Availability Zone>) ではサポートされていません...EC2 インスタンスの起動に失敗しました)

原因: Auto Scaling グループで指定されているアベイラビリティーゾーンでは、選択したインスタンスタイプがサポートされていません。

解決策:

1. describe-instance-type-offerings コマンドを使用するか、Amazon EC2 コンソールで [インスタンスタイプ] ページのネットワークペインにある アベイラビリティゾーン の値を確認してして、どのアベイラビリティーゾーンが選択したインスタンスタイプをサポートしているかを確認します。
2. update-auto-scaling-group コマンドを使用して、Auto Scaling グループの設定でサポートされていないゾーンのサブネットを更新または削除します。詳細については、「アベイラビリティーゾーンを追加および削除する」を参照してください。

設定したポットリクエスト料金 0.015 は、スポットリクエストに最低限必要な料金 0.0735 を下回っています...。

原因: リクエストしたスポットの上限価格が、選択したインスタンスタイプでのスポット料金を下回っています。

解決策: より高いスポット上限価格 (オンデマンド料金が利用できます) を指定して、新しいリクエストを送信します。以前は、支払うスポット料金は入札に基づいていました。現在は、その時点のスポット料金が適用されます。上限価格を高く設定することで、要求されたキャパシティーを Amazon EC2 スポットサービスが確保し、それを維持できる可能性を高められます。

デバイス名 <device name> が無効です/無効なデバイス名をアップロードしようとしています。EC2 インスタンスの起動に失敗しました。

原因 1: 起動テンプレートまたは起動設定のブロックデバイスマッピングに、利用が不可能な、または現在サポートされていないブロックデバイス名が含まれている可能性があります。

解決策:

1. ご使用の具体的なインスタンス設定で、使用が可能なデバイス名を確認します。デバイスの命名の詳細については、「Amazon EC2 ユーザーガイド」の「Linux インスタンスのデバイス名」を参照してください。 Amazon EC2
2. Auto Scaling グループには含まれない Amazon EC2 インスタンスを手動で作成して、問題を調査します。ブロックデバイスの名前付けに関する設定と、Amazon マシンイメージ (AMI) にある名前とが競合している場合、インスタンスは起動に失敗します。詳細については、Amazon EC2 ユーザーガイド」の「ブロックデバイスマッピング」を参照してください。
3. インスタンスが正常に起動したことを確認したら、describe-volumes コマンドを使用して、どのようにボリュームがインスタンスに公開されているかを確認します。
4. ボリュームの説明にリストされているデバイス名を使用して、新しい起動テンプレートもしくは起動設定を作成します。
5. update-auto-scaling-group コマンドにより、新しい起動テンプレートか起動設定を使用する、Auto Scaling グループを更新します。

パラメータ virtualName の値 (<インスタンスストレージのデバイスに関連付けられている名前>) は無効です。EC2 インスタンスの起動に失敗しました。

原因: ブロックデバイスに関連付けられている仮想名の指定形式が正しくありません。

解決策:

1. `virtualName` パラメータでデバイス名を指定して、新しい起動テンプレートか起動設定を作成します。デバイス名の形式の詳細については、「Amazon EC2 ユーザーガイド」の「Linux インスタンスでのデバイスの命名」を参照してください。 Amazon EC2
2. update-auto-scaling-group コマンドにより、新しい起動テンプレートか起動設定を使用する、Auto Scaling グループを更新します。

EBS ブロックデバイスマッピングはインスタンスストア AMI ではサポートされません。

原因: 起動テンプレートあるいは起動設定で指定されたブロックデバイスマッピングが、お使いのインスタンスではサポートされていません。

解決策:

1. お使いのインスタンスタイプでサポートされるブロックデバイスマッピングを指定して、新しい起動テンプレートか起動設定を作成します。詳細については、「Amazon EC2 ユーザーガイド」の「ブロックデバイスマッピング」を参照してください。 Amazon EC2
2. update-auto-scaling-group コマンドにより、新しい起動テンプレートか起動設定を使用する、Auto Scaling グループを更新します。

プレイスメントグループは、「<instance type>」タイプのインスタンスでは使用できません。EC2 インスタンスの起動に失敗しました。

原因: お使いのクラスタープレイスメントグループに無効なインスタンスタイプが含まれています。

解決策:

1. プレイスメントグループでサポートされている有効なインスタンスタイプについては、Amazon EC2 ユーザーガイド」の「プレイスメントグループ」を参照してください。
2. 新しいプレイスメントグループを作成するには、「プレイスメントグループ」で説明している手順に従います。
3. または、サポートされるインスタンスタイプを使用する、新しい起動テンプレートもしくは起動設定を作成します。
4. 新しいプレイスメントグループや新しい起動テンプレートまたは起動設定により、Auto Scaling グループを更新するには、update-auto-scaling-group コマンドを使用します。

Client.InternalError: 起動時のクライアントエラー。

問題: Amazon EC2 Auto Scaling は、暗号化された EBS ボリュームを持つインスタンスを起動しようとしますが、サービスにリンクされたロールに、暗号化に使用される AWS KMS カスタマーマネージドキーへのアクセスがありません。詳細については、「暗号化されたボリュームで使用するために必要な AWS KMS キーポリシー」を参照してください。

原因 1: 適切なサービスにリンクされたロールがカスタマーマネージドキーを使用するための、アクセス許可を付与できるキーポリシーが必要です。

解決策 1: 次のように、サービスにリンクされたロールがカスタマーマネージドキーを使用することを許可します。

1. この Auto Scaling グループで使用するサービスにリンクされたロールを決定します。
2. カスタマーマネージドキーのキーポリシーを更新して、サービスにリンクされたロールでカスタマーマネージドキーが使用できるように許可します。
3. サービスにリンクされたロールを使用するため、Auto Scaling グループを更新します。

サービスにリンクされたロールにカスタマーマネージドキーの使用を許可するキーポリシーの例については、「例 1: カスタマー管理キーへのアクセスを許可するキーポリシーセクション」を参照してください。

原因 2: カスタマーマネージドキーと Auto Scaling グループが異なる AWS アカウントにある場合は、カスタマーマネージドキーを使用するアクセス許可を適切なサービスにリンクされたロールに付与するために、カスタマーマネージドキーへのクロスアカウントアクセスを設定する必要があります。

解決策 2: 次のように、外部アカウントのサービスにリンクされたロールによる、ローカルアカウント内のカスタマーマネージドキーの使用を許可します。

1. カスタマーマネージドキーのキーポリシーを更新して、Auto Scaling グループアカウントが、そのカスタマーマネージドキーにアクセスできるようにします。
2. グラントを作成できる Auto Scaling グループアカウント内の IAM ユーザーまたはロールを定義します。
3. この Auto Scaling グループで使用するサービスにリンクされたロールを決定します。
4. 適切なサービスにリンクされたロールを付与対象プリンシパルとして、カスタマー管理キーに対するグラントを作成します。
5. サービスにリンクされたロールを使用するため、Auto Scaling グループを更新します。

詳細については、「例 2: カスタマー管理キーへのクロスアカウントアクセスを許可するキーポリシーセクション」を参照してください。

解決策 3: Auto Scaling グループと同じ AWS アカウントにあるカスタマーマネージドキーを使用する

1. スナップショットをコピーし、Auto Scaling グループと同じアカウントに属する別のカスタマーマネージドキーを使用して再暗号化を行います。
2. サービスにリンクされたロールに新しいカスタマーマネージドキーの使用を許可します。解決策 1 のステップを参照してください。

現在、お客様がリクエストしたアベイラビリティゾーン内には、<instance type> の十分なキャパシティーがありません...。EC2 インスタンスの起動に失敗しました。

エラーメッセージ: We currently do not have sufficient <instance type> capacity in the Availability Zone you requested (<requested Availability Zone>) (現在、リクエストされたアベイラビリティーゾーン (<requested Availability Zone>) に、<instance type> の十分なキャパシティーがありません) 追加のキャパシティーをプロビジョニングする作業を進めます。現時点では、リクエストでアベイラビリティーゾーンを指定しないか、<このインスタンスタイプを現在サポートしているアベイラビリティーゾーンのリスト> から選択することによって、<インスタンスタイプ> のキャパシティーを取得できます。EC2 インスタンスの起動に失敗しました。

原因: 現時点では、リクエストされたインスタンスタイプとアベイラビリティーゾーンの組み合わせはサポートされていません。

解決策 : 問題を解決するには、以下を試してください。

- Amazon EC2 Auto Scaling が、他の有効なアベイラビリティーゾーンでこのインスタンスタイプのキャパシティーを検出するまで数分お待ちください。
- Auto Scaling グループを他のアベイラビリティーゾーンに拡張してください。詳細については、「アベイラビリティーゾーンを追加および削除する」を参照してください。
- ベストプラクティスとして、さまざまなインスタンスタイプのセットを使用することで、特定の 1 つのインスタンスタイプに依存することを防ぎます。詳細については、「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。

リクエストされた予約には、このリクエストに十分な互換性と使用可能な容量がありません。EC2 インスタンスの起動に失敗しました。

原因 1: `targeted`オンデマンドキャパシティーの予約で起動できるインスタンス数の上限に達しました。

解決策 1: targeted オンデマンドキャパシティーの予約で起動できるインスタンスの数を増やすか、キャパシティーの予約グループを使用して、リザーブドキャパシティー以外のものを定期的にオンデマンドキャパシティーとして起動します。詳細については、「オンデマンドキャパシティ予約を使用して特定のアベイラビリティーゾーンのキャパシティを予約する」を参照してください。

原因 2: キャパシティブロックで起動できるインスタンスの数の上限に達しました。

キャパシティブロックでは、最初に購入したキャパシティの量によって制約されます。予想よりも多くの起動が発生し、使用可能なすべての容量を使い果たすと、起動が失敗します。インスタンスの終了は、完全に終了する前に、クリーンアッププロセスが長くなります。この間は再利用できません。これにより、起動が失敗する可能性があります。詳細については、「機械学習ワークロードCapacity Blocksに を使用する」を参照してください。

解決策 2: 問題を解決するには、以下を試してください。

- リクエストはそのままにしておきます。キャパシティブロックインスタンスが終了する場合は、インスタンスが終了し、キャパシティが再び利用可能になるまで数分待つ必要があります。Amazon EC2 Auto Scaling は、キャパシティーが利用可能になるまで、引き続き自動的に起動リクエストを行います。
- このエラーが頻繁に発生しないように、ピーク時のワークロードに対応するのに十分な容量を購入してください。

キャパシティブロック予約 <予約 ID> はまだアクティブではありません。EC2 インスタンスの起動に失敗しました。

原因 : 指定されたキャパシティブロックがまだアクティブではありません。

解決策 : キャパシティブロックの推奨アプローチに従い、スケジュールされたスケーリングを使用します。これにより、予約がアクティブな場合にのみ Auto Scaling グループの希望するキャパシティを増やし、予約が終了する前に減らすことができます。

リクエストに適合した、使用可能なスポットキャパシティーはありません。EC2 インスタンスの起動に失敗しました。

原因: 現時点では、このスポットインスタンスのリクエストを満たすのに十分な空きキャパシティーがありません。

解決策 : 問題を解決するには、以下を試してください。

- キャパシティーは頻繁に変化するので、数分間待ってください。Amazon EC2 Auto Scaling は、キャパシティーが利用可能になるまで、引き続き自動的に起動リクエストを行います。
- Auto Scaling グループを他のアベイラビリティーゾーンに拡張してください。詳細については、「アベイラビリティーゾーンを追加および削除する」を参照してください。
- ベストプラクティスとして、さまざまなインスタンスタイプのセットを使用することで、特定の 1 つのインスタンスタイプに依存することを防ぎます。詳細については、「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。

<インスタンス数> 個のインスタンスがすでに実行中です。EC2 インスタンスの起動に失敗しました。

原因: 1 つのリージョンで起動できるインスタンスの合計数には制限があります。 AWS アカウントを作成すると、リージョンごとに実行できるインスタンスの数にデフォルトの制限が設定されます。

解決策 : 問題を解決するには、以下を試してください。

- 現在の制限がニーズに見合っていない場合は、リージョンごとにクォータの引き上げをリクエストできます。詳細については、Amazon EC2 ユーザーガイド」の「Amazon EC2 サービスクォータAmazon EC2」を参照してください。
- この新しいリクエストでは、インスタンス数を減らして送信します (これは後で増やすことができます)。

Amazon EC2 Auto Scaling をトラブルシューティングする: AMI 問題

このページでは、AMI に関連する問題、考えられる原因、問題を解決するために実行できる手順に関する情報を提供します。

エラーメッセージを取得するには、「スケーリングアクティビティからのエラーメッセージを取得する」を参照してください。

AMI に関する問題が原因で EC2 インスタンスの起動に失敗する場合、以下のエラーメッセージが 1 つ以上表示される可能性があります。

AMI に関する問題

- AMI ID <お使いの AMI の ID> は存在しません。EC2 インスタンスの起動に失敗しました。

- AMI <AMI ID> は保留中のため実行できません。EC2 インスタンスの起動に失敗しました。
- デバイス名 <device name> が無効です。EC2 インスタンスの起動に失敗しました。
- 指定されたインスタンスタイプのアーキテクチャ「arm64」が、指定された AMI のアーキテクチャ「x86_64」と一致しません。EC2 インスタンスの起動に失敗しました。
- AMI '<AMI ID>' は無効になっており、実行できません。EC2 インスタンスの起動に失敗しました。

> **Important**
>
> AWS では、AMI アクセス許可を変更することで、AMI を別の AWS アカウントとプライベートに共有できます。AMI を共有せずにプライベートにすると、新しいインスタンスを起動するときに認証エラーが発生する可能性があります。プライベート AMIsAmazon EC2 ユーザーガイド」の「特定の AWS アカウントと AMI を共有する」を参照してください。

AMI ID <お使いの AMI の ID> は存在しません。EC2 インスタンスの起動に失敗しました。

- 原因: 起動テンプレートもしくは起動設定の作成後に、AMI が削除された可能性があります。
- 解決策:
  1. 有効な AMI を使用して、新しい起動テンプレートか起動設定を作成します。
  2. update-auto-scaling-group コマンドにより、新しい起動テンプレートか起動設定を使用する、Auto Scaling グループを更新します。

AMI <AMI ID> は保留中のため実行できません。EC2 インスタンスの起動に失敗しました。

原因: (実行中のインスタンスのスナップショットを取得するか、またはそのほかの方法で) AMI を作成したばかりのため、まだ利用可能ではない可能性があります。

解決策: AMI が利用可能になるのを待ってから、起動テンプレートもしくは起動設定を作成する必要があります。

デバイス名 <device name> が無効です。EC2 インスタンスの起動に失敗しました。

原因: EBS ボリュームを EC2 インスタンスにアタッチするときは、ボリュームに有効なデバイス名が指定されていません。選択した AMI がこのデバイス名をサポートしている必要があります。

解決策:

1. 新しい起動テンプレートまたは起動設定を作成し、AMI の正しいデバイス名を指定します。推奨される命名規則は、AMI の仮想化タイプによって異なります。詳細については、「Amazon EC2 ユーザーガイド」の「デバイス名」を参照してください。 Amazon EC2
2. update-auto-scaling-group コマンドにより、新しい起動テンプレートか起動設定を使用する、Auto Scaling グループを更新します。

指定されたインスタンスタイプのアーキテクチャ「arm64」が、指定された AMI のアーキテクチャ「x86_64」と一致しません。EC2 インスタンスの起動に失敗しました。

原因 1: AMI のアーキテクチャと、起動テンプレートまたは起動設定で使用されているインスタンスタイプが同じでない場合、Amazon EC2 Auto Scaling が互換性のないインスタンス設定を使用してインスタンスを起動しようとするとエラーが発生します。

解決策 1:

1. describe-images コマンドを使用するか、Amazon EC2 コンソールの [Amazon マシンイメージ (AMI)] ページの詳細ペインで アーキテクチャ値を確認して、AMI のアーキテクチャを確認します。
2.]describe-instance-types コマンドを使用するか、Amazon EC2 コンソールで [インスタンスタイプ] 画面の [アーキテクチャ] 列を確認して、AMI と同じアーキテクチャを持つインスタンスタイプを見つけます。 互換性のあるインスタンスタイプの選択の詳細については、「Amazon EC2 ユーザーガイド」の「インスタンスタイプを変更するための互換性」を参照してください。 Amazon EC2
3. AMI と同じアーキテクチャのインスタンスタイプを使用して、新しい起動テンプレートまたは起動設定を作成します。
4. update-auto-scaling-group コマンドにより、新しい起動テンプレートか起動設定を使用する、Auto Scaling グループを更新します。

原因 2: Amazon EC2 Auto Scaling は Auto Scaling グループの混合インスタンスポリシーで指定されているインスタンスタイプを起動しようとしますが、そのインスタンスタイプは起動テンプレートで指定されている AMI と同じアーキテクチャではありません。

解決策 1:混合インスタンスポリシーには、アーキテクチャの異なるインスタンスタイプを含めないでください。

1. describe-images コマンドを使用するか、Amazon EC2 コンソールの [Amazon マシンイメージ (AMI)] ページの詳細ペインで アーキテクチャ値を確認して、AMI のアーキテクチャを確認します。
2. describe-instance-types コマンドを使用するか、Amazon EC2 コンソールで [インスタンスタイプ] 画面の [アーキテクチャ] 列を確認して、混合インスタンス ポリシーに含める予定の各インスタンス タイプのアーキテクチャを確認します。互換性のあるインスタンスタイプの選択の詳細については、「Amazon EC2 ユーザーガイド」の「インスタンスタイプを変更するための互換性」を参照してください。 Amazon EC2
3. update-auto-scaling-group コマンドを使用して Auto Scaling グループから互換性のないインスタンスタイプを更新または削除します。

解決策 2: 同じ Auto Scaling グループ内の Arm (Graviton2) インスタンスと x86_64 (Intel) インスタンスの両方を起動するには、ARM 互換 AMI と Intel x86 互換 AMI でサポートされている起動テンプレートをそれぞれ使用して、混合インスタンスポリシーのインスタンスタイプと一致させる必要があります。

1. describe-images コマンドを使用するか、Amazon EC2 コンソールの [Amazon マシンイメージ (AMI)] ページの詳細ペインで アーキテクチャ値を確認して、既存の起動テンプレート内の AMI のアーキテクチャを確認します。
2. 使用予定の他のアーキテクチャと一致する AMI を使用して、新しい起動テンプレートを作成します。
3. Auto Scaling グループを更新して既存の起動テンプレートをオーバーライドし、update-auto-scaling-group コマンドを使用して互換性のあるインスタンス タイプごとに新しい起動テンプレートを指定します。詳細については、「インスタンスタイプに異なる起動テンプレートを使用する」を参照してください。

AMI '<AMI ID>' は無効になっており、実行できません。EC2 インスタンスの起動に失敗しました。

原因 : 無効になっている AMI からインスタンスを起動しようとしています。詳細については、「Amazon EC2 ユーザーガイド」の「AMI を無効にする」を参照してください。 Amazon EC2

解決策:

1. 新しい起動テンプレートまたは起動設定を作成し、無効にされていない AMI を指定します。
2. update-auto-scaling-group コマンドにより、新しい起動テンプレートか起動設定を使用する、Auto Scaling グループを更新します。

Amazon EC2 Auto Scaling をトラブルシューティングする: ロードバランサー問題

このページでは、Auto Scaling グループに関連付けられているロードバランサーが原因で発生する問題、考えられる原因、問題を解決するために実行できるステップに関する情報を提供します。

エラーメッセージを取得するには、「スケーリングアクティビティからのエラーメッセージを取得する」を参照してください。

Auto Scaling グループに関連付けられているロードバランサーで発生する問題が原因で、EC2 インスタンスが起動しない場合、以下のエラーメッセージのうち 1 つ以上が表示されることがあります。

ロードバランサーに関する問題

- 1 つ以上のターゲットグループが見つかりませんでした。ロードバランサーの設定の確認が失敗しました。
- Load Balancer <ご使用のロードバランサー> が見つかりません。ロードバランサーの設定の確認が失敗しました。
- <ロードバランサー名> という名前のアクティブなロードバランサーはありません。ロードバランサーの設定の更新が失敗しました。
- EC2 インスタンス <インスタンス ID> は VPC にありません。ロードバランサーの設定の更新が失敗しました。

> **Note**
>
> Reachability Analyzer を使用して、Auto Scaling グループのインスタンスにロードバランサー経由でアクセスできるかどうかを確認することで、接続の問題をトラブルシューティングできます。Reachability Analyzer によって自動的に検出されるさまざまなネットワーク設定ミスの問題については、「Reachability Analyzer ユーザーガイド」の「Reachability Analyzer explanation codes」(Reachability Analyzer の説明コード) を参照してください。

1 つ以上のターゲットグループが見つかりませんでした。ロードバランサーの設定の確認が失敗しました。

問題: Auto Scaling グループがインスタンスを起動すると、Amazon EC2 Auto Scaling は、Auto Scaling グループに関連付けられた Elastic Load Balancing リソースが存在することを検証しようとします。ターゲットグループが見つからない場合、スケーリングアクティビティが失敗し、`One or more target groups not found. Validating load balancer configuration failed.` というエラーが表示されます。

原因 1: Auto Scaling グループにアタッチされているターゲットグループが削除されています。

解決策 1: Amazon EC2 Auto Scaling コンソールまたは detach-load-balancer-target-groups コマンドを使用して、ターゲットグループを使用せず新しい Auto Scaling グループを作成するか、Auto Scaling グループから未使用のターゲットグループを削除できます。

原因 2: ターゲットグループは存在しますが、Auto Scaling グループの作成時にターゲットグループ ARN を指定しようとして問題が発生しました。リソースが正しい順序で作成されていません。

解決策 2: 新しい Auto Scaling グループを作成して、最後にターゲットグループを指定します。

Load Balancer <ご使用のロードバランサー> が見つかりません。ロードバランサーの設定の確認が失敗しました。

問題: Auto Scaling グループがインスタンスを起動すると、Amazon EC2 Auto Scaling は、Auto Scaling グループに関連付けられた Elastic Load Balancing リソースが存在することを検証しようとします。Classic Load Balancer が見つからない場合、スケーリングアクティビティは失敗し、`Cannot find Load Balancer <your load balancer>. Validating load balancer configuration failed.` というエラーが表示されます。

原因 1: この Classic Load Balancer は削除されました。

解決策 1: ロードバランサーを使用せず新しい Auto Scaling グループを作成するか、Amazon EC2 Auto Scaling コンソールまたは detach-load-balancers コマンドを使用して、未使用のロードバランサーを Auto Scaling グループから削除できます。

原因 2: Classic Load Balancer は存在しますが、Auto Scaling グループの作成時に、ロードバランサーの名前を指定しようとして問題が発生しました。リソースが正しい順序で作成されていません。

解決策 2: 新しい Auto Scaling グループを作成する際は、最後にロードバランサーの名前を指定します。

<ロードバランサー名> という名前のアクティブなロードバランサーはありません。ロードバランサーの設定の更新が失敗しました。

原因: 指定されたロードバランサーが削除された可能性があります。

解決策: 新しいロードバランサーを作成してから新たに Auto Scaling グループを作成する、またはロードバランサーを指定せずに新しい Auto Scaling グループを作成します。

EC2 インスタンス <インスタンス ID> は VPC にありません。ロードバランサーの設定の更新が失敗しました。

原因: 指定されたインスタンスは VPC に存在しません。

解決策: インスタンスに関連付けられているロードバランサーを削除するか、新しい Auto Scaling グループを作成します。

Amazon EC2 Auto Scaling をトラブルシューティングする: 起動テンプレート

次の情報を使用して、Auto Scaling グループの起動テンプレートを指定しようとする際に発生する可能性のある一般的な問題を診断して修正します。

インスタンスを起動できない

すでに指定された起動テンプレートを使用してインスタンスを起動できない場合は、一般的なトラブルシューティングについて以下を確認してください: Amazon EC2 Auto Scaling をトラブルシューティングする: EC2 インスタンス起動の失敗。

完全な形式の有効な起動テンプレートを使用する必要があります (無効な値)

問題: Auto Scaling グループの起動テンプレートを指定しようとすると、`You must use a valid fully-formed launch template` というエラーが表示されます。起動テンプレートの値は、起動テンプレートを使用している Auto Scaling グループが作成または更新された場合にのみ検証されるため、このエラーが発生する可能性があります。

原因 1: `You must use a valid fully-formed launch template` エラーが発生した場合、Amazon EC2 Auto Scaling が起動テンプレートを無効と見なす原因となる問題が存在します。これは一般的なエラーで、いくつかの原因が考えられます。

解決策 1: 次のステップを実行して、トラブルシューティングを行います。

1. エラーメッセージの 2 つ目の部分に注目して、詳細を確認してください。`You must use a valid fully-formed launch template` のエラーに続いて、対処する必要のある問題を特定する、より具体的なエラーメッセージを参照してください。
2. 原因が見つからない場合は、「run-instances」コマンドで起動テンプレートをテストします。次の例のように、`--dry-run` オプションを使用します。これにより問題を再現し、その原因に関するインサイトを提供します。

```
aws ec2 run-instances --launch-template LaunchTemplateName=my-template,Version='1' --dry-run
```

3. 値が有効でない場合は、指定された適切なリソースが存在することを確認してください。例えば、Amazon EC2 キーペアを指定する場合、リソースはアカウント内および Auto Scaling グループを作成/更新しているリージョンに存在している必要があります。
4. 期待した情報がない場合は、設定を確認し、必要に応じて起動テンプレートを調整します。
5. 変更を加えた後、`--dry-run` オプションで「run-instances」コマンドを再実行し、起動テンプレートが有効な値を使用していることを検証します。

詳細については、「Auto Scaling グループの起動テンプレートを作成する」を参照してください。

起動テンプレートを使用する権限がありません (許可が不十分)。

問題: Auto Scaling グループの起動テンプレートを指定しようとすると、`You are not authorized to use launch template` というエラーが表示されます。

原因 1: 起動テンプレートを使用しようとした際、それに十分なアクセス許可をIAM 認証情報が持っていない場合に、その起動テンプレートを使用する権限がないという旨のエラーが表示されます。

解決策 1: 問題を解決するには、以下を試してください。

- リクエストの作成に使用している IAM 認証情報に、 アクションを含む必要な EC2 API `ec2:RunInstances`アクションを呼び出すアクセス許可があることを確認します。何らかのタグを起動テンプレートで指定している場合は、`ec2:CreateTags` アクションを使用するためのアクセス許可も必要です。
- または、リクエストの作成に使用している IAM 認証情報に`AmazonEC2FullAccess`ポリシーが割り当てられていることを確認します。この AWS 管理ポリシーは、Amazon EC2 Auto Scaling 、、Elastic Load Balancing を含むすべての Amazon EC2 リソース CloudWatchおよび関連サービスへのフルアクセスを許可します。

IAM ポリシーの例など、起動テンプレートを使用するために必要なアクセス許可の詳細については、「Amazon EC2 ユーザーガイド」の「IAM アクセス許可を持つ起動テンプレートへのアクセスを制御する」を参照してください。 Amazon EC2 その他の IAM ポリシーの例については、「」を参照してください起動テンプレートのサポート。

原因 2: 使用する起動テンプレートでインスタンスプロファイルを指定している場合は、そのインスタンスプロファイルに関連付けられた IAM ロールを渡すために、IAM のアクセス許可が必要です。

解決策 2: リクエストの作成に使用している IAM 認証情報に、指定されたロールを Amazon EC2 Auto Scaling サービスに渡すための正しい`iam:PassRole`アクセス許可があることを確認します。詳細および IAM ポリシーの例については、「Amazon EC2 インスタンスで実行中のアプリケーション用の IAM ロール」を参照してください。インスタンスプロファイルのトラブルシューティングに関連する他のトピックについては、「IAM ユーザーガイド」の「IAM および Amazon EC2 のトラブルシューティング」を参照してください。

原因 3: 別の で AMI を指定する起動テンプレートを使用しようとしたときに AWS アカウント、その AMI がプライベートであり、 AWS アカウント 使用している と共有されていない場合、起動テンプレートを使用する権限がないというエラーが表示されます。

解決策 3: AMI のアクセス許可に、使用しているアカウントが含まれていることを確認します。詳細については、「Amazon EC2 ユーザーガイド」の「特定の と AMI を共有する AWS アカウント」を参照してください。 Amazon EC2

関連情報

このサービスを利用する際に役立つ関連リソースは以下の通りです。

リソース	説明
「Amazon EC2 Auto Scaling API Reference」	各 API オペレーションのドキュメントには、リクエストパラメータと XML レスポンスが示され、言語固有の SDK リファレンストピックへのリンクが掲載されています。
「autoscaling」（「AWS CLI Command Reference」）	Auto Scaling グループを操作するために使用できる AWS CLI コマンドの説明。
AWS Tools for PowerShell コマンドレットリファレンス	AWS Tools for PowerShell を使用すると、PowerShell コマンドラインから AWS リソースに対するオペレーションをスクリプト化できます。
AWS CloudFormationで Auto Scaling グループを作成する	AWS::AutoScaling::AutoScalingグループリソースを使用すると、手動アクションなしで Auto Scaling グループを構築、モデル化、管理できます。
「Amazon EC2 Auto Scaling エンドポイントとクォータ」（「AWS 全般のリファレンス」）	Amazon EC2 Auto Scaling のリージョンとエンドポイントに関する情報
製品ページ	Amazon EC2 Auto Scaling に関する情報の主要なウェブページ。
AWS re:Post	AWS マネージド型の質問と回答 (Q & A) サービスは、技術的な質問に対するクラウドソースの、専門家によるレビュー済みの回答を提供します。
Amazon EC2 ユーザーガイド」の「AMI を作成する」	インスタンスからカスタム Amazon マシンイメージ (AMI) を作成する方法について説明します。

リソース	説明
「Amazon EC2 ユーザーガイド」の「Linux インスタンスに接続する方法Amazon EC2」	起動する Linux インスタンスに接続する方法について説明します。
「Amazon EC2 ユーザーガイド」の「Windows インスタンスに接続する方法 Amazon EC2」	起動する Windows インスタンスに接続する方法について説明します。
「Amazon ユーザーガイド」の「推定 AWS 請求額をモニタリングする請求アラームの作成 CloudWatch 」	を使用して推定請求額をモニタリングする方法について説明します CloudWatch。
アプリケーション Auto Scaling ユーザーガイドhttps://docs.aws.amazon.com/autoscaling/application/userguide/	Amazon EC2 を超えて Amazon Web Services のスケーラブルなリソースのオートスケーリングを設定する方法について説明します。

の詳細については、以下の一般的なリソースを参照してください AWS。

- クラスとワークショップ – AWS スキルを磨き、実践的な経験を積むのに役立つセルフペースラボに加えて、ロールベースのコースや専門コースへのリンクです。
- AWS デベロッパーセンター – チュートリアルを詳しく調べたり、ツールをダウンロードしたり、デ AWS ベロッパーイベントについて学びます。
- AWS デベロッパーツール – AWS アプリケーションを開発および管理するためのデベロッパーツール、 SDKs、IDE ツールキット、コマンドラインツールへのリンク。
- 入門リソースセンター – をセットアップし AWS アカウント、 AWS コミュニティに参加して、最初のアプリケーションを起動する方法について説明します。
- ハンズオンチュートリアル — step-by-step チュートリアルに従って、 で最初のアプリケーションを起動します AWS。
- AWS ホワイトペーパー – ソリューションアーキテクトや他の技術専門家が AWS 作成したアーキテクチャ、セキュリティ、経済などのトピックを網羅した、技術 AWS ホワイトペーパーの包括的なリストへのリンクです。
- AWS Support センター – AWS Support ケースを作成および管理するためのハブ。フォーラム、技術上のFAQs、サービスヘルスステータス、 など、その他の役立つリソースへのリンクも含まれています AWS Trusted Advisor。

- AWS Support – クラウドでのアプリケーションの構築と実行に役立つ AWS Support one-on-one、高速応答サポートチャネルに関する情報のプライマリウェブページ。
- お問い合わせ - AWS の請求、アカウント、イベント、不正使用、その他の問題などに関するお問い合わせの受付窓口です。
- AWS サイト規約 – 当社の著作権と商標、お客様のアカウント、ライセンス、サイトアクセス、およびその他のトピックに関する詳細情報。

ドキュメント履歴

次の表は、2018 年 7 月以降の Amazon EC2 Auto Scaling ドキュメントへの重要な追加項目をまとめたものです。このドキュメントの更新に関する通知については、RSS フィードにサブスクライブできます。

変更	説明	日付
セキュリティ IAM 更新	AutoScalingServiceRolePolicy管理ポリシーにより Amazon EC2 (`ec2:GetSecurityGroupsForVpc` と`ec2:GetInstanceTypesFromInstanceRequirements`) に追加のアクセス権限が付与されるようになりました。	2024 年 2 月 29 日
ウォームプールのハイバネーションが新たにサポートされました AWS リージョン	2 つの追加リージョン AWS GovCloud (米国東部) と (米国西部) のウォームプール内のインスタンスを休止状態にできるようになりました。 AWS GovCloud ウォームプールの詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling のウォームプール」を参照してください。	2024 年 2 月 26 日
ウォームプールのハイバネーションがさらにサポートされました AWS リージョン	ヨーロッパ (チューリッヒ) と中東 (UAE) の 2 つのリージョンのウォームプールでインスタンスを休止できるようになりました。ウォームプールの詳細については、「Amazon EC2 Auto Scaling ユーザーガ	2024 年 2 月 21 日

	イド」の「Amazon EC2 Auto Scaling のウォームプール」を参照してください。	
クロスアカウントパラメータ使用のSupport	Amazon EC2 Auto Scaling AWS Systems Manager AWS アカウント で他のユーザーと共有しているパラメータを使用できるようになりました。詳細については、Amazon EC2 Auto Scaling ユーザーガイドの「起動テンプレートで AMI ID AWS Systems Manager の代わりにパラメータを使用する」を参照してください。	2024 年 2 月 21 日
新しいスポット価格保護オプション	属性ベースのインスタンスタイプ選択を使用するときに、スポットインスタンスの価格保護基準をオンデマンド価格に対するパーセンテージとして定義できるようになりました。詳細については、Amazon EC2 Auto Scaling ユーザーガイドの「価格保護」を参照してください。	2024 年 1 月 29 日

インスタンスメンテナンスポリシー	インスタンスメンテナンスポリシーを使用して、インスタンスの更新など、インスタンスが交換される原因となるイベント中に、既存のインスタンスが終了される前にインスタンスを起動するか、後に起動するかを定義できるようになりました。詳細については、Amazon EC2 Auto Scaling ユーザーガイドの「インスタンスメンテナンスポリシー」を参照してください。	2023 年 11 月 15 日
機械学習用のキャパシティブロック	起動テンプレートの作成時にキャパシティブロックの予約 ID を指定することで、キャパシティブロック内にインスタンスを起動できるようになりました。キャパシティブロックを使用すると、GPU インスタンスを将来の日付で予約して、短期間の機械学習 (ML) ワークロードをサポートすることができます。詳細については、Amazon EC2 Auto Scaling ユーザーガイドの「機械学習ワークロードにキャパシティブロックを使用する」を参照してください。	2023 年 10 月 31 日

新しいインスタンスの更新機能	インスタンスの更新を設定してステータスを失敗に設定し、 CloudWatch ALARMオプションで指定したアラームがその状態になったことを検出したときにロールバックできるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「ロールバックで変更の取り消し」を参照してください。	2023 年 7 月 31 日
ガイドの変更点	キャパシティ予約へのオンデマンドインスタンスの起動に関する新しいトピックがガイドに追加されました。キャパシティ予約の詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Use On-Demand Capacity Reservations to reserve capacity in specific Availability Zones」を参照してください。	2023 年 7 月 28 日

ガイドの変更点	AWS CloudFormation 起動設定から起動テンプレートへのスタックの移行に関する新しいトピックがガイドに追加されました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「AWS CloudFormation スタックを起動設定から起動テンプレートに移行する」を参照してください。	2023 年 4 月 18 日

新しい API オペレーションのサポート	このリリースでは、AttachTrafficSources 、DetachTrafficSources 、および DescribeTrafficSources という 3 つの新しい API オペレーションが追加されています。また、新しいフィールドである TrafficSources が DescribeAutoScalingGroups オペレーションの結果に追加されました。新しいアクティビティステータスである WaitingForConnectionDraining が DescribeScalingActivities オペレーションの結果に追加されました。Amazon EC2 Auto Scaling は、CreateAutoScalingGroup 、UpdateAutoScalingGroup 、および DescribeAutoScalingGroups オペレーションで HealthCheckType フィールドの新しい値である VPC_LATTICE もサポートします。詳細については、「Amazon EC2 Auto Scaling API Reference」(Amazon EC2 Auto Scaling API リファレンス) を参照してください。	2023 年 3 月 31 日

Amazon VPC Lattice のサポート	これは、VPC Lattice for Amazon EC2 Auto Scaling の一般提供向けリリースです。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「VPC Lattice ターゲットグループを使用して Auto Scaling グループにトラフィックをルーティングする」を参照してください。	2023 年 3 月 31 日
ガイドの変更点	Elastic Load Balancing AWS CLI の使用例を含むセクションに、新しい例と更新された例が追加されました。詳細については、Amazon EC2 Auto Scaling ユーザーガイドの「AWS Command Line Interface (AWS CLI) を使用して Elastic Load Balancing を使用する場合の例」を参照してください。	2023 年 3 月 31 日
予測スケーリングのSupportが追加されました AWS リージョン	中東 (UAE) と AWS GovCloud (米国東部) リージョンで予測スケーリングポリシーを作成できるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling の予測スケーリング」を参照してください。	2023 年 3 月 16 日

新しいインスタンスの更新機能	スタンバイ状態のインスタンスを終了または無視し、スケールインから保護されているインスタンスが交換可能になるのを待たず、置き換えまたは無視することができます。失敗したインスタンスの更新による変更をロールバックすることもできます。この更新の一環として、ドキュメントが拡張され、インスタンス更新のロールバック、インスタンス更新のキャンセル、インスタンス更新の設定可能なパラメータのデフォルト値の説明に関するトピックが含まれるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「インスタンスの更新に基づく Auto Scaling インスタンスの置き換え」を参照してください。	2023 年 2 月 10 日
AMI ID AWS Systems Manager のパラメータの使用のSupport	起動テンプレートの AMI ID の代わりに、Systems Manager パラメータを使用できるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「起動テンプレートで AMI ID の代わりに AWS Systems Manager パラメータを使用」を参照してください。	2023 年 1 月 19 日

予測スケーリングの推奨事項	Amazon EC2 Auto Scaling コンソールから適した予測スケーリングポリシーを評価して選択するため、推奨事項を確認できるようになりました。の詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「予測スケーリングポリシーの評価」を参照してください。	2023 年 1 月 18 日
予測スケーリングによる予測	予測スケーリングによる予測の生成は、更新頻度が毎日から 6 時間ごとに変更されました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling の予測スケーリング」を参照してください。	2023 年 1 月 6 日

CloudWatch メトリック計算のSupport	ターゲット追跡スケーリングポリシーの作成時に Metric Math を使用できるようになりました。メトリクス演算では、 CloudWatch 複数のメトリクスをクエリし、数式を使用してこれらのメトリクスに基づいて新しい時系列を作成できます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Create a target tracking scaling policy for Amazon EC2 Auto Scaling using metric math」(Metric Math を使用する Amazon EC2 Auto Scaling のターゲット追跡スケーリングポリシーを作成する) を参照してください。	2022 年 12 月 8 日
IAM サービスリンクロール許可に対する更新	`AutoScalingServiceRolePolicy` ポリシーが、Amazon EC2 Auto Scaling に追加のアクセス許可を付与するようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling 用のAWS マネージドポリシー」を参照してください。	2022 年 12 月 6 日

新しいスポット配分戦略	価格とキャパシティを最適化した配分戦略を使用して、中断される可能性が最も低く、価格も可能な限り低いスポットプールからスポットインスタンスをリクエストできるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「配分戦略」を参照してください。	2022 年 11 月 10 日
アジアパシフィック (ジャカルタ) リージョンでの予測スケーリングのサポート	アジアパシフィック (ジャカルタ) リージョンで予測スケーリングポリシーを作成できるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling の予測スケーリング」を参照してください。	2022 年 10 月 13 日
コンソールでの予測スケーリングのためのカスタムメトリクスに対するサポート	Amazon EC2 Auto Scaling コンソールから予測スケーリングポリシーを作成するときに、カスタムメトリクスを使用できるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling の予測スケーリング」を参照してください。	2022 年 10 月 13 日

CloudWatch プレディクティブ・スケーリング・メトリクスのモニタリング	を使用して予測スケーリング用のモニタリングデータにアクセスできるようになりました。 CloudWatchこれにより、Metric Math を使用して、予測データの精度を表示する新しい時系列を作成できます。詳細については、Amazon EC2 Auto Scaling ユーザーガイドの「CloudWatch予測スケーリングメトリクスをモニタリングする」を参照してください。	2022 年 7 月 7 日
アジアパシフィック (大阪) リージョンでの予測スケーリングのサポート	アジアパシフィック (大阪) リージョンで予測スケーリングポリシーを作成できるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling の予測スケーリング」を参照してください。	2022 年 7 月 6 日

ウォームプール休止状態をサポートするリージョンの追加	アフリカ (ケープタウン)、アジアパシフィック (ジャカルタ)、アジアパシフィック (大阪)、欧州 (ミラノ) の 4 つのリージョンでウォームプール内のインスタンスを休止できるようになりました。ウォームプールの詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling のウォームプール」を参照してください。	2022 年 7 月 5 日
ヘルスチェックに対する更新	Amazon EC2 Auto Scaling は、ヘルスチェックを実行するときに、一時的な問題や誤設定されたヘルスチェックが原因で発生する可能性があるダウンタイムを最小限に抑えることができるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling によってダウンタイムが最小限に抑えられる仕組み」を参照してください。	2022 年 5 月 21 日

インスタンスのデフォルトウォームアップ	Auto Scaling グループのすべてのウォームアップとクールダウンの設定を統合し、デフォルトのインスタンスウォームアップを有効にすることで継続的にスケーリングするスケーリングポリシーのパフォーマンスを最適化できるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling グループに対するインスタンスのデフォルトウォームアップを設定する」を参照してください。	2022 年 4 月 19 日
ガイドの変更点	AWS 他のサービスとの統合に関する新しい章がガイドに追加されました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling と統合されたAWS のサービス」を参照してください。	2022 年 3 月 29 日
IAM サービスリンクロール許可に対する更新	`AutoScalingServiceRolePolicy` ポリシーが、Amazon EC2 Auto Scaling に追加の読み取り許可を付与するようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling 用のAWS マネージドポリシー」を参照してください。	2022 年 3 月 28 日

インスタンスメタデータがターゲットライフサイクルステータスを提供	インスタンスメタデータから Auto Scaling インスタンスのターゲットライフサイクル状態を取得できます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「インスタンスメタデータを使用してターゲットライフサイクル状態を取得する」を参照してください。	2022 年 3 月 24 日
新しいウォームプール機能をサポート	ウォームプール内のインスタンスを休止して、メモリコンテンツ (RAM) を削除せずにインスタンスを停止できるようになりました。また、後で必要になるインスタンスキャパシティを常に終了する代わりに、スケールイン時にインスタンスをウォームプールに戻すこともできるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling のウォームプール」を参照してください。	2022 年 2 月 24 日

ガイドの変更点	Amazon EC2 Auto Scaling コンソールが追加オプションとともに更新され、スキップマッチングを有効化し、希望する設定を指定して、インスタンスの更新を開始できるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」のインスタンスの更新のスタートまたはキャンセル (コンソール)を参照してください。	2022 年 2 月 3 日
予測スケーリングポリシーのカスタムメトリクス	予測スケーリングポリシーを作成する際に、カスタムメトリクスの使用を選択できるようになりました。メトリクスの数式を使用して、ポリシーに含めるメトリクスをさらにカスタマイズすることもできます。詳細については、「Advanced predictive scaling policy configurations using custom metrics」(カスタムメトリクスを使用した高度な予測スケーリングポリシーの設定) を参照してください。	2021 年 11 月 24 日

新しいオンデマンド配分戦略	混合インスタンスポリシーを使用する Auto Scaling グループを作成する際に、価格に基づくオンデマンドインスタンスの起動 (最低価格のインスタンスタイプが最初に起動) を選択できるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「配分戦略」を参照してください。	2021 年 10 月 27 日
属性ベースのインスタンスタイプの選択	Amazon EC2 Auto Scaling では、属性ベースのインスタンスタイプの選択をサポートするようになりました。インスタンスタイプを手動で選択するのではなく、インスタンス要件を vCPU、メモリ、ストレージなどの一連の属性として表現できます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Creating an Auto Scaling group using attribute-based instance type selection」(属性ベースのインスタンスタイプの選択を使用した Auto Scaling グループの作成) を参照してください。	2021 年 10 月 27 日

タグによるグループのフィルタリングのサポート	`describe-auto-scaling-groups` コマンドを使用して Auto Scaling グループに関する情報を取得する際に、タグフィルターを使用して Auto Scaling グループをフィルタリングできるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Use tags to filter Auto Scaling groups」(タグを使用した Auto Scaling グループのフィルタリング) を参照してください。	2021 年 10 月 14 日
ガイドの変更点	Amazon EC2 Auto Scaling コンソールが更新され、でカスタム終了ポリシーを作成できるようになりました AWS Lambda。コンソールのドキュメントも、それに応じて改訂されました。詳細については、「Using different termination policies (console)」(異なる終了ポリシーを使用する (コンソール)) を参照してください。	2021 年 10 月 14 日

起動設定を起動テンプレートへコピーするためのSupport	AWS リージョンのすべての起動設定を Amazon EC2 Auto Scaling コンソールから新しい起動テンプレートにコピーできるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「起動設定を起動テンプレートにコピー」を参照してください。	2021 年 8 月 9 日
インスタンスの更新機能を拡張	希望する設定を`start-instance-refresh` コマンドに追加することで、起動テンプレートの新しいバージョンなど、アップデートを含むことができるようになりました。スキップマッチングを有効にすることで、すでに希望の設定があるインスタンスの置き換えをスキップすることもできます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「インスタンスの更新に基づく Auto Scaling インスタンスの置き換え」を参照してください。	2021 年 8 月 5 日

カスタム終了ポリシーのSupport	でカスタム終了ポリシーを作成できるようになりましたAWS Lambda。詳細については、「Lambda を使用したカスタム終了ポリシーの作成」を参照してください。それに伴い、終了 ポリシーを指定するためのドキュメントが更新されました。	2021 年 7 月 29 日
ガイドの変更点	Amazon EC2 Auto Scaling コンソールが更新され、機能が追加して拡張され、タイムゾーンを指定したスケジュールアクションを作成できるようになりました。それに応じてスケジュールされたスケーリングのドキュメントが改訂されました。	2021 年 6 月 3 日
起動設定の gp3 ボリューム	起動設定のブロックデバイスマッピングで gp3 ボリュームを指定することができるようになりました。	2021 年 6 月 2 日

予測スケーリングのSupport	予測スケーリングを使用して スケーリング ポリシーを使用する、Amazon EC2 Auto Scaling グループを事前にスケールできるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling の予測スケーリング」を参照してください。今回の更新により、AutoScalingServiceRolePolicy管理ポリシーに `cloudwatch:GetMetricData` API アクションを呼び出す権限が含まれるようになりました。	2021 年 5 月 19 日
ガイドの変更点	GitHubからライフサイクルフックのサンプルテンプレートにアクセスできるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling のライフサイクルフック」を参照してください。	2021 年 4 月 9 日

ウォームプールのSupport	Auto Scaling グループにウォームプールを追加することで、初回の起動時間の長いアプリケーションのパフォーマンス（コールドスタートを最小限にする）とコスト（インスタンスキャパシティーのオーバー-プロビジョニングを停止）のバランスをとることができるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling のウオームプール」を参照してください。	2021 年 4 月 8 日
チェックポイントのSupport	インスタンスの更新にチェックポイントを追加して、インスタンスを段階的に置き換え、特定のポイントでインスタンスの検証を実行できるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「インスタンスの更新へのチェックポイントの追加」を参照してください。	2021 年 3 月 18 日

ガイドの変更点	Amazon EC2 Auto Scaling EventBridge イベントとライフサイクルフックで使用するためのドキュメントが改善されました。詳細については、Amazon EC2 Auto Scaling ユーザーガイドの「Amazon EC2 Auto Scaling を使用する」 EventBridge および「チュートリアル:Lambda 関数を呼び出すライフサイクルフックを設定する」を参照してください。	2021 年 3 月 18 日
「ローカルタイムゾーンの Support」	`--time-zone` オプションをput-scheduled-update-group-actionコマンドに追加することで、ローカルタイムゾーンで定期的にスケジュールされたアクションを作成できるようになりました。タイムゾーンが夏時間 (DST) を遵守している場合、定期的なアクションは自動的に夏時間に調整されます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Scheduled scaling (スケジュールされたスケーリング) 」を参照してください。	2021 年 3 月 9 日

混合インスタンスポリシーの機能を拡張	混合インスタンスポリシーを使用する場合に、スポットキャパシティのインスタンスタイプを優先できるようになりました。Amazon EC2 Auto Scaling は、ベストエフォートベースで優先順位を履行しようとしますが、まずはキャパシティに合わせて最適化します。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。	2021 年 3 月 8 日
削除されたグループのアクティビティをスケーリング	`--include-deleted-groups` オプションを describe-scaling-activities コマンドを追加することによって、削除された Auto Scaling グループのスケーリングアクティビティを表示できるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling のトラブルシューティング」を参照してください。	2021 年 2 月 23 日

コンソールの改善	Amazon EC2 Auto Scaling コンソールから Application Load Balancer または Network Load Balancer を作成およびアタッチできるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling コンソールを使用して、Application Load Balancer または Network Load Balancer を設定します。」を参照してください。	2020 年 11 月 24 日
「複数のネットワークインターフェイス」	複数のネットワークインターフェイスを指定する Auto Scaling グループの起動テンプレートを設定できるようになりました。詳細については、「VPCのネットワークインターフェイス」を参照してください。	2020 年 11 月 23 日
複数の起動テンプレート	Auto Scaling グループで複数の起動テンプレートを使用できるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「インスタンスタイプに異なる起動テンプレートを指定する」を参照してください。	2020 年 11 月 19 日

Gateway Load Balancer	Amazon EC2 Auto Scaling によって起動されたアプライアンスインスタンスが自動的にメンバーとなり、ロードバランサー から登録解除されるように、ゲートウェイ ロードバランサーを Auto Scaling グループにアタッチする方法を示すガイドが更新されました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Elastic Load Balancing タイプ」と「ロードバランサーをAuto Scaling グループへアタッチ」を参照してください。	2020 年 11 月 10 日
最大インスタンス有効期間	インスタンスの最大有効期間を 1 日 (86400 秒) に減らすことができるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「最大インスタンス有効期間に基づいた Auto Scaling インスタンスの置き換え」を参照してください。	2020 年 11 月 9 日

キャパシティーの再調整	Amazon EC2 が再調整のレコメンデーションを発行したときに、代わりのスポットインスタンスを起動するようにAuto Scaling グループを設定できます。詳細については、Amazon EC2 Auto Scaling ユーザーガイドの Amazon EC2 Auto Scaling キャパシティーの再調整を参照してください。	2020 年 11 月 4 日
「インスタンス メタデータ サービス バージョン 2」	起動設定を使用するときに、インスタンスメタデータのリクエストに、セッション志向な方法であるインスタンス メタデータサービス バージョン 2の使用を要求することができます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「インスタンスメタデータオプションの設定」を参照してください。	2020 年 7 月 28 日
ガイドの変更点	「スケールイン中の Auto Scaling インスタンス終了を制御」、「Auto Scaling インスタンスおよびグループのモニタリング」,「起動テンプレート」, および「Amazon EC2 Auto Scaling ユーザーガイド」の起動設定セクションにおけるさまざまな改善点と新しいコンソール手順。	2020 年 7 月 28 日

インスタンスの更新	設定を変更したときに、インスタンスの更新をスタートして、Auto Scaling グループ内のすべてのインスタンスを更新します。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「インスタンスの更新に基づく Auto Scaling インスタンスの置き換え」を参照してください。	2020 年 6 月 16 日
ガイドの変更点	「インスタンスの最大有効期間に基づいた Auto Scaling インスタンスの置き換え」,「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」,「Amazon SQS に基づくスケーリング」, および「Amazon EC2 Auto Scaling ユーザーガイド」の Auto Scaling グループとインスタンスのタグ付けにおけるさまざまな改善点。	2020 年 5 月 6 日
ガイドの変更点	IAM ドキュメントに対するさまざまな改善点。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「起動テンプレートの Support」および「Amazon EC2 Auto Scaling の アイデンティティ ベースのポリシーの例」を参照してください。	2020 年 3 月 4 日

スケーリングポリシーの無効化	スケーリングポリシーを無効化して、再有効化することができるようになりました。この機能を使用すると、設定の詳細を保持しながら、スケーリングポリシーを一時的に無効化して、後でポリシーを再有効化することができます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling グループのスケーリングポリシーの無効化」を参照してください。	2020 年 2 月 18 日
通知機能の追加	Amazon EC2 Auto Scaling は、セキュリティグループまたは起動テンプレートがないために Auto Scaling AWS Health Dashboard グループがスケールアウトできない場合に、お客様にイベントを送信するようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「AWS Health Dashboard Amazon EC2 Auto Scalingの 通知」を参照してください。	2020 年 2 月 12 日

ガイドの変更点	「Amazon EC2 Auto Scaling の IAM を使っての働き」，「Amazon EC2 Auto Scaling の アイデンティティ ベースのポリシーの例」，「暗号化されたボリュームに使用する必須の CMK キーポリシー」，および「Amazon EC2 Auto Scaling ユーザーガイド」のAuto Scaling インスタンスおよびグループのモニタリングのセクションにおけるさまざまな改善点と修正点。	2020 年 2 月 10 日
ガイドの変更点	インスタンスの重み付けを使用する Auto Scaling グループのドキュメントが改善されました。「キャパシティーユニット」を使用して希望するキャパシティーを測定するときに、スケーリングポリシーを使用する方法について説明します。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」のスケーリングポリシーのしくみおよびスケーリング調整タイプを参照してください。	2020 年 2 月 6 日

新しい「セキュリティ」章	「Amazon EC2 Auto Scaling ユーザーガイド」の新しい「セキュリティ」の章は、Amazon EC2 Auto Scaling を使用する際の責任共有モデルの適用方法を理解するのに役立ちます。この更新のパートとして、ユーザーガイドの章「Amazon EC2 Auto Scaling リソースに対するアクセスのコントロール」が、新しくより役立つセクション「Amazon EC2 Auto Scaling の Identity and Access Management」に置き換えられました。	2020 年 2 月 4 日
インスタンスタイプに関する推奨事項	AWS Compute Optimizer には、パフォーマンスの向上、コストの節約、またはその両方に役立つ Amazon EC2 インスタンスの推奨事項が記載されています。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「インスタンスタイプに関する推奨事項の取得」を参照してください。	2019 年 12 月 3 日

Dedicated Hosts およびホストリソースグループ	ホストリソースグループを指定する起動テンプレートの作成方法を示すガイドを更新しました。これにより、Dedicated Hosts で使用する BYOL AMI を指定する起動テンプレートを使用して Auto Scaling グループを作成できます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling グループの起動テンプレートの作成」を参照してください。	2019 年 12 月 3 日
Amazon VPC エンドポイントのサポート	これで、VPC と Amazon EC2 Auto Scaling との間でプライベート接続を確立できます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto ScalingとインターフェースVPC エンドポイント」を参照してください。	2019 年 11 月 22 日

最大インスタンス有効期間	インスタンスが稼働できる最大時間を指定することにより、インスタンスを自動的に置き換えることができるようになりました。インスタンスがこの制限に近づいてくれば、Amazon EC2 Auto Scaling は段階的にそれらのインスタンスを置き換えます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「最大インスタンス有効期間に基づいた Auto Scaling インスタンスの置き換え」を参照してください。	2019 年 11 月 19 日
インスタンスの重み付け	複数のインスタンスタイプを含む Auto Scaling グループの場合、オプションで、各インスタンス タイプからグループのキャパシティに割り当てるキャパシティの単位数を定義できるようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon EC2 Auto Scaling のインスタンスの重み付け」を参照してください。	2019 年 11 月 19 日

インスタンスタイプの最小数	スポット、オンデマンド、リザーブドインスタンスのグループに、追加のインスタンスタイプを指定する必要がなくなりました。すべての Auto Scaling グループで、インスタンスタイプの最小値が 1 つになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。	2019 年 9 月 16 日
新しいスポット配分戦略のサポート	Amazon EC2 Auto Scaling は、使用可能なスポットキャパシティーに基づいてオプションで選択された最適なスポットインスタンス プールを使用してリクエストを満たす、新しいスポット配分戦略「キャパシティー最適化」をサポートするようになりました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。	2019 年 8 月 12 日

ガイドの変更点	Amazon EC2 Auto Scaling ドキュメントの「サービスにリンクされたロール」および「暗号化されたボリュームで使用するのに必須の CMK キーポリシー」トピックを改善しました。	2019 年 8 月 1 日
タグ付け強化のサポート	Amazon EC2 Auto Scaling では、インスタンスを起動する同じ API コールのパートとして、Amazon EC2 インスタンスにタグを追加するようになりました。詳細については、「Auto Scaling Group とインスタンスのタグ付け」を参照してください。	2019 年 7 月 26 日
ガイドの変更点	「スケーリング プロセスの中断と再開」トピックのAmazon EC2 Auto Scaling ドキュメントを改善しました。「カスタマー マネージド ポリシーの例」を更新し、ユーザーが特定のカスタムサフィックスのサービスリンクされたロールのみを Amazon EC2 Auto Scaling に渡すことができるポリシーの例が含まれています。	2019 年 6 月 13 日

新しい Amazon EBS 機能のサポート	起動テンプレートのトピックに新しい Amazon EBS 機能のサポートを追加しました。スナップショットからの復元中に EBS ボリュームの暗号化状態を変更します。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling グループの起動テンプレートの作成」を参照してください。	2019 年 5 月 13 日
ガイドの変更点	以下のセクションの Amazon EC2 Auto Scaling ドキュメントが改善:「スケールイン中に終了する Auto Scaling インスタンスの制御」、「Auto Scaling グループ」、「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」、および「Amazon EC2 Auto Scaling の動的スケーリング」。	2019 年 3 月 12 日

インスタンスタイプと購入オプションの組み合わせのサポート	1 つの Auto Scaling グループ内で、購入オプション (スポット、オンデマンド、リザーブドインスタンス) およびインスタンスタイプ間でインスタンスをプロビジョンおよび自動的にスケーリングします。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。	2018 年 11 月 13 日
「Amazon SQS に基づくスケーリングのトピックが更新されました」	Amazon SQS キューの需要の変化に応じて、カスタムメトリクスを使用して Auto Scaling グループをスケールする方法を説明するガイドが更新されました。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Amazon SQSに基づくスケーリング」を参照してください。	2018 年 7 月 26 日

次の表は、2018 年 7 月以前の Amazon EC2 Auto Scaling ドキュメントへの重要な変更点を説明しているものです。

機能	説明	リリース日
ターゲット追跡スケーリングポリシーのサポート	わずかなステップで、アプリケーションの動的スケーリングを設定します。詳細については、「Amazon EC2 Auto Scaling のターゲット追跡スケーリング ポリシー」を参照してください。	2017 年 7 月 12 日

機能	説明	リリース日
リソースレベルのアクセス許可のサポート	リソースレベルでアクセスを制御する IAM ポリシーを作成します。詳細については、 Amazon EC2 Auto Scaling リソースへのアクセスを制御を参照してください。	2017 年 5 月 15 日
改善のモニタリング	Auto Scaling グループ メトリクスでは、詳細モニタリングを有効にする必要はなくなりました。コンソールの [Monitoring] タブから、グループメトリクスの収集とメトリクスグラフの表示を有効にできるようになりました。詳細については、「Amazon を使用して Auto Scaling グループとインスタンスをモニタリングする」を参照してください CloudWatch。	2016 年 8 月 18 日
Application Load Balancers の Support	新しいまたは既存の Auto Scaling グループに 1 つまたは複数のターゲットグループをアタッチします。詳細については、「Auto Scaling グループにロードバランサーをアタッチする」を参照してください。	2016 年 8 月 11 日
ライフサイクルフックのイベント	Amazon EC2 Auto Scaling は、 EventBridge ライフサイクルフックを呼び出すときにイベントを送信します。詳細については、「Auto Scaling EventBridge グループのスケーリングタイミングの取得」を参照してください。	2016 年 2 月 24 日
インスタンスの保護	スケーリング インする際に、Amazon EC2 Auto Scaling が特定のインスタンスを選択して終了することを防ぎます。詳細については、「インスタンスの保護」を参照してください。	2015 年 12 月 7 日
ステップスケーリングポリシー	超過アラームのサイズに基づいてスケールできるようになるスケーリングポリシーを作成します。詳細については、「スケーリング ポリシータイプ」を参照してください。	2015 年 7 月 6 日

機能	説明	リリース日
ロードバランサーの更新	既存の Auto Scaling グループへロードバランサーをアタッチ、またはAuto Scaling グループへロードバランサーをデタッチします。詳細については、「Auto Scaling グループにロードバランサーをアタッチする」を参照してください。	2015 年 6 月 11 日
のSupport ClassicLink	Auto Scaling グループの EC2-Classic インスタンスを VPC にリンクすることで、これらのリンクされた EC2-Classic インスタンスと VPC 内のインスタンスがプライベート IP アドレスを使用して通信できるようになりました。詳細については、「EC2-Classic インスタンスの VPC へのリンク」を参照してください。	2015 年 1 月 19 日
ライフサイクルフック	新しく起動されたインスタンスまたは終了中のインスタンスに対してアクションを実行する間、これらのインスタンスを保留状態に維持できるようになりました。詳細については、Amazon EC2 Auto Scaling ライフサイクルフックを参照してください。	2014 年 7 月 30 日
インスタンスのデタッチ	Auto Scaling グループからインスタンスをデタッチします。詳細については、「Auto Scaling グループから EC2 インスタンスをデタッチする」を参照してください。	2014 年 7 月 30 日
Standby 状態へのインスタンスの移行	`InService` 状態にあるインスタンスを `Standby` 状態に移行できるようになりました。詳細については、「Auto Scaling グループからのインスタンスの一時的な削除」を参照してください。	2014 年 7 月 30 日
タグの管理	AWS Management Consoleを使って、Auto Scaling グループを管理する 詳細については、「Auto Scaling Group とインスタンスのタグ付け」を参照してください。	2014 年 5 月 01 日

機能	説明	リリース日
ハードウェア専有インスタンスのサポート	起動設定を作成するときにプレイスメントテナンシー属性を指定して、ハードウェア専有インスタンスを起動できるようになりました。詳細については、「インスタンスのプレイスメント テナンシー」を参照してください。	2014 年 4 月 23 日
EC2 インスタンスからのグループまたは起動設定の作成	EC2 インスタンスを使用して Auto Scaling グループまたは起動設定を作成します。EC2 インスタンスを使用して起動設定を作成する方法については、「EC2 インスタンスを使用して起動設定の作成」を参照してください。EC2 インスタンスを使用して Auto Scaling グループを作成する方法については、「EC2 インスタンスを使用して Auto Scaling グループの作成」を参照してください。	2014 年 1 月 2 日
インスタンスのアタッチ	インスタンスを既存の Auto Scaling グループにアタッチすることで、EC2 インスタンスに対するオートスケーリングを有効にできます。詳細については、「Auto Scaling グループから EC2 インスタンスをアタッチする」を参照してください。	2014 年 1 月 2 日
アカウントの制限の表示	アカウントの Auto Scaling リソースの制限を表示します。詳細については、「Auto Scaling の制限」を参照してください。	2014 年 1 月 2 日
Amazon EC2 Auto Scaling のコンソールSupport	を使用して Amazon EC2 Auto Scaling にアクセスします AWS Management Console。詳細については、Amazon EC2 Auto Scalingの開始方法を参照してください。	2013 年 12 月 10 日
パブリック IP アドレスの割り当て	VPC で起動されるインスタンスにパブリック IP アドレスの割り当てられるようになりました。詳細については「VPC での Auto Scaling インスタンスの起動」を参照してください。	2013 年 9 月 19 日

機能	説明	リリース日
インスタンス終了ポリシー	EC2 インスタンスを終了するときに 、Amazon EC2 Auto Scaling で使用するインスタンス終了ポリシーを指定します。詳細については、「スケールイン時にどの Auto Scaling インスタンスを終了するかを制御」を参照してください。	2012 年 9 月 17 日
IAM ロールの Support	IAM インスタンスプロファイルを使用して EC2 インスタンスを起動する。この特徴を使用してインスタンスに IAM ロールを割り当てると、アプリケーションは他の AWS に安全にアクセスできます。詳細については、「IAM ロールを使用した Auto Scaling インスタンスの起動」を参照してください。	2012 年 6 月 11 日
スポットインスタンスのサポート	起動設定でスポットインスタンスを起動します。詳細については、「Requesting Spot Instances for fault-tolerant and flexible applications」(柔軟で耐障害性を備えたアプリケーションのスポットインスタンスのリクエスト) を参照してください。	2012 年 6 月 7 日
グループとインスタンスへのタグ付け	Auto Scaling グループにタグを付けることができるようになりました。さらに、タグの作成後に起動された EC2 インスタンスにもタグが適用されるように指定します。詳細については、「Auto Scaling Group とインスタンスのタグ付け」を参照してください。	2012 年 1 月 26 日

機能	説明	リリース日
Amazon SNS の Support	Amazon SNS を使用して、Amazon EC2 Auto Scaling による EC2 インスタンスの起動または終了のたびに、通知を受け取ります。詳細については、「Auto Scaling グループのスケール時に SNS 通知の取得」を参照してください。 Amazon EC2 Auto Scaling には以下の新機能も追加されました。 • 構文を使用して反復的なスケーリングアクティビティを設定する機能。詳細については、PutScheduledUpdateGroupAction API オペレーションを参照してください。 • 起動したインスタンスをロードバランサーに追加せずにスケールアウトできる新しい設定です (LoadBalancer)。詳細については、ProcessType API データ型を参照してください。 • Amazon EC2 Auto Scalingに、インスタンスが最初に終了するのを待たずに、インスタンスに関連付けられたAuto Scalingグループを削除するよう指示するDeleteAutoScalingGroup オペレーション内の ForceDelete フラグ。詳細については、DeleteAutoScalingGroup API オペレーションを参照してください。	2011 年 7 月 20 日
スケジュールされたスケーリングアクション	スケジュールに基づくスケーリングアクションのために追加されたサポート。詳細については、「Amazon EC2 Auto Scaling のスケジュールされたスケーリング」を参照してください。	2010 年 12 月 2 日
Amazon VPC の Support	Amazon VPCの追加されたSupport 詳細については「VPC での Auto Scaling インスタンスの起動」を参照してください。	2010 年 12 月 2 日

機能	説明	リリース日
HPC クラスターのサポート	ハイパフォーマンスコンピューティング (HPC) クラスターのサポートが追加されました。	2010 年 12 月 2 日
ヘルスチェックのサポート	Amazon EC2 Auto Scaling マネージドEC2 インスタンスで Elastic Load Balancing ヘルスチェックを使用するSupportが追加されました。詳細については、「Auto Scaling グループのインスタンスのHealth チェック」を参照してください。	2010 年 12 月 2 日
CloudWatch アラームのSupport	古いトリガーメカニズムを削除し、 CloudWatch アラーム機能を使用するように Amazon EC2 Auto Scaling を再設計しました。詳細については、「Amazon EC2 Auto Scaling のDynamcスケーリング」を参照してください。	2010 年 12 月 2 日
スケーリングの一時停止と再開	スケーリングプロセスを停止および再開するために追加されたサポート。	2010 年 12 月 2 日
IAM のSupport	IAM のサポートが追加されました。詳細については、Amazon EC2 Auto Scaling リソースへのアクセスを制御を参照してください。	2010 年 12 月 2 日

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。