デベロッパーガイド

AWS Cloud Map

AWS Cloud Map: デベロッパーガイド

Table of Contents

とは AWS Cloud Map 1
のコンポーネント AWS Cloud Map 1
アクセス AWS Cloud Map 2
AWS Identity and Access Management 4
AWS Cloud Map 料金 4
AWS Cloud Map および AWS クラウドコンプライアンス 5
はじめに 6
セットアップする 6
にサインアップする AWS 7
API、 AWS CLI AWS Tools for Windows PowerShell、または AWS SDKsにアクセスする 9
AWS Command Line Interface または をセットアップする AWS Tools for Windows PowerShell 11
AWS SDK のダウンロード 11
DNS クエリと API コール AWS Cloud Map で を使用する 11
前提条件 11
ステップ 1: 名前空間を作成する 12
ステップ 2: サービスを作成する 13
ステップ 3: サービスインスタンスを作成する 14
ステップ 4: サービスインスタンスを検出する 15
ステップ 5 : クリーンアップ 16
を使用して DNS クエリと API コールで AWS Cloud Map サービス検出を使用する AWS CLI ... 17
前提条件 17
AWS Cloud Map 名前空間を作成する 17
AWS Cloud Map サービスを作成する 18
AWS Cloud Map サービスインスタンスを登録する 20
AWS Cloud Map サービスインスタンスを検出する 21
リソースをクリーンアップする 23
カスタム属性 AWS Cloud Map で を使用する 24
前提条件 24
ステップ 1: 名前空間を作成する 24
ステップ 2: DynamoDB テーブルを作成する 25
ステップ 3: データサービスを作成する 25
ステップ 4: 実行ロールを作成する 26
ステップ 5: データを書き込む Lambda 関数を作成する 26

ステップ 6: アプリサービスを作成する 28
ステップ 7: データを読み取る Lambda 関数を作成する 29
ステップ 8: サービスインスタンスを作成する 30
ステップ 9: クライアントアプリケーションを作成して実行する 30
ステップ 10: クリーンアップする 33
を使用してカスタム属性で AWS Cloud Map サービス検出を使用する AWS CLI 34
前提条件 34
AWS Cloud Map 名前空間を作成する 34
DynamoDB テーブルを作成する 35
AWS Cloud Map データサービスを作成し、DynamoDB テーブルを登録する 36
Lambda 関数の IAM ロールを作成する 36
Lambda 関数を作成してデータを書き込む 38
AWS Cloud Map アプリケーションサービスを作成し、Lambda 書き込み関数を登録する 40
Lambda 関数を作成してデータを読み取る 41
Lambda 読み取り関数をサービスインスタンスとして登録する 43
クライアントアプリケーションを作成して実行する 43
リソースをクリーンアップする 46
名前空間 48
ネームスペースの作成 48
インスタンス検出オプション 49
手順 53
次の手順 56
名前空間の一覧表示 56
名前空間の削除 59
共有名前空間 60
名前空間の共有に関する考慮事項 61
AWS Cloud Map 名前空間の共有 62
AWS Cloud Map 名前空間の共有を停止する 63
共有 AWS Cloud Map 名前空間の識別 63
名前空間を共有するアクセス許可の付与 65
共有名前空間の責任とアクセス許可 66
請求と使用量測定 66
クォータ 67
サービス 68
ヘルスチェックの設定 69
Route 53 ヘルスチェック 69

カスタムヘルスチェック 70
DNS 設定 71
ルーティングポリシー 71
レコードタイプ 72
サービスの作成 74
次の手順 79
サービスの更新 79
名前空間でのサービスの一覧表示 81
サービスの削除 83
サービスインスタンス 86
サービスインスタンスの登録 86
サービスインスタンスの一覧表示 92
サービスインスタンスの更新 94
サービスインスタンスのカスタム属性の更新 94
サービスインスタンスの登録解除 95
セキュリティ 97
Identity and Access Management 97
オーディエンス 98
アイデンティティを使用した認証 98
ポリシーを使用したアクセスの管理 100
が IAM と AWS Cloud Map 連携する方法 101
アイデンティティベースのポリシーの例 108
AWS マネージドポリシー 115
AWS Cloud Map API アクセス許可リファレンス 117
トラブルシューティング 121
コンプライアンス検証 123
耐障害性 123
インフラストラクチャセキュリティ 123
AWS PrivateLink 124
モニタリング 127
を使用した AWS Cloud Map API コールのログ記録 AWS CloudTrail 127
データイベント 129
管理イベント 130
イベント例 130
リソースのタグ付け 134
リソースのタグ付け方法 134

制限事項 135
AWS Cloud Map リソースのタグの更新 136
Service Quotas 138
サービスクォータの管理 139
DiscoverInstances API リクエストスロットリングの処理 140
スロットリングの適用方法 141
API スロットリングのクォータの調整 142
ドキュメント履歴 143
..... cxlvi

とは AWS Cloud Map

AWS Cloud Map は、アプリケーションが依存するバックエンドサービスとリソースに論理名をマッピングするために使用できるフルマネージドソリューションです。また、アプリケーションが AWS SDKs、RESTful API コール、または DNS クエリのいずれかを使用してリソースを検出するのに役立ちます。 は、Amazon DynamoDB (DynamoDB) テーブル、Amazon Simple Queue Service (Amazon SQS) キュー、Amazon Elastic Compute Cloud (Amazon EC2) インスタンスまたは Amazon Elastic Container Service (Amazon ECS) タスクを使用して構築された上位レベルのアプリケーションサービスなど、正常なリソースのみ AWS Cloud Map を提供します。

のコンポーネント AWS Cloud Map

名前空間

開始するには、まずアプリケーションのサービスをグループ化する方法として機能する AWS Cloud Map 名前空間を作成します。名前空間は、リソースの検索に使用する名前を識別し、リソースの検索方法を指定します。 AWS Cloud Map DiscoverInstances API コール、VPC 内の DNS クエリ、またはパブリック DNS クエリを使用します。通常、1 つの名前空間には 1 つのアプリケーション (例: 請求アプリケーション) のサービスがすべて含まれています。詳細については、「AWS Cloud Map 名前空間」を参照してください。

サービス

名前空間を作成したら、エンドポイント AWS Cloud Map の検索に使用するリソースのタイプごとに AWS Cloud Map サービスを作成します。たとえば、ウェブサーバーおよびデータベースサーバーのサービスを作成します。

サービスは、アプリケーションが別のウェブサーバーなどの別のリソースを追加するときに が AWS Cloud Map 使用するテンプレートです。名前空間を作成したときに DNS を使用してリソースを検索することを選択した場合、サービスにはウェブサーバーの検索に使用するレコードの種類に関する情報が含まれています。サービスは、リソースの状態を確認するかどうか、および Amazon Route 53 ヘルスチェックとサードパーティーのヘルスチェッカーのどちらを使用するかも示します。詳細については、「AWS Cloud Map サービス」を参照してください。

サービスインスタンス

アプリケーションがリソースを追加すると、コードで AWS Cloud Map RegisterInstance API アクションを呼び出して、サービスに AWS Cloud Map サービスインスタンスを作成できます。

サービスインスタンスには、DNS を使用するか、 AWS Cloud Map DiscoverInstances API アクションを使用するかにかかわらず、アプリケーションがリソースを見つける方法に関する情報が含まれています。

アプリケーションがリソースに接続する必要がある場合、DiscoverInstances を呼び出すか、リソースに関連付けられている名前空間とサービスを指定してパブリックまたはプライベート DNS クエリを使用します。 は、1 つ以上のリソースを見つける方法に関する情報 AWS Cloud Map を返します。サービスの作成時にヘルスチェックを指定した場合、 は正常なインスタンスのみ AWS Cloud Map を返します。詳細については、「AWS Cloud Map サービスインスタンス」を参照してください。

アクセス AWS Cloud Map

には AWS Cloud Map 、次の方法でアクセスできます。

- AWS マネジメントコンソール – このガイドの手順では、 を使用してタスク AWS マネジメントコンソール を実行する方法について説明します。
- AWS SDKs – SDK AWS を提供するプログラミング言語を使用している場合は、 SDK を使用してにアクセスできます AWS Cloud Map。SDK によって認証が簡素化され、開発環境との統合が容易になり、 AWS Cloud Map コマンドにアクセスすることができます。詳細については、「Amazon ウェブ サービスのツール」を参照してください。
- AWS Command Line Interface – 詳細については、「 ユーザーガイド」の「 の開始方法 AWS CLI」を参照してください。 AWS Command Line Interface
- AWS Tools for Windows PowerShell – 詳細については、「 ユーザーガイド」の「 の開始方法 AWS Tools for Windows PowerShell」を参照してください。 AWS Tools for PowerShell
- AWS Cloud Map API – SDK が使用できないプログラミング言語を使用している場合は、 API アクションと API リクエストの作成方法については、 API AWS Cloud Map リファレンスを参照してください。

> **Note**
>
> IPv6 クライアントサポート – 2023 年 6 月 22 日現在、すべての新しいリージョンで、IPv6クライアント AWS Cloud Map から に送信されたコマンドは、新しいデュアルスタックエンドポイント () にルーティングされますservicediscovery.<region>.api.aws。 AWS Cloud Map IPv6専用ネットワークは、2023 年 6 月 22 日より前にリリースされた以下のリージョンのレガシー

(servicediscovery.<region>.amazonaws.com) とデュアルスタックエンドポイントの両方にアクセスできます。

- 米国東部 (オハイオ) – us-east-2
- 米国東部 (バージニア北部) – us-east-1
- 米国西部 (北カリフォルニア) – us-west-1
- 米国西部 (オレゴン) – us-west-2
- アフリカ (ケープタウン) – af-south-1
- アジアパシフィック (香港) – ap-east-1
- アジアパシフィック (ハイデラバード) ap-south-2
- アジアパシフィック (ジャカルタ) – ap-southeast-3
- アジアパシフィック (メルボルン) ap-southeast-4
- アジアパシフィック (ムンバイ) – ap-south-1
- アジアパシフィック (大阪) - ap-northeast-3
- アジアパシフィック (ソウル) – ap-northeast-2
- アジアパシフィック (シンガポール) – ap-southeast-1
- アジアパシフィック (シドニー) – ap-southeast-2
- アジアパシフィック (東京) – ap-northeast-1
- カナダ (中部) – ca-central-1
- 欧州 (フランクフルト) – eu-central-1
- 欧州 (アイルランド) – eu-west-1
- 欧州 (ロンドン) – eu-west-2
- 欧州 (ミラノ) – eu-south-1
- 欧州 (パリ) – eu-west-3
- 欧州 (スペイン) eu-south-2
- 欧州 (ストックホルム) - eu-north-1
- 欧州 (チューリッヒ) eu-central-2
- 中東 (バーレーン) – me-south-1
- 中東 (UAE) me-central-1
- 南米 (サンパウロ) – sa-east-1
- AWS GovCloud (米国東部) – us-gov-east-1

- AWS GovCloud (米国西部) – us-gov-west-1

AWS Identity and Access Management

AWS Cloud Map は、組織が以下のアクションを実行するために使用できるサービスである AWS Identity and Access Management (IAM) と統合されます。

- 組織の AWS アカウントでユーザーとグループを作成する
- AWS アカウントのユーザー間でアカウントリソースを効率的に共有する
- 各ユーザーに一意のセキュリティ認証情報を割り当てる
- サービスやリソースに対するユーザーのアクセス権を細分化して制御する

たとえば、 で IAM を使用して AWS Cloud Map 、 AWS アカウント内のどのユーザーが新しい名前空間を作成またはインスタンスを登録できるかを制御できます。

IAM の一般的な情報については、以下のリソースを参照してください。

- の Identity and Access Management AWS Cloud Map
- AWS Identity and Access Management
- IAM ユーザーガイド

AWS Cloud Map 料金

AWS Cloud Map の料金は、サービスレジストリに登録したリソースと、それらを検出するために行う API コールに基づいています。 AWS Cloud Map では前払いはなく、使用した分のみお支払いいただきます。

必要に応じて、IP アドレスを持つリソースに対して DNS ベースの検出を有効にできます。また、Amazon Route 53 ヘルスチェックを使用してリソースのヘルスチェックを有効にすることもできます。API コールまたは DNS クエリを使用してインスタンスを検出するかどうかは関係ありません。Route 53 DNS およびヘルスチェックの使用量に応じて、追加料金が発生します。

詳細については、AWS Cloud Map の料金を参照してください。

AWS Cloud Map および AWS クラウドコンプライアンス

さまざまなセキュリティコンプライアンス規制および監査標準への AWS Cloud Map 準拠については、以下のページを参照してください。

- AWS クラウドコンプライアンス
- AWS コンプライアンスプログラムによる対象範囲内のサービス

の開始方法 AWS Cloud Map

以下のガイドでは、 をセットアップして、 AWS Cloud Map 名前空間を使用して一般的なタスク AWS Cloud Map を実行する方法を示します。

ガイドの概要	詳細情報
にサインアップ AWS して を使用する準備をする AWS Cloud Map	を使用するように をセットアップする AWS Cloud Map
DNS クエリと API コールを使用してバックエンドサービスを検出します。	DNS クエリと API コールで AWS Cloud Map サービス検出を使用する方法について説明します。
DNS クエリと API コールを使用して、 を使用してバックエンドサービスを検出します AWS CLI。	を使用して DNS クエリと API コールで AWS Cloud Map サービス検出を使用する方法について説明します。 AWS CLI
サンプルアプリケーションを作成し、コードでカスタム属性を使用してリソースを検出します。	カスタム属性で AWS Cloud Map サービス検出を使用する方法について説明します。
サンプルアプリケーションを作成し、コードでカスタム属性を使用して を使用してリソースを検出します AWS CLI。	を使用してカスタム属性で AWS Cloud Map サービス検出を使用する方法について説明します。 AWS CLI

を使用するように をセットアップする AWS Cloud Map

以下のセクションの概要と手順は、 の使用を開始し AWS 、使用を開始する準備をするのに役立ちます AWS Cloud Map。

トピック

- にサインアップする AWS
- API、 AWS CLI AWS Tools for Windows PowerShell、または AWS SDKsにアクセスする
- AWS Command Line Interface または をセットアップする AWS Tools for Windows PowerShell
- AWS SDK のダウンロード

にサインアップする AWS

にサインアップする AWS アカウント

がない場合は AWS アカウント、次の手順を実行して作成します。

にサインアップするには AWS アカウント

1. https://portal.aws.amazon.com/billing/signup を開きます。
2. オンラインの手順に従います。

   サインアップ手順の一環として、電話またはテキストメッセージを受け取り、電話キーパッドで検証コードを入力します。

   にサインアップすると AWS アカウント、 AWS アカウントのルートユーザー が作成されます。ルートユーザーには、アカウントのすべての AWS のサービス とリソースへのアクセス権があります。セキュリティベストプラクティスとして、ユーザーに管理アクセス権を割り当て、ルートユーザーアクセスが必要なタスクの実行にはルートユーザーのみを使用するようにしてください。

AWS サインアッププロセスが完了すると、 から確認メールが送信されます。https://aws.amazon.com/ の [マイアカウント] をクリックして、いつでもアカウントの現在のアクティビティを表示し、アカウントを管理することができます。

管理アクセスを持つユーザーを作成する

にサインアップしたら AWS アカウント、日常的なタスクにルートユーザーを使用しないように AWS アカウントのルートユーザー、 を保護し AWS IAM アイデンティティセンター、 を有効にして管理ユーザーを作成します。

を保護する AWS アカウントのルートユーザー

1. ルートユーザーを選択し、 AWS アカウント E メールアドレスを入力して、アカウント所有者AWS マネジメントコンソールとして にサインインします。次のページでパスワードを入力します。

   ルートユーザーを使用してサインインする方法については、「AWS サインイン ユーザーガイド」の「ルートユーザーとしてサインインする」を参照してください。
2. ルートユーザーの多要素認証 (MFA) を有効にします。

手順については、IAM ユーザーガイドの AWS アカウント「ルートユーザー (コンソール) の仮想 MFA デバイスを有効にする」を参照してください。

管理アクセスを持つユーザーを作成する

1. IAM アイデンティティセンターを有効にします。

   手順については、「AWS IAM アイデンティティセンター ユーザーガイド」の「AWS IAM アイデンティティセンターの有効化」を参照してください。
2. IAM アイデンティティセンターで、ユーザーに管理アクセスを付与します。

   を ID ソース IAM アイデンティティセンターディレクトリ として使用する方法のチュートリアルについては、AWS IAM アイデンティティセンター「 ユーザーガイド」の「デフォルトを使用してユーザーアクセスを設定する IAM アイデンティティセンターディレクトリ」を参照してください。

管理アクセス権を持つユーザーとしてサインインする

- IAM アイデンティティセンターのユーザーとしてサインインするには、IAM アイデンティティセンターのユーザーの作成時に E メールアドレスに送信されたサインイン URL を使用します。

  IAM Identity Center ユーザーを使用してサインインする方法については、AWS サインイン「 ユーザーガイド」の AWS 「 アクセスポータルにサインインする」を参照してください。

追加のユーザーにアクセス権を割り当てる

1. IAM アイデンティティセンターで、最小特権のアクセス許可を適用するというベストプラクティスに従ったアクセス許可セットを作成します。

   手順については、「AWS IAM アイデンティティセンター ユーザーガイド」の「アクセス許可セットを作成する」を参照してください。
2. グループにユーザーを割り当て、そのグループにシングルサインオンアクセス権を割り当てます。

   手順については、「AWS IAM アイデンティティセンター ユーザーガイド」の「グループを追加する」を参照してください。

API、 AWS CLI AWS Tools for Windows PowerShell、または AWS SDKs にアクセスする

API、 AWS CLI AWS Tools for Windows PowerShell、または AWS SDKs を使用するには、アクセスキーを作成する必要があります。これらのキーはアクセスキー ID とシークレットアクセスキーで構成されており、 AWSに行うプログラム的なリクエストに署名するために使用されます。

ユーザーが の AWS 外部で を操作する場合は、プログラムによるアクセスが必要です AWS マネジメントコンソール。プログラムによるアクセスを許可する方法は、 がアクセスするユーザーのタイプによって異なります AWS。

ユーザーにプログラムによるアクセス権を付与するには、以下のいずれかのオプションを選択します。

プログラムによるアクセス権を必要とするユーザー	目的	方法
IAM	(推奨) コンソール認証情報を一時的な認証情報として使用して AWS CLI、、 AWS SDKs、または AWS APIs。	使用するインターフェイスの指示に従ってください。 • については AWS CLI、AWS Command Line Interface「 ユーザーガイド」のAWS「ローカル開発用ログイン」を参照してください。 • AWS SDKs「 SDK およびツールリファレンスガイド」の「Login for AWS local development」を参照してください。 AWS SDKs
ワークフォースアイデンティティ (IAM アイデンティティセンターで管理されているユーザー)	一時的な認証情報を使用して AWS CLI、、 AWS SDKs、または AWS APIs。	使用するインターフェイスの指示に従ってください。 • については AWS CLI、「 AWS Command Line Interface ユーザーガイド」の「 を使用する AWS CLI

プログラムによるアクセス権を必要とするユーザー	目的	方法
		ように を設定する AWS IAM アイデンティティセンター」を参照してください。 • AWS SDKs、ツール、API については、 AWS APIs「 SDK およびツールリファレンスガイド」の「IAM アイデンティティセンター認証」を参照してください。 AWS SDKs
IAM	一時的な認証情報を使用して AWS CLI、、 AWS SDKs、または AWS APIs。	「IAM ユーザーガイド」の「 AWS リソースでの一時的な認証情報の使用」の手順に従います。
IAM	(非推奨) 長期認証情報を使用して、 AWS CLI、 AWS SDKs、または AWS APIs。	使用するインターフェイスの指示に従ってください。 • については AWS CLI、「 AWS Command Line Interface ユーザーガイド」の「IAM ユーザー認証情報を使用した認証」を参照してください。 • AWS SDKs「 SDK とツールリファレンスガイド」の「長期認証情報を使用した認証」を参照してください。 AWS SDKs • API AWS APIs「IAM ユーザーのアクセスキーの管理」を参照してください。

AWS Command Line Interface または をセットアップする AWS Tools for Windows PowerShell

AWS Command Line Interface (AWS CLI) は、 AWS のサービスを管理するための統合ツールです。をインストールして設定する方法については AWS CLI、「 AWS Command Line Interface 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。

Windows PowerShell の使用経験がある場合は、 AWS Tools for Windows PowerShellを使用することもできます。詳細については、AWS Tools for PowerShell ユーザーガイドの「AWS Tools for Windows PowerShellのセットアップ」を参照してください。

AWS SDK のダウンロード

SDK AWS を提供するプログラミング言語を使用している場合は、 AWS Cloud Map API の代わりに SDK を使用することをお勧めします。SDK の使用には、いくつかの利点があります。SDK によって認証が簡素化され、開発環境との統合が容易になり、 AWS Cloud Map コマンドにアクセスすることができます。詳細については、Tools for Amazon Web Services を参照してください。

DNS クエリと API コールで AWS Cloud Map サービス検出を使用する方法について説明します。

次のチュートリアルでは、2 つのバックエンドサービスを使用したマイクロサービスアーキテクチャをシミュレートします。最初のサービスは、DNS クエリを使用して検出できます。2 番目のサービスは API を使用して AWS Cloud Map のみ検出できます。

> **Note**
>
> ドメイン名や IP アドレスなどのリソースの詳細は、シミュレーションのみを目的としています。インターネット経由で解決することはできません。

このチュートリアルのend-to-end AWS CLI のバージョンについては、「」を参照してくださいを使用して DNS クエリと API コールで AWS Cloud Map サービス検出を使用する方法について説明します。 AWS CLI。

前提条件

チュートリアルを正常に完了するには、次の前提条件を満たす必要があります。

- 開始する前に、「を使用するように をセットアップする AWS Cloud Map」のステップを完了します。
- をまだインストールしていない場合は AWS Command Line Interface、「 の最新バージョンのインストールまたは更新 AWS CLI」の手順に従ってインストールします。

  このチュートリアルでは、コマンドを実行するためのコマンドラインターミナルまたはシェルが必要です。Linux および macOS では、任意のシェルとパッケージマネージャーを使用してください。

  > **Note**
  >
  > Windows では、Lambda でよく使用される一部の Bash CLI コマンド (zip など) が、オペレーティングシステムの組み込みターミナルでサポートされていません。Ubuntu および Bash の Windows 統合バージョンを取得するには、Windows Subsystem for Linux をインストールします。

- このチュートリアルでは、digDNS ルックアップユーティリティコマンドを使用するローカル環境が必要です。

ステップ 1: AWS Cloud Map 名前空間を作成する

このステップでは、パブリック AWS Cloud Map 名前空間を作成します。 は、同じ名前でユーザーに代わって Route 53 ホストゾーン AWS Cloud Map を作成します。これにより、パブリック DNS レコードまたは AWS Cloud Map API コールを使用して、この名前空間で作成されたサービスインスタンスを検出できます。

1. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。
2. [名前空間の作成] を選択します。
3. 名前空間名には、 を指定しますcloudmap-tutorial.com。

   > **Note**
   >
   > これを本番環境で使用する場合は、所有またはアクセス権のあるドメインの名前を指定する必要があります。ただし、このチュートリアルでは、実際に使用されているドメインである必要はありません。

4. (オプション) 名前空間の説明に、名前空間を使用する対象の説明を指定します。
5. インスタンス検出では、API コールとパブリック DNS クエリを選択します。
6. 残りのデフォルト値のままにして、名前空間の作成を選択します。

ステップ 2: AWS Cloud Map サービスを作成する

このステップでは、2 つのサービスを作成します。最初のサービスは、パブリック DNS コールと API コールを使用して検出できます。2 番目のサービスは API コールを使用してのみ検出できます。

1. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。
2. 左側のナビゲーションペインで、名前空間を選択して、作成した名前空間を一覧表示します。
3. 名前空間のリストから`cloudmap-tutorial.com`名前空間を選択し、詳細の表示を選択します。
4. 「サービス」セクションで「サービスの作成」を選択し、以下を実行して最初のサービスを作成します。

   a. [サービス名] に `public-service` と入力します。サービス名は、 が AWS Cloud Map 作成する DNS レコードに適用されます。使用される形式は です*`<service-name>`*.*`<namespace-name>`*。

   b. サービス検出設定で、API と DNS を選択します。

   c. DNS 設定セクションのルーティングポリシーで、複数値回答ルーティングを選択します。

   > **Note**
   >
   > 選択すると、コンソールはこれを MULTIVALUE に変換します。使用可能なルーティングオプションの詳細については、Route 53 デベロッパーガイドの「ルーティングポリシーの選択」を参照してください。

   d. 残りのデフォルト値のままにして、名前空間の詳細ページに戻るサービスの作成を選択します。
5. 「サービス」セクションで「サービスの作成」を選択し、以下を実行して 2 番目のサービスを作成します。

   a. [サービス名] に `backend-service` と入力します。

   b. Service Discovery Configuration では、API のみを選択します。

c. 残りのデフォルト値のままにして、サービスの作成を選択します。

ステップ 3: AWS Cloud Map サービスインスタンスを登録する

このステップでは、名前空間内のサービスごとに 1 つずつ、2 つのサービスインスタンスを作成します。

1. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。
2. 名前空間のリストから、ステップ 1 で作成した名前空間を選択し、詳細の表示を選択します。
3. 名前空間の詳細ページで、サービスのリストからpublic-serviceサービスを選択し、詳細の表示を選択します。
4. 「サービスインスタンス」セクションで「サービスインスタンスの登録」を選択し、以下を実行して最初のサービスインスタンスを作成します。

   a. サービスインスタンス ID には、 を指定しますfirst。

   b. IPv4 アドレスには、 を指定します192.168.2.1。

   c. 残りのデフォルト値のままにして、サービスインスタンスの登録を選択します。
5. ページの上部にあるパンくずリストを使用して、cloudmap-tutorial.com を選択して名前空間の詳細ページに戻ります。
6. 名前空間の詳細ページで、サービスのリストからバックエンドサービスを選択し、詳細の表示を選択します。
7. 「サービスインスタンス」セクションで「サービスインスタンスの登録」を選択し、以下を実行して 2 番目のサービスインスタンスを作成します。

   a. サービスインスタンス ID で、 を指定secondして、これが 2 番目のサービスインスタンスであることを示します。

   b. インスタンスタイプで、別のリソースの識別情報を選択します。

   c. カスタム属性の場合は、 をキーservice-nameとして、 を値backendとしてキーと値のペアを追加します。

   d. [サービスインスタンスの登録] を選択します。

ステップ 4: AWS Cloud Map サービスインスタンスを検出する

AWS Cloud Map 名前空間、サービス、サービスインスタンスが作成されたら、インスタンスを検出することで、すべてが機能していることを確認できます。dig コマンドを使用してパブリック DNS 設定を確認し、 AWS Cloud Map API を使用してバックエンドサービスを確認します。dig コマンドの詳細については、「dig - DNS lookup utility」を参照してください。

1. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/route53/ で Route 53 コンソールを開きます。
2. 左のナビゲーションで、[Hosted zones] (ホストゾーン) を選択します。
3. cloudmap-tutorial.com ホストゾーンを選択します。これにより、ホストゾーンの詳細が別のペインに表示されます。ホストゾーンに関連付けられたネームサーバーをメモしておきます。次のステップで使用します。
4. dig コマンドとホストゾーンの Route 53 ネームサーバーのいずれかを使用して、サービスインスタンスの DNS レコードをクエリします。

   ```
   dig @hosted-zone-nameserver public-service.cloudmap-tutorial.com
   ```

   出力ANSWER SECTIONの には、public-serviceサービスに関連付けられた IPv4 アドレスが表示されます。

   ```
   ;; ANSWER SECTION:
   public-service.cloudmap-tutorial.com. 300 IN A 192.168.2.1
   ```

5. を使用して AWS CLI、2 番目のサービスインスタンスの属性をクエリします。

   ```
   aws servicediscovery discover-instances --namespace-name cloudmap-tutorial.com --
   service-name backend-service --region region
   ```

   出力には、サービスに関連付けられた属性がキーと値のペアとして表示されます。

   ```
   {
       "Instances": [
           {
               "InstanceId": "second",
               "NamespaceName": "cloudmap-tutorial.com",
               "ServiceName": "backend-service",
               "HealthStatus": "UNKNOWN",
               "Attributes": {
   ```

```
                "service-name": "backend"
            }
        }
    ],
    "InstancesRevision": 71462688285136850
}
```

ステップ 5: リソースをクリーンアップする

チュートリアルを完了したら、リソースを削除できます。 AWS Cloud Map は、逆の順序でクリーンアップし、サービスインスタンスを最初に、次にサービス、最後に 名前空間をクリーンアップする必要があります。 AWS Cloud Map は、これらのステップを実行すると、ユーザーに代わって Route 53 リソースをクリーンアップします。

1. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。
2. 名前空間のリストから、`cloudmap-tutorial.com`名前空間を選択し、詳細の表示を選択します。
3. 名前空間の詳細ページで、サービスのリストから`public-service`サービスを選択し、詳細の表示を選択します。
4. サービスインスタンスセクションで、`first`インスタンスを選択し、登録解除を選択します。
5. ページの上部にあるパンくずリストを使用して、cloudmap-tutorial.com を選択して名前空間の詳細ページに戻ります。
6. 名前空間の詳細ページで、サービスのリストからパブリックサービスを選択し、削除を選択します。
7. に対してステップ 3～6 を繰り返します`backend-service`。
8. 左側のナビゲーションで、名前空間を選択します。
9. `cloudmap-tutorial.com` 名前空間を選択し、削除を選択します。

> **Note**
>
> はユーザーに代わって Route 53 リソースを AWS Cloud Map クリーンアップしますが、Route 53 コンソールに移動して、`cloudmap-tutorial.com`ホストゾーンが削除されたことを確認できます。

を使用して DNS クエリと API コールで AWS Cloud Map サービス検出を使用する方法について説明します。 AWS CLI

このチュートリアルでは、 AWS Command Line Interface (CLI) を使用して AWS Cloud Map サービス検出を使用する方法を示します。2 つのバックエンドサービスを使用してマイクロサービスアーキテクチャを作成します。1 つは DNS クエリを使用して検出可能、もう 1 つは AWS Cloud Map API のみを使用して検出可能になります。

AWS Cloud Map コンソールの手順を含むチュートリアルについては、「」を参照してくださいDNS クエリと API コールで AWS Cloud Map サービス検出を使用する方法について説明します。。

前提条件

チュートリアルを正常に完了するには、次の前提条件を満たす必要があります。

- 開始する前に、「を使用するように をセットアップする AWS Cloud Map」のステップを完了します。
- をまだインストールしていない場合は AWS Command Line Interface、「 の最新バージョンのインストールまたは更新 AWS CLI」の手順に従ってインストールします。

  このチュートリアルでは、コマンドを実行するためのコマンドラインターミナルまたはシェルが必要です。Linux および macOS では、任意のシェルとパッケージマネージャーを使用してください。

  > **Note**
  >
  > Windows では、Lambda でよく使用される一部の Bash CLI コマンド (`zip` など) が、オペレーティングシステムの組み込みターミナルでサポートされていません。Ubuntu および Bash の Windows 統合バージョンを取得するには、Windows Subsystem for Linux をインストールします。

- このチュートリアルでは、`dig`DNS ルックアップユーティリティコマンドを使用するローカル環境が必要です。

AWS Cloud Map 名前空間を作成する

まず、パブリック AWS Cloud Map 名前空間を作成します。 AWS Cloud Map は、同じ名前の Route 53 ホストゾーンを作成し、DNS レコードと API コールの両方でサービス検出を有効にします。

1. パブリック DNS 名前空間を作成します。

```
aws servicediscovery create-public-dns-namespace \
    --name cloudmap-tutorial.com \
    --creator-request-id cloudmap-tutorial-request-1 \
    --region us-east-2
```

コマンドは、名前空間作成のステータスをチェックするために使用できるオペレーション ID を返します。

```
{
    "OperationId": "gv4g5meo7ndmeh4fqskygvk23d2fijwa-k9xmplyzd"
}
```

2. オペレーションステータスをチェックして、名前空間が正常に作成されたことを確認します。

```
aws servicediscovery get-operation \
    --operation-id gv4g5meo7ndmeh4fqskygvk23d2fijwa-k9xmplyzd \
    --region us-east-2
```

3. オペレーションが成功したら、名前空間 ID を取得します。

```
aws servicediscovery list-namespaces \
    --region us-east-2 \
    --query "Namespaces[?Name=='cloudmap-tutorial.com'].Id" \
    --output text
```

このコマンドは、後続のステップに必要な名前空間 ID を返します。

```
ns-abcd1234xmplefgh
```

AWS Cloud Map サービスを作成する

次に、名前空間内に 2 つのサービスを作成します。最初のサービスは DNS コールと API コールの両方を使用して検出でき、2 番目のサービスは API コールのみを使用して検出できます。

1. DNS 検出を有効にして最初のサービスを作成します。

```
aws servicediscovery create-service \
```

```
    --name public-service \
    --namespace-id ns-abcd1234xmplefgh \
    --dns-config "RoutingPolicy=MULTIVALUE,DnsRecords=[{Type=A,TTL=300}]" \
    --region us-east-2
```

コマンドは、作成されたサービスの詳細を返します。

```
{
    "Service": {
        "Id": "srv-abcd1234xmplefgh",
        "Arn": "arn:aws:servicediscovery:us-east-2:123456789012:service/srv-
abcd1234xmplefgh",
        "Name": "public-service",
        "NamespaceId": "ns-abcd1234xmplefgh",
        "DnsConfig": {
            "NamespaceId": "ns-abcd1234xmplefgh",
            "RoutingPolicy": "MULTIVALUE",
            "DnsRecords": [
                {
                    "Type": "A",
                    "TTL": 300
                }
            ]
        },
        "CreateDate": 1673613600.000,
        "CreatorRequestId": "public-service-request"
    }
}
```

2. API のみの検出を使用して 2 番目のサービスを作成します。

```
aws servicediscovery create-service \
    --name backend-service \
    --namespace-id ns-abcd1234xmplefgh \
    --type HTTP \
    --region us-east-2
```

コマンドは、作成されたサービスの詳細を返します。

```
{
    "Service": {
        "Id": "srv-ijkl5678xmplmnop",
```

```
        "Arn": "arn:aws:servicediscovery:us-east-2:123456789012:service/srv-
ijkl5678xmplmnop",
        "Name": "backend-service",
        "NamespaceId": "ns-abcd1234xmplefgh",
        "Type": "HTTP",
        "CreateDate": 1673613600.000,
        "CreatorRequestId": "backend-service-request"
    }
}
```

AWS Cloud Map サービスインスタンスを登録する

次に、各サービスのサービスインスタンスを登録します。これらのインスタンスは、検出される実際のリソースを表します。

1. DNS 検出用の IPv4 アドレスに最初のインスタンスを登録します。

```
aws servicediscovery register-instance \
    --service-id srv-abcd1234xmplefgh \
    --instance-id first \
    --attributes AWS_INSTANCE_IPV4=192.168.2.1 \
    --region us-east-2
```

コマンドはオペレーション ID を返します。

```
{
    "OperationId": "4yejorelbukcjzpnr6tlmrghsjwpngf4-k9xmplyzd"
}
```

2. オペレーションステータスをチェックして、インスタンスが正常に登録されたことを確認します。

```
aws servicediscovery get-operation \
    --operation-id 4yejorelbukcjzpnr6tlmrghsjwpngf4-k9xmplyzd \
    --region us-east-2
```

3. 2 番目のインスタンスを API 検出用のカスタム属性に登録します。

```
aws servicediscovery register-instance \
    --service-id srv-ijkl5678xmplmnop \
```

```
    --instance-id second \
    --attributes service-name=backend \
    --region us-east-2
```

コマンドはオペレーション ID を返します。

```
{
    "OperationId": "7zxcvbnmasdfghjklqwertyuiop1234-k9xmplyzd"
}
```

4. オペレーションステータスをチェックして、インスタンスが正常に登録されたことを確認します。

```
aws servicediscovery get-operation \
    --operation-id 7zxcvbnmasdfghjklqwertyuiop1234-k9xmplyzd \
    --region us-east-2
```

AWS Cloud Map サービスインスタンスを検出する

サービスインスタンスを作成して登録したので、DNS クエリと AWS Cloud Map API の両方を使用してそれらを検出することで、すべてが機能していることを確認できます。

1. まず、Route 53 ホストゾーン ID を取得します。

```
aws route53 list-hosted-zones-by-name \
    --dns-name cloudmap-tutorial.com \
    --query "HostedZones[0].Id" \
    --output text
```

これにより、ホストゾーン ID が返されます。

```
/hostedzone/Z1234ABCDXMPLEFGH
```

2. ホストゾーンのネームサーバーを取得します。

```
aws route53 get-hosted-zone \
    --id Z1234ABCDXMPLEFGH \
    --query "DelegationSet.NameServers[0]" \
    --output text
```

これにより、ネームサーバーの 1 つが返されます。

```
ns-1234.awsdns-12.org
```

3. dig コマンドを使用して、パブリックサービスの DNS レコードをクエリします。

```
dig @ns-1234.awsdns-12.org public-service.cloudmap-tutorial.com
```

出力には、サービスに関連付けられた IPv4 アドレスが表示されます。

```
;; ANSWER SECTION:
public-service.cloudmap-tutorial.com. 300 IN A 192.168.2.1
```

4. を使用してバックエンドサービスインスタンス AWS CLI を検出します。

```
aws servicediscovery discover-instances \
    --namespace-name cloudmap-tutorial.com \
    --service-name backend-service \
    --region us-east-2
```

出力には、サービスに関連付けられた属性が表示されます。

```
{
    "Instances": [
        {
            "InstanceId": "second",
            "NamespaceName": "cloudmap-tutorial.com",
            "ServiceName": "backend-service",
            "HealthStatus": "UNKNOWN",
            "Attributes": {
                "service-name": "backend"
            }
        }
    ],
    "InstancesRevision": 71462688285136850
}
```

リソースをクリーンアップする

チュートリアルを完了したら、料金が発生しないようにリソースをクリーンアップします。 AWS Cloud Map では、サービスインスタンスを最初に、次にサービス、最後に名前空間という逆の順序でクリーンアップする必要があります。

1. 最初のサービスインスタンスの登録を解除します。

```
aws servicediscovery deregister-instance \
    --service-id srv-abcd1234xmplefgh \
    --instance-id first \
    --region us-east-2
```

2. 2 番目のサービスインスタンスの登録を解除します。

```
aws servicediscovery deregister-instance \
    --service-id srv-ijkl5678xmplmnop \
    --instance-id second \
    --region us-east-2
```

3. パブリックサービスを削除します。

```
aws servicediscovery delete-service \
    --id srv-abcd1234xmplefgh \
    --region us-east-2
```

4. バックエンドサービスを削除します。

```
aws servicediscovery delete-service \
    --id srv-ijkl5678xmplmnop \
    --region us-east-2
```

5. 名前空間を削除します。

```
aws servicediscovery delete-namespace \
    --id ns-abcd1234xmplefgh \
    --region us-east-2
```

6. Route 53 ホストゾーンが削除されていることを確認します。

```
aws route53 list-hosted-zones-by-name \
```

```
    --dns-name cloudmap-tutorial.com
```

カスタム属性で AWS Cloud Map サービス検出を使用する方法について説明します。

次のチュートリアルでは、 AWS Cloud Map API を使用して検出可能なカスタム属性で AWS Cloud Map サービス検出を使用する方法を示します。このチュートリアルでは、 を使用してクライアントアプリケーションを作成および実行する手順を説明します AWS CloudShell。アプリケーションは 2 つの Lambda 関数を使用してデータを DynamoDB テーブルに書き込み、テーブルから読み取ります。Lambda 関数と DynamoDB テーブルは、サービスインスタンス AWS Cloud Map として に登録されます。クライアントアプリケーションと Lambda 関数のコードは AWS Cloud Map 、カスタム属性を使用して、ジョブの実行に必要なリソースを検出します。

このチュートリアルの AWS CLIベースのバージョンについては、「」を参照してください[を使用してカスタム属性で AWS Cloud Map サービス検出を使用する方法について説明します。 AWS CLI](#)。

> **Important**
>
> ワークショップ中に AWS リソースを作成し、アカウント AWS でコストが発生します。コストを最小限に抑えるために、ワークショップが終了したらすぐにリソースをクリーンアップすることをお勧めします。

前提条件

開始する前に、「[を使用するように をセットアップする AWS Cloud Map](#)」のステップを完了します。

ステップ 1: AWS Cloud Map 名前空間を作成する

このステップでは、 AWS Cloud Map 名前空間を作成します。名前空間は、アプリケーションのサービスをグループ化するために使用されるコンストラクトです。名前空間を作成するときは、リソースの検出方法を指定します。このステップで作成した名前空間で作成されたリソースは、カスタム属性を使用した AWS Cloud Map API コールで検出できます。

1. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。

2. [名前空間の作成] を選択します。
3. 名前空間名には、 を指定しますcloudmap-tutorial。
4. (オプション) 名前空間の説明には、名前空間を使用する対象の説明を指定します。
5. インスタンス検出では、 API コールを選択します。
6. 残りのデフォルト値のままにして、名前空間の作成を選択します。

ステップ 2: DynamoDB テーブルを作成する

このステップでは、DynamoDB テーブルを作成します。テーブルは、次のステップで作成するサンプルアプリケーションのデータを保存および取得するために使用されます。

DynamoDB の作成方法については、「DynamoDB デベロッパーガイドDynamoDB 」の「ステップ 1: DynamoDB でテーブルを作成する」を参照してください。次の表を使用して、指定するオプションを決定します。 DynamoDB

オプション	値
テーブル名	クラウドマップ
パーティションキー	id

残りの設定のデフォルト値を保持し、テーブルを作成します。

ステップ 3: AWS Cloud Map データサービスを作成し、DynamoDB テーブルをインスタンスとして登録する

このステップでは、 AWS Cloud Map サービスを作成し、最後のステップで作成した DynamoDB テーブルをサービスインスタンスとして登録します。

1. https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。
2. 名前空間のリストから、cloudmap-tutorial名前空間を選択し、詳細の表示を選択します。
3. 「サービス」セクションで「サービスの作成」を選択し、次の操作を行います。
    a. [サービス名] に data-service と入力します。
    b. 残りのデフォルト値のままにして、サービスの作成を選択します。

4. サービスセクションで、data-serviceサービスを選択し、詳細の表示を選択します。
5. 「サービスインスタンス」セクションで、「サービスインスタンスの登録」を選択します。
6. サービスインスタンスの登録ページで、次の操作を行います。
   a. インスタンスタイプで、別のリソースの識別情報を選択します。
   b. サービスインスタンス ID には、 を指定しますdata-instance。
   c. Custom attributes セクションで、key = tablename、value = のキーと値のペアを指定しますcloudmap。

ステップ 4: AWS Lambda 実行ロールを作成する

このステップでは、次のステップで AWS Lambda 関数が使用する IAM ロールを作成します。ロールはこのチュートリアルでのみ使用されcloudmap-tutorial-role、後で削除できるため、IAM ロールに名前を付け、アクセス許可の境界を省略できます。

Lambda のサービスロールを作成するには (IAM コンソール)

1. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/iam/ で IAM コンソールを開きます。
2. IAM コンソールのナビゲーションペインで、[ロール]、[ロールを作成] を選択します。
3. 信頼できるエンティティタイプ で、AWS のサービス を選択します。
4. サービスまたはユースケースでは、Lambda を選択し、Lambda ユースケースを選択します。
5. [次へ] を選択します。
6. PowerUserAccess ポリシーを検索し、ポリシーの横にあるボックスを選択し、次へを選択します。
7. [次へ] を選択します。
8. ロール名には、 を指定しますcloudmap-tutorial-role。
9. ロールを確認したら、[ロールを作成] を選択します。

ステップ 5: データを書き込む Lambda 関数を作成する

このステップでは、 AWS Cloud Map API を使用して作成した AWS Cloud Map サービスをクエリすることで、DynamoDB テーブルにデータを書き込む Lambda 関数を最初から作成します。

Lambda 関数の作成の詳細については、「 AWS Lambda デベロッパーガイド」の「コンソールを使用して Lambda 関数を作成する」を参照してください。次の表を使用して、指定または選択するオプションを決定します。

オプション	値
関数名	書き込み関数
実行時間	Python 3.12
アーキテクチャ	x86_64
アクセス許可	既存のロールを使用する
既存のロール	cloudmap-tutorial-role

関数を作成したら、次の Python コードを反映するようにサンプルコードを更新し、関数をデプロイします。DynamoDB テーブル用に作成した AWS Cloud Map サービスインスタンスに関連付けられたdatatableカスタム属性を指定することに注意してください。関数は 1 ~ 100 の乱数であるキーを生成し、呼び出されたときに関数に渡される値に関連付けます。

```
import json
import boto3
import random

def lambda_handler(event, context):

    serviceclient = boto3.client('servicediscovery')

    response = serviceclient.discover_instances(
        NamespaceName='cloudmap-tutorial',
        ServiceName='data-service')

    tablename = response["Instances"][0]["Attributes"]["tablename"]

    dynamodbclient = boto3.resource('dynamodb')

    table = dynamodbclient.Table(tablename)

    response = table.put_item(
        Item={ 'id': str(random.randint(1,100)), 'todo': event })
```

```
    return {
        'statusCode': 200,
        'body': json.dumps(response)
    }
```

関数をデプロイした後、タイムアウトエラーを回避するには、関数のタイムアウトを 5 秒に更新します。詳細については、AWS Lambda デベロッパーガイドの「Lambda 関数のタイムアウトを設定する」を参照してください。

ステップ 6: AWS Cloud Map アプリケーションサービスを作成し、Lambda 書き込み関数をインスタンスとして登録する

このステップでは、 AWS Cloud Map サービスを作成し、Lambda 書き込み関数をサービスインスタンスとして登録します。

1. https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。
2. 左側のナビゲーションで、名前空間を選択します。
3. 名前空間のリストから、`cloudmap-tutorial`名前空間を選択し、詳細の表示を選択します。
4. 「サービス」セクションで「サービスの作成」を選択し、次の操作を行います。
    a. [サービス名] に `app-service` と入力します。
    b. 残りのデフォルト値のままにし、サービスの作成を選択します。
5. サービスセクションで、`app-service`サービスを選択し、詳細の表示を選択します。
6. 「サービスインスタンス」セクションで、「サービスインスタンスの登録」を選択します。
7. サービスインスタンスの登録ページで、次の操作を行います。
    a. インスタンスタイプで、別のリソースの識別情報を選択します。
    b. サービスインスタンス ID には、 を指定します`write-instance`。
    c. カスタム属性セクションで、次のキーと値のペアを指定します。
        - key = `action`、value = `write`
        - key = `functionname`、value = `writefunction`

ステップ 7: データを読み取る Lambda 関数を作成する

このステップでは、作成した DynamoDB テーブルにデータを書き込む Lambda 関数を最初から作成します。

Lambda 関数の作成の詳細については、「 AWS Lambda デベロッパーガイド」の「コンソールを使用して Lambda 関数を作成する」を参照してください。次の表を使用して、指定または選択するオプションを決定します。

オプション	値
関数名	読み取り関数
実行時間	Python 3.12
アーキテクチャ	x86_64
アクセス許可	既存のロールを使用する
既存のロール	cloudmap-tutorial-role

関数を作成したら、次の Python コードを反映するようにサンプルコードを更新し、関数をデプロイします。関数はテーブル amd をスキャンして、すべての項目を返します。

```
import json
import boto3

def lambda_handler(event, context):
    serviceclient = boto3.client('servicediscovery')

    response = serviceclient.discover_instances(NamespaceName='cloudmap-tutorial',
 ServiceName='data-service')

    tablename = response["Instances"][0]["Attributes"]["tablename"]

    dynamodbclient = boto3.resource('dynamodb')

    table = dynamodbclient.Table(tablename)

    response = table.scan(Select='ALL_ATTRIBUTES')
```

```
    return {
        'statusCode': 200,
        'body': json.dumps(response)
    }
```

関数をデプロイした後、タイムアウトエラーを回避するには、関数のタイムアウトを 5 秒に更新します。詳細については、AWS Lambda デベロッパーガイドの「Lambda 関数のタイムアウトを設定する」を参照してください。

ステップ 8: Lambda 読み取り関数を AWS Cloud Map サービスインスタンスとして登録する

このステップでは、以前に作成したサービスに Lambda 読み取り関数を`app-service`サービスインスタンスとして登録します。

1. https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。
2. 左側のナビゲーションで、名前空間を選択します。
3. 名前空間のリストから、`cloudmap-tutorial`名前空間を選択し、詳細の表示を選択します。
4. サービスセクションで、`app-service`サービスを選択し、詳細の表示を選択します。
5. 「サービスインスタンス」セクションで、「サービスインスタンスの登録」を選択します。
6. サービスインスタンスの登録ページで、次の操作を行います。
    a. インスタンスタイプで、別のリソースの識別情報を選択します。
    b. サービスインスタンス ID には、 を指定します`read-instance`。
    c. カスタム属性セクションで、次のキーと値のペアを指定します。
        - key = `action`、value = `read`
        - key = `functionname`、value = `readfunction`

ステップ 9: で読み取り/書き込みクライアントを作成して実行する AWS CloudShell

コード AWS CloudShell を使用して で設定したサービスを検出し、これらのサービスを呼び出すクライアントアプリケーションを作成 AWS Cloud Map および実行できます。

1. https://console.aws.amazon.com/cloudshell/ で AWS CloudShell コンソールを開きます。

2. 次のコマンドを使用して、 という名前のファイルを作成しますwritefunction.py。

```
vim writeclient.py
```

3. writeclient.py ファイルで、 i ボタンを押して挿入モードを入力します。次に、次のコードをコピーして貼り付けます。このコードは、app-serviceサービスname=writeservice内のカスタム属性を検索してデータを書き込む Lambda 関数を検出します。DynamoDB テーブルへのデータの書き込みを担当する Lambda 関数の名前が返されます。次に、Lambda 関数が呼び出され、値としてテーブルに書き込まれるサンプルペイロードが渡されます。

```
import boto3

serviceclient = boto3.client('servicediscovery')

response = serviceclient.discover_instances(NamespaceName='cloudmap-tutorial',
 ServiceName='app-service', QueryParameters={ 'action': 'write' })

functionname = response["Instances"][0]["Attributes"]["functionname"]

lambdaclient = boto3.client('lambda')

resp = lambdaclient.invoke(FunctionName=functionname, Payload='"This is a test
 data"')

print(resp["Payload"].read())
```

4. エスケープキーを押し、 と入力し:wq、Enter キーを押してファイルを保存し、終了します。
5. Python コードを実行するには、次のコマンドを使用します。

```
python3 writeclient.py
```

出力は、次のような200レスポンスである必要があります。

```
b'{"statusCode": 200, "body": "{\\"ResponseMetadata\\": {\\"RequestId\\": \
\"Q0M038IT0BPBVBJK8OCKK6I6M7VV4KQNSO5AEMVJF66Q9ASUAAJG\\", \\"HTTPStatusCode\
\": 200, \\"HTTPHeaders\\": {\\"server\\": \\"Server\\", \\"date\\": \\"Wed, 06
 Mar 2024 22:46:09 GMT\\", \\"content-type\\": \\"application/x-amz-json-1.0\\",
 \\"content-length\\": \\"2\\", \\"connection\\": \\"keep-alive\\", \\"x-amzn-
requestid\\": \\"Q0M038IT0BPBVBJK8OCKK6I6M7VV4KQNSO5AEMVJF66Q9ASUAAJG\\", \\"x-amz-
crc32\\": \\"2745614147\\"}, \\"RetryAttempts\\": 0}}"}'
```

6. 前のステップで書き込みが成功したことを確認するには、読み取りクライアントを作成します。

   a. 次のコマンドを使用して、 というファイルを作成しますreadfunction.py。

   ```
   vim readclient.py
   ```

   b. readclient.py ファイルで、 i ボタンを押して挿入モードに入ります。次に、次のコードをコピーして貼り付けます。このコードはテーブルをスキャンし、前のステップでテーブルに書き込んだ値を返します。

   ```
   import boto3

   serviceclient = boto3.client('servicediscovery')

   response = serviceclient.discover_instances(NamespaceName='cloudmap-tutorial',
    ServiceName='app-service', QueryParameters={ 'action': 'read' })

   functionname = response["Instances"][0]["Attributes"]["functionname"]

   lambdaclient = boto3.client('lambda')

   resp = lambdaclient.invoke(FunctionName=functionname,
    InvocationType='RequestResponse')

   print(resp["Payload"].read())
   ```

   c. エスケープキーを押し、 と入力し:wq、Enter キーを押してファイルを保存し、終了します。

   d. Python コードを実行するには、次のコマンドを使用します。

   ```
   python3 readclient.py
   ```

   出力は次のようになります。 を実行してテーブルに書き込まれた値writefunction.pyと、Lambda 書き込み関数で生成されたランダムキーを一覧表示します。

   ```
     b'{"statusCode": 200, "body": "{\\"Items\\": [{\\"id\\": \\"45\
   \", \\"todo\\": \\"This is a test data\\"}], \\"Count\\": 1, \
   \"ScannedCount\\": 1, \\"ResponseMetadata\\": {\\"RequestId\\": \
   \"9JF8J6SFQCKR6IDT5JG5NOM3CNVV4KQNSO5AEMVJF66Q9ASUAAJG\\", \\"HTTPStatusCode\
   \": 200, \\"HTTPHeaders\\": {\\"server\\": \\"Server\\", \\"date\\": \\"Thu, 25
   ```

```
 Jul 2024 20:43:33 GMT\\", \\"content-type\\": \\"application/x-amz-json-1.0\
\", \\"content-length\\": \\"91\\", \\"connection\\": \\"keep-alive\\", \\"x-
amzn-requestid\\": \\"9JF8J6SFQCKR6IDT5JG5NOM3CNVV4KQNSO5AEMVJF66Q9ASUAAJG\\",
 \\"x-amz-crc32\\": \\"1163081893\\"}, \\"RetryAttempts\\": 0}}"}'
```

ステップ 10: リソースをクリーンアップする

チュートリアルを完了したら、追加料金が発生しないようにリソースを削除します。 AWS Cloud Map では、リソースを逆の順序でクリーンアップし、最初にサービスインスタンス、次にサービス、最後に名前空間をクリーンアップする必要があります。次の手順では、チュートリアルで使用される AWS Cloud Map リソースをクリーンアップする手順を説明します。

AWS Cloud Map リソースを削除するには

1. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。
2. 名前空間のリストから、cloudmap-tutorial名前空間を選択し、詳細の表示を選択します。
3. 名前空間の詳細ページで、サービスのリストからdata-serviceサービスを選択し、詳細の表示を選択します。
4. サービスインスタンスセクションで、data-instanceインスタンスを選択し、登録解除を選択します。
5. ページの上部にあるパンくずリストを使用して、cloudmap-tutorial.com を選択して名前空間の詳細ページに戻ります。
6. 名前空間の詳細ページで、サービスのリストからデータサービスを選択し、削除を選択します。
7. app-service サービス、 および write-instanceread-instanceサービスインスタンスに対してステップ 3～6 を繰り返します。
8. 左側のナビゲーションで、名前空間を選択します。
9. cloudmap-tutorial 名前空間を選択し、削除を選択します。

次の表に、チュートリアルで使用されている他のリソースを削除するために実行できる手順を示します。

[リソース]	Steps	
DynamoDB テーブル	ステップ 6: (オプション) Amazon DynamoDB デベロッパーガイドの「DynamoDB テーブルを削除してリソースをクリーンアップするDynamoDB」	
Lambda 関数と関連する IAM 実行ロール	AWS Lambda デベロッパーガイドの「クリーンアップ」	

を使用してカスタム属性で AWS Cloud Map サービス検出を使用する方法について説明します。 AWS CLI

このチュートリアルでは、カスタム属性で AWS Cloud Map サービス検出を使用する方法を示します。カスタム属性を使用してリソースを動的に検出 AWS Cloud Map するために を使用するマイクロサービスアプリケーションを作成します。アプリケーションは、DynamoDB テーブルとの間でデータを書き込み、読み取る 2 つの Lambda 関数で構成され、すべてのリソースが登録されています AWS Cloud Map。

チュートリアル AWS マネジメントコンソール のバージョンについては、「」を参照してくださいカスタム属性で AWS Cloud Map サービス検出を使用する方法について説明します。。

前提条件

このチュートリアルを開始する前に、「」の手順を完了してくださいを使用するように をセットアップする AWS Cloud Map。

AWS Cloud Map 名前空間を作成する

名前空間は、アプリケーションのサービスをグループ化するために使用されるコンストラクトです。このステップでは、 AWS Cloud Map API コールを通じてリソースを検出できるようにする名前空間を作成します。

1. 次のコマンドを実行して、HTTP 名前空間を作成します。

```
aws servicediscovery create-http-namespace \
  --name cloudmap-tutorial \
  --creator-request-id cloudmap-tutorial-request
```

コマンドはオペレーション ID を返します。オペレーションのステータスは、次のコマンドで確認できます。

```
aws servicediscovery get-operation \
  --operation-id operation-id
```

2. 名前空間が作成されたら、後続のコマンドで使用する ID を取得できます。

```
aws servicediscovery list-namespaces \
  --query "Namespaces[?Name=='cloudmap-tutorial'].Id" \
  --output text
```

3. 後で使用するために、名前空間 ID を変数に保存します。

```
NAMESPACE_ID=$(aws servicediscovery list-namespaces \
  --query "Namespaces[?Name=='cloudmap-tutorial'].Id" \
  --output text)
```

DynamoDB テーブルを作成する

次に、アプリケーションのデータを保存する DynamoDB テーブルを作成します。

1. 次のコマンドを実行してテーブルを作成します。

```
aws dynamodb create-table \
  --table-name cloudmap \
  --attribute-definitions AttributeName=id,AttributeType=S \
  --key-schema AttributeName=id,KeyType=HASH \
  --billing-mode PAY_PER_REQUEST
```

2. テーブルがアクティブになるまで待ってから次に進みます。

```
aws dynamodb wait table-exists --table-name cloudmap
```

このコマンドは、テーブルが完全に作成され、使用できる状態になるまで待機します。

AWS Cloud Map データサービスを作成し、DynamoDB テーブルを登録する

次に、名前空間にサービスを作成して、データストレージリソースを表します。

1. 次のコマンドを実行して、データストレージリソース AWS Cloud Map のサービスを作成します。

```
aws servicediscovery create-service \
  --name data-service \
  --namespace-id $NAMESPACE_ID \
  --creator-request-id data-service-request
```

2. データサービスのサービス ID を取得します。

```
DATA_SERVICE_ID=$(aws servicediscovery list-services \
  --query "Services[?Name=='data-service'].Id" \
  --output text)
```

3. DynamoDB テーブルをサービスインスタンスとして、テーブル名を指定するカスタム属性に登録します。

```
aws servicediscovery register-instance \
  --service-id $DATA_SERVICE_ID \
  --instance-id data-instance \
  --attributes tablename=cloudmap
```

   カスタム属性`tablename=cloudmap`を使用すると、他の のサービスが DynamoDB テーブル名を動的に検出できます。

Lambda 関数の IAM ロールを作成する

Lambda 関数がリソースへのアクセス AWS に使用する IAM ロールを作成します。

1. 次の JSON を使用して、IAM ロールの信頼ポリシードキュメントを作成します。

   JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. 次のコマンドを実行して、信頼ポリシーを使用して IAM ロールを作成します。

```
aws iam create-role \
  --role-name cloudmap-tutorial-role \
  --assume-role-policy-document file://lambda-trust-policy.json
```

3. 次の JSON を使用して、最小特権のアクセス許可を持つカスタム IAM ポリシーのファイルを作成します。

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Resource": "arn:aws:logs:*:*:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "servicediscovery:DiscoverInstances"
      ],
      "Resource": "*"
    },
```

```
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:PutItem",
        "dynamodb:Scan"
      ],
      "Resource": "arn:aws:dynamodb:*:*:table/cloudmap"
    }
  ]
}
```

4. ポリシーを作成して IAM ロールにアタッチします。

```
aws iam create-policy \
  --policy-name CloudMapTutorialPolicy \
  --policy-document file://cloudmap-policy.json

POLICY_ARN=$(aws iam list-policies \
  --query "Policies[?PolicyName=='CloudMapTutorialPolicy'].Arn" \
  --output text)

aws iam attach-role-policy \
  --role-name cloudmap-tutorial-role \
  --policy-arn $POLICY_ARN

aws iam attach-role-policy \
  --role-name cloudmap-tutorial-role \
  --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole
```

Lambda 関数を作成してデータを書き込む

DynamoDB テーブルにデータを書き込む Lambda 関数を作成するには、次の手順に従います。

1. 書き込み関数の Python ファイルを作成します。

```
cat > writefunction.py << EOF
import json
import boto3
import random

def lambda_handler(event, context):
    try:
```

```
        serviceclient = boto3.client('servicediscovery')

        response = serviceclient.discover_instances(
            NamespaceName='cloudmap-tutorial',
            ServiceName='data-service')

        if not response.get("Instances"):
            return {
                'statusCode': 500,
                'body': json.dumps({"error": "No instances found"})
            }

        tablename = response["Instances"][0]["Attributes"].get("tablename")
        if not tablename:
            return {
                'statusCode': 500,
                'body': json.dumps({"error": "Table name attribute not found"})
            }

        dynamodbclient = boto3.resource('dynamodb')

        table = dynamodbclient.Table(tablename)

        # Validate input
        if not isinstance(event, str):
            return {
                'statusCode': 400,
                'body': json.dumps({"error": "Input must be a string"})
            }

        response = table.put_item(
            Item={ 'id': str(random.randint(1,100)), 'todo': event })

        return {
            'statusCode': 200,
            'body': json.dumps(response)
        }
    except Exception as e:
        return {
            'statusCode': 500,
            'body': json.dumps({"error": str(e)})
        }
EOF
```

この関数は AWS Cloud Map 、 を使用してカスタム属性から DynamoDB テーブル名を検出し、テーブルにデータを書き込みます。

2. Lambda 関数をパッケージ化してデプロイします。

```
zip writefunction.zip writefunction.py

ROLE_ARN=$(aws iam get-role --role-name cloudmap-tutorial-role \
  --query 'Role.Arn' --output text)

aws lambda create-function \
  --function-name writefunction \
  --runtime python3.12 \
  --role $ROLE_ARN \
  --handler writefunction.lambda_handler \
  --zip-file fileb://writefunction.zip \
  --architectures x86_64
```

3. 関数のタイムアウトを更新して、タイムアウトエラーを回避します。

```
aws lambda update-function-configuration \
  --function-name writefunction \
  --timeout 5
```

AWS Cloud Map アプリケーションサービスを作成し、Lambda 書き込み関数を登録する

名前空間にアプリケーション関数を表す別のサービスを作成するには、次の手順に従います。

1. アプリケーション関数のサービスを作成します。

```
aws servicediscovery create-service \
  --name app-service \
  --namespace-id $NAMESPACE_ID \
  --creator-request-id app-service-request
```

2. アプリサービスのサービス ID を取得します。

```
APP_SERVICE_ID=$(aws servicediscovery list-services \
  --query "Services[?Name=='app-service'].Id" \
```

```
  --output text)
```

3. Lambda 書き込み関数をカスタム属性を持つサービスインスタンスとして登録します。

```
aws servicediscovery register-instance \
  --service-id $APP_SERVICE_ID \
  --instance-id write-instance \
  --attributes action=write,functionname=writefunction
```

カスタム属性action=writeと functionname=writefunctionは、クライアントがその目的に基づいてこの関数を検出できるようにします。

Lambda 関数を作成してデータを読み取る

DynamoDB テーブルからデータを読み取る Lambda 関数を作成するには、次の手順に従います。

1. 読み取り関数の Python ファイルを作成します。

```
cat > readfunction.py << EOF
import json
import boto3

def lambda_handler(event, context):
    try:
        serviceclient = boto3.client('servicediscovery')

        response = serviceclient.discover_instances(
            NamespaceName='cloudmap-tutorial',
            ServiceName='data-service')

        if not response.get("Instances"):
            return {
                'statusCode': 500,
                'body': json.dumps({"error": "No instances found"})
            }

        tablename = response["Instances"][0]["Attributes"].get("tablename")
        if not tablename:
            return {
                'statusCode': 500,
                'body': json.dumps({"error": "Table name attribute not found"})
            }
```

```
        dynamodbclient = boto3.resource('dynamodb')

        table = dynamodbclient.Table(tablename)

        # Use pagination for larger tables
        response = table.scan(
            Select='ALL_ATTRIBUTES',
            Limit=50  # Limit results for demonstration purposes
        )

        # For production, you would implement pagination like this:
        # items = []
        # while 'LastEvaluatedKey' in response:
        #     items.extend(response['Items'])
        #     response = table.scan(
        #         Select='ALL_ATTRIBUTES',
        #         ExclusiveStartKey=response['LastEvaluatedKey']
        #     )
        # items.extend(response['Items'])

        return {
            'statusCode': 200,
            'body': json.dumps(response)
        }
    except Exception as e:
        return {
            'statusCode': 500,
            'body': json.dumps({"error": str(e)})
        }
EOF
```

この関数は、 AWS Cloud Map を使用して DynamoDB テーブル名を検出し、テーブルからデータを読み取ります。これには、エラー処理とページ分割コメントが含まれます。

2. Lambda 関数をパッケージ化してデプロイします。

```
zip readfunction.zip readfunction.py

aws lambda create-function \
  --function-name readfunction \
  --runtime python3.12 \
  --role $ROLE_ARN \
```

```
  --handler readfunction.lambda_handler \
  --zip-file fileb://readfunction.zip \
  --architectures x86_64
```

3. 関数のタイムアウトを更新します。

```
aws lambda update-function-configuration \
  --function-name readfunction \
  --timeout 5
```

Lambda 読み取り関数をサービスインスタンスとして登録する

Lambda 読み取り関数をアプリケーションサービス内の別のサービスインスタンスとして登録するには、次の手順に従います。

```
aws servicediscovery register-instance \
  --service-id $APP_SERVICE_ID \
  --instance-id read-instance \
  --attributes action=read,functionname=readfunction
```

カスタム属性`action=read`と `functionname=readfunction`は、クライアントがその目的に基づいてこの関数を検出できるようにします。

クライアントアプリケーションを作成して実行する

書き込み関数を検出して呼び出す AWS Cloud Map ために が使用する Python クライアントアプリケーションを作成するには、次の手順に従います。

1. 書き込みクライアントアプリケーション用の Python ファイルを作成します。

```
cat > writeclient.py << EOF
import boto3
import json

try:
    serviceclient = boto3.client('servicediscovery')

    print("Discovering write function...")
    response = serviceclient.discover_instances(
        NamespaceName='cloudmap-tutorial',
```

```
        ServiceName='app-service',
        QueryParameters={ 'action': 'write' }
    )

    if not response.get("Instances"):
        print("Error: No instances found")
        exit(1)

    functionname = response["Instances"][0]["Attributes"].get("functionname")
    if not functionname:
        print("Error: Function name attribute not found")
        exit(1)

    print(f"Found function: {functionname}")

    lambdaclient = boto3.client('lambda')

    print("Invoking Lambda function...")
    resp = lambdaclient.invoke(
        FunctionName=functionname,
        Payload='"This is a test data"'
    )

    payload = resp["Payload"].read()
    print(f"Response: {payload.decode('utf-8')}")

except Exception as e:
    print(f"Error: {str(e)}")
EOF
```

このクライアントは、 QueryParametersオプションを使用して、 action=write 属性を持つサービスインスタンスを検索します。

2. 読み取りクライアントアプリケーションの Python ファイルを作成します。

```
cat > readclient.py << EOF
import boto3
import json

try:
    serviceclient = boto3.client('servicediscovery')

    print("Discovering read function...")
```

```
    response = serviceclient.discover_instances(
        NamespaceName='cloudmap-tutorial',
        ServiceName='app-service',
        QueryParameters={ 'action': 'read' }
    )

    if not response.get("Instances"):
        print("Error: No instances found")
        exit(1)

    functionname = response["Instances"][0]["Attributes"].get("functionname")
    if not functionname:
        print("Error: Function name attribute not found")
        exit(1)

    print(f"Found function: {functionname}")

    lambdaclient = boto3.client('lambda')

    print("Invoking Lambda function...")
    resp = lambdaclient.invoke(
        FunctionName=functionname,
        InvocationType='RequestResponse'
    )

    payload = resp["Payload"].read()
    print(f"Response: {payload.decode('utf-8')}")

except Exception as e:
    print(f"Error: {str(e)}")
EOF
```

3. 書き込みクライアントを実行して、DynamoDB テーブルにデータを追加します。

```
python3 writeclient.py
```

出力には、HTTP ステータスコード 200 で成功したレスポンスが表示されます。

4. 読み取りクライアントを実行して、DynamoDB テーブルからデータを取得します。

```
python3 readclient.py
```

出力には、ランダムに生成された ID や「これはテストデータです」値など、テーブルに書き込まれたデータが表示されます。

リソースをクリーンアップする

チュートリアルが終了したら、追加料金が発生しないようにリソースをクリーンアップします。

1. まず、次のコマンドを実行してサービスインスタンスの登録を解除します。

```
aws servicediscovery deregister-instance \
  --service-id $APP_SERVICE_ID \
  --instance-id read-instance

aws servicediscovery deregister-instance \
  --service-id $APP_SERVICE_ID \
  --instance-id write-instance

aws servicediscovery deregister-instance \
  --service-id $DATA_SERVICE_ID \
  --instance-id data-instance
```

2. 次のコマンドを実行して、サービスを削除します。

```
aws servicediscovery delete-service \
  --id $APP_SERVICE_ID

aws servicediscovery delete-service \
  --id $DATA_SERVICE_ID
```

3. 次のコマンドを実行して、名前空間を削除します。

```
aws servicediscovery delete-namespace \
  --id $NAMESPACE_ID
```

4. 次のコマンドを実行して、Lambda 関数を削除します。

```
aws lambda delete-function --function-name writefunction
aws lambda delete-function --function-name readfunction
```

5. 次のコマンドを実行して、IAM ロールとポリシーを削除します。

```
aws iam detach-role-policy \
  --role-name cloudmap-tutorial-role \
  --policy-arn $POLICY_ARN

aws iam detach-role-policy \
  --role-name cloudmap-tutorial-role \
  --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole

aws iam delete-policy \
  --policy-arn $POLICY_ARN

aws iam delete-role --role-name cloudmap-tutorial-role
```

6. 次のコマンドを実行して、DynamoDB テーブルを削除します。

```
aws dynamodb delete-table --table-name cloudmap
```

7. 次のコマンドを実行して、一時ファイルをクリーンアップします。

```
rm -f lambda-trust-policy.json cloudmap-policy.json writefunction.py
 readfunction.py writefunction.zip readfunction.zip writeclient.py readclient.py
```

AWS Cloud Map 名前空間

名前空間は、アプリケーションのサービスを共通の名前と検出可能性レベルでグループ化するために使用される の論理エンティティ AWS Cloud Map です。名前空間を作成するときは、以下を指定します。

- アプリケーションがインスタンスを検出するために使用する名前。
- に登録したサービスインスタンスを検出 AWS Cloud Map できる方法。リソースをインターネット経由でパブリックに検出する必要があるか、特定の仮想プライベートクラウド (VPC) でプライベートに検出する必要があるか、または API コールのみで検出する必要があるかを決定できます。

名前空間に関する一般的な概念を次に示します。

- 名前空間は、作成された AWS リージョン に固有です。を複数のリージョン AWS Cloud Map で使用するには、リージョンごとに名前空間を作成する必要があります。
- VPC 内の DNS クエリによるインスタンス検出を許可する名前空間を作成すると、 AWS Cloud Map は自動的にプライベート Route 53 ホストゾーンを作成します。このホストゾーンは、複数の VPCsに関連付けることができます。詳細については、Amazon Route 53 API リファレンスの「AssociateVPCWithHostedZone」を参照してください。

トピック

- アプリケーションサービスをグループ化する AWS Cloud Map 名前空間の作成
- AWS Cloud Map 名前空間の一覧表示
- AWS Cloud Map 名前空間の削除
- 共有 AWS Cloud Map 名前空間

アプリケーションサービスをグループ化する AWS Cloud Map 名前空間の作成

名前空間を作成して、API コールまたは DNS クエリを介してアプリケーションリソースを検出できるわかりやすい名前でアプリケーションのサービスをグループ化できます。

インスタンス検出オプション

次の表は、 のさまざまなインスタンス検出オプション AWS Cloud Map と、アプリケーションのサービスと設定に応じて作成できる対応する名前空間タイプをまとめたものです。

名前空間タイプ	インスタンス検出方法	仕組み	追加情報
HTTP	API コール	アプリケーションのリソースは、 `DiscoverInstances` API のみを呼び出すことで他のリソースを検出できます。	• DiscoverInstances • CreateHttpNamespace
プライベート DNS	VPC 内の API コールと DNS クエリ	プライベート DNS 名前空間を作成すると、 は対応する Amazon Route 53 プライベートホストゾーン AWS Cloud Map を作成します。アプリケーションのリソースは、 `DiscoverInstances` API を呼び出し、 AWS Cloud Map が自動的に作成するプライベート Route 53 ホストゾーンのネームサーバーをクエリすることで、他のリソースを検出できます。	• DiscoverInstances • CreatePrivateDnsNamespace

名前空間タイプ	インスタンス検出方法	仕組み	追加情報
		によって作成されたホストゾーン AWS Cloud Map の名前は名前空間と同じで、*service-name* .*namespace-name* 形式の名前を持つ DNS レコードが含まれています。 **Note** Route 53 リゾルバーは、プライベートホストゾーンのレコードを使用して、VPC で発生した DNS クエリを解決します。DNS クエリのドメイン名に一致するレコードが、プライベートホストゾーンに含まれていない場合、Route 53 は、NXDOMAIN (存在しないドメイン) でク	

名前空間タイプ	インスタンス検出方法	仕組み	追加情報
		エリに応答します。	

名前空間タイプ	インスタンス検出方法	仕組み	追加情報
[パブリック DNS]	API コールとパブリック DNS クエリ	パブリック DNS 名前空間を作成すると、は対応する Amazon Route 53 パブリックホストゾーン AWS Cloud Map を作成します。アプリケーションのリソースは、`DiscoverInstances` API を呼び出し、AWS Cloud Map が自動的に作成するパブリック Route 53 ホストゾーンのネームサーバーをクエリすることで、他のリソースを検出できます。 パブリックホストゾーンの名前は名前空間と同じで、*service-name*.*namespace-name* 形式の名前を持つ DNS レコードが含まれています。 **Note** この場合の名前空間名は、登録したドメイン名である	• DiscoverInstances • CreatePublicDnsNamespace

名前空間タイプ	インスタンス検出方法	仕組み	追加情報
		必要があります。	

手順

以下の手順に従って、 AWS CLI、 AWS マネジメントコンソール、または SDK for Python を使用して名前空間を作成できます。

AWS マネジメントコンソール

1. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。
2. [名前空間の作成] を選択します。
3. 名前空間名には、インスタンスの検出に使用される名前を入力します。

   > **Note**
   >
   > - パブリック DNS クエリ用に設定された名前空間は、最上位ドメインで終わる必要があります。例えば、`.com`。
   > - まず名前を Punycode に変換する場合は、国際化ドメイン名 (IDN) を指定します。オンラインコンバーターについては、インターネットで「punycode コンバーター」を検索してください。
   >
   >   プログラムで名前空間を作成する際、国際化ドメイン名 (IDN) を Punycode に変換することもできます。たとえば、Java を使用する場合は、java.net.IDN ライブラリの `toASCII` メソッドを使って Unicode 値を Punycode に変換できます。

4. (オプション) 名前空間の説明には、名前空間ページと名前空間情報の下に表示される名前空間に関する情報を入力します。この情報を使用して、名前空間を簡単に識別できます。
5. インスタンス検出では、API コール、VPCs 内の API コールと DNS クエリ、API コールとパブリック DNS クエリのいずれかを選択して、それぞれ HTTP、プライベート DNS、またはパブリック DNS 名前空間を作成できます。詳細については、「インスタンス検出オプション」を参照してください。

選択内容に基づいて、以下の手順を実行します。

- VPCs で API コールと DNS クエリを選択した場合、VPC の場合は、名前空間を関連付ける Virtual Private Cloud (VPC) を選択します。
- VPCs で API コールと DNS クエリを選択した場合、または API コールとパブリック DNS クエリを選択した場合は、TTL に数値を秒単位で指定します。有効期限 (TTL) 値は、DNS リゾルバーが名前空間で作成された Route 53 ホストゾーンの認証開始 (SOA) DNS レコードの情報をキャッシュする期間を決定します。TTL の詳細については、「Amazon Route 53 デベロッパーガイド」の「TTL (秒)」を参照してください。

6. (オプション) タグ でタグを追加 を選択し、名前空間にタグを付けるキーと値を指定します。1 つ以上のタグを指定して、 名前空間に追加することができます。タグを使用すると、AWS リソースをより簡単に管理できるようにリソースを分類できます。詳細については、「AWS Cloud Map リソースのタグ付け」を参照してください。
7. [名前空間の作成] を選択します。ListOperations を使用してオペレーションのステータスを表示できます。詳細については、 AWS Cloud Map API リファレンスのListOperations」を参照してください。

AWS CLI

- 希望するインスタンスディスカバリタイプのコマンドで名前空間を作成します (*red* の値は独自の値に置換):
  - `create-http-namespace` を使用して HTTP 名前空間を作成します。HTTP 名前空間を使用して登録したサービスインスタンスは、`DiscoverInstances` リクエストを使用して検出できますが、DNS を使用して検出することはできません。

    ```
    aws servicediscovery create-http-namespace --name name-of-namespace
    ```

  - DNS に基づいてプライベート名前空間を作成します。これは、`create-private-dns-namespace`を使用して、指定した Amazon VPC 内でのみ表示されます。 リクエストまたは DNS`DiscoverInstances` を使用して、プライベート DNS 名前空間に登録されたインスタンスを検出できます。

    ```
    aws servicediscovery create-private-dns-namespace --name name-of-namespace --
    vpc vpc-xxxxxxxxx
    ```

- create-public-dns-namespace を使用して DNS に基づいてパブリック名前空間を作成します。これは、インターネットで表示されます。DiscoverInstances リクエストまたは DNS を使用して、プライベート DNS 名前空間で登録したインスタンスを検出できます。

```
aws servicediscovery create-public-dns-namespace --name name-of-namespace
```

AWS SDK for Python (Boto3)

1. まだBoto3がインストールしていない場合は、[こちら]のインストール、設定、使用に関する説明をBoto3参照してください。
2. Boto3をインポートしてサービスとしてservicediscoveryを使用してください。

```
import boto3
client = boto3.client('servicediscovery')
```

3. 希望するインスタンスディスカバリタイプのコマンドで名前空間を作成します (*red* の値は独自の値に置換):

- create_http_namespace( ) を使用して HTTP 名前空間を作成します。HTTP 名前空間を使用して登録したサービスインスタンスは、discover_instances( ) を使用して検出できますが、DNS を使用して検出することはできません。

```
response = client.create_http_namespace(
    Name='name-of-namespace',
)
# If you want to see the response
print(response)
```

- DNS に基づいてプライベート名前空間を作成します。これは、create_private_dns_namespace( )を使用して、指定した Amazon VPC 内でのみ表示されます。discover_instances( )または DNS を使用して、プライベート DNS 名前空間に登録されたインスタンスを検出できます。

```
response = client.create_private_dns_namespace(
    Name='name-of-namespace',
    Vpc='vpc-1c56417b',
)
# If you want to see the response
```

```
print(response)
```

- `create_public_dns_namespace()` を使用して DNS に基づいてパブリック名前空間を作成します。これは、インターネットで表示されます。`discover_instances()` リクエストまたは DNS を使用して、プライベート DNS 名前空間で登録したインスタンスを検出できます。

```
response = client.create_public_dns_namespace(
    Name='name-of-namespace',
)
# If you want to see the response
print(response)
```

- レスポンスオブジェクトの例

```
{
    'OperationId': 'gv4g5meo7ndmeh4fqskygvk23d2fijwa-k9302yzd',
    'ResponseMetadata': {
        '...': '...',
    },
}
```

次の手順

名前空間を作成したら、名前空間にサービスを作成して、アプリケーション内の特定の目的にまとめて対応するアプリケーションリソースをグループ化できます。サービスは、アプリケーションリソースをインスタンスとして登録するためのテンプレートとして機能します。 AWS Cloud Map サービスの作成の詳細については、「」を参照してくださいアプリケーションコンポーネントの AWS Cloud Map サービスの作成。

AWS Cloud Map 名前空間の一覧表示

名前空間を作成したら、以下の手順に従って作成した名前空間のリストを表示できます。

AWS マネジメントコンソール

1. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。

2. ナビゲーションペインで、名前空間を選択して名前空間のリストを表示します。名前空間は、名前、説明、インスタンス検出モード、所有者、または名前空間 ID で並べ替えることができます。検索フィールドに名前空間名または ID を入力して、特定の名前空間を見つけて表示することもできます。

AWS CLI

- list-namespaces コマンドで名前空間を一覧表示します。

```
aws servicediscovery list-namespaces
```

AWS SDK for Python (Boto3)

1. まだBoto3がインストールしていない場合は、[こちら]のインストール、設定、使用に関する説明をBoto3参照してください。
2. Boto3をインポートしてサービスとしてservicediscoveryを使用してください。

```
import boto3
client = boto3.client('servicediscovery')
```

3. list_namespaces() で名前空間を一覧表示します。

```
response = client.list_namespaces()
# If you want to see the response
print(response)
```

レスポンスオブジェクトの例

```
{
    'Namespaces': [
        {
            'Arn': 'arn:aws::servicediscovery:us-west-2:123456789012:namespace/
ns-xxxxxxxxxxxxxx',
            'CreateDate': 1585354387.357,
            'Id': 'ns-xxxxxxxxxxxxxx',
            'Name': 'myFirstNamespace',
            'Properties': {
                'DnsProperties': {
                    'HostedZoneId': 'Z06752353VBUDTC32S84S',
```

```
                },
                'HttpProperties': {
                    'HttpName': 'myFirstNamespace',
                },
            },
            'Type': 'DNS_PRIVATE',
        },
        {
            'Arn': 'arn:aws::servicediscovery:us-west-2:123456789012:namespace/
ns-xxxxxxxxxxxxxx',
            'CreateDate': 1586468974.698,
            'Description': 'My second namespace',
            'Id': 'ns-xxxxxxxxxxxxxx',
            'Name': 'mySecondNamespace.com',
            'Properties': {
                'DnsProperties': {
                },
                'HttpProperties': {
                    'HttpName': 'mySecondNamespace.com',
                },
            },
            'Type': 'HTTP',
        },
        {
            'Arn': 'arn:aws::servicediscovery:us-west-2:123456789012:namespace/
ns-xxxxxxxxxxxxxx',
            'CreateDate': 1587055896.798,
            'Id': 'ns-xxxxxxxxxxxxxx',
            'Name': 'myThirdNamespace.com',
            'Properties': {
                'DnsProperties': {
                    'HostedZoneId': 'Z09983722P0QME1B3KC8I',
                },
                'HttpProperties': {
                    'HttpName': 'myThirdNamespace.com',
                },
            },
            'Type': 'DNS_PRIVATE',
        },
    ],
    'ResponseMetadata': {
        '...': '...',
    },
```

```
}
```

AWS Cloud Map 名前空間の削除

名前空間の使用が完了したら、削除できます。削除した名前空間は、サービスインスタンスの登録または検出に使用できなくなります。

> **Note**
>
> DNS 名前空間を削除すると、 は名前空間の作成時に作成された対応する Amazon Route 53 ホストゾーン AWS Cloud Map を削除します。

名前空間を削除する前に、すべてのサービスインスタンスの登録を解除してから、名前空間で作成されたすべてのサービスを削除する必要があります。詳細については、「AWS Cloud Map サービスインスタンスの登録解除」および「AWS Cloud Map サービスの削除」を参照してください。

名前空間で作成されたインスタンスの登録を解除し、サービスを削除したら、以下の手順に従って名前空間を削除します。

AWS マネジメントコンソール

1. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。
2. ナビゲーションペインで [名前空間] を選択します。
3. 削除する名前空間を選択し、削除を選択します。
4. 削除を再度選択して、サービスを削除することを確認します。

AWS CLI

- 名前空間を `delete-namespace` コマンドで削除します (*`red`* の値は独自の値で置き換え)。名前空間に 1 つ以上のサービスが含まれている場合、リクエストは失敗します。

```
aws servicediscovery delete-namespace --id ns-xxxxxxxxxxx
```

AWS SDK for Python (Boto3)

1. まだBoto3がインストールしていない場合は、[こちら]のインストール、設定、使用に関する説明をBoto3参照してください。
2. Boto3をインポートしてサービスとしてservicediscoveryを使用してください。

```
import boto3
client = boto3.client('servicediscovery')
```

3. 名前空間を delete_namespace()で削除します (*red* の値は独自の値で置き換え)。名前空間に 1 つ以上のサービスが含まれている場合、リクエストは失敗します。

```
response = client.delete_namespace(
    Id='ns-xxxxxxxxxxx',
)
# If you want to see the response
print(response)
```

レスポンスオブジェクトの例

```
{
    'OperationId': 'gv4g5meo7ndmeh4fqskygvk23d2fijwa-k98y6drk',
    'ResponseMetadata': {
        '...': '...',
    },
}
```

共有 AWS Cloud Map 名前空間

AWS Cloud Map を使用すると、名前空間の所有者は、名前空間を の他の AWS アカウント または組織内で共有 AWS Organizations して、クロスアカウントサービス検出とサービスレジストリを簡素化できます。これにより、 AWS 組織内の他の AWS アカウント またはチームが管理する名前空間を簡単に使用できます。

AWS Cloud Map は AWS Resource Access Manager (AWS RAM) と統合してリソース共有を有効にします。 AWS RAM は、一部の AWS Cloud Map リソースを他の AWS アカウント または と共有できるようにするサービスです AWS Organizations。では AWS RAM、リソース共有を作成して、所有しているリソースを共有します。リソース共有は、共有するリソースと、それらを共有するコンシューマーを指定します。コンシューマーには以下が含まれます。

- の組織 AWS アカウント 内で固有 AWS Organizations
- の組織内の組織単位 AWS Organizations
- の組織全体 AWS Organizations

詳細については AWS RAM、AWS RAM 「 ユーザーガイド」を参照してください。

このトピックでは、所有しているリソースの共有方法と、共有されているリソースの使用方法を説明します。

内容

- 名前空間の共有に関する考慮事項
- AWS Cloud Map 名前空間の共有
- AWS Cloud Map 名前空間の共有を停止する
- 共有 AWS Cloud Map 名前空間の識別
- 名前空間を共有するアクセス許可の付与
- 共有名前空間の責任とアクセス許可
- 請求と使用量測定
- クォータ

名前空間の共有に関する考慮事項

- 名前空間を共有するには、 でその名前空間を所有している必要があります AWS アカウント。つまり、自分のアカウントにそのリソースが割り当てられているか、プロビジョニングされている必要があります。共有されている名前空間を共有することはできません。
- 名前空間を組織または の組織単位と共有するには AWS Organizations、 との共有を有効にする必要があります AWS Organizations。詳細については、「AWS RAM ユーザーガイド」の「Enable Sharing with AWS Organizations」を参照してください。
- 共有プライベート DNS 名前空間で DNS クエリを使用するサービス検出の場合、名前空間所有者は、名前空間に関連付けられたプライベートホストゾーンの ID とコンシューマーの VPC `create-vpc-association-authorization`を使用して を呼び出す必要があります。

```
aws route53 create-vpc-association-authorization --hosted-zone-id Z1234567890ABC --
vpc VPCRegion=us-east-1,VPCId=vpc-12345678
```

名前空間コンシューマーは、プライベートホストゾーンの ID を使用して `associate-vpc-with-hosted-zone` を呼び出す必要があります。

```
aws route53 associate-vpc-with-hosted-zone --hosted-zone-id Z1234567890ABC --vpc
 VPCRegion=us-east-1,VPCId=vpc-12345678
```

詳細については、「Amazon Route 53 デベロッパーガイド」の「異なる で作成した Amazon VPC とプライベートホストゾーンの関連付け AWS アカウント」を参照してください。

- 共有 DNS 名前空間に関連付けられたサービスのup-to-dateネットワークロケーションを検出した後、異なる VPCs にある場合、サービスと通信するように VPC 間接続を設定する必要がある場合があります。これは、VPC ピアリング接続を使用して実現できます。詳細については、「Amazon Virtual Private Cloud ピアリングガイド」の「VPC ピアリング接続の作成または削除」を参照してください。
- を使用して`ListOperations`、他のアカウントによって実行される共有名前空間に対するオペレーションを一覧表示することはできません。
- タグ付けは、共有名前空間ではサポートされていません。

AWS Cloud Map 名前空間の共有

所有している AWS Cloud Map 名前空間を他の AWS アカウント (コンシューマー) と共有する場合、これらのアカウントを有効にして、一時的な認証情報を必要とせずに、名前空間内のサービスのup-to-dateネットワークロケーションを検出できます。

名前空間を共有するには、リソース共有に追加する必要があります。リソース共有とは、 AWS アカウント間で自身のリソースを共有するための AWS RAM リソースです。リソース共有では、共有対象のリソースと、共有先のコンシューマーを指定します。名前空間を新しいリソース共有に追加するには、まず AWS RAM コンソールを使用してリソース共有を作成する必要があります。

の組織に属 AWS Organizations していて、組織内での共有が有効になっている場合、組織内のコンシューマーには共有名前空間へのアクセスが自動的に付与されます。それ以外の場合、コンシューマーはリソース共有への参加の招待を受け取り、招待を承諾すると共有名前空間へのアクセスが許可されます。

AWS RAM コンソールまたは を使用して、所有している名前空間を共有できます AWS CLI。

AWS RAM console

AWS RAM コンソールを使用して所有している名前空間を共有するには

「AWS RAM ユーザーガイド」の「Creating a resource share in AWS RAM」を参照してください。

AWS CLI

を使用して所有している名前空間を共有するには AWS CLI

AWS RAM create-resource-share コマンドを使用します。

AWS Cloud Map 名前空間の共有を停止する

名前空間が共有されなくなった場合、名前空間とそれに関連付けられたサービスおよびインスタンスにコンシューマーがアクセスできなくなります AWS アカウント。これには、コンシューマーが名前空間にアクセスしたときに名前空間で作成されたリソースが含まれます。

所有している名前空間の共有を停止するには、リソース共有から削除する必要があります。これを行うには、 AWS RAM コンソールまたは を使用します AWS CLI。

AWS RAM console

AWS RAM コンソールを使用して所有している名前空間の共有を停止するには

「AWS RAM ユーザーガイド」の「リソース共有の更新」を参照してください。

AWS CLI

を使用して所有している名前空間の共有を停止するには AWS CLI

disassociate-resource-share コマンドを使用します。

共有 AWS Cloud Map 名前空間の識別

所有者とコンシューマーは、 AWS Cloud Map コンソールと を使用して共有名前空間を識別できます AWS CLI。名前空間所有者は、 `ResourceOwner`プロパティを使用して識別できます。サービス AWS アカウント を作成するか、共有名前空間にインスタンスを登録する は、`CreatedByAccount`プロパティを使用して識別できます。

AWS Cloud Map console

AWS Cloud Map コンソールを使用して共有名前空間を識別するには

1. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。
2. 名前空間ページのリソース所有者の下に、名前空間を所有 AWS アカウント する の ID が表示されます。
3. 識別する名前空間のドメイン名を選択します。
4. 名前空間: ***namespace-name*** ページの「名前空間情報」セクションの「リソース所有者」で、名前空間を所有 AWS アカウント する の ID を確認できます。

AWS CLI

を使用して共有名前空間を識別するには AWS CLI、list-namespaces コマンドを使用します。コマンドは、所有している名前空間と共有されている名前空間を返します。ResourceOwner フィールドには、名前空間所有者の AWS アカウント ID が表示されます。

次のlist-namespaces呼び出しはアカウント によって行われます111122223333。

```
aws servicediscovery list-namespaces
```

出力:

```
{
    "Namespaces": [
        {
            "Arn": "arn:aws:servicediscovery:us-west-2:111122223333:namespace/ns-
abcdef01234567890",
            "CreateDate": 1585354387.357,
            "Id": "ns-abcdef01234567890",
            "Name": "local",
            "Properties": {
                "DnsProperties": {
                    "HostedZoneId": "Z06752353VBUDTC32S84S"
                },
                "HttpProperties": {
                    "HttpName": "local"
                 }
            },
```

```
            "Type": "DNS_PRIVATE",
            "ServiceCount": 2,
           "ResourceOwner": "111122223333"
        },
        {
            "Arn": "arn:aws:servicediscovery:us-west-2:444455556666:namespace/
ns-021345abcdef6789",
            "CreateDate": 1586468974.698,
            "Description": "Shared second namespace",
            "Id": "ns-021345abcdef6789",
            "Name": "My-second-namespace",
            "Properties": {
                "DnsProperties": {},
                "HttpProperties": {
                    "HttpName": "Shared-second-namespace"
                }
            },
            "Type": "HTTP",
            "ServiceCount": 0,
            "ResourceOwner": "444455556666"
        }
    ]
}
```

このシナリオでは、名前空間ns-abcdef01234567890は によって作成および所有111122223333され、名前空間ns-021345abcdef6789は によって作成および所有されます444455556666。名前空間ns-021345abcdef6789は、アカウント 111122223333によってアカウント と共有されます444455556666。

名前空間を共有するアクセス許可の付与

IAM プリンシパルが名前空間を共有するには、最小限のアクセス許可のセットが必要です。AWSCloudMapFullAccess および AWSResourceAccessManagerFullAccess管理ポリシーを使用して、IAM プリンシパルが共有名前空間を共有および使用するために必要なアクセス許可を持っていることを確認することをお勧めします。

カスタム IAM ポリシーを使用する場合は、名前空間の共有に servicediscovery:PutResourcePolicy、servicediscovery:GetResourcePolicy、および servicediscovery:DeleteResourcePolicyアクションが必要です。これらはアクセス許可のみの IAM アクションです。IAM プリンシパルにこれらのアクセス許可が付与されていない場合、を使用して名前空間を共有しようとするとエラーが発生します AWS RAM。

が IAM AWS RAM を使用する方法の詳細については、AWS RAM 「 ユーザーガイド」の「 が IAM AWS RAM を使用する方法」を参照してください。

共有名前空間の責任とアクセス許可

名前空間の所有者とコンシューマーは、共有名前空間に対して異なるアクションを実行できます。

所有者のアクセス許可

名前空間所有者は、共有名前空間で次のアクションを実行できます。

- コンシューマーアカウントによって作成されたサービスや、これらのサービスに登録されたインスタンスなど、名前空間に関連付けられたサービスにアクセスします。
- コンシューマーアカウントによって作成されたサービスやこれらのサービスに登録されたインスタンスへのアクセスなど、 名前空間へのアクセスを取り消します。
- コンシューマーまたは名前空間所有者が共有名前空間で作成したサービスでインスタンスを登録および登録解除するアクセス許可を他のアカウントに設定します。
- コンシューマーアカウントによって作成されたサービスや登録されたインスタンスなど、サービスを削除し、インスタンスの登録を解除します。
- 共有名前空間を更新または削除します。

コンシューマーのアクセス許可

名前空間コンシューマーは、共有名前空間で次のアクションを実行できます。

- 名前空間でサービスを作成および削除します。
- 名前空間で作成されたサービスでインスタンスを登録および登録解除します。
- 名前空間で作成されたサービスに登録されているインスタンスを検出します。

コンシューマーは、共有名前空間を更新または削除することはできません。共有名前空間へのアクセスが失われると、コンシューマーアカウントは名前空間で作成したサービスにもアクセスできなくなります。

請求と使用量測定

所有者は、共有名前空間に登録したインスタンスと、これらのインスタンスの登録時に作成された Route 53 ヘルスチェックに対して課金されます。コンシューマーは、名前空間に登録した

インスタンスと、これらのインスタンスの登録時に作成された Route 53 ヘルスチェックに対して課金されます。共有名前空間が DNS 名前空間の場合、名前空間の所有者は、名前空間にサービスが作成されたときに作成された Route 53 DNS レコードに対して課金されます。所有者は、DiscoverInstancesおよび DiscoverInstancesRevision呼び出しに対して課金されます。コンシューマーは、 DiscoverInstancesおよび DiscoverInstancesRevision呼び出しに対して課金されます。

クォータ

共有名前空間は、リージョンクォータあたりの名前空間所有者の名前空間にのみカウントされます。共有名前空間にコンシューマーによって登録されたインスタンスは、名前空間クォータごとに所有者のインスタンスにカウントされます。コンシューマーが共有名前空間にサービスを作成する場合、サービスに登録されたインスタンスは、サービスクォータあたりのコンシューマーのインスタンスにカウントされます。所有者が共有名前空間にサービスを作成する場合、サービスに登録されたインスタンスは、サービスクォータあたりの所有者のインスタンスにカウントされます。

AWS Cloud Map サービス

AWS Cloud Map サービスは、サービスインスタンスを登録するためのテンプレートであり、該当する場合、サービス名と DNS 設定で構成されます。ヘルスチェックを設定して、サービス内のインスタンスのヘルスステータスを判断し、異常なリソースを除外することもできます。サービスはアプリケーションのコンポーネントを表すことができます。たとえば、アプリケーションでの支払いを処理するリソース用のサービスを作成し、ユーザーを管理するリソース用のサービスを作成できます。

サービスを使用すると、リソースへの接続に使用できる 1 つ以上のエンドポイントを取得して、アプリケーションのリソースを見つけることができます。リソースの場所は、名前空間の設定方法に応じて、DNS クエリまたは AWS Cloud Map `DiscoverInstances` API アクションを使用して行われます。 AWS Cloud Map コンソールを使用して、サービスレベルでインスタンス検出の範囲を絞り込むことができます。

`UpdateServiceAttributes` API を使用して、サービスレベルでカスタムメタデータを属性として指定することもできます。インスタンス間で属性が重複しないようにサービス属性を設定し、インスタンス属性を変更することなくこれらの属性を変更できます。サービスレベルで属性として指定できる情報には、以下が含まれますが、これらに限定されません。

- プログレッシブデプロイ中にトラフィックをシフトするためのエンドポイントの重み。
- API タイムアウトや推奨される再試行ポリシーなどのサービス設定。

詳細については、 AWS Cloud Map API リファレンスの「UpdateServiceAttributes」を参照してください。

以下のトピックでは、 サービスのヘルスチェックと DNS 設定について説明し、サービスの作成、一覧表示、更新、削除の手順を示します。

トピック

- AWS Cloud Map サービスヘルスチェック設定
- AWS Cloud Map サービス DNS 設定
- アプリケーションコンポーネントの AWS Cloud Map サービスの作成
- AWS Cloud Map サービスの更新
- 名前空間での AWS Cloud Map サービスの一覧表示
- AWS Cloud Map サービスの削除

AWS Cloud Map サービスヘルスチェック設定

ヘルスチェックは、サービスインスタンスが正常かどうかを判断するのに役立ちます。サービスの作成中にヘルスチェックを設定しない場合、トラフィックはインスタンスのヘルスステータスに関係なくサービスインスタンスにルーティングされます。ヘルスチェックを設定すると、 はデフォルトで正常なリソース AWS Cloud Map を返します。`DiscoverInstances` API の `HealthStatus`パラメータを使用して、ヘルスステータスでリソースをフィルタリングし、異常なリソースのリストを取得できます。`GetInstancesHealthStatus` API を使用して、特定のサービスインスタンスのヘルスステータスを取得することもできます。

AWS Cloud Map サービスを作成するときに、Route 53 ヘルスチェックまたはカスタムのサードパーティーヘルスチェックを設定できます。

Route 53 ヘルスチェック

Amazon Route 53 ヘルスチェックの設定を指定すると、 はインスタンスを登録するたびに Route 53 ヘルスチェック AWS Cloud Map を作成し、インスタンスの登録を解除するとヘルスチェックを削除します。

パブリック DNS 名前空間の場合、 は、インスタンスの登録時に が AWS Cloud Map 作成する Route 53 レコードにヘルスチェックを AWS Cloud Map 関連付けます。サービスの DNS 設定で `A` と の両方の`AAAA`レコードタイプを指定すると、 は IPv4 アドレスを使用してリソースのヘルスをチェックするヘルスチェック AWS Cloud Map を作成します。IPv4 アドレスで指定されたエンドポイントが異常である場合、Route 53 は レコード`A`と `AAAA`レコードの両方を異常と見なします。サービスの DNS 設定で`CNAME`レコードタイプを指定した場合、Route 53 ヘルスチェックを設定することはできません。

API コールを使用してインスタンスを検出する名前空間の場合、 AWS Cloud Map は Route 53 ヘルスチェックを作成します。ただし、 がヘルスチェックを関連付け AWS Cloud Map る DNS レコードはありません。ヘルスチェックが正常かどうかを判断するには、Route 53 コンソールまたは Amazon CloudWatch を使用してモニタリングを設定することができます。Route 53 コンソールの使用方法の詳細については、Amazon Route 53 デベロッパーガイドの「ヘルスチェックが失敗した場合に通知を取得する」を参照してください。CloudWatch の使用方法の詳細については、Amazon CloudWatch API リファレンスの 「PutMetricAlarm」を参照してください。

> **Note**
>
> - プライベート DNS 名前空間で作成されたサービスの Amazon Route 53 ヘルスチェックを設定することはできません。
> - 各ヘルスチェックの Route 53 ヘルスチェッカーは、30 秒ごとにヘルスチェックリクエストをエンドポイント AWS リージョン に送信します。平均して、エンドポイントは約 2 秒ごとにヘルスチェックリクエストを受け取ることになります。ただし、ヘルスチェッカーは互いに調整しません。そのため、1 秒以内に複数のリクエストを受け取った後で数秒間、ヘルスチェックを受け取らないという状況が発生する場合があります。ヘルスチェックリージョンのリストについては、「リージョン」を参照してください。

Route 53 ヘルスチェックの料金については、「Route 53 の料金」を参照してください。

カスタムヘルスチェック

インスタンスの登録時にカスタムヘルスチェックを使用する AWS Cloud Map ように を設定する場合は、サードパーティーのヘルスチェッカーを使用してリソースのヘルスを評価する必要があります。カスタムヘルスチェックは、以下の状況で役立ちます。

- インターネット経由でリソースにアクセスできないため、Route 53 ヘルスチェックを使用することができません。例えば、Amazon VPC にあるインスタンスがあるとします。このインスタンスにはカスタムヘルスチェックを使用できます。ただし、ヘルスチェックが機能するには、ヘルスチェッカーもインスタンスと同じ VPC にある必要があります。
- リソースの場所に関係なく、サードパーティーのヘルスチェッカーを使用します。

カスタムヘルスチェックを使用する場合、 AWS Cloud Map は特定のリソースの状態を直接チェックしません。代わりに、サードパーティーのヘルスチェッカーはリソースの状態をチェックし、アプリケーションにステータスを返します。その後、アプリケーションはこのステータスをリレーする`UpdateInstanceCustomHealthStatus`リクエストを送信する必要があります AWS Cloud Map。リレーされた初期ステータスが であり`UNHEALTHY`、30 秒`UpdateInstanceCustomHealthStatus`以内に のステータスをリレーする別の がない場合`HEALTHY`、リソースは異常であると見なされます。 はそのリソースへのトラフィックのルーティング AWS Cloud Map を停止します。

AWS Cloud Map サービス DNS 設定

DNS クエリによるインスタンス検出をサポートする名前空間にサービスを作成すると、 は Route 53 DNS レコード AWS Cloud Map を作成します。が AWS Cloud Map 作成するすべての Route 53 DNS レコードに適用される Route 53 ルーティングポリシーと DNS レコードタイプを指定する必要があります。

ルーティングポリシー

ルーティングポリシーは、サービスインスタンスの検出に使用される DNS クエリに Route 53 がどのように応答するかを決定します。サポートされているルーティングポリシーとその関連 AWS Cloud Map は次のとおりです。

加重ルーティング

Route 53 は、同じサービスを使用して登録したインスタンスの中からランダムに選択された AWS Cloud Map 1 つの AWS Cloud Map サービスインスタンスから該当する値を返します。レコードの加重はすべて同じであるため、トラフィック量を増減してインスタンスにルーティングすることはできません。

例えば、サービスに A レコード 1 つとヘルスチェックが含まれており、そのサービスを使用して、10 のインスタンスを登録するとします。Route 53 は、DNS クエリに対して、正常なインスタンスの中からランダムに選択した 1 つのインスタンスの IP アドレスを返します。正常なインスタンスがない場合、Route 53 は、すべてのインスタンスが正常であるかのように DNS クエリに応答します。

サービスでヘルスチェックを定義していない場合、Route 53 はすべてのインスタンスが正常であると仮定し、ランダムに選択した 1 つのインスタンスの該当する値を返します。

詳細については、Amazon Route 53 デベロッパーガイドの加重ルーティングを参照してください。

複数値回答ルーティング

サービスでヘルスチェックを定義し、ヘルスチェックが正常であれば、Route 53 は最大 8 のインスタンスについて該当する値を返します。

例えば、サービスに 1 つのA レコードとヘルスチェックの設定が含まれているとします。このサービスを使用して、10 個のインスタンスを登録します。Route 53 は、DNS クエリに対して最大 8 の正常なインスタンスの IP アドレスを返します。正常なインスタンスが 8 未満の場合、Route 53 は DNS クエリに対してすべての正常なインスタンスの IP アドレスを返します。

サービスでヘルスチェックを定義していない場合、Route 53 はすべてのインスタンスが正常であると仮定し、最大 8 個のインスタンスの値を返します。

詳細については、Amazon Route 53 デベロッパーガイドのマルチバリューアンサールーティングを参照してください。

レコードタイプ

Route 53 DNS レコードタイプは、サービスインスタンスの検出に使用される DNS クエリに応答して Route 53 が返す値のタイプを決定します。指定できるさまざまな DNS レコードタイプと、クエリに応答して Route 53 によって返される関連値は次のとおりです。

A

このタイプを指定すると、Route 53 は 192.0.2.44 などの IPv4 形式でリソースの IP アドレスを返します。

AAAA

このタイプを指定すると、Route 53 は 2001:0db8:85a3:0000:0000:abcd:0001:2345 などの IPv6 形式でリソースの IP アドレスを返します。

CNAME

このタイプを指定すると、Route 53 はリソースのドメイン名 (www.example.com など) を返します。

> **Note**
>
> - CNAME DNS レコードを設定するには、加重ルーティングルーティングポリシーを指定する必要があります。
> - CNAME DNS レコードを設定する場合、Route 53 ヘルスチェックを設定することはできません。

SRV

このタイプを指定すると、Route 53 は SRVレコードの値を返します。SRV レコードの値には、次の値が使用されます。

```
priority weight port service-hostname
```

以下の点を考慮してください。

- `priority` および `weight` の値は 1 に設定され、変更することはできません。
- の場合`port`、インスタンスの登録時にポート (AWS_INSTANCE_PORT) に指定した値 AWS Cloud Map を使用します。
- 値 `service-hostname` は、次の値を連結したものです。
  - インスタンスの登録時にサービスインスタンス ID (InstanceID) に指定する値
  - サービスの名前
  - 名前空間の名前

  たとえば、インスタンスを登録するときにテストをインスタンス ID として指定するとします。サービスの名前はバックエンドであり、名前空間の名前は example.com です。 AWS Cloud Map は、SRV レコードで `service-hostname` 属性に次の値を割り当てます。

  ```
  test.backend.example.com
  ```

> **Note**
>
> インスタンスの登録時に IPv4 アドレス、IPv6 アドレス、またはその両方の値を指定すると、 は SRV レコード`service-hostname`の の値と同じ名前の A および/または AAAA レコード AWS Cloud Map を自動的に作成します。

レコードタイプは、次の組み合わせで指定することができます。

- A
- AAAA
- A および AAAA
- CNAME
- SRV

レコードタイプ [A] および [AAAA] を指定した場合は、インスタンス登録時に IPv4 IP アドレス、IPv6 IP アドレス、またはその両方を指定することができます。

アプリケーションコンポーネントの AWS Cloud Map サービスの作成

名前空間を作成したら、特定の目的を果たすアプリケーションのさまざまなコンポーネントを表すサービスを作成できます。たとえば、支払いを処理するアプリケーション内のリソースのサービスを作成できます。

> **Note**
>
> DNS クエリによってアクセス可能な複数のサービスを、大文字と小文字だけが異なる名前 (EXAMPLE や例など) で作成することはできません。そうしようとすると、これらのサービスが同じ DNS 名になります。API コールでのみアクセス可能な名前空間を使用する場合は、大文字と小文字によってのみ異なる名前を持つサービスを作成できます。

AWS マネジメントコンソール、、および SDK for Python を使用してサービスを作成するには AWS CLI、次の手順に従います。

AWS マネジメントコンソール

1. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。
2. ナビゲーションペインで [名前空間] を選択します。
3. [名前空間] ページで、サービスを追加する名前空間を選択します。
4. [名前空間: ***namespace-name***] ページで、[サービスの作成] を選択します。
5. サービス名には、このサービスを使用するときに登録するインスタンスを説明する名前を入力します。値は、API コールまたは DNS クエリで AWS Cloud Map サービスインスタンスを検出するために使用されます。

   > **Note**
   >
   > インスタンスの登録時に SRV レコード AWS Cloud Map を作成し、特定の SRV 形式 (HAProxy など) を必要とするシステムを使用している場合は、サービス名に以下を指定します。
   >
   > - 例えば、アンダースコア (_) で名前を開始します。例、_exampleservice。
   > - *._protocol* で名前を終わらせる。例、._tcp。

インスタンスを登録すると、 は SRV レコード AWS Cloud Map を作成し、サービス名と名前空間名を連結して名前を割り当てます。次に例を示します。_exampleservice._tcp.example.com

6. (オプション) サービスの説明に、サービスの説明を入力します。ここで入力した説明は、サービスページと各サービスの詳細ページに表示されます。
7. 名前空間が DNS クエリをサポートしている場合は、サービス検出設定で、サービスレベルで検出可能性を設定できます。API コールと DNS クエリの両方を許可するか、このサービス内のインスタンスを検出するための API コールのみを許可するかを選択します。

> **Note**
>
> API コールを選択した場合、 AWS Cloud Map はインスタンスの登録時に SRV レコードを作成しません。

API と DNS を選択した場合は、以下の手順に従って DNS レコードを設定します。DNS レコードを追加または削除できます。

1. ルーティングポリシーで、インスタンスの登録時に が作成する DNS レコード AWS Cloud Map の Amazon Route 53 ルーティングポリシーを選択します。加重ルーティングと複数値回答ルーティングを選択できます。詳細については、「ルーティングポリシー」を参照してください。

   > **Note**
   >
   > コンソールを使用して、インスタンスの登録時に Route 53 エイリアスレコードを作成する AWS Cloud Map ように を設定することはできません。プログラムでインスタンスを登録するときに Elastic Load Balancing ロードバランサーのエイリアスレコード AWS Cloud Map を作成する場合は、ルーティングポリシーの加重ルーティングを選択します。

2. レコードタイプで、DNS クエリに応答して Route 53 が返す内容を決定する DNS レコードタイプを選択します AWS Cloud Map。詳細については、「レコードタイプ」を参照してください。

3. TTL には、サービスレベルで有効期限 (TTL) 値を秒単位で定義する数値を指定します。TTL の値は、リゾルバーが別の DNS クエリを Amazon Route 53 に転送して、更新された設定を取得するまでに、DNS リゾルバーがこのレコードの情報をキャッシュする期間を決定します。

8. ヘルスチェック設定のヘルスチェックオプションで、サービスインスタンスに適用されるヘルスチェックのタイプを選択します。ヘルスチェックを設定しないか、Route 53 ヘルスチェックまたはインスタンスの外部ヘルスチェックのいずれかを選択できます。詳細については、「AWS Cloud Map サービスヘルスチェック設定」を参照してください。

> **Note**
>
> Route 53 ヘルスチェックは、パブリック DNS 名前空間のサービスに対してのみ設定できます。

Route 53 ヘルスチェックを選択した場合は、次の情報を入力します。

1. Failure しきい値には、サービスインスタンスがヘルスステータスを変更するために合格または不合格にする連続する Route 53 ヘルスチェックの数を定義する 1 ~ 10 の数値を指定します。
2. ヘルスチェックプロトコルでは、Route 53 がサービスインスタンスの状態をチェックするために使用する方法を選択します。
3. HTTP または HTTPS ヘルスチェックプロトコルを選択した場合、ヘルスチェックパスに、ヘルスチェックの実行時に Amazon Route 53 がリクエストするパスを指定します。パスには、ファイル `/docs/route53-health-check.html` などの任意の値を指定できます。リソースが正常である場合、2xx または 3xx 形式の HTTP ステータスコードが返されます。クエリ文字列パラメータ (`/welcome.html?language=jp&login=y` など) を含めることもできます。 AWS Cloud Map コンソールでは、先行するスラッシュ (/) 文字が自動的に追加されます。

Route 53 ヘルスチェックの詳細については、Amazon Route 53 デベロッパーガイドの「Amazon Route 53 がヘルスチェックが正常かどうかを判断する方法」を参照してください。

9. (オプション) タグ でタグを追加 を選択し、名前空間にタグを付けるキーと値を指定します。1 つ以上のタグを指定して、 名前空間に追加することができます。タグを使用すると、

AWS リソースをより簡単に管理できるようにリソースを分類できます。詳細については、「AWS Cloud Map リソースのタグ付け」を参照してください。

10. [Create service (サービスの作成)] を選択します。

AWS CLI

- `create-service` コマンドを使用してサービスを作成します。`###` の値を自分の値に置き換えてください。

```
aws servicediscovery create-service \
    --name service-name \
    --namespace-id  ns-xxxxxxxxxxx \
    --dns-config "NamespaceId=ns-
xxxxxxxxxxx,RoutingPolicy=MULTIVALUE,DnsRecords=[{Type=A,TTL=60}]"
```

出力:

```
{
        "Service": {
        "Id": "srv-xxxxxxxxxxx",
        "Arn": "arn:aws:servicediscovery:us-west-2:123456789012:service/srv-
xxxxxxxxxxx",
        "Name": "service-name",
        "NamespaceId": "ns-xxxxxxxxxxx",
        "DnsConfig": {
            "NamespaceId": "ns-xxxxxxxxxxx",
            "RoutingPolicy": "MULTIVALUE",
            "DnsRecords": [
                {
                    "Type": "A",
                    "TTL": 60
                }
            ]
        },
        "CreateDate": 1587081768.334,
        "CreatorRequestId": "567c1193-6b00-4308-bd57-ad38a8822d25"
    }
}
```

AWS SDK for Python (Boto3)

まだBoto3がインストールしていない場合は、[こちら]のインストール、設定、使用に関する説明をBoto3参照してください。

1. Boto3をインポートしてサービスとしてservicediscoveryを使用してください。

```
import boto3
client = boto3.client('servicediscovery')
```

2. を使用してサービスを作成しますcreate_service()。### の値を自分の値に置き換えてください。詳細については、「create_service」を参照してください。

```
response = client.create_service(
    DnsConfig={
        'DnsRecords': [
            {
                'TTL': 60,
                'Type': 'A',
            },
        ],
        'NamespaceId': 'ns-xxxxxxxxxxx',
        'RoutingPolicy': 'MULTIVALUE',
    },
    Name='service-name',
    NamespaceId='ns-xxxxxxxxxxx',
)
```

レスポンスオブジェクトの例

```
{
    'Service': {
        'Arn': 'arn:aws:servicediscovery:us-west-2:123456789012:service/srv-
xxxxxxxxxxx',
        'CreateDate': 1587081768.334,
        'DnsConfig': {
            'DnsRecords': [
                {
                    'TTL': 60,
                    'Type': 'A',
                },
            ],
```

```
            'NamespaceId': 'ns-xxxxxxxxxxx',
            'RoutingPolicy': 'MULTIVALUE',
        },
        'Id': 'srv-xxxxxxxxxxx',
        'Name': 'service-name',
        'NamespaceId': 'ns-xxxxxxxxxxx',
    },
    'ResponseMetadata': {
        '...': '...',
    },
}
```

次の手順

サービスを作成したら、アプリケーションリソースを、アプリケーションがリソースを見つける方法に関する情報を含むサービスインスタンスとして登録できます。 AWS Cloud Map サービスインスタンスの登録の詳細については、「」を参照してくださいリソースを AWS Cloud Map サービスインスタンスとして登録する。

サービスの作成後に、エンドポイントの重み、API タイムアウト、再試行ポリシーなどのカスタムメタデータをサービス属性として指定することもできます。詳細については、 AWS Cloud Map API リファレンスのServiceAttributes」およびUpdateServiceAttributes「」を参照してください。

AWS Cloud Map サービスの更新

サービスの設定に応じて、DNS リゾルバーのタグ、Route 53 ヘルスチェックの失敗しきい値、有効期限 (TTL) を更新できます。サービスインスタンスを更新するには、次の手順を使用します。

> **Note**
>
> HTTP 名前空間に関連付けられたサービスの設定を更新することはできません。

AWS マネジメントコンソール

1. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。
2. ナビゲーションペインで [名前空間] を選択します。

3. 名前空間ページで、サービスが作成される名前空間を選択します。
4. 名前空間: ***namespace-name*** ページで、編集するサービスを選択し、詳細の表示を選択します。
5. サービス: ***service-name*** ページで、編集を選択します。

> **Note**
>
> 編集ボタンワークフローを使用して、インスタンス検出の API コールのみを許可するサービスの値を編集することはできません。ただし、サービス: ***service-name*** ページでタグを追加または削除することはできます。

6. サービスの編集ページのサービスの説明で、以前に設定したサービスの説明を更新したり、新しい説明を追加したりできます。タグを追加したり、DNS リゾルバーの TTL を更新したりすることもできます。
7. DNS 設定では、TTL に更新された期間を秒単位で指定できます。これにより、リゾルバーが別の DNS クエリを Amazon Route 53 に転送して更新された設定を取得するまでに、DNS リゾルバーがこのレコードの情報をキャッシュする時間を決定します。
8. Route 53 ヘルスチェックを設定している場合、障害しきい値には、サービスインスタンスがヘルスステータスを変更するために合格または不合格にする連続する Route 53 ヘルスチェックの数を定義する 1 から 10 までの新しい数を指定できます。
9. サービスの更新を選択します。

AWS CLI

- update-service コマンドでサービスを更新します (*red* の値は独自の値で置き換え)。

```
aws servicediscovery update-service \
    --id  srv-xxxxxxxxxxx \
    --service "Description=new
 description,DnsConfig={DnsRecords=[{Type=A,TTL=60}]}"
```

出力:

```
{
    "OperationId": "l3pfx7f4ynndrbj3cfq5fm2qy2z37bms-5m6iaoty"
}
```

AWS SDK for Python (Boto3)

1. まだBoto3がインストールしていない場合は、[こちら]のインストール、設定、使用に関する説明をBoto3参照してください。
2. Boto3をインポートしてサービスとしてservicediscoveryを使用してください。

```
import boto3
client = boto3.client('servicediscovery')
```

3. update_service()でサービスを作成します (*red* の値は独自の値で置き換え)。

```
response = client.update_service(
    Id='srv-xxxxxxxxxxx',
    Service={
        'DnsConfig': {
            'DnsRecords': [
                {
                    'TTL': 300,
                    'Type': 'A',
                },
            ],
        },
        'Description': "new description",
    }
)
```

レスポンスオブジェクトの例

```
{
    "OperationId": "l3pfx7f4ynndrbj3cfq5fm2qy2z37bms-5m6iaoty"
}
```

名前空間での AWS Cloud Map サービスの一覧表示

名前空間に作成したサービスのリストを表示するには、次の手順を実行します。

AWS マネジメントコンソール

1. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。

2. ナビゲーションペインで [名前空間] を選択します。
3. 一覧表示するサービスを含む名前空間のドメイン名を選択します。サービスですべてのサービスのリストを表示し、検索フィールドにサービス名または ID を入力して、特定のサービスを検索できます。作成元フィールドを使用してサービスを AWS アカウント 作成した を識別し、リソース所有者フィールドを使用してサービスを所有するアカウントを識別できます。

> **Note**
>
> 名前空間が共有名前空間の場合、リソース所有者の下の AWS アカウント ID は、名前空間を作成して共有したアカウントです。名前空間コンシューマーがサービスを作成した場合、 によって作成された のアカウント ID はリソース所有者の ID と異なる場合があります。アカウント IDs は、アカウント ID と同じではない場合があります。共有名前空間の詳細については、「」を参照してください共有 AWS Cloud Map 名前空間。

AWS CLI

- `list-services` コマンドでサービスを一覧表示します。次のコマンドは、名前空間 ID をフィルターとして使用して、名前空間内のすべてのサービスを一覧表示します。赤の値を独自の値に置き換えます。

```
aws servicediscovery list-services --filters
 Name=NAMESPACE_ID,Values=ns-1234567890abcdef,Condition=EQ
```

AWS SDK for Python (Boto3)

1. まだBoto3がインストールしていない場合は、[こちら]のインストール、設定、使用に関する説明をBoto3参照してください。
2. Boto3をインポートしてサービスとしてservicediscoveryを使用してください。

```
import boto3
client = boto3.client('servicediscovery')
```

3. `list_services()` でサービスを一覧表示する。

```
response = client.list_services()
# If you want to see the response
print(response)
```

レスポンスオブジェクトの例

```
{
    'Services': [
        {
            'Arn': 'arn:aws:servicediscovery:us-west-2:123456789012:service/srv-
xxxxxxxxxxxxxxx',
            'CreateDate': 1587081768.334,
            'DnsConfig': {
                'DnsRecords': [
                    {
                        'TTL': 60,
                        'Type': 'A',
                    },
                ],
                'RoutingPolicy': 'MULTIVALUE',
            },
            'Id': 'srv-xxxxxxxxxxxxxxxx',
            'Name': 'myservice',
        },
    ],
    'ResponseMetadata': {
        '...': '...',
    },
}
```

AWS Cloud Map サービスの削除

サービスを削除する前に、サービスを使用して登録されたサービスインスタンスはすべて、登録解除する必要があります。詳細については、「AWS Cloud Map サービスインスタンスの登録解除」を参照してください。

サービスを使用して登録されたすべてのインスタンスを登録解除したら、次の手順を実行してサービスを削除します。

AWS マネジメントコンソール

1. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。
2. ナビゲーションペインで [名前空間] を選択します。
3. 削除するサービスを含む名前空間のオプションを選択します。
4. [名前空間:*namespace-name*] ページで、削除するサービスのオプションを選択します。
5. [削除] をクリックします。
6. サービスを削除することを確認します。

AWS CLI

- `delete-service` コマンドでサービスを削除します (*red* の値は独自の値で置き換え)。

```
aws servicediscovery delete-service --id srv-xxxxxx
```

AWS SDK for Python (Boto3)

1. まだBoto3がインストールしていない場合は、[こちら]のインストール、設定、使用に関する説明をBoto3参照してください。
2. Boto3をインポートしてサービスとしてservicediscoveryを使用してください。

```
import boto3
client = boto3.client('servicediscovery')
```

3. delete_service()でサービスを削除します (*red* の値は独自の値で置き換え)。

```
response = client.delete_service(
    Id='srv-xxxxxx',
)
# If you want to see the response
print(response)
```

レスポンスオブジェクトの例

```
{
    'ResponseMetadata': {
```

```
        '...': '...',
    },
}
```

AWS Cloud Map サービスインスタンス

サービスインスタンスには、アプリケーションのリソース (例: ウェブサーバー) を検索する方法に関する情報が含まれます。インスタンスを登録したら、DNS クエリまたは AWS Cloud Map DiscoverInstances API アクションを使用してインスタンスを見つけます。登録できるリソースには以下が含まれますが、これらに限定されません。

- Amazon EC2 インスタンス
- Amazon DynamoDB テーブル
- Amazon S3 バケット
- Amazon Simple Queue Service (Amazon SQS) キュー
- Amazon APIsデプロイされた API Amazon API Gateway

サービスインスタンスの属性値を指定でき、クライアントはこれらの属性を使用して が AWS Cloud Map 返すリソースをフィルタリングできます。たとえば、アプリケーションから、BETA や PROD などの特定のデプロイステージでリソースをリクエストできます。バージョニングに属性を使用することもできます。

次の手順では、アプリケーション内のリソースをサービスインスタンスとして登録する方法、サービス内の登録済みインスタンスのリストを表示する方法、特定のインスタンスパラメータを編集する方法、およびインスタンスの登録を解除する方法について説明します。

トピック

- リソースを AWS Cloud Map サービスインスタンスとして登録する
- AWS Cloud Map サービスインスタンスの一覧表示
- AWS Cloud Map サービスインスタンスの更新
- AWS Cloud Map サービスインスタンスの登録解除

リソースを AWS Cloud Map サービスインスタンスとして登録する

アプリケーションのリソースを AWS Cloud Map サービス内のインスタンスとして登録できます。たとえば、ユーザーデータを管理するすべてのアプリケーションリソース`users`に対して というサービスを作成しているとします。その後、このサービスのインスタンスとしてユーザーデータを保存するために使用される DynamoDB テーブルを登録できます。

> **Note**
>
> 以下の機能は AWS Cloud Map コンソールでは使用できません。
>
> - コンソールを使用してサービスインスタンスを登録する場合は、 Elastic Load Balancing (ELB) ロードバランサーにトラフィックをルーティングするエイリアスレコードは作成できません。インスタンスを登録する場合は、`AWS_ALIAS_DNS_NAME` 属性を含める必要があります。詳細については、AWS Cloud Map API リファレンスの RegisterInstance を参照してください。
> - カスタムのヘルスチェックを含むサービスを使用してインスタンスを登録する場合、カスタムのヘルスチェックの初期ステータスは指定できません。デフォルトでは、カスタムのヘルスチェックの初期ステータスは [正常] です。初期のヘルスステータスを [異常] にするには、プログラムでインスタンスを登録し、`AWS_INIT_HEALTH_STATUS` 属性を含めます。詳細については、AWS Cloud Map API リファレンスの RegisterInstance を参照してください。

サービスにインスタンスを登録するには、次の手順に従います。

AWS マネジメントコンソール

1. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。
2. ナビゲーションペインで [名前空間] を選択します。
3. [名前空間] ページで、サービスインスタンスを登録するためのテンプレートとして使用するサービスを含む名前空間を選択します。
4. [名前空間:***namespace-name***] ページで、使用するサービスを選択します。
5. [Service:***service-name***] ページで、[サービスインスタンスの登録] を選択します。
6. サービスインスタンスの登録ページで、インスタンスタイプを選択します。名前空間インスタンス検出設定に応じて、IP アドレス、Amazon EC2 インスタンス ID、または IP アドレスを持たないリソースのその他の識別情報を指定できます。

   > **Note**
   >
   > EC2 インスタンスは HTTP 名前空間でのみ選択できます。

7. サービスインスタンス ID には、サービスインスタンスに関連付けられた識別子を指定します。

> **Note**
>
> 既存のインスタンスを更新する場合は、更新するインスタンスに関連付けられた識別子を指定します。次に、次のステップを使用して値を更新し、インスタンスを再登録します。

8. 選択したインスタンスタイプに基づいて、次の手順を実行します。

> **Important**
>
> カスタム属性を指定する場合、キーでAWS_プレフィックス (大文字と小文字を区別しない) を使用することはできません。

インスタンスタイプ	Steps
IP アドレス	a. 標準属性の IPv4 アドレスには、アプリケーションがこのサービスインスタンスに関連付けられているリソースにアクセスできる IPv4 アドレスがあれば指定します。 b. IPv6 アドレスの場合は、アプリケーションがこのサービスインスタンスに関連付けられているリソースにアクセスできる IPv6 IP アドレスがあれば指定します。 c. ポートには、このサービスインスタンスに関連付けられているリソースに

インスタンスタイプ	Steps	
	アクセスするためにアプリケーションに含める必要があるポートを指定します。ポートは、サービスに SRV レコードまたは Amazon Route 53 ヘルスチェックが含まれている場合に必要です。 d. (オプション) カスタム属性で、リソースに関連付けるキーと値のペアを指定します。	
EC2 インスタンス	a. EC2 インスタンス ID で、 AWS Cloud Map サービスインスタンスとして登録する Amazon EC2 インスタンスの ID を選択します。 b. (オプション) カスタム属性で、リソースに関連付けるキーと値のペアを指定します。	

インスタンスタイプ	Steps	
別のリソースを特定するための情報	a. 標準属性で、サービス設定に CNAME DNS レコードが含まれている場合、CNAME フィールドが表示されます。CNAME には、DNS クエリに応答して Route 53 が返すドメイン名を指定します (例: `example.com`)。 b. カスタム属性で、IP アドレスまたは Amazon EC2 インスタンス ID ではないリソースの識別情報をキーと値のペアとして指定します。たとえば、 というキーを指定`function`し、Lambda 関数の名前を値として指定することで、Lambda 関数を登録できます。 という名前のキーを指定`name`し、プログラムによるインスタンス検出に使用できる名前を指定することもできます。	

9. [サービスインスタンスの登録] を選択します。

AWS CLI

- `RegisterInstance` リクエストを送信すると:
  - `ServiceId` で指定したサービスで定義した DNS レコードごとに、対応する名前空間に関連付けられたホストゾーンでレコードが作成または更新されます。

- サービスに HealthCheckConfig が含まれる場合、ヘルスチェック設定の設定に基づいてヘルスチェックが作成されます。
- ヘルスチェックは、新しいレコードまたは更新された各レコードに関連付けられます。

register-instance コマンドでサービスインスタンスを登録します (*red* の値は独自の値で置き換え)。

```
aws servicediscovery register-instance \
    --service-id srv-xxxxxxxxx \
    --instance-id myservice-xx \
    --attributes=AWS_INSTANCE_IPV4=172.2.1.3,AWS_INSTANCE_PORT=808
```

AWS SDK for Python (Boto3)

1. まだBoto3がインストールしていない場合は、[こちら]のインストール、設定、使用に関する説明をBoto3参照してください。
2. Boto3をインポートしてサービスとしてservicediscoveryを使用してください。

```
import boto3
client = boto3.client('servicediscovery')
```

3. RegisterInstance リクエストを送信すると:

- ServiceId で指定したサービスで定義した DNS レコードごとに、対応する名前空間に関連付けられたホストゾーンでレコードが作成または更新されます。
- サービスに HealthCheckConfig が含まれる場合、ヘルスチェック設定の設定に基づいてヘルスチェックが作成されます。
- ヘルスチェックは、新しいレコードまたは更新された各レコードに関連付けられます。

register_instance() コマンドでサービスインスタンスを登録します (*red* の値は独自の値で置き換え)。

```
response = client.register_instance(
    Attributes={
        'AWS_INSTANCE_IPV4': '172.2.1.3',
        'AWS_INSTANCE_PORT': '808',
    },
```

```
    InstanceId='myservice-xx',
    ServiceId='srv-xxxxxxxxx',
)
# If you want to see the response
print(response)
```

レスポンスオブジェクトの例

```
{
    'OperationId': '4yejorelbukcjzpnr6tlmrghsjwpngf4-k95yg2u7',
    'ResponseMetadata': {
        '...': '...',
    },
}
```

AWS Cloud Map サービスインスタンスの一覧表示

サービスを使用して登録したサービスインスタンスのリストを表示するには、次の手順を実行します。

AWS マネジメントコンソール

1. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。
2. ナビゲーションペインで [名前空間] を選択します。
3. サービスインスタンスを一覧表示するサービスを含む名前空間の名前を選択します。
4. サービスインスタンスの作成に使用したサービスの名前を選択します。サービスインスタンスの下にインスタンスのリストが表示されます。検索フィールドにインスタンス ID を入力して、特定のインスタンスを一覧表示できます。作成者フィールドには、インスタンスを登録 AWS アカウント した の ID が表示されます。

> **Note**
>
> インスタンスが登録されている名前空間が共有名前空間である場合、 によって作成された の AWS アカウント ID はアカウント ID と同じではない可能性があります。共有名前空間の詳細については、「」を参照してください共有 AWS Cloud Map 名前空間。

AWS CLI

- list-instances コマンドでサービスインスタンスを一覧表示します (*red* の値は独自の値で置き換え)。

```
aws servicediscovery list-instances --service-id srv-xxxxxxxxx
```

AWS SDK for Python (Boto3)

1. まだBoto3がインストールしていない場合は、[こちら]のインストール、設定、使用に関する説明をBoto3参照してください。
2. Boto3をインポートしてサービスとしてservicediscoveryを使用してください。

```
import boto3
client = boto3.client('servicediscovery')
```

3. list_instances() コマンドでサービスインスタンスを一覧表示します (*red* の値は独自の値で置き換え)。

```
response = client.list_instances(
    ServiceId='srv-xxxxxxxxx',
)
# If you want to see the response
print(response)
```

レスポンスオブジェクトの例

```
{
    'Instances': [
        {
            'Attributes': {
                'AWS_INSTANCE_IPV4': '172.2.1.3',
                'AWS_INSTANCE_PORT': '808',
            },
            'Id': 'i-xxxxxxxxxxxxxxxx',
        },
    ],
    'ResponseMetadata': {
        '...': '...',
    },
```

```
}
```

AWS Cloud Map サービスインスタンスの更新

更新する値に応じて、次の 2 つの方法でサービスインスタンスを更新できます。

- 値を更新する: カスタム属性を含め、登録時にサービスインスタンスに指定した値のいずれかを更新する場合は、サービスインスタンスを再登録し、すべての値を再指定する必要があります。「」の手順に従ってリソースを AWS Cloud Map サービスインスタンスとして登録する、サービスインスタンス ID の既存のサービスインスタンスのインスタンス ID を指定します。

  または、 `RegisterInstance` API を使用することもできます。および `InstanceIdServiceId`パラメータを使用して既存のインスタンスとサービスの ID を指定し、他の値を再指定できます。
- カスタム属性のみの更新: サービスインスタンスのカスタム属性のみを更新する場合は、インスタンスを再登録する必要はありません。これらの値のみを更新できます。「サービスインスタンスのカスタム属性の更新」を参照してください。

サービスインスタンスのカスタム属性の更新

サービスインスタンスのカスタム属性のみを更新するには

1. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。
2. ナビゲーションペインで [名前空間] を選択します。
3. [名前空間] ページで、サービスインスタンスを登録するために最初に使用したサービスを含む名前空間を選択します。
4. [Namespace:***namespace-name***] ページで、サービスインスタンスの登録に使用したサービスを選択します。
5. [Service: ***service-name***] ページで、更新するサービスインスタンスの名前を選択します。
6. [Custom attributes (カスタム属性)] セクションで、[Edit (編集)] を選択します。
7. [Edit service instance: ***instance-name***] ページで、カスタム属性の追加、削除、更新を行います。既存の属性のキーと値の両方を更新できます。
8. [Update service instance (サービスインスタンスを更新)] を選択します。

AWS Cloud Map サービスインスタンスの登録解除

サービスを削除する前に、サービスを使用して登録されたサービスインスタンスはすべて、登録解除する必要があります。

サービスインスタンスの登録を解除するには、次の手順を使用します。

AWS マネジメントコンソール

1. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/cloudmap/ で AWS Cloud Map コンソールを開きます。
2. ナビゲーションペインで [名前空間] を選択します。
3. 登録を解除するサービスインスタンスを含む名前空間のオプションを選択します。
4. 名前空間: ***namespace-name*** ページで、サービスインスタンスの登録に使用したサービスを選択します。
5. サービス: ***service-name*** ページで、登録解除するサービスインスタンスを選択します。
6. [Deregister] (登録解除) を選択します。
7. サービスインスタンスの登録を解除することを確認します。

AWS CLI

- `deregister-instance` コマンドでサービスインスタンスを登録解除します (*red* の値は独自の値で置き換え)。このコマンドは、Amazon Route 53 DNS レコードと、指定されたインスタンス用に が AWS Cloud Map 作成したヘルスチェックを削除します。

```
aws servicediscovery deregister-instance \
    --service-id srv-xxxxxxxxx \
    --instance-id myservice-53
```

AWS SDK for Python (Boto3)

1. まだBoto3がインストールしていない場合は、[こちら]のインストール、設定、使用に関する説明をBoto3参照してください。
2. Boto3をインポートしてサービスとしてservicediscoveryを使用してください。

```
import boto3
```

```
client = boto3.client('servicediscovery')
```

3. deregister-instance()でサービスインスタンスを登録解除します (*red* の値は独自の値で置き換え)。このコマンドは、Amazon Route 53 DNS レコードと、指定されたインスタンス用に が AWS Cloud Map 作成したヘルスチェックを削除します。

```
response = client.deregister_instance(
    InstanceId='myservice-53',
    ServiceId='srv-xxxxxxxxx',
)
# If you want to see the response
print(response)
```

レスポンスオブジェクトの例

```
{
    'OperationId': '4yejorelbukcjzpnr6tlmrghsjwpngf4-k98rnaiq',
    'ResponseMetadata': {
        '...': '...',
    },
}
```

のセキュリティ AWS Cloud Map

のクラウドセキュリティが最優先事項 AWS です。お客様は AWS 、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を活用できます。

セキュリティは、 AWS とお客様の間の責任共有です。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 – AWS 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 AWS 責任があります。 AWS また、 では、安全に使用できるサービスも提供しています。「AWS 」 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。が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 Cloud Map、AWS 「コンプライアンスプログラムによる対象範囲内のサービス」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、ユーザーは、データの機密性、会社の要件、適用される法律や規制など、その他の要因についても責任を負います。

以下のドキュメントは、 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 AWS Cloud Map。以下のトピックでは、セキュリティおよびコンプライアンスの目的を達成する AWS Cloud Map ように を設定する方法を示します。また、 AWS Cloud Map リソースのモニタリングや保護に役立つ他の AWS サービスの使用方法についても説明します。

トピック

- の Identity and Access Management AWS Cloud Map
- のコンプライアンス検証 AWS Cloud Map
- の耐障害性 AWS Cloud Map
- のインフラストラクチャセキュリティ AWS Cloud Map

の Identity and Access Management AWS Cloud Map

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) し、誰に AWS

Cloud Map リソースの使用を許可する (アクセス許可を付与する) かを制御します。IAM は、追加料金なしで使用できる AWS のサービス です。

トピック

- オーディエンス
- アイデンティティを使用した認証
- ポリシーを使用したアクセスの管理
- が IAM と AWS Cloud Map 連携する方法
- のアイデンティティベースのポリシーの例 AWS Cloud Map
- AWS の 管理ポリシー AWS Cloud Map
- AWS Cloud Map API アクセス許可リファレンス
- AWS Cloud Map ID とアクセスのトラブルシューティング

オーディエンス

AWS Identity and Access Management (IAM) の使用方法は、ロールによって異なります。

- サービスユーザー - 機能にアクセスできない場合は、管理者にアクセス許可をリクエストします (「AWS Cloud Map ID とアクセスのトラブルシューティング」を参照)。
- サービス管理者 - ユーザーアクセスを決定し、アクセス許可リクエストを送信します (「が IAM と AWS Cloud Map 連携する方法」を参照)
- IAM 管理者 - アクセスを管理するためのポリシーを作成します (「のアイデンティティベースのポリシーの例 AWS Cloud Map」を参照)

アイデンティティを使用した認証

認証は、ID 認証情報 AWS を使用して にサインインする方法です。、IAM ユーザー AWS アカウントのルートユーザー、または IAM ロールを引き受けることで認証される必要があります。

AWS IAM アイデンティティセンター (IAM Identity Center)、シングルサインオン認証、Google/Facebook 認証情報などの ID ソースからの認証情報を使用して、フェデレーティッド ID としてサインインできます。サインインの詳細については、「AWS サインイン ユーザーガイド」の「AWS アカウントにサインインする方法」を参照してください。

プログラムによるアクセスの場合、 は SDK と CLI AWS を提供してリクエストを暗号化して署名します。詳細については、「IAM ユーザーガイド」の「API リクエストに対するAWS 署名バージョン 4」を参照してください。

AWS アカウント ルートユーザー

を作成するときは AWS アカウント、すべての AWS のサービス および リソースへの完全なアクセス権を持つ AWS アカウント ルートユーザーと呼ばれる 1 つのサインインアイデンティティから始めます。日常的なタスクには、ルートユーザーを使用しないことを強くお勧めします。ルートユーザー認証情報を必要とするタスクについては、「IAM ユーザーガイド」の「ルートユーザー認証情報が必要なタスク」を参照してください。

フェデレーテッドアイデンティティ

ベストプラクティスとして、人間のユーザーが一時的な認証情報 AWS のサービス を使用して にアクセスするには、ID プロバイダーとのフェデレーションを使用する必要があります。

フェデレーティッド ID は、エンタープライズディレクトリ、ウェブ ID プロバイダー、または ID Directory Service ソースの認証情報 AWS のサービス を使用して にアクセスするユーザーです。フェデレーテッドアイデンティティは、一時的な認証情報を提供するロールを引き受けます。

アクセスを一元管理する場合は、 AWS IAM アイデンティティセンターをお勧めします。詳細については、「AWS IAM アイデンティティセンター ユーザーガイド」の「IAM アイデンティティセンターとは」を参照してください。

IAM ユーザーとグループ

IAM ユーザーは、特定の個人やアプリケーションに対する特定のアクセス許可を持つアイデンティティです。長期認証情報を持つ IAM ユーザーの代わりに一時的な認証情報を使用することをお勧めします。詳細については、IAM ユーザーガイドの「ID プロバイダーとのフェデレーションを使用して にアクセスする必要がある AWS」を参照してください。

IAM グループは、IAM ユーザーの集合を指定し、大量のユーザーに対するアクセス許可の管理を容易にします。詳細については、「IAM ユーザーガイド」の「IAM ユーザーに関するユースケース」を参照してください。

IAM ロール

IAM ロールは、特定のアクセス許可を持つアイデンティであり、一時的な認証情報を提供します。ユーザーから IAM ロール (コンソール) に切り替えるか、 または API オペレーションを呼び出すこ

とで、ロールを引き受けることができます。 AWS CLI AWS 詳細については、「IAM ユーザーガイド」の「ロールを引き受けるための各種方法」を参照してください。

IAM ロールは、フェデレーションユーザーアクセス、一時的な IAM ユーザーのアクセス許可、クロスアカウントアクセス、クロスサービスアクセス、および Amazon EC2 で実行するアプリケーションに役立ちます。詳細については、IAM ユーザーガイド の IAM でのクロスアカウントリソースアクセス を参照してください。

ポリシーを使用したアクセスの管理

でアクセスを制御する AWS には、ポリシーを作成し、ID AWS またはリソースにアタッチします。ポリシーは、ID またはリソースに関連付けられたときにアクセス許可を定義します。 は、プリンシパルがリクエストを行うときにこれらのポリシー AWS を評価します。ほとんどのポリシーは JSON ドキュメント AWS として に保存されます。JSON ポリシードキュメントの詳細については、「IAM ユーザーガイド」の「JSON ポリシー概要」を参照してください。

管理者は、ポリシーを使用して、どのプリンシパルがどのリソースに対して、どのような条件でアクションを実行できるかを定義することで、誰が何にアクセスできるかを指定します。

デフォルトでは、ユーザーやロールにアクセス許可はありません。IAM 管理者は IAM ポリシーを作成してロールに追加し、このロールをユーザーが引き受けられるようにします。IAM ポリシーは、オペレーションの実行方法を問わず、アクセス許可を定義します。

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のポリシーは、アイデンティティ (ユーザー、グループ、またはロール) にアタッチできる JSON アクセス許可ポリシードキュメントです。これらのポリシーは、アイデンティティがどのリソースに対してどのような条件下でどのようなアクションを実行できるかを制御します。アイデンティティベースポリシーの作成方法については、IAM ユーザーガイド の カスタマー管理ポリシーでカスタム IAM アクセス許可を定義する を参照してください。

アイデンティティベースのポリシーは、インラインポリシー (単一の ID に直接埋め込む) または管理ポリシー (複数の ID にアタッチされたスタンドアロンポリシー) にすることができます。管理ポリシーとインラインポリシーのいずれかを選択する方法については、「IAM ユーザーガイド」の「管理ポリシーとインラインポリシーのいずれかを選択する」を参照してください。

リソースベースのポリシー

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。例としては、IAM ロール信頼ポリシーや Amazon S3 バケットポリシーなどがあります。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。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。

リソースベースのポリシーは、そのサービス内にあるインラインポリシーです。リソースベースのポリシーでは、IAM の AWS マネージドポリシーを使用できません。

その他のポリシータイプ

AWS は、より一般的なポリシータイプによって付与されるアクセス許可の最大数を設定できる追加のポリシータイプをサポートしています。

- アクセス許可の境界 – アイデンティティベースのポリシーで IAM エンティティに付与することのできるアクセス許可の数の上限を設定します。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。
- サービスコントロールポリシー (SCP) - AWS Organizations内の組織または組織単位の最大のアクセス許可を指定します。詳細については、「AWS Organizations ユーザーガイド」の「サービスコントロールポリシー」を参照してください。
- リソースコントロールポリシー (RCP) – は、アカウント内のリソースで利用できる最大数のアクセス許可を定義します。詳細については、「AWS Organizations ユーザーガイド」の「リソースコントロールポリシー (RCP)」を参照してください。
- セッションポリシー – ロールまたはフェデレーションユーザーの一時セッションを作成する際にパラメータとして渡される高度なポリシーです。詳細については、「IAM ユーザーガイド」の「セッションポリシー」を参照してください。

複数のポリシータイプ

1 つのリクエストに複数のタイプのポリシーが適用されると、結果として作成されるアクセス許可を理解するのがさらに難しくなります。が複数のポリシータイプが関与する場合にリクエストを許可するかどうか AWS を決定する方法については、IAM ユーザーガイドの「ポリシー評価ロジック」を参照してください。

が IAM と AWS Cloud Map 連携する方法

IAM を使用して へのアクセスを管理する前に AWS Cloud Map、 で使用できる IAM 機能を確認してください AWS Cloud Map。

IAM 機能	AWS Cloud Map 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	なし
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)	あり
一時的な認証情報	あり
転送アクセスセッション (FAS)	あり
サービスロール	いいえ
サービスリンクロール	はい

AWS Cloud Map およびその他の AWS のサービスがほとんどの IAM 機能とどのように連携するかの概要については、「IAM ユーザーガイド」のAWS 「IAM と連携する のサービス」を参照してください。

のアイデンティティベースのポリシー AWS Cloud Map

アイデンティティベースのポリシーのサポート: 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 許可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、および条件をコントロールします。アイデンティティベースポリシーの作成方法については、「IAM ユーザーガイド」の「カスタマー管理ポリシーでカスタム IAM アクセス許可を定義する」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。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。

のアイデンティティベースのポリシーの例 AWS Cloud Map

AWS Cloud Map アイデンティティベースのポリシーの例を表示するには、「」を参照してくださいのアイデンティティベースのポリシーの例 AWS Cloud Map。

内のリソースベースのポリシー AWS Cloud Map

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コントロールできます。ポリシーがアタッチ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、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、全体のアカウント、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。詳細については、IAM ユーザーガイドのIAM でのクロスアカウントリソースアクセスを参照してください。

> **Note**
>
> AWS Resource Access Manager (AWS RAM) を使用して、 AWS Cloud Map 名前空間を安全に共有できます。リソースベースのポリシーは、 AWS RAM サービスによって名前空間に適用されます。詳細については、「共有 AWS Cloud Map 名前空間」を参照してください。

のポリシーアクション AWS Cloud Map

ポリシーアクションのサポート: あり

管理者は JSON AWS 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。

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。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。

AWS Cloud Map アクションのリストを確認するには、「サービス認可リファレンス」の「 で定義されるアクション AWS Cloud Map」を参照してください。

のポリシーアクションは、アクションの前に次のプレフィックス AWS Cloud Map を使用します。

```
servicediscovery
```

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": [
      "servicediscovery:action1",
      "servicediscovery:action2"
         ]
```

AWS Cloud Map アイデンティティベースのポリシーの例を表示するには、「」を参照してくださいのアイデンティティベースのポリシーの例 AWS Cloud Map。

のポリシーリソース AWS Cloud Map

ポリシーリソースのサポート: あり

管理者は JSON AWS 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 ポリシー要素はアクションが適用されるオブジェクトを指定します。ベストプラクティスとして、Amazon リソースネーム (ARN) を使用してリソースを指定します。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": "*"
```

AWS Cloud Map リソースタイプとその ARNs「 で定義されるリソース AWS Cloud Map」を参照してください。 どのアクションで各リソースの ARN を指定できるかについては、「AWS Cloud Map で定義されるアクション」を参照してください。

AWS Cloud Map アイデンティティベースのポリシーの例を表示するには、「」を参照してくださいのアイデンティティベースのポリシーの例 AWS Cloud Map。

のポリシー条件キー AWS Cloud Map

サービス固有のポリシー条件キーのサポート: あり

管理者は JSON AWS 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。

Condition 要素は、定義された基準に基づいてステートメントが実行される時期を指定します。イコールや未満などの条件演算子を使用して条件式を作成して、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。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。

AWS Cloud Map 条件キーのリストを確認するには、「サービス認可リファレンス」の「 の条件キー AWS Cloud Map」を参照してください。条件キーを使用できるアクションとリソースについては、「 で定義されるアクション AWS Cloud Map」を参照してください。

AWS Cloud Map では、IAM ポリシーをきめ細かくフィルタリングするために使用できる以下のサービス固有の条件キーがサポートされています。

`servicediscovery:NamespaceArn`

関連する名前空間の Amazon リソースネーム (ARN) を指定することで、オブジェクトの取得を可能にするフィルター。

`servicediscovery:NamespaceName`

関連する名前空間の名前を指定することで、オブジェクトの取得を可能にするフィルター。

`servicediscovery:ServiceArn`

関連するサービスの Amazon リソースネーム (ARN) を指定することで、オブジェクトの取得を可能にするフィルター。

`servicediscovery:ServiceName`

関連するサービスの名前を指定することで、オブジェクトの取得を可能にするフィルター。

servicediscovery:ServiceCreatedByAccount

サービスを AWS アカウント 作成した の ID を指定してオブジェクトを取得できるフィルター。

AWS Cloud Map アイデンティティベースのポリシーの例を表示するには、「」を参照してくださいのアイデンティティベースのポリシーの例 AWS Cloud Map。

ACLs AWS Cloud Map

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アクセス許可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。

を使用した ABAC AWS Cloud Map

ABAC (ポリシー内のタグ) のサポート: あり

属性ベースのアクセス制御 (ABAC) は、タグと呼ばれる属性に基づいてアクセス許可を定義する認可戦略です。IAM エンティティと AWS リソースにタグをアタッチし、プリンシパルのタグがリソースのタグと一致するときにオペレーションを許可するように ABAC ポリシーを設計できます。

タグに基づいてアクセスを管理するには、`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ありです。サービスが一部のリソースタイプに対してのみ 3 つの条件キーのすべてをサポートする場合、値は「部分的」になります。

ABAC の詳細については、「IAM ユーザーガイド」の「ABAC 認可でアクセス許可を定義する」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コントロール (ABAC) を使用する」を参照してください。

での一時的な認証情報の使用 AWS Cloud Map

一時的な認証情報のサポート: あり

一時的な認証情報は、 AWS リソースへの短期的なアクセスを提供し、フェデレーションまたはスイッチロールの使用時に自動的に作成されます。長期的なアクセスキーを使用する代わりに、一時的な認証情報を動的に生成 AWS することをお勧めします。詳細については、「IAM ユーザーガイド」の「IAM の一時的な認証情報」および「AWS のサービス と IAM との連携」を参照してください。

の転送アクセスセッション AWS Cloud Map

転送アクセスセッション (FAS) のサポート: あり

転送アクセスセッション (FAS) は、 を呼び出すプリンシパルのアクセス許可と AWS のサービス、ダウンストリームサービス AWS のサービス へのリクエストをリクエストする を使用します。FAS リクエストを行う際のポリシーの詳細については、「転送アクセスセッション」を参照してください。

のサービスロール AWS Cloud Map

サービスロールのサポート: なし

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 のサービスに許可を委任するロールを作成するを参照してください。

> **Warning**
>
> サービスロールの権限を変更すると、 AWS Cloud Map の機能が破損する可能性があります。 AWS Cloud Map が指示する場合にのみ、サービスロールを編集します。

のサービスにリンクされたロール AWS Cloud Map

サービスリンクロールのサポート: あり

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、サービスにリンクされたロールのアクセス許可を表示できますが、編集することはできません。

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、「IAM と提携するAWS のサービス」を参照してください。表の「サービスリンクロール」列に Yes と記載されたサービスを見つ

けます。サービスにリンクされたロールに関するドキュメントをサービスで表示するには、[はい] リンクを選択します。

のアイデンティティベースのポリシーの例 AWS Cloud Map

デフォルトでは、 ユーザーおよびロールには、 AWS Cloud Map リソースを作成または変更する権限はありません。IAM 管理者は、リソースで必要なアクションを実行するための権限をユーザーに付与する IAM ポリシーを作成できます。

これらのサンプルの JSON ポリシードキュメントを使用して IAM 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 ポリシーを作成する (コンソール)」を参照してください。

各リソースタイプの ARN の形式など AWS Cloud Map、 で定義されるアクションとリソースタイプの詳細については、「サービス認可リファレンス」の「 のアクション、リソース、および条件キー AWS Cloud Map」を参照してください。 ARNs

トピック

- ポリシーに関するベストプラクティス
- AWS Cloud Map コンソールの使用
- AWS Cloud Map コンソールアクセスの例
- AWS Cloud Map ユーザーに独自のアクセス許可の表示を許可する
- すべての AWS Cloud Map リソースへの読み取りアクセスを許可する
- AWS Cloud Map サービスインスタンスの例
- AWS Cloud Map サービスの作成例
- AWS Cloud Map 名前空間の作成の例

ポリシーに関するベストプラクティス

ID ベースのポリシーは、誰かがアカウント内の AWS Cloud Map リソースを作成、アクセス、または削除できるかどうかを決定します。これらのアクションでは、 AWS アカウントに費用が発生する場合があります。アイデンティティベースポリシーを作成したり編集したりする際には、以下のガイドラインと推奨事項に従ってください:

- AWS 管理ポリシーを開始し、最小特権のアクセス許可に移行 – ユーザーとワークロードにアクセス許可の付与を開始するには、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AWS 管理

ポリシーを使用します。これらは で利用できます AWS アカウント。ユースケースに固有の AWS カスタマー管理ポリシーを定義することで、アクセス許可をさらに減らすことをお勧めします。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 または ジョブ機能のAWS マネージドポリシー を参照してください。

- 最小特権を適用する – IAM ポリシーでアクセス許可を設定する場合は、タスクの実行に必要な許可のみを付与します。これを行うには、特定の条件下で特定のリソースに対して実行できるアクションを定義します。これは、最小特権アクセス許可とも呼ばれています。IAM を使用して許可を適用する方法の詳細については、IAM ユーザーガイド の IAM でのポリシーとアクセス許可 を参照してください。
- IAM ポリシーで条件を使用してアクセスをさらに制限する - ポリシーに条件を追加して、アクションやリソースへのアクセスを制限できます。たとえば、ポリシー条件を記述して、すべてのリクエストを SSL を使用して送信するように指定できます。条件を使用して、サービスアクションが などの特定の を通じて使用されている場合に AWS のサービス、サービスアクションへのアクセスを許可することもできます CloudFormation。詳細については、IAM ユーザーガイド の IAM JSON ポリシー要素:条件 を参照してください。
- IAM アクセスアナライザー を使用して IAM ポリシーを検証し、安全で機能的な権限を確保する - IAM アクセスアナライザー は、新規および既存のポリシーを検証して、ポリシーが IAM ポリシー言語 (JSON) および IAM のベストプラクティスに準拠するようにします。IAM アクセスアナライザーは 100 を超えるポリシーチェックと実用的な推奨事項を提供し、安全で機能的なポリシーの作成をサポートします。詳細については、IAM ユーザーガイド の IAM Access Analyzer でポリシーを検証する を参照してください。
- 多要素認証 (MFA) を要求する – で IAM ユーザーまたはルートユーザーを必要とするシナリオがある場合は AWS アカウント、MFA をオンにしてセキュリティを強化します。API オペレーションが呼び出されるときに MFA を必須にするには、ポリシーに MFA 条件を追加します。詳細については、IAM ユーザーガイド の MFA を使用した安全な API アクセス を参照してください。

IAM でのベストプラクティスの詳細については、IAM ユーザーガイド の IAM でのセキュリティのベストプラクティス を参照してください。

AWS Cloud Map コンソールの使用

AWS Cloud Map コンソールにアクセスするには、最小限のアクセス許可のセットが必要です。これらのアクセス許可により、 の AWS Cloud Map リソースの詳細を一覧表示および表示できます AWS アカウント。最小限必要な許可よりも制限が厳しいアイデンティティベースのポリシーを作成すると、そのポリシーを持つエンティティ (ユーザーまたはロール) に対してコンソールが意図したとおりに機能しません。

AWS CLI または AWS API のみを呼び出すユーザーには、最小限のコンソールアクセス許可を付与する必要はありません。代わりに、実行しようとしている API オペレーションに一致するアクションのみへのアクセスが許可されます。

ユーザーとロールが引き続き AWS Cloud Map コンソールを使用できるようにするには、エンティティに または AWS Cloud Map *ConsoleAccessReadOnly* AWS 管理ポリシーもアタッチします。詳細については、「IAM ユーザーガイド」の「ユーザーへのアクセス許可の追加」を参照してください。

AWS Cloud Map コンソールアクセスの例

AWS Cloud Map コンソールへのフルアクセスを許可するには、次のアクセス許可ポリシーでアクセス許可を付与します。

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "servicediscovery:*",
            "route53:GetHostedZone",
            "route53:ListHostedZonesByName",
            "route53:CreateHostedZone",
            "route53:DeleteHostedZone",
            "route53:ChangeResourceRecordSets",
            "route53:CreateHealthCheck",
            "route53:GetHealthCheck",
            "route53:DeleteHealthCheck",
            "route53:UpdateHealthCheck",
            "ec2:DescribeInstances",
            "ec2:DescribeVpcs",
            "ec2:DescribeRegions"
         ],
         "Resource":"*"
      }
   ]
}
```

アクセス許可が必要な理由は次のとおりです。

servicediscovery:*

すべての AWS Cloud Map アクションを実行できます。

route53:CreateHostedZone, **route53:GetHostedZone**, **route53:ListHostedZonesByName**, **route53:DeleteHostedZone**

パブリックおよびプライベート DNS 名前空間を作成および削除するときに、 がホストゾーン AWS Cloud Map を管理できるようにします。

route53:CreateHealthCheck, **route53:GetHealthCheck**, **route53:DeleteHealthCheck**, **route53:UpdateHealthCheck**

サービスの作成時に Amazon Route 53 ヘルスチェックを含めると、 でヘルスチェック AWS Cloud Map を管理できます。

ec2:DescribeVpcs および **ec2:DescribeRegions**

がプライベートホストゾーン AWS Cloud Map を管理できるようにします。

AWS Cloud Map ユーザーに独自のアクセス許可の表示を許可する

この例では、ユーザーアイデンティティにアタッチされたインラインおよびマネージドポリシーの表示を IAM ユーザーに許可するポリシーの作成方法を示します。このポリシーには、コンソールで、または AWS CLI または AWS API を使用してプログラムでこのアクションを実行するアクセス許可が含まれています。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
```

```
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

すべての AWS Cloud Map リソースへの読み取りアクセスを許可する

次のアクセス許可ポリシーでは、すべての AWS Cloud Map リソースへの読み取り専用アクセスをユーザーに付与します。

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "servicediscovery:Get*",
            "servicediscovery:List*",
            "servicediscovery:DiscoverInstances"
         ],
         "Resource":"*"
      }
   ]
}
```

AWS Cloud Map サービスインスタンスの例

次の例は、サービスインスタンスを登録、登録解除、検出するアクセス許可をユーザーに付与するアクセス許可ポリシーを示しています。Sid (ステートメント ID) はオプションです。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid" : "AllowInstancePermissions",
         "Effect": "Allow",
         "Action": [
            "servicediscovery:RegisterInstance",
            "servicediscovery:DeregisterInstance",
            "servicediscovery:DiscoverInstances",
            "servicediscovery:Get*",
            "servicediscovery:List*",
            "route53:GetHostedZone",
            "route53:ListHostedZonesByName",
            "route53:ChangeResourceRecordSets",
            "route53:CreateHealthCheck",
            "route53:GetHealthCheck",
            "route53:DeleteHealthCheck",
            "route53:UpdateHealthCheck",
            "ec2:DescribeInstances"
         ],
         "Resource": "*"
      }
   ]
}
```

このポリシーでは、サービスインスタンスの登録と管理に必要なアクションに対するアクセス許可が付与されます。パブリックまたはプライベート DNS 名前空間を使用している場合は、Route 53 アクセス許可が必要です。インスタンスを登録および登録解除すると、 は Route 53 レコードとヘルスチェック AWS Cloud Map を作成、更新、削除するためです。のワイルドカード文字 (*) は、すべての AWS Cloud Map インスタンス、および現在の AWS アカウントが所有する Route 53 レコードとヘルスチェックへのアクセスResourceを許可します。

AWS Cloud Map サービスの作成例

IAM ID がサービスを作成 AWS Cloud Map できるようにするアクセス許可ポリシーを追加する場合は、リソースフィールドに名前空間とサービスの両方 AWS Cloud Map の Amazon リソースネーム (ARN) を指定する必要があります。ARN には、リージョン、アカウント ID、名前空間 ID が含まれます。サービスのサービス ID がまだわからないため、ワイルドカードを使用することをお勧めします。以下は、ポリシースニペットの例です。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "servicediscovery:CreateService"
            ],
            "Resource": [
                "arn:aws:servicediscovery:us-east-1:111122223333:namespace/ns-
p32123EXAMPLE",
                "arn:aws:servicediscovery:us-east-1:111122223333:service/*"
            ]
        }
    ]
}
```

AWS Cloud Map 名前空間の作成の例

次のアクセス許可ポリシーでは、ユーザーはすべてのタイプの AWS Cloud Map 名前空間を作成できます。

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
```

```
         "Action":[
            "servicediscovery:CreateHttpNamespace",
            "servicediscovery:CreatePrivateDnsNamespace",
            "servicediscovery:CreatePublicDnsNamespace",
            "route53:CreateHostedZone",
            "route53:GetHostedZone",
            "route53:ListHostedZonesByName",
            "ec2:DescribeVpcs",
            "ec2:DescribeRegions"
         ],
         "Resource":"*"
      }
   ]
}
```

AWS の 管理ポリシー AWS Cloud Map

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。ユースケースに固有のカスタマー管理ポリシーを定義して、アクセス許可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 マネージド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) に影響します。 AWS は、新しい が起動されるか、新しい API オペレーション AWS のサービス が既存のサービスで使用できるようになったときに、 AWS マネージド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。

AWS マネージドポリシー: AWSCloudMapDiscoverInstanceAccess

IAM エンティティに `AWSCloudMapDiscoverInstanceAccess` をアタッチできます。 AWS Cloud Map Discovery API へのアクセスを提供します。

このポリシーに対する許可を確認するには、「AWS マネージドポリシーリファレンス」の「AWSCloudMapDiscoverInstanceAccess」を参照してください。

AWS マネージドポリシー: AWSCloudMapReadOnlyAccess

IAM エンティティに AWSCloudMapReadOnlyAccess をアタッチできます。すべての AWS Cloud Map アクションへの読み取り専用アクセスを許可します。

このポリシーに対する許可を確認するには、「AWS マネージドポリシーリファレンス」の「AWSCloudMapReadOnlyAccess」を参照してください。

AWS マネージドポリシー: AWSCloudMapRegisterInstanceAccess

IAM エンティティに AWSCloudMapRegisterInstanceAccess をアタッチできます。名前空間とサービスへの読み取り専用アクセス権を付与し、サービスインスタンスの登録と登録解除を行うアクセス許可を付与します。

このポリシーに対する許可を確認するには、「AWS マネージドポリシーリファレンス」の「AWSCloudMapRegisterInstanceAccess」を参照してください。

AWS マネージドポリシー: AWSCloudMapFullAccess

IAM エンティティに AWSCloudMapFullAccess をアタッチできます。すべての AWS Cloud Map アクションへのフルアクセスを提供します

このポリシーに対する許可を確認するには、「AWS マネージドポリシーリファレンス」の「AWSCloudMapFullAccess」を参照してください。

AWS Cloud Map AWS 管理ポリシーの更新

このサービスがこれらの変更の追跡を開始 AWS Cloud Map してからの の AWS 管理ポリシーの更新に関する詳細を表示します。変更に関する自動アラートについては、 AWS Cloud Map ドキュメント履歴ページの RSS フィードにサブスクライブしてください。

変更	説明	日付
AWSCloudMapDiscoverInstanceAccess、AWSCloudMapRegisterInstance	AWS Cloud Map は、これらのポリシーを更新して、新しい AWS Cloud Map DiscoverI	2023 年 8 月 15 日

変更	説明	日付
Access、AWSCloudMapReadOnlyAccess — 既存のポリシーの更新。	nstanceRevision API オペレーションへのアクセスを提供しました。	

AWS Cloud Map API アクセス許可リファレンス

アクセスコントロールを設定し、IAM アイデンティティ (アイデンティティベースのポリシー) にアタッチできるアクセス許可ポリシーを記述するときは、次のリストをリファレンスとして使用できます。リストには、各 AWS Cloud Map API アクションと、アクセス許可を付与する必要があるアクションが含まれます。ポリシーの `Action`フィールドにアクションを指定します。`Resource` フィールドまたは IAM ポリシーで指定する必要があるリソース値の詳細については、「サービス認可リファレンス」の「 のアクション、リソース、および条件キー AWS Cloud Map」を参照してください。

一部のオペレーションでは、IAM ポリシーで AWS Cloud Map固有の条件キーを使用できます。詳細については、「Service Authorization Reference」の「Condition keys for AWS Cloud Map」を参照してください。

アクションを指定するには、`servicediscovery` プレフィックス、API アクション名の順の文字列を使用します (例: `servicediscovery:CreatePublicDnsNamespace` および `route53:CreateHostedZone`)。

AWS Cloud Map アクションに必要なアクセス許可

CreateHttpNamespace

必要なアクセス許可 (API アクション):

- `servicediscovery:CreateHttpNamespace`

CreatePrivateDnsNamespace

必要なアクセス許可 (API アクション):

- `servicediscovery:CreatePrivateDnsNamespace`
- `route53:CreateHostedZone`
- `route53:GetHostedZone`
- `route53:ListHostedZonesByName`

- ec2:DescribeVpcs
- ec2:DescribeRegions

CreatePublicDnsNamespace

必要なアクセス許可 (API アクション):

- servicediscovery:CreatePublicDnsNamespace
- route53:CreateHostedZone
- route53:GetHostedZone
- route53:ListHostedZonesByName

CreateService

必要なアクセス許可 (API アクション): servicediscovery:CreateService

DeleteNamespace

必要なアクセス許可 (API アクション):

- servicediscovery:DeleteNamespace

DeleteService

必要なアクセス許可 (API アクション): servicediscovery:DeleteService

DeleteServiceAttributes

必要なアクセス許可 (API アクション): servicediscovery:DeleteServiceAttributes

DeregisterInstance

必要なアクセス許可 (API アクション):

- servicediscovery:DeregisterInstance
- route53:GetHealthCheck
- route53:DeleteHealthCheck
- route53:UpdateHealthCheck

DiscoverInstances

必要なアクセス許可 (API アクション): servicediscovery:DiscoverInstances

GetInstance

必要なアクセス許可 (API アクション): servicediscovery:GetInstance

GetInstancesHealthStatus

必要なアクセス許可 (API アクション): servicediscovery:GetInstancesHealthStatus

GetNamespace

必要なアクセス許可 (API アクション): servicediscovery:GetNamespace

GetOperation

必要なアクセス許可 (API アクション): servicediscovery:GetOperation

GetService

必要なアクセス許可 (API アクション): servicediscovery:GetService

GetServiceAttributes

必要なアクセス許可 (API アクション): servicediscovery:GetServiceAttributes

ListInstances

必要なアクセス許可 (API アクション): servicediscovery:ListInstances

ListNamespaces

必要なアクセス許可 (API アクション): servicediscovery:ListNamespaces

ListOperations

必要なアクセス許可 (API アクション): servicediscovery:ListOperations

ListServices

必要なアクセス許可 (API アクション): servicediscovery:ListServices

ListTagsForResource

必要なアクセス許可 (API アクション): servicediscovery:ListTagsForResource

RegisterInstance

必要なアクセス許可 (API アクション):

- servicediscovery:RegisterInstance
- route53:GetHealthCheck
- route53:CreateHealthCheck
- route53:UpdateHealthCheck

- ec2:DescribeInstances

TagResource

必要なアクセス許可 (API アクション): servicediscovery:TagResource

UntagResource

必要なアクセス許可 (API アクション): servicediscovery:UntagResource

UpdateHttpNamespace

必要なアクセス許可 (API アクション): servicediscovery:UpdateHttpNamespace

UpdateInstanceCustomHealthStatus

必要なアクセス許可 (API アクション):
servicediscovery:UpdateInstanceCustomHealthStatus

UpdatePrivateDnsNamespace

必要なアクセス許可 (API アクション):

- servicediscovery:UpdatePrivateDnsNamespace
- route53:ChangeResourceRecordSets

UpdatePublicDnsNamespace

必要なアクセス許可 (API アクション):

- servicediscovery:UpdatePublicDnsNamespace
- route53:ChangeResourceRecordSets

UpdateService

必要なアクセス許可 (API アクション):

- servicediscovery:UpdateService
- route53:GetHealthCheck
- route53:CreateHealthCheck
- route53:DeleteHealthCheck
- route53:UpdateHealthCheck

UpdateServiceAttributes

必要なアクセス許可 (API アクション): servicediscovery:UpdateServiceAttributes

AWS Cloud Map ID とアクセスのトラブルシューティング

次の情報は、 および IAM の使用時に発生する可能性がある一般的な問題の診断 AWS Cloud Map と修正に役立ちます。

トピック

- でアクションを実行する権限がありません AWS Cloud Map
- iam:PassRole を実行する権限がありません
- 自分の 以外のユーザーに自分の AWS Cloud Map リソース AWS アカウント へのアクセスを許可したい

でアクションを実行する権限がありません AWS Cloud Map

アクションを実行する権限がないというエラーが表示された場合は、そのアクションを実行できるようにポリシーを更新する必要があります。

次のエラー例は、mateojackson IAM ユーザーがコンソールを使用して、ある *my-example-widget* リソースに関する詳細情報を表示しようとしたことを想定して、その際に必要な servicediscovery:*GetWidget* アクセス許可を持っていない場合に発生するものです。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 servicediscovery:GetWidget on resource: my-example-widget
```

この場合、servicediscovery:*GetWidget* アクションを使用して *my-example-widget* リソースへのアクセスを許可するように、mateojackson ユーザーのポリシーを更新する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン認証情報を提供した担当者が管理者です。

iam:PassRole を実行する権限がありません

iam:PassRole アクションを実行する権限がないというエラーが表示された場合は、ポリシーを更新して AWS Cloud Mapにロールを渡すことができるようにする必要があります。

一部の AWS のサービス では、新しいサービスロールまたはサービスにリンクされたロールを作成する代わりに、そのサービスに既存のロールを渡すことができます。そのためには、サービスにロールを渡す権限が必要です。

以下の例のエラーは、marymajor という IAM ユーザーがコンソールを使用して AWS Cloud Mapでアクションを実行しようとする場合に発生します。ただし、このアクションをサービスが実行するには、サービスロールから付与された権限が必要です。Mary には、ロールをサービスに渡すアクセス許可がありません。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

この場合、Mary のポリシーを更新してメアリーに iam:PassRole アクションの実行を許可する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン資格情報を提供した担当者が管理者です。

自分の 以外のユーザーに自分の AWS Cloud Map リソース AWS アカウント へのアクセスを許可したい

他のアカウントのユーザーや組織外の人が、リソースにアクセスするために使用できるロールを作成できます。ロールの引き受けを委託するユーザーを指定できます。リソースベースのポリシーまたはアクセスコントロールリスト (ACL) をサポートするサービスの場合、それらのポリシーを使用して、リソースへのアクセスを付与できます。

詳細については、以下を参照してください:

- がこれらの機能 AWS Cloud Map をサポートしているかどうかを確認するには、「」を参照してくださいが IAM と AWS Cloud Map 連携する方法。
- 所有 AWS アカウント している 全体のリソースへのアクセスを提供する方法については、IAM ユーザーガイドの「所有 AWS アカウント している別の の IAM ユーザーへのアクセスを提供する」を参照してください。
- リソースへのアクセスをサードパーティーに提供する方法については AWS アカウント、IAM ユーザーガイドの「サードパーティー AWS アカウント が所有する へのアクセスを提供する」を参照してください。
- ID フェデレーションを介してアクセスを提供する方法については、IAM ユーザーガイド の 外部で認証されたユーザー (ID フェデレーション) へのアクセスの許可 を参照してください。
- クロスアカウントアクセスにおけるロールとリソースベースのポリシーの使用方法の違いについては、「IAM ユーザーガイド」の「IAM でのクロスアカウントのリソースへのアクセス」を参照してください。

のコンプライアンス検証 AWS Cloud Map

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AWS のサービス プログラムによる対象範囲内」の「コンプライアンス」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、「Downloading Reports in AWS Artifact」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、適用可能な法律および規制によって決まります。を使用する際のコンプライアンス責任の詳細については AWS のサービス、AWS 「 セキュリティドキュメント」を参照してください。

の耐障害性 AWS Cloud Map

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。 AWS リージョンは、低レイテンシー、高スループット、高度に冗長なネットワークで接続された、物理的に分離および分離された複数のアベイラビリティーゾーンを提供します。アベイラビリティーゾーンでは、アベイラビリティーゾーン間で中断せずに、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、耐障害性、およびスケーラビリティが優れています。

AWS Cloud Map は主にグローバルサービスです。ただし、 AWS Cloud Map を使用して、Amazon EC2 インスタンスや Elastic Load Balancing ロードバランサーなど、特定のリージョンのリソースのヘルスをチェックする Route 53 ヘルスチェックを作成できます。

AWS リージョンと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。

のインフラストラクチャセキュリティ AWS Cloud Map

マネージドサービスである AWS Cloud Map 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環境を AWS 設計するには、「Security Pillar AWS Well-Architected Framework」の「Infrastructure Protection」を参照してください。

AWS が公開した API コールを使用して、ネットワーク AWS Cloud Map 経由で 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 が必須で、TLS 1.3 をお勧めします。
- 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。

インターフェイス VPC エンドポイントを使用する AWS Cloud Map ように を設定することで、VPC のセキュリティ体制を改善できます。詳細については、「インターフェイスエンドポイント (AWS PrivateLink) AWS Cloud Map を使用した へのアクセス」を参照してください。

インターフェイスエンドポイント (AWS PrivateLink) AWS Cloud Map を使用した へのアクセス

を使用して AWS PrivateLink 、VPC と の間にプライベート接続を作成できます AWS Cloud Map。インターネットゲートウェイ、NAT デバイス、VPN 接続、または Direct Connect 接続を使用せずに、VPC 内にある AWS Cloud Map かのように にアクセスできます。VPC 内のインスタンスは AWS Cloud Mapにアクセスするためにパブリック IP アドレスを必要としません。

このプライベート接続を確立するには、 AWS PrivateLinkを利用したインターフェイスエンドポイントを作成します。インターフェイスエンドポイントに対して有効にする各サブネットにエンドポイントネットワークインターフェイスを作成します。これらは、 AWS Cloud Map宛てのトラフィックのエントリポイントとして機能するリクエスタ管理型ネットワークインターフェイスです。

詳細については「AWS PrivateLink ガイド」の「Access AWS のサービス through AWS PrivateLink」を参照してください。

に関する考慮事項 AWS Cloud Map

のインターフェイスエンドポイントを設定する前に AWS Cloud Map、「 AWS PrivateLink ガイド」の「考慮事項」を参照してください。

Amazon VPC にインターネットゲートウェイがなく、タスクが `awslogs` ログドライバーを使用して、ログ情報を CloudWatch Logs に送信する場合、CloudWatch Logs 用のインターフェース VPC

エンドポイントを作成する必要があります。詳細については、Amazon CloudWatch Logs ユーザーガイド」の「インターフェイス VPC エンドポイントでの CloudWatch Logs の使用」を参照してください。

VPC エンドポイントは AWS 、クロスリージョンリクエストをサポートしていません。 AWS Cloud Mapに対して API コールを発行するリージョンと同じリージョンにエンドポイントを作成してください。

VPC エンドポイントでは、Amazon Route 53 を介して Amazon 提供の DNS のみがサポートされています。独自の DNS を使用したい場合は、条件付き DNS 転送を使用できます。詳細については、Amazon VPC ユーザーガイド の「DHCP オプション設定」を参照してください。

VPC エンドポイントにアタッチされたセキュリティグループは、Amazon VPC のプライベートサブネットからのポート443での着信接続を許可する必要があります。

のインターフェイスエンドポイントを作成する AWS Cloud Map

Amazon VPC コンソールまたは AWS Command Line Interface () AWS Cloud Map を使用して、 のインターフェイスエンドポイントを作成できますAWS CLI。詳細については、「AWS PrivateLink ガイド」の「インターフェイスエンドポイントを作成」を参照してください。

次のサービス名 AWS Cloud Map を使用して のインターフェイスエンドポイントを作成します。

> **Note**
>
> `DiscoverInstances` API は、これら 2 つのエンドポイントでは使用できません。

```
com.amazonaws.region.servicediscovery
```

```
com.amazonaws.region.servicediscovery-fips
```

次のサービス名を使用して AWS Cloud Map 、データプレーンが `DiscoverInstances` API にアクセスするためのインターフェイスエンドポイントを作成します。

```
com.amazonaws.region.data-servicediscovery
```

```
com.amazonaws.region.data-servicediscovery-fips
```

> **Note**
>
> データプレーンエンドポイントのリージョンまたはゾーンごとの VPCE DNS 名で呼び出す場合は、`DiscoverInstances` ホストプレフィックスインジェクションを無効にする必要があります。 AWS CLI 各 API オペレーションを呼び出すと、サービスエンドポイントの前にさまざまなホストプレフィックスが付加され、 AWS SDKs エンドポイントを指定すると無効な URL が生成されます。

インターフェイスエンドポイントのプライベート DNS を有効にすると、リージョンのデフォルト DNS 名を使用して、 AWS Cloud Map への API リクエストを実行できます。例えば、`servicediscovery.us-east-1.amazonaws.com`。

VPCE AWS PrivateLink 接続は、 が AWS Cloud Map サポートされているすべてのリージョンでサポートされていますが、エンドポイントを定義する前に、VPCE をサポートするアベイラビリティーゾーンを確認する必要があります。リージョン内のインターフェイス VPC エンドポイントでサポートされているアベイラビリティーゾーンを確認するには、describe-vpc-endpoint-services コマンドを使用するか、 AWS マネジメントコンソールを使用します。たとえば、次のコマンドは、米国東部 (オハイオ) リージョン内の AWS Cloud Map インターフェイス VPC エンドポイントを配置できるアベイラビリティ ゾーンを返します：

```
aws --region us-east-2 ec2 describe-vpc-endpoint-services --query 'ServiceDetails[?
ServiceName==`com.amazonaws.us-east-2.servicediscovery`].AvailabilityZones[]'
```

モニタリング AWS Cloud Map

モニタリングは、 AWS ソリューションの信頼性、可用性、および性能を維持するうえで重要な部分です。マルチポイント障害が発生した場合は、その障害をより簡単にデバッグできるように、 AWS ソリューションのすべての部分からモニタリングデータを収集する必要があります。ただし、モニタリングを開始する前に、以下の質問に対する回答を反映したモニタリング計画を作成する必要があります。

- どのような目的でモニタリングしますか?
- どのリソースをモニタリングしますか?
- どのくらいの頻度でこれらのリソースをモニタリングしますか?
- どのモニタリングツールを利用しますか?
- 誰がモニタリングタスクを実行しますか?
- 問題が発生したときに誰が通知を受け取りますか?

トピック

- を使用した AWS Cloud Map API コールのログ記録 AWS CloudTrail

を使用した AWS Cloud Map API コールのログ記録 AWS CloudTrail

AWS Cloud Map は、ユーザーAWS CloudTrail、ロール、または によって実行されたアクションを記録するサービスである と統合されています AWS のサービス。CloudTrail は、 のすべての API コールをイベント AWS Cloud Map 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、AWS Cloud Map コンソールからの呼び出しと AWS Cloud Map API オペレーションへのコード呼び出しが含まれます。CloudTrail によって収集された情報を使用して、リクエストの実行元の IP アドレス AWS Cloud Map、リクエストの実行日時などの詳細を確認できます。

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。

- ルートユーザーまたはユーザー認証情報のどちらを使用してリクエストが送信されたか。
- リクエストが IAM Identity Center ユーザーに代わっ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 AWS のサービスによって送信されたかどうか。

CloudTrail は、アカウントを作成する AWS アカウント と でアクティブになり、CloudTrail イベント履歴に自動的にアクセスできます。CloudTrail の [イベント履歴] では、 AWS リージョンで過去 90 日間に記録された 管理イベントの表示、検索、およびダウンロードが可能で、変更不可能な記録を確認できます。詳細については、「AWS CloudTrail ユーザーガイド」の「CloudTrail イベント履歴の使用」を参照してください。[イベント履歴] の閲覧には CloudTrail の料金はかかりません。

AWS アカウント 過去 90 日間のイベントの継続的な記録については、証跡または CloudTrail Lake イベントデータストアを作成します。

CloudTrail 証跡

証跡により、CloudTrail はログファイルを Amazon S3 バケットに配信できます。を使用して作成されたすべての証跡 AWS マネジメントコンソール はマルチリージョンです。 AWS CLIを使用する際は、単一リージョンまたは複数リージョンの証跡を作成できます。アカウント AWS リージョン 内のすべての でアクティビティをキャプチャするため、マルチリージョン証跡を作成することをお勧めします。単一リージョンの証跡を作成する場合、証跡の AWS リージョンに記録されたイベントのみを表示できます。証跡の詳細については、「AWS CloudTrail ユーザーガイド」の「AWS アカウントの証跡の作成」および「組織の証跡の作成」を参照してください。

証跡を作成すると、進行中の管理イベントのコピーを 1 つ無料で CloudTrail から Amazon S3 バケットに配信できますが、Amazon S3 ストレージには料金がかかります。CloudTrail の料金の詳細については、「AWS CloudTrail の料金」を参照してください。Amazon S3 の料金に関する詳細については、「Amazon S3 の料金」を参照してください。

CloudTrail Lake イベントデータストア

[CloudTrail Lake] を使用すると、イベントに対して SQL ベースのクエリを実行できます。CloudTrail Lake は、行ベースの JSON 形式の既存のイベントを Apache ORC 形式に変換します。ORC は、データを高速に取得するために最適化された単票ストレージ形式です。イベントは、イベントデータストアに集約されます。イベントデータストアは、高度なイベントセレクタを適用することによって選択する条件に基づいた、イベントのイミュータブルなコレクションです。どのイベントが存続し、クエリに使用できるかは、イベントデータストアに適用するセレクタが制御します。CloudTrail Lake の詳細については、 AWS CloudTrail ユーザーガイドのAWS CloudTrail 「Lake の使用」を参照してください。

CloudTrail Lake のイベントデータストアとクエリにはコストがかかります。イベントデータストアを作成する際に、イベントデータストアに使用する料金オプションを選択します。料金オプションによって、イベントの取り込みと保存にかかる料金、および、そのイベントデータストアのデフォルトと最長の保持期間が決まります。CloudTrail の料金の詳細については、「AWS CloudTrail の料金」を参照してください。

AWS Cloud Map CloudTrail のデータイベント

データイベントは、リソース上またはリソース内で実行されるリソースオペレーションに関する情報を提供します (たとえば、名前空間内の登録済みインスタンスの検出)。これらのイベントは、データプレーンオペレーションとも呼ばれます。データイベントは、多くの場合、高ボリュームのアクティビティです。デフォルトでは、CloudTrail はデータイベントをログ記録しません。CloudTrail [イベント履歴] にはデータイベントは記録されません。

追加の変更がイベントデータに適用されます。CloudTrail の料金の詳細については、「AWS CloudTrail の料金」を参照してください。

CloudTrail コンソール、または CloudTrail CloudTrail API オペレーションを使用して AWS CLI、AWS Cloud Map リソースタイプのデータイベントを記録できます。データイベントをログに記録する方法の詳細については、「AWS CloudTrail ユーザーガイド」の「AWS マネジメントコンソールを使用したデータイベントのログ記録」および「AWS Command Line Interfaceを使用したデータイベントのログ記録」を参照してください。

次の表に、データイベントを記録できる AWS Cloud Map リソースタイプを示します。データイベントタイプ (コンソール) 列には、CloudTrail コンソールの[データイベントタイプ]リストから選択する値が表示されます。resources.type 値列には、 AWS CLI または CloudTrail APIs を使用して高度なイベントセレクタを設定するときに指定する `resources.type`値が表示されます。CloudTrail に記録されたデータ API 列には、リソース タイプの CloudTrail にログ記録された API コールが表示されます。

データイベントタイプ (コンソール)	resources.type 値	CloudTrail にログ記録されたデータ API
AwsApiCall	`AWS::ServiceDiscovery::Namespace`	• DiscoverInstances • DiscoverInstancesRevision
AwsApiCall	`AWS::ServiceDiscovery::Service`	• DiscoverInstances • DiscoverInstancesRevision

データイベントタイプ (コンソール)	resources.type 値	CloudTrail にログ記録されたデータ API
		• GetServiceAttributes

eventName、readOnly、および resources.ARN フィールドでフィルタリングして、自分にとって重要なイベントのみをログに記録するように高度なイベントセレクタを設定できます。オブジェクトの詳細については、「AWS CloudTrail API リファレンス」の「AdvancedFieldSelector」を参照してください。

次の例は、すべての AWS Cloud Map データイベントをログに記録するように高度なイベントセレクタを設定する方法を示しています。

```
"AdvancedEventSelectors":
[
    {
        "Name": "Log all AWS Cloud Map data events",
        "FieldSelectors": [
            { "Field": "eventCategory", "Equals": ["Data"] },
            { "Field": "resources.type", "Equals":
 ["AWS::ServiceDiscovery::Namespace"] }
        ]
    }
]
```

AWS Cloud Map CloudTrail の管理イベント

管理イベントは、 のリソースで実行される管理オペレーションに関する情報を提供します AWS アカウント。これらのイベントは、コントロールプレーンオペレーションとも呼ばれます。CloudTrail は、デフォルトで管理イベントをログ記録します。

AWS Cloud Map は、すべての AWS Cloud Map コントロールプレーンオペレーションを管理イベントとしてログに記録します。CloudTrail に AWS Cloud Map ログ記録する AWS Cloud Map コントロールプレーンオペレーションのリストについては、 AWS Cloud Map API リファレンスを参照してください。

AWS Cloud Map イベントの例

各イベントは任意の送信元からの単一のリクエストを表し、リクエストされた API オペレーション、オペレーションの日時、リクエストパラメータなどに関する情報を含みます。CloudTrail ログ

ファイルは、パブリック API コールの順序付けられたスタックトレースではないため、イベントは特定の順序で表示されません。

次の例は、 CreateHTTPNamespaceオペレーションを示す CloudTrail 管理イベントを示しています。


```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:alejandro_rosalez",
        "arn": "arn:aws:sts::111122223333:assumed-role/users/alejandro_rosalez",
        "accountId": "111122223333",
        "accessKeyId": "AIDACKCEVSQ6C2EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROA123456789EXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/readonly-role",
                "accountId": "111122223333",
                "userName": "alejandro_rosalez"
            },
            "attributes": {
                "creationDate": "2024-03-19T16:15:37Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2024-03-19T19:23:13Z",
    "eventSource": "servicediscovery.amazonaws.com",
    "eventName": "CreateHttpNamespace",
    "awsRegion": "eu-west-3",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/122.0.0.0 Safari/537.36",
    "requestParameters": {
        "name": "example-namespace",
        "creatorRequestId": "eda8b524-ca14-4f68-a176-dc4dfd165c26",
        "tags": []
    },
    "responseElements": {
        "operationId": "7xm4i7ghhkaalma666nrg6itf2eylcbp-gwipo38o"
    },
```

```
    "requestID": "641274d0-dbbe-4e64-9b53-685769a086c7",
    "eventID": "4a1ab076-ef1b-4bcf-aa95-cec5fb64f2bd",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "servicediscovery.eu-west-3.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}
```

次の例は、 DiscoverInstancesオペレーションを示す CloudTrail データイベントを示しています。

```
{
            "eventVersion": "1.09",
            "userIdentity": {
                "type": "AssumedRole",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE:alejandro_rosalez",
                "arn": "arn:aws:sts::111122223333:assumed-role/role/Admin",
                "accountId": "111122223333",
                "accessKeyId": "AIDACKCEVSQ6C2EXAMPLE",
                "sessionContext": {
                    "sessionIssuer": {
                        "type": "Role",
                        "principalId": "AROA123456789EXAMPLE",
                        "arn": "arn:aws:iam::"111122223333":role/Admin",
                        "accountId": "111122223333",
                        "userName": "Admin"
                    },
                    "attributes": {
                        "creationDate": "2024-03-19T16:15:37Z",
                        "mfaAuthenticated": "false"
                    }
                }
            },
            "eventTime": "2024-03-19T21:19:12Z",
            "eventSource": "servicediscovery.amazonaws.com",
            "eventName": "DiscoverInstances",
```

```
            "awsRegion": "eu-west-3",
            "sourceIPAddress": "13.38.34.79",
            "userAgent": "Boto3/1.20.34 md/Botocore#1.34.60 ua/2.0 os/linux#6.5.0-1014-
aws md/arch#x86_64 lang/python#3.10.12 md/pyimpl#CPython cfg/retry-mode#legacy
 Botocore/1.34.60",
            "requestParameters": {
                "namespaceName": "example-namespace",
                "serviceName": "example-service",
                "queryParameters": {"example-key": "example-value"}
            },
            "responseElements": null,
            "requestID": "e5ee36f1-edb0-4814-a4ba-2e8c97621c79",
            "eventID": "503cedb6-9906-4ee5-83e0-a64dde27bab0",
            "readOnly": true,
            "resources": [
                {
                    "accountId": "111122223333",
                    "type": "AWS::ServiceDiscovery::Namespace",
                    "ARN": "arn:aws:servicediscovery:eu-west-3:111122223333:namespace/
ns-vh4nbmhEXAMPLE"
                },
                {
                    "accountId": "111122223333",
                    "type": "AWS::ServiceDiscovery::Service",
                    "ARN": "arn:aws:servicediscovery:eu-west-3:111122223333:service/
srv-h46op6ylEXAMPLE"
                }
            ],
            "eventType": "AwsApiCall",
            "managementEvent": false,
            "recipientAccountId": "111122223333",
            "eventCategory": "Data",
            "tlsDetails": {
                "tlsVersion": "TLSv1.3",
                "cipherSuite": "TLS_AES_128_GCM_SHA256",
                "clientProvidedHostHeader": "data-servicediscovery.eu-
west-3.amazonaws.com"
            },
            "sessionCredentialFromConsole": "true"
        }
```


CloudTrail レコードの内容については、「AWS CloudTrail ユーザーガイド」の「CloudTrail record contents」を参照してください。

AWS Cloud Map リソースのタグ付け

タグは、 AWS リソースに割り当てるラベルです。タグはそれぞれ、1 つのキーとオプションの 1 つの値で構成されており、どちらもお客様側が定義します。

タグを使用すると、目的、所有者、環境などで AWS リソースを分類できます。同じ型のリソースが多い場合に、割り当てたタグに基づいて特定のリソースをすばやく識別できます。たとえば、 AWS Cloud Map サービスのタグのセットを定義して、各サービスの所有者とスタックレベルを追跡できます。リソースタイプごとに一貫した一連のタグキーを考案することをお勧めします。

タグは自動的にはリソースに割り当てられません。タグを追加したら、いつでもタグキーと値は編集でき、タグはリソースからいつでも削除できます。リソースを削除すると、リソースのタグも削除されます。

タグには に対する意味論的な意味はなく AWS Cloud Map 、厳密に文字列として解釈されます。タグの値を空の文字列に設定することはできますが、タグの値を null に設定することはできません。特定のリソースについて既存のタグと同じキーを持つタグを追加した場合、以前の値は新しい値によって上書きされます。

タグは、 AWS マネジメントコンソール、、 AWS CLIおよび AWS Cloud Map API を使用して操作できます。

AWS Identity and Access Management (IAM) を使用している場合は、 AWS アカウントのどのユーザーにタグを作成、編集、または削除するためのアクセス許可があるかを制御できます。

リソースのタグ付け方法

新規または既存の AWS Cloud Map 名前空間とサービスにタグを付けることができます。

AWS Cloud Map コンソールを使用している場合は、関連するリソースページのタグタブを使用して、新しいリソースの作成時または既存のリソースにタグをいつでも適用できます。

AWS Cloud Map API、、または AWS SDK を使用している場合は、関連する API アクションの `tags`パラメータを使用して新しいリソースにタグを適用するか AWS CLI、TagResource API アクションを使用して既存のリソースにタグを適用できます。詳細については、「TagResource」を参照してください。

リソース作成アクションによっては、リソースの作成時にリソースのタグを指定できます。リソースの作成時にタグを適用できない場合、リソースの作成プロセスは失敗します。これにより、作成時にタグ付けするリソースが、指定したタグで作成されるか、まったく作成されないことが確認されま

す。作成時にリソースにタグを付ける場合、リソースの作成後にカスタムのタグ付けスクリプトを実行する必要はありません。

次の表は、タグ付けできる AWS Cloud Map リソースと、作成時にタグ付けできるリソースを示しています。

AWS Cloud Map リソースのタグ付けのサポート

リソース	タグをサポート	タグの伝播をサポート	作成時のタグ付けをサポート (AWS Cloud Map API、 AWS CLI、 AWS SDK)
AWS Cloud Map 名前空間	はい	いいえ。名前空間タグは、名前空間に関連付けられた他のリソースには伝達されません。	はい
AWS Cloud Map サービス	はい	いいえ。サービスタグは、サービスに関連付けられた他のリソースには伝達されません。	はい

制限事項

タグには以下のベーシックな制限があります。

- それぞれのリソースに付けることができるタグの最大数は 50 です。
- タグキーはリソースごとにそれぞれ一意である必要があります。また、各タグキーに設定できる値は 1 つのみです。
- キーの最大長 - UTF-8 の 128 Unicode 文字
- 値の最大長 - UTF-8 の 256 Unicode 文字
- タグ付けスキーマが複数の AWS サービスとリソースで使用されている場合は、他のサービスで許可される文字に制限がある可能性があることに注意してください。一般的に使用が許可される文字は、UTF-8 で表現できる文字、数字、スペース、および +、-、=、.、_、:、/、@。

- タグのキーと値では、大文字と小文字が区別されます。
- aws:、AWS:、またはキーや値のプレフィックスなど、 の大文字または小文字の組み合わせは使用しないでください。使用のために予約されています AWS 。このプレフィックスを持つタグのキーや値を編集または削除することはできません。このプレフィックスを持つタグは、リソースあたりのタグ数の制限にはカウントされません。

AWS Cloud Map リソースのタグの更新

次の AWS CLI コマンドまたは AWS Cloud Map API オペレーションを使用して、リソースのタグを追加、更新、一覧表示、削除します。

AWS Cloud Map リソースのタグ付けのサポート

タスク	API アクション	AWS CLI	AWS Tools for Windows PowerShell
1 つ以上のタグを追加、または上書きします。	TagResource	タグリソース	Add-SDResourceTag
1 つ以上のタグを削除します。	UntagResource	タグなしリソース	Remove-SDResourceTag
リソースのタグの一覧表示	ListTagsForResource	list-tags-for-resource	Get-SDResourceTag

以下の例では、 AWS CLIを使用して、リソースに対してタグ付けまたはタグ削除する方法を示しています。

例 1: 既存のリソースにタグ付けする

次のコマンドでは、既存のリソースにタグ付けします。

```
aws servicediscovery tag-resource --resource-arn resource_ARN --tags team=devs
```

例 2: 既存のリソースからタグを削除する

次のコマンドでは、既存のリソースからタグを削除します。

```
aws servicediscovery untag-resource --resource-arn resource_ARN --tag-keys tag_key
```

例 3: リソースのタグのリスト取得

次のコマンドは、既存のリソースに関連付けられているタグのリストを取得します。

```
aws servicediscovery list-tags-for-resource --resource-arn resource_ARN
```

一部のリソース作成アクションでは、リソースの作成時にタグを指定できます。以下のアクションでは、作成時のタグ付けがサポートされます。

タスク	API アクション	AWS CLI	AWS Tools for Windows PowerShell
HTTP 名前空間を作成する	CreateHttpNamespace	create-http-namespace	New-SDHttpNamespace
DNS に基づくプライベート名前空間を作成する	CreatePrivateDnsNamespace	create-private-dns-namespace	New-SDPrivateDnsNamespace
DNS に基づくパブリック名前空間を作成する	CreatePublicDnsNamespace	create-public-dns-namespace	New-SDPublicDnsNamespace
サービスを作成する	CreateService	サービスの作成	New-SDService

AWS Cloud Map サービスクォータ

AWS Cloud Map リソースには、次のアカウントレベルのサービスクォータが適用されます。リストされている各クォータは、 AWS Cloud Map リソースを作成する各 AWS リージョンに適用されます。

名前	デフォルト	引き上げ可能	説明
インスタンスあたりのカスタム属性	サポートされている各リージョン: 30	いいえ	インスタンス登録時に指定できるカスタム属性の最大数。
アカウントあたりの DiscoverInstances バーストレート	サポートされている各リージョン: 2,000	あり	単一のアカウントから DiscoverInstances オペレーションを呼び出す最大バーストレート。
アカウントあたりの DiscoverInstances オペレーション定常率	サポートされている各リージョン: 1,000	あり	単一のアカウントから DiscoverInstances オペレーションを呼び出す最大定常率。
アカウントレートごとの DiscoverInstancesRevision オペレーション	サポートされている各リージョン: 3,000	あり	単一のアカウントから DiscoverInstancesRevision オペレーションを呼び出す最大レート。
名前空間あたりのインスタンス	サポートされている各リージョン: 2,000	あり	同じ名前空間を使用して登録できるサービスインスタンスの最大数。

名前	デフォルト	引き上げ可能	説明
サービスあたりのインスタンス	サポートされている各リージョン: 1,000	いいえ	同じサービスを使用してリージョンに登録できるインスタンスの最大数。
リージョンあたりの名前空間	サポートされている各リージョン: 50	可能	リージョンごとに作成できる名前空間の最大数。

* 名前空間を作成すると、Amazon Route 53 ホストゾーンが自動的に作成されます。このホストゾーンは、 AWS アカウントで作成できるホストゾーン数のクォータに対してカウントされます。詳細については、Amazon Route 53 デベロッパーガイドのホストゾーンのクォータを参照してください。

** AWS Cloud Map 用のDNS 名前空間あたりのインスタンスを増やすには、Route 53 ホストゾーンあたりのレコード数の制限を増やす必要があり、これには追加料金が発生します。

AWS Cloud Map サービスクォータの管理

AWS Cloud Map は Service Quotas と統合されています。Service Quotas は、クォータを一元的に表示および管理できる AWS サービスです。詳細については、「Service Quotas ユーザーガイド」の「Service Quotas とは」を参照してください。

Service Quotas を使用すると、 AWS Cloud Map サービスクォータの値を簡単に検索できます。

AWS マネジメントコンソール

を使用して AWS Cloud Map サービスクォータを表示するには AWS マネジメントコンソール

1. https://console.aws.amazon.com/servicequotas/ で Service Quotas コンソールを開きます。
2. ナビゲーションペインで、[AWS サービス] を選択します。
3. [AWS サービス] リストから、[AWS Cloud Map]]] を探して選択します。

4. のサービスクォータリストには AWS Cloud Map、サービスクォータ名、適用された値 (使用可能な場合) 、 AWS デフォルトのクォータ、クォータ値が調整可能かどうかが表示されます。

   説明など、サービスクォータに関する追加情報を表示するには、クォータ名を選択してクォータの詳細を表示します。

5. (オプション) クォータの引き上げをリクエストするには、引き上げるクォータを選択し、アカウントレベルで引き上げをリクエストを選択します。

を使用してサービスクォータをさらに操作するには、Service Quotas ユーザーガイド AWS マネジメントコンソール 」を参照してください。

AWS CLI

を使用して AWS Cloud Map サービスクォータを表示するには AWS CLI

次のコマンドを実行して、デフォルトの AWS Cloud Map クォータを表示します。

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code AWSCloudMap \
    --output table
```

次のコマンドを実行して、適用された AWS Cloud Map クォータを表示します。

```
aws service-quotas list-service-quotas \
    --service-code AWSCloudMap
```

を使用したサービスクォータの操作の詳細については AWS CLI、Service Quotas AWS CLI コマンドリファレンス」を参照してください。クォータの引き上げをリクエストするには、AWS CLI Command Reference の `request-service-quota-increase` コマンドを参照してください。

Handle AWS Cloud Map DiscoverInstances API リクエストスロットリング

AWS Cloud Map は、リージョンごとに各 AWS アカウントの DiscoverInstances API リクエストを調整します。スロットリングは、サービスのパフォーマンスを向上させ、すべての AWS Cloud

Map お客様に公平な使用を提供するのに役立ちます。スロットリングにより、 AWS Cloud Map DiscoverInstances API への呼び出しが DiscoverInstances API リクエストの最大許容クォータを超えないようになります。次のソースのいずれかから発信される DiscoverInstances API コールは、リクエストクォータの対象となります。

- サードパーティーのアプリケーション
- コマンドラインツール
- AWS Cloud Map コンソール

API スロットリングクォータを超えると、`RequestLimitExceeded` エラーコードが返されます。詳細については、「the section called “リクエストレート制限”」を参照してください。

スロットリングの適用方法

AWS Cloud Map はトークンバケットアルゴリズムを使用して API スロットリングを実装します。このアルゴリズムでは、アカウントには、特定の数のトークンを保持するバケットがあります。バケット内のトークンの数は、特定の秒におけるスロットリングクォータを表します。単一のリージョンに対して 1 つのバケットがあり、リージョン内のすべてのエンドポイントに適用されます。

リクエストレート制限

スロットリングにより、実行できる DiscoverInstances API リクエストの数が制限されます。各リクエストは、バケットから 1 つのトークンを削除します。例えば、DiscoverInstances API オペレーションのバケットサイズは 2,000 トークンでありるので、1 秒間に最大 2,000 の DiscoverInstances リクエストを作成できます。1 秒で 2,000 リクエストを超えると、スロットルされ、その秒以内の残りのリクエストは失敗します。

バケットは設定されたレートで自動的に補充されます。バケットが容量に達していない場合、バケットが容量に達するまで、設定された数のトークンが毎秒追加されます。リフィルトークンが到着したときにバケットが容量に達している場合、これらのトークンは破棄されます。DiscoverInstances API オペレーションのバケットサイズは 2,000 トークンで、リフィルレートは毎秒 1,000 トークンです。1 秒間に 2,000 の DiscoverInstances API リクエストを行うと、バケットはすぐにゼロ (0) トークンに削減されます。バケットは、最大容量の 2000 トークンに達するまで、毎秒最大 1,000 トークンで補充されます。

トークンはバケットに追加されたときに使用できます。API リクエストを行う前に、バケットが最大容量になるのを待つ必要はありません。1 秒間に 2,000 の DiscoverInstances API リクエストを作成してバケットを使い果たした場合でも、必要な限り、その後は毎秒最大 1,000 の DiscoverInstances

API リクエストを作成できます。つまり、バケットに追加されたリフィルトークンをすぐに使用できます。バケットは、リフィルレートよりも毎秒少ない API リクエスト数を作成する場合にのみ、最大容量への補充を開始します。

再試行またはバッチ処理

API リクエストが失敗した場合、アプリケーションでリクエストを再試行する必要がある場合があります。API リクエストの数を下げるには、連続するリクエストの間に適切なスリープ間隔を使用します。最良の結果を得るには、漸増または可変スリープ間隔を使用します。

スリープ間隔の計算

API リクエストをポーリングまたは再試行する必要がある場合は、エクスポネンシャルバックオフアルゴリズムを使用して API コール間のスリープ間隔を計算することをお勧めします。連続したエラーレスポンスの再試行間隔の待機時間を徐々に長くすることで、失敗リクエストの数を減らすことができます。このアルゴリズムの詳細と実装例については、 SDK およびツールリファレンスガイドの「再試行動作」を参照してください。 AWS SDKs

API スロットリングのクォータの調整

AWS アカウントの API スロットリングクォータの引き上げをリクエストできます。クォータの調整をリクエストするには、AWS サポート センターまでお問い合わせください。

のドキュメント履歴 AWS Cloud Map

次の表は、「AWS Cloud Map デベロッパーガイド」の主な更新や新機能の一覧です。また、お客様からいただいたフィードバックに対応するために、ドキュメントを頻繁に更新しています。

変更	説明	日付
AWS Cloud Map クロスアカウント名前空間共有	で AWS Resource Access Manager （ AWS RAM) AWS Organizations を使用して名前空間を他の AWS アカウントまたは組織内で共有し、クロスアカウントサービスの検出とレジストリを簡素化できるようになりました。	2025 年 8 月 14 日
AWS Cloud Map サービス属性	サービスレベルに属性を指定して、サービスに登録されているインスタンス間で属性が重複しないようにできるようになりました。これらの属性は、複雑なトラフィックルーティング、タイムアウト値と再試行値の設定、サービスと外部統合の調整に使用できます。	2024 年 12 月 13 日
チュートリアルの追加	の使用に関する一般的なユースケースを示す 2 つのチュートリアル AWS Cloud Map を追加しました。	2024 年 3 月 27 日
CloudTrail 統合ドキュメントが更新されました	API アクティビティを記録するための CloudTrail と AWS Cloud Map の統合について説明するドキュメントが更新されました。	2024 年 3 月 20 日

マネージドポリシーの更新	AWSCloudMapDiscoverInstanceAccess 、AWSCloudMapRegisterInstanceAccess 、AWSCloudMapReadOnlyAccess ポリシーが更新されました。	2023 年 9 月 20 日
クラウドマップと AWS PrivateLink	を使用して、VPC と の間にプライベート接続 AWS PrivateLink を作成できるようになりました AWS Cloud Map。	2023 年 9 月 15 日
マネージドポリシーの更新	AWSCloudMapDiscoverInstanceAccess ポリシーが更新されました。	2023 年 8 月 15 日
AWS SDK for Python	Python コマンドラインの例を追加しました。	2022 年 9 月 13 日
IPv6 サポート	API エンドポイントが IPv6 専用のネットワークで使用可能になりました。	2022 年 1 月 28 日
サービスインスタンスの検出	AWS Cloud Map は、DiscoverInstances API オペレーションを使用してのみ検出でき、DNS クエリを使用しない DNS クエリをサポートする名前空間でサービスを作成するためのサポートを追加しました。	2021 年 3 月 24 日

リソースへのタグ付け	AWS Cloud Map は、 を使用して名前空間とサービスにメタデータタグを追加するためのサポートを追加しました AWS マネジメントコンソール。	2021 年 2 月 8 日
リソースへのタグ付け	AWS Cloud Map は、 AWS CLI および APIs。	2020年6月22日
初回リリース	これは AWS Cloud Map デベロッパーガイドの初回リリースです。	2018 年 11 月 28 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。