aws

リファレンスガイド

AWS SDKsとツール

AWS SDKsとツール: リファレンスガイド

Table of Contents

AWS SDKs」で説明されている内容

多くの SDK とツールは、共有設計仕様または共有ライブラリを通じて、いくつかの共通機能を共有しています。

このガイドには以下に関する情報が含まれています。

- AWS SDKsとツールをグローバルに設定する – 共有 `config`および `credentials` ファイルまたは環境変数を使用して AWS SDKsとツールを設定する方法。
- AWS SDKs とツールを使用した認証とアクセス – を使用して開発 AWS するときに、コードまたはツールが で認証する方法を確立します AWS のサービス。
- AWS SDKsとツールの設定リファレンス – 認証と設定に使用できるすべての標準設定のリファレンス。
- AWS 共通ランタイム (CRT) ライブラリ – ほぼすべての SDKs で使用できる共有 AWS 共通ランタイム (CRT) ライブラリの概要。
- AWS SDKsメンテナンスポリシー では、Mobile and Internet of Things (IoT) SDKs やその基盤となる依存関係など、 AWS Software Development Kit (SDKs とツールのメンテナンスポリシーとバージョニングについて説明します。

この AWS SDKsおよびツールリファレンスガイドは、複数の SDKsおよびツールに適用される情報の基礎となることを目的としています。ここに記載されている情報に加えて、使用している SDK またはツール固有のガイドも使用してください。このガイドでの資料の関連セクションを含む SDK とツールは次のとおりです。

次を使用している場合:	このガイドの関連セクションは、以下のとおりです。
• 任意の SDK またはツール	AWS SDKsメンテナンスポリシー
• AWS Cloud9 • AWS CDK • AWS Toolkit for Azure DevOps • AWS Toolkit for JetBrains • AWS Toolkit for Visual Studio	AWS SDKsとツールをグローバルに設定する AWS SDKs とツールを使用した認証とアクセス AWS SDKsメンテナンスポリシー

次を使用している場合:	このガイドの関連セクションは、以下のとおりです。
• AWS Toolkit for Visual Studio Code • AWS Serverless Application Model • AWS CodeArtifact • AWS CodeBuild • Amazon CodeCatalyst • AWS CodeCommit • AWS CodeDeploy • AWS CodePipeline	
• AWS CLI • AWS SDK for C++ • AWS SDK for Go • AWS SDK for Java • AWS SDK for JavaScript • AWS SDK for Kotlin • AWS SDK for .NET • AWS SDK for PHP • AWS SDK for Python (Boto3) • AWS SDK for Ruby • AWS SDK for Rust • AWS SDK for Swift • AWS Tools for Windows PowerShell	AWS SDKsとツールをグローバルに設定する AWS SDKs とツールを使用した認証とアクセス AWS SDKsとツールの設定リファレンス AWS 共通ランタイム (CRT) ライブラリ AWS SDKsメンテナンスポリシー AWS SDKsとツールのバージョンライフサイクル

- でアプリケーションを開発するのに役立つツールの概要については AWS、「構築するツール AWS」を参照してください。
- サポートに関する情報は、「AWS ナレッジセンター」 を参照してください。
- AWS 用語については、 AWS の用語集 リファレンスのAWS 用語集を参照してください。

デベロッパーリソース

Amazon Q Developer は、生成 AI を活用した会話型アシスタントで、 AWS アプリケーションの理解、構築、拡張、運用に役立ちます。構築を加速するために AWS、Amazon Q を強化するモデルは、より完全で実用的な参照された回答を生成するために、高品質の AWS コンテンツで強化されています。詳細については、「Amazon Q Developer ユーザーガイド」の「What is Amazon Q Developer?」を参照してください。

ツールキットのテレメトリー通知

AWS 統合開発環境 (IDE) ツールキットは、IDE 内の AWS サービスへのアクセスを可能にするプラグインと拡張機能です。Amazon Q IDE プラグインと拡張機能により、IDE で生成 AI 支援が有効になります。各 IDE ツールキットの詳細については、前述のテーブルのツールキットユーザーガイドを参照してください。IDE での Amazon Q の使用の詳細については、「Amazon Q Developer ガイド」の「Using Amazon Q in the IDE」のトピックを参照してください。

AWS IDE Toolkits と Amazon Q は、今後の AWS Toolkit と Amazon Q のリリースに関する決定を通知するために、クライアント側のテレメトリデータを収集して保存することがあります。収集されたデータは、 AWS Toolkit と Amazon Q の使用を定量化します。

すべての IDE Toolkits と Amazon Q AWS で収集されたテレメトリデータの詳細については、`aws-toolkit-common`Github リポジトリの commonDefinitions.json ドキュメントを参照してください。

各 AWS IDE Toolkit と Amazon Q 拡張機能によって収集されたテレメトリデータの詳細については、次の AWS Toolkit GitHub リポジトリのリソースドキュメントを参照してください。

- AWS Amazon Q を使用した Visual Studio Toolkit
- AWS Toolkit for Visual Studio Code VS Code の および Amazon Q 拡張機能
- AWS Toolkit for JetBrains JetBrains 用 および Amazon Q プラグイン
- Amazon Q for Eclipse

AWS Toolkits でアクセスできる特定の AWS サービスは、追加のクライアント側のテレメトリデータを収集する場合があります。個々の AWS サービスによって収集されるデータの種類の詳細については、関心のある特定のサービスのAWS ドキュメントトピックを参照してください。

AWS SDKsとツールをグローバルに設定する

AWS SDKs や AWS Command Line Interface (AWS CLI) などのその他のデ AWS ベロッパーツールを使用すると、 AWS サービス APIsを操作できます。ただし、その前に、要求された操作を実行するために必要な情報を SDK またはツールに設定する必要があります。

この情報には以下のアイテムが含まれます。

- API の呼び出し元を識別する認証情報。認証情報は、 AWS サーバーへのリクエストを暗号化するために使用されます。この情報を使用して、 はアイデンティティ AWS を確認し、それに関連付けられたアクセス許可ポリシーを取得できます。次に、どのようなアクションを実行できるかを判断できます。
- リクエストの処理方法、リクエストの送信先 (AWS サービスエンドポイント) 、レスポンスの解釈または表示方法を AWS CLI または SDK に指示するために使用するその他の設定の詳細。

各 SDK またはツールは、必要な認証情報と設定情報を供給するために使用できる複数のソースをサポートしています。ソースの中には SDK やツールに独自のものもあるため、その方法の使用方法の詳細については、そのツールまたは SDK のドキュメントを参照する必要があります。

ただし、 AWS SDKsとツールは、コード自体以外のプライマリソースからの一般的な設定をサポートしています。このセクションでは、次のトピックについて説明します。

トピック

共有ファイル**config**と **credentials** ファイルを使用して AWS SDKs とツールをグローバルに設定する

共有 AWS `config` ファイルと `credentials`ファイルは、 AWS SDK またはツールへの認証と設定を指定できる最も一般的な方法です。

共有 credentials および config ファイルには一連のプロファイルが含まれています。プロファイルは、 AWS SDKs、 AWS Command Line Interface (AWS CLI)、およびその他のツールで使用されるキーと値のペアの一連の設定です。プロファイルを使用するときに SDK / ツールの一部を設定するために、設定値がプロファイルに添付されます。これらのファイルは、値がユーザーのローカル環境にあるすべてのアプリケーション、プロセス、または SDK に影響するという点で「共有」されます。

共有 config ファイルと credentials ファイルはどちらも ASCII 文字 (UTF-8 でエンコードされた) のみを含むプレーンテキストファイルです。これらは一般に INI ファイルと呼ばれる形式をとります。

プロファイル

共有 config ファイルと credentials ファイル内の設定は特定のプロファイルに関連付けられます。ファイル内で複数のプロファイルを定義し、異なる設定を作成して異なる開発環境に適用することができます。

[default] プロファイルには、特定の名前付きプロファイルが指定されていない場合に SDK またはツールオペレーションで使用される値が含まれます。名前で明示的に参照できる個別のプロファイルを作成することもできます。各プロファイルは、アプリケーションやシナリオに応じて異なる設定と値を使用できます。

> **Note**
>
> [default] は単に名前のないプロファイルです。このプロファイルは、ユーザーがプロファイルを指定しない場合に SDK が使用するデフォルトのプロファイルであるため、default の名前が付けられています。継承されたデフォルト値を他のプロファイルに使用することはありません。[default] プロファイルに何かを設定し、それを名前付きプロファイルでは設定しなかった場合、名前付きプロファイルを使用してもその値は設定されません。

名前付きプロファイルの設定

[default] プロファイルと複数の名前付きプロファイルは、同じファイル内に存在できます。次の設定を使用して、コードの実行時に SDK またはツールがどのプロファイルの設定を使用するかを選択します。プロファイルはコード内で選択することも、 AWS CLIを使用する場合はコマンドごとに選択することもできます。

この機能を設定するには、以下のいずれかの設定を使用します。

`AWS_PROFILE` - 環境変数

この環境変数を名前付きプロファイルまたは「デフォルト」に設定すると、すべての SDK コードと AWS CLI コマンドはそのプロファイルの設定を使用します。

Linux/macOS のコマンドラインによる環境変数の設定の例を以下に示します。

```
export AWS_PROFILE="my_default_profile_name";
```

Windows のコマンドラインによる環境変数の設定の例を以下に示します。

```
setx AWS_PROFILE "my_default_profile_name"
```

`aws.profile` - JVM システムプロパティ

JVM の SDK for Kotlin と SDK for Java 2.x では、`aws.profile` システムプロパティを設定できます。SDK はサービスクライアントを作成するときに、コードで設定が上書きされない限り、名前付きプロファイルの設定を使用します。SDK for Java 1.x は、このシステムプロパティをサポートしていません。

> **Note**
>
> アプリケーションが複数のアプリケーションを実行しているサーバー上にある場合は、デフォルトのプロファイルではなく、常に名前付きプロファイルを使用することをお勧めします。デフォルトのプロファイルは、環境内の任意の AWS アプリケーションによって自動的に取得され、それらの間で共有されます。したがって、他のユーザーがアプリケーションのデフォルトプロファイルを更新すると、意図せず他のユーザーに影響を与える可能性があります。これを防ぐには、共有 `config` ファイルで名前付きプロファイルを定義し、コードに名前付きプロファイルを設定して、アプリケーションでその名前付きプロファイルを使用します。名前付きプロファイルのスコープがアプリケーションにのみ影響することがわかっている場合は、環境変数または JVM システムプロパティを使用して名前付きプロファイルを設定できます。

設定ファイルの形式

config ファイルは、セクションにまとめられています。セクションは、設定の名前付きコレクションであり、別のセクション定義の行が検出されるまで続きます。

config ファイルは、次の形式を使用するプレーンテキストファイルです。

- セクション内のすべてのエントリは、setting-name=value の一般的な形式になります。
- 行の先頭にハッシュタグ (#) を付けると、行をコメントアウトできます。

セクションタイプ

セクション定義は、設定のコレクションに名前を付ける行です。セクション定義行の先頭と末尾は角括弧 ([ ]) です。括弧内には、セクションタイプ識別子とセクションのカスタム名があります。英文字、数字、ハイフン (-)、アンダースコア (_) は使用できますが、スペースは使用できません。

セクションタイプ: **default**

セクション定義行の例：[default]

[default] は、profile セクション識別子を必要としない唯一のプロファイルです。

[default] プロファイルのある基本 config ファイルの例を以下に示します。region を設定します。別のセクション定義が見つかるまで、この行に続くすべての設定は、このプロファイルの一部になります。

```
[default]
#Full line comment, this text is ignored.
region = us-east-2
```

セクションタイプ: **profile**

セクション定義行の例：[profile *dev*]

profile セクション定義行は、さまざまな開発シナリオに適用できる名前付き設定グループです。名前付きプロファイルについての理解を深めるには、前のセクションの「プロファイル」を参照してください。

次の例は、profile セクション定義行と foo という名前付きプロファイルを含む config ファイルを示しています。別のセクション定義が見つかるまで、この行に続くすべての設定は、この名前付きプロファイルの一部になります。

```
[profile foo]
...settings...
```

次の例の s3 設定やサブ設定など、一部の設定には独自のサブ設定グループがネストされています。サブ設定を 1 つまたは複数のスペースでインデントしてグループに関連付けます。

```
[profile test]
region = us-west-2
s3 =
    max_concurrent_requests=10
    max_queue_size=1000
```

セクションタイプ: **sso-session**

セクション定義行の例：[sso-session *my-sso*]

sso-session セクション定義行には、 を使用して AWS 認証情報を解決するようにプロファイルを設定するために使用される設定のグループの名前が付けられます AWS IAM アイデンティティセンター。シングルサインオン認証の設定の詳細については、「IAM Identity Center を使用して AWS SDK とツールを認証する」を参照してください。プロファイルは、キーと値のペアによって sso-session セクションにリンクされます。ここで、sso-session はキー、sso-session セクションの名前は値です (sso-session = <name-of-sso-session-section> など) 。

次の例では、「my-sso」のトークンを使用して「111122223333」アカウントの「SampleRole」IAM ロールの短期 AWS 認証情報を取得するプロファイルを設定しています。「my-sso」sso-session セクションは、profile セクションの中で sso-session キーを使用して名前で参照されます。

```
[profile dev]
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole

[sso-session my-sso]
sso_region = us-east-1
sso_start_url = https://my-sso-portal.awsapps.com/start
```

セクションタイプ: **services**

セクション定義行の例：[services *dev*]

> **Note**
>
> services セクションはサービス固有のエンドポイントのカスタマイズをサポートしており、この機能を含む SDK とツールでのみ使用できます。お使いの SDK でこの機能が使用できるかどうかを確認するには、「サービス固有のエンドポイント」の AWS SDKsとツールによるサポート を参照してください。

services セクション定義行は、 AWS のサービス リクエストのカスタムエンドポイントを設定する設定のグループに名前を付けます。プロファイルは、キーと値のペアによって services セクションにリンクされます。ここで、services はキー、services セクションの名前は値です (services = <name-of-services-section> など) 。

services セクションはさらに<SERVICE> = 、行ごとにサブセクションに分割されます。ここで、 <SERVICE>は AWS のサービス 識別子キーです。 AWS のサービス 識別子は、すべてのスペースをアンダースコアに置き換え、すべての文字を小文字に置き換えserviceIdることで、API モデルの に基づいています。services セクションで使用するすべてのサービス識別子キーのリストについては、「サービス固有のエンドポイントの識別子」を参照してください。サービス識別子キーの後には、ネストされた設定 (それぞれが 1 行にあり、2 つのスペースでインデントされている) が続きます。

次の例では、services 定義を使用して、 Amazon DynamoDB サービスに対して行われたリクエストにのみ使用するようにエンドポイントを設定しています。"local-dynamodb" services セクションは、profile セクションの中で services キーを使用して名前で参照されます。 AWS のサービス 識別子キーは ですdynamodb。 Amazon DynamoDB サービスサブセクションは行 で始まりますdynamodb = 。直後のインデントされた行はすべてそのサブセクションに含まれ、そのサービスに適用されます。

```
[profile dev]
services = local-dynamodb

[services local-dynamodb]
dynamodb =
  endpoint_url = http://localhost:8000
```

カスタムエンドポイントの設定の詳細については、「サービス固有のエンドポイント」を参照してください。

認証情報ファイルの形式

プロファイルセクションが単語 profile で始まらないことを除けば、credentials ファイルのルールは一般的に config ファイルのルールと同じです。角括弧内にはプロファイル名だけを使用してください。次の例は、foo という名前付きプロファイルセクションを含む credentials ファイルを示しています。

```
[foo]
...credential settings...
```

credentials ファイルに保存できるのは、「シークレット」または機密と見なされる次の設定のみです。aws_access_key_id、aws_secret_access_key、および aws_session_token。これらの設定は共有 config ファイルに配置することもできますが、これらの機密の値は別の credentials ファイルに保持することをお勧めします。これにより、必要に応じてファイルごとに個別のアクセス許可を与えることができます。

[default] プロファイルのある基本 credentials ファイルの例を以下に示します。aws_access_key_id、aws_secret_access_key、および aws_session_token グローバル設定を設定します。

```
[default]
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
aws_session_token=IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZVERYLONGSTR1
```

credentials ファイルで名前付きプロファイルを使用するか「default」を使用するかにかかわらず、ここでの設定は、同じプロファイル名を使用する config ファイルの設定と組み合わせられます。同じ名前を共有するプロファイルの両方のファイルに認証情報がある場合、認証情報ファイルのキーが優先されます。

AWS SDKs とツールの共有configファイルとcredentialsファイルの場所の検索と変更

共有 AWS config ファイルと credentialsファイルは、 AWS SDKsとツールの設定情報を保持するプレーンテキストファイルです。ファイルは環境内にローカルに存在し、SDK コードまたはその環境で実行する AWS CLI コマンドによって自動的に使用されます。例えば、独自のコンピュータや Amazon Elastic Compute Cloud インスタンスで開発する場合などです。

SDK またはツールは実行時にこれらのファイルをチェックし、使用可能な設定をロードします。ファイルがまだ存在しない場合は、SDK またはツールによって基本ファイルが自動的に作成されます。

デフォルトでは、ファイルは home またはユーザーフォルダに配置された .aws という名前のフォルダにあります。

オペレーティングシステム	ファイルのデフォルトの場所と名前
Linux および macOS	~/.aws/config ~/.aws/credentials
Server	%USERPROFILE%\.aws\config %USERPROFILE%\.aws\credentials

ホームディレクトリ解決

~ は、次の場合にホームディレクトリの解決にのみ使用されます。

- パスの開始
- 直後に / またはプラットフォーム固有の区切り文字が続く場合。Windows では、~/ と ~\ の両方がホームディレクトリに解決されます。

ホームディレクトリを決定するときに、次の変数がチェックされます。

- （全プラットフォーム）HOME 環境変数
- （Windows プラットフォーム）USERPROFILE 環境変数
- (Windows プラットフォーム) HOMEDRIVE と HOMEPATH 環境変数の連結 ($HOMEDRIVE $HOMEPATH)
- （SDK またはツールごとのオプション）SDK またはツール固有のホームパス解決関数または変数

可能な場合は、パスの先頭にユーザーのホームディレクトリが指定されている場合（例：~username/）、要求されたユーザー名のホームディレクトリ（例：/home/username/.aws/config）に解決されます。

これらのファイルのデフォルトロケーションの変更

次のいずれかを使用して、これらのファイルが SDK またはツールによってどこからロードされるかを上書きできます。

環境変数を使用します。

以下の環境変数を設定して、これらのファイルの場所または名前をデフォルト値からカスタム値に変更できます。

- config ファイルの環境変数：**AWS_CONFIG_FILE**
- credentials ファイルの環境変数：**AWS_SHARED_CREDENTIALS_FILE**

Linux/macOS

Linux または macOS で次の[エクスポート]コマンドを実行して、別の場所を指定できます。

```
$ export AWS_CONFIG_FILE=/some/file/path/on/the/system/config-file-name
$ export AWS_SHARED_CREDENTIALS_FILE=/some/other/file/path/on/the/system/
credentials-file-name
```

Windows

Windows で次の [setx] コマンドを実行して、別の場所を指定できます。

```
C:\> setx AWS_CONFIG_FILE c:\some\file\path\on\the\system\config-file-name
C:\> setx AWS_SHARED_CREDENTIALS_FILE c:\some\other\file\path\on\the\system
\credentials-file-name
```

環境変数を使用したシステムの設定の詳細については、「環境変数を使用して AWS SDKsとツールをグローバルに設定する」を参照してください。

JVM システムプロパティの使用

JVM で実行されている SDK for Kotlin および SDK for Java 2.x では、次の JVM システムプロパティを設定して、これらのファイルの場所または名前をデフォルトからカスタム値に変更できます。

- config ファイルの JVM システムプロパティ: **aws.configFile**
- credentials ファイルの環境変数：**aws.sharedCredentialsFile**

JVM システムプロパティを設定する方法については、「the section called “JVM システムプロパティを設定する方法”」を参照してください。SDK for Java 1.x は、これらのシステムプロパティをサポートしていません。

環境変数を使用して AWS SDKsとツールをグローバルに設定する

環境変数は、 AWS SDKs とツールを使用するときに設定オプションと認証情報を指定する別の方法を提供します。環境変数はスクリプト処理や、名前付きプロファイルを一時的にデフォルトとして設定する場合に活用できます。ほとんどの SDK でサポートされている環境変数のリストについては、「環境変数の一覧」を参照してください。

オプションの優先順位

- 環境変数を使用して設定を指定すると、共有 AWS `config` ファイルと `credentials` ファイル内のプロファイルからロードされたすべての値が上書きされます。
- AWS CLI コマンドラインで パラメータを使用して設定を指定すると、対応する環境変数または設定ファイルのプロファイルのいずれかの値が上書きされます。

環境変数の設定方法

次の例では、デフォルトのユーザーの環境変数を設定する方法を示します。

Linux, macOS, or Unix

```
$ export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
$ export
 AWS_SESSION_TOKEN=AQoEXAMPLEH4aoAH0gNCAPy...truncated...zrkuWJOgQs8IZZaIv2BXIa2R4Olgk
$ export AWS_REGION=us-west-2
```

環境変数を設定すると、シェルセッションの終了時まで、または変数に別の値を設定するまで、使用する値が変更されます。変数をシェルのスタートアップスクリプトで設定することで、変数をこれからのセッションで永続的にすることができます。

Windows Command Prompt

```
C:\> setx AWS_ACCESS_KEY_ID AKIAIOSFODNN7EXAMPLE
C:\> setx AWS_SECRET_ACCESS_KEY wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

```
C:\> setx
 AWS_SESSION_TOKEN AQoEXAMPLEH4aoAH0gNCAPy...truncated...zrkuWJOgQs8IZZaIv2BXIa2R4Olgk
C:\> setx AWS_REGION us-west-2
```

set を使用して環境変数を設定すると、現在のコマンドプロンプトセッションの終了時まで、または変数を別の値に設定するまで、使用する値が変更されます。setx を使用して環境変数を設定すると、現在のコマンドプロンプトセッションおよびコマンド実行後に作成するすべてのコマンドプロンプトセッションで使用する値が変更されます。これは、コマンド実行時にすでに実行されている他のコマンドシェルには影響を及ぼしません。

PowerShell

```
PS C:\> $Env:AWS_ACCESS_KEY_ID="AKIAIOSFODNN7EXAMPLE"
PS C:\> $Env:AWS_SECRET_ACCESS_KEY="wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"
PS C:
\> $Env:AWS_SESSION_TOKEN="AQoEXAMPLEH4aoAH0gNCAPy...truncated...zrkuWJOgQs8IZZaIv2BXIa2R4Ol
PS C:\> $Env:AWS_REGION="us-west-2"
```

前の例に示すように PowerShell プロンプトで環境変数を設定した場合は、現在のセッションの期間だけ値が保存されます。PowerShell およびコマンドプロンプトセッション間で環境変数を永続的に設定するには、[コントロールパネル] の [システム] アプリケーションを使用して変数を保存します。または、変数を PowerShell プロファイルに追加すると、その変数を今後のすべての PowerShell セッションに設定できます。環境変数の保存やそれをセッション間で永続的に維持する詳細については、「PowerShell documentation」(PowerShell ドキュメント) を参照してください。

サーバーレス環境変数設定

開発にサーバーレスアーキテクチャを使用する場合、環境変数を設定するための他のオプションもあります。コンテナによっては、非クラウド環境と同様に、そのコンテナで実行されるコードに対して異なる方法を使用して環境変数を表示したりアクセスしたりすることができます。

たとえば、 を使用すると AWS Lambda、環境変数を直接設定できます。詳細については、「 AWS Lambda デベロッパーガイド」のAWS Lambda 「環境変数の使用」を参照してください。

サーバーレスフレームワークでは、環境設定の下のプロバイダーキーの下の `serverless.yml` ファイルに SDK 環境変数を設定できることがよくあります。この `serverless.yml` ファイルについて詳しくは、「サーバーレスフレームワーク」ドキュメントの 「一般関数設定」 を参照してください。

コンテナ環境変数の設定にどのメカニズムを使用するかにかかわらず、コンテナによって予約されているものもあります。たとえば、Lambdaの「定義済みランタイム環境変数」で説明されているものなどです。環境変数の処理方法や制限の有無については、使用しているコンテナの公式ドキュメントを必ず確認してください。

JVM システムプロパティを使用して AWS SDK for Java と をグローバルに設定する AWS SDK for Kotlin

JVM システムプロパティは、 や などの JVM で実行される SDKs の設定オプション AWS SDK for Java と認証情報を指定する別の方法を提供します AWS SDK for Kotlin。SDK がサポートする JVM システムプロパティのリストについては、「Settings reference」を参照してください。

オプションの優先順位

- JVM システムプロパティを使用して設定を指定した場合、環境変数で見つかった値や共有 AWS `config` および `credentials` ファイル内のプロファイルからロードされた値は上書きされます。
- 環境変数を使用して設定を指定した場合、共有 AWS `config` および `credentials` ファイル内のプロファイルからロードされた値は上書きされます。

JVM システムプロパティを設定する方法

さまざまな方法で JVM システムプロパティを設定できます。

コマンドラインでの設定

`-D` スイッチを使用して `java` コマンドを呼び出すときに、コマンドラインに JVM システムプロパティを設定します。次のコマンドは、コード内の値を明示的に上書きしない限り、すべてのサービスクライアントに対して AWS リージョン をグローバルに設定します。

```
java -Daws.region=us-east-1 -jar <your_application.jar> <other_arguments>
```

複数の JVM システムプロパティを設定する必要がある場合は、`-D` スイッチを複数回指定します。

環境変数の使用

アプリケーションを実行するために JVM を呼び出すコマンドラインにアクセスできない場合は、`JAVA_TOOL_OPTIONS` 環境変数を使用してコマンドラインオプションを設定できます。このア

プローチは、Java ランタイムで AWS Lambda 関数を実行したり、埋め込み JVM でコードを実行したりする状況で役立ちます。

次の例では、コード内の値を明示的に上書きしない限り、すべてのサービスクライアントに対してを AWS リージョン グローバルに設定します。

Linux, macOS, or Unix

```
$ export JAVA_TOOL_OPTIONS="-Daws.region=us-east-1"
```

環境変数を設定すると使用する値が変更され、その値はシェルセッションが終了するか、または変数に別の値が設定されるまで有効です。変数をシェルのスタートアップスクリプトで設定することで、変数をこれからのセッションで永続的にすることができます。

Windows Command Prompt

```
C:\> setx JAVA_TOOL_OPTIONS -Daws.region=us-east-1
```

set を使用して環境変数を設定すると、現在のコマンドプロンプトセッションの終了時まで、または変数を別の値に設定するまで、使用する値が変更されます。setx を使用して環境変数を設定すると、現在のコマンドプロンプトセッションおよびコマンド実行後に作成するすべてのコマンドプロンプトセッションで使用する値が変更されます。これは、コマンド実行時にすでに実行されている他のコマンドシェルには影響を及ぼしません。

実行時

次の例に示すように、System.setProperty メソッドを使用して実行時にコードで JVM システムプロパティを設定することもできます。

```
System.setProperty("aws.region", "us-east-1");
```

> **Important**
>
> SDK サービスクライアントを初期化する前に JVM システムプロパティを設定してください。そうしないと、サービスクライアントが他の値を使用することがあります。

AWS SDKs とツールを使用した認証とアクセス

AWS SDK アプリケーションを開発するとき、または使用する AWS ツールを使用する場合は AWS のサービス、コードまたはツールの認証方法を確立する必要があります AWS。 AWS リソースへのプログラムによるアクセスは、コードが実行される環境と利用可能な AWS アクセスに応じて、さまざまな方法で設定できます。

以下のオプションは、認証情報プロバイダーチェーンの一部です。つまり、共有 AWS `config`ファイルと `credentials` ファイルを適切に設定することで、 AWS SDK またはツールはその認証方法を自動的に検出して使用します。

アプリケーションコードを認証する方法の選択

アプリケーション AWS によって に対して行われた呼び出しを認証する方法を選択します。

コード INSIDE AWS のサービス (Amazon EC2、Lambda、Amazon ECS、Amazon EKS、CodeBuild など) を実行していますか?

コードが実行されると AWS、アプリケーションが認証情報を自動的に使用できるようになります。例えば、アプリケーションが Amazon Elastic Compute Cloud でホストされていて、そのリソースに関連付けられた IAM ロールがある場合、認証情報はアプリケーションで自動的に使用可能になります。同様に、Amazon ECS または Amazon EKS コンテナを使用する場合、IAM ロールの認証情報セットは、SDK の認証情報プロバイダーチェーンを介してコンテナ内で実行されるコードによって自動的に取得できます。

コードは Amazon Elastic Compute Cloud インスタンスにありますか?

IAM ロールを使用して Amazon EC2 にデプロイされたアプリケーションを認証する — IAM ロールを使用して、Amazon EC2 インスタンスでアプリケーションを安全に実行します。

コードは AWS Lambda 関数にありますか?

Lambda 関数を作成するときに、Lambda により最小限のアクセス許可で実行ロールが作成されます。 AWS SDK またはツールは、実行時に Lambda 実行環境を介して Lambda にアタッチされた IAM ロールを自動的に使用します。

コードは Amazon Elastic Container Service (Amazon EC2 または Amazon ECS AWS Fargate の場合) にありますか?

タスク用の IAM ロールを使用します。タスクロールを作成し、Amazon ECS タスク定義でそのロールを指定する必要があります。 AWS SDK またはツールは、実行時にタスクに割り当てられた IAM ロールを Amazon ECS メタデータを介して自動的に使用します。

コードは Amazon Elastic Kubernetes Service にありますか?

Amazon EKS Pod Identity を使用することをお勧めします。

注: サービスアカウントの IAM ロール (IRSA) が自身のニーズに適していると思われる場合は、「Amazon EKS ユーザーガイド」の「EKS Pod Identity と IRSA の比較」を参照してください。

コードは で実行されていますか AWS CodeBuild

「Using identity-based policies for CodeBuild」を参照してください。

コードは別の AWS のサービスにありますか?

AWS のサービスの専用のガイドを参照してください。でコードを実行すると AWS、SDK 認証情報プロバイダーチェーンは認証情報を自動的に取得して更新できます。

モバイルアプリケーションまたはクライアントベースのウェブアプリケーションを作成していますか?

アクセスを必要とするモバイルアプリケーションまたはクライアントベースのウェブアプリケーションを作成する場合は AWS、ウェブ ID フェデレーションを使用して一時的な AWS セキュリティ認証情報を動的にリクエストするようにアプリケーションを構築します。

ウェブ ID フェデレーションを使用すると、カスタムサインインコードを作成したり独自のユーザー ID を管理したりする必要はありません。その代わりに、アプリのユーザーは、よく知られている外部 ID プロバイダー (IdP) (例: Login with Amazon、Facebook、Google などの OpenID Connect (OIDC) 互換の IdP) を使用してサインインすることができます。認証トークンを受け取り、そのトークンを の一時的なセキュリティ認証情報と交換して、 AWS のリソースを使用するアクセス許可を持つ IAM ロールにマッピングできます AWS アカウント。

SDK またはツールへ設定する方法については、「ウェブ ID または OpenID Connect でロールを引き受けて AWS SDKsとツールを認証する」を参照してください。

モバイルアプリケーションに対しては、Amazon Cognito の使用をお勧めします。Amazon Cognito は ID ブローカーとして機能し、ユーザーの代わりに多くのフェデレーション作業を行います。詳細

については、「IAM ユーザーガイド」の「モバイルアプリに対する Amazon Cognito の使用」を参照してください。

コードをローカルで開発して実行していますか?

コンソール認証情報を使用して AWS SDKsとツールを認証するをお勧めします。

ブラウザベースの迅速な認証フローの後、 は CLI AWS Tools for PowerShell や SDK AWS などのローカル開発ツールで動作する一時的な認証情報 AWS を自動的に生成します。 AWS SDKs

AWS アカウントアクセスに Identity Center を使用する場合

AWS アカウントへのアクセス権が既にある場合、またはワークフォースのアクセスを管理する必要がある場合は、IAM Identity Center を使用して AWS SDK とツールを認証します。セキュリティのベストプラクティスとして、IAM Identity Center AWS Organizations で を使用して、すべての AWS アカウントへのアクセスを管理することをお勧めします。IAM Identity Center でユーザーを作成するか、Microsoft Active Directory を使用するか、SAML 2.0 ID プロバイダー (IdP) を使用するか、IdP を個別に AWS アカウントにフェデレーションできます。リージョンが IAM アイデンティティセンターをサポートしているかどうかを確認するには、Amazon Web Services 全般のリファレンスのIAM Identity Center を使用して AWS SDK とツールを認証する「IAM アイデンティティセンターのエンドポイントとクォータ」を参照してください。

他の認証方法をお探しの場合

ターゲットロール`sts:AssumeRole`への アクセス許可を持つ最小特権の IAM ユーザーを作成します。次に、そのユーザーの `source_profile` セットアップを使用してロールを引き受けるようにプロファイルを設定します。

環境変数または共有認証情報ファイルを介して一時的な IAM AWS 認証情報を使用することもできます。「 AWS SDKs」を参照してください。

注: サンドボックス環境または学習環境でのみ、長期的な認証情報を使用して AWS SDKsとツールを認証することを検討できます。

このコードはオンプレミスまたはハイブリッド/オンデマンド VM (Amazon S3 から読み書きするサーバーやクラウドにデプロイする Jenkins など) で実行されていますか?

X.509 クライアント証明書を使用していますか?

はい:「IAM Roles Anywhere を使用した AWS SDKsとツールの認証」を参照してください。IAM Roles Anywhere を使用して、 の外部で実行されるサーバー、コンテナ、アプリケーションなどの

ワークロードの一時的なセキュリティ認証情報を IAM で取得できます AWS。IAM Roles Anywhere を使用するには、ワークロードで X.509 証明書を使用する必要があります。

環境は、フェデレーティッド ID プロバイダー (Microsoft Entra や Okta など) に安全に接続して一時的な AWS 認証情報をリクエストできますか?

はい:「プロセス認証情報プロバイダー」を使用します

プロセス認証情報プロバイダー を使用して、実行時に認証情報を自動的に取得します。これらのシステムでは、ヘルパーツールまたはプラグインを使用して認証情報を取得し、`sts:AssumeRole` を使用してバックグラウンドで IAM ロールを引き受ける場合があります。

いいえ: 経由で挿入された一時的な認証情報を使用する AWS Secrets Manager

経由で挿入された一時的な認証情報を使用します AWS Secrets Manager。有効期間の短いアクセスキーを取得するオプションについては、「IAM ユーザーガイド」の「一時的なセキュリティ認証情報をリクエストする」を参照してください。これらの一時的な認証情報を保存するオプションについては、「AWS アクセスキー」を参照してください。

これらの認証情報を使用して、本番環境のシークレットまたは有効期間の長いロールベースの認証情報を保存できる Secrets Manager からより広範なアプリケーションのアクセス許可を安全に取得できます。

にないサードパーティー製ツールを使用していますか AWS?

認証情報を取得するための最適なガイダンスについては、サードパーティープロバイダーが作成したドキュメントを使用してください。

サードパーティーがドキュメントを提供していない場合、一時的な認証情報を安全に挿入できますか?

はい: 環境変数と一時的な AWS STS 認証情報を使用します。

いいえ: 暗号化されたシークレットマネージャー に保存されている静的アクセスキーを使用します (最後の手段)。

認証方法

AWS 環境内で実行されるコードの認証方法

コードが実行されると AWS、アプリケーションが認証情報を自動的に使用できるようになります。例えば、アプリケーションが Amazon Elastic Compute Cloud でホストされていて、そのリソースに

関連付けられた IAM ロールがある場合、認証情報はアプリケーションで自動的に使用可能になります。同様に、Amazon ECS または Amazon EKS コンテナを使用する場合、IAM ロールの認証情報セットは、SDK の認証情報プロバイダーチェーンを介してコンテナ内で実行されるコードによって自動的に取得できます。

- IAM ロールを使用して Amazon EC2 にデプロイされたアプリケーションを認証する — IAM ロールを使用して、Amazon EC2 インスタンスでアプリケーションを安全に実行します。
- IAM Identity Center AWS を使用して、次の方法でプログラムで とやり取りできます。
  - AWS CloudShell コンソールから AWS CLI コマンドを実行するには、 を使用します。
  - ソフトウェア開発チーム向けのクラウドベースのコラボレーションスペースを試すには、「Amazon CodeCatalyst」 のご使用を検討ください。

ウェブベースのアイデンティティープロバイダーによる認証 - モバイルまたはクライアントベースのウェブアプリケーション

アクセスを必要とするモバイルアプリケーションまたはクライアントベースのウェブアプリケーションを作成する場合は AWS、ウェブ ID フェデレーションを使用して一時的な AWS セキュリティ認証情報を動的にリクエストするようにアプリケーションを構築します。

ウェブ ID フェデレーションを使用すると、カスタムサインインコードを作成したり独自のユーザー ID を管理したりする必要はありません。その代わりに、アプリのユーザーは、よく知られている外部 ID プロバイダー (IdP) (例: Login with Amazon、Facebook、Google などの OpenID Connect (OIDC) 互換の IdP) を使用してサインインすることができます。認証トークンを受け取り、そのトークンを の一時的なセキュリティ認証情報と交換して、 AWS のリソースを使用するアクセス許可を持つ IAM ロールにマッピングできます AWS アカウント。

SDK またはツールへ設定する方法については、「ウェブ ID または OpenID Connect でロールを引き受けて AWS SDKsとツールを認証する」を参照してください。

モバイルアプリケーションに対しては、Amazon Cognito の使用をお勧めします。Amazon Cognito は ID ブローカーとして機能し、ユーザーの代わりに多くのフェデレーション作業を行います。詳細については、「IAM ユーザーガイド」の「モバイルアプリに対する Amazon Cognito の使用」を参照してください。

ローカル (AWS以外) で実行されるコードの認証オプション

- コンソール認証情報を使用して AWS SDKsとツールを認証する – この機能は コマンドラインインターフェイスと Tools for PowerShell AWS の両方で動作し、 AWS CLI、Tools for PowerShell、などのローカル開発ツールで動作する更新可能な認証情報を提供します AWS。

- IAM Identity Center を使用して AWS SDK とツールを認証する – セキュリティのベストプラクティスとして、IAM Identity Center AWS Organizations で を使用して、すべての へのアクセスを管理することをお勧めします AWS アカウント。でユーザーを作成する AWS IAM アイデンティティセンターか、Microsoft Active Directory を使用するか、SAML 2.0 ID プロバイダー (IdP) を使用するか、IdP を個別にフェデレーションできます AWS アカウント。お使いのリージョンが IAM Identity Center をサポートしているかどうかを確認するには、Amazon Web Services 全般のリファレンス の「AWS IAM アイデンティティセンター エンドポイントとクォータ」を参照してください。
- IAM Roles Anywhere を使用した AWS SDKsとツールの認証 – IAM Roles Anywhere を使用して、の外部で実行されるサーバー、コンテナ、アプリケーションなどのワークロードの一時的なセキュリティ認証情報を IAM で取得できます AWS。IAM Roles Anywhere を使用するには、ワークロードで X.509 証明書を使用する必要があります。
- AWS SDKsとツールを認証するための AWS 認証情報を持つロールの引き受け – IAM ロールを引き受けて、それ以外の場合はアクセスできない AWS リソースに一時的にアクセスできます。
- AWS アクセスキーを使用して AWS SDKsとツールを認証する – 利便性が低い、または AWS リソースのセキュリティリスクが高まる可能性があるその他のオプション。

アクセス管理に関する詳細情報

IAM ユーザーガイドには、 AWS リソースへのアクセスを安全に制御するための以下の情報が記載されています。

- IAM ID (ユーザー、ユーザーグループ、ロール) — での ID の基本を理解します AWS。
- IAM におけるセキュリティのベストプラクティス —「責任分担モデル」 に従って AWS アプリケーションを開発する際に従うべきセキュリティ上の推奨事項。

Amazon Web Services 全般のリファレンス には、以下に関する基本的な基本事項があります。

- AWS 認証情報の理解と取得 — コンソールアクセスとプログラムアクセスの両方に関するアクセスキーオプションと管理プラクティス。

AWS のサービスにアクセスするための IAM Identity Center の信頼できる ID の伝播 (TIP) プラグイン

- TIP プラグインを使用して にアクセスする AWS のサービス – Amazon Q Business または信頼できる ID の伝播をサポートする他のサービス用のアプリケーションを作成し、 AWS SDK for Java

または を使用している場合は AWS SDK for JavaScript、TIP プラグインを使用して認可を合理化できます。

AWS ビルダー ID

は、すでに所有 AWS アカウント している、または作成する可能性のあるものを AWS ビルダー ID 補完します。は作成した AWS リソースのコンテナ AWS アカウント として機能し、それらのリソースのセキュリティ境界を提供しますが、 はユーザーを個人として AWS ビルダー ID 表します。を使用してサインイン AWS ビルダー ID すると、Amazon Q や Amazon CodeCatalyst などの開発者ツールやサービスにアクセスできます。

- AWS サインイン ユーザーガイドの「 でサインイン AWS ビルダー IDする – を作成して使用する方法 AWS ビルダー ID と、ビルダー ID が提供する内容について説明します。
- CodeCatalyst の概念 - Amazon CodeCatalyst ユーザーガイドでの AWS ビルダー ID – CodeCatalyst での AWS ビルダー IDの使用方法について説明します。

コンソール認証情報を使用して AWS SDKsとツールを認証する

ローカル環境やその他のAWS 非コンピューティングサービス環境で AWS アプリケーションを開発するときは、コンソール AWS 認証情報を使用することをお勧めします。Amazon Elastic Compute Cloud (Amazon EC2) や AWS CloudShell などの AWS リソースで開発する場合は、代わりにそのサービスから認証情報を取得することをお勧めします。

IAM Identity Center を使用して認証することもできますIAM Identity Center を使用して AWS SDK とツールを認証する。このオプションは、組織がワークフォースのアクセスを管理する一般的な方法であり、アイデンティティセンターを有効にする必要があります。

動作の仕組み

コンソール認証情報を使用して AWS ローカル開発にログインすると、既存の AWS マネジメントコンソールのサインイン認証情報を使用して AWS サービスにプログラムでアクセスできます。ブラウザベースの認証フローの後、 は CLI、Tools for PowerShell、SDK AWS などのローカル開発ツールで動作する一時的な認証情報 AWS を生成します。 AWS SDKs この機能は、特に長期的なアクセスキーの管理よりもインタラクティブ認証を希望する場合、CLI AWS 認証情報の設定と管理のプロセスを簡素化します。

このプロセスでは、最初のアカウント設定時に作成されたルート認証情報、IAM ユーザー、または ID プロバイダーからのフェデレーティッド ID を使用して認証できます。

開発に SDKs を使用する場合、SDK クライアントは を介して一時的な認証情報を使用しますAWS SDKs標準化された認証情報プロバイダー。を設定することもできますログイン認証情報プロバイダー。

ログインコマンドによる認証は、CLI と Tools for PowerShell AWS の両方でサポートされています。

- コンソール認証情報を使用して AWS ローカル開発にログインする
- AWS Tools for PowerShell ユーザーガイドのコンソール認証情報を使用してログインする

IAM Identity Center を使用して AWS SDK とツールを認証する

AWS IAM アイデンティティセンター は、AWS 非コンピューティングサービス環境でアプリケーションを開発する AWS ときに AWS 認証情報を提供するために使用できます。Amazon Elastic Compute Cloud (Amazon EC2) や などの AWS リソースで開発している場合は AWS Cloud9、代わりにそのサービスから認証情報を取得することをお勧めします。

AWS アカウントアクセスに Identity Center を既に使用している場合、または組織のアクセスを管理する必要がある場合は、IAM Identity Center 認証を使用します。

このチュートリアルでは、IAM Identity Center アクセスを確立し、 AWS アクセスポータルと を使用して SDK またはツール用に設定します AWS CLI。

- AWS アクセスポータルは、IAM アイデンティティセンターに手動でサインインするウェブの場所です。URL のフォーマットは `d-xxxxxxxxxx.awsapps.com/start`、または *`your_subdomain`*`.awsapps.com/start` です。 AWS アクセスポータルにサインインすると、そのユーザー用に設定された ロール AWS アカウント と ロールを表示できます。この手順では、AWS アクセスポータルを使用して、SDK/ツール認証プロセスに必要な設定値を取得します。
- AWS CLI は、コードによって行われた API コールに IAM アイデンティティセンター認証を使用するように SDK またはツールを設定するために使用されます。この 1 回限りのプロセスでは、共有 AWS `config`ファイルが更新され、コードの実行時に SDK またはツールによって使用されます。

前提条件

この手順を開始する前に、次の手順を完了しておく必要があります。

- がない場合は AWS アカウント、 にサインアップします AWS アカウント。
- IAM Identity Center をまだ有効にしていない場合は、「AWS IAM アイデンティティセンター ユーザーガイド」の手順に従って IAM Identity Center を有効にします。

IAM Identity Center を使用してプログラムによるアクセスを設定します

ステップ 1：アクセスを確立し、適切なアクセス許可セットを選択します

AWS 認証情報にアクセスするには、次のいずれかの方法を選択します。

IAM Identity Center 経由のアクセスを確立していません

1. 「AWS IAM アイデンティティセンター ユーザーガイド」の「Configure user access with the default IAM Identity Center directory」の手順に従って、ユーザーを追加し、管理アクセス許可を追加します。
2. `AdministratorAccess` アクセス許可セットは、通常の開発には使用しないでください。代わりに、雇用主がこの目的のためにカスタムアクセス許可セットを作成していない限り、定義済みの `PowerUserAccess` アクセス許可セットを使用することをお勧めします。

   再度同じ「Configure user access with the default IAM Identity Center directory」の手順に従います。ただし今回は次のようにします。

   - *`Admin team`* グループを作成する代わりに *`Dev team`* グループを作成し、その後の手順ではこれを代わりに使用します。
   - 既存のユーザーを使用できますが、そのユーザーを新しい *`Dev team`* グループに追加する必要があります。
   - *`AdministratorAccess`* アクセス許可セットを作成する代わりに *`PowerUserAccess`* アクセス許可セットを作成し、その後の手順ではこれを代わりに使用します。

   完了したら、以下があるはずです。

   - `Dev team` グループ。
   - `Dev team` グループにアタッチされた `PowerUserAccess` アクセス許可セット。
   - `Dev team` グループに追加されたユーザー。
3. ポータルを終了し、再度サインインして、 `Administrator`または の AWS アカウント および オプションを確認しますPowerUserAccess。ツール/SDK を使用する場合は `PowerUserAccess` を選択します。

雇用主が管理するフェデレーティッド ID プロバイダー (Microsoft Entra や Okta など) AWS を通じてに既にアクセスしている

ID プロバイダーのポータル AWS から にサインインします。クラウド管理者がユーザー `PowerUserAccess` (開発者) にアクセス許可を付与している場合は、アクセスできる AWS アカウント とアクセス許可セットが表示されます。アクセス許可セットの名前の横に、そのアクセス許可セットを使用してアカウントに手動またはプログラムでアクセスするオプションが表示されます。

カスタム実装では、アクセス許可セット名が異なるなど、エクスペリエンスが異なる場合があります。どのアクセス許可セットを使用すればよいかわからない場合は、IT チームにお問い合わせください。

雇用主が管理する AWS アクセスポータル AWS から に既にアクセスできる

AWS アクセスポータル AWS から にサインインします。クラウド管理者から `PowerUserAccess` (開発者) アクセス許可を付与されている場合は、アクセス権のある AWS アカウント とアクセス許可セットが表示されます。アクセス許可セットの名前の横に、そのアクセス許可セットを使用してアカウントに手動またはプログラムでアクセスするオプションが表示されます。

雇用主が管理するフェデレーティッドカスタム ID プロバイダー AWS を通じて に既にアクセスできる

サポートについては、IT チームにお問い合わせください。

ステップ 2：IAM Identity Center を使用するように SDK とツールを設定します

1. 開発マシンに最新の AWS CLIをインストールします。

   a. 「AWS Command Line Interface ユーザーガイド」の「AWS CLI の最新バージョンをインストールまたは更新します。」を参照してください。

   b. (オプション) AWS CLI が動作していることを確認するには、コマンドプロンプトを開き、`aws --version` コマンドを実行します。

2. AWS アクセスポータルにサインインします。この URL は、雇用主から提供されたり、「ステップ 1: アクセスを確立する」の後に E メールで取得したりする場合があります。これ以外の場合は、https://console.aws.amazon.com/singlesignon/ のダッシュボードで AWS アクセスポータル URL を確認できます。

   a. AWS アクセスポータルの アカウント タブで、管理する個々のアカウントを選択します。ユーザーのロールが表示されます。[アクセスキー] を選択して、適切なアクセス許可セット

のコマンドラインまたはプログラムによるアクセスの認証情報を取得します。事前定義された PowerUserAccess 許可セットを使用するか、またはユーザーもしくは雇用主が開発のために最小特権の許可を適用するために作成した許可セットを使用してください。

b. [認証情報の取得] ダイアログボックスで、オペレーティングシステムに応じて、[MacOS と Linux] または [Windows] を選択します。

c. [IAM Identity Center 認証情報] メソッドを選択して、次のステップに必要な Issuer URL と SSO Region の値を取得します。注: SSO Start URL は Issuer URL と置き換えて使用できます。

3. AWS CLI コマンドプロンプトで、 aws configure sso コマンドを実行します。プロンプトが表示されたら、前のステップで収集した設定値を入力します。この AWS CLI コマンドの詳細については、aws configure sso「ウィザードでプロファイルを設定する」を参照してください。

a. プロンプト SSO Start URL で、Issuer URL 用に取得した値を入力します。

b. [CLI プロファイル名]には、開始時に#####を入力することをお勧めします。デフォルト以外の（名前付き）プロファイルとそれに関連する環境変数を設定する方法については、「プロファイル」を参照してください。

4. (オプション) AWS CLI コマンドプロンプトで、 aws sts get-caller-identity コマンドを実行してアクティブなセッション ID を確認します。レスポンスには、設定した IAM Identity Center アクセス許可セットが表示されるはずです。

5. AWS SDK を使用している場合は、開発環境で SDK 用のアプリケーションを作成します。

a. 一部の SDK では、IAM Identity Center 認証を使用する前に、SSO や SSOOIDC などの追加パッケージをアプリケーションに追加する必要があります。詳細については、具体的な SDK を参照してください。

b. へのアクセスを以前に設定している場合は AWS、共有 AWS credentialsファイルで を確認しますAWS アクセスキー。認証情報プロバイダーチェーンを理解する 優先順位により、SDK またはツールが IAM Identity Center の認証情報を使用する前に、静的認証情報をすべて削除する必要があります。

SDK とツールがこの設定を使用して認証情報を使用および更新する方法についての詳細は、「AWS SDKsとツールの IAM Identity Center 認証の解決方法」を参照してください。

IAM Identity Center プロバイダー設定を共有 config ファイルで直接設定するには、このガイドの「IAM Identity Center 認証情報プロバイダー」を参照してください。

ポータルアクセスセッションを更新する

アクセスはいずれ期限切れになり、SDK またはツールで認証エラーが発生します。この有効期限は、設定したセッションの長さによって異なります。必要に応じてアクセスポータルセッションを再度更新するには、 AWS CLI を使用して `aws sso login` コマンドを実行します。

IAM Identity Center アクセスポータルのセッション期間とアクセス許可セットのセッション期間の両方を延長できます。これにより、 AWS CLIに手動でサインインし直す必要が出るまでにコードを実行できる時間が長くなります。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」の以下のトピックを参照してください。

- IAM Identity Center セッション期間 – ユーザーが AWS ポータルにアクセスするセッションの期間を設定します
- アクセス許可セットセッション期間 – セッション期間を設定します

AWS SDKsとツールの IAM Identity Center 認証の解決方法

IAM Identity Center の関連条項

以下の用語は、 AWS IAM アイデンティティセンターの背後にあるプロセスと設定を理解するのに役立ちます。 AWS SDK APIs のドキュメントでは、これらの認証概念の一部に IAM Identity Center とは異なる名前を使用しています。両方の名前を知っておくと役に立ちます。

次の表は、別名の相互関係を示しています。

IAM アイデンティティセンターの名前	SDK API の名前	説明
アイデンティティセンター	`sso`	AWS Single Sign-On の名前は変更されますが、`sso`API 名前空間は下位互換性のために元の名前を保持します。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「What is IAM Identity Center」(IAM Identity

IAM アイデンティティセンターの名前	SDK API の名前	説明
		Center とは) を参照してください。
IAM Identity Center コンソール 管理コンソール		シングルサインオンの設定に使用するコンソール。
AWS アクセスポータル URL		IAM Identity Center のアカウントに一意の URL (`https://xxx.awsapps.com/start` など) 。IAM Identity Center のサインイン認証情報を使用してこのポータルにサインインします。
IAM Identity Center アクセスポータルセッション	認証セッション	発信者がベアラーアクセストークンを取得できます。
アクセス許可設定セッション		SDK が内部的に AWS のサービス 呼び出しに使用する IAM セッション。非公式な議論では、このセッションが誤って「ロールセッション」と呼ばれることがあります。
アクセス許可セット認証情報	AWS 認証情報 sigv4 認証情報	SDK がほとんどの AWS のサービス 呼び出し (特にすべての sigv4 AWS のサービス 呼び出し) に実際に使用する認証情報。非公式な議論では、このセッションが誤って「ロール認証情報」と呼ばれることがあります。

IAM アイデンティティセンターの名前	SDK API の名前	説明
IAM Identity Center 認証情報プロバイダー	SSO 認証情報プロバイダー	認証情報の取得方法（機能を提供するクラスやモジュールなど）。

の SDK 認証情報解決を理解する AWS のサービス

IAM Identity Center API は、ベアラートークンの認証情報を sigv4 の認証情報と交換します。ほとんど AWS のサービス は sigv4 APIs、 Amazon CodeWhisperer や などのいくつかの例外があります Amazon CodeCatalyst。以下では、 を通じてアプリケーションコードのほとんどの AWS のサービス 呼び出しをサポートするための認証情報解決プロセスについて説明します AWS IAM アイデンティティセンター。

AWS アクセスポータルセッションを開始する

- まず、認証情報を使用してセッションにサインインします。
  - AWS Command Line Interface () で `aws sso login` コマンドを使用しますAWS CLI。アクティブなセッションがまだない場合は、新しい IAM Identity Center セッションが開始されます。
- 新しいセッションを開始すると、IAM Identity Center から更新トークンとアクセストークンを受け取ります。 AWS CLI また、 は SSO キャッシュ JSON ファイルを新しいアクセストークンと更新トークンで更新し、SDKs で使用できるようにします。
- 既にアクティブなセッションがある場合、 AWS CLI コマンドは既存のセッションを再利用し、既存のセッションの有効期限が切れるたびに期限切れになります。IAM Identity Center セッションの長さを設定する方法については、「 AWS IAM アイデンティティセンター ユーザーガイド」の「ユーザーの AWS アクセスポータルセッションの期間を設定する」を参照してください。
  - 頻繁にサインインする必要性を減らすため、セッションの最大期間が 90 日間に延長されました。

SDK が AWS のサービス 呼び出しの認証情報を取得する方法

SDKsは、サービスごとにクライアントオブジェクトをインスタンス化 AWS のサービス するときにへのアクセスを提供します。共有 AWS `config`ファイルの選択したプロファイルが IAM アイデン

ティティセンターの認証情報解決用に設定されている場合、IAM アイデンティティセンターを使用してアプリケーションの認証情報を解決します。

- 認証情報解決プロセスは、ランタイムにクライアントが作成されるときに完了します。

IAM Identity Center のシングルサインオンを使用して sigv4 API の認証情報を取得するために、SDK は IAM Identity Center のアクセストークンを使用して IAM セッションを取得します。この IAM セッションはアクセス許可セットセッションと呼ばれ、IAM ロールを引き受けることで SDK AWS へのアクセスを提供します。

- アクセス許可セットのセッション期間は IAM Identity Center のセッション期間とは独立して設定されます。
  - アクセス許可セットのセッション期間を設定する方法については、「AWS IAM アイデンティティセンター ユーザーガイド」の「セッション期間の設定」を参照してください。
- アクセス許可セット認証情報は、ほとんどの AWS SDK API ドキュメントでAWS 認証情報と sigv4 認証情報とも呼ばれることに注意してください。

アクセス許可セットの認証情報は、IAM Identity Center API の getRoleCredentials への呼び出しから SDK に返されます。SDK のクライアントオブジェクトは、引き受けた IAM ロールを使用して AWS のサービス、アカウントのバケットを一覧表示するように Amazon S3 に依頼するなど、 を呼び出します。クライアントオブジェクトは、アクセス許可セットセッションの有効期限が切れるまで、それらのアクセス許可セット認証情報を使用して操作を続けることができます。

セッションの有効期限と更新

SSO トークンプロバイダー設定 を使用する場合、IAM Identity Center から取得した 1 時間単位のアクセストークンは、更新トークンを使用して自動的に更新されます。

- SDK がアクセストークンを使用しようとしたときにそのアクセストークンの有効期限が切れている場合、SDK は更新トークンを使用して新しいアクセストークンの取得を試みます。IAM Identity Center は、更新トークンを IAM Identity Center のアクセスポータルのセッション期間と比較します。更新トークンの有効期限が切れていない場合、IAM Identity Center は別のアクセストークンで応答します。
- このアクセストークンは、既存のクライアントのアクセス許可セットセッションを更新したり、新しいクライアントの認証情報を解決したりするために使用できます。

ただし、IAM Identity Center アクセスポータルセッションの有効期限が切れると、新しいアクセストークンは付与されません。そのため、アクセス許可セットの有効期間は更新できません。既存のクライアントのキャッシュされたアクセス許可セットセッションの長さがタイムアウトになると、有効期限が切れます (アクセスも失われます)。

IAM Identity Center セッションの有効期限が切れるとすぐに、新しいクライアントを作成するコードは認証に失敗します。これは、アクセス許可セットの認証情報がキャッシュされないためです。有効なアクセストークンが得られるまで、コードで新しいクライアントを作成したり、認証情報解決プロセスを完了したりすることはできません。

まとめると、SDK が新しいアクセス許可セット認証情報を必要とする場合、SDK はまず有効な既存の認証情報を確認し、それらを使用します。これは、認証情報が新しいクライアントのものか、認証情報の有効期限が切れた既存のクライアントのものかに関係なく適用されます。認証情報が見つからない、または有効でない場合、SDK は IAM Identity Center API を呼び出して新しい認証情報を取得します。API を呼び出すには、アクセストークンが必要です。アクセストークンの有効期限が切れている場合、SDK は更新トークンを使用して、IAM Identity Center サービスから新しいアクセストークンを取得しようとします。このトークンは、IAM Identity Center アクセスポータルセッションの有効期限が切れていない場合に付与されます。

IAM Roles Anywhere を使用した AWS SDKsとツールの認証

IAM Roles Anywhere を使用して、 の外部で実行されるサーバー、コンテナ、アプリケーションなどのワークロードの一時的なセキュリティ認証情報を IAM で取得できます AWS。IAM Roles Anywhere を使用するには、ワークロードで X.509 証明書を使用する必要があります。IAM Roles Anywhere を認証情報プロバイダーとして設定するのに必要な証明書とプライベートキーは、クラウド管理者が提供する必要があります。

ステップ 1：IAM Roles Anywhere を設定します

IAM Roles Anywhere は、 の外部で実行されるワークロードまたはプロセスの一時的な認証情報を取得する方法を提供します AWS。認証機関との間でトラストアンカーが確立され、関連する IAM ロールの一時的な認証情報を取得できます。このロールは、コードが IAM Roles Anywhere で認証されるときにワークロードが持つアクセス許可を設定します。

トラストアンカー、IAM ロール、IAM Roles Anywhere プロファイルを設定する手順については、IAM Roles Anywhere ユーザーガイドのAWS Identity and Access Management 「Roles Anywhere でのトラストアンカーとプロファイルの作成」を参照してください。

> **Note**
>
> 「IAM Roles Anywhere ユーザーガイド」のプロファイルは、IAM Roles Anywhere サービス内の独特の概念を指しています。共有 AWS configファイル内のプロファイルとは関係ありません。

ステップ 2：IAM Roles Anywhere の使用

IAM Roles Anywhere から一時的なセキュリティ認証情報を取得するには、IAM Roles Anywhere にある認証情報ヘルパーツールを使用してください。この認証情報ツールは IAM Roles Anywhere の署名プロセスを実装します。

認証情報ヘルパーツールをダウンロードする手順については、IAM AWS Identity and Access Management Roles Anywhere ユーザーガイドの「Roles Anywhere から一時的なセキュリティ認証情報を取得する」を参照してください。

AWS SDKs と で IAM Roles Anywhere の一時的なセキュリティ認証情報を使用するには AWS CLI、共有 AWS configファイルで credential_process 設定を設定できます。SDKs と は、 が認証credential_processに使用するプロセス認証情報プロバイダー AWS CLI をサポートします。credential_process を設定する一般的な構造を以下に示します。

```
credential_process = [path to helper tool] [command] [--parameter1 value] [--
parameter2 value] [...]
```

ヘルパーツールの credential-process コマンドは、credential_process 設定と互換性のある標準 JSON 形式で一時的な認証情報を返します。コマンド名にはハイフンが含まれていますが、設定名にはアンダースコアが含まれていることに注意してください。コマンドには以下のパラメータが必要となります。

- private-key – リクエストに署名したプライベートキーへのパス。
- certificate – 証明書へのパス。
- role-arn – 一時的な認証情報を取得するロールの ARN。
- profile-arn – 指定されたロールのマッピングを行うプロファイルの ARN。
- trust-anchor-arn – 認証に使用するトラストアンカーの ARN。

クラウド管理者は、証明書とプライベートキーを提供する必要があります。3 つの ARN 値はすべて AWS マネジメントコンソールからコピーできます。次の例は、ヘルパーツールから一時的な認証情報を取得するように設定した共有 config ファイルを示しています。

```
[profile dev]
credential_process = ./aws_signing_helper credential-process --certificate /
path/to/certificate --private-key /path/to/private-key --trust-anchor-
arn arn:aws:rolesanywhere:region:account:trust-anchor/TA_ID --profile-
arn arn:aws:rolesanywhere:region:account:profile/PROFILE_ID --role-
arn arn:aws:iam::account:role/ROLE_ID
```

オプションのパラメータとその他のヘルパーツールの詳細については、GitHub の「IAM Roles Anywhere 認証情報ヘルパー」を参照してください。

SDK 設定自体とプロセス認証情報プロバイダーの詳細については、このガイドの「プロセス認証情報プロバイダー」を参照してください。

AWS SDKsとツールを認証するための AWS 認証情報を持つロールの引き受け

ロールでは、他の方法ではアクセスできない AWS リソースへのアクセスに、一時的なセキュリティ認証情報のセットを使用する必要があるとします。これらの一時的な認証情報は、アクセスキー ID、シークレットアクセスキー、およびセキュリティトークンで構成されています。 AWS Security Token Service (AWS STS) API リクエストの詳細については、「AWS Security Token Service API リファレンス」の「アクション」を参照してください。

ロールを引き受けるように SDK またはツールを設定するには、まず引き受けるのための特定のロールを作成または特定する必要があります。IAM ロールは、ロール (Amazon リソースネーム (「ARN」)で一意に識別されます。ロールは別のエンティティとの信頼関係を確立します。ロールを使用する信頼されたエンティティは、 AWS のサービス または別のエンティティである可能性があります AWS アカウント。ロールの作成と使用の詳細については、「IAM ユーザーガイド」の「IAM ロールを使用する」を参照してください。

IAM ロールが特定されると、そのロールから信頼されている場合は、そのロールによって付与された権限を使用するように SDK またはツールを設定できます。

> **Note**
>
> 可能な限りリージョンエンドポイントを使用し、 を設定することが AWS ベストプラクティスですAWS リージョン。

IAM ロールの継承

ロールを引き受けると、 は一時的なセキュリティ認証情報のセット AWS STS を返します。これらの認証情報は、別のプロファイル、またはコードが実行されているインスタンスまたはコンテナから取得されます。通常、このタイプのロールの引き受けは、あるアカウントで AWS 認証情報を持っているが、アプリケーションが別のアカウントのリソースにアクセスする必要がある場合に使用されます。

ステップ 1: IAM ロールを設定する

ロールを引き受けるように SDK またはツールを設定するには、まず引き受けるのための特定のロールを作成または特定する必要があります。IAM ロールはロール「ARN」 を使用して一意に識別されます。ロールは別のエンティティとの信頼関係を確立します。通常はアカウント内またはクロスアカウントアクセス用です。詳細については、「IAM ユーザーガイド」の「IAM ロールの作成」を参照してください。

ステップ 2: SDK またはツールを設定する

`credential_source` または `source_profile` から認証情報を取得するように SDK またはツールを設定します。

`credential_source` を使用してAmazon ECS コンテナ、Amazon EC2 インスタンス、または環境変数から認証情報を取得します。

`source_profile` を使用して別のプロファイルから認証情報を取得します。 `source_profile` はまた、ロールチェイニングもサポートしています。ロールチェイニングとは、引き受けたロールを使って別のロールを引き受けるプロファイルの階層構造です。

プロファイルでこれを指定すると、SDK またはツールは対応する AWS STS `AssumeRole` API コールを自動的に実行します。ロールを引き受けて一時的な認証情報を取得して使用するには、共有 AWS `config`ファイルで次の設定値を指定します。これらの設定の詳細については、「ロール認証情報プロバイダーを引き受けます」を参照してください。

- `role_arn` - ステップ 1 で作成された IAM ロール

- credential_source または source_profile のいずれかを設定します
- (オプション) duration_seconds
- (オプション) external_id
- (オプション) mfa_serial
- (オプション) role_session_name

次の例は、 config 共有ファイル内の 2 つの引き受けロールオプションの設定を示しています。

```
role_arn = arn:aws:iam::123456789012:role/my-role-name
credential_source = Ec2InstanceMetadata
```

```
[profile-with-user-that-can-assume-role]
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
aws_session_token=IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZVERYLONGSTRI

[profile dev]
region = us-east-1
output = json
role_arn = arn:aws:iam::123456789012:role/my-role-name
source_profile = profile-with-user-that-can-assume-role
role_session_name = my_session
```

すべての引き受けロールの認証情報プロバイダーの設定の詳細については、このガイドの「ロール認証情報プロバイダーを引き受けます」を参照してください。

ウェブ ID または OpenID Connect でロールを引き受けて AWS SDKsとツールを認証する

ロールでは、他の方法ではアクセスできない AWS リソースへのアクセスに、一時的なセキュリティ認証情報のセットを使用する必要があるとします。これらの一時的な認証情報は、アクセスキー ID、シークレットアクセスキー、およびセキュリティトークンで構成されています。 AWS Security Token Service (AWS STS) API リクエストの詳細については、「AWS Security Token Service API リファレンス」の「アクション」を参照してください。

ロールを引き受けるように SDK またはツールを設定するには、まず引き受けるのための特定のロールを作成または特定する必要があります。IAM ロールは、ロール (Amazon リソースネーム

(「ARN」)で一意に識別されます。ロールは別のエンティティとの信頼関係を確立します。ロールを使用する信頼できるエンティティは、ウェブ ID プロバイダーまたは OpenID Connect (OIDC)、または SAML フェデレーションである可能性があります。IAM ロールの詳細については、「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。

SDK で IAM ロールを設定した後、そのロールが ID プロバイダーを信頼するように設定されている場合は、一時的な AWS 認証情報を取得するために、そのロールを引き受けるように SDK をさらに設定できます。

> **Note**
>
> 可能な限りリージョンエンドポイントを使用し、 を設定することが AWS ベストプラクティスですAWS リージョン。

ウェブアイデンティティまたは OpenID Connect でのフェデレーション

Login With Amazon、Facebook、Google などのパブリック ID プロバイダーの JSON ウェブトークン (JWTs) を使用して、 を使用して一時的な AWS 認証情報を取得できますAssumeRoleWithWebIdentity。使用方法によって、これらの JWT は ID トークンまたはアクセストークンと呼ばれます。EntraId や PingFederate などの OIDC の検出プロトコルと互換性のある ID プロバイダー (IdP) から発行された JWT を使用することもできます。

Amazon Elastic Kubernetes Service を使用している場合、この機能により、Amazon EKS クラスターのサービスアカウントごとに異なる IAM ロールを指定できます。この Kubernetes 機能はJWTs をポッドに配布します。ポッドは、一時的な AWS 認証情報を取得するためにこの認証情報プロバイダーによって使用されます。この Amazon EKS の設定の詳細については、「Amazon EKS ユーザーガイド」の「サービスアカウントの IAM ロール」を参照してください。ただし、より単純なオプションとして、SDK がサポートしている場合は、代わりに Amazon EKS Pod Identities を利用することをお勧めします。

ステップ 1: ID プロバイダーと IAM ロールを設定する

外部 IdP とのフェデレーションを設定するには、IAM ID プロバイダーを使用して、外部 IdP とその設定 AWS について に通知します。これにより、 AWS アカウント と外部 IdP 間の信頼が確立されます。認証のために JSON Web Token (JWT) を使用するように SDK を設定する前に、まず ID プロバイダー (IdP) と、それにアクセスするための IAM ロールを設定する必要があります。これらを設定するには、「IAM ユーザーガイド」の「ウェブ OpenID Connect フェデレーション用のロールの作成 (コンソール)」 を参照してください。

ステップ 2: SDK またはツールを設定する

AWS STS 認証に からの JSON ウェブトークン (JWT) を使用するように SDK またはツールを設定します。

プロファイルでこれを指定すると、SDK またはツールは対応する AWS STS `AssumeRoleWithWebIdentity` API コールを自動的に実行します。ウェブ ID フェデレーションを使用して一時的な認証情報を取得して使用するには、共有 AWS `config`ファイルに次の設定値を指定します。これらの設定の詳細については、「ロール認証情報プロバイダーを引き受けます」を参照してください。

- `role_arn` - ステップ 1 で作成された IAM ロール
- `web_identity_token_file`-外部 IdP から
- (オプション) `duration_seconds`
- (オプション) `role_session_name`

ウェブ IDを使用してロールを引き受ける `config` 共有ファイル設定の例を次に示します。

```
[profile web-identity]
role_arn=arn:aws:iam::123456789012:role/my-role-name
web_identity_token_file=/path/to/a/token
```

> **Note**
>
> モバイルアプリケーションに対しては、Amazon Cognito の使用をお勧めします。Amazon Cognito は ID ブローカーとして機能し、ユーザーの代わりに多くのフェデレーション作業を行います。ただし、Amazon Cognito ID プロバイダーは、他の ID プロバイダーのように SDK やツールのコアライブラリには含まれていません。Amazon Cognito API にアクセスするには、SDK またはツールのビルドまたはライブラリに Amazon Cognito サービスクライアントを含めてください。 AWS SDKsAmazon Cognito デベロッパーガイド」の「コード例」を参照してください。

すべての引き受けロールの認証情報プロバイダーの設定の詳細については、このガイドの「ロール認証情報プロバイダーを引き受けます」を参照してください。

AWS アクセスキーを使用して AWS SDKsとツールを認証する

AWS アクセスキーの使用は、 AWS SDKsとツールを使用する場合の認証のオプションです。

短期の認証情報を使用します

セッション期間の長いオプションを使用するには、IAM Identity Center を使用して AWS SDK とツールを認証する を使用するように SDK またはツールを設定することをお勧めします。

ただし、SDK またはツールの一時認証情報を直接設定する方法については、「短期認証情報を使用して AWS SDKsとツールを認証する」を参照してください。

長期認証情報の使用

> **Warning**
>
> セキュリティリスクを避けるため、専用ソフトウェアを開発するときや実際のデータを扱うときは、IAM ユーザーを認証に使用しないでください。代わりに、AWS IAM アイデンティティセンター などの ID プロバイダーとのフェデレーションを使用してください。

間のアクセスを管理する AWS アカウント

セキュリティのベストプラクティスとして、IAM Identity Center AWS Organizations で を使用して、すべての へのアクセスを管理することをお勧めします AWS アカウント。詳細については、「IAM ユーザーガイド」の「IAM でのセキュリティベストプラクティス」を参照してください。

IAM アイデンティティセンターでユーザーを作成するか、Microsoft Active Directory を使用するか、SAML 2.0 ID プロバイダー (IdP) を使用するか、IdP を個別にフェデレーションできます AWS アカウント。これらのアプローチのいずれかを使用して、ユーザーにシングルサインオンのエクスペリエンスを提供できます。多要素認証 (MFA) を適用し、一時的な認証情報を使用して AWS アカウント アクセスすることもできます。これは IAM ユーザーとは異なります。IAM ユーザーは、共有できる長期的な認証情報であり、 AWS リソースに対するセキュリティリスクが高まる可能性があります。

サンドボックス環境専用の IAM ユーザーを作成する

を初めて使用する場合は AWS、テスト IAM ユーザーを作成し、それを使用してチュートリアルを実行し、 が提供する AWS ものを調べることができます。学習中はこの種の資格情報を使用しても問題ありませんが、サンドボックス環境以外では使用しないことをお勧めします。

以下のユースケースでは、 で IAM ユーザーの使用を開始するのが理にかなっている場合がありますAWS。

- AWS SDK またはツールの使用を開始し、 AWS のサービス サンドボックス環境で探索する。
- 学習の一環として、人間によるサインインプロセスをサポートしない、スケジュールされたスクリプト、ジョブ、その他の自動プロセスを実行する。

これらのユースケース以外で IAM ユーザーを使用している場合は、IAM Identity Center に移行するか、ID プロバイダーを AWS アカウント できるだけ早く にフェデレーションします。詳細については、「AWSでの ID フェデレーション」を参照してください。

IAM ユーザーのアクセスキーを保護する

IAM ユーザーのアクセスキーは定期的に更新する必要があります。「IAM ユーザーガイド」の「Manage access keys for IAM users」のガイダンスに従ってください。IAM ユーザーのアクセスキーを誤って共有したと思われる場合は、アクセスキーを更新してください。

IAM ユーザーアクセスキーは、ローカルマシンの共有 AWS `credentials`ファイルに保存する必要があります。IAM ユーザーのアクセスキーをコードに保存しないでください。IAM ユーザーのアクセスキーを含む設定ファイルは、いずれのソースコード管理ソフトウェアにも含めないでください。オープンソースプロジェクトの git-secrets などの外部ツールを使用すると、機密情報を誤って Git リポジトリにコミットすることを防ぐことができます。詳細については、「IAM ユーザーガイド」の「IAM アイデンティティ (ユーザー、ユーザーグループ、ロール)」を参照してください。

はじめに IAM ユーザーを設定するには、「長期認証情報を使用して AWS SDKsとツールを認証する」を参照してください。

短期認証情報を使用して AWS SDKsとツールを認証する

拡張セッション期間オプションIAM Identity Center を使用して AWS SDK とツールを認証するで使用するように AWS SDK またはツールを設定することをお勧めします。ただし、 AWS アクセスポータルで利用可能な一時的な認証情報をコピーして使用できます。有効期限が切れたら、新しい認証情報をコピーする必要があります。一時的な認証情報は、プロファイルで使用することも、システムプロパティや環境変数の値として使用することもできます。

ベストプラクティス: 認証情報ファイル内のアクセスキーとトークンを手動で管理する代わりに、アプリケーションでは以下から配信される一時的な認証情報を使用することをお勧めします。

- Amazon Elastic Compute Cloud または でアプリケーションを実行するなどのコンピューティング AWS サービス AWS Lambda。
- IAM Identity Center を使用して AWS SDK とツールを認証する などの、認証情報プロバイダーチェーンの別のオプション。
- または プロセス認証情報プロバイダー を使用して一時的な認証情報を取得します。

AWS アクセスポータルから取得した短期認証情報を使用して認証情報ファイルを設定する

1. 認証情報の共有ファイルの作成。
2. 認証情報ファイルに、作業用の一時認証情報を貼り付けるまで、次のプレースホルダーテキストを貼り付けます。

```
[default]
aws_access_key_id=<value from AWS access portal>
aws_secret_access_key=<value from AWS access portal>
aws_session_token=<value from AWS access portal>
```

3. ファイルを保存します。これで、~/.aws/credentials ファイルはローカルの開発システムに存在しているはずです。このファイルには、特定の名前付きプロファイルが指定されていない場合に SDK またはツールが使用する [デフォルト] プロファイルが含まれています。
4. AWS アクセスポータルにサインインします。
5. アクセス AWS ポータルから IAM ロールの認証情報をコピーするには、認証情報の手動更新の手順に従います。

   a. リンク先の手順のステップ 4 で、開発ニーズに合ったアクセスを許可する IAM ロールの名前を選択します。通常、このロールには PowerUserAccess や Developer などの名前が付いています。

   b. リンク先の手順のステップ 7 で、[AWS 認証情報ファイルにプロファイルを手動で追加] オプションを選択し、内容をコピーします。

6. コピーした認証情報をローカル credentials ファイルに貼り付けます。default プロファイルを使用する場合、生成されたプロファイル名は必要ありません。ファイルは以下のようになります。

```
[default]
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
aws_session_token=IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZVERYLONG
```

7. `credentials` ファイルを保存します。

SDK は、サービスクライアントを作成するときに、これらの一時的な認証情報にアクセスしてリクエストごとに使用します。ステップ 5a で選択した IAM ロールの設定により、一時的な認証情報の有効期間が決まります。最大期間は 12 時間です。

一時的な認証情報の有効期限が切れたら、ステップ 4 ~ 7 を繰り返します。

長期認証情報を使用して AWS SDKsとツールを認証する

> **Warning**
>
> セキュリティリスクを避けるため、専用ソフトウェアを開発するときや実際のデータを扱うときは、IAM ユーザーを認証に使用しないでください。代わりに、AWS IAM アイデンティティセンター などの ID プロバイダーとのフェデレーションを使用してください。

IAM ユーザーを使用してコードを実行する場合、開発環境の SDK またはツールは、共有 AWS `credentials`ファイルで長期的な IAM ユーザー認証情報を使用して認証します。「IAM トピックのセキュリティのベストプラクティス」 を確認し、できるだけ早く IAM Identity Center またはその他の一時的な認証情報に移行してください。

認証情報に関する重要な警告とガイダンス

認証情報に関する警告

- お使いのアカウントのルート認証情報を使用して AWS リソースにアクセスしないでください。これらの認証情報は無制限のアカウントアクセスを提供し、取り消すのが困難です。
- アプリケーションファイルにリテラルアクセスキーや認証情報を配置しないでください。これを行うと、パブリックリポジトリにプロジェクトをアップロードするなど、誤って認証情報が公開されるリスクが発生します。
- プロジェクト領域に認証情報を含むファイルを含めないでください。
- 共有 AWS `credentials`ファイルに保存されている認証情報はすべてプレーンテキストで保存されることに注意してください。

認証情報を安全に管理するための追加のガイダンス

AWS 認証情報を安全に管理する方法の一般的な説明については、『』のAWS 「アクセスキーを管理するためのベストプラクティス」を参照してくださいAWS 全般のリファレンス。そこでの説明に加えて、以下の点を考慮してください。

- Amazon Elastic Container Service (Amazon ECS) タスクで、タスク用の IAM ロールを使用します。
- Amazon EC2 インスタンスで実行中のアプリケーションに対して、IAM ロールを使用します。

前提条件: AWS アカウントを作成する

IAM ユーザーを使用して AWS サービスにアクセスするには、 AWS アカウントと AWS 認証情報が必要です。

1. アカウントを作成する。

   AWS アカウントを作成するには、「 AWS アカウント管理 リファレンスガイド」の「開始方法: 初めての AWS ユーザーですか?」を参照してください。

2. 管理者ユーザーを作成します。

   マネジメントコンソールとサービスへのアクセスには、root ユーザーアカウント (作成した初期アカウント) を使用しないでください。代わりに、「IAM ユーザーガイド」の「管理ユーザーを作成する」で説明されているように、管理ユーザーアカウントを作成します。

   管理ユーザーアカウントを作成してログインの詳細を記録したら、ルートユーザーアカウントから確実にサインアウトし、管理者アカウントを使用して再度サインインします。

これらのアカウントはいずれも、 での開発 AWS や でのアプリケーションの実行には適していません AWS。そのためには、これらのタスクに適したユーザー、アクセス許可セットまたはサービスロールを作成する必要があります。詳細については、「IAM ユーザーガイド」の「最小特権アクセス許可を適用する」を参照してください。

ステップ 1: IAM ユーザーを作成する

- 「IAM ユーザーガイド」の「IAM ユーザーの作成 (コンソール)」の手順に従って IAM ユーザーを作成します。IAM ユーザーを作成する場合:

- [AWS マネジメントコンソールへのユーザーアクセスを提供] を選択することをお勧めします。これにより、 AWS CloudTrail 診断ログの確認や Amazon Simple Storage Service へのファイルのアップロードなど、ビジュアル環境で実行しているコードに関連する AWS のサービス を表示できます。これは、コードをデバッグするときに役立ちます。
- [アクセス許可を設定] - [アクセス許可オプション] で、このユーザーにアクセス許可を割り当てる方法として [ポリシーを直接アタッチする] を選択します。
  - ほとんどの「開始方法」 SDK チュートリアルでは、Amazon S3 サービスを例として使用しています。アプリケーションに Amazon S3 へのフルアクセスを提供するには、このユーザーにアタッチする `AmazonS3FullAccess` ポリシーを選択します。
- アクセス許可の境界またはタグの設定に関する手順のオプションのステップは無視できます。

ステップ 2: アクセスキーを取得する

1. IAM コンソールのナビゲーションペインで [ユーザー] を選択し、以前に作成したユーザーの **`User name`** を選択します。
2. ユーザーのページで、[セキュリティ認証情報] ページを選択します。次に、[アクセスキー] で [アクセスキーの作成] を選択します。
3. [アクセスキーを作成ステップ1]で、[コマンドラインインターフェイス (CLI)]または[ローカルコード]を選択します。どちらのオプションも、 AWS CLI と SDKs の両方で使用するのと同じタイプのキーを生成します。
4. [アクセスキーの作成ステップ 2] で、オプションのタグを入力して [次へ] を選択します。
5. [アクセスキーの作成ステップ 3] で、[.csv ファイルをダウンロード] を選択し、IAM ユーザーのアクセスキーとシークレットアクセスキーを含む `.csv` ファイルを保存します。この情報は後で必要になります。

   > **Warning**
   >
   > 適切なセキュリティ対策を講じてこれらの認証情報を安全に保管してください。

6. [完了] を選択します。

ステップ 3: `credentials` 共有ファイルを更新する

1. 共有 AWS `credentials` ファイルを作成するか、開きます。このファイルは、`~/.aws/credentials` Linux および macOS システム、および `%USERPROFILE%\.aws\credentials` Windows 上にあります。詳細については、「認証情報ファイルの場所」 を参照してください。
2. 共有 `credentials` ファイルに次のテキストを追加します。ID 値の例とキー値の例を、先にダウンロードした `.csv` ファイルの値に置き換えます。

```
[default]
aws_access_key_id = AKIAIOSFODNN7EXAMPLE
aws_secret_access_key = wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

3. ファイルを保存します。

認証情報を保存する最も一般的な方法は `credentials` 共有ファイルです。これらは環境変数として設定することもできます。AWS アクセスキーの環境変数名を参照してください。これは始めるための方法ですが、IAM Identity Center やその他の一時的な認証情報にできるだけ早く移行することをお勧めします。長期認証情報の使用から移行した後は、必ず `credentials` 共有ファイルからこれらの認証情報を削除してください。

IAM ロールを使用して Amazon EC2 にデプロイされたアプリケーションを認証する

この例では、Amazon Elastic Compute Cloud インスタンスにデプロイされたアプリケーションで使用する Amazon S3 アクセスを持つ AWS Identity and Access Management ロールの設定について説明します。

Amazon Elastic Compute Cloud インスタンスで AWS SDK アプリケーションを実行するには、IAM ロールを作成し、Amazon EC2 インスタンスにそのロールへのアクセス権を付与します。詳細については、Amazon EC2 ユーザーガイドの「Amazon EC2 の IAM ロール」を参照してください。

IAM ロールを作成する

開発する AWS SDK アプリケーションは、アクションを実行 AWS のサービス するために少なくとも 1 つにアクセスする可能性があります。アプリケーションを実行するために必要なアクセス許可を付与する IAM ロールを作成します。

この手順では、例として Amazon S3 への読み取り専用アクセスを付与するロールを作成します。多くの AWS SDK ガイドには、Amazon S3 から読み取る「開始方法」チュートリアルがあります。

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/iam/ で IAM コンソールを開きます。
2. ナビゲーションペインで [ロール]、[ロールを作成] の順に選択します。
3. [信頼されたエンティティタイプ] で [信頼されたエンティティを選択] するため、[AWS のサービス] を選択します。
4. [Use case] (ユースケース) で [Amazon EC2] を選択し、[Next] (次へ) を選択します。
5. [権限の追加] では、ポリシーリストから [Amazon S3 読み取り専用アクセス] のチェックボックスを選択し、[次へ] を選択します。
6. ロールの名前を入力し、[ロールの作成] を選択します。この名前は Amazon EC2 インスタンスを作成するときに必要になるため、忘れないようにしてください。

Amazon EC2 インスタンスを起動して IAM ロールを指定します

IAM ロールを使用して Amazon EC2 インスタンスを作成し、起動するには、次の手順を実行します。

- 「Amazon EC2 ユーザーガイド」の「インスタンスをすばやく起動する」に従います。ただし、最後の送信のステップの前に、以下も実行します。
  - [高度な詳細] の [IAM インスタンスプロファイル] で、前のステップで作成したロールを選択します。

この IAM と Amazon EC2 のセットアップで、Amazon EC2 インスタンスにアプリケーションをデプロイすることができます。これにより、アプリケーションに Amazon S3 サービスへの読み取りアクセスが付与されます。

EC2 インスタンスへの接続

Amazon EC2 インスタンスに接続することで、アプリケーションをそのインスタンスに転送して、アプリケーションを実行できるようにします。インスタンスを作成した際に [キーペア (ログイン)] で使用したキーペアのプライベート部分を含むファイル、すなわち PEM ファイルが必要になります。

これを行うには、お使いのインスタンスタイプ向けのガイダンス、「SSH を使用した Linux インスタンスへの接続」または「RDP を使用した Windows インスタンスへの接続」に従います。接続する際は、開発マシンからインスタンスにファイルを転送できるように接続してください。

> **Note**
>
> Linux または macOS ターミナルでは、セキュアコピーコマンドを使用してアプリケーションをコピーできます。キーペアで scp を使用するには、次のコマンドを使用します。`scp -i path/to/key file/to/copy ec2-user@ec2-xx-xx-xxx-xxx.compute.amazonaws.com:~`
> Windows の詳細については、「RDP を使用して Windows インスタンスにファイルを転送する」を参照してください。

Toolkit を使用している場合は、 AWS Toolkit を使用してインスタンスに接続することもできます。詳細については、 ご利用されている Toolkit の特定のユーザーガイドをご参照ください。

EC2 インスタンスでのアプリケーションの実行

1. ローカルドライブから Amazon EC2 インスタンスにアプリケーションファイルをコピーします。
2. アプリケーションを起動し、開発マシンと同じ実行結果が得られることを確認します。
3. （オプション）アプリケーションで、IAM ロールによって提供されている認証情報が使用されていることを確認します。
    a. にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/ec2/ で Amazon EC2 コンソールを開きます。
    b. インスタンスを選択してください。
    c. [アクション]、[セキュリティ]、[IAM ロールを変更] の順に選択します。
    d. [IAM ロール] では、[IAM ロールなし] を選択して IAM ロールをデタッチします。
    e. [IAM ロールの更新] を選択してください。
    f. アプリケーションを再度実行して、認可エラーが返されることを確認します。

TIP プラグインを使用して にアクセスする AWS のサービス

信頼できる ID 伝達 (TIP) は、 の管理者がグループの関連付けなどのユーザー属性に基づいてアクセス許可 AWS のサービス を付与 AWS IAM アイデンティティセンター できるようにする の機能です。信頼できる ID の伝播により、ID コンテキストが IAM ロールに追加され、 AWS リソースへのアクセスをリクエストしているユーザーを識別します。このコンテキストは他の AWS のサービスに伝達されます。

ID コンテキストは、 がアクセスリクエストを受信したときに承認の決定を行うために AWS のサービス 使用する情報で構成されます。この情報には、リクエスタ (IAM Identity Center ユーザーなど) 、アクセスがリクエスト AWS のサービス される (Amazon Redshift など) 、アクセス範囲 (読み取り専用アクセスなど) を識別するメタデータが含まれます。受信側 AWS のサービス は、このコンテキストとユーザーに割り当てられたアクセス許可を使用して、そのリソースへのアクセスを承認します。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「信頼できる ID の伝播の概要」の「」を参照してください。

TIP プラグインは、信頼できる ID の伝播 AWS のサービス をサポートする で使用できます。リファレンスユースケースとして、「Amazon Q Business User Guide」の「Configuring an Amazon Q Business application using AWS IAM アイデンティティセンター」を参照してください。

> **Note**
>
> Amazon Q Business を使用している場合は、「Configuring an Amazon Q Business application using AWS IAM アイデンティティセンター」でサービス固有の手順を参照してください。

TIP プラグインを使用するための前提条件

プラグインを機能させるには、次のリソースが必要です。

1. AWS SDK for Java または のいずれかを使用する必要があります AWS SDK for JavaScript。
2. 使用しているサービスが信頼できる ID の伝播をサポートしていることを確認します。

   「AWS IAM アイデンティティセンター ユーザーガイド」にある「AWS managed applications that integrate with IAM Identity Center」の表の「Enables trusted identity propagation through IAM Identity Center」列を参照してください。
3. IAM Identity Center と信頼できる ID の伝播を有効にします。

「AWS IAM アイデンティティセンター ユーザーガイド」の「TIP prerequisites and considerations」を参照してください。

4. Identity-Center-integrated アプリケーションが必要です。

   「AWS IAM アイデンティティセンター ユーザーガイド」の「AWS managed applications」または「Customer managed applications」を参照してください。

5. 信頼できるトークン発行者 (TTI) を設定し、サービスを IAM Identity Center に接続する必要があります。

   「AWS IAM アイデンティティセンター ユーザーガイド」の「Prerequisites for trusted token issuers」および「Tasks for setting up a trusted token issuer」を参照してください。

コードで TIP プラグインを使用するには

1. 信頼できる ID の伝播プラグインのインスタンスを作成します。
2. とやり取りするためのサービスクライアントインスタンスを作成し AWS のサービス 、信頼できる ID 伝達プラグインを追加してサービスクライアントをカスタマイズします。

TIP プラグインは以下の入力パラメータを使用します。

- **`webTokenProvider`**: 外部 ID プロバイダーから OpenID トークンを取得するためにお客様が実装する関数。
- **`accessRoleArn`**: ユーザーの ID コンテキストを使用してプラグインが引き受ける IAM ロール ARN。ID 拡張認証情報を取得します。
- **`applicationArn`**: クライアントまたはアプリケーションの一意の識別子文字列。この値は、OAuth 許可が設定されたアプリケーション ARN です。
- **`ssoOidcClient`**: (オプション) `SsoOidcClient` for Java や `client-sso-oidc` for JavaScript など、お客様が定義した設定の SSO OIDC クライアント。指定しない場合、`applicationRoleArn` を使用する OIDC クライアントがインスタンス化されて使用されます。
- **`stsClient`**: (オプション) お客様が定義した設定の AWS STS クライアント。ユーザーの ID コンテキストで `accessRoleArn` を引き受けるために使用されます。指定しない場合、 を使用する AWS STS クライアント`applicationRoleArn`はインスタンス化されて使用されます。

- **applicationRoleArn**: (オプション) OIDC と AWS STS クライアントをブートストラップAssumeRoleWithWebIdentityできるように で引き受ける IAM ロール ARN。
  - 指定しない場合は、ssoOidcClient と stsClient のパラメータの両方を指定する必要があります。
  - 指定した場合、applicationRoleArn は accessRoleArn のパラメータと同じ値にすることはできません。 applicationRoleArn は accessRole を引き受けるために使用される stsClient の構築に使用されます。applicationRole と の両方に同じロールが使用されている場合accessRole、ロールを使用してそれ自体を引き受ける (自己ロールの引き受け) ことを意味します。これは推奨されません AWS。詳細については、お知らせを参照してください。

ssoOidcClient、stsClient、および applicationRoleArn のパラメータに関する考慮事項

TIP プラグインを設定するときは、指定するパラメータに基づいて、次のアクセス許可要件を考慮してください。

- ssoOidcClient と stsClient を指定する場合:
  - ssoOidcClient の認証情報には、アイデンティティセンターを呼び出してアイデンティティセンター固有のユーザーコンテキストを取得するための oauth:CreateTokenWithIAM アクセス許可が必要です。
  - stsClient の認証情報には、sts:AssumeRole と、accessRole の sts:SetContext のアクセス許可が必要です。また、accessRole には、stsClient の認証情報との信頼関係を構成する必要があります。
- applicationRoleArn を指定する場合:
  - applicationRole は OIDC および STS クライアントの構築に使用されるため、必要なリソース (IdC インスタンス、accessRole) での oauth:CreateTokenWithIAM、sts:AssumeRole、および sts:SetContext のアクセス許可が必要です。
  - webToken はプラグインによる AssumeRoleWithWebIdentity 呼び出しを介して applicationRole を引き受けるために使用されるため、applicationRole は、webToken の生成に使用される ID プロバイダーとの信頼関係が必要です。

ApplicationRole 構成の例:

ウェブトークンプロバイダーとの信頼ポリシー:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::ACCOUNT_ID:oidc-provider/
IDENTITY_PROVIDER_URL"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "IDENTITY_PROVIDER_URL:aud": "CLIENT_ID_TO_BE_TRUSTED"
                }
            }
        }
    ]
}
```

アクセス許可ポリシー:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRole",
                "sts:SetContext"
            ],
            "Resource": [
                "accessRoleArn"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "sso-oauth:CreateTokenWithIAM"
            ],
            "Resource": [
                "*"
            ]
        }
```

```
    ]
}
```

TIP を使用したコードの例

以下の例は、 AWS SDK for Java または を使用してコードに TIP プラグインを実装する方法を示しています AWS SDK for JavaScript。

Java

AWS SDK for Java プロジェクトで TIP プラグインを使用するには、プロジェクトの pom.xml ファイルで TIP プラグインを依存関係として宣言する必要があります。

```
<dependency>
<groupId>software.amazon.awsidentity.trustedIdentityPropagation</groupId>
<artifactId>aws-sdk-java-trustedIdentityPropagation-java-plugin</artifactId>
   <version>2.0.0</version>
</dependency>
```

ソースコードに、software.amazon.awssdk.trustedidentitypropagation に必要なパッケージステートメントを含めます。

次の例は、信頼できる ID の伝播プラグインのインスタンスを作成し、サービスクライアントに追加する 2 つの方法を示しています。どちらの例も Amazon S3 をサービスとして使用し、S3AccessGrantsPluginを使用してユーザー固有のアクセス許可を管理しますが、信頼できる ID 伝達 (TIP) AWS のサービス をサポートする任意の に適用できます。

> **Note**
>
> これらの例では、S3 Access Grants からユーザー固有のアクセス許可を設定する必要があります。詳細については、S3 Access Grants のドキュメントを参照してください。

オプション 1: OIDC クライアントと STS クライアントを構築して渡す

```
SsoOidcClient oidcClient = SsoOidcClient.builder()
    .region(Region.US_EAST_1)
    .credentialsProvider(credentialsProvider).build();

StsClient stsClient = StsClient.builder()
```

```
    .region(Region.US_EAST_1)
    .credentialsProvider(credentialsProvider).build();

TrustedIdentityPropagationPlugin trustedIdentityPropagationPlugin =
 TrustedIdentityPropagationPlugin.builder()
        .webTokenProvider(() -> webToken)
        .applicationArn(idcApplicationArn)
        .accessRoleArn(accessRoleArn)
        .ssoOidcClient(oidcClient)
        .stsClient(stsClient)
        .build();

S3AccessGrantsPlugin accessGrantsPlugin = S3AccessGrantsPlugin.builder()
        .build();

S3Client s3Client =
        S3Client.builder().region(Region.US_EAST_1)
                .crossRegionAccessEnabled(true)
                .addPlugin(trustedIdentityPropagationPlugin)
                .addPlugin(accessGrantsPlugin)
                .build();

final var resp = s3Client.getObject(GetObjectRequest.builder()
        .key("path/to/object/fileName")
        .bucket("bucketName")
        .build());
```

オプション 2: applicationRoleArn を渡してクライアント作成をプラグインに任せる

```
TrustedIdentityPropagationPlugin trustedIdentityPropagationPlugin =
 TrustedIdentityPropagationPlugin.builder()
        .webTokenProvider(() -> webToken)
        .applicationArn(idcApplicationArn)
        .accessRoleArn(accessRoleArn)
        .applicationRoleArn(applicationRoleArn)
        .build();

S3AccessGrantsPlugin accessGrantsPlugin = S3AccessGrantsPlugin.builder()
        .build();

S3Client s3Client =
        S3Client.builder().region(Region.US_EAST_1)
                .crossRegionAccessEnabled(true)
```

```
                .addPlugin(trustedIdentityPropagationPlugin)
                .addPlugin(accessGrantsPlugin)
                .build();

final var resp = s3Client.getObject(GetObjectRequest.builder()
        .key("path/to/object/fileName")
        .bucket("bucketName")
        .build());
```

詳細とソースについては、GitHub の「trusted-identity-propagation-java」を参照してください。

JavaScript

次のコマンドを実行して、TIP 認証プラグインパッケージを AWS SDK for JavaScript プロジェクトにインストールします。

```
$  npm i @aws-sdk-extension/trusted-identity-propagation
```

最後の `package.json` には、次のような依存関係を含める必要があります。

```
  "dependencies": {
"@aws-sdk-extension/trusted-identity-propagation": "^2.0.0"
  },
```

ソースコードで、必要な `TrustedIdentityPropagationExtension` 依存関係をインポートします。

次の例は、信頼できる ID の伝播プラグインのインスタンスを作成し、サービスクライアントに追加する 2 つの方法を示しています。どちらの例も Amazon S3 をサービスとして使用し、Amazon S3 Access Grants を使用してユーザー固有のアクセス許可を管理しますが、信頼できる ID 伝達 (TIP) AWS のサービス をサポートする任意の に適用できます。

> **Note**
>
> これらの例については、Amazon S3 Access Grants からユーザー固有のアクセス許可を設定する必要があります。詳細については、S3 Access Grants のドキュメントを参照してください。

オプション 1: OIDC クライアントと STS クライアントを構築して渡す

```
import { S3Client, GetObjectCommand } from "@aws-sdk/client-s3";
import { S3ControlClient, GetDataAccessCommand } from "@aws-sdk/client-s3-control";
import { TrustedIdentityPropagationExtension } from "@aws-sdk-extension/trusted-
identity-propagation";

const s3ControlClient = new S3ControlClient({
    region: "us-east-1",
    extensions: [
        TrustedIdentityPropagationExtension.create({
            webTokenProvider: async () => {
                return 'ID_TOKEN_FROM_YOUR_IDENTITY_PROVIDER';
            },
            ssoOidcClient: customOidcClient,
            stsClient: customStsClient,
            accessRoleArn: accessRoleArn,
            applicationArn: applicationArn,
        }),
    ],
});

const getDataAccessParams = {
  Target: "S3_URI_PATH",
  Permission: "READ",
  AccountId: ACCOUNT_ID,
  InstanceArn: S3_ACCESS_GRANTS_ARN,
  TargetType: "Object",
};

try {
  const command = new GetDataAccessCommand(getDataAccessParams);
  const response = await s3ControlClient.send(command);

  const credentials = response.Credentials;

  // Create a new S3 client with the temporary credentials
  const temporaryS3Client = new S3Client({
    region: "us-east-1",
    credentials: {
      accessKeyId: credentials.AccessKeyId,
      secretAccessKey: credentials.SecretAccessKey,
      sessionToken: credentials.SessionToken,
    },
  });
```

```
  // Use the temporary S3 client to perform the operation
  const s3Params = {
    Bucket: "BUCKET_NAME",
    Key: "S3_OBJECT_KEY",
  };
  const getObjectCommand = new GetObjectCommand(s3Params);
  const s3Object = await temporaryS3Client.send(getObjectCommand);

  const fileContent = await s3Object.Body.transformToString();

  // Process the S3 object data
  console.log("Successfully retrieved S3 object:", fileContent);
} catch (error) {
  console.error("Error accessing S3 data:", error);
}
```

オプション 2: applicationRoleArn を渡してクライアント作成をプラグインに任せる

```
import { S3Client, GetObjectCommand } from "@aws-sdk/client-s3";
import { S3ControlClient, GetDataAccessCommand } from "@aws-sdk/client-s3-control";
import { TrustedIdentityPropagationExtension } from "@aws-sdk-extension/trusted-
identity-propagation";

const s3ControlClient = new S3ControlClient({
    region: "us-east-1",
    extensions: [
        TrustedIdentityPropagationExtension.create({
            webTokenProvider: async () => {
                return 'ID_TOKEN_FROM_YOUR_IDENTITY_PROVIDER';
            },
            accessRoleArn: accessRoleArn,
            applicationRoleArn: applicationRoleArn,
            applicationArn: applicationArn,
        }),
    ],
});

// Same S3 AccessGrants workflow as Option 1
const getDataAccessParams = {
  Target: "S3_URI_PATH",
  Permission: "READ",
  AccountId: ACCOUNT_ID,
```

```
  InstanceArn: S3_ACCESS_GRANTS_ARN,
  TargetType: "Object",
};

try {
  const command = new GetDataAccessCommand(getDataAccessParams);
  const response = await s3ControlClient.send(command);

  const credentials = response.Credentials;

  const temporaryS3Client = new S3Client({
    region: "us-east-1",
    credentials: {
      accessKeyId: credentials.AccessKeyId,
      secretAccessKey: credentials.SecretAccessKey,
      sessionToken: credentials.SessionToken,
    },
  });

  const s3Params = {
    Bucket: "BUCKET_NAME",
    Key: "S3_OBJECT_KEY",
  };
  const getObjectCommand = new GetObjectCommand(s3Params);
  const s3Object = await temporaryS3Client.send(getObjectCommand);

  const fileContent = await s3Object.Body.transformToString();

  console.log("Successfully retrieved S3 object:", fileContent);
} catch (error) {
  console.error("Error accessing S3 data:", error);
}
```

詳細とソースについては、GitHub の「trusted-identity-propagation-js」を参照してください。

AWS SDKsとツールの設定リファレンス

SDKsは、言語固有の APIsを提供します AWS のサービス。認証、再試行動作など、API コールを正常に行うために必要な面倒な作業の一部はこれらによって処理されます。そのために、SDK にはリクエストに使用する認証情報の取得、各サービスで使用する設定の管理、グローバル設定に使用する値の取得といった柔軟な戦略があります。

設定の詳細については、以下のセクションを参照してください。

- AWS SDKs標準化された認証情報プロバイダー – 複数の SDK で標準化された共通の認証情報プロバイダー。
- AWS SDKs標準化された機能 – 複数の SDK で標準化された共通機能。

サービスクライアントの作成

プログラムで にアクセスするために AWS のサービス、SDKsはそれぞれのクライアントクラス/オブジェクトを使用します AWS のサービス。たとえば、アプリケーションが Amazon EC2 にアクセスする必要がある場合、アプリケーションはそのサービスとインターフェイスをとる Amazon EC2 クライアントオブジェクトを作成します。次に、サービスクライアントを使用して、その AWS のサービスに対してリクエストを実行します。ほとんどの SDK ではサービスクライアントオブジェクトはイミュータブルであるため、リクエストを実行する各サービスについて、または異なった設定を使用する同じサービスにリクエストを実行するために、新しいクライアントを作成する必要があります。

設定の優先順位

グローバル設定は、ほとんどの SDK でサポートされ、 AWS のサービス全体に幅広く影響する機能、認証情報プロバイダー、およびその他の機能を設定します。すべての SDK には、グローバル設定の値を見つけるための一連の場所 (またはソース) があります。設定検索の優先順位は次のとおりです。

1. コードまたはサービスクライアント自体に設定されている明示的な設定は、他の設定よりも優先されます。
   - 一部の設定はオペレーションごとに設定でき、呼び出すオペレーションごとに必要に応じて変更できます。 AWS CLI または の場合 AWS Tools for PowerShell、これらはコマンドラインに入力したオペレーションごとのパラメータの形式になります。SDK の場合、明示的な割り当て

は、 AWS のサービス クライアントまたは設定オブジェクトをインスタンス化するとき、または個々の API を呼び出すときに設定したパラメータの形式になります。

2. Java/Kotlin のみ: 設定の JVM システムプロパティがチェックされます。設定されている場合は、その値を使用してクライアントが設定されます。
3. 環境変数が確認されます。設定されている場合は、その値を使用してクライアントが設定されます。
4. SDK は、設定の共有 `credentials` ファイルを確認します。設定されている場合、クライアントはそれを使用します。
5. 設定の共有 `config` ファイル。設定が存在する場合、SDK はその設定を使用します。
   - `AWS_PROFILE` 環境変数または `aws.profile` システムプロパティを使用して、どのプロファイルが SDK によってロードされるかを指定できます。
6. SDK ソースコード自体が提供するデフォルト値が最後に使用されます。

> **Note**
>
> SDK やツールによっては、チェックの順序が異なる場合があります。また、SDK やツールの中には、他の方法でパラメータを保存したり取得したりできるものもあります。たとえば、 は SDK ストアと呼ばれる追加のソース AWS SDK for .NET をサポートしています。SDK またはツールにのみ存在するプロバイダーについて詳しくは、使用している SDK またはツールの特定のガイドを参照してください。

順序によって、どのメソッドが優先され、他のメソッドをオーバーライドするかが決まります。たとえば、共有 `config` ファイルにプロファイルを設定した場合、そのプロファイルは SDK またはツールが最初に他の場所を確認した後にのみ検出され、使用されます。つまり、`credentials` ファイルに設定を入力すると、`config` ファイルにある設定の代わりにその設定が使用されます。環境変数に設定と値を設定すると、`credentials` と `config` ファイルの両方の設定がオーバーライドされます。最後に、個々のオペレーション (AWS CLI コマンドラインパラメータまたは API パラメータ) またはコード内の設定は、その 1 つのコマンドの他のすべての値をオーバーライドします。

このガイドの設定ページについて

このガイドの「設定リファレンス」セクションのページには、さまざまなメカニズムで設定できる利用可能な設定の詳細が記載されています。次の表は、設定と認証情報のファイル設定、環境変数、お

よび (Java と Kotlin SDK の場合) 機能を設定するためにコードの外部で使用できる JVM 設定を示しています。各リストのリンクされたそれぞれのトピックから、対応する設定ページに移動できます。

- Config ファイル設定リスト
- Credentials ファイル設定リスト
- 環境変数の一覧
- JVM システムプロパティリスト

それぞれの認証情報プロバイダーまたは機能には、その機能の設定に使用される設定が一覧表示されるページがあります。それぞれの設定では多くの場合、設定を設定ファイルに追加するか、環境変数を設定するか、または Java および Kotlin の場合は JVM システムプロパティを設定して 値を指定できます。それぞれの設定には、説明の詳細の上にあるブロックで値を設定するためにサポートされているすべてのメソッドが一覧表示されます。優先順位は異なりますが、結果の機能は設定方法に関係なく同じになりす。

説明には、何もしない場合に有効になるデフォルト値 (ある場合) も含まれています。また、その設定の有効な値も定義します。

例として、「リクエスト圧縮」機能のページから設定を見てみましょう。

「disable_request_compression」の例の設定の情報には、以下が記載されています。

- コードベース外でリクエスト圧縮を制御するには、3 つの同等の方法があります。次のいずれかを行うことができます。
  - disable_request_compression を使用して設定ファイルで設定する
  - AWS_DISABLE_REQUEST_COMPRESSION を使用して環境変数として設定する
  - aws.disableRequestCompression を使用して JVM システムプロパティとして設定する (Java または Kotlin SDK を使用している場合)

  > **Note**
  >
  > また、コード内で同じ機能を直接設定する方法もありますが、その方法は各 SDK に固有であるため、このリファレンスでは取り上げていません。コード自体で設定する場合は、固有の SDK ガイドまたは API リファレンスを参照してください。

- 何もしない場合、値はデフォルトで false になります。
- このブール設定の有効な値は、true と false のみです。

各機能ページの下部には、 AWS SDKsとツールの表があります。

この表は、SDK がページにリストされている設定をサポートしているかどうかを示しています。Supported 列は、次の値を使用してサポートレベルを示します。

- Yes – 設定は、記述されたとおりに SDK によって完全にサポートされています。
- Partial – 一部の設定がサポートされているか、動作が説明とは異なります。Partial の場合、その違いを示す補足説明があります。
- No – どの設定もサポートされていません。これは、コードで同じ機能を達成できるかどうかについては主張していません。リストされている外部構成設定がサポートされていないことを示しているだけです。

Config ファイル設定リスト

次の表に示す設定は、共有 AWS configファイルで割り当てることができます。これらはグローバルで、すべての AWS のサービスに影響します。また、SDK とツールは、一意の設定と環境変数をサポートする場合もあります。個々の SDK またはツールでのみサポートされている設定と環境変数を確認するには、その SDK またはツール固有のガイドを参照してください。

設定名	Details	
account_id_endpoint_mode	アカウントベースのエンドポイント	
api_versions	一般設定	
auth_scheme_preference	認証スキーム	
aws_access_key_id	AWS アクセスキー	
aws_account_id	アカウントベースのエンドポイント	
aws_secret_access_key	AWS アクセスキー	

設定名	Details
aws_session_token	AWS アクセスキー
ca_bundle	一般設定
credential_process	プロセス認証情報プロバイダー
credential_source	ロール認証情報プロバイダーを引き受けます
defaults_mode	スマート設定デフォルト
disable_host_prefix_injection	ホストプレフィックスインジェクション
disable_request_compression	リクエスト圧縮
duration_seconds	ロール認証情報プロバイダーを引き受けます
ec2_metadata_service_endpoint	IMDS 認証情報プロバイダー
ec2_metadata_service_endpoint_mode	IMDS 認証情報プロバイダー
ec2_metadata_v1_disabled	IMDS 認証情報プロバイダー

設定名	Details	
endpoint_discovery_enabled	エンドポイント検出	
endpoint_url	サービス固有のエンドポイント	
external_id	ロール認証情報プロバイダーを引き受けます	
ignore_configured_endpoint_urls	サービス固有のエンドポイント	
max_attempts	再試行動作	
metadata_service_num_attempts	Amazon EC2 インスタンスメタデータ	
metadata_service_timeout	Amazon EC2 インスタンスメタデータ	
mfa_serial	ロール認証情報プロバイダーを引き受けます	
output	一般設定	
parameter_validation	一般設定	
region	AWS リージョン	
request_checksum_calculation	Amazon S3 のデータ整合性保護	

設定名	Details	
request_min_compression_size_bytes	リクエスト圧縮	
response_checksum_validation	Amazon S3 のデータ整合性保護	
retry_mode	再試行動作	
role_arn	ロール認証情報プロバイダーを引き受けます	
role_session_name	ロール認証情報プロバイダーを引き受けます	
s3_disable_express_session_auth	S3 Express One Zone セッション認証	
s3_disable_multiregion_access_points	Amazon S3 マルチリージョンアクセスポイント	
s3_use_arn_region	Amazon S3 アクセスポイント	
sdk_ua_app_id	アプリケーション ID	
sigv4a_signing_region_set	認証スキーム	
source_profile	ロール認証情報プロバイダーを引き受けます	
sso_account_id	IAM Identity Center 認証情報プロバイダー	

設定名	Details
sso_region	IAM Identity Center 認証情報プロバイダー
sso_regis tration_scopes	IAM Identity Center 認証情報プロバイダー
sso_role_name	IAM Identity Center 認証情報プロバイダー
sso_start_url	IAM Identity Center 認証情報プロバイダー
sts_regio nal_endpoints	AWS STS リージョンエンドポイント
use_duals tack_endpoint	デュアルスタックと FIPS エンドポイント
use_fips_ endpoint	デュアルスタックと FIPS エンドポイント
web_ident ity_token_file	ロール認証情報プロバイダーを引き受けます

Credentials ファイル設定リスト

次の表に示す設定は、共有 AWS credentialsファイルで割り当てることができます。これらはグローバルで、すべての AWS のサービスに影響します。また、SDK とツールは、一意の設定と環境変数をサポートする場合もあります。個々の SDK またはツールでのみサポートされている設定と環境変数を確認するには、その SDK またはツール固有のガイドを参照してください。

設定名	Details
aws_acces s_key_id	AWS アクセスキー
aws_secre t_access_key	AWS アクセスキー

設定名	Details
aws_sessi on_token	AWS アクセスキー

環境変数の一覧

ほとんどの SDK でサポートされる環境変数は、以下の表に示されています。これらはグローバルで、すべての AWS のサービスに影響します。また、SDK とツールは、一意の設定と環境変数をサポートする場合もあります。個々の SDK またはツールでのみサポートされている設定と環境変数を確認するには、その SDK またはツール固有のガイドを参照してください。

設定名	Details
AWS_ACCES S_KEY_ID	AWS アクセスキー
AWS_ACCOUNT_ID	アカウントベースのエンドポイント
AWS_ACCOU NT_ID_END POINT_MODE	アカウントベースのエンドポイント
AWS_AUTH_ SCHEME_PR EFERENCE	認証スキーム
AWS_CA_BUNDLE	一般設定
AWS_CONFIG_FILE	AWS SDKs とツールの共有configファイルとcredentials ファイルの場所の検索と変更
AWS_CONTA INER_AUTH ORIZATION _TOKEN	コンテナ認証情報プロバイダー

設定名	Details	
AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE	コンテナ認証情報プロバイダー	
AWS_CONTAINER_CREDENTIALS_FULL_URI	コンテナ認証情報プロバイダー	
AWS_CONTAINER_CREDENTIALS_RELATIVE_URI	コンテナ認証情報プロバイダー	
AWS_DEFAULTS_MODE	スマート設定デフォルト	
AWS_DISABLE_HOST_PREFIX_INJECTION	ホストプレフィックスインジェクション	
AWS_DISABLE_REQUEST_COMPRESSION	リクエスト圧縮	
AWS_EC2_METADATA_DISABLED	IMDS 認証情報プロバイダー	
AWS_EC2_METADATA_SERVICE_ENDPOINT	IMDS 認証情報プロバイダー	

設定名	Details	
AWS_EC2_METADATA_SERVICE_ENDPOINT_MODE	IMDS 認証情報プロバイダー	
AWS_EC2_METADATA_V1_DISABLED	IMDS 認証情報プロバイダー	
AWS_ENABLE_ENDPOINT_DISCOVERY	エンドポイント検出	
AWS_ENDPOINT_URL	サービス固有のエンドポイント	
AWS_ENDPOINT_URL_<SERVICE>	サービス固有のエンドポイント	
AWS_IGNORE_CONFIGURED_ENDPOINT_URLS	サービス固有のエンドポイント	
AWS_MAX_ATTEMPTS	再試行動作	
AWS_METADATA_SERVICE_NUM_ATTEMPTS	Amazon EC2 インスタンスメタデータ	
AWS_METADATA_SERVICE_TIMEOUT	Amazon EC2 インスタンスメタデータ	

設定名	Details
AWS_PROFILE	共有ファイルconfigと credentials ファイルを使用して AWS SDKs とツールをグローバルに設定する
AWS_REGION	AWS リージョン
AWS_REQUEST_CHECKSUM_CALCULATION	Amazon S3 のデータ整合性保護
AWS_REQUEST_MIN_COMPRESSION_SIZE_BYTES	リクエスト圧縮
AWS_RESPONSE_CHECKSUM_VALIDATION	Amazon S3 のデータ整合性保護
AWS_RETRY_MODE	再試行動作
AWS_ROLE_ARN	ロール認証情報プロバイダーを引き受けます
AWS_ROLE_SESSION_NAME	ロール認証情報プロバイダーを引き受けます
AWS_S3_DISABLE_EXPRESS_SESSION_AUTH	S3 Express One Zone セッション認証
AWS_S3_DISABLE_MULTIREGION_ACCESS_POINTS	Amazon S3 マルチリージョンアクセスポイント

設定名	Details
AWS_S3_USE_ARN_REGION	Amazon S3 アクセスポイント
AWS_SDK_UA_APP_ID	アプリケーション ID
AWS_SECRET_ACCESS_KEY	AWS アクセスキー
AWS_SESSION_TOKEN	AWS アクセスキー
AWS_SHARED_CREDENTIALS_FILE	AWS SDKs とツールの共有configファイルとcredentials ファイルの場所の検索と変更
AWS_SIGV4A_SIGNING_REGION_SET	認証スキーム
AWS_STS_REGIONAL_ENDPOINTS	AWS STS リージョンエンドポイント
AWS_USE_DUALSTACK_ENDPOINT	デュアルスタックと FIPS エンドポイント
AWS_USE_FIPS_ENDPOINT	デュアルスタックと FIPS エンドポイント
AWS_WEB_IDENTITY_TOKEN_FILE	ロール認証情報プロバイダーを引き受けます

JVM システムプロパティリスト

AWS SDK for Java および AWS SDK for Kotlin (JVM をターゲットとする) には、次の JVM システムプロパティを使用できます。JVM システムプロパティを設定する方法については、「the section called “JVM システムプロパティを設定する方法”」を参照してください。

設定名	Details	
aws.accessKeyId	AWS アクセスキー	
aws.accountId	アカウントベースのエンドポイント	
aws.accountIdEndpointMode	アカウントベースのエンドポイント	
aws.authSchemePreference	認証スキーム	
aws.configFile	AWS SDKs とツールの共有configファイルとcredentials ファイルの場所の検索と変更	
aws.defaultsMode	スマート設定デフォルト	
aws.disableEc2MetadataV1	IMDS 認証情報プロバイダー	
aws.disableHostPrefixInjection	ホストプレフィックスインジェクション	
aws.disableRequestCompression	リクエスト圧縮	

設定名	Details	
aws.disableS3ExpressAuth	S3 Express One Zone セッション認証	
aws.ec2MetadataServiceEndpoint	IMDS 認証情報プロバイダー	
aws.ec2MetadataServiceEndpointMode	IMDS 認証情報プロバイダー	
aws.endpointDiscoveryEnabled	エンドポイント検出	
aws.endpointUrl	サービス固有のエンドポイント	
aws.endpointUrl<ServiceName>	サービス固有のエンドポイント	
aws.ignoreConfiguredEndpointUrls	サービス固有のエンドポイント	
aws.maxAttempts	再試行動作	
aws.profile	共有ファイルconfigと credentials ファイルを使用して AWS SDKs とツールをグローバルに設定する	
aws.region	AWS リージョン	

設定名	Details
aws.requestChecksumCalculation	Amazon S3 のデータ整合性保護
aws.requestMinCompressionSizeBytes	リクエスト圧縮
aws.responseChecksumValidation	Amazon S3 のデータ整合性保護
aws.retryMode	再試行動作
aws.roleArn	ロール認証情報プロバイダーを引き受けます
aws.roleSessionName	ロール認証情報プロバイダーを引き受けます
aws.s3DisableMultiRegionAccessPoints	Amazon S3 マルチリージョンアクセスポイント
aws.s3UseArnRegion	Amazon S3 アクセスポイント
aws.secretAccessKey	AWS アクセスキー
aws.sessionToken	AWS アクセスキー
aws.sharedCredentialsFile	AWS SDKs とツールの共有configファイルとcredentials ファイルの場所の検索と変更

設定名	Details
aws.useDualstackEndpoint	デュアルスタックと FIPS エンドポイント
aws.useFipsEndpoint	デュアルスタックと FIPS エンドポイント
aws.webIdentityTokenFile	ロール認証情報プロバイダーを引き受けます
sdk.ua.appId	アプリケーション ID

AWS SDKs標準化された認証情報プロバイダー

多くの認証情報プロバイダーは、一貫したデフォルト値に標準化されており、多くの SDK で同じように動作します。この一貫性により、複数の SDK 間のコーディングの生産性と明確性が向上します。すべての設定はコード内で上書きすることができます。詳細については、お使いの特定の SDK API を参照してください。

> **Important**
>
> すべての SDK がすべてのプロバイダーをサポートしているわけではなく、プロバイダー内のあらゆる側面をサポートしているわけでもありません。

トピック

- 認証情報プロバイダーチェーンを理解する
- SDK 固有およびツール固有の認証情報プロバイダーチェーン
- AWS アクセスキー
- ログイン認証情報プロバイダー
- ロール認証情報プロバイダーを引き受けます
- コンテナ認証情報プロバイダー
- IAM Identity Center 認証情報プロバイダー

- IMDS 認証情報プロバイダー
- プロセス認証情報プロバイダー

認証情報プロバイダーチェーンを理解する

すべての SDK には、 AWS のサービスへのリクエストに使用する有効な認証情報を確認するための一連の場所 (またはソース) があります。有効な認証情報が見つかると、検索は停止されます。この体系的な検索は、認証情報プロバイダーチェーンと呼ばれます。

標準化された認証情報プロバイダーのいずれかを使用する場合、 AWS SDKs有効期限が切れると認証情報を自動的に更新しようとします。組み込みの認証情報プロバイダーチェーンは、チェーンで使用しているプロバイダーに関係なく、アプリケーションが認証情報を更新できるようにします。SDK がこれを行うのに追加のコードは必要ありません。

SDK によって使用される個別のチェーンは異なりますが、ほとんどの場合、次のようなソースが含まれます。

認証情報プロバイダー	説明
AWS アクセスキー	AWS IAM ユーザーの アクセスキー (`AWS_ACCESS_KEY_ID` 、、 など`AWS_SECRET_ACCESS_KEY`)。
ウェブアイデンティティまたは OpenID Connect でのフェデレーション - ロール認証情報プロバイダーを引き受けます	よく知られている外部 ID プロバイダー (IdP) (例: Login with Amazon、Facebook、Google などの OpenID Connect (OIDC) 互換の IdP) を使用してサインインします。 AWS Security Token Service () の JSON ウェブトークン (JWT) を使用して IAM ロールのアクセス許可を引き受けますAWS STS。
ログイン認証情報プロバイダー	ログインしている新規または既存のコンソールセッションの認証情報を取得します。
IAM Identity Center 認証情報プロバイダー	から認証情報を取得します AWS IAM アイデンティティセンター。
ロール認証情報プロバイダーを引き受けます	IAM ロールのアクセス許可を引き受けることで、他のリソースにアクセスできます。 (ロールの一時認証情報を取得して使用します) 。

認証情報プロバイダー	説明
コンテナ認証情報プロバイダー	Amazon Elastic Container Service (Amazon ECS) および Amazon Elastic Kubernetes Service (Amazon EKS) の認証情報。コンテナ認証情報プロバイダーは、お客様のコンテナ化されたアプリケーションの認証情報を取得します。
プロセス認証情報プロバイダー	カスタム認証情報プロバイダー。認証情報を外部ソースまたはプロセス (IAM Roles Anywhere) から取得します。
IMDS 認証情報プロバイダー	Amazon Elastic Compute Cloud (Amazon EC2) インスタンスプロファイルの認証情報。IAM ロールを各 EC2 インスタンスに関連付けます。そのロールの一時認証情報は、インスタンスで実行中のコードで使用できるようになります。認証情報は、Amazon EC2 メタデータサービスを通じて配信されます。

チェーン内の各ステップには、設定値を割り当てる方法が複数あります。コードで指定されている設定値が常に優先されます。ただし、環境変数 と 共有ファイルconfigと credentials ファイルを使用して AWS SDKs とツールをグローバルに設定する もあります。詳細については、「設定の優先順位」を参照してください。

SDK 固有およびツール固有の認証情報プロバイダーチェーン

SDK またはツールに固有の認証情報プロバイダーチェーンの詳細に直接アクセスするには、以下から SDK またはツールを選択します。

- AWS CLI
- SDK for C++
- SDK for Go
- SDK for Java
- SDK for JavaScript
- SDK for Kotlin
- SDK for .NET
- SDK for PHP

- SDK for Python (Boto3)
- SDK for Ruby
- SDK for Rust
- SDK for Swift
- Tools for PowerShell

AWS アクセスキー

> **Warning**
>
> セキュリティリスクを避けるため、専用ソフトウェアを開発するときや実際のデータを扱うときは、IAM ユーザーを認証に使用しないでください。代わりに、AWS IAM アイデンティティセンター などの ID プロバイダーとのフェデレーションを使用してください。

AWS IAM ユーザーの アクセスキーを AWS 認証情報として使用できます。 AWS SDK は自動的にこれらの AWS 認証情報を使用して API リクエストに署名するため AWS、ワークロードは AWS リソースとデータに安全かつ便利にアクセスできます。認証情報が一時的なもので、有効期限が切れると無効になるように、常に `aws_session_token` を使用することをおすすめします。長期の認証情報を使用することはお勧めしません。

> **Note**
>
> AWS がこれらの一時的な認証情報を更新できない場合、ワークロードに影響が及ばないように認証情報の有効性 AWS を拡張できます。

共有 AWS `credentials`ファイルは、アプリケーションソースディレクトリの外部で安全であり、共有`config`ファイルの SDK 固有の設定とは別のものであるため、認証情報を保存するために推奨される場所です。

AWS 認証情報とアクセスキーの使用の詳細については、IAM ユーザーガイドのAWS 「セキュリティ認証情報」と「IAM ユーザーのアクセスキーの管理」を参照してください。

この機能を設定するには、以下のように使用します。

aws_access_key_id - 共有 AWS **config**ファイル設定, **aws_access_key_id** - 共有 AWS **credentials**ファイル設定 (推奨メソッド), **AWS_ACCESS_KEY_ID** - 環境変数, **aws.accessKeyId** - JVM システムプロパティ: Java/Kotlin のみ

ユーザーを認証するための認証情報の一部として使用される AWS アクセスキーを指定します。

aws_secret_access_key - 共有 AWS **config**ファイル設定, **aws_secret_access_key** - 共有 AWS **credentials**ファイル設定 (推奨メソッド), **AWS_SECRET_ACCESS_KEY** - 環境変数, **aws.secretAccessKey** - JVM システムプロパティ: Java/Kotlin のみ

ユーザーを認証するための認証情報の一部として使用される AWS シークレットキーを指定します。

aws_session_token - 共有 AWS **config**ファイル設定, **aws_session_token** - 共有 AWS **credentials**ファイル設定 (推奨メソッド), **AWS_SESSION_TOKEN** - 環境変数, **aws.sessionToken** - JVM システムプロパティ: Java/Kotlin のみ

ユーザーを認証するための認証情報の一部として使用される AWS セッショントークンを指定します。この値は、ロールを引き受けるリクエストが正常に終了すると返される一時的な認証情報の一部として受け取ります。セッショントークンは、一時的なセキュリティ認証情報を手動で指定する場合にのみ必要です。ただし、長期の認証情報ではなく、一時的なセキュリティ認証情報を常に使用することをお勧めします。セキュリティの推奨事項については、「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。

これらの値を取得する方法については、「短期認証情報を使用して AWS SDKsとツールを認証する」を参照してください。

config または credentials ファイルに必要な値を設定する例を以下に示します。

```
[default]
aws_access_key_id = AKIAIOSFODNN7EXAMPLE
aws_secret_access_key = wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
aws_session_token = AQoEXAMPLEH4aoAH0gNCAPy...truncated...zrkuWJOgQs8IZZaIv2BXIa2R4Olgk
```

Linux/macOS のコマンドラインによる環境変数の設定の例を以下に示します。

```
export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
export
 AWS_SESSION_TOKEN=AQoEXAMPLEH4aoAH0gNCAPy...truncated...zrkuWJOgQs8IZZaIv2BXIa2R4Olgk
```

Windows のコマンドラインによる環境変数の設定の例を以下に示します。

```
setx AWS_ACCESS_KEY_ID AKIAIOSFODNN7EXAMPLE
setx AWS_SECRET_ACCESS_KEY wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
setx
 AWS_SESSION_TOKEN AQoEXAMPLEH4aoAH0gNCAPy...truncated...zrkuWJOgQs8IZZaIv2BXIa2R4Olgk
```

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	はい	
SDK for C++	はい	共有 config ファイルはサポートされていません。
SDK for Go V2 (1.x)	はい	
SDK for Go 1.x (V1)	はい	共有 config ファイル設定を使用するには、設定ファイルからの読み込みを有効にする必要があります。「セッション」を参照してください。
SDK for Java 2.x	はい	
SDK for Java 1.x	はい	
SDK for JavaScript 3.x	はい	

SDK	サポート	注意または詳細情報
SDK for JavaScript 2.x	はい	
SDK for Kotlin	はい	
SDK for .NET 4.x	はい	
SDK for .NET 3.x	はい	
SDK for PHP 3.x	はい	
SDK for Python (Boto3)	はい	
SDK for Ruby 3.x	はい	
SDK for Rust	はい	
SDK for Swift	はい	
PowerShell V5 用のツール	はい	
PowerShell V4 のツール	はい	環境変数はサポートされていません。

ログイン認証情報プロバイダー

既存の AWS マネジメントコンソールのサインイン認証情報を使用して、プログラムによるアクセスに使用できる短期認証情報を取得できます。ブラウザベースの認証フローを完了すると、 は AWS CLI、 AWS Tools for PowerShell、 AWS SDKsなどのローカル開発ツールで動作する一時的な認証情報 AWS を生成します。

これらの認証情報を生成するには、CLI AWS で `aws login` コマンドを実行するか、 AWS Tools for PowerShell `Invoke-AWSLogin` で コマンドレットを実行します。生成された短期認証情報はローカルにキャッシュされ、 AWS SDKs で再利用できます。短期認証情報は 15 分で期限切れになりますが、必要に応じて CLI と SDKs によって最大 12 時間自動的に更新されます。更新トークンの有効期限が切れると、CLI または PowerShell を使用して再度ログインするように求められます。

ログインコマンドは、指定したプロファイルを `login_session`設定で更新します。これにより、ログインワークフロー中に選択した管理コンソールセッションの ID が保存されます。

```
[profile console]
login_session = arn:aws:iam::0123456789012:user/username
region = us-west-2
```

デフォルトでは、短期認証情報と更新トークンは、Linux、macOS、または `%USERPROFILE%\.aws\login\cache` Windows の `~/.aws/login/cache` ディレクトリの JSON ファイルに保存されます。ファイル名はログインセッション名に基づいています。ディレクトリを上書きするには、`AWS_LOGIN_CACHE_DIRECTORY`環境変数を設定します。

ログインプロバイダーの設定

この機能を設定するには、以下のように使用します。

`AWS_LOGIN_CACHE_DIRECTORY` - 環境変数

CLI と SDKs がログインセッションプロファイルにマッピングされるキャッシュされた認証情報を保存する代替ディレクトリ。

デフォルト値: `~/.aws/login/cache` Linux および macOS、または Windows `%USERPROFILE%\.aws\login\cache`の場合。

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	はい	
SDK for C++	はい	
SDK for Go V2 (1.x)	不可	
SDK for Go 1.x (V1)	はい	
SDK for Java 2.x	はい	
SDK for Java 1.x	不可	
SDK for JavaScript 3.x	はい	
SDK for JavaScript 2.x	不可	
SDK for Kotlin	はい	
SDK for .NET 4.x	はい	

SDK	サポート	注意または詳細情報
SDK for .NET 3.x	はい	
SDK for PHP 3.x	はい	
SDK for Python (Boto3)	はい	CRT が必要
SDK for Ruby 3.x	はい	
SDK for Rust	はい	
PowerShell V5 用のツール	はい	
PowerShell V4 のツール	不可	

ロール認証情報プロバイダーを引き受けます

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

ロールでは、他の方法ではアクセスできない AWS リソースへのアクセスに、一時的なセキュリティ認証情報のセットを使用する必要があるとします。これらの一時的な認証情報は、アクセスキー ID、シークレットアクセスキー、およびセキュリティトークンで構成されています。

ロールを引き受けるように SDK またはツールを設定するには、まず引き受けるのための特定のロールを作成または特定する必要があります。IAM ロールは、ロール (Amazon リソースネーム (「ARN」)で一意に識別されます。ロールは別のエンティティとの信頼関係を確立します。ロールを使用する信頼されたエンティティは AWS のサービス、、別の AWS アカウント、ウェブ ID プロバイダーまたは OIDC、または SAML フェデレーションです。

IAM ロールが特定されると、そのロールから信頼されている場合は、そのロールによって付与された権限を使用するように SDK またはツールを設定できます。これを行うには、次のコマンドを使用します。

これらの設定の使用を開始するためのガイダンスについては、このガイドの「AWS SDKsとツールを認証するための AWS 認証情報を持つロールの引き受け」を参照してください。

ロール認証情報プロバイダーを引き受けます

この機能を設定するには、以下のように使用します。

`credential_source` - 共有 AWS **`config`**ファイル設定

Amazon EC2 インスタンスまたは Amazon Elastic Container Service のコンテナ内で使用され、`role_arn` パラメータで指定したロールを引き受けるために使用する認証情報を SDK またはツールが検索できる場所を指定します。

デフォルト値： なし

有効な値:

- Environment – SDK またはツールが環境変数 `AWS_ACCESS_KEY_ID` と `AWS_SECRET_ACCESS_KEY` からソース認証情報を取得することを指定します。
- EC2InstanceMetadata – SDK またはツールが EC2 インスタンスプロファイルにアタッチされた IAM ロールを使用してソースの認証情報を取得することを指定します。
- EcsContainer – SDK またはツールが、ソースの認証情報を取得するために Amazon ECS コンテナにアタッチされた IAM ロールまたは Amazon EKS コンテナにアタッチされた IAM ロールを使用することを指定します。

`credential_source` と `source_profile` の両方を同じプロファイルで指定することはできません。

認証情報を Amazon EC2 から取得する必要があることを示すために、`config` ファイルにこれを設定する例を以下に示します。

```
credential_source = Ec2InstanceMetadata
role_arn = arn:aws:iam::123456789012:role/my-role-name
```

duration_seconds - 共有 AWS **config**ファイル設定

ロールセッションの最大期間を秒単位で指定します。

この設定は、プロファイルがロールの継承を指定している場合にのみ適用されます。

デフォルト値： 3600 秒 (1 時間) です

有効な値： この値は 900 秒 (15 分) からロールの最大セッション期間設定 (上限は 43200、すなわち12時間) までの範囲を指定できます。詳細については、「IAM ユーザーガイド」の「ロールの最大セッション期間設定を表示する」 を参照してください。

config ファイルにこれを設定する例を以下に示します。

```
duration_seconds = 43200
```

external_id - 共有 AWS **config**ファイル設定

お客様のアカウントでサードパーティーがロールを引き受けるために使用される独自の識別子を指定します。

この設定は、プロファイルが役割を引き受けるように指定していて、その役割の信頼ポリシーで ExternalId の値が必要な場合にのみ適用されます。この値は、プロファイルがロールを指定するときに AssumeRole 操作に渡される ExternalId パラメータにマップされます。

デフォルト値: なし。

有効な値: IAM ユーザーガイドの AWS 「リソースへのアクセス権を第三者に付与するときに外部 ID を使用する方法」を参照してください。

config ファイルにこれを設定する例を以下に示します。

```
external_id = unique_value_assigned_by_3rd_party
```

mfa_serial - 共有 AWS **config**ファイル設定

ロールを引き受けるときに使用する必要がある多要素認証 (MFA) デバイスの ID もしくはシリアル番号を指定します。

ロールの信頼ポリシーに MFA 認証を必要とする条件が含まれているロールを引き受ける場合に必要です。MFA の詳細については、「IAM ユーザーガイド」の「IAM のAWS 多要素認証」を参照してください。

デフォルト値: なし。

有効な値： 値には、ハードウェアデバイスのシリアルナンバー (GAHT12345678など) または仮想 MFA デバイス (など) の Amazon リソースネーム (ARN) のいずれか指定できます。ARN の形式は以下のようになります。`arn:aws:iam::account-id:mfa/mfa-device-name`

config ファイルにこれを設定する例を以下に示します。

この例では、アカウント用に作成され、ユーザーに対して有効になっている MyMFADevice という仮想 MFA デバイスを想定しています。

```
mfa_serial = arn:aws:iam::123456789012:mfa/MyMFADevice
```

role_arn - 共有 AWS **config**ファイル設定, **AWS_ROLE_ARN** - 環境変数, **aws.roleArn** - JVM システムプロパティ: Java/Kotlin のみ

このプロファイルを使用してリクエストされた操作の実行に使用する IAM ロールの Amazon リソースネーム (ARN) を指定します。

デフォルト値： なし。

有効な値： 値は、以下の `arn:aws:iam::account-id:role/role-name` 形式の IAM ロールの ARN である必要があります。

さらに、以下の設定のいずれかを指定する必要があります。

- source_profile — そのプロファイルでその役割を引き受ける権限を持つ認証情報を検索するために使用する別のプロファイルを識別する。
- credential_source — 現在の環境変数で識別される認証情報、または Amazon EC2 インスタンスプロファイルに添付されている認証情報、または Amazon ECS コンテナインスタンスを使用する。
- web_identity_token_file — モバイルまたはウェブアプリケーションで認証されたユーザーにパブリックOpenID Connect (OIDC) 互換の ID プロバイダーを使用する。

role_session_name - 共有 AWS **config**ファイル設定, **AWS_ROLE_SESSION_NAME** - 環境変数, **aws.roleSessionName** - JVM システムプロパティ: Java/Kotlin のみ

ロールセッションにアタッチする名前を指定します。この名前は、このセッションに関連付けられたエントリの AWS CloudTrail ログに表示されます。詳細については、「AWS CloudTrail ユーザーガイド」の「CloudTrail userIdentity element」を参照してください。

デフォルト値：オプションパラメータ。この値を指定しない場合、セッション名は自動的に生成されます。

有効な値: AWS CLI または AWS API がユーザーに代わって AssumeRoleオペレーション (または オペレーションなどの AssumeRoleWithWebIdentity オペレーション) を呼び出すときに RoleSessionNameパラメータに提供されます。この値は、クエリ可能な想定ロールユーザーの Amazon リソースネーム (ARN) の一部になり、このプロファイルによって呼び出されるオペレーションの CloudTrail ログエントリの一部として表示されます。

arn:aws:sts::*123456789012*:assumed-role/*my-role-name*/***my-role_session_name***.

config ファイルにこれを設定する例を以下に示します。

```
role_session_name = my-role-session-name
```

source_profile - 共有 AWS **config**ファイル設定

元のプロファイル内の role_arn 設定で指定されたロールを継承するために使用される認証情報の別のプロファイルを指定します。共有 AWS config ファイルと credentials ファイルでプロファイルがどのように使用されるかについては、「」を参照してください共有 config および credentials ファイル。

ロール引き受けプロファイルでもあるプロファイルを指定すると、認証情報が完全に解決されるように、各ロールが順番に引き継がれます。SDK が認証情報を含むプロファイルに遭遇すると、このチェーンは停止します。ロールの連鎖は、 AWS CLI または AWS API ロールセッションを最大 1 時間に制限し、増やすことはできません。詳細については、「IAM ユーザーガイド」 の「ロールの主な用語と概念」を参照してください。

デフォルト値： NONE。

有効な値： config および credentials ファイルで定義されているプロファイルの名前で構成されるテキスト文字列。現在のプロファイルの role_arn に対する値も指定する必要があります。

credential_source と source_profile の両方を同じプロファイルで指定することはできません。

設定ファイルでこれを設定する例:

```
[profile A]
source_profile = B
role_arn =  arn:aws:iam::123456789012:role/RoleA
role_session_name = ProfileARoleSession

[profile B]
credential_process = ./aws_signing_helper credential-process --certificate /
path/to/certificate --private-key /path/to/private-key --trust-anchor-
arn arn:aws:rolesanywhere:region:account:trust-anchor/TA_ID --profile-
arn arn:aws:rolesanywhere:region:account:profile/PROFILE_ID --role-arn
 arn:aws:iam::account:role/ROLE_ID
```

前の例では、A プロファイルは SDK またはツールに、リンクされた B プロファイルの認証情報を自動的に検索するように指示します。この場合、B プロファイルは IAM Roles Anywhere を使用した AWS SDKsとツールの認証 が提供する認証情報ヘルパーツールを使用して AWS SDK の認証情報を取得します。これらの一時的な認証情報は、次に AWS リソースへのアクセスに使用されます。指定されたロールには、 コマンド AWS のサービスや API メソッドなど、リクエストされたコードの実行を許可する IAM アクセス許可ポリシーがアタッチされている必要があります。プロファイル A によって実行されるすべてのアクションには、CloudTrail ログに含まれるロールセッション名が含まれます。

ロールチェーンの 2 番目の例では、Amazon Elastic Compute Cloud インスタンスにアプリケーションがあり、そのアプリケーションに別のロールを引き受けさせる場合、次の設定を使用できます。

```
[profile A]
source_profile = B
role_arn =  arn:aws:iam::123456789012:role/RoleA
role_session_name = ProfileARoleSession

[profile B]
credential_source=Ec2InstanceMetadata
```

プロファイル A は、Amazon EC2 インスタンスの認証情報を使用して指定されたロールを引き受け、認証情報を自動的に更新します。

web_identity_token_file - 共有 AWS **config**ファイル設定, **AWS_WEB_IDENTITY_TOKEN_FILE** - 環境変数, **aws.webIdentityTokenFile** - JVM システムプロパティ: Java/Kotlin のみ

対応する OAuth 2.0 プロバイダー または、OpenID Connect ID 識別情報プロバイダー からのアクセストークンを含むファイルへのパスを指定します。

この設定により、「Google」、「Facebook」、「Amazon」 などの多数のウェブ ID フェデレーションプロバイダーを使用して認証を行うことができます。SDK または開発者ツールはこのファイルの内容をロードし、ユーザーに代わって `AssumeRoleWithWebIdentity` オペレーションを呼び出すときに `WebIdentityToken` 引数として渡します。

デフォルト値： NONE。

有効な値： この値はパスとファイル名でなければなりません。ファイルには、認識情報プロバイダーから提供される、OAuth 2.0 アクセストークンまたは OpenID Connect ID トークンが含まれていなければなりません。相対パスは、プロセスの作業ディレクトリを基準にした相対パスとして扱われます。

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	はい	
SDK for C++	部分的	`credential_source` はサポートされていません。`duration_seconds` はサポートされていません。`mfa_serial` はサポートされていません。
SDK for Go V2 (1.x)	はい	

SDK	サポート	注意または詳細情報
SDK for Go 1.x (V1)	はい	共有 config ファイル設定を使用するには、設定ファイルからの読み込みを有効にする必要があります。「セッション」を参照してください。
SDK for Java 2.x	部分的	`mfa_serial` はサポートされていません。`duration_seconds` はサポートされていません。
SDK for Java 1.x	部分的	`credential_source` はサポートされていません。`mfa_serial` はサポートされていません。JVM システムプロパティはサポートされていません。
SDK for JavaScript 3.x	はい	
SDK for JavaScript 2.x	部分的	`credential_source` はサポートされていません。
SDK for Kotlin	はい	
SDK for .NET 4.x	はい	
SDK for .NET 3.x	はい	
SDK for PHP 3.x	はい	
SDK for Python (Boto3)	はい	

SDK	サポート	注意または詳細情報
SDK for Ruby 3.x	はい	
SDK for Rust	はい	
SDK for Swift	はい	
PowerShell V5 用のツール	はい	
PowerShell V4 用のツール	はい	

コンテナ認証情報プロバイダー

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

コンテナ認証情報プロバイダーは、お客様のコンテナ化されたアプリケーションの認証情報を取得します。この認証情報プロバイダーは、Amazon Elastic Container Service (Amazon ECS) および Amazon Elastic Kubernetes Service (Amazon EKS) をご利用のお客様に役立ちます。SDK は GET リクエストを通じて指定された HTTP エンドポイントから認証情報をロードします。

Amazon ECS を利用する場合は、認証情報の分離、認可、監査可能性を改善するために、タスク IAM ロールを使用することをお勧めします。Amazon ECS を設定すると、SDK とツールが認証情報を取得するために使用する `AWS_CONTAINER_CREDENTIALS_RELATIVE_URI` 環境変数が設定されます。この機能用に Amazon ECS を設定するには、「Amazon Elastic Container Service デベロッパーガイド」の「タスク IAM ロール」を参照してください。

Amazon EKS を利用する場合は、認証情報の分離、最小特権、監査可能性、独立したオペレーション、再利用性、およびスケーラビリティを改善するために、Amazon EKS Pod Identity を利用することをお勧めします。ポッドと IAM ロールの両方は、アプリケーション用に認証情報を管理するために Kubernetes サービスアカウントに関連付けられています。Amazon EKS Pod Identity の詳細については、「Amazon EKS ユーザーガイド」の「EKS Pod Identity が Pod に AWS サービスへのアクセス権を付与する仕組みを学ぶ」を参照してください。Amazon EKS を設定すると、SDK とツールが認証情報を取得するために使用する AWS_CONTAINER_CREDENTIALS_FULL_URI および AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE 環境変数が設定されます。セットアップの詳細については、「Amazon EKS ユーザーガイド」の「Amazon EKS Pod Identity エージェントのセットアップ」または「 ブログウェブサイト」の「Amazon EKS Pod Identity は、Amazon EKS クラスター上のアプリケーションの IAM アクセス許可を簡素化します。 AWS

この機能を設定するには、以下のように使用します。

AWS_CONTAINER_CREDENTIALS_FULL_URI - 環境変数

SDK が認証情報をリクエストするときに使用するフル HTTP URL エンドポイントを指定します。これにはスキームとホストの両方が含まれます。

デフォルト値： なし。

有効な値：有効な URI。

注意：この設定は *AWS_CONTAINER_CREDENTIALS_RELATIVE_URI* の代替であり、*AWS_CONTAINER_CREDENTIALS_RELATIVE_URI* が設定されていない場合にのみ使用されます。

Linux/macOS のコマンドラインによる環境変数の設定の例を以下に示します。

```
export AWS_CONTAINER_CREDENTIALS_FULL_URI=http://localhost/get-credentials
```

または

```
export AWS_CONTAINER_CREDENTIALS_FULL_URI=http://localhost:8080/get-credentials
```

AWS_CONTAINER_CREDENTIALS_RELATIVE_URI - 環境変数

SDK が認証情報をリクエストするときに使用する相対 HTTP URL エンドポイントを指定します。値は、デフォルトの Amazon ECS ホスト名 169.254.170.2 に付加されます。

デフォルト値: [なし]。

有効な値：有効な相対 URI。

Linux/macOS のコマンドラインによる環境変数の設定の例を以下に示します。

```
export AWS_CONTAINER_CREDENTIALS_RELATIVE_URI=/get-credentials?a=1
```

AWS_CONTAINER_AUTHORIZATION_TOKEN - 環境変数

認可トークンをプレーンテキストで指定します。この変数が設定されている場合、SDK は HTTP リクエストの認可ヘッダーに環境変数の値を設定します。

デフォルト値： なし。

有効な値： 文字列。

注意：この設定は *AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE* の代替であり、*AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE* が設定されていない場合にのみ使用されます。

Linux/macOS のコマンドラインによる環境変数の設定の例を以下に示します。

```
export AWS_CONTAINER_CREDENTIALS_FULL_URI=http://localhost/get-credential
export AWS_CONTAINER_AUTHORIZATION_TOKEN=Basic abcd
```

AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE - 環境変数

プレーンテキストの認可トークンを含むファイルへの絶対ファイルパスを指定します。

デフォルト値: [なし]。

有効な値： 文字列。

Linux/macOS のコマンドラインによる環境変数の設定の例を以下に示します。

```
export AWS_CONTAINER_CREDENTIALS_FULL_URI=http://localhost/get-credential
export AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE=/path/to/token
```

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	はい	
SDK for C++	はい	
SDK for Go V2 (1.x)	はい	
SDK for Go 1.x (V1)	はい	
SDK for Java 2.x	はい	Lambda SnapStart がアクティブ化されると、AWS_CONTAINER_CREDENTIALS_FULL_URI と AWS_CONTAINER_AUTHORIZATION_TOKEN は自動的に認証に使用されます。
SDK for Java 1.x	はい	Lambda SnapStart がアクティブ化されると、AWS_CONTAINER_CREDENTIALS_FULL_URI と AWS_CONTAINER_AUTHORIZATION_TOKEN は自動的に認証に使用されます。
SDK for JavaScript 3.x	はい	
SDK for JavaScript 2.x	はい	
SDK for Kotlin	はい	
SDK for .NET 4.x	はい	Lambda SnapStart がアクティブ化されると、AWS_CONTAINER_CREDENTIALS_FULL_URI と AWS_CONTA

SDK	サポート	注意または詳細情報
		INER_AUTHORIZATION_TOKEN は自動的に認証に使用されます。
SDK for .NET 3.x	はい	Lambda SnapStart がアクティブ化されると、AWS_CONTAINER_CREDENTIALS_FULL_URI と AWS_CONTAINER_AUTHORIZATION_TOKEN は自動的に認証に使用されます。
SDK for PHP 3.x	はい	
SDK for Python (Boto3)	はい	Lambda SnapStart がアクティブ化されると、AWS_CONTAINER_CREDENTIALS_FULL_URI と AWS_CONTAINER_AUTHORIZATION_TOKEN は自動的に認証に使用されます。
SDK for Ruby 3.x	はい	
SDK for Rust	はい	
SDK for Swift	はい	
PowerShell V5 用のツール	はい	
PowerShell V4 のツール	はい	

IAM Identity Center 認証情報プロバイダー

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

この認証メカニズムは、 AWS IAM アイデンティティセンター を使用して、コード AWS のサービスの へのシングルサインオン (SSO) アクセスを取得します。

> **Note**
>
> AWS SDK API ドキュメントでは、IAM Identity Center 認証情報プロバイダーは SSO 認証情報プロバイダーと呼ばれます。

IAM Identity Center を有効にしたら、共有 AWS `config`ファイルで設定のプロファイルを定義します。このプロファイルは IAM Identity Center アクセスポータルへの接続に使用されます。ユーザーが IAM Identity Center で正常に認証されると、ポータルはそのユーザーに関連付けられた IAM ロールの短期認証情報を返します。SDK が設定から一時的な認証情報を取得し、 AWS のサービス リクエストに使用する方法については、「」を参照してくださいAWS SDKsとツールの IAM Identity Center 認証の解決方法。

`config` ファイルを使用して IAM Identity Center を設定するには 2 つの方法があります。

- (推奨) SSO トークンプロバイダー設定 – セッション期間が延長されます。カスタムセッション期間のサポートが含まれています。
- 更新不可のレガシー構成 — 固定の 8 時間のセッションを使用します。

どちらの構成でも、セッションの有効期限が切れたら再度サインインする必要があります。

次の 2 つのガイドには、IAM Identity Center に関する追加情報が含まれています。

- AWS IAM アイデンティティセンター ユーザーガイド
- AWS IAM アイデンティティセンター ポータル API リファレンス

SDK とツールがこの構成を使用して認証情報を使用および更新する方法の詳細については、「AWS SDKsとツールの IAM Identity Center 認証の解決方法」を参照してください。

前提条件

最初に IAM Identity Center を有効にしておく必要があります。IAM アイデンティティセンターの認証を有効にする方法の詳細については、「AWS IAM アイデンティティセンター ユーザーガイド」の「Enabling AWS IAM アイデンティティセンター」を参照してください。

> **Note**
>
> または、完全な前提条件と、このページで説明されている必要な共有 config ファイル設定について、「IAM Identity Center を使用して AWS SDK とツールを認証する」のセットアップガイドを参照してください。

SSO トークンプロバイダー設定

SSO トークンプロバイダー設定を使用すると、 AWS SDK またはツールは延長されたセッション期間までセッションを自動的に更新します。セッション期間と最大期間の詳細については、「 AWS IAM アイデンティティセンター ユーザーガイド」の AWS 「アクセスポータルと IAM Identity Center 統合アプリケーションのセッション期間を設定する」を参照してください。

config ファイルの sso-sessionセクションは、SSO アクセストークンを取得するための設定変数をグループ化するために使用され、認証情報を取得するために使用できます AWS 。config ファイル内のこのセクションの詳細については、「設定ファイルの形式」を参照してください。

次の共有 config ファイルの例では、dev プロファイルを使用して SDK またはツールを設定し、IAM Identity Center の認証情報をリクエストします。

```
[profile dev]
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole

[sso-session my-sso]
sso_region = us-east-1
sso_start_url = https://my-sso-portal.awsapps.com/start
sso_registration_scopes = sso:account:access
```

前の例は sso-session セクションの定義とプロファイルへの関連付けを示しています。通常、SDK が AWS 認証情報をリクエストできるように、 profile セクションで sso_account_idと を設定sso_role_nameする必要があります。sso_region、sso_start_url、 は sso-sessionセクション内で設定sso_registration_scopesする必要があります。

sso_account_id と sso_role_name は SSO トークン設定のすべてのシナリオで必須というわけではありません。アプリケーション AWS のサービス がベアラー認証をサポートする のみを使用する場合、従来の AWS 認証情報は必要ありません。ベアラー認証は、ベアラートークンと呼ばれるセキュリティトークンを使用する HTTP 認証スキームです。このシナリオでは、sso_account_id と sso_role_name は必須ではありません。サービスがベアラートークン認可をサポートしているかどうかを確認するには、個々の AWS のサービス ガイドを参照してください。

登録スコープは sso-session の一部として設定されます。スコープは、ユーザーのアカウントに対するアプリケーションのアクセスを制限する OAuth 2.0 のメカニズムです。前の例では、アカウントとロールを一覧表示するために必要なアクセスを許可するように sso_registration_scopes を設定しています。

次の例は、複数のプロファイルで同じ sso-session 設定を再利用する方法を示しています。

```
[profile dev]
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole

[profile prod]
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole2

[sso-session my-sso]
sso_region = us-east-1
sso_start_url = https://my-sso-portal.awsapps.com/start
sso_registration_scopes = sso:account:access
```

認証トークンは、セッション名に基づいたファイル名を使用して、~/.aws/sso/cache ディレクトリの下のディスクにキャッシュされます。

更新不可のレガシー設定

トークンの自動更新は、更新不可のレガシー設定ではサポートされていません。代わりに、SSO トークンプロバイダー設定 の使用をお勧めします。

更新不可のレガシー設定を使用するには、プロファイル内で次の設定を指定する必要があります。

- sso_start_url
- sso_region
- sso_account_id
- sso_role_name

プロファイルのユーザーポータルは、 sso_start_url および sso_region 設定を使用して指定します。アクセス許可は sso_account_id および sso_role_name 設定で指定します。

次の例では、 config ファイルに 4 つの必要となる値を設定します。

```
[profile my-sso-profile]
sso_start_url = https://my-sso-portal.awsapps.com/start
sso_region = us-west-2
sso_account_id = 111122223333
sso_role_name = SSOReadOnlyRole
```

認証トークンは、sso_start_url に基づいたファイル名を使用して、~/.aws/sso/cache ディレクトリの下のディスクにキャッシュされます。

IAM Identity Center 認証情報プロバイダーの設定

この機能は以下を使用して設定します。

sso_start_url - 共有 AWS **config**ファイル設定

組織の IAM Identity Center の発行者 URL またはアクセスポータル URL を指す URL。詳しくは、「AWS IAM アイデンティティセンター ユーザーガイド」の「Setting up and using the AWS access portal」を参照してください。

この値を確認するには、IAM Identity Center コンソールを開き、[ダッシュボード] を表示して、[AWS アクセスポータル URL] を検索します。

- または、 のバージョン 2.22.0 以降では AWS CLI、代わりにAWS 発行者 URL の値を使用できます。

sso_region - 共有 AWS **config**ファイル設定

IAM Identity Center ポータルホスト AWS リージョン を含む 、つまり IAM Identity Center を有効にする前に選択したリージョン。これはデフォルトの AWS リージョンとは独立しており、異なる場合があります。

AWS リージョン とそのコードの完全なリストについては、『』の「リージョンエンドポイント」を参照してくださいAmazon Web Services 全般のリファレンス。この値を確認するには、IAM Identity Centerコンソールを開いて[ダッシュボード]を表示し、[リージョン]を探します。

sso_account_id - 共有 AWS **config**ファイル設定

認証に使用する AWS Organizations サービスを通じて AWS アカウント 追加された の数値 ID。

使用可能なアカウントのリストを確認するには、IAM Identity Center コンソールに移動して[AWS アカウント] ページを開きます。AWS IAM アイデンティティセンター ポータル API リファレンスの 「ListAccounts」 API メソッドを使用して利用可能なアカウントのリストを確認することもできます。たとえば、 AWS CLI メソッド list-accounts を呼び出すことができます。

sso_role_name - 共有 AWS **config**ファイル設定

ユーザーのアクセス許可を定義するIAM ロールとしてプロビジョニングされたアクセス許可セットの名前。ロールは、 で AWS アカウント 指定された に存在する必要がありますsso_account_id。ロールの Amazon リソースネーム (ARN) ではなく、ロール名を使用してください。

アクセス許可セットには IAM ポリシーとカスタムアクセス許可ポリシーがアタッチされており、ユーザーに割り当てられた AWS アカウントに付与されるアクセスレベルを定義します。

ごとに使用可能なアクセス許可セットのリストを表示するには AWS アカウント、IAM Identity Center コンソールに移動してAWS アカウントページを開きます。 AWS アカウント テーブルにリストされている正しいアクセス許可セット名を選択します。AWS IAM アイデンティティセンター ポータル API リファレンスの 「ListAccountRoles」 API メソッドを使用して、使用可能なアクセス許可セットのリストを確認することもできます。たとえば、 AWS CLI メソッド list-account-roles を呼び出すことができます。

sso_registration_scopes - 共有 AWS **config**ファイル設定

sso-session に許可するスコープのカンマ区切りのリストです。アプリケーションは 1 つ以上のスコープをリクエストでき、アプリケーションに発行されたアクセストークンは付与されたスコープに限定されます。IAM Identity Center サービスから更新トークンを取得するに

は、sso:account:access の最低限のスコープを付与する必要があります。利用可能なアクセススコープのオプションのリストについては、「AWS IAM アイデンティティセンター ユーザーガイド」の「Access scopes」を参照してください。

これらのスコープは、登録された OIDC クライアントがリクエストできるアクセス許可と、クライアントが取得するアクセストークンを定義します。スコープは、IAM Identity Center ベアラートークンで承認されたエンドポイントへのアクセスを許可します。

この設定は、更新できない従来の設定には適用されません。レガシー構成を使用して発行されたトークンは、暗黙的にsso:account:access スコープに制限されます。

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	はい	
SDK for C++	はい	
SDK for Go V2 (1.x)	はい	
SDK for Go 1.x (V1)	はい	共有 config ファイル設定を使用するには、設定ファイルからの読み込みを有効にする必要があります。「セッション」を参照してください。
SDK for Java 2.x	はい	設定値はcredentials ファイルでもサポートされています。
SDK for Java 1.x	不可	

SDK	サポート	注意または詳細情報
SDK for JavaScript 3.x	はい	
SDK for JavaScript 2.x	はい	
SDK for Kotlin	はい	
SDK for .NET 4.x	はい	
SDK for .NET 3.x	はい	
SDK for PHP 3.x	はい	
SDK for Python (Boto3)	はい	
SDK for Ruby 3.x	はい	
SDK for Rust	部分的	更新不可のレガシー設定のみ。
SDK for Swift	はい	
PowerShell V5 のツール	はい	

SDK	サポート	注意または詳細情報
PowerShell V4 のツール	はい	

IMDS 認証情報プロバイダー

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

インスタンスメタデータサービス (IMDS) は、インスタンスに関するデータで、実行中のインスタンスを設定または管理するために使用します。利用可能なデータの詳細については、「Amazon EC2 ユーザーガイド」の「インスタンスメタデータを使用して EC2 インスタンスを管理する」を参照してください。Amazon EC2 では、インスタンスにさまざまな情報を提供できるローカルエンドポイントをインスタンスで利用可能です。インスタンスにロールがアタッチされている場合は、そのロールに有効な認証情報のセットが使用できます。SDK はそのエンドポイントを使用して、デフォルトの認証情報プロバイダーチェーンの一部として認証情報を解決できます。インスタンスメタデータサービスバージョン 2 (IMDSv2) は、IMDS のより安全なバージョンであり、デフォルトで使用されます。再試行できない条件 (HTTP エラーコード 403、404、405) が原因で失敗した場合は、IMDSv1 がフォールバックとして使用されます。

この機能を設定するには、以下のように使用します。

`AWS_EC2_METADATA_DISABLED` - 環境変数

認証情報の取得に Amazon EC2 インスタンスメタデータサービス (IMDS) を使用するかどうか。

デフォルト値: `false`。

有効な値:

- **`true`** – 認証情報を取得するために IMDS を使用しません。
- **`false`** – 認証情報を取得するために IMDS を使用します。

ec2_metadata_v1_disabled - 共有 AWS **config**ファイル設定, **AWS_EC2_METADATA_V1_DISABLED** - 環境変数, **aws.disableEc2MetadataV1** - JVM システムプロパティ: Java/Kotlin のみ

IMDSv2 が失敗した場合に、Instance Metadata Service Version 1 (IMDSv1) をフォールバックとして使用するかどうか。

> **Note**
>
> 新しい SDK は IMDSv1 をサポートしていないため、この設定はサポートされていません。詳細については、表 AWS SDKsとツールによるサポート を参照してください。

デフォルト値: `false`。

有効な値:

- **`true`** – IMDSv1 をフォールバックとして使用しません。
- **`false`** – IMDSv1 をフォールバックとして使用します。

ec2_metadata_service_endpoint - 共有 AWS **config**ファイル設定, **AWS_EC2_METADATA_SERVICE_ENDPOINT** - 環境変数, **aws.ec2MetadataServiceEndpoint** - JVM システムプロパティ: Java/Kotlin のみ

IMDS のエンドポイント。この値は、 AWS SDKsとツールが Amazon EC2 インスタンスメタデータを検索するデフォルトの場所を上書きします。

デフォルト値： `ec2_metadata_service_endpoint_mode` が `IPv4` に等しい場合、デフォルトエンドポイントは `http://169.254.169.254` です。`ec2_metadata_service_endpoint_mode` が `IPv6` に等しい場合、デフォルトのエンドポイントは `http://[fd00:ec2::254]` です。

有効な値：有効な URI。

ec2_metadata_service_endpoint_mode - 共有 AWS **config**ファイル設定, **AWS_EC2_METADATA_SERVICE_ENDPOINT_MODE** - 環境変数, **aws.ec2MetadataServiceEndpointMode** - JVM システムプロパティ: Java/Kotlin のみ

IMDS のエンドポイントモード。

デフォルト値：`IPv4`。

有効な値： `IPv4`、`IPv6`。

> **Note**
>
> IMDS 認証情報プロバイダーは 認証情報プロバイダーチェーンを理解する の一部です。ただし、IMDS 認証情報プロバイダーは、この一連で他のいくつかのプロバイダーの後にのみチェックされます。そのため、プログラムでこのプロバイダーの認証情報を使用する場合は、設定から他の有効な認証情報プロバイダーを削除するか、別のプロファイルを使用する必要があります。あるいは、認証情報プロバイダチェーンに頼ってどのプロバイダが有効な認証情報を返すかを自動的に検出する代わりに、コード内で IMDS 認証情報プロバイダの使用を指定してください。サービスクライアントを作成するときに、認証情報ソースを直接指定できます。

IMDS 認証情報のセキュリティ

デフォルトでは、 AWS SDK に有効な認証情報が設定されていない場合、SDK は Amazon EC2 インスタンスメタデータサービス (IMDS) を使用して AWS ロールの認証情報を取得しようとします。この動作は、`AWS_EC2_METADATA_DISABLED` 環境変数を `true` に設定することで無効にできます。これにより、不必要なネットワークアクティビティが防止され、Amazon EC2 インスタンスメタデータサービスが偽装される可能性がある信頼できないネットワークのセキュリティが強化されます。

> **Note**
>
> AWS 有効な認証情報で設定された SDK クライアントは、これらの設定に関係なく、IMDS を使用して認証情報を取得しません。

Amazon EC2 IMDS 認証情報の使用の無効化

この環境変数の設定方法は、使用中のオペレーティングシステムと、変更を持続的にしたいかどうかによって異なります。

Linux および macOS

Linux または macOS を使用しているお客様は、次のコマンドを使用して、この環境変数を設定できます。

```
$ export AWS_EC2_METADATA_DISABLED=true
```

この設定を複数のシェルセッションやシステムの再起動後も維持したい場合は、.bash_profile、.zsh_profile、.profile などの上記のコマンドをシェルプロファイルファイルに追加できます。

Server

Windows を使用しているお客様は、次のコマンドを使用して、この環境変数を設定できます。

```
$ set AWS_EC2_METADATA_DISABLED=true
```

この設定を複数のシェルセッションやシステムの再起動後も維持したい場合は、代わりに以下のコマンドを使用できます。

```
$ setx AWS_EC2_METADATA_DISABLED=true
```

> **Note**
>
> setx コマンドは現在のシェルセッションに値を適用しないため、変更を有効にするにはシェルをリロードするか再度開く必要があります。

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	はい	
SDK for C++	はい	
SDK for Go V2 (1.x)	はい	

SDK	サポート	注意または詳細情報
SDK for Go 1.x (V1)	はい	共有 config ファイル設定を使用するには、設定ファイルからの読み込みを有効にする必要があります。「セッション」を参照してください。
SDK for Java 2.x	はい	
SDK for Java 1.x	部分的	JVM システムプロパティ: aws.disableEc2MetadataV1 の代わりに com.amazonaws.sdk.disableEc2MetadataV1 を使用します。aws.ec2MetadataServiceEndpoint と aws.ec2MetadataServiceEndpointMode はサポートされません。
SDK for JavaScript 3.x	はい	
SDK for JavaScript 2.x	はい	
SDK for Kotlin	はい	IMDSv1 フォールバックを使用しません。
SDK for .NET 4.x	はい	
SDK for .NET 3.x	はい	
SDK for PHP 3.x	はい	
SDK for Python (Boto3)	はい	

SDK	サポート	注意または詳細情報
SDK for Ruby 3.x	はい	
SDK for Rust	はい	IMDSv1 フォールバックを使用しません。
SDK for Swift	はい	
PowerShell V5 のツール	はい	`[Amazon.Util.EC2InstanceMetadata]::EC2MetadataV1Disabled = $true` を使用して、コードで IMDSv1 フォールバックを明示的に無効にできます。
PowerShell V4 用のツール	はい	`[Amazon.Util.EC2InstanceMetadata]::EC2MetadataV1Disabled = $true` を使用して、コードで IMDSv1 フォールバックを明示的に無効にできます。

プロセス認証情報プロバイダー

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

SDK には、カスタムユースケースに合わせて認証情報プロバイダーチェーンを拡張する方法の機能があります。このプロバイダーは、オンプレミスの認証情報ストアからの認証情報の取得やオンプレミスの ID プロバイダーとの統合など、カスタム実装を提供するために使用できます。

例えば、IAM Roles Anywhere は `credential_process` を使用して、アプリケーションに代わって一時的な認証情報を取得します。この用途に合わせて `credential_process` を設定するには、IAM Roles Anywhere を使用した AWS SDKsとツールの認証 を参照してください。

> **Note**
>
> 外部プロセスから認証情報を取得する方法を以下に示します。これは、 AWSの外部でソフトウェアを実行している場合に使用できます。 AWS コンピューティングリソース上に構築する場合は、他の認証情報プロバイダーを使用します。このオプションを使用する場合は、使用しているオペレーティングシステムのセキュリティ上のベストプラクティスに従って、設定ファイルができるだけロックされていることを確認する必要があります。SDKs と はそのような情報を AWS CLI キャプチャしてログに記録しStdErr、権限のないユーザーに公開する可能性があるため、カスタム認証情報ツールが にシークレット情報を書き込まないことを確認します。

この機能を設定するには、以下のように使用します。

`credential_process` - 共有 AWS **`config`**ファイル設定

使用する認証情報を生成あるいは取得するために ユーザーに代わって SDK またはツールが実行する外部のコマンドを指定します。この設定では、SDK が呼び出すプログラム/コマンドの名前を指定します。SDK がプロセスを呼び出すと、プロセスが JSON データを `stdout` に書き込むのを待ちます。カスタムプロバイダーは、特定の形式で情報を返す必要があります。この情報には、 SDK またはツールがユーザーを認証するために使用できる認証情報が含まれています。

> **Note**
>
> プロセス認証情報プロバイダーは 認証情報プロバイダーチェーンを理解する の一部です。ただし、プロセス認証情報プロバイダーは、このシリーズの他のいくつかのプロバイダーの後にのみチェックされます。そのため、プログラムでこのプロバイダーの認証情報を使用する場合は、設定から他の有効な認証情報プロバイダーを削除するか、別のプロファイルを使用する必要があります。あるいは、認証情報プロバイダーチェーンに頼ってどのプロバイダーが有効な認証情報を返すかを自動的に検出する代わりに、プロセス認証情報プロバイダーの使用をコードで指定してください。サービスクライアントを作成するときに、認証情報ソースを直接指定できます。

認証情報プログラムへのパスの指定

設定の値は、SDK または開発ツールがユーザーに代わって実行するプログラムへのパスを含む文字列です。

- パスとファイル名には、A～Z、a～z、0～9、ハイフン (-)、アンダースコア (_)、ピリオド (.)、フォワードスラッシュ (/)、バックスラッシュ (\)、スペースのみを使用できます。
- パスまたはファイル名にスペースが含まれている場合は、完全なパスとファイル名を二重引用符 (" ") で囲みます。
- パラメータ名またはパラメータ値にスペースが含まれている場合は、その要素を二重引用符 (" ") で囲みます。囲むのは、名前または値のみであり、そのペアではありません。
- 文字列に環境変数を含めないでください。例えば、$HOME または %USERPROFILE% を含めることはできません。
- ホームフォルダを ~ として指定しないでください。* フルパスまたはベースファイル名を指定する必要があります。ベースファイル名がある場合、システムは PATH 環境変数で指定されたフォルダー内でプログラムを検索しようとします。次のようにパスはオペレーティングシステムによって異なります。

  次の例は、Linux/macOS上の config 共有ファイルに credential_process を設定する方法を示しています。

  ```
  credential_process = "/path/to/credentials.sh" parameterWithoutSpaces "parameter with
   spaces"
  ```

  次の例は、Windows 上の config 共有ファイルに credential_process を設定する方法を示しています。

  ```
  credential_process = "C:\Path\To\credentials.cmd" parameterWithoutSpaces "parameter
   with spaces"
  ```

- 以下の専用プロファイル内で指定できます。

  ```
  [profile cred_process]
  credential_process = /Users/username/process.sh
  region = us-east-1
  ```

認証情報プログラムからの有効な出力

SDK はプロファイルで指定されたようにコマンドを実行し、次に標準出力からデータを読み取ります。スクリプトであるかバイナリープログラムであるかに関わらず、指定するコマンドは、以下の構文と一致する JSON 出力を STDOUT に生成する必要があります。

```
{
    "Version": 1,
    "AccessKeyId": "an AWS access key",
    "SecretAccessKey": "your AWS secret access key",
    "SessionToken": "the AWS session token for temporary credentials",
    "Expiration": "RFC3339 timestamp for when the credentials expire"
}
```

> **Note**
>
> 本文書の執筆時点では、Version キーは 1 に設定する必要があります。構造が進化するため、時間の経過と共に増えていく可能性があります。

Expiration キーは、RFC3339 形式のタイムスタンプです。Expiration キーがツールの出力にない場合、SDK はこの認証情報が更新されない長期の認証情報であると判断します。それ以外の認証情報は一時的な認証情報と見なされ、有効期限が切れる前に credential_process を再実行して自動的に更新されます。

> **Note**
>
> SDK は、外部プロセスの認証情報をロールを引き受けるような認証情報としてキャッシュしません。キャッシュが必要な場合は、外部プロセス内で実装する必要があります。

外部プロセスはゼロ以外のリターンコードを返して、認証情報の取得時にエラーが発生したことを示すことができます。

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	はい	
SDK for C++	はい	
SDK for Go V2 (1.x)	はい	
SDK for Go 1.x (V1)	はい	共有 config ファイル設定を使用するには、設定ファイルからの読み込みを有効にする必要があります。「セッション」を参照してください。
SDK for Java 2.x	はい	
SDK for Java 1.x	はい	
SDK for JavaScript 3.x	はい	
SDK for JavaScript 2.x	はい	
SDK for Kotlin	はい	
SDK for .NET 4.x	はい	
SDK for .NET 3.x	はい	

SDK	サポート	注意または詳細情報
SDK for PHP 3.x	はい	
SDK for Python (Boto3)	はい	
SDK for Ruby 3.x	はい	
SDK for Rust	はい	
SDK for Swift	はい	
PowerShell V5 用のツール	はい	
PowerShell V4 のツール	はい	

AWS SDKs標準化された機能

多くの機能は、一貫したデフォルトに標準化されており、多くの SDK で同じように動作します。この一貫性により、複数の SDK 間のコーディングの生産性と明確性が向上します。すべての設定はコード内で上書きできます。詳細については、特定の SDK API を参照してください。

> **Important**
>
> すべての SDK がすべての機能をサポートしているわけではなく、機能内のすべての側面をサポートしているわけでもありません。

トピック

アカウントベースのエンドポイント

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

アカウントベースのエンドポイントは、 AWS アカウント アカウント ID を使用し、この機能をサポートしているサービスのリクエストをルーティングして、高いパフォーマンスとスケーラビリティを確保するのに役立ちます。アカウントベースのエンドポイントをサポートする AWS SDK とサービスを使用する場合、SDK クライアントはリージョンのエンドポイントではなくアカウントベースのエンドポイントを構築して使用します。アカウント ID が SDK クライアントに表示されない場

合、クライアントはリージョンのエンドポイントを使用します。アカウントベースのエンドポイントは の形式をとりhttps://*<account-id>*.ddb.*<region>*.amazonaws.com、 *<account-id>*と*<region>*は AWS アカウント ID と です AWS リージョン。

この機能を設定するには、以下のように使用します。

aws_account_id - 共有 AWS **config**ファイル設定, **AWS_ACCOUNT_ID** - 環境変数, **aws.accountId** - JVM システムプロパティ: Java/Kotlin のみ

AWS アカウント ID。アカウントベースのエンドポイントのルーティングに使用されます。 AWS アカウント ID の形式は 111122223333 のようになります。

アカウントベースのエンドポイントのルーティングは、一部のサービスのリクエストパフォーマンスを向上させます。

account_id_endpoint_mode - 共有 AWS **config**ファイル設定, **AWS_ACCOUNT_ID_ENDPOINT_MODE** - 環境変数, **aws.accountIdEndpointMode** - JVM システムプロパティ: Java/Kotlin のみ

この設定は、必要に応じてアカウントベースのエンドポイントのルーティングをオフにし、アカウントベースのルールをバイパスするために使用されます。

デフォルト値: preferred

有効な値:

- **preferred** – アカウント ID が使用可能な場合、エンドポイントに含める必要があります。
- **disabled** – 解決済みのエンドポイントにアカウント ID は含まれません。
- **required** – エンドポイントにアカウント ID が含まれる必要があります。アカウント ID が使用できない場合、SDK はエラーをスローします。

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	SDK バージョンでリリース	注意または詳細情報
AWS CLI v2	はい	2.25.0	
AWS CLI v1	はい	1.38.0	
SDK for C++	不可		
SDK for Go V2 (1.x)	はい	v1.35.0	
SDK for Go 1.x (V1)	不可		
SDK for Java 2.x	はい	v2.28.4	
SDK for Java 1.x	はい	v1.12.771	
SDK for JavaScript 3.x	はい	v3.656.0	
SDK for JavaScript 2.x	不可		
SDK for Kotlin	はい	v1.3.37	
SDK for .NET 4.x	はい	4.0.0	
SDK for .NET 3.x	不可		

SDK	サポート	SDK バージョンでリリース	注意または詳細情報
SDK for PHP 3.x	はい	v3.318.0	
SDK for Python (Boto3)	はい	1.37.0	
SDK for Ruby 3.x	はい	v1.123.0	
SDK for Rust	はい	リリース-2025-04-24	
SDK for Swift	はい	1.2.0	
PowerShell V5 のツール	不可		
PowerShell V4 のツール	不可		

アプリケーション ID

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

単一の を複数のカスタマーアプリケーションで使用して、 を呼び出す AWS アカウント ことができます AWS のサービス。アプリケーション ID は、お客様が を使用して一連の呼び出しを行ったソースアプリケーションを識別する方法を提供します AWS アカウント。 AWS SDKs とサービスは、この値を 以外の方法で使用または解釈して、お客様の通信に表示します。たとえば、この値を運用 E

メールや に含める AWS Health Dashboard ことで、通知に関連付けられているアプリケーションを一意に識別できます。

この機能を設定するには、以下のように使用します。

sdk_ua_app_id - 共有 AWS **config**ファイル設定, **AWS_SDK_UA_APP_ID** - 環境変数, **sdk.ua.appId** - JVM システムプロパティ: Java/Kotlin のみ

この設定は、特定の 内のどのアプリケーションが を呼び AWS アカウント 出すかを識別するためにアプリケーションに割り当てる一意の文字列です AWS。

デフォルト値: None

有効な値: 最大長が 50 の文字列。文字、数字、および次の特殊文字を使用できます: !、#、$%、&、'、*+-、`、.^_、|、~。

config ファイルにこの値を設定する例を以下に示します。

```
[default]
sdk_ua_app_id=ABCDEF
```

Linux/macOS のコマンドラインによる環境変数の設定の例を以下に示します。

```
export AWS_SDK_UA_APP_ID=ABCDEF
export AWS_SDK_UA_APP_ID="ABC DEF"
```

Windows のコマンドラインによる環境変数の設定の例を以下に示します。

```
setx AWS_SDK_UA_APP_ID ABCDEF
setx AWS_SDK_UA_APP_ID="ABC DEF"
```

使用するシェルに特別な意味を持つ記号を含める場合は、必要に応じて値を エスケープします。

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	はい	
SDK for C++	はい	共有 config ファイルはサポートされていません。
SDK for Go V2 (1.x)	はい	
SDK for Go 1.x (V1)	不可	
SDK for Java 2.x	部分的	共有 config ファイル設定はサポートされておらず、環境変数はサポートされていません。
SDK for Java 1.x	不可	
SDK for JavaScript 3.x	はい	
SDK for JavaScript 2.x	不可	
SDK for Kotlin	はい	JVM システムプロパティは aws.userAgentAppId です。
SDK for .NET 4.x	はい	
SDK for .NET 3.x	はい	

SDK	サポート	注意または詳細情報
SDK for PHP 3.x	はい	
SDK for Python (Boto3)	はい	
SDK for Ruby 3.x	はい	
SDK for Rust	はい	
SDK for Swift	はい	
PowerShell V5 のツール	はい	
PowerShell V4 用のツール	はい	

Amazon EC2 インスタンスメタデータ

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

Amazon EC2 では、インスタンスメタデータサービス (IMDS) と呼ばれるサービスがインスタンスで使用できます。このサービスの詳細については、「Amazon EC2 ユーザーガイド」の「インスタンスメタデータを使用して EC2 インスタンスを管理する」を参照してください。IAM ロールで設定された Amazon EC2 インスタンスで認証情報の取得を試行すると、インスタンスメタデータサービスへの接続が調整可能になります。

この機能を設定するには、以下のように使用します。

metadata_service_num_attempts - 共有 AWS **config**ファイル設定,
AWS_METADATA_SERVICE_NUM_ATTEMPTS - 環境変数

この設定は、インスタンスメタデータサービスからデータの取得を試行するとき、停止するまでに試行する総回数を指定します。

デフォルト値： 1

有効な値：1 以上の数値。

metadata_service_timeout - 共有 AWS **config**ファイル設定,
AWS_METADATA_SERVICE_TIMEOUT - 環境変数

インスタンスメタデータサービスからデータの取得を試行するときにタイムアウトするまでの秒数を指定。

デフォルト値： 1

有効な値：1 以上の数値。

config ファイルに次の値を設定する例を以下に示します。

```
[default]
metadata_service_num_attempts=10
metadata_service_timeout=10
```

Linux/macOS のコマンドラインによる環境変数の設定の例を以下に示します。

```
export AWS_METADATA_SERVICE_NUM_ATTEMPTS=10
export AWS_METADATA_SERVICE_TIMEOUT=10
```

Windows のコマンドラインによる環境変数の設定の例を以下に示します。

```
setx AWS_METADATA_SERVICE_NUM_ATTEMPTS 10
setx AWS_METADATA_SERVICE_TIMEOUT 10
```

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	はい	
SDK for C++	不可	
SDK for Go V2 (1.x)	不可	
SDK for Go 1.x (V1)	不可	
SDK for Java 2.x	部分的	`AWS_METADATA_SERVICE_TIMEOUT` のみサポートされています。
SDK for Java 1.x	部分的	`AWS_METADATA_SERVICE_TIMEOUT` のみサポートされています。
SDK for JavaScript 3.x	不可	
SDK for JavaScript 2.x	不可	
SDK for Kotlin	不可	

SDK	サポート	注意または詳細情報
SDK for .NET 4.x	不可	
SDK for .NET 3.x	不可	
SDK for PHP 3.x	はい	
SDK for Python (Boto3)	はい	
SDK for Ruby 3.x	不可	
SDK for Rust	不可	
SDK for Swift	不可	
PowerShell V5 のツール	不可	
PowerShell V4 のツール	不可	

Amazon S3 アクセスポイント

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

Amazon S3 サービスでは、Amazon S3 バケットを操作する代替方法としてアクセスポイントが使用できます。アクセスポイントには、バケットに直接ではなく、一意のポリシーと設定を適用できます。 AWS SDKs を使用すると、バケット名を明示的に指定する代わりに、バケットフィールドのアクセスポイント Amazon リソースネーム (ARNs) を API オペレーションに使用できます。アクセスポイント ARN と `GetObject` を使用してバケットからオブジェクトを取得したり、アクセスポイント ARN と `PutObject` を使用してバケットにオブジェクトを追加したりするなど、特定の操作に使用されます。

Amazon S3 Access Points と ARN の詳細については、「Amazon S3 ユーザーガイド」の「アクセスポイントの使用」を参照してください。

この機能を設定するには、以下のように使用します。

`s3_use_arn_region` - 共有 AWS **`config`**ファイル設定, **`AWS_S3_USE_ARN_REGION`** - 環境変数, **`aws.s3UseArnRegion`** - JVM システムプロパティ: Java/Kotlin のみ, コード内で値を直接設定するには、使用している SDK に直接問い合わせてください。

この設定は、SDK がアクセスポイント ARN を使用してリクエストのリージョンエンドポイント AWS リージョン を構築するかどうかを制御します。SDK AWS リージョン は、ARN がクライアントの設定と同じ AWS パーティションによって処理されていることを検証 AWS リージョンし、ほとんどの場合失敗するクロスパーティション呼び出しを防止します。複数定義した場合、コードで設定されたものが優先され、次に環境変数設定が続きます。

デフォルト値: `false`

有効な値:

- **`true`** – SDK は、クライアントの設定ではなく、エンドポイントを構築する AWS リージョン ときに ARN を使用します AWS リージョン。例外: クライアントの AWS リージョン が FIPS に設定されている場合は AWS リージョン、ARN の と一致する必要があります AWS リージョン。そうしないと、エラーが発生します。
- **`false`** – SDK は、エンドポイントを構築するときに、クライアントで設定された の代わりに ARN の AWS リージョン を使用します。

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	はい	
SDK for C++	はい	
SDK for Go V2 (1.x)	はい	
SDK for Go 1.x (V1)	はい	共有 config ファイル設定を使用するには、設定ファイルからの読み込みを有効にする必要があります。「セッション」を参照してください。
SDK for Java 2.x	はい	
SDK for Java 1.x	はい	JVM システムプロパティはサポートされていません。
SDK for JavaScript 3.x	はい	
SDK for JavaScript 2.x	はい	
SDK for Kotlin	はい	
SDK for .NET 4.x	はい	
SDK for .NET 3.x	はい	標準の優先順位には従いません。共有 config ファイルの値が環境変数よりも優先されます。

SDK	サポート	注意または詳細情報
SDK for PHP 3.x	はい	
SDK for Python (Boto3)	はい	
SDK for Ruby 3.x	はい	
SDK for Rust	不可	
SDK for Swift	不可	
PowerShell V5 用のツール	はい	標準の優先順位には従いません。共有 config ファイルの値が環境変数よりも優先されます。
PowerShell V4 のツール	はい	標準の優先順位には従いません。共有 config ファイルの値が環境変数よりも優先されます。

Amazon S3 マルチリージョンアクセスポイント

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

Amazon S3 マルチリージョンアクセスポイントを使用すると、アプリケーションが複数の AWS リージョンにあるAmazon S3 バケットからのリクエストを実行するために使用できるグローバルエンドポイントを作成できます。マルチリージョンアクセスポイントを使用して、単一のリージョンで使用されるのと同じアーキテクチャでマルチリージョンアプリケーションを構築し、世界中のどこでもこれらのアプリケーションを実行することができます。

マルチリージョンアクセスポイントの詳細については、「Amazon S3 ユーザーガイド」の「Amazon S3 のマルチリージョンアクセスポイント」を参照してください。

マルチリージョンアクセスポイントの Amazon リソースネーム (ARN) の機能の詳細については、「Amazon S3 ユーザーガイド」の「マルチリージョンアクセスポイントを使用したリクエスト」を参照してください。

マルチリージョンアクセスポイント作成の詳細については、「Amazon S3 ユーザーガイド」の「マルチリージョンアクセスポイントの管理」を参照してください。

SigV4A アルゴリズムは、グローバルリージョンリクエストの署名に使用される署名実装です。このアルゴリズムは、AWS 共通ランタイム (CRT) ライブラリ への依存関係を通じて SDK によって取得されます。

この機能を設定するには、以下のように使用します。

s3_disable_multiregion_access_points - 共有 AWS **config**ファイル設定, **AWS_S3_DISABLE_MULTIREGION_ACCESS_POINTS** - 環境変数, **aws.s3DisableMultiRegionAccessPoints** - JVM システムプロパティ: Java/Kotlin のみ, コード内で値を直接設定するには、使用している SDK を直接調べてください。

この設定は、SDK がクロスリージョンリクエストを試みる可能性があるかどうかを制御します。複数定義した場合、コードで設定されたものが優先され、次に環境変数設定が続きます。

デフォルト値: `false`

有効な値:

- **true** – クロスリージョンリクエストの使用を停止します。
- **false** – マルチリージョンアクセスポイントを使用したクロスリージョンリクエストを有効にします。

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	はい	
SDK for C++	はい	
SDK for Go V2 (1.x)	はい	
SDK for Go 1.x (V1)	不可	
SDK for Java 2.x	はい	
SDK for Java 1.x	不可	
SDK for JavaScript 3.x	はい	
SDK for JavaScript 2.x	不可	
SDK for Kotlin	はい	
SDK for .NET 4.x	はい	
SDK for .NET 3.x	はい	
SDK for PHP 3.x	はい	

SDK	サポート	注意または詳細情報
SDK for Python (Boto3)	はい	
SDK for Ruby 3.x	はい	
SDK for Rust	はい	
SDK for Swift	不可	
PowerShell V5 のツール	はい	
PowerShell V4 のツール	はい	

S3 Express One Zone セッション認証

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

S3 Express One Zone は頻繁にアクセスされるデータに対して 1 桁ミリ秒のレイテンシーを実現する、Amazon S3 の高性能ストレージクラスです。S3 Express One Zone バケットを使用すると、AWS SDKsとツールは、データリクエストの低レイテンシー認可用に最適化されたセッションベースの認証を自動的に使用します。セッショントークンをゾーン (オブジェクトレベル) オペレーションで使用して認可関連のレイテンシーをセッション内の多数のリクエストに分散させることで、認証のオーバーヘッドが減少し、全体的なリクエストのパフォーマンスが向上します。

S3 Express One Zone バケットは、bucket-name--usw2-az1--x-s3 などのアベイラビリティーゾーン ID を含む特定の命名形式を使用します。SDK はこの命名パターンを検出すると、リクエストを適切な S3 Express One Zone エンドポイントに自動的にルーティングし、最適化された認証フローを適用します。セッション認証は、バケットへの低レイテンシーアクセスを実現する一時的なバケット固有の認証情報を作成し、SDK によって自動的にキャッシュおよび更新されます。詳細については、「Amazon S3 ユーザーガイド」の「S3 Express One Zone」を参照してください。

S3 Express One Zone バケットでは、セッション認証はデフォルトで有効になっています。

この機能を設定するには、以下のように使用します。

s3_disable_express_session_auth - 共有 AWS **config**ファイル設定, **AWS_S3_DISABLE_EXPRESS_SESSION_AUTH** - 環境変数, **aws.disableS3ExpressAuth** - JVM システムプロパティ: Java/Kotlin のみ

S3 Express One Zone セッション認証を無効にするかどうかを制御します。true に設定すると、SDK はセッション認証の代わりに S3 Express One Zone バケットに標準 SigV4 認証を使用します。

デフォルト値: false

有効な値:

- **true** – S3 Express One Zone セッション認証を無効にします。
- **false** – S3 Express One Zone セッション認証を有効にします。

config ファイルにこの値を設定する例を以下に示します。

```
[default]
s3_disable_express_session_auth=true
```

Linux/macOS のコマンドラインによる環境変数の設定の例を以下に示します。

```
export AWS_S3_DISABLE_EXPRESS_SESSION_AUTH=true
```

Windows のコマンドラインによる環境変数の設定の例を以下に示します。

```
setx AWS_S3_DISABLE_EXPRESS_SESSION_AUTH true
```

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	不可	
AWS CLI v1	不可	
SDK for C++	はい	
SDK for Go V2 (1.x)	はい	
SDK for Go 1.x (V1)	不可	共有 config ファイル設定を使用するには、設定ファイルからの読み込みを有効にする必要があります。「セッション」を参照してください。
SDK for Java 2.x	はい	
SDK for Java 1.x	不可	
SDK for JavaScript 3.x	はい	
SDK for JavaScript 2.x	不可	

SDK	サポート	注意または詳細情報
SDK for Kotlin	はい	JVM システムプロパティは `aws.s3DisableExpressSessionAuth` です。
SDK for .NET 4.x	はい	
SDK for .NET 3.x	はい	
SDK for PHP 3.x	はい	
SDK for Python (Boto3)	不可	
SDK for Ruby 3.x	はい	
SDK for Rust	はい	
SDK for Swift	はい	
PowerShell V5 のツール	はい	
PowerShell V4 のツール	はい	

認証スキーム

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

AWS サービスは、 AWS 署名バージョン 4 (SigV4) や AWS 署名バージョン 4a (SigV4a) など、複数の認証スキームをサポートしています。デフォルトでは、SDK はサービスモデル定義に基づいて認証スキームを選択し、最適な互換性を提供するスキームを優先します。ただし、特定の要件に合わせて最適化するために、優先する認証スキームを設定できます。

SigV4 とは異なり、SigV4a で署名されたリクエストは複数の AWS リージョンで有効です。SigV4a は、クロスリージョンリクエスト署名により可用性を強化し、リージョンの中断時にバックアップリージョンへの自動フェイルオーバーを可能にします。これは、 AWS Identity and Access Management や Amazon CloudFront などのグローバルサービスに特に役立ちます。

これらの 2 つの認証スキームの詳細については、「IAM ユーザーガイド」の「API リクエストに対するAWS Signature Version 4」を参照してください。

この機能を設定するには、以下のように使用します。

auth_scheme_preference - 共有 AWS **config**ファイル設定, **AWS_AUTH_SCHEME_PREFERENCE** - 環境変数, **aws.authSchemePreference** - JVM システムプロパティ: Java/Kotlin のみ

優先する認証スキームのカンマ区切りリストを優先する順に指定します。サービスが複数の認証スキームをサポートしている場合、SDK は指定された順序でこのリストのスキームを使用しようとし、優先するスキームが利用できない場合はデフォルトの動作に戻ります。

デフォルト値: なし。

有効な値: 次の 1 つ以上のカンマ区切りリスト。

- **sigv4** – Signature Version 4 (最速のパフォーマンス、単一リージョン)
- **sigv4a** – Signature Version 4a (可用性の向上、クロスリージョンサポート、SigV4 よりも署名パフォーマンスが遅い)
- **httpBearerAuth** – HTTP ベアラートークン認証

スキーム名間のスペース文字とタブ文字は無視されます。

config ファイルにこの値を設定して SigV4a を優先する例を以下に示します。

```
[default]
auth_scheme_preference=sigv4a,sigv4
```

sigv4a_signing_region_set - 共有 AWS **config**ファイル設定,
AWS_SIGV4A_SIGNING_REGION_SET - 環境変数

SigV4a マルチリージョン署名 AWS リージョン 用の のカンマ区切りリストを指定します。これは、SigV4a が選択された認証スキームである場合に、リクエストのデフォルトのリージョンセットとして使用されます。

デフォルト値: リクエストによって決定されます。

有効な値: AWS リージョンのカンマ区切りリスト。リージョン間のスペース文字とタブ文字は無視されます。

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	はい	
SDK for C++	はい	
SDK for Go V2 (1.x)	はい	
SDK for Go 1.x (V1)	不可	

SDK	サポート	注意または詳細情報
SDK for Java 2.x	はい	
SDK for Java 1.x	不可	
SDK for JavaScript 3.x	はい	
SDK for JavaScript 2.x	不可	
SDK for Kotlin	はい	
SDK for .NET 4.x	はい	
SDK for .NET 3.x	不可	
SDK for PHP 3.x	はい	
SDK for Python (Boto3)	はい	
SDK for Ruby 3.x	はい	
SDK for Rust	はい	
SDK for Swift	はい	

SDK	サポート	注意または詳細情報
PowerShell V5 用のツール	はい	
PowerShell V4 のツール	不可	

AWS リージョン

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

AWS リージョン は、 を使用する際に理解しておくべき重要な概念です AWS のサービス。

を使用すると AWS リージョン、特定の地域に AWS のサービス 物理的に存在する にアクセスできます。これは、ユーザーがアクセスする場所の近くでのデータとアプリケーションの実行を維持するために有効です。リージョンでは耐障害性や安定性が提供され、レイテンシーを低減することもできます。これにより、リージョンの障害の影響を受けずに利用できる冗長リソースを作成できます。

ほとんどの AWS のサービス リクエストは、特定の地理的リージョンに関連付けられています。あるリージョンで作成したリソースは、 AWS のサービスで提供されるレプリケーション機能を明示的に使用しないかぎり、他のリージョンに存在することはありません。たとえば、Amazon S3 と Amazon EC2 はクロスリージョンのレプリケーションをサポートしています。IAM などの一部のサービスには、リージョンリソースがありません。

AWS 全般のリファレンス には、以下の情報が含まれています。

- リージョンとエンドポイントの関係を理解し、既存のリージョンエンドポイントのリストを表示するには、「AWS サービスエンドポイント」 を参照してください。
- 各 AWS のサービスでサポートされているすべてのリージョンおよびエンドポイントの現在のリストを確認する方法については、の「サービスとエンドポイントの割り当て」を参照してください。

サービスクライアントの作成

プログラムで にアクセスするには AWS のサービス、SDKsは各 にクライアントクラス/オブジェクトを使用します AWS のサービス。たとえば、アプリケーションが Amazon EC2 にアクセスする必要がある場合、アプリケーションはそのサービスとやり取りする Amazon EC2 クライアントオブジェクトを作成します。

コード自体のクライアントにリージョンが明示的に指定されていない場合、クライアントはデフォルトで以下の `region` 設定で設定されたリージョンを使用します。ただし、クライアントのアクティブリージョンは個々のクライアントオブジェクトに明示的に設定できます。この方法でのリージョンの設定は、特定のサービスクライアントのグローバル設定よりも優先されます。代替リージョンは、クライアントのインスタンス化時に SDK に固有に指定されます (特定の SDK ガイドまたは SDK のコードベースを確認してください)。

この機能を設定するには、以下のように使用します。

`region` - 共有 AWS **`config`**ファイル設定, **`AWS_REGION`** - 環境変数, **`aws.region`** - JVM システムプロパティ: Java/Kotlin のみ

AWS リクエスト AWS リージョン に使用するデフォルトを指定します。このリージョンは、使用するリージョンが指定されていない SDK サービスリクエストに使用されます。

デフォルト値： NONE。この値は明示的に指定する必要があります。

有効な値:

- 「AWS 全般リファレンス」の「AWS サービスエンドポイント」に記載されているように、選択したサービスで使用できるどのリージョンコードでも指定できます。たとえば、この `us-east-1` 値により、エンドポイントは AWS リージョン 米国東部 (バージニア北部)に設定されます。
- `aws-global` は、 AWS Security Token Service (AWS STS) や Amazon Simple Storage Service (Amazon S3) などのリージョンエンドポイントに加えて、別のグローバルエンドポイントをサポートするサービスのグローバルエンドポイントを指定します。

`config` ファイルにこの値を設定する例を以下に示します。

```
[default]
region = us-west-2
```

Linux/macOS のコマンドラインによる環境変数の設定の例を以下に示します。

```
export AWS_REGION=us-west-2
```

Windows のコマンドラインによる環境変数の設定の例を以下に示します。

```
setx AWS_REGION us-west-2
```

ほとんどの SDK には、アプリケーションコード内からデフォルトリージョンを設定するための「設定」オブジェクトがあります。詳細については、特定の AWS SDK 開発者ガイドを参照してください。

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	はい	AWS CLI v2 は、 の値AWS_REGION の前に の値を使用します AWS_DEFAULT_REGION (両方の変数がチェックされます)。
AWS CLI v1	はい	AWS CLI v1 はこの目的のために という名前AWS_DEFAULT_REGION の環境変数を使用します。
SDK for C++	はい	
SDK for Go V2 (1.x)	はい	
SDK for Go 1.x (V1)	はい	共有 config ファイル設定を使用するには、設定ファイルからの読み込みを有効にする必要があります。「セッション」を参照してください。
SDK for Java 2.x	はい	

SDK	サポート	注意または詳細情報
SDK for Java 1.x	はい	
SDK for JavaScript 3.x	はい	
SDK for JavaScript 2.x	はい	
SDK for Kotlin	はい	
SDK for .NET 4.x	はい	
SDK for .NET 3.x	はい	
SDK for PHP 3.x	はい	
SDK for Python (Boto3)	はい	この SDK は、この目的のために AWS_DEFAULT_REGION と名付けられた環境変数を使用します。
SDK for Ruby 3.x	はい	
SDK for Rust	はい	
SDK for Swift	はい	
PowerShell V5 のツール	はい	

SDK	サポート	注意または詳細情報
PowerShell V4 のツール	はい	

AWS STS リージョンエンドポイント

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

AWS Security Token Service (AWS STS) は、グローバルサービスとリージョンサービスの両方で利用できます。 AWS SDKs と CLIs の中には、デフォルトでグローバルサービスエンドポイント (`https://sts.amazonaws.com`) を使用するものもあれば、リージョンサービスエンドポイント () を使用するものもありますhttps://sts.*{region_identifier}*.*{partition_domain}*。デフォルトでは有効になっているリージョンでは、 AWS STS グローバルエンドポイントへのリクエストは、リクエスト元のリージョンと同じリージョンで自動的に処理されます。オプトインリージョンでは、 AWS STS グローバルエンドポイントへのリクエストは AWS リージョン、単一の米国東部 (バージニア北部) によって処理されます。 AWS STS エンドポイントの詳細については、AWS Security Token Service 「 API リファレンス」の「エンドポイント」またはAWS Identity and Access Management 「 ユーザーガイドAWS STS 」の「 の管理 AWS リージョン」を参照してください。

可能な限りリージョンエンドポイントを使用し、 を設定することが AWS ベストプラクティスですAWS リージョン。商用以外のパーティションのお客様は、リージョンエンドポイントを使用する必要があります。すべての SDK とツールがこの設定をサポートしているわけではありませんが、グローバルエンドポイントとリージョンエンドポイントに関する動作はすべて定義されています。詳細については、次の「」セクションを参照してください。

> **Note**
>
> AWS は、回復力とパフォーマンスを向上させるためにデフォルトで有効になっているリージョンの AWS Security Token Service (AWS STS) グローバルエンドポイント (`https://sts.amazonaws.com`) を変更しました。グローバルエンドポイントへの AWS STS リクエ

ストは、ワークロード AWS リージョン と同じ で自動的に処理されます。これらの変更はオプトインリージョンにはデプロイされません。適切な AWS STS リージョンエンドポイントを使用することをお勧めします。詳細については、「AWS Identity and Access Management ユーザーガイド」の「AWS STS グローバルエンドポイントの変更」を参照してください。

この設定をサポートする SDK とツールでは、お客様は以下を使用して機能を設定できます。

sts_regional_endpoints - 共有 AWS **config**ファイル設定, **AWS_STS_REGIONAL_ENDPOINTS** - 環境変数

この設定は、SDK またはツールが AWS Security Token Service () との通信に使用するエンドポイントを決定する AWS のサービス 方法を指定しますAWS STS。

デフォルト値: `regional`。次の表の例外を参照してください。

> **Note**
>
> 2022 年 7 月以降にリリースされるすべての SDK メジャーバージョンは、デフォルトで `regional` に設定されます。新しい SDK メジャーバージョンでは、この設定が削除され、`regional` 動作が使用する可能性があります。この変更による将来的な影響を減らすため、可能な場合はアプリケーションで `regional` の使用を開始することをお勧めします。

有効な値： (推奨値：*`regional`*)

- **`legacy`** – グローバル AWS STS エンドポイント を使用しますsts.amazonaws.com。
- **`regional`** – SDK またはツールは、現在設定されているリージョンの AWS STS エンドポイントを常に使用します。たとえば、クライアントが を使用するように設定されている場合us-west-2、 へのすべての呼び出し AWS STS は、グローバルエンドポイントではなくsts.us-west-2.amazonaws.com、リージョンsts.amazonaws.comエンドポイント に対して行われます。この設定が有効なときにグローバルエンドポイントにリクエストを送信するには、リージョンを `aws-global` に設定します。

`config` ファイルに次の値を設定する例を以下に示します。

```
[default]
sts_regional_endpoints = regional
```

Linux/macOS のコマンドラインによる環境変数の設定の例を以下に示します。

```
export AWS_STS_REGIONAL_ENDPOINTS=regional
```

Windows のコマンドラインによる環境変数の設定の例を以下に示します。

```
setx AWS_STS_REGIONAL_ENDPOINTS regional
```

AWS SDKsとツールによるサポート

> **Note**
>
> 可能な限りリージョンエンドポイントを使用し、 を設定することが AWS ベストプラクティスですAWS リージョン。

次の表は、SDK またはツールの概要を示しています。

- サポートの設定: STS リージョンエンドポイントの共有 `config` ファイル変数と環境変数がサポートされるかどうか。
- デフォルト設定値: 設定がサポートされている場合のデフォルト値。
- デフォルトのサービスクライアントターゲット STS エンドポイント: 変更する設定が使用できない場合でも、クライアントが使用するデフォルトのエンドポイント。
- サービスクライアントのフォールバック動作: リージョンエンドポイントを使用することになっているが、リージョンが設定されていない場合の SDK の動作。この動作は、デフォルトでリージョンエンドポイントを使用しているか、設定で `regional` が選択されているかどうかは関係ありません。

表では以下の値も使用します。

- グローバルエンドポイント: `https://sts.amazonaws.com`。
- リージョンエンドポイント: アプリケーションが使用する設定済みの AWS リージョンに基づきます。
- **`us-east-1`** (リージョン): `us-east-1` リージョンエンドポイントを使用しますが、一般的なグローバルリクエストよりも長いセッショントークンを使用します。

SDK		デフォルト設定値	デフォルトのサービスクライアントターゲット STS エンドポイント	サービスクライアントのフォールバック動作	注意または詳細情報
AWS CLI v2	不可	該当なし	リージョンエンドポイント	グローバルエンドポイント	
AWS CLI v1	はい	legacy	グローバルエンドポイント	グローバルエンドポイント	
SDK for C++	不可	該当なし	リージョンエンドポイント	us-east-1 (リージョン)	
SDK for Go V2 (1.x)	不可	該当なし	リージョンエンドポイント	リクエストの失敗	
SDK for Go 1.x (V1)	はい	legacy	グローバルエンドポイント	グローバルエンドポイント	共有 config ファイル設定を使用するには、設定ファイルからの読み込みを有効にする必要があります。「セッション」を参照してください。
SDK for Java 2.x	不可	該当なし	リージョンエンドポイント	リクエストの失敗	リージョンが設定されていない場合、AssumeRole と AssumeRoleWithWebIdentity はグローバル STS エンドポイントを使用します。
SDK for Java 1.x	はい	legacy	グローバルエンドポイント	グローバルエンドポイント	
SDK for JavaScript 3.x	不可	該当なし	リージョンエンドポイント	us-east-1 (リージョン)	

SDK		デフォルト設定値	デフォルトのサービスクライアントターゲット STS エンドポイント	サービスクライアントのフォールバック動作	注意または詳細情報
SDK for JavaScript 2.x	はい	`legacy`	グローバルエンドポイント	グローバルエンドポイント	
SDK for Kotlin	不可	該当なし	リージョンエンドポイント	グローバルエンドポイント	
SDK for .NET 4.x	不可	該当なし	リージョンエンドポイント	`us-east-1` (リージョン)	
SDK for .NET 3.x	はい	`regional`	グローバルエンドポイント	グローバルエンドポイント	
SDK for PHP 3.x	はい	`regional`	グローバルエンドポイント	リクエストの失敗	
SDK for Python (Boto3)	はい	`regional`	グローバルエンドポイント	グローバルエンドポイント	
SDK for Ruby 3.x	はい	`regional`	リージョンエンドポイント	リクエストの失敗	
SDK for Rust	不可	該当なし	リージョンエンドポイント	リクエストの失敗	
SDK for Swift	不可	該当なし	リージョンエンドポイント	リクエストの失敗	
PowerShell V5 のツール	はい	`regional`	グローバルエンドポイント	グローバルエンドポイント	

SDK		デフォルト設定値	デフォルトのサービスクライアントターゲット STS エンドポイント	サービスクライアントのフォールバック動作	注意または詳細情報
PowerShell V4 用のツール	はい	regional	グローバルエンドポイント	グローバルエンドポイント	

Amazon S3 のデータ整合性保護

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

しばらくの間、 AWS SDKs Amazon Simple Storage Service にデータをアップロードするとき、または Amazon Simple Storage Service からデータをダウンロードするときに、データ整合性チェックをサポートしています。以前は、これらのチェックはオプトインされていました。現在、CRC32 や CRC64NVME などの CRC ベースのアルゴリズムを使用して、これらのチェックがデフォルトで有効になっています。各 SDK またはツールにはデフォルトのアルゴリズムがありますが、別のアルゴリズムを選択できます。必要に応じて、アップロード用に事前に計算されたチェックサムを引き続き手動で指定することもできます。アップロード、マルチパートアップロード、ダウンロード、暗号化モード間で一貫した動作により、クライアント側の整合性チェックが簡素化されます。

最新バージョンの AWS SDKs と は、アップロードごとに巡回冗長チェック (CRC) ベースのチェックサム AWS CLI を自動的に計算し、Amazon S3 に送信します。Amazon S3 はサーバー側のチェックサムを個別に計算し、指定された値で検証してから、オブジェクトとそのチェックサムをオブジェクトのメタデータに永続的に保存します。オブジェクトと共にメタデータにチェックサムを保存することで、オブジェクトのダウンロード時に同じチェックサムが自動的に返され、ダウンロードの検証にも使用することができます。オブジェクトのメタデータに保存されているチェックサムはいつでも確認できます。

チェックサムオペレーション、マルチパートアップロード、またはサポートされているチェックサムアルゴリズムのリストの詳細については、「Amazon Simple Storage Service ユーザーガイド」の「Amazon S3 でのオブジェクトの整合性のチェック」を参照してください。

マルチパートアップロード:

Amazon S3 は、単一パートアップロードとマルチパートアップロードにわたって一貫性のある完全なフルオブジェクトチェックサムをデベロッパーに提供します。

マルチパートでファイルをアップロードする場合、SDK は各パートのチェックサムを計算します。Amazon S3 は、これらのチェックサムを使用し、`UploadPart` API を通じて各パートの整合性を検証します。さらに、Amazon S3 は `CompleteMultipartUpload` API を呼び出すときにファイル全体のサイズとチェックサムを検証します。

SDK にマルチパートアップロードを支援する Amazon S3 Transfer Manager がある場合、チェックサムは、AWS SDKsとツールによるサポート テーブルにある SDK 固有のデフォルトアルゴリズムを使用してパートについて検証されます。`checksum_type` を `FULL_OBJECT` に設定するか、CRC64NVME アルゴリズムを使用することを選択することで、完全なオブジェクトチェックサムにオプトインできます。

古いバージョンの SDK または AWS CLIを使用している場合:

アプリケーションが 2024 年 12 月より前のバージョンの SDK またはツールを使用している場合でも、Amazon S3 は新しいオブジェクトに対して CRC64NVME チェックサムを計算し、今後の参照のためにオブジェクトのメタデータに保存します。後で、保存された CRC をユーザー側で計算された CRC と比較し、ネットワーク送信が正常であったことを確認できます。また、独自の事前計算されたチェックサムに `PutObject` または `UploadPart` リクエストを提供することで、整合性保護を手動で拡張することもできます。これは、古いバージョンでの対処のための標準的な手法です。

この機能を設定するには、以下のように使用します。

`request_checksum_calculation` - 共有 AWS **`config`**ファイル設定, **`AWS_REQUEST_CHECKSUM_CALCULATION`** - 環境変数, **`aws.requestChecksumCalculation`** - JVM システムプロパティ: Java/Kotlin のみ

> デフォルトでは、ユーザーはリクエストの送信時にリクエストのチェックサムを計算するようオプトインされます。ユーザーは、リクエストの構築の一環として、使用可能なチェックサムアルゴリズムのいずれかを選択できます。選択しない場合は、SDK 固有のデフォルトアルゴリズムが使用されます。各 SDK またはツールのデフォルトアルゴリズムについては、AWS SDKsとツールによるサポート テーブルを参照してください。

デフォルト値: WHEN_SUPPORTED

有効な値:

- **WHEN_SUPPORTED** – チェックサム検証は、Amazon S3 へのデータ転送など、API オペレーションでサポートされている場合、すべてのリクエストペイロードで実行されます。
- **WHEN_REQUIRED** – チェックサム検証は、API オペレーションで必要な場合にのみ実行されます。

response_checksum_validation - 共有 AWS **config**ファイル設定, **AWS_RESPONSE_CHECKSUM_VALIDATION** - 環境変数, **aws.responseChecksumValidation** - JVM システムプロパティ: Java/Kotlin のみ

デフォルトでは、ユーザーはリクエストの送信時にレスポンスのチェックサムの検証にオプトインされます。チェックサムはレスポンスペイロードに対して計算され、チェックサムのレスポンスヘッダーと比較されます。チェックサムの検証に失敗すると、ペイロードの読み取り時にエラーが発生します。

チェックサムのレスポンスヘッダーには、チェックサムのアルゴリズムも示されます。Amazon S3 クライアントは、チェックサムをサポートするすべての Amazon S3 API オペレーションのレスポンスのチェックサム検証を試みます。ただし、SDK が指定されたチェックサムアルゴリズムを実装していない場合、この検証はスキップされます。

デフォルト値: WHEN_SUPPORTED

有効な値:

- **WHEN_SUPPORTED** – チェックサム検証は、Amazon S3 へのデータ転送など、API オペレーションでサポートされている場合、すべてのレスポンスペイロードで実行されます。
- **WHEN_REQUIRED** – チェックサム検証は、API オペレーションでサポートされ、呼び出し元がオペレーションのチェックサムを明示的に有効にしている場合にのみ実行されます。例えば、Amazon S3 `GetObject` API が呼び出され、`ChecksumMode` パラメータが有効に設定されている場合などです。

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

> **Note**
>
> 次の表では、「CRT」は AWS 共通ランタイム (CRT) ライブラリ を参照しており、プロジェクトへの依存関係の追加が必要になる場合があります。

SDK	サポート	デフォルトのチェックサムアルゴリズム	サポートされているチェックサムアルゴリズム	注意または詳細情報
AWS CLI v2	はい	CRC64NVME	CRC64NVME、CRC32、CRC32C、SHA1、SHA256	AWS CLI v1 の場合、デフォルトのアルゴリズムとサポートされているアルゴリズムは Python (Boto3) と同じになります。
SDK for C++	はい	CRC64NVME	CRC64NVME、CRC32、CRC32C、SHA1、SHA256	
SDK for Go V2 (1.x)	はい	CRC32	CRC64NVME、CRC32、CRC32C、SHA1、SHA256	
SDK for Go 1.x (V1)	不可			
SDK for Java 2.x	はい	CRC32	CRC64NVME (CRT のみ)、CRC32、CRC32C、SHA1、SHA256	
SDK for Java 1.x	不可			

SDK	サポート	デフォルトのチェックサムアルゴリズム	サポートされているチェックサムアルゴリズム	注意または詳細情報
SDK for JavaScript 3.x	はい	CRC32	CRC32、CRC32C、SHA1、SHA256	
SDK for JavaScript 2.x	不可			
SDK for Kotlin	はい	CRC32	CRC32、CRC32C、SHA1、SHA256	
SDK for .NET 4.x	はい	CRC32	CRC32、CRC32C、SHA1、SHA256	
SDK for .NET 3.x	はい	CRC32	CRC32、CRC32C、SHA1、SHA256	
SDK for PHP 3.x	はい	CRC32	CRC32、CRC32C (CRT 経由のみ)、SHA1、SHA256	CRC32C を使用するには `awscrt` 拡張機能が必要です。
SDK for Python (Boto3)	はい	CRC32	CRC64NVME (CRT のみ)、CRC32、CRC32C (CRT のみ)、SHA1、SHA256	
SDK for Ruby 3.x	はい	CRC32	CRC64NVME (CRT のみ)、CRC32、CRC32C (CRT のみ)、SHA1、SHA256	
SDK for Rust	はい	CRC32	CRC64NVME、CRC32、CRC32C、SHA1、SHA256	

SDK	サポート	デフォルトのチェックサムアルゴリズム	サポートされているチェックサムアルゴリズム	注意または詳細情報
SDK for Swift	はい	CRC32	CRC64NVME、CRC32、CRC32C、SHA1、SHA256	すべてのアルゴリズムに必要な CRT 依存関係。
PowerShell V5 のツール	はい	CRC32	CRC32、CRC32C、SHA1、SHA256	
PowerShell V4 のツール	はい	CRC32	CRC32、CRC32C、SHA1、SHA256	

デュアルスタックと FIPS エンドポイント

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

この機能を設定するには、以下のように使用します。

use_dualstack_endpoint - 共有 AWS **config**ファイル設定, **AWS_USE_DUALSTACK_ENDPOINT** - 環境変数, **aws.useDualstackEndpoint** - JVM システムプロパティ: Java/Kotlin のみ

SDK がデュアルスタックのエンドポイントにリクエストを送信するかどうかをオンまたはオフにします。IPv4 と IPv6 の両方のトラフィックをサポートするデュアルスタックエンドポイントの詳細については、「Amazon Simple Storage Service ユーザーガイド」の「Amazon S3 デュアルスタックエンドポイントの使用」を参照してください。デュアルスタックのエンドポイントは、一部のリージョンでは一部のサービスで利用できます。

デフォルト値: `false`

有効な値:

- **true** – SDK またはツールは、デュアルスタックエンドポイントを使用してネットワークリクエストを行おうと試みます。サービスや AWS リージョンにデュアルスタックエンドポイントが存在しない場合、リクエストは失敗します。
- **false** – SDK またはツールは、ネットワークリクエストを行うためにデュアルスタックエンドポイントを使用しません。

use_fips_endpoint - 共有 AWS **config**ファイル設定, **AWS_USE_FIPS_ENDPOINT** - 環境変数, **aws.useFipsEndpoint** - JVM システムプロパティ: Java/Kotlin のみ

SDK またはツールが FIPS 準拠のエンドポイントにリクエストを送信するかどうかをオンまたはオフにします。連邦情報処理標準 (FIPS) は、データとその暗号化に関する米国政府のセキュリティ要件をまとめたものです。政府機関、パートナー、および連邦政府との取引を希望する者は、FIPS ガイドラインを遵守する必要があります。標準 AWS エンドポイントとは異なり、FIPS エンドポイントは FIPS 140 に対して検証された TLS ソフトウェアライブラリを使用します。この設定が有効で、 のサービスに FIPS エンドポイントが存在しない場合 AWS リージョン、 AWS 呼び出しが失敗する可能性があります。 サービス固有のエンドポイントおよび `--endpoint-url`のオプションは、この設定を AWS Command Line Interface 上書きします。

で FIPS エンドポイントを指定するその他の方法の詳細については AWS リージョン、「サービス別の FIPS エンドポイント」を参照してください。Amazon Elastic Compute Cloud サービスのエンドポイントの詳細については、「Amazon EC2 API リファレンス」の「デュアルスタック (IPv4 と IPv6) エンドポイント」を参照してください。

デフォルト値: `false`

有効な値:

- **true** – SDK またはツールが FIPS 準拠のエンドポイントにリクエストを送信します。
- **false** – SDK またはツールが FIPS 準拠のエンドポイントにリクエストを送信しません。

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	はい	
SDK for C++	はい	
SDK for Go V2 (1.x)	はい	
SDK for Go 1.x (V1)	はい	共有 config ファイル設定を使用するには、設定ファイルからの読み込みを有効にする必要があります。「セッション」を参照してください。
SDK for Java 2.x	はい	
SDK for Java 1.x	不可	
SDK for JavaScript 3.x	はい	
SDK for JavaScript 2.x	はい	
SDK for Kotlin	はい	
SDK for .NET 4.x	はい	
SDK for .NET 3.x	はい	

SDK	サポート	注意または詳細情報
SDK for PHP 3.x	はい	
SDK for Python (Boto3)	はい	
SDK for Ruby 3.x	はい	
SDK for Rust	はい	
SDK for Swift	はい	
PowerShell V5 のツール	はい	
PowerShell V4 用のツール	はい	

エンドポイント検出

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

SDKsエンドポイント検出を使用してサービスエンドポイント (さまざまなリソースにアクセスするための URLs) にアクセスしますが、 が必要に応じて URLs を変更 AWS するための柔軟性は維持されます。これにより、コードは新しいエンドポイントを自動的に検出できます。一部のサービスには固定エンドポイントはありません。代わりに、最初にエンドポイントを取得するようにリクエストすることで、ランタイムに利用可能なエンドポイントを取得します。使用可能なエンドポイントを取

得したら、コードはそのエンドポイントを使用して他の操作にアクセスします。たとえば、Amazon Timestream の場合、SDK は利用可能なエンドポイントを取得する `DescribeEndpoints` リクエストを行い、それらのエンドポイントを使用して `CreateDatabase` や `CreateTable` などの特定の操作を実行します。

この機能を設定するには、以下のように使用します。

`endpoint_discovery_enabled` - 共有 AWS **`config`**ファイル設定, **`AWS_ENABLE_ENDPOINT_DISCOVERY`** - 環境変数, **`aws.endpointDiscoveryEnabled`** - JVM システムプロパティ: Java/Kotlin のみ, コード内で値を直接設定するには、使用している SDK に直接問い合わせてください。

DynamoDB のエンドポイント検出を有効または無効にします。

エンドポイント検出は Timestream では必須で、Amazon DynamoDB ではオプションです。デフォルトでは、この設定はサービスがエンドポイント検出を必要とするかどうかに応じて、`true` または `false` のどちらかになります。Timestream リクエストのデフォルトは `true`、Amazon DynamoDB リクエストのデフォルトは `false` です。

有効な値:

- **`true`** – エンドポイント検出がオプションであるサービスの場合、SDK はエンドポイントを自動的に検出しようとする必要があります。
- **`false`** – エンドポイント検出がオプションであるサービスの場合、SDK はエンドポイントを自動的に検出しようとする必要がありません。

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	はい	

SDK	サポート	注意または詳細情報
SDK for C++	はい	
SDK for Go V2 (1.x)	はい	
SDK for Go 1.x (V1)	はい	共有 config ファイル設定を使用するには、設定ファイルからの読み込みを有効にする必要があります。「セッション」を参照してください。
SDK for Java 2.x	はい	SDK for Java 2.x は、環境変数名に AWS_ENDPOINT_DISCOVERY_ENABLED を使用します。
SDK for Java 1.x	部分的	JVM システムプロパティはサポートされていません。
SDK for JavaScript 3.x	はい	
SDK for JavaScript 2.x	はい	
SDK for Kotlin	はい	
SDK for .NET 4.x	はい	
SDK for .NET 3.x	はい	
SDK for PHP 3.x	はい	

SDK	サポート	注意または詳細情報
SDK for Python (Boto3)	はい	
SDK for Ruby 3.x	はい	
SDK for Rust	部分的	Timestream でのみサポートされます。
SDK for Swift	不可	
PowerShell V5 用のツール	はい	
PowerShell V4 のツール	はい	

一般設定

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

SDK は SDK の全体的な動作を設定する一般設定の一部をサポートします。

この機能を設定するには、以下のように使用します。

`api_versions` - 共有 AWS **`config`**ファイル設定

一部の AWS サービスでは、下位互換性をサポートするために複数の API バージョンが維持されています。デフォルトでは、SDK と AWS CLI オペレーションは最新の API バージョンを使用

します。リクエストに特定の API バージョンを使用することを要求するには、プロファイルに api_versions 設定を含めてください。

デフォルト値： なし。(SDK では最新の API バージョンが使用されます。)

有効な値: これはネストされた設定で、その後に 1 つ以上のインデントされた行が続き、それぞれが 1 つの AWS サービスと使用する API バージョンを識別します。利用可能な API バージョンについては、 AWS サービスのドキュメントを参照してください。

この例では、 config ファイル内の 2 つの AWS サービスに特定の API バージョンを設定します。これらの API バージョンは、この設定を含むプロファイルで実行するコマンドにのみ使用されます。その他のサービスのコマンドは、そのサービスの API の最新バージョンを使用します。

```
api_versions =
    ec2 = 2015-03-01
    cloudfront = 2015-09-017
```

ca_bundle - 共有 AWS **config**ファイル設定, **AWS_CA_BUNDLE** - 環境変数

SSL/TLS 接続を確立するときに使用するカスタム証明書バンドル (拡張子 .pem のファイル) へのパスを指定します。

デフォルト値： なし

有効な値： フルパスまたはベースファイル名を指定します。ベースファイル名がある場合、システムは PATH 環境変数で指定されたフォルダー内でプログラムを検索しようとします。

config ファイルにこの値を設定する例を以下に示します。

```
[default]
ca_bundle = dev/apps/ca-certs/cabundle-2019mar05.pem
```

オペレーティングシステムのパスの処理方法やパス文字のエスケープ方法が異なるため、Windows の config ファイルでこの値を設定する例を次に示します。

```
[default]
ca_bundle = C:\\Users\\username\\.aws\\aws-custom-bundle.pem
```

Linux/macOS のコマンドラインによる環境変数の設定の例を以下に示します。

```
export AWS_CA_BUNDLE=/dev/apps/ca-certs/cabundle-2019mar05.pem
```

Windows のコマンドラインによる環境変数の設定の例を以下に示します。

```
setx AWS_CA_BUNDLE C:\dev\apps\ca-certs\cabundle-2019mar05.pem
```

output - 共有 AWS **config**ファイル設定

およびその他の AWS SDKs AWS CLI およびツールでの結果のフォーマット方法を指定します。

デフォルト値: `json`

有効な値:

- `json` - 出力は JSON 文字列としてフォーマットされます。
- **`yaml`** - 出力は YAML 文字列としてフォーマットされます。
- https://docs.aws.amazon.com/cli/latest/userguide/cli-usage-output-format.html#yaml-stream-output - 出力はストリームされ、YAML 文字列としてフォーマットされます。ストリーミングにより、大きなデータタイプの処理を高速化できます。
- `text` - 出力は、複数行のタブ区切りの文字列値としてフォーマットされます。これは、`grep`、`sed`、または `awk` などのテキストプロセッサに出力を渡すのに役立ちます。
- `table` - 出力は、テーブルとしてフォーマットされ、文字の「+|-」を使用してセルの境界を形成します。通常、情報は他の形式よりも読みやすい「わかりやすい」形式で表示されますが、プログラムとしては役立ちません。

parameter_validation - 共有 AWS **config**ファイル設定

AWS サービスエンドポイントに送信する前に、SDK またはツールがコマンドラインパラメータの検証を試行するかどうかを指定します。

デフォルト値: `true`

有効な値:

- **`true`**-デフォルト。SDK またはツールは、コマンドラインパラメータのクライアント側検証を実行します。これにより、SDK またはツールはパラメーターが有効であることを確認し、エラーを検出できます。SDK またはツールは、 AWS サービスエンドポイントにリクエストを送信する前に、有効でないリクエストを拒否できます。
- **`false`** – SDK またはツールは、 AWS サービスエンドポイントに送信する前にコマンドラインパラメータを検証しません。 AWS サービスエンドポイントは、すべてのリクエストを検証し、無効なリクエストを拒否する責任があります。

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	部分的	`api_versions` はサポートされていません。
SDK for C++	はい	
SDK for Go V2 (1.x)	部分的	`api_versions` および `parameter_validation` はサポートされていません。
SDK for Go 1.x (V1)	部分的	`api_versions` および `parameter_validation` はサポートされていません。共有 `config` ファイル設定を使用するには、設定ファイルからの読み込みを有効にする必要があります。「セッション」を参照してください。
SDK for Java 2.x	不可	
SDK for Java 1.x	不可	
SDK for JavaScript 3.x	はい	
SDK for JavaScript 2.x	はい	

SDK	サポート	注意または詳細情報
SDK for Kotlin	不可	
SDK for .NET 4.x	不可	
SDK for .NET 3.x	不可	
SDK for PHP 3.x	はい	
SDK for Python (Boto3)	はい	
SDK for Ruby 3.x	はい	
SDK for Rust	不可	
SDK for Swift	不可	
PowerShell V5 用のツール	不可	
PowerShell V4 のツール	不可	

ホストプレフィックスインジェクション

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

ホストプレフィックスインジェクションは、 AWS SDKsが特定の API オペレーションのサービスエンドポイントのホスト名にプレフィックスを自動的に付加する機能です。このプレフィックスは、静的文字列でも、リクエストパラメータのデータを含む動的な値でもかまいません。

例えば、Amazon Simple Storage Service を使用して Amazon S3 オブジェクトまたはバケットに対してアクションを実行する場合、SDK は最終的な API エンドポイントでバケット名と AWS アカウント ID を置き換えます。

この動作は通常の AWS サービスエンドポイントに必要ですが、VPC エンドポイントやローカルテストツールなどのカスタムエンドポイントを使用すると問題が発生する可能性があります。このような場合は、ホストプレフィックスインジェクションの無効化が必要になる場合があります。

この機能を設定するには、以下のように使用します。

`disable_host_prefix_injection` - 共有 AWS **`config`**ファイル設定, **`AWS_DISABLE_HOST_PREFIX_INJECTION`** - 環境変数, **`aws.disableHostPrefixInjection`** - JVM システムプロパティ: Java/Kotlin のみ

この設定は、SDK またはツールが SDK のクライアントオブジェクトまたは変数で定義されているホストプレフィックスを付加してエンドポイントホスト名を変更するかどうかを制御します。

デフォルト値: `false`

有効な値:

- **`true`** - ホストプレフィックスインジェクションを無効にします。SDK はエンドポイントのホスト名を変更しません。
- **`false`** – ホストプレフィックスインジェクションを有効にします。SDK は、エンドポイントのホスト名の先頭にホストプレフィックスを付加します。

`config` ファイルにこの値を設定する例を以下に示します。

```
[default]
disable_host_prefix_injection = true
```

Linux/macOS のコマンドラインによる環境変数の設定の例を以下に示します。

```
export AWS_DISABLE_HOST_PREFIX_INJECTION=true
```

Windows のコマンドラインによる環境変数の設定の例を以下に示します。

```
setx AWS_DISABLE_HOST_PREFIX_INJECTION true
```

ホストプレフィックスインジェクションの例

次の例の表は、ホストプレフィックスインジェクションが有効になっている場合と無効になっている場合に SDK が最後のエンドポイントを変更する方法を示しています。

- ホストプレフィックス: SDK のクライアントオブジェクトまたはコード内の変数に設定されたホストプレフィックスプロパティ文字列のテンプレート。
- 入力: SDK のクライアントオブジェクトまたはコード内の変数に設定された追加の入力。
- クライアントエンドポイント: クライアントの派生エンドポイント。
- 設定値: 前の設定の解決された値。
- 結果のエンドポイント: SDK クライアントが API コールを行うために使用する結果のエンドポイント。

ホストプレフィックス	入力	クライアントエンドポイント	設定値	結果のエンドポイント
"data."	{}	"https://service.us-west-2.amazonaws.com"	false	"https://data.service.us-west-2.amazonaws.com"
"{Bucket}-{AccountId}."	Bucket: "amzn-s3-demo-bucket1", AccountId:"123456789012"	"https://service.us-west-2.amazonaws.com"	false	"https://amzn-s3-demo-bucket1-123456789012.se

ホストプレフィックス	入力	クライアントエンドポイント	設定値	結果のエンドポイント
				rvice.us-west-2.amazonaws.com"
"data."	{}	"https://override.us-west-2.amazonaws.com" (as an override endpoint)	true	"https://override.us-west-2.amazonaws.com"

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	はい	
SDK for C++	不可	設定はサポートされていませんが、`enableHostPrefixInjection` を使用してクライアントのコードで設定できます。
SDK for Go V2 (1.x)	不可	ミドルウェアを使用して無効にできます。
SDK for Go 1.x (V1)	不可	

SDK	サポート	注意または詳細情報
SDK for Java 2.x	不可	設定はサポートされていませんが、SdkAdvancedClientOption.DISABLE_HOST_PREFIX_INJECTION を使用してクライアントのコードで設定できます。
SDK for Java 1.x	不可	設定はサポートされていませんが、withDisableHostPrefixInjection を使用してクライアントのコードで設定できます。
SDK for JavaScript 3.x	不可	設定はサポートされていませんが、disableHostPrefix を使用してクライアントのコードで設定できます。
SDK for JavaScript 2.x	不可	設定はサポートされていませんが、hostPrefixEnabled を使用してクライアントのコードで設定できます。
SDK for Kotlin	不可	
SDK for .NET 4.x	不可	設定はサポートされていませんが、DisableHostPrefixInjection を使用してクライアントのコードで設定できます。
SDK for .NET 3.x	不可	設定はサポートされていませんが、DisableHostPrefixInjection を使用してクライアントのコードで設定できます。
SDK for PHP 3.x	不可	設定はサポートされていませんが、disable_host_prefix_injection を使用してクライアントのコードで設定できます。
SDK for Python (Boto3)	はい	inject_host_prefix を使用してクライアントのコードで設定できます。
SDK for Ruby 3.x	不可	設定はサポートされていませんが、disable_host_prefix_injection を使用してクライアントのコードで設定できます。

SDK	サポート	注意または詳細情報
SDK for Rust	不可	
SDK for Swift	不可	
PowerShell V5 のツール	不可	設定はサポートされていませんが、パラメータ `-ClientConfig @{DisableHostPrefixInjection = $true}` を使用して特定のコマンドレットに含めることができます。
PowerShell V4 のツール	不可	設定はサポートされていませんが、パラメータ `-ClientConfig @{DisableHostPrefixInjection = $true}` を使用して特定のコマンドレットに含めることができます。

IMDS クライアント

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

SDK は、セッション指向リクエストを使用してインスタンスメタデータサービスのバージョン 2 (IMDSv2) クライアントを実装します。IMDSv2 の詳細については、「Amazon EC2 ユーザーガイド」の「インスタンスメタデータサービスを使用してインスタンスメタデータにアクセスする」を参照してください。IMDS クライアントは、SDK コードベースにあるクライアント設定オブジェクトを使用して設定できます。

この機能を設定するには、以下のように使用します。

`retries` - クライアント設定オブジェクトメンバー

リクエストが失敗した場合の追加再試行の回数。

デフォルト値： 3

有効な値： 0 より大きい数値。

port - クライアント設定オブジェクトメンバー

エンドポイントのポート。

デフォルト値： 80

有効な値： 数値。

token_ttl - クライアント設定オブジェクトメンバー

トークンの TTL。

デフォルト値：21,600 秒 (6 時間、割り当てられた最大時間)。

有効な値： 数値。

endpoint - クライアント設定オブジェクトメンバー

IMDS のエンドポイント。

デフォルト値： `endpoint_mode` が `IPv4` に等しい場合、デフォルトエンドポイントは `http://169.254.169.254` です。`endpoint_mode` が `IPv6` に等しい場合、デフォルトのエンドポイントは `http://[fd00:ec2::254]` です。

有効な値： 有効な URI。

大半の SDK では以下のオペレーションがサポートされています。詳細については、特定の SDK コードベースを参照してください。

endpoint_mode - クライアント設定オブジェクトメンバー

IMDS のエンドポイントモード。

デフォルト値: `IPv4`

有効な値:`IPv4`、`IPv6`|

http_open_timeout - クライアント設定オブジェクトメンバー (名前は異なる場合があります)

接続が開くのを待つ秒数。

デフォルト値： 1 秒。

有効な値： 0 より大きい数値。

http_read_timeout - クライアント設定オブジェクトメンバー (名前は異なる場合があります)

1 つのデータチャンクが読み取られるまでの秒数。

デフォルト値： 1 秒。

有効な値： 0 より大きい数値。

http_debug_output - クライアント設定オブジェクトメンバー (名前は異なる場合があります)

デバッグ用の出力ストリームを設定します。

デフォルト値： なし。

有効な値： STDOUT のような有効な I/O ストリーム。

backoff - クライアント設定オブジェクトメンバー (名前は異なる場合があります)

リトライ間またはお客様が用意したバックオフ関数を呼び出すまでの間にスリープする秒数。これは、デフォルトのエクスポネンシャルバックオフ戦略を使用するよう置き換えます。

デフォルト値： サービスによって異なります。

有効な値： SDK によって異なります。数値でも、カスタム関数の呼び出しでもかまいません。

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	はい	
SDK for C++	不可	
SDK for Go V2 (1.x)	はい	

SDK	サポート	注意または詳細情報
SDK for Go 1.x (V1)	はい	
SDK for Java 2.x	はい	
SDK for Java 1.x	はい	
SDK for JavaScript 3.x	はい	
SDK for JavaScript 2.x	はい	
SDK for Kotlin	不可	
SDK for .NET 4.x	はい	
SDK for .NET 3.x	はい	
SDK for PHP 3.x	はい	
SDK for Python (Boto3)	はい	
SDK for Ruby 3.x	はい	
SDK for Rust	はい	

SDK	サポート	注意または詳細情報
SDK for Swift	はい	
PowerShell V5 のツール	はい	
PowerShell V4 のツール	はい	

再試行動作

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

再試行動作には、SDK が AWS のサービスへのリクエストによる障害からの回復を試みるかに関する設定が含まれます。

この機能を設定するには、以下のように使用します。

`retry_mode` - 共有 AWS **`config`**ファイル設定, **`AWS_RETRY_MODE`** - 環境変数, **`aws.retryMode`** - JVM システムプロパティ: Java/Kotlin のみ

SDK または開発者ツールが再試行を試みる方法を指定します。

デフォルト値: この値は SDK に固有です。特定の SDK ガイドまたは SDK のコードベースでデフォルトの `retry_mode` を確認してください。

有効な値:

- `standard` – (推奨) AWS SDKs 全体で推奨される再試行ルールのセット。このモードには、再試行されるエラーの標準セットが含まれており、再試行回数を自動的に調整して可用性と安定性を最大化します。このモードは、マルチテナントアプリケーションで安全に使用できま

す。max_attempts が明示的に設定されていない限り、このモードでのデフォルトの最大試行回数は 3 回です。

- adaptive – 標準モードの機能やクライアント側のレートの自動制限を含む、特殊なユースケースにのみ適した再試行モード。この再試行モードは、アプリケーションテナントを慎重に分離しない限り、マルチテナントアプリケーションにはお勧めしません。詳細については「standard と adaptive の再試行モードを選択する」を参照してください。このモードは実験段階であり、将来的に動作が変更される可能性があります。
- legacy – (非推奨) ご使用の SDK に固有 (特定の SDK ガイドまたは SDK のコードベースを確認してください)。

max_attempts - 共有 AWS **config**ファイル設定, **AWS_MAX_ATTEMPTS** - 環境変数, **aws.maxAttempts** - JVM システムプロパティ: Java/Kotlin のみ

1 回のリクエストで行う最大試行回数を指定します。

デフォルト値：この値が指定されていない場合、デフォルトは retry_mode の設定の値によって異なります。

- retry_mode が legacy の場合 – SDK 固有のデフォルト値を使用します (max_attempts デフォルトについては、特定の SDK ガイドまたは SDK のコードベースを確認してください)。
- retry_mode が standard の場合 – 3 回試行します。
- retry_mode が adaptive の場合 – 3 回試行します。

有効な値： 0 より大きい数値。

standard と adaptive の再試行モードを選択する

使用状況が adaptive に適していることが確実でない限り、standard 再試行モードを使用することをお勧めします。

> **Note**
>
> adaptive モードは、バックエンドサービスがリクエストをスロットルする範囲に基づいてクライアントをプールしていることを前提としています。これを行わないと、両方のリソースに同じクライアントを使用している場合、1 つのリソースのスロットリングにより、無関係なリソースのリクエストが遅延する可能性があります。

標準	アダプティブ
アプリケーションのユースケース: すべて。	アプリケーションのユースケース: 1. レイテンシーの影響を受けません。 2. クライアントは 1 つのリソースにのみアクセスするか、アクセスするサービスリソースによってクライアントを個別にプールするロジックを提供します。
サーキットブレークをサポートして、停止中に SDK が再試行するのを防ぎます。	サーキットブレークをサポートして、停止中に SDK が再試行するのを防ぎます。
障害発生時にジッターされたエクスポネンシャルバックオフを使用します。	動的バックオフ期間を使用して、レイテンシーが増加する可能性と引き換えに、失敗したリクエストの数を最小限に抑えるよう試みます。
最初のリクエストの試行が遅延することはなく、再試行のみが遅延します。	最初のリクエスト試行をスロットルまたは遅延できます。

`adaptive` モードを使用する場合、アプリケーションはスロットルされる可能性のある各リソースを中心に設計されたクライアントを構築する必要があります。この場合、リソースは、それぞれについて考えるよりも細かく調整されます AWS のサービス。 は、リクエストを調整するために使用する追加のディメンションを持つ AWS のサービス ことができます。Amazon DynamoDB サービスを例として使用しましょう。DynamoDB は、 AWS リージョン とアクセスされるテーブルを使用してリクエストを調整します。つまり、コードがアクセスしている 1 つのテーブルは、他のテーブルよりもスロットルされる可能性があります。コードが同じクライアントを使用してすべてのテーブルにアクセスし、それらのテーブルのいずれかへのリクエストがスロットルされた場合、アダプティブ再試行モードでは、すべてのテーブルのリクエストレートが低下します。コードは、リージョンとテーブルのペアごとに 1 つのクライアントを持つように設計する必要があります。`adaptive` モードの使用時に予期しないレイテンシーが発生した場合は、使用しているサービスの特定の AWS ドキュメントガイドを参照してください。

再試行モードの実装の詳細

AWS SDKsトークンバケットを使用して、リクエストを再試行するかどうか、およびリクエストの送信速度 (adaptive再試行モードの場合) を決定します。SDK では、再試行トークンバケットとリクエストレートトークンバケットの 2 つのトークンバケットが使用されます。

- 再試行トークンバケットは、SDK が停止中にアップストリームサービスとダウンストリームサービスを保護するために再試行を一時的に無効にする必要があるかどうかを判断するために使用されます。再試行する前にバケットからトークンが取得され、リクエストが成功するとバケットにトークンが返されます。再試行時にバケットが空の場合、SDK はリクエストを再試行しません。
- リクエストレートトークンバケットは adaptive 再試行モードでのみ使用され、リクエストの送信レートを決定します。リクエストが送信される前にバケットからトークンが取得され、サービスによって返されるスロットリングレスポンスに基づいて動的に決定されたレートでバケットにトークンが返されます。

以下は、standard と adaptive 再試行モードの両方の大まかな擬似コードです。

```
MakeSDKRequest() {
  attempts = 0
  loop {
    GetSendToken()
    response = SendHTTPRequest()
    RequestBookkeeping(response)
    if not Retryable(response)
      return response
    attempts += 1
    if attempts >= MAX_ATTEMPTS:
      return response
    if not HasRetryQuota(response)
      return response
    delay = ExponentialBackoff(attempts)
    sleep(delay)
  }
}
```

擬似コードで使用されるコンポーネントの詳細は次のとおりです。

GetSendToken:

このステップは adaptive 再試行モードでのみ使用されます。このステップでは、リクエストレートトークンバケットからトークンを取得します。トークンが使用できない場合、トークンが利用可能になるまで待機します。SDK には、待機する代わりにリクエストを失敗させる設定オプションが用意されている場合があります。バケット内のトークンは、クライアントが受信したスロットリングレスポンスの数に基づいて動的に決定されるレートで補充されます。

SendHTTPRequest:

このステップでは、リクエストを に送信します AWS。 AWS SDKsは、HTTP リクエストを行うときに接続プールを使用して既存の接続を再利用する HTTP ライブラリを使用します。一般的に、スロットリングエラーが原因でリクエストが失敗した場合、接続は再利用されますが、一時的なエラーが原因でリクエストが失敗した場合は再利用されません。

RequestBookkeeping:

リクエストが成功すると、トークンがトークンバケットに追加されます。adaptive 再試行モードの場合のみ、リクエストレートトークンバケットのフィルレートは、受信したレスポンスのタイプに基づいて更新されます。

Retryable:

このステップでは、以下に基づいて応答を再試行できるかどうかを判断します。

- HTTP ステータスコード 。
- サービスから返されたエラーコード。
- 接続エラーとは、SDK が受信したエラーの中で、サービスからの HTTP 応答が受信されないすべてのエラーを指します。

一時的なエラー (HTTP ステータスコード 400、408、500、502、503、504) とスロットリングエラー (HTTP ステータスコード 400、403、429、502、503、509) はすべて再試行される可能性があります。SDK の再試行動作は、エラーコードまたはサービスからのその他のデータと組み合わせて決定されます。

MAX_ATTEMPTS:

デフォルトの最大試行回数は、max_attempts 設定によって上書きされない限り、retry_mode 設定によって設定されます。

HasRetryQuota

このステップでは、再試行トークンバケットからトークンを取得します。再試行トークンバケットが空の場合、リクエストは再試行されません。

ExponentialBackoff

再試行可能なエラーの場合、再試行遅延は台形型エクスポネンシャルバックオフを使用して計算されます。SDK はジッター付きの切り捨て二進エクスポネンシャルバックオフを使用します。次のアルゴリズムは、i リクエストに対する応答の休止時間（秒単位）がどのように定義されているかを示しています。

```
seconds_to_sleep_i = min(b*r^i, MAX_BACKOFF)
```

前述のアルゴリズムでは、以下の値が適用されます。

`b = random number within the range of: 0 <= b <= 1`

`r = 2`

ほとんどの SDK では `MAX_BACKOFF = 20 seconds` です。確認のため、特定の SDK ガイドまたはソースコードを参照してください。

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	はい	
SDK for C++	はい	
SDK for Go V2 (1.x)	はい	

SDK	サポート	注意または詳細情報
SDK for Go 1.x (V1)	不可	
SDK for Java 2.x	はい	
SDK for Java 1.x	はい	JVM システムプロパティ: `aws.maxAttempts` の代わりに `com.amazonaws.sdk.maxAttempts` を使用し、`aws.retryMode` の代わりに `com.amazonaws.sdk.retryMode` を使用します。
SDK for JavaScript 3.x	はい	
SDK for JavaScript 2.x	不可	最大再試行回数、ジッターを伴うエクスポネンシャルバックオフ、再試行バックオフのカスタムメソッドのオプションをサポートします。
SDK for Kotlin	はい	
SDK for .NET 4.x	はい	
SDK for .NET 3.x	はい	
SDK for PHP 3.x	はい	
SDK for Python (Boto3)	はい	
SDK for Ruby 3.x	はい	

SDK	サポート	注意または詳細情報
SDK for Rust	はい	
SDK for Swift	はい	
PowerShell V5 用のツール	はい	
PowerShell V4 用のツール	はい	

リクエスト圧縮

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

AWS SDKsとツールは、圧縮されたペイロードの受信 AWS のサービス をサポートする にリクエストを送信するときに、ペイロードを自動的に圧縮できます。サービスに送信する前にクライアントでペイロードを圧縮すると、サービスにデータを送信するために必要なリクエストの総数と帯域幅が減り、ペイロードサイズに対するサービスの制限を理由として失敗するリクエストも減る可能性があります。圧縮では、SDK またはツールは、サービスと SDK の両方によってサポートされるエンコーディングアルゴリズムを選択します。ただし、可能なエンコーディングの現在のリストは gzip のみで構成されていますが、将来的には拡張される可能性があります。

リクエスト圧縮は、アプリケーションが Amazon CloudWatch を利用している場合に特に役立ちます。CloudWatch は、ログ、メトリクス、イベントの形式でモニタリングおよび運用データを収集するモニタリングおよびオブザーバビリティサービスです。圧縮をサポートするサービスオペレーションの一例として、CloudWatch の PutMetricDataAPI メソッドを挙げることができます。

この機能を設定するには、以下のように使用します。

disable_request_compression - 共有 AWS **config**ファイル設定, **AWS_DISABLE_REQUEST_COMPRESSION** - 環境変数, **aws.disableRequestCompression** - JVM システムプロパティ: Java/Kotlin のみ

オンまたはオフにして、SDK またはツールがリクエストを送信する前にペイロードを圧縮するかどうかを決定します。

デフォルト値: `false`

有効な値:

- **true** – リクエスト圧縮をオフにします。
- **false** – 可能な場合はリクエスト圧縮を使用します。

request_min_compression_size_bytes - 共有 AWS **config**ファイル設定, **AWS_REQUEST_MIN_COMPRESSION_SIZE_BYTES** - 環境変数, **aws.requestMinCompressionSizeBytes** - JVM システムプロパティ: Java/Kotlin のみ

SDK またはツールが圧縮する必要があるリクエスト本文の最小サイズ (バイト) を設定します。小さなペイロードは圧縮すると長くなる可能性があるため、圧縮を実行することが有意義である下限が存在します。この値は包括的であり、この値以上のリクエストサイズは圧縮されます。

デフォルト値: 10,240 バイト

有効な値: 0 ~ 10,485,760 バイトの整数値。

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	はい	
SDK for C++	はい	

SDK	サポート	注意または詳細情報
SDK for Go V2 (1.x)	はい	
SDK for Go 1.x (V1)	不可	
SDK for Java 2.x	はい	
SDK for Java 1.x	不可	
SDK for JavaScript 3.x	はい	
SDK for JavaScript 2.x	不可	
SDK for Kotlin	はい	
SDK for .NET 4.x	はい	
SDK for .NET 3.x	はい	
SDK for PHP 3.x	はい	
SDK for Python (Boto3)	はい	
SDK for Ruby 3.x	はい	

SDK	サポート	注意または詳細情報
SDK for Rust	はい	
SDK for Swift	不可	
PowerShell V5 のツール	はい	
PowerShell V4 のツール	はい	

サービス固有のエンドポイント

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

サービス固有のエンドポイント設定により、API リクエストに任意のエンドポイントを使用するオプションが得られ、この選択は持続します。これらの設定により、ローカルエンドポイント、VPC エンドポイント、およびサードパーティのローカル AWS 開発環境を柔軟にサポートできます。テスト環境と本番環境には異なるエンドポイントを使用できます。エンドポイント URL は個別の AWS のサービスに指定できます。

この機能を設定するには、以下のように使用します。

`endpoint_url` - 共有 AWS **`config`**ファイル設定, **`AWS_ENDPOINT_URL`** - 環境変数, **`aws.endpointUrl`** - JVM システムプロパティ: Java/Kotlin のみ

プロファイル内で直接指定するか、環境変数として指定した場合、この設定はすべてのサービスリクエストに使用されるエンドポイントを指定します。このエンドポイントは、設定されているサービス固有のエンドポイントによって上書きされます。

共有ファイルの servicesセクション AWS config内でこの設定を使用して、特定のサービスのカスタムエンドポイントを設定することもできます。services 内のサブセクションで使用するすべてのサービス識別子キーのリストについては、「サービス固有のエンドポイントの識別子」を参照してください。

デフォルト値: none

有効な値： エンドポイントのスキームとホストを含む URL。URL は、必要に応じて 1 つ以上のパスセグメントを含むパスコンポーネントを含めることができます。

AWS_ENDPOINT_URL_<SERVICE> 環境変数, **aws.endpointUrl<ServiceName>** - JVM システムプロパティ: Java/Kotlin のみ

AWS_ENDPOINT_URL_<SERVICE>は識別子<SERVICE>で AWS のサービス 、特定のサービスのカスタムエンドポイントを設定します。サービス固有の環境変数のリストについては、「サービス固有のエンドポイントの識別子」を参照してください。

このサービス固有のエンドポイントは、AWS_ENDPOINT_URL に設定されているグローバルエンドポイントよりも優先されます。

デフォルト値: none

有効な値： エンドポイントのスキームとホストを含む URL。URL は、必要に応じて 1 つ以上のパスセグメントを含むパスコンポーネントを含めることができます。

ignore_configured_endpoint_urls - 共有 AWS **config**ファイル設定, **AWS_IGNORE_CONFIGURED_ENDPOINT_URLS** - 環境変数, **aws.ignoreConfiguredEndpointUrls** - JVM システムプロパティ: Java/Kotlin のみ

この設定は、すべてのカスタムエンドポイント設定を無視するために使用されます。

コードまたはサービスクライアント自体に設定されている明示的なエンドポイントは、この設定に関係なく使用されることに注意してください。たとえば、 AWS CLI コマンドに--endpoint-urlコマンドラインパラメータを含めたり、エンドポイント URL をクライアントコンストラクタに渡したりすると、常に有効になります。

デフォルト値: false

有効な値:

- **true** — SDK またはツールは、エンドポイント URL を設定するための config 共有ファイルや環境変数からカスタム設定オプションを読み取ることはありません。

- **false** — SDK またはツールは、config 共有ファイルまたは環境変数からユーザーが提供したエンドポイントをすべて使用します。

環境変数を使用したエンドポイントの設定

すべてのサービスのリクエストをカスタムエンドポイント URL にルーティングするには、AWS_ENDPOINT_URL グローバル環境変数を設定します。

```
export AWS_ENDPOINT_URL=http://localhost:4567
```

特定の のリクエストをカスタムエンドポイント URL AWS のサービス にルーティングするには、 AWS_ENDPOINT_URL_<SERVICE>環境変数を使用します。 の Amazon DynamoDB は serviceIdですDynamoDB。このサービスのエンドポイント URL 環境変数は AWS_ENDPOINT_URL_DYNAMODB です。このエンドポイントは、このサービスのために AWS_ENDPOINT_URL に設定されているグローバルエンドポイントよりも優先されます。

```
export AWS_ENDPOINT_URL_DYNAMODB=http://localhost:5678
```

別の例として、 には serviceIdの AWS Elastic Beanstalk がありますElastic Beanstalk。AWS のサービス 識別子は、すべてのスペースをアンダースコアに置き換え、すべての文字を大文字にserviceIdすることで、API モデルの に基づいています。このサービスにエンドポイントを設定するための、対応する環境変数は AWS_ENDPOINT_URL_ELASTIC_BEANSTALK です。サービス固有の環境変数のリストについては、「サービス固有のエンドポイントの識別子」を参照してください。

```
export AWS_ENDPOINT_URL_ELASTIC_BEANSTALK=http://localhost:5567
```

config 共有ファイルを使用してエンドポイントを設定します

config 共有ファイルでは、 endpoint_url がさまざまな場所でさまざまな機能に使用されます。

- profile 内で endpoint_url を直接指定すると、そのエンドポイントがグローバルエンドポイントになります。
- services セクション内のサービス ID キーの下に endpoint_url をネストすると、そのエンドポイントはそのサービスに対して行われたリクエストにのみ適用されます。共有 config ファイル内の services セクションの定義について詳しくは、「設定ファイルの形式」を参照してください。

次の例では、services 定義を使用して Amazon S3 に使用されることとなるサービス固有のエンドポイント URL と、他のすべてのサービスに使用されることとなるカスタムグローバルエンドポイントを設定します。

```
[profile dev-s3-specific-and-global]
endpoint_url = http://localhost:1234
services = s3-specific

[services s3-specific]
s3 =
  endpoint_url = https://play.min.io:9000
```

1 つのプロファイルで複数のサービスのエンドポイントを設定できます。この例では、Amazon S3 と AWS Elastic Beanstalk のサービス固有のエンドポイント URLs を同じプロファイルに設定する方法を示します。 には serviceIdの AWS Elastic Beanstalk がありますElastic Beanstalk。AWS のサービス 識別子は、すべてのスペースをアンダースコアに置き換え、すべての文字を小文字に置き換えserviceIdることで、API モデルの に基づいています。したがって、サービス ID キーは elastic_beanstalk になり、このサービスの設定は elastic_beanstalk = の行から開始されます。services セクションで使用するすべてのサービス識別子キーのリストについては、「サービス固有のエンドポイントの識別子」を参照してください。

```
[services testing-s3-and-eb]
s3 =
  endpoint_url = http://localhost:4567
elastic_beanstalk =
  endpoint_url = http://localhost:8000

[profile dev]
services = testing-s3-and-eb
```

サービス設定セクションは複数のプロファイルで使用できます。たとえば、2つのプロファイルが同じ定義 services を使用し、他のプロファイルプロパティを変更することができます。

```
[services testing-s3]
s3 =
  endpoint_url = https://localhost:4567

[profile testing-json]
output = json
services = testing-s3
```

```
[profile testing-text]
output = text
services = testing-s3
```

ロールベースの認証情報を使用してプロファイル内のエンドポイントを設定します

プロファイルに IAM Assume Role 機能の source_profile パラメータによって設定されたロールベースの認証情報がある場合、SDK は指定されたプロファイルのサービス設定のみを使用します。ロールチェーンされたプロファイルは使用されません。例えば、次の共有 config ファイルを使用します。

```
[profile A]
credential_source = Ec2InstanceMetadata
endpoint_url = https://profile-a-endpoint.aws/

[profile B]
source_profile = A
role_arn = arn:aws:iam::123456789012:role/roleB
services = profileB

[services profileB]
ec2 =
  endpoint_url = https://profile-b-ec2-endpoint.aws
```

プロファイル B を使用してコード内で Amazon EC2 を呼び出すと、エンドポイントは https://profile-b-ec2-endpoint.aws として解決されます。コードが他のサービスにリクエストを送信した場合、エンドポイントの解決はカスタムロジックには従いません。エンドポイントはプロファイル A で定義されたグローバルエンドポイントには解決されません。グローバルエンドポイントを B プロファイルに対して有効にするには、プロファイル B 内で直接 endpoint_url を設定する必要があります。source_profile 設定の詳細については、ロール認証情報プロバイダーを引き受けますを参照してください。

設定の優先順位

この機能の設定は同時に使用できますが、1 つのサービスにつき 1 つの値が優先されます。特定のに対して行われた API コールでは AWS のサービス、次の順序を使用して値を選択します。

1. コードまたはサービスクライアント自体に設定されている明示的な設定は、他の設定よりも優先されます。

- の場合 AWS CLI、これは--endpoint-urlコマンドラインパラメータによって提供される値です。SDK の場合、明示的な割り当ては、 AWS のサービス クライアントまたは設定オブジェクトをインスタンス化するときに設定したパラメータの形式になります。

2. サービス固有の環境変数 (AWS_ENDPOINT_URL_DYNAMODB など) によって提供される値。
3. AWS_ENDPOINT_URL グローバルエンドポイント環境変数によって提供される値。
4. endpoint_url 設定によって得られる値は、config 共有ファイルの services セクション内のサービス ID キーの下にネストされます。
5. 共有 config ファイルの profile 内で直接指定された endpoint_url 設定によって得られる値。
6. それぞれの のデフォルトのエンドポイント URL AWS のサービス が最後に使用されます。

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	はい	
SDK for C++	はい	
SDK for Go V2 (1.x)	はい	
SDK for Go 1.x (V1)	いいえ	
SDK for Java 2.x	はい	

SDK	サポート	注意または詳細情報
SDK for Java 1.x	いいえ	
SDK for JavaScript 3.x	はい	
SDK for JavaScript 2.x	いいえ	
SDK for Kotlin	はい	
SDK for .NET 4.x	はい	
SDK for .NET 3.x	はい	
SDK for PHP 3.x	はい	
SDK for Python (Boto3)	はい	
SDK for Ruby 3.x	はい	
SDK for Rust	はい	
SDK for Swift	はい	

SDK	サポート	注意または詳細情報
PowerShell V5 のツール	はい	
PowerShell V4 のツール	はい	

サービス固有のエンドポイントの識別子

次の表の識別子の使用方法と使用場所については、「サービス固有のエンドポイント」を参照してください。

serviceId	共有AWS config ファイルのサービス識別子キー	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
AccessAnalyzer	access-analyzer	AWS_ENDPOINT_URL_ACCESSANALYZER
Account	account	AWS_ENDPOINT_URL_ACCOUNT
ACM	acm	AWS_ENDPOINT_URL_ACM

serviceId	共有AWS**config**ファイルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
ACM PCA	ac	AWS_ENDPOINT_URL_ACM_PCA
Alexa For Business	a _f	AWS_ENDPOINT_URL_ALEXA_FOR_BUSINESS
amp	ar	AWS_ENDPOINT_URL_AMP
Amplify	ar	AWS_ENDPOINT_URL_AMPLIFY
AmplifyBackend	ar cl	AWS_ENDPOINT_URL_AMPLIFYBACKEND
AmplifyUIBuilder	ar bu	AWS_ENDPOINT_URL_AMPLIFYUIBUILDER
API Gateway	ap ay	AWS_ENDPOINT_URL_API_GATEWAY
ApiGatewayManagementApi	ap yr nt	AWS_ENDPOINT_URL_APIGATEWAYMANAGEMENTAPI

serviceId	共有AWS**config**ファイルのサービス識別子キー	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
ApiGatewayV2	apigatewayv2	AWS_ENDPOINT_URL_APIGATEWAYV2
AppConfig	appconfig	AWS_ENDPOINT_URL_APPCONFIG
AppConfigData	appconfigdata	AWS_ENDPOINT_URL_APPCONFIGDATA
AppFabric	appfabric	AWS_ENDPOINT_URL_APPFABRIC
Appflow	appflow	AWS_ENDPOINT_URL_APPFLOW
AppIntegrations	appintegrations	AWS_ENDPOINT_URL_APPINTEGRATIONS
Application Auto Scaling	applicationautoscaling	AWS_ENDPOINT_URL_APPLICATION_AUTO_SCALING

serviceId	共有AWS**config**ファイルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
Application Insights	application-insights	AWS_ENDPOINT_URL_APPLICATION_INSIGHTS
ApplicationCostProfiler	application-cost-profiler	AWS_ENDPOINT_URL_APPLICATIONCOSTPROFILER
App Mesh	app-mesh	AWS_ENDPOINT_URL_APP_MESH
AppRunner	apprunner	AWS_ENDPOINT_URL_APPRUNNER
AppStream	appstream	AWS_ENDPOINT_URL_APPSTREAM
AppSync	appsync	AWS_ENDPOINT_URL_APPSYNC
ARC Zonal Shift	arc-zonal-shift	AWS_ENDPOINT_URL_ARC_ZONAL_SHIFT
Artifact	artifact	AWS_ENDPOINT_URL_ARTIFACT

serviceId	共有AWS configファイルのサービス識別子キ	AWS_ENDPOINT_URL_<SERVICE> 環境変数
Athena	at	AWS_ENDPOINT_URL_ATHENA
AuditManager	au ge	AWS_ENDPOINT_URL_AUDITMANAGER
Auto Scaling	au in	AWS_ENDPOINT_URL_AUTO_SCALING
Auto Scaling Plans	au in	AWS_ENDPOINT_URL_AUTO_SCALING_PLANS
b2bi	b2	AWS_ENDPOINT_URL_B2BI
Backup	ba	AWS_ENDPOINT_URL_BACKUP
Backup Gateway	ba te	AWS_ENDPOINT_URL_BACKUP_GATEWAY
BackupStorage	ba ra	AWS_ENDPOINT_URL_BACKUPSTORAGE
Batch	ba	AWS_ENDPOINT_URL_BATCH

serviceId	共有AWS**config**ファイルのサービス識別子キー	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
BCM Data Exports	bcm-data-exports	AWS_ENDPOINT_URL_BCM_DATA_EXPORTS
Bedrock	bedrock	AWS_ENDPOINT_URL_BEDROCK
Bedrock Agent	bedrock-agent	AWS_ENDPOINT_URL_BEDROCK_AGENT
Bedrock Agent Runtime	bedrock-agent-runtime	AWS_ENDPOINT_URL_BEDROCK_AGENT_RUNTIME
Bedrock Runtime	bedrock-runtime	AWS_ENDPOINT_URL_BEDROCK_RUNTIME
billingconductor	billingconductor	AWS_ENDPOINT_URL_BILLINGCONDUCTOR
Braket	braket	AWS_ENDPOINT_URL_BRAKET
Budgets	budgets	AWS_ENDPOINT_URL_BUDGETS

serviceId	共有AWS**c**イルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
Cost Explorer	co o:	AWS_ENDPOINT_URL_COST_EXPLORER
chatbot	ch	AWS_ENDPOINT_URL_CHATBOT
Chime	ch	AWS_ENDPOINT_URL_CHIME
Chime SDK Identity	ch _:	AWS_ENDPOINT_URL_CHIME_SDK_IDENTITY
Chime SDK Media Pipelines	ch _r pe	AWS_ENDPOINT_URL_CHIME_SDK_MEDIA_PIPELINES
Chime SDK Meetings	ch _r	AWS_ENDPOINT_URL_CHIME_SDK_MEETINGS
Chime SDK Messaging	ch _r g	AWS_ENDPOINT_URL_CHIME_SDK_MESSAGING

serviceId	共有AWS**config**ファイルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
Chime SDK Voice	ch_v	AWS_ENDPOINT_URL_CHIME_SDK_VOICE
CleanRooms	cls	AWS_ENDPOINT_URL_CLEANROOMS
CleanRoomsML	clsm	AWS_ENDPOINT_URL_CLEANROOMSML
Cloud9	cl	AWS_ENDPOINT_URL_CLOUD9
CloudControl	clcr	AWS_ENDPOINT_URL_CLOUDCONTROL
CloudDirectory	cldct	AWS_ENDPOINT_URL_CLOUDDIRECTORY
CloudFormation	clfn	AWS_ENDPOINT_URL_CLOUDFORMATION
CloudFront	cft	AWS_ENDPOINT_URL_CLOUDFRONT

serviceId	共 有 A **c** イ ル の サ ビ ス 識 別 子 キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
CloudFront KeyValueStore	c t_ e	AWS_ENDPOINT_URL_CLOUDFRONT_KEYVALUESTORE
CloudHSM	c	AWS_ENDPOINT_URL_CLOUDHSM
CloudHSM V2	c v	AWS_ENDPOINT_URL_CLOUDHSM_V2
CloudSearch	c c	AWS_ENDPOINT_URL_CLOUDSEARCH
CloudSearch Domain	c c	AWS_ENDPOINT_URL_CLOUDSEARCH_DOMAIN
CloudTrail	c l	AWS_ENDPOINT_URL_CLOUDTRAIL
CloudTrail Data	c l_	AWS_ENDPOINT_URL_CLOUDTRAIL_DATA

serviceId	共有AWS**config**ファイルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
CloudWatch	cloudwatch	AWS_ENDPOINT_URL_CLOUDWATCH
codeartifact	codeartifact	AWS_ENDPOINT_URL_CODEARTIFACT
CodeBuild	codebuild	AWS_ENDPOINT_URL_CODEBUILD
CodeCatalyst	codecatalyst	AWS_ENDPOINT_URL_CODECATALYST
CodeCommit	codecommit	AWS_ENDPOINT_URL_CODECOMMIT
CodeDeploy	codedeploy	AWS_ENDPOINT_URL_CODEDEPLOY
CodeGuru Reviewer	codeguru-reviewer	AWS_ENDPOINT_URL_CODEGURU_REVIEWER
CodeGuru Security	codeguru-security	AWS_ENDPOINT_URL_CODEGURU_SECURITY

serviceId	共有AWS**co**イルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
CodeGuruProfiler	co ro	AWS_ENDPOINT_URL_CODEGURUPROFILER
CodePipeline	co in	AWS_ENDPOINT_URL_CODEPIPELINE
CodeStar	co	AWS_ENDPOINT_URL_CODESTAR
CodeStar connections	co co ns	AWS_ENDPOINT_URL_CODESTAR_CONNECTIONS
codestar notifications	co no ic	AWS_ENDPOINT_URL_CODESTAR_NOTIFICATIONS
Cognito Identity	co de	AWS_ENDPOINT_URL_COGNITO_IDENTITY
Cognito Identity Provider	co de ro	AWS_ENDPOINT_URL_COGNITO_IDENTITY_PROVIDER

serviceId	共有AWS**config**ファイルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
Cognito Sync	co yn	AWS_ENDPOINT_URL_COGNITO_SYNC
Comprehend	co d	AWS_ENDPOINT_URL_COMPREHEND
ComprehendMedical	co dr	AWS_ENDPOINT_URL_COMPREHENDMEDICAL
Compute Optimizer	co pt	AWS_ENDPOINT_URL_COMPUTE_OPTIMIZER
Config Service	co rv	AWS_ENDPOINT_URL_CONFIG_SERVICE
Connect	co	AWS_ENDPOINT_URL_CONNECT
Connect Contact Lens	co or ns	AWS_ENDPOINT_URL_CONNECT_CONTACT_LENS
ConnectCampaigns	co mp	AWS_ENDPOINT_URL_CONNECTCAMPAIGNS

serviceId	共有AWS**c**イルのサビス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
ConnectCases	c s	AWS_ENDPOINT_URL_CONNECTCASES
ConnectParticipant	c r	AWS_ENDPOINT_URL_CONNECTPARTICIPANT
ControlTower	c w	AWS_ENDPOINT_URL_CONTROLTOWER
Cost Optimization Hub	c m h	AWS_ENDPOINT_URL_COST_OPTIMIZATION_HUB
Cost and Usage Report Service	c u o c	AWS_ENDPOINT_URL_COST_AND_USAGE_REPO RT_SERVICE
Customer Profiles	c p	AWS_ENDPOINT_URL_CUSTOMER_PROFILES
DataBrew	d	AWS_ENDPOINT_URL_DATABREW

serviceId	共有AWS**config**ファイルのサービス識別子キー	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
DataExchange	data_exchange	AWS_ENDPOINT_URL_DATAEXCHANGE
Data Pipeline	data_pipeline	AWS_ENDPOINT_URL_DATA_PIPELINE
DataSync	datasync	AWS_ENDPOINT_URL_DATASYNC
DataZone	datazone	AWS_ENDPOINT_URL_DATAZONE
DAX	dax	AWS_ENDPOINT_URL_DAX
Detective	detective	AWS_ENDPOINT_URL_DETECTIVE
Device Farm	device_farm	AWS_ENDPOINT_URL_DEVICE_FARM
DevOps Guru	devops_guru	AWS_ENDPOINT_URL_DEVOPS_GURU
Direct Connect	direct_connect	AWS_ENDPOINT_URL_DIRECT_CONNECT

serviceId	共有AWS**config**ファイルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
Application Discovery Service	application_discovery_service	AWS_ENDPOINT_URL_APPLICATION_DISCOVERY_SERVICE
DLM	dlm	AWS_ENDPOINT_URL_DLM
Database Migration Service	database_migration_service	AWS_ENDPOINT_URL_DATABASE_MIGRATION_SERVICE
DocDB	docdb	AWS_ENDPOINT_URL_DOCDB
DocDB Elastic	docdb-elastic	AWS_ENDPOINT_URL_DOCDB_ELASTIC
drs	drs	AWS_ENDPOINT_URL_DRS
Directory Service	directory_service	AWS_ENDPOINT_URL_DIRECTORY_SERVICE
DynamoDB	dynamodb	AWS_ENDPOINT_URL_DYNAMODB

serviceId	共有AWS**config**ファイルのサービス識別子キー	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
DynamoDB Streams	dynamodbstreams	AWS_ENDPOINT_URL_DYNAMODB_STREAMS
EBS	ebs	AWS_ENDPOINT_URL_EBS
EC2	ec2	AWS_ENDPOINT_URL_EC2
EC2 Instance Connect	ec2instanceconnect	AWS_ENDPOINT_URL_EC2_INSTANCE_CONNECT
ECR	ecr	AWS_ENDPOINT_URL_ECR
ECR PUBLIC	ecrpublic	AWS_ENDPOINT_URL_ECR_PUBLIC
ECS	ecs	AWS_ENDPOINT_URL_ECS
EFS	efs	AWS_ENDPOINT_URL_EFS
EKS	eks	AWS_ENDPOINT_URL_EKS
EKS Auth	eksauth	AWS_ENDPOINT_URL_EKS_AUTH

serviceId	共有AWS**config**ファイルのサービス識別子キー	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
Elastic Inference	elastic_inference	AWS_ENDPOINT_URL_ELASTIC_INFERENCE
ElastiCache	elasticache	AWS_ENDPOINT_URL_ELASTICACHE
Elastic Beanstalk	elastic_beanstalk	AWS_ENDPOINT_URL_ELASTIC_BEANSTALK
Elastic Transcoder	elastic_transcoder	AWS_ENDPOINT_URL_ELASTIC_TRANSCODER
Elastic Load Balancing	elastic_load_balancing	AWS_ENDPOINT_URL_ELASTIC_LOAD_BALANCING
Elastic Load Balancing v2	elastic_load_balancing	AWS_ENDPOINT_URL_ELASTIC_LOAD_BALANCING_V2
EMR	emr	AWS_ENDPOINT_URL_EMR

serviceId	共有AWS**config**ファイルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
EMR containers	emr-containers	AWS_ENDPOINT_URL_EMR_CONTAINERS
EMR Serverless	emr-serverless	AWS_ENDPOINT_URL_EMR_SERVERLESS
EntityResolution	entityresolution	AWS_ENDPOINT_URL_ENTITYRESOLUTION
Elasticsearch Service	es	AWS_ENDPOINT_URL_ELASTICSEARCH_SERVICE
EventBridge	eventbridge	AWS_ENDPOINT_URL_EVENTBRIDGE
Evidently	evidently	AWS_ENDPOINT_URL_EVIDENTLY
finspace	finspace	AWS_ENDPOINT_URL_FINSPACE
finspace data	finspace-data	AWS_ENDPOINT_URL_FINSPACE_DATA

serviceId	共有AWS**co**イルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
Firehose	f:	AWS_ENDPOINT_URL_FIREHOSE
fis	f:	AWS_ENDPOINT_URL_FIS
FMS	fr	AWS_ENDPOINT_URL_FMS
forecast	fo	AWS_ENDPOINT_URL_FORECAST
forecastquery	fo ue	AWS_ENDPOINT_URL_FORECASTQUERY
FraudDetector	f: ct	AWS_ENDPOINT_URL_FRAUDDETECTOR
FreeTier	f:	AWS_ENDPOINT_URL_FREETIER
FSx	fs	AWS_ENDPOINT_URL_FSX
GameLift	ga	AWS_ENDPOINT_URL_GAMELIFT
Glacier	gl	AWS_ENDPOINT_URL_GLACIER

serviceId	共有AWS**config**ファイルのサービス識別子キー	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
Global Accelerator	globalaccelerator	AWS_ENDPOINT_URL_GLOBAL_ACCELERATOR
Glue	glue	AWS_ENDPOINT_URL_GLUE
grafana	grafana	AWS_ENDPOINT_URL_GRAFANA
Greengrass	greengrass	AWS_ENDPOINT_URL_GREENGRASS
GreengrassV2	greengrassv2	AWS_ENDPOINT_URL_GREENGRASSV2
GroundStation	groundstation	AWS_ENDPOINT_URL_GROUNDSTATION
GuardDuty	guardduty	AWS_ENDPOINT_URL_GUARDDUTY
Health	health	AWS_ENDPOINT_URL_HEALTH

serviceId	共有AWS**config**ファイルのサービス識別子キー	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
HealthLake	health	AWS_ENDPOINT_URL_HEALTHLAKE
Honeycode	honeycode	AWS_ENDPOINT_URL_HONEYCODE
IAM	iam	AWS_ENDPOINT_URL_IAM
identitystore	identitystore	AWS_ENDPOINT_URL_IDENTITYSTORE
imagebuilder	imagebuilder	AWS_ENDPOINT_URL_IMAGEBUILDER
ImportExport	importexport	AWS_ENDPOINT_URL_IMPORTEXPORT
Inspector	inspector	AWS_ENDPOINT_URL_INSPECTOR
Inspector Scan	inspector_scan	AWS_ENDPOINT_URL_INSPECTOR_SCAN

serviceId	共有AWSconfigファイルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
Inspector2	inspector2	AWS_ENDPOINT_URL_INSPECTOR2
InternetMonitor	internetmonitor	AWS_ENDPOINT_URL_INTERNETMONITOR
IoT	iot	AWS_ENDPOINT_URL_IOT
IoT Data Plane	iot-data	AWS_ENDPOINT_URL_IOT_DATA_PLANE
IoT Jobs Data Plane	iot-jobs-data	AWS_ENDPOINT_URL_IOT_JOBS_DATA_PLANE
IoT 1Click Devices Service	iot1click-devices-service	AWS_ENDPOINT_URL_IOT_1CLICK_DEVICES_SERVICE
IoT 1Click Projects	iot1click-projects	AWS_ENDPOINT_URL_IOT_1CLICK_PROJECTS

serviceId	共有AWS config ファイルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
IoTAnalytics	iot-analytics	AWS_ENDPOINT_URL_IOTANALYTICS
IotDeviceAdvisor	iot-device-advisor	AWS_ENDPOINT_URL_IOTDEVICEADVISOR
IoT Events	iot-events	AWS_ENDPOINT_URL_IOT_EVENTS
IoT Events Data	iot-events-data	AWS_ENDPOINT_URL_IOT_EVENTS_DATA
IoTFleetHub	iot-fleethub	AWS_ENDPOINT_URL_IOTFLEETHUB
IoTFleetWise	iot-fleetwise	AWS_ENDPOINT_URL_IOTFLEETWISE
IoTSecureTunneling	iot-securetunneling	AWS_ENDPOINT_URL_IOTSECURETUNNELING

serviceId	共有AWS configイルのサービス識別子キ	AWS_ENDPOINT_URL_<SERVICE> 環境変数
IoTSiteWise	io se	AWS_ENDPOINT_URL_IOTSITEWISE
IoTThingsGraph	io gr	AWS_ENDPOINT_URL_IOTTHINGSGRAPH
IoTTwinMaker	io ke	AWS_ENDPOINT_URL_IOTTWINMAKER
IoT Wireless	io es	AWS_ENDPOINT_URL_IOT_WIRELESS
ivs	iv	AWS_ENDPOINT_URL_IVS
IVS RealTime	iv in	AWS_ENDPOINT_URL_IVS_REALTIME
ivschat	iv	AWS_ENDPOINT_URL_IVSCHAT
Kafka	ka	AWS_ENDPOINT_URL_KAFKA
KafkaConnect	ka ec	AWS_ENDPOINT_URL_KAFKACONNECT

serviceId	共有AWS**config**ファイルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
kendra	ke	AWS_ENDPOINT_URL_KENDRA
Kendra Ranking	ke nl	AWS_ENDPOINT_URL_KENDRA_RANKING
Keyspaces	ke	AWS_ENDPOINT_URL_KEYSPACES
Kinesis	ki	AWS_ENDPOINT_URL_KINESIS
Kinesis Video Archived Media	ki iv a	AWS_ENDPOINT_URL_KINESIS_VIDEO_ARCHIVED_MEDIA
Kinesis Video Media	ki ia	AWS_ENDPOINT_URL_KINESIS_VIDEO_MEDIA
Kinesis Video Signaling	ki ia	AWS_ENDPOINT_URL_KINESIS_VIDEO_SIGNALING

serviceId	共有AWS**config**ファイルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
Kinesis Video WebRTC Storage	kinesisvideowebrtcstorage	AWS_ENDPOINT_URL_KINESIS_VIDEO_WEBRTC_STORAGE
Kinesis Analytics	kinesisanalytics	AWS_ENDPOINT_URL_KINESIS_ANALYTICS
Kinesis Analytics V2	kinesisanalyticsv2	AWS_ENDPOINT_URL_KINESIS_ANALYTICS_V2
Kinesis Video	kinesisvideo	AWS_ENDPOINT_URL_KINESIS_VIDEO
KMS	kms	AWS_ENDPOINT_URL_KMS
LakeFormation	lakeformation	AWS_ENDPOINT_URL_LAKEFORMATION
Lambda	lambda	AWS_ENDPOINT_URL_LAMBDA

serviceId	共有AWS**config**ファイルのサービス識別子キー	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
Launch Wizard	launchwizard	AWS_ENDPOINT_URL_LAUNCH_WIZARD
Lex Model Building Service	lex-model-building-service	AWS_ENDPOINT_URL_LEX_MODEL_BUILDING_SERVICE
Lex Runtime Service	lex-runtime-service	AWS_ENDPOINT_URL_LEX_RUNTIME_SERVICE
Lex Models V2	lex-models-v2	AWS_ENDPOINT_URL_LEX_MODELS_V2
Lex Runtime V2	lex-runtime-v2	AWS_ENDPOINT_URL_LEX_RUNTIME_V2
License Manager	license-manager	AWS_ENDPOINT_URL_LICENSE_MANAGER

serviceId	共有AWS**config**ファイルのサービス識別子キー	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
License Manager Linux Subscriptions	license_manager_linux_subscriptions	AWS_ENDPOINT_URL_LICENSE_MANAGER_LINUX_SUBSCRIPTIONS
License Manager User Subscriptions	license_manager_user_subscriptions	AWS_ENDPOINT_URL_LICENSE_MANAGER_USER_SUBSCRIPTIONS
Lightsail	lightsail	AWS_ENDPOINT_URL_LIGHTSAIL
Location	location	AWS_ENDPOINT_URL_LOCATION
CloudWatch Logs	cloudwatch_logs	AWS_ENDPOINT_URL_CLOUDWATCH_LOGS
LookoutEquipment	lookoutequipment	AWS_ENDPOINT_URL_LOOKOUTEQUIPMENT
LookoutMetrics	lookoutmetrics	AWS_ENDPOINT_URL_LOOKOUTMETRICS

serviceId	共有AWS**config**ファイルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
LookoutVision	lo s:	AWS_ENDPOINT_URL_LOOKOUTVISION
m2	m2	AWS_ENDPOINT_URL_M2
Machine Learning	ma ea	AWS_ENDPOINT_URL_MACHINE_LEARNING
Macie2	ma	AWS_ENDPOINT_URL_MACIE2
ManagedBlockchain	ma oc	AWS_ENDPOINT_URL_MANAGEDBLOCKCHAIN
ManagedBlockchain Query	ma oc qu	AWS_ENDPOINT_URL_MANAGEDBLOCKCHAIN_QUERY
Marketplace Agreement	ma ce er	AWS_ENDPOINT_URL_MARKETPLACE_AGREEMENT

serviceId	共有AWS**config**ファイルのサービス識別子キー	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
Marketplace Catalog	marketplacecatalog	AWS_ENDPOINT_URL_MARKETPLACE_CATALOG
Marketplace Deployment	marketplacedeployment	AWS_ENDPOINT_URL_MARKETPLACE_DEPLOYMENT
Marketplace Entitlement Service	marketplaceentitlementservice	AWS_ENDPOINT_URL_MARKETPLACE_ENTITLEMENT_SERVICE
Marketplace Commerce Analytics	marketplacecommerceanalytics	AWS_ENDPOINT_URL_MARKETPLACE_COMMERCE_ANALYTICS
MediaConnect	mediaconnect	AWS_ENDPOINT_URL_MEDIACONNECT

serviceId	共有AWS **co**イルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
MediaConvert	me e:	AWS_ENDPOINT_URL_MEDIACONVERT
MediaLive	me	AWS_ENDPOINT_URL_MEDIALIVE
MediaPackage	me ag	AWS_ENDPOINT_URL_MEDIAPACKAGE
MediaPackage Vod	me ag	AWS_ENDPOINT_URL_MEDIAPACKAGE_VOD
MediaPackageV2	me ag	AWS_ENDPOINT_URL_MEDIAPACKAGEV2
MediaStore	me e	AWS_ENDPOINT_URL_MEDIASTORE
MediaStore Data	me e_	AWS_ENDPOINT_URL_MEDIASTORE_DATA
MediaTailor	me o:	AWS_ENDPOINT_URL_MEDIATAILOR

serviceId	共有AWS**config**ファイルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
Medical Imaging	medical-imaging	AWS_ENDPOINT_URL_MEDICAL_IMAGING
MemoryDB	memorydb	AWS_ENDPOINT_URL_MEMORYDB
Marketplace Metering	marketplace-metering	AWS_ENDPOINT_URL_MARKETPLACE_METERING
Migration Hub	migration-hub	AWS_ENDPOINT_URL_MIGRATION_HUB
mgn	mgn	AWS_ENDPOINT_URL_MGN
Migration Hub Refactor Spaces	migration-hub-refactor-spaces	AWS_ENDPOINT_URL_MIGRATION_HUB_REFACTOR_SPACES
MigrationHub Config	migrationhub-config	AWS_ENDPOINT_URL_MIGRATIONHUB_CONFIG

serviceId	共有AWSconfigファイルのサービス識別子キ	AWS_ENDPOINT_URL_<SERVICE> 環境変数
MigrationHubOrchestrator	migrationhub-orchestration	AWS_ENDPOINT_URL_MIGRATIONHUBORCHESTRATOR
MigrationHubStrategy	migrationhub-strategy	AWS_ENDPOINT_URL_MIGRATIONHUBSTRATEGY
Mobile	mobile	AWS_ENDPOINT_URL_MOBILE
mq	mq	AWS_ENDPOINT_URL_MQ
MTurk	mturk	AWS_ENDPOINT_URL_MTURK
MWAA	mwaa	AWS_ENDPOINT_URL_MWAA
Neptune	neptune	AWS_ENDPOINT_URL_NEPTUNE
Neptune Graph	neptune-graph	AWS_ENDPOINT_URL_NEPTUNE_GRAPH
neptunedata	neptunedata	AWS_ENDPOINT_URL_NEPTUNEDATA

serviceId	共有AWS configイルのサービス識別子キ	AWS_ENDPOINT_URL_<SERVICE> 環境変数
Network Firewall	ni i:	AWS_ENDPOINT_URL_NETWORK_FIREWALL
NetworkManager	ne na	AWS_ENDPOINT_URL_NETWORKMANAGER
NetworkMonitor	ne n:	AWS_ENDPOINT_URL_NETWORKMONITOR
nimble	n:	AWS_ENDPOINT_URL_NIMBLE
OAM	oa	AWS_ENDPOINT_URL_OAM
Omics	or	AWS_ENDPOINT_URL_OMICS
OpenSearch	op h	AWS_ENDPOINT_URL_OPENSEARCH
OpenSearchServerless	op hs ss	AWS_ENDPOINT_URL_OPENSEARCHSERVERLESS
OpsWorks	op	AWS_ENDPOINT_URL_OPSWORKS

serviceId	共有AWS**config**ファイルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
OpsWorksCM	opsworkscm	AWS_ENDPOINT_URL_OPSWORKSCM
Organizations	organizations	AWS_ENDPOINT_URL_ORGANIZATIONS
OSIS	osis	AWS_ENDPOINT_URL_OSIS
Outposts	outposts	AWS_ENDPOINT_URL_OUTPOSTS
p8data	p8data	AWS_ENDPOINT_URL_P8DATA
p8data	p8data	AWS_ENDPOINT_URL_P8DATA
Panorama	panorama	AWS_ENDPOINT_URL_PANORAMA
Payment Cryptography	paymentcryptography	AWS_ENDPOINT_URL_PAYMENT_CRYPTOGRAPHY
Payment Cryptography Data	paymentcryptography	AWS_ENDPOINT_URL_PAYMENT_CRYPTOGRAPHY_DATA

serviceId	共有AWS**config**ファイルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
Pca Connector Ad	pca-connector-ad	AWS_ENDPOINT_URL_PCA_CONNECTOR_AD
Personalize	personalize	AWS_ENDPOINT_URL_PERSONALIZE
Personalize Events	personalize-events	AWS_ENDPOINT_URL_PERSONALIZE_EVENTS
Personalize Runtime	personalize-runtime	AWS_ENDPOINT_URL_PERSONALIZE_RUNTIME
PI	pi	AWS_ENDPOINT_URL_PI
Pinpoint	pinpoint	AWS_ENDPOINT_URL_PINPOINT
Pinpoint Email	pinpoint-email	AWS_ENDPOINT_URL_PINPOINT_EMAIL

serviceId	共有AWS**config**ファイルのサービス識別子キー	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
Pinpoint SMS Voice	pinpoint_sms_voice	AWS_ENDPOINT_URL_PINPOINT_SMS_VOICE
Pinpoint SMS Voice V2	pinpoint_sms_voice_v2	AWS_ENDPOINT_URL_PINPOINT_SMS_VOICE_V2
Pipes	pipes	AWS_ENDPOINT_URL_PIPES
Polly	polly	AWS_ENDPOINT_URL_POLLY
Pricing	pricing	AWS_ENDPOINT_URL_PRICING
PrivateNetworks	privatenetworks	AWS_ENDPOINT_URL_PRIVATENETWORKS
Proton	proton	AWS_ENDPOINT_URL_PROTON
QBusiness	qbusiness	AWS_ENDPOINT_URL_QBUSINESS
QConnect	qconnect	AWS_ENDPOINT_URL_QCONNECT

serviceId	共有AWS**config**ファイルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
QLDB	qldb	AWS_ENDPOINT_URL_QLDB
QLDB Session	qldb-session	AWS_ENDPOINT_URL_QLDB_SESSION
QuickSight	quicksight	AWS_ENDPOINT_URL_QUICKSIGHT
RAM	ram	AWS_ENDPOINT_URL_RAM
rbin	rbin	AWS_ENDPOINT_URL_RBIN
RDS	rds	AWS_ENDPOINT_URL_RDS
RDS Data	rds-data	AWS_ENDPOINT_URL_RDS_DATA
Redshift	redshift	AWS_ENDPOINT_URL_REDSHIFT
Redshift Data	redshift-data	AWS_ENDPOINT_URL_REDSHIFT_DATA
Redshift Serverless	redshift-serverless	AWS_ENDPOINT_URL_REDSHIFT_SERVERLESS

serviceId	共有AWS**config**ファイルのサービス識別子キー	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
Rekognition	rekognition	AWS_ENDPOINT_URL_REKOGNITION
repostspace	repostspace	AWS_ENDPOINT_URL_REPOSTSPACE
resiliencehub	resiliencehub	AWS_ENDPOINT_URL_RESILIENCEHUB
Resource Explorer 2	resource-explorer-2	AWS_ENDPOINT_URL_RESOURCE_EXPLORER_2
Resource Groups	resource-groups	AWS_ENDPOINT_URL_RESOURCE_GROUPS
Resource Groups Tagging API	resource-groups-tagging-api	AWS_ENDPOINT_URL_RESOURCE_GROUPS_TAGGING_API
RoboMaker	robomaker	AWS_ENDPOINT_URL_ROBOMAKER

serviceId	共有AWS**config**ファイルのサービス識別子キー	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
RolesAnywhere	rolesanywhere	AWS_ENDPOINT_URL_ROLESANYWHERE
Route 53	route_53	AWS_ENDPOINT_URL_ROUTE_53
Route53 Recovery Cluster	route53_recovery_cluster	AWS_ENDPOINT_URL_ROUTE53_RECOVERY_CLUSTER
Route53 Recovery Control Config	route53_recovery_control_config	AWS_ENDPOINT_URL_ROUTE53_RECOVERY_CONTROL_CONFIG
Route53 Recovery Readiness	route53_recovery_readiness	AWS_ENDPOINT_URL_ROUTE53_RECOVERY_READINESS
Route 53 Domains	route_53_domains	AWS_ENDPOINT_URL_ROUTE_53_DOMAINS
Route53Resolver	route53resolver	AWS_ENDPOINT_URL_ROUTE53RESOLVER

serviceId	共有AWS**config**ファイルのサービス識別子キー	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
RUM	ru	AWS_ENDPOINT_URL_RUM
S3	s3	AWS_ENDPOINT_URL_S3
S3 Control	s3 l	AWS_ENDPOINT_URL_S3_CONTROL
S3Outposts	s3 s	AWS_ENDPOINT_URL_S3OUTPOSTS
SageMaker	sa	AWS_ENDPOINT_URL_SAGEMAKER
SageMaker A2I Runtime	sa _a in	AWS_ENDPOINT_URL_SAGEMAKER_A2I_RUNTIME
Sagemaker Edge	sa _e	AWS_ENDPOINT_URL_SAGEMAKER_EDGE

serviceId	共有AWS **config** ファイルのサービス識別子キー	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
SageMaker FeatureStore Runtime	sa _f to in	AWS_ENDPOINT_URL_SAGEMAKER_FEATURESTORE_RUNTIME
SageMaker Geospatial	sa _g a	AWS_ENDPOINT_URL_SAGEMAKER_GEOSPATIAL
SageMaker Metrics	sa _m	AWS_ENDPOINT_URL_SAGEMAKER_METRICS
SageMaker Runtime	sa _r	AWS_ENDPOINT_URL_SAGEMAKER_RUNTIME
savingsplans	sa an	AWS_ENDPOINT_URL_SAVINGSPLANS
Scheduler	sc	AWS_ENDPOINT_URL_SCHEDULER
schemas	sc	AWS_ENDPOINT_URL_SCHEMAS

serviceId	共有AWS**config**ファイルのサービス識別子キー	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
SimpleDB	s:	AWS_ENDPOINT_URL_SIMPLEDB
Secrets Manager	se ar	AWS_ENDPOINT_URL_SECRETS_MANAGER
SecurityHub	se ub	AWS_ENDPOINT_URL_SECURITYHUB
SecurityLake	se al	AWS_ENDPOINT_URL_SECURITYLAKE
ServerlessApplicationRepository	se sa io to	AWS_ENDPOINT_URL_SERVERLESSAPPLICATIONREPOSITORY
Service Quotas	se uo	AWS_ENDPOINT_URL_SERVICE_QUOTAS
Service Catalog	se at	AWS_ENDPOINT_URL_SERVICE_CATALOG

serviceId	共有AWS**co**イルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
Service Catalog AppRegistry	se at pi	AWS_ENDPOINT_URL_SERVICE_CATALOG_APP REGISTRY
ServiceDiscovery	se sc	AWS_ENDPOINT_URL_SERVICEDISCOVERY
SES	se	AWS_ENDPOINT_URL_SES
SESv2	se	AWS_ENDPOINT_URL_SESV2
Shield	sh	AWS_ENDPOINT_URL_SHIELD
signer	si	AWS_ENDPOINT_URL_SIGNER
SimSpaceWeaver	si ea	AWS_ENDPOINT_URL_SIMSPACEWEAVER
SMS	sn	AWS_ENDPOINT_URL_SMS

serviceId	共有AWS**co**イルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
Snow Device Management	sn ce me	AWS_ENDPOINT_URL_SNOW_DEVICE_MANAGEMENT
Snowball	sn	AWS_ENDPOINT_URL_SNOWBALL
SNS	sn	AWS_ENDPOINT_URL_SNS
SQS	sq	AWS_ENDPOINT_URL_SQS
SSM	ss	AWS_ENDPOINT_URL_SSM
SSM Contacts	ss ct	AWS_ENDPOINT_URL_SSM_CONTACTS
SSM Incidents	ss er	AWS_ENDPOINT_URL_SSM_INCIDENTS
Ssm Sap	ss	AWS_ENDPOINT_URL_SSM_SAP
SSO	ss	AWS_ENDPOINT_URL_SSO
SSO Admin	ss	AWS_ENDPOINT_URL_SSO_ADMIN

serviceId	共有AWS**co**イルのサービス識別子キ	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
SSO OIDC	ss	AWS_ENDPOINT_URL_SSO_OIDC
SFN	st	AWS_ENDPOINT_URL_SFN
Storage Gateway	st at	AWS_ENDPOINT_URL_STORAGE_GATEWAY
STS	st	AWS_ENDPOINT_URL_STS
SupplyChain	su in	AWS_ENDPOINT_URL_SUPPLYCHAIN
Support	su	AWS_ENDPOINT_URL_SUPPORT
Support App	su pp	AWS_ENDPOINT_URL_SUPPORT_APP
SWF	sw	AWS_ENDPOINT_URL_SWF
synthetics	sy s	AWS_ENDPOINT_URL_SYNTHETICS
Textract	te	AWS_ENDPOINT_URL_TEXTRACT

serviceId	共有AWS configファイルのサービス識別子キ	AWS_ENDPOINT_URL_<SERVICE> 環境変数
Timestream InfluxDB	timestream-influxdb	AWS_ENDPOINT_URL_TIMESTREAM_INFLUXDB
Timestream Query	timestream-query	AWS_ENDPOINT_URL_TIMESTREAM_QUERY
Timestream Write	timestream-write	AWS_ENDPOINT_URL_TIMESTREAM_WRITE
tnb	tnb	AWS_ENDPOINT_URL_TNB
Transcribe	transcribe	AWS_ENDPOINT_URL_TRANSCRIBE
Transfer	transfer	AWS_ENDPOINT_URL_TRANSFER
Translate	translate	AWS_ENDPOINT_URL_TRANSLATE
TrustedAdvisor	trustedadvisor	AWS_ENDPOINT_URL_TRUSTEDADVISOR

serviceId	共有AWS configファイルのサービス識別子キ	AWS_ENDPOINT_URL_<SERVICE> 環境変数
VerifiedPermissions	ve e: s	AWS_ENDPOINT_URL_VERIFIEDPERMISSIONS
Voice ID	vo	AWS_ENDPOINT_URL_VOICE_ID
VPC Lattice	vp ce	AWS_ENDPOINT_URL_VPC_LATTICE
WAF	wa	AWS_ENDPOINT_URL_WAF
WAF Regional	wa na	AWS_ENDPOINT_URL_WAF_REGIONAL
WAFV2	wa	AWS_ENDPOINT_URL_WAFV2
WellArchitected	we te	AWS_ENDPOINT_URL_WELLARCHITECTED
Wisdom	wi	AWS_ENDPOINT_URL_WISDOM
WorkDocs	wo	AWS_ENDPOINT_URL_WORKDOCS
WorkLink	wo	AWS_ENDPOINT_URL_WORKLINK

serviceId	共有AWS**config**ファイルのサービス識別子キー	**AWS_ENDPOINT_URL_<SERVICE>** 環境変数
WorkMail	workmail	AWS_ENDPOINT_URL_WORKMAIL
WorkMailMessageFlow	workmailmessagefl ow	AWS_ENDPOINT_URL_WORKMAILMESSAGEFLOW
WorkSpaces	workspaces	AWS_ENDPOINT_URL_WORKSPACES
WorkSpaces Thin Client	workspaces_thin_cli ent	AWS_ENDPOINT_URL_WORKSPACES_THIN_CLIENT
WorkSpaces Web	workspaces_web	AWS_ENDPOINT_URL_WORKSPACES_WEB
XRay	xray	AWS_ENDPOINT_URL_XRAY

スマート設定デフォルト

> **Note**
>
> 設定ページのレイアウトの理解、または以下の AWS SDKs「」を参照してくださいこのガイドの設定ページについて。

スマート設定のデフォルト機能を使用すると、 AWS SDKs他の設定用に事前定義された最適化されたデフォルト値を提供できます。

この機能を設定するには、以下のように使用します。

`defaults_mode` - 共有 AWS **`config`**ファイル設定, **`AWS_DEFAULTS_MODE`** - 環境変数, **`aws.defaultsMode`** - JVM システムプロパティ: Java/Kotlin のみ

この設定では、アプリケーションアーキテクチャに合ったモードを選択できます。これにより、アプリケーションに最適なデフォルト値が使用できるようになります。 AWS SDK 設定に値が明示的に設定されている場合、その値は常に優先されます。 AWS SDK 設定に値が明示的に設定されておらず、レガシーと等しくない場合、この機能`defaults_mode`はアプリケーションに最適化されたさまざまな設定に異なるデフォルト値を提供することができます。設定には、HTTP 通信設定、再試行動作、サービスの地域エンドポイント設定、および SDK 関連のあらゆる設定が含まれる可能性があります。この機能を使用するお客様は、一般的な使用シナリオに合わせた新しいデフォルト設定を取得できます。`defaults_mode` が `legacy` と等しくない場合は、SDK をアップグレードするときにアプリケーションのテストを行うことをおすすめします。これは、ベストプラクティスの進化によってこの機能のデフォルト値が変わる可能性があるためです。

デフォルト値: `legacy`

注意：SDK の新しいメジャーバージョンでのデフォルトは `standard` になります。

有効な値:

- `legacy` – SDK によって異なり、`defaults_mode` が確立される前から存在していたデフォルト設定を使用します。
- `standard` – ほとんどのシナリオで安全に実行できる最新の推奨デフォルト値を使用します。
- `in-region` – 標準モード上に構築され、同じ 内 AWS のサービス から を呼び出すアプリケーションに合わせた最適化が含まれています AWS リージョン。

- cross-region – 標準モードに基づいて構築され、別のリージョン AWS のサービス で を呼び出すアプリケーション向けにカスタマイズされた最適化が含まれています。
- mobile – 標準モードに基づいて構築されており、モバイルアプリケーションに合わせた最適化が含まれています。
- auto – 標準モードに基づいて構築されており、実験的な機能が含まれています。SDK はランタイム環境を検出して適切な設定を自動的に決定しようとします。自動検出はヒューリスティックに基づいてあり、100% の精度は得られません。ランタイム環境を特定できない場合は、standard モードが使用されます。自動検出では、インスタンスのメタデータをクエリすることがあり、レイテンシーが発生する可能性があります。起動時のレイテンシーがアプリケーションにとって最重要な場合は、代わりに明示的な defaults_mode を選択することをおすすめします。

config ファイルにこの値を設定する例を以下に示します。

```
[default]
defaults_mode = standard
```

以下のパラメータは、defaults_mode の選択に基づいて最適化される可能性があります。

- retryMode – SDK が再試行を試みる方法を指定します。「再試行動作」を参照してください。
- stsRegionalEndpoints – SDK が AWS Security Token Service () との通信に使用する AWS のサービス エンドポイントを決定する方法を指定しますAWS STS。「AWS STS リージョンエンドポイント」を参照してください。
- s3UsEast1RegionalEndpoints – SDK がus-east-1リージョンの Amazon S3 と通信するために使用する AWS サービスエンドポイントを決定する方法を指定します。
- connectTimeoutInMillis – ソケットで初めて接続を試みた後、タイムアウトするまでの時間。クライアントが接続ハンドシェイクの完了を受け取らない場合、クライアントは断念しオペレーションは失敗します。
- tlsNegotiationTimeoutInMillis – CLIENT HELLO メッセージが送信されてから、クライアントとサーバーが暗号を完全にネゴシエートしてキーを交換するまでの TLS ハンドシェイクにかかる最大時間。

各設定のデフォルト値は、アプリケーションで選択した defaults_mode によって異なります。これらの値は、現在以下のように設定されています (変更される可能性があります) 。

パラメータ	**standard** モード	**in-region** モード	**cross-region** モード	**mobile** モード
retryMode	standard	standard	standard	standard
stsRegionalEndpoints	regional	regional	regional	regional
s3UsEast1RegionalEndpoints	regional	regional	regional	regional
connectTimeoutInMillis	3100	1100	3100	30000
tlsNegotiationTimeoutInMillis	3100	1100	3100	30000

たとえば、選択した defaults_mode が standard の場合、standard の値が (有効な retry_mode オプションから) retry_mode に割り当てられ、regional の値が (有効な stsRegionalEndpoints オプションから) stsRegionalEndpoints に割り当てられます。

AWS SDKsとツールによるサポート

以下の SDK は、このトピックで説明する機能と設定をサポートします。部分的な例外があれば、すべて記載されています。JVM システムプロパティ設定は、 AWS SDK for Java と AWS SDK for Kotlin でのみサポートされます。

SDK	サポート	注意または詳細情報
AWS CLI v2	不可	
SDK for C++	はい	最適化されていないパラメーター : stsRegion

SDK	サポート	注意または詳細情報
		alEndpoints 、s3UsEast1RegionalEndpoints 、tlsNegotiationTimeoutInMillis 。
SDK for Go V2 (1.x)	はい	最適化されていないパラメーター：retryMode 、stsRegionalEndpoints 、s3UsEast1RegionalEndpoints 。
SDK for Go 1.x (V1)	不可	
SDK for Java 2.x	はい	最適化されていないパラメーター：stsRegionalEndpoints 。
SDK for Java 1.x	不可	
SDK for JavaScript 3.x	はい	最適化されていないパラメーター：stsRegionalEndpoints 、s3UsEast1RegionalEndpoints 、tlsNegotiationTimeoutInMillis 。connectTimeoutInMillis は connectionTimeout と呼ばれます。
SDK for JavaScript 2.x	不可	
SDK for Kotlin	不可	

SDK	サポート	注意または詳細情報
SDK for .NET 4.x	はい	最適化されていないパラメーター：connectTimeoutInMillis 、tlsNegotiationTimeoutInMillis 。
SDK for .NET 3.x	はい	最適化されていないパラメーター：connectTimeoutInMillis 、tlsNegotiationTimeoutInMillis 。
SDK for PHP 3.x	はい	最適化されていないパラメーター：tlsNegotiationTimeoutInMillis 。
SDK for Python (Boto3)	はい	最適化されていないパラメーター：tlsNegotiationTimeoutInMillis 。
SDK for Ruby 3.x	はい	
SDK for Rust	不可	
SDK for Swift	不可	
PowerShell V5 のツール	はい	最適化されていないパラメーター：connectTimeoutInMillis 、tlsNegotiationTimeoutInMillis 。

SDK	サポート	注意または詳細情報
PowerShell V4 用のツール	はい	最適化されていないパラメーター：connectTimeoutInMillis 、tlsNegotiationTimeoutInMillis 。

AWS 共通ランタイム (CRT) ライブラリ

AWS 共通ランタイム (CRT) ライブラリは SDKs。CRT は C で書かれた独立パッケージのモジュラーファミリーで、各パッケージはパフォーマンスが高く、必要なさまざまな機能にフットプリントを最小限に抑えます。これらの機能はすべての SDK に共通で共有しているため、コードの再利用、最適化、精度が向上します。パッケージは以下のとおりです。

- awslabs/aws-c-auth: AWS クライアント側の認証 (標準認証情報プロバイダーと署名 (sigv4))
- awslabs/aws-c-cal：暗号プリミティブ型、ハッシュ (MD5、SHA256、SHA256 HMAC)、署名者、AES
- awslabs/aws-c-common：基本データ構造、スレッド / 同期プリミティブ型、バッファ管理、stdlib 関連関数
- awslabs/aws-c-compression：圧縮アルゴリズム (ハフマンエンコーディング / デコーディング)
- awslabs/aws-c-event-stream：イベントストリームメッセージ処理 (ヘッダー、プレリュード、ペイロード、crc/トレーラー)、イベントストリーム経由のリモートプロシージャ呼び出し (RPC) 実装
- awslabs/aws-c-http：C99 による、HTTP/1.1 仕様と、HTTP/2 仕様の実装
- awslabs/aws-c-io：ソケット (TCP、UDP)、DNS、パイプ、イベントループ、チャネル、SSL/TLS
- awslabs/aws-c-iot: デバイスとの AWS IoT クラウドサービス統合の C99 実装
- awslabs/aws-c-mqtt：モノのインターネット (IoT) 向けの標準の軽量メッセージングプロトコル
- awslabs/aws-c-s3：Amazon S3 サービスと通信するための C99 ライブラリ実装。高帯域幅の Amazon EC2 インスタンスでスループットを最大化するように設計されています
- awslabs/aws-c-sdkutils: AWS プロファイルを解析および管理するためのユーティリティライブラリ
- awslabs/aws-checksums：効率的なソフトウェア実装へのフォールバック機能を備えた、クロスプラットフォームのハードウェア加速化による CRC32c と CRC32
- awslabs/aws-lc: Google BoringSSL AWS プロジェクトと OpenSSL プロジェクトのコードに基づいて、Cryptography チームが AWS とその顧客のために管理する汎用暗号化ライブラリ
- awslabs/s2n：C99 による TLS/SSL プロトコルの 実装。セキュリティを優先して小型かつ高速に動作するように設計

CRT は Go と Rust を除くすべての SDK で使用できます。

CRT の依存関係

CRT ライブラリは複雑な関係と依存関係を形成しています。これらの関係を知っておくと、CRT をソースから直接構築する必要がある場合に役立ちます。ただし、ほとんどのユーザーは、言語 SDK (AWS SDK for C++ や AWS SDK for Java など) または言語 IoT デバイス SDK (AWS IoT SDK for C++ や AWS IoT SDK for Java など) を介して CRT 機能にアクセスします。以下の図の「言語 CRT バインディング」ボックスは、特定の言語 SDK の CRT ライブラリをラップするパッケージを示しています。これは `aws-crt-*` 形式のパッケージの集まりで、「*」は SDK 言語 (`aws-crt-cpp` や `aws-crt-java` など) です。

CRT ライブラリの階層的な依存関係を以下に示します。
個々の CRT ライブラリが互いにどのように関連しているかを示す CRT 依存関係図。

AWS SDKsメンテナンスポリシー

概要:

このドキュメントでは、モバイル SDKs と IoT SDK を含む AWS Software Development Kit (SDK) とツールのメンテナンスポリシー、およびその基盤となる依存関係の概要を説明します。 AWS は、SDK AWS SDKs とツールに、新規または更新された AWS APIs、新機能、機能強化、バグ修正、セキュリティパッチ、またはドキュメントの更新のサポートを含む更新を定期的に提供します。SDKs 更新では、依存関係、言語ランタイム、オペレーティングシステムの変更にも対処できます。AWS SDK リリースはパッケージマネージャー (Maven、NuGet、PyPI など) に公開され、GitHub でソースコードとして利用できます。

最新の機能、セキュリティアップデート、および基本的な依存関係を維持するために、SDK のリリースをユーザーが常に更新することをお勧めします。サポート対象外の SDK バージョンを継続して使用することはお勧めできません。ユーザーの判断で行ってください。

バージョニング

AWS SDK リリースバージョンは X.Y.Z の形式で、X はメジャーバージョンを表します。SDK のメジャーバージョンを増やすということは、その SDK がその言語の新しいイディオムやパターンをサポートするために大幅に変更されたことを意味します。メジャーバージョンは、パブリックインターフェイス (クラス、メソッド、タイプなど) 、動作、またはセマンティクスが変更された時点で導入されます。アプリケーションを最新の SDK バージョンで動作させるには、更新する必要があります。メジャーバージョンは、 AWSに記載されているアップグレードガイドラインに従って慎重に更新することが重要です。

SDK のメジャーバージョンライフサイクル

SDK とツールのメジャーバージョンのライフサイクルは 5 つのフェーズで構成され、その概要は以下のとおりです。

- 開発者プレビュー (フェーズ 0) -このフェーズでは SDK のサポートはされないため、本番環境では使用できません。また、SDK は早期アクセスとフィードバックのみを目的としています。今後のリリースでは、非互換性の変更が導入される可能性があります。がリリースを安定した製品として AWS 識別すると、リリース候補としてマークされる場合があります。リリース候補は、重大な

バグが発生しない限り GA リリースの準備ができており、フル AWS サポートを受けることができます。

- 一般提供 (GA) (フェーズ 1) - このフェーズでは、SDKsが完全にサポートされています。 AWS は、新しい サービスのサポート、既存の サービスの API 更新、バグとセキュリティの修正を含む定期的な SDK リリースを提供します。ツールの場合、 AWS は新機能の更新とバグ修正を含む定期的なリリースを提供します。 AWS は SDK の GA バージョンを少なくとも 24 か月間サポートします。
- メンテナンス発表 (フェーズ 2) - AWS SDK がメンテナンスモードに入る少なくとも 6 か月前に公開発表を行います。この期間中、SDK は引き続き完全にサポートされます。通常、メンテナンスモードは次のメジャーバージョンが GA に移行されると同時に発表されます。
- メンテナンス (フェーズ 3) -メンテナンスモードでは、 AWS は重大なバグ修正とセキュリティ問題のみに対処するよう SDK のリリースを制限します。SDK は、新規または既存のサービスの API 更新を受け取ることも、新しいリージョンをサポートするように更新されることもありません。特に指定がない限り、メンテナンスモードのデフォルト期間は 12 か月です。
- サポート終了 (フェーズ 4) - : SDK がサポート終了になると、更新やリリースは受け取れなくなります。以前に公開されたリリースは引き続き公開パッケージマネージャーから入手でき、コードはGitHub に残ります。GitHub リポジトリはアーカイブされる可能性があります。サポートが終了した SDK の使用は、ユーザーの裁量で行われます。ユーザーには新しいメジャーバージョンへのアップグレードをお勧めします。

以下は、SDK メジャーバージョンのライフサイクルを視覚的に示しています。以下に示すタイムラインは例示であり、拘束力はないことに注意してください。
メンテナンスポリシーのタイムライン

依存関係のライフサイクル

AWS SDKsには、言語ランタイム、オペレーティングシステム、サードパーティーのライブラリやフレームワークなど、基盤となる依存関係があります。これらの依存関係は、通常、言語コミュニティや特定のコンポーネントを所有するベンダーに関係しています。各コミュニティまたはベンダーは、自社製品のサポート終了スケジュールを独自に公開しています。

基礎となるサードパーティの依存関係を分類するには、以下の用語が使用されます。

- オペレーティングシステム (OS)：例としては、Amazon Linux AMI、Amazon Linux 2、Windows 2008、Windows 2012、Windows 2016 などがあります。

- 言語ランタイム：例としては、Java 7、Java 8、Java 11、.NET Core、.NET Standard、.NET PCL などがあります。
- サードパーティのライブラリ / フレームワーク：例としては、OpenSSL、.NET Framework 4.5、Java EE などがあります。

コミュニティまたはベンダーが依存関係のサポートを終了した後も、少なくとも 6 か月間は SDK の依存関係をサポートし続けることが当社の方針です。ただし、このポリシーは、特定の依存関係によって異なる場合があります。

> **Note**
>
> AWS は、主要な SDK バージョンを増やすことなく、基盤となる依存関係のサポートを停止する権利を保持します。

コミュニケーションの方法

メンテナンスのお知らせは、以下のように伝えられます。

- 該当するアカウントには、特定の SDK バージョンのサポートを終了する計画を知らせる E メールが送信されます。E メールには、サポート終了までの道筋を概説し、キャンペーンのタイムラインを指定し、アップグレードのガイダンスを提供します。
- AWS API リファレンスドキュメント、ユーザーガイド、SDK 製品マーケティングページ、GitHub readme (複数可) などの SDK ドキュメントが更新され、キャンペーンのタイムラインが示され、影響を受けるアプリケーションのアップグレードに関するガイダンスが提供されます。
- サポートend-of-supportまでの道筋を概説し、キャンペーンのタイムラインを繰り返す AWS ブログ記事が公開されました。
- サポート終了までの道筋を概説し、SDK ドキュメントへのリンクを示す非推奨警告が SDK に追加されました。

使用可能な AWS SDKs「」を参照してくださいバージョンライフサイクル。

AWS SDKsとツールのバージョンライフサイクル

次の表は、使用可能な AWS Software Development Kit (SDK) メジャーバージョンのリストと、それらがメンテナンスライフサイクルのどの段階にあるか、および関連するタイムラインを示しています。 AWS SDKs「」を参照してくださいメンテナンスポリシー。

SDK	メジャーバージョン	現在のフェーズ	一般提供日	注意事項
AWS CLI	1.x	メンテナンスのお知らせ	9/2/2013	詳細と日付については、「お知らせ」を参照してください。
AWS CLI	2.x	一般提供	2/10/2020	
SDK for C++	1.x	一般提供	9/2/2015	
SDK for Go V2	V2 1.x	一般提供	1/19/2021	
SDK for Go	1.x	サポートの終了	11/19/2015	
SDK for Java	1.x	サポートの終了	3/25/2010	
SDK for Java	2.x	一般提供	2018 年 11 月 20 日	
SDK for JavaScript	1.x	サポートの終了	5/6/2013	
SDK for JavaScript	2.x	サポートの終了	6/19/2014	
SDK for JavaScript	3.x	一般提供	12/15/2020	
SDK for Kotlin	1.x	一般提供	11/27/2023	
SDK for .NET	1.x	サポートの終了	2009 年 11 月	

SDK	メジャーバージョン	現在のフェーズ	一般提供日	注意事項
SDK for .NET	2.x	サポートの終了	11/8/2013	
SDK for .NET	3.x	一般提供	7/28/2015	
SDK for .NET	4.x	一般提供	4/28/2025	
SDK for PHP	2.x	サポートの終了	11/2/2012	
SDK for PHP	3.x	一般提供	5/27/2015	
SDK for Python (Boto2)	1.x	サポートの終了	7/13/2011	
SDK for Python (Boto3)	1.x	一般提供	6/22/2015	
SDK for Python (Botocore)	1.x	一般提供	6/22/2015	
SDK for Ruby	1.x	サポートの終了	7/14/2011	
SDK for Ruby	2.x	サポートの終了	2/15/2015	
SDK for Ruby	3.x	一般提供	8/29/2017	
SDK for Rust	1.x	一般提供	11/27/2023	
SDK for Swift	1.x	一般提供	9/17/2024	
Tools for PowerShell	2.x	サポートの終了	11/8/2013	
Tools for PowerShell	3.x	サポートの終了	7/29/2015	
Tools for PowerShell	4.x	一般提供	11/21/2019	

SDK	メジャーバージョン	現在のフェーズ	一般提供日	注意事項
Tools for PowerShell	5.x	一般提供	6/23/2025	

お探しの SDK またはツールが記載されていませんか? 例えば、Encryption SDK、IoT Device SDK、Mobile SDK などはこのガイドに含まれていません。これらの他のツールに関するドキュメントについては、「Toolbox」を参照してください。

AWS SDKsとツールのドキュメント履歴リファレンスガイド

次の表は、「AWS SDK およびツールリファレンスガイド」への重要な追加と更新をまとめたものです。このドキュメントの更新に関する通知については、RSS フィードにサブスクライブできます。

変更	説明	日付
新しい S3 Express One Zone 設定の追加	セッション認証を無効にする新しい S3 Express One Zone 設定を追加しています。	2025 年 10 月 13 日
新しい認証決定木の追加	オプション間の認証決定を支援する新しい決定木を追加しています。	2025 年 9 月 23 日
新しい認証スキーム機能の追加	新しい認証スキーム機能を追加しています。 AWS STS リージョンエンドポイントの更新。	2025 年 8 月 18 日
Tools for PowerShell の新しいバージョンの追加	Tools for PowerShell サポートの最新バージョンをすべての Setting reference Compatibility with AWS SDKs テーブルに追加します。ホストプレフィックスインジェクション機能を追加しました。	2025 年 6 月 23 日
ページタイトルの更新	タイトル、テーブルタイトル、要約、SEO の更新を追加しています。	2025 年 3 月 5 日
ページタイトルの更新	よりわかりやすいタイトルを使用するようにコンテンツを更新しています。	2025 年 2 月 24 日

設定リファレンスへの Swift SDK の追加	Swift SDK サポートをすべての Setting reference Compatibility with AWS SDKs テーブルに追加する。	2024 年 9 月 17 日
SDK for Java 1.x システムプロパティ	AWS SDK for Java 1.x でサポートされている JVM システム設定の詳細を追加します。	2024 年 5 月 30 日
設定の更新	JVM システムの構成設定を追加しています。	2024 年 3 月 27 日
互換性テーブルの更新	SDK サポートの互換性を更新し、IAM Identity Center の手順を更新しています。	2024 年 2 月 20 日
コンテナ認証情報の更新。IMDS の更新。	Amazon EKS 向けのサポートを追加しました。IMDSv1 フォールバックを無効にする設定を追加しました。	2023 年 12 月 29 日
リクエスト圧縮	リクエスト圧縮機能の設定を追加しました。	2023 年 12 月 27 日
互換性に関する表	SDK とツールの機能に関する互換性テーブルが更新され、SDK for Kotlin、SDK for Rust、および AWS Tools for PowerShellが含まれるようになりました。	2023 年 12 月 10 日
認証の更新	SDK およびツールでサポートされている認証方法を更新しました。	2023 年 7 月 1 日

IAM ベストプラクティスの更新	IAM ベストプラクティスに沿ってガイドを更新しました。詳細については、「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。	2023 年 2 月 27 日
SSO の更新	新しい SSO トークン設定の SSO 認証情報を更新しました。	2022 年 11 月 19 日
設定の更新	一般設定と Amazon S3 マルチリージョンアクセスポイントのサポート表を更新しました。	2022 年 11 月 17 日
設定の更新	IMDS クライアントと IMDS 認証情報が明確になるように更新しました。環境変数を更新しました。	2022 年 11 月 4 日
ウェルカムページを更新	Amazon CodeWhisperer の発表。	2022 年 9 月 22 日
シングルサインオンのサービス名の変更	SSO AWS が呼び出されるようになったことを反映する更新 AWS IAM アイデンティティセンター。	2022 年 7 月 26 日
設定の更新	設定ファイルの詳細とサポートされる設定のマイナーな更新。	2022 年 6 月 15 日
更新	本ガイドのほぼすべての部分を大幅に更新。	2022 年 2 月 1 日
初回リリース	このガイドの最初のリリースが一般に公開されています。	2020 年 3 月 13 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。