Guide de référence

AWS SDK et outils

AWS SDK et outils: Guide de référence

Table of Contents

Ce qui est couvert dans le guide de référence sur les outils AWS SDKs et

De nombreux SDKs outils partagent certaines fonctionnalités communes, soit par le biais de spécifications de conception partagées, soit par le biais d'une bibliothèque partagée.

Ce guide contient des informations concernant :

- Configuration globale AWS SDKs et outils— Comment utiliser les `credentials` fichiers partagés ou `config` les variables d'environnement pour configurer vos outils AWS SDKs et.
- Utilisation et outils d'authentification AWS SDKs et d'accès— Déterminez comment votre code ou outil s'authentifie AWS lorsque vous développez avec Services AWS.
- AWS SDKs et référence des paramètres des outils— Référence pour tous les paramètres standardisés disponibles pour l'authentification et la configuration.
- AWS bibliothèques CRT (Common Runtime)— Vue d'ensemble des bibliothèques CRT (AWS Common Runtime) partagées accessibles à presque tous SDKs.
- AWS SDKs et politique de maintenance des outilscouvre la politique de maintenance et le contrôle des versions des kits de développement AWS logiciel (SDKs) et des outils, notamment le mobile et l'Internet des objets (IoT) SDKs, ainsi que leurs dépendances sous-jacentes.

Ce guide de référence AWS SDKs et les outils sont destinés à être une base d'informations applicable à SDKs de multiples outils. Le guide spécifique au SDK ou à l'outil que vous utilisez doit être utilisé en plus des informations présentées ici. Le SDK et les outils suivants contiennent des sections pertinentes dans ce guide :

Si vous utilisez :	Les sections pertinentes de ce guide pour vous sont les suivantes :
• N'importe quel SDK ou outil	AWS SDKs et politique de maintenance des outils
• AWS Cloud9 • AWS CDK • AWS Toolkit pour Azure DevOps • AWS Toolkit for JetBrains	Configuration globale AWS SDKs et outils Utilisation et outils d'authentification AWS SDKs et d'accès

Si vous utilisez :	Les sections pertinentes de ce guide pour vous sont les suivantes :
• AWS Toolkit for Visual Studio • AWS Toolkit for Visual Studio Code • AWS Serverless Application Model • AWS CodeArtifact • AWS CodeBuild • Amazon CodeCatalyst • AWS CodeCommit • AWS CodeDeploy • AWS CodePipeline	AWS SDKs et politique de maintenance des outils
• AWS CLI • AWS SDK pour C++ • AWS SDK pour Go • AWS SDK pour Java • AWS SDK pour JavaScript • AWS SDK pour Kotlin • AWS SDK pour .NET • AWS SDK pour PHP • AWS SDK pour Python (Boto3) • AWS SDK pour Ruby • AWS SDK pour Rust • AWS SDK pour Swift • AWS Tools for Windows PowerShell	Configuration globale AWS SDKs et outils Utilisation et outils d'authentification AWS SDKs et d'accès AWS SDKs et référence des paramètres des outils AWS bibliothèques CRT (Common Runtime) AWS SDKs et politique de maintenance des outils AWS SDKs et cycle de vie des versions des outils

- Pour un aperçu des outils qui peuvent vous aider à développer des applications AWS, consultez la section Outils sur lesquels vous pouvez vous appuyer AWS.
- Pour plus d'informations sur le support, consultez le centre de AWS connaissances.
- Pour AWS la terminologie, voir le AWS glossaire dans la Glossaire AWS référence.

Ressources pour développeurs

Amazon Q Developer est un assistant conversationnel génératif alimenté par l'IA qui peut vous aider à comprendre, créer, étendre et exploiter des applications. AWS Pour accélérer votre développement AWS, le modèle sur lequel repose Amazon Q est complété par AWS du contenu de haute qualité afin de produire des réponses plus complètes, exploitables et référencées. Pour plus d'informations, consultez la section What is Amazon Q Developer? du guide de l'utilisateur Amazon Q Developer.

Notification de télémétrie du kit d'outils

AWS Les boîtes à outils de l'environnement de développement intégré (IDE) sont des plugins et des extensions qui permettent d'accéder aux AWS services de votre IDE. Les plugins et extensions Amazon Q IDE permettent une assistance générative basée sur l'IA dans votre environnement de développement intégré. Pour des informations détaillées sur chacun des kits d'outils IDE, consultez les guides de l'utilisateur du kit d'outils dans le tableau précédent. Pour en savoir plus sur l'utilisation d'Amazon Q dans votre IDE, consultez la rubrique Utilisation d'Amazon Q dans l'IDE du guide du développeur Amazon Q.

AWS IDE Toolkits et Amazon Q peuvent collecter et stocker des données de télémétrie côté client afin d'éclairer les décisions concernant les futures versions de Toolkit AWS et d'Amazon Q. Les données collectées quantifient votre utilisation du AWS Toolkit et d'Amazon Q.

Pour en savoir plus sur les données de télémétrie collectées dans tous les kits d'outils AWS IDE et Amazon Q, consultez le document CommonDefinitions.json dans le référentiel Github. `aws-toolkit-common`

Pour obtenir des informations détaillées sur les données de télémétrie collectées par chacun des kits d'outils AWS IDE et des extensions Amazon Q, consultez les documents de ressources dans les référentiels de boîtes à outils suivants : AWS GitHub

- AWS Boîte à outils Visual Studio avec Amazon Q
- AWS Toolkit for Visual Studio Code et extension Amazon Q pour VS Code
- AWS Toolkit for JetBrains et le plugin Amazon Q pour JetBrains
- Amazon Q pour Eclipse

Certains AWS services accessibles dans les boîtes à AWS outils peuvent collecter des données de télémétrie supplémentaires côté client. Pour des informations détaillées sur le type de données

collectées par chaque AWS service individuel, consultez la rubrique AWS Documentation du service spécifique qui vous intéresse.

Configuration globale AWS SDKs et outils

Avec AWS SDKs d'autres AWS outils de développement, tels que le AWS Command Line Interface (AWS CLI), vous pouvez interagir avec le AWS service APIs. Avant d'essayer, vous devez toutefois configurer le SDK ou l'outil avec les informations dont il a besoin pour effectuer l'opération demandée.

Ces informations incluent les éléments suivants :

- Informations d'identification qui identifient la personne qui appelle l'API. Les informations d'identification sont utilisées pour chiffrer la demande adressée aux AWS serveurs. À l'aide de ces informations, vous AWS confirmez votre identité et pouvez récupérer les politiques d'autorisation qui y sont associées. Il peut ensuite déterminer les actions que vous êtes autorisé à effectuer.
- Autres détails de configuration que vous utilisez pour indiquer au AWS CLI SDK comment traiter la demande, où envoyer la demande (à quel point de terminaison de AWS service) et comment interpréter ou afficher la réponse.

Chaque SDK ou outil prend en charge plusieurs sources que vous pouvez utiliser pour fournir les informations d'identification et de configuration requises. Certaines sources sont propres au SDK ou à l'outil, et vous devez consulter la documentation de cet outil ou de ce SDK pour savoir comment utiliser cette méthode.

Cependant, les outils AWS SDKs et prennent en charge les paramètres courants provenant de sources principales autres que le code lui-même. Cette section couvre les rubriques suivantes :

Rubriques

Utilisation du partage **`config`** et **`credentials`** des fichiers pour une configuration globale AWS SDKs et des outils

Le partage AWS `config` et `credentials` les fichiers constituent le moyen le plus courant de spécifier l'authentification et la configuration d'un AWS SDK ou d'un outil.

Les `credentials` fichiers partagés `config` contiennent un ensemble de profils. Un profil est un ensemble de paramètres de configuration, sous forme de paires clé-valeur, utilisé par AWS SDKs, the AWS Command Line Interface (AWS CLI) et d'autres outils. Les valeurs de configuration sont associées à un profil afin de configurer certains aspects du SDK/tool moment où ce profil est utilisé. Ces fichiers sont « partagés » dans la mesure où les valeurs ont un effet sur les applications, les processus ou SDKs sur l'environnement local d'un utilisateur.

Les fichiers partagés `config` et `credentials` les fichiers sont des fichiers en texte brut contenant uniquement des caractères ASCII (encodés en UTF-8). Ils prennent la forme de ce que l'on appelle généralement des fichiers INI.

Profils

Les paramètres du partage `config` et `credentials` des fichiers sont associés à un profil spécifique. Plusieurs profils peuvent être définis dans le fichier afin de créer différentes configurations de paramètres à appliquer dans différents environnements de développement.

Le `[default]` profil contient les valeurs utilisées par un SDK ou une opération d'outil si aucun profil nommé spécifique n'est spécifié. Vous pouvez également créer des profils distincts auxquels vous pouvez explicitement faire référence par leur nom. Chaque profil peut utiliser des paramètres et des valeurs différents selon les besoins de votre application et de votre scénario.

> **Note**
>
> `[default]`est simplement un profil anonyme. Ce profil est nommé `default` car il s'agit du profil par défaut utilisé par le SDK si l'utilisateur ne spécifie aucun profil. Il ne fournit pas de valeurs par défaut héritées aux autres profils. Si vous définissez un élément dans le `[default]` profil et que vous ne le définissez pas dans un profil nommé, la valeur n'est pas définie lorsque vous utilisez le profil nommé.

Définissez un profil nommé

Le `[default]` profil et plusieurs profils nommés peuvent exister dans le même fichier. Utilisez le paramètre suivant pour sélectionner les paramètres du profil utilisés par votre SDK ou votre outil lors de l'exécution de votre code. Les profils peuvent également être sélectionnés dans le code ou par commande lorsque vous travaillez avec le AWS CLI.

Configurez cette fonctionnalité en définissant l'une des options suivantes :

`AWS_PROFILE`- variable d'environnement

Lorsque cette variable d'environnement est définie sur un profil nommé ou « par défaut », tous les codes et AWS CLI commandes du SDK utilisent les paramètres de ce profil.

Exemple Linux/macOS de définition de variables d'environnement via la ligne de commande :

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

Exemple Windows de définition de variables d'environnement via la ligne de commande :

```
setx AWS_PROFILE "my_default_profile_name"
```

`aws.profile`- Propriété du système JVM

Pour le SDK pour Kotlin sur la JVM et le SDK pour Java 2.x, vous pouvez définir la propriété du système. `aws.profile` Lorsque le SDK crée un client de service, il utilise les paramètres du profil nommé, sauf si le paramètre est remplacé dans le code. Le SDK pour Java 1.x ne prend pas en charge cette propriété système.

> **Note**
>
> Si votre application se trouve sur un serveur exécutant plusieurs applications, nous vous recommandons de toujours utiliser des profils nommés plutôt que le profil par défaut. Le profil par défaut est automatiquement sélectionné par n'importe quelle AWS application de l'environnement et est partagé entre elles. Ainsi, si quelqu'un d'autre met à jour le profil par défaut de son application, cela peut avoir un impact involontaire sur les autres. Pour éviter cela, définissez un profil nommé dans le `config` fichier partagé, puis utilisez-le dans votre application en définissant le profil nommé dans votre code. Vous pouvez utiliser la variable

d'environnement ou la propriété du système JVM pour définir le profil nommé si vous savez que sa portée n'affecte que votre application.

Format du fichier de configuration

Le `config` fichier est organisé en sections. Une section est une collection nommée de paramètres qui continue jusqu'à ce qu'une autre ligne de définition de section soit rencontrée.

Le `config` fichier est un fichier en texte brut qui utilise le format suivant :

- Toutes les entrées d'une section prennent la forme générale `setting-name=value`.
- Les lignes peuvent être commentées en commençant par un hashtag (#).

Types de sections

Une définition de section est une ligne qui donne un nom à un ensemble de paramètres. Les lignes de définition de section commencent et se terminent par des crochets (`[]`). À l'intérieur des crochets, il y a un identifiant de type de section et un nom personnalisé pour la section. Vous pouvez utiliser des lettres, des chiffres, des traits d'union (`-`) et des traits de soulignement (`_`), mais pas d'espaces.

Type de section : **`default`**

Exemple de ligne de définition de section : `[default]`

`[default]`est le seul profil qui ne nécessite pas l'identifiant de `profile` section.

L'exemple suivant montre un `config` fichier de base avec un `[default]` profil. Il définit le `region`réglage. Tous les paramètres qui suivent cette ligne, jusqu'à ce qu'une autre définition de section soit trouvée, font partie de ce profil.

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

Type de section : **`profile`**

Exemple de ligne de définition de section : `[profile dev]`

La ligne de définition de `profile` section est un groupe de configuration nommé que vous pouvez appliquer à différents scénarios de développement. Pour mieux comprendre les profils nommés, consultez la section précédente sur les profils.

L'exemple suivant montre un `config` fichier avec une ligne de définition de `profile` section et un profil nommé appelé`foo`. Tous les paramètres qui suivent cette ligne, jusqu'à ce qu'une autre définition de section soit trouvée, font partie de ce profil nommé.

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

Certains paramètres possèdent leur propre groupe imbriqué de sous-paramètres, tels que le `s3` paramètre et les sous-paramètres de l'exemple suivant. Associez les sous-paramètres au groupe en les indentant d'un ou de plusieurs espaces.

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

Type de section : **`sso-session`**

Exemple de ligne de définition de section : `[sso-session my-sso]`

La ligne de définition de `sso-session` section nomme un groupe de paramètres que vous utilisez pour configurer un profil afin de résoudre les AWS informations d'identification utilisées AWS IAM Identity Center. Pour plus d'informations sur la configuration de l'authentification unique, consultezUtilisation d'IAM Identity Center pour authentifier le AWS SDK et les outils. Un profil est lié à une `sso-session` section par une paire clé-valeur où `sso-session` est la clé et le nom de votre `sso-session` section est la valeur, par exemple. `sso-session = <name-of-sso-session-section>`

L'exemple suivant configure un profil qui obtiendra des informations d' AWS identification à court terme pour le rôle IAM « SampleRole » dans le compte « 111122223333 » à l'aide d'un jeton du « my-sso ». La section « my-sso » est référencée dans la `sso-session` `profile` section par son nom à l'aide de la `sso-session` clé.

```
[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
```

Type de section : **`services`**

Exemple de ligne de définition de section : `[services dev]`

> **Note**
>
> La `services` section prend en charge les personnalisations de point de terminaison spécifiques au service et n'est disponible que dans les SDKs outils qui incluent cette fonctionnalité. Pour savoir si cette fonctionnalité est disponible pour votre SDK, consultez la section relative aux points de terminaison Support par AWS SDKs et outils spécifiques au service.

La ligne de définition de `services` section nomme un groupe de paramètres qui configurent les points de terminaison personnalisés pour les Service AWS demandes. Un profil est lié à une `services` section par une paire clé-valeur où `services` est la clé et le nom de votre `services` section est la valeur, par exemple. `services = <name-of-services-section>`

La `services` section est ensuite séparée en sous-sections par des `<SERVICE> =` lignes, où se `<SERVICE>` trouve la clé d' Service AWS identification. L' Service AWS identifiant est basé sur le modèle d'API `serviceId` en remplaçant tous les espaces par des traits de soulignement et en minuscules toutes les lettres. Pour obtenir la liste de toutes les clés d'identification de service à utiliser dans la section `services`, consultez Identifiants pour les points de terminaison spécifiques au service. La clé d'identification du service est suivie de paramètres imbriqués, chacun sur sa propre ligne et indenté de deux espaces.

L'exemple suivant utilise une `services` définition pour configurer le point de terminaison à utiliser pour les demandes adressées uniquement au Amazon DynamoDB service. La `"local-dynamodb"` `services` section est référencée dans la `profile` section par son nom à l'aide de la `services` clé. La clé Service AWS d'identification est`dynamodb`. La sous-section des Amazon DynamoDB services commence sur la ligne`dynamodb =` . Toutes les lignes indentées qui suivent immédiatement sont incluses dans cette sous-section et s'appliquent à ce service.

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

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

Pour plus d'informations sur la configuration personnalisée des points de terminaison, consultezPoints de terminaison spécifiques au service.

Format du fichier d'informations d'identification

Les règles du `credentials` fichier sont généralement identiques à celles du `config` fichier, sauf que les sections du profil ne commencent pas par le mot`profile`. Utilisez uniquement le nom du profil lui-même entre crochets. L'exemple suivant montre un `credentials` fichier dont la section de profil nommée est appelée`foo`.

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

Seuls les paramètres considérés comme « secrets » ou sensibles peuvent être enregistrés dans le `credentials` fichier :`aws_access_key_id`,`aws_secret_access_key`, et`aws_session_token`. Bien que ces paramètres puissent également être placés dans le `config` fichier partagé, nous vous recommandons de conserver ces valeurs sensibles dans un `credentials` fichier distinct. Ainsi, vous pouvez fournir des autorisations distinctes pour chaque fichier, si nécessaire.

L'exemple suivant montre un `credentials` fichier de base avec un `[default]` profil. Il définit les paramètres `aws_access_key_id`, `aws_secret_access_key`, et `aws_session_token` globaux.

```
[default]
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
aws_session_token=IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZVERYLONGSTRI
```

Que vous utilisiez un profil nommé ou un « `default` » dans votre `credentials` fichier, tous les paramètres de ce fichier seront combinés avec ceux de votre `config` fichier qui utilisent le même nom de profil. S'il existe des informations d'identification dans les deux fichiers pour un profil partageant le même nom, les clés du fichier d'informations d'identification ont priorité.

Recherche et modification de l'emplacement du partage, des `credentials` fichiers `config` AWS SDKs et des outils

Les fichiers partagés sont AWS `config` des `credentials` fichiers en texte brut contenant des informations de configuration pour les outils AWS SDKs et. Les fichiers résident localement dans votre environnement et sont utilisés automatiquement par le code du SDK ou par AWS CLI les commandes que vous exécutez dans cet environnement. Par exemple, sur votre propre ordinateur ou lors du développement sur une instance Amazon Elastic Compute Cloud.

Lorsque le SDK ou l'outil s'exécute, il vérifie la présence de ces fichiers et charge tous les paramètres de configuration disponibles. Si les fichiers n'existent pas déjà, un fichier de base est automatiquement créé par le SDK ou l'outil.

Par défaut, les fichiers se trouvent dans un dossier nommé `.aws` qui est placé dans votre dossier home ou dans celui de l'utilisateur.

Système d’exploitation	Emplacement et nom par défaut des fichiers
Linux et macOS	`~/.aws/config` `~/.aws/credentials`
Windows	`%USERPROFILE%\.aws\config` `%USERPROFILE%\.aws\credentials`

Résolution du répertoire personnel

~n'est utilisé pour la résolution du répertoire personnel que lorsqu'il :

- Démarre le chemin
- Est immédiatement suivi par / ou par un séparateur spécifique à la plate-forme. Sous Windows, ~/ et ~\ les deux se résolvent dans le répertoire de base.

Lors de la détermination du répertoire de base, les variables suivantes sont vérifiées :

- (Toutes les plateformes) La variable d'`HOME`environnement

- (Plateformes Windows) La variable d'`USERPROFILE`environnement
- (Plateformes Windows) La concaténation de variables d'`HOMEPATH`environnement `HOMEDRIVE` et de variables d'environnement () `$HOMEDRIVE$HOMEPATH`
- (Facultatif par SDK ou outil) Fonction ou variable de résolution du chemin d'accueil spécifique au SDK ou à l'outil

Dans la mesure du possible, si le répertoire personnel d'un utilisateur est spécifié au début du chemin (par exemple,`~username/`), il est résolu dans le répertoire personnel du nom d'utilisateur demandé (par exemple,`/home/username/.aws/config`).

Modifier l'emplacement par défaut de ces fichiers

Vous pouvez utiliser l'une des méthodes suivantes pour modifier l'emplacement à partir duquel ces fichiers sont chargés par le SDK ou l'outil.

Utiliser des variables d'environnement

Les variables d'environnement suivantes peuvent être définies pour modifier l'emplacement ou le nom de ces fichiers de la valeur par défaut à une valeur personnalisée :

- `config`variable d'environnement de fichier : **`AWS_CONFIG_FILE`**
- `credentials`variable d'environnement de fichier : **`AWS_SHARED_CREDENTIALS_FILE`**

Linux/macOS

Vous pouvez spécifier un autre emplacement en exécutant les commandes d'exportation suivantes sous Linux ou macOS.

```
$ 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

Vous pouvez spécifier un autre emplacement en exécutant les commandes setx suivantes sous Windows.

```
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
```

Pour plus d'informations sur la configuration de votre système à l'aide de variables d'environnement, consultezUtilisation de variables d'environnement pour une configuration globale AWS SDKs et des outils.

Utiliser les propriétés du système JVM

Pour le SDK pour Kotlin exécuté sur la JVM et pour le SDK for Java 2.x, vous pouvez définir les propriétés du système JVM suivantes pour modifier l'emplacement ou le nom de ces fichiers de la valeur par défaut à une valeur personnalisée :

- `config`propriété du système JVM de fichiers : **`aws.configFile`**
- `credentials`variable d'environnement de fichier : **`aws.sharedCredentialsFile`**

Pour obtenir des instructions sur la façon de définir les propriétés du système JVM, consultezthe section called “Comment définir les propriétés du système JVM”. Le SDK pour Java 1.x ne prend pas en charge ces propriétés système.

Utilisation de variables d'environnement pour une configuration globale AWS SDKs et des outils

Les variables d'environnement constituent un autre moyen de spécifier les options de configuration et les informations d'identification lors de l'utilisation AWS SDKs d'outils. Les variables d'environnement peuvent être utiles pour créer des scripts ou définir temporairement un profil nommé par défaut. Pour la liste des variables d'environnement prises en charge par la plupart SDKs, consultezListe des variables d'environnement.

Priorité d’options

- Si vous spécifiez un paramètre à l'aide de sa variable d'environnement, il remplace toute valeur chargée à partir d'un profil dans le partage AWS `config` et `credentials` les fichiers.
- Si vous spécifiez un paramètre à l'aide d'un paramètre sur la ligne de AWS CLI commande, il remplace toute valeur de la variable d'environnement correspondante ou d'un profil du fichier de configuration.

Comment définir les variables d'environnement

Les exemples suivants montrent comment vous pouvez configurer des variables d'environnement pour l'utilisateur par défaut.

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
```

La définition de la variable d'environnement permet de modifier la valeur utilisée jusqu'à la fin de votre session shell, ou jusqu'à ce que vous définissiez la variable sur une autre valeur. Vous pouvez rendre les variables persistantes dans de futures sessions en les définissant dans votre script de démarrage de shell.

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
```

L'utilisation `set` pour définir une variable d'environnement modifie la valeur utilisée jusqu'à la fin de la session d'invite de commande en cours ou jusqu'à ce que vous définissiez une valeur différente pour la variable. Le fait `setx`de définir une variable d'environnement modifie la valeur utilisée à la fois dans la session d'invite de commande en cours et dans toutes les sessions d'invite de commandes que vous créez après avoir exécuté la commande. Cela n'affecte pas les autres shells de commande qui sont déjà en cours d'exécution lorsque vous exécutez la commande.

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"
```

Si vous définissez une variable d'environnement à l' PowerShell invite, comme indiqué dans les exemples précédents, elle enregistre la valeur uniquement pendant la durée de la session en cours. Pour que le paramètre de variable d'environnement soit persistant dans toutes les sessions PowerShell et dans les sessions d'invite de commande, stockez-le à l'aide de l'application système du Panneau de configuration. Vous pouvez également définir la variable pour toutes les PowerShell sessions futures en l'ajoutant à votre PowerShell profil. Consultez la PowerShell documentation pour plus d'informations sur le stockage des variables d'environnement ou leur persistance d'une session à l'autre.

Configuration de variables d'environnement sans serveur

Si vous utilisez une architecture sans serveur pour le développement, vous disposez d'autres options pour définir les variables d'environnement. En fonction de votre conteneur, vous pouvez utiliser différentes stratégies pour exécuter le code dans ces conteneurs afin de voir et d'accéder aux variables d'environnement, comme dans les environnements non cloud.

Par exemple, avec AWS Lambda, vous pouvez définir directement des variables d'environnement. Pour plus de détails, consultez la section Utilisation de variables d' AWS Lambda environnement dans le Guide du AWS Lambda développeur.

Dans Serverless Framework, vous pouvez souvent définir des variables d'environnement du SDK dans le `serverless.yml` fichier sous la clé du fournisseur sous le paramètre d'environnement. Pour plus d'informations sur le `serverless.yml` fichier, consultez la section Paramètres généraux des fonctions dans la documentation du Serverless Framework.

Quel que soit le mécanisme que vous utilisez pour définir les variables d'environnement du conteneur, certaines sont réservées par le conteneur, comme celles décrites pour Lambda dans la section Variables d'environnement d'exécution définies. Consultez toujours la documentation officielle du conteneur que vous utilisez pour déterminer comment les variables d'environnement sont traitées et s'il existe des restrictions.

Utilisation des propriétés du système JVM pour configurer AWS SDK pour Java globalement et AWS SDK pour Kotlin

Les propriétés du système JVM fournissent un autre moyen de spécifier les options de configuration et les informations d'identification pour SDKs celles exécutées sur la JVM, telles que le AWS SDK

pour Java et le. AWS SDK pour Kotlin Pour obtenir la liste des propriétés du système JVM prises en charge par SDKs, consultez la section Référence des paramètres.

Priorité d'options

- Si vous spécifiez un paramètre à l'aide de sa propriété système JVM, il remplace toute valeur trouvée dans les variables d'environnement ou chargée à partir d'un profil dans l'AWS `config` et `credentials` les fichiers partagés.
- Si vous spécifiez un paramètre à l'aide de sa variable d'environnement, il remplace toute valeur chargée à partir d'un profil dans l'AWS `config` et `credentials` les fichiers partagés.

Comment définir les propriétés du système JVM

Vous pouvez définir les propriétés du système JVM de plusieurs manières.

Sur la ligne de commande

Définissez les propriétés du système JVM sur la ligne de commande lorsque vous appelez la `java` commande à l'aide du commutateur. `-D` La commande suivante configure le Région AWS globalement pour tous les clients du service, sauf si vous remplacez explicitement la valeur dans le code.

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

Si vous devez définir plusieurs propriétés du système JVM, spécifiez le `-D` commutateur plusieurs fois.

Avec une variable d'environnement

Si vous ne pouvez pas accéder à la ligne de commande pour appeler la machine virtuelle Java afin d'exécuter votre application, vous pouvez utiliser la variable d'`JAVA_TOOL_OPTIONS`environnement pour configurer les options de ligne de commande. Cette approche est utile dans des situations telles que l'exécution d'une AWS Lambda fonction sur le runtime Java ou l'exécution de code dans une machine virtuelle Java intégrée.

L'exemple suivant configure le Région AWS globalement pour tous les clients du service, sauf si vous remplacez explicitement la valeur dans le code.

Linux, macOS, or Unix

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

La définition de la variable d'environnement permet de modifier la valeur utilisée jusqu'à la fin de votre session shell, ou jusqu'à ce que vous définissiez la variable sur une autre valeur. Vous pouvez rendre les variables persistantes dans de futures sessions en les définissant dans votre script de démarrage de shell.

Windows Command Prompt

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

L'utilisation `set` pour définir une variable d'environnement modifie la valeur utilisée jusqu'à la fin de la session d'invite de commande en cours ou jusqu'à ce que vous définissiez une valeur différente pour la variable. Le fait `setx`de définir une variable d'environnement modifie la valeur utilisée à la fois dans la session d'invite de commande en cours et dans toutes les sessions d'invite de commandes que vous créez après avoir exécuté la commande. Cela n'affecte pas les autres shells de commande qui sont déjà en cours d'exécution lorsque vous exécutez la commande.

Au moment de l'exécution

Vous pouvez également définir les propriétés du système JVM lors de l'exécution dans le code en utilisant la `System.setProperty` méthode illustrée dans l'exemple suivant.

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

> **Important**
>
> Définissez les propriétés du système JVM avant d'initialiser les clients du service SDK, sinon les clients de service peuvent utiliser d'autres valeurs.

Utilisation et outils d'authentification AWS SDKs et d'accès

Lorsque vous développez une application AWS SDK ou que vous utilisez AWS des outils à utiliser Services AWS, vous devez définir la manière dont votre code ou outil s'authentifie. AWS Vous pouvez configurer l'accès programmatique aux AWS ressources de différentes manières, en fonction de l'environnement dans lequel le code s'exécute et de l' AWS accès dont vous disposez.

Les options ci-dessous font partie de la chaîne des fournisseurs d'informations d'identification. Cela signifie qu'en configurant votre partage AWS `config` et vos `credentials` fichiers en conséquence, votre AWS SDK ou outil découvrira et utilisera automatiquement cette méthode d'authentification.

Choisissez une méthode pour authentifier le code de votre application

Choisissez une méthode pour authentifier les appels effectués AWS par votre application.

Exécutez-vous du code INSIDE an Service AWS (tel qu'Amazon EC2, Lambda, Amazon ECS, Amazon EKS) ? CodeBuild

Si votre code continue à s'exécuter AWS, les informations d'identification peuvent être automatiquement mises à la disposition de votre application. Par exemple, si votre application est hébergée sur Amazon Elastic Compute Cloud et qu'un rôle IAM est associé à cette ressource, les informations d'identification sont automatiquement mises à la disposition de votre application. De même, si vous utilisez des conteneurs Amazon ECS ou Amazon EKS, les informations d'identification définies pour le rôle IAM peuvent être automatiquement obtenues par le code exécuté dans le conteneur via la chaîne de fournisseurs d'informations d'identification du SDK.

Votre code se trouve-t-il dans une instance Amazon Elastic Compute Cloud ?

Utilisation des rôles IAM pour authentifier les applications déployées sur Amazon EC2— Utilisez les rôles IAM pour exécuter votre application en toute sécurité sur une instance Amazon EC2.

Votre code se trouve-t-il dans une AWS Lambda fonction ?

Lambda crée un rôle d'exécution avec des autorisations minimales lorsque vous créez une fonction Lambda. Le AWS SDK ou l'outil utilise ensuite automatiquement le rôle IAM attaché au Lambda lors de l'exécution, via l'environnement d'exécution Lambda.

Votre code se trouve-t-il dans Amazon Elastic Container Service (sur Amazon EC2 ou pour AWS Fargate Amazon ECS) ?

Utilisez le rôle IAM pour la tâche. Vous devez créer un rôle de tâche et le spécifier dans votre définition de tâche Amazon ECS. Le AWS SDK ou l'outil utilise ensuite automatiquement le rôle IAM attribué à la tâche lors de l'exécution, via les métadonnées Amazon ECS.

Votre code se trouve-t-il dans Amazon Elastic Kubernetes Service ?

Nous vous recommandons d'utiliser Amazon EKS Pod Identities.

Remarque : si vous pensez que les rôles IAM pour les comptes de service (IRSA) sont mieux adaptés à vos besoins spécifiques, consultez la section Comparaison entre EKS Pod Identity et IRSA dans le guide de l'utilisateur Amazon EKS.

Votre code s'exécute-t-il dans AWS CodeBuild

Consultez la section Utilisation de politiques basées sur l'identité pour. CodeBuild

Votre code se trouve-t-il dans un autre Service AWS ?

Consultez le guide dédié à votre Service AWS. Lorsque vous exécutez du code AWS, la chaîne de fournisseurs d'informations d'identification du SDK peut automatiquement obtenir et actualiser les informations d'identification pour vous.

Créez-vous des applications mobiles ou des applications Web basées sur le client ?

Si vous créez des applications mobiles ou des applications Web basées sur des clients qui nécessitent un accès à AWS, créez votre application de manière à ce qu'elle demande des informations d'identification de AWS sécurité temporaires de manière dynamique à l'aide de la fédération d'identité Web.

Lors de l'utilisation de la fédération d'identité web, il n'est pas nécessaire de créer de code de connexion personnalisé ni de gérer vos propres identités utilisateur. Les utilisateurs de l'application peuvent plutôt se connecter à l'aide d'un fournisseur d'identité externe (IdP) connu, tel que Login with Amazon, Facebook, Google ou tout autre IdP compatible avec OpenID Connect (OIDC).
Ils peuvent recevoir un jeton d'authentification, puis échanger ce jeton contre des informations d'identification de sécurité temporaires associées à un rôle IAM autorisé à utiliser les ressources de votre Compte AWS. AWS

Pour savoir comment le configurer pour votre SDK ou votre outil, consultezAssumer un rôle avec l'identité Web ou OpenID Connect pour l'authentification et les outils AWS SDKs.

Pour les applications mobiles, pensez à utiliser Amazon Cognito. Amazon Cognito agit en tant que courtier d'identité et effectue une grande partie du travail de fédération à votre place. Pour plus d'informations, consultez la section Utilisation d'Amazon Cognito pour les applications mobiles dans le guide de l'utilisateur IAM.

Développez-vous et exécutez-vous le code LOCALEMENT ?

Nous recommandons Utilisation des informations d'identification de la console pour l'authentification AWS SDKs et des outils

Après un flux d'authentification rapide basé sur un navigateur, génère AWS automatiquement des informations d'identification temporaires qui fonctionnent avec les outils de développement locaux tels que la AWS CLI et Outils AWS pour PowerShell . AWS SDKs

Si vous utilisez Identity Center pour accéder au AWS compte

Utilisez IAM Identity Center pour authentifier le AWS SDK et les outils si vous avez déjà accès aux AWS comptes and/or nécessaires pour gérer l'accès de votre personnel. Pour des raisons de sécurité, nous vous recommandons AWS Organizations d'utiliser IAM Identity Center pour gérer l'accès à tous vos AWS comptes. Vous pouvez créer des utilisateurs dans IAM Identity Center, utiliser Microsoft Active Directory, utiliser un fournisseur d'identité (IdP) SAML 2.0 ou fédérer individuellement votre IdP avec des comptes. AWS Pour vérifier si votre région prend en charge le centre d'identité IAM, consultez la section Points de terminaison et quotas du centre d'identité Utilisation d'IAM Identity Center pour authentifier le AWS SDK et les outils IAM dans le manuel Amazon Web Services General Reference.

Si vous recherchez d'autres moyens d'authentification

Créez un utilisateur IAM le moins privilégié autorisé à accéder à votre rôle `sts:AssumeRole` cible. Configurez ensuite votre profil pour qu'il assume un rôle à l'aide d'une `source_profile` configuration pour cet utilisateur.

Vous pouvez également utiliser des informations d'identification IAM temporaires via des variables d'environnement ou le fichier AWS d'informations d'identification partagé. Consultez la section Utilisation d'informations d'identification à court terme pour l'authentification AWS SDKs et les outils.

Remarque : dans les environnements sandbox ou d'apprentissage uniquement, vous pouvez envisager d'utiliser des informations d'identification à long terme pour l'authentification AWS SDKs et les outils.

Ce code s'exécute-t-il sur site ou sur une machine virtuelle hybride/à la demande (par exemple, un serveur qui lit ou écrit sur Amazon S3, ou un déploiement de Jenkins dans le cloud) ?

Utilisez-vous des certificats clients X.509 ?

Oui : vous voyezUtilisation d'IAM Roles Anywhere pour l'authentification et les outils AWS SDKs. Vous pouvez utiliser IAM Roles Anywhere pour obtenir des informations d'identification de sécurité temporaires dans IAM pour les charges de travail telles que les serveurs, les conteneurs et les applications qui s'exécutent en dehors de. AWS Pour utiliser IAM Roles Anywhere, vos charges de travail doivent utiliser des certificats X.509.

L'environnement peut-il se connecter en toute sécurité à un fournisseur d'identité fédéré (tel que Microsoft Entra ou Okta) pour demander des informations d'identification temporaires ? AWS

Oui : utiliser Fournisseur d'identifiants de processus

Fournisseur d'identifiants de processusÀ utiliser pour récupérer automatiquement les informations d'identification lors de l'exécution. Ces systèmes peuvent utiliser un outil d'assistance ou un plug-in pour obtenir les informations d'identification et peuvent assumer un rôle IAM en arrière-plan en utilisant. `sts:AssumeRole`

Non : utilisez des informations d'identification temporaires injectées via AWS Secrets Manager

Utilisez des informations d'identification temporaires injectées via AWS Secrets Manager. Pour connaître les options permettant d'obtenir des clés d'accès de courte durée, consultez la section Demander des informations d'identification de sécurité temporaires dans le guide de l'utilisateur IAM. Pour connaître les options relatives au stockage de ces informations d'identification temporaires, consultezAWS clés d'accès.

Vous pouvez utiliser ces informations d'identification pour récupérer en toute sécurité des autorisations d'application plus étendues auprès de Secrets Manager, où vos secrets de production ou vos informations d'identification à long terme basées sur les rôles peuvent être stockés.

Utilisez-vous un outil tiers qui n'est pas inclus AWS ?

Utilisez la documentation rédigée par votre fournisseur tiers pour obtenir les meilleurs conseils sur l'obtention des informations d'identification.

Si votre tiers n'a pas fourni de documentation, pouvez-vous injecter des informations d'identification temporaires en toute sécurité ?

Oui : utilisez des variables d'environnement et des AWS STS informations d'identification temporaires.

Non : utilisez des clés d'accès statiques stockées dans un gestionnaire de secrets chiffrés (dernier recours).

Méthodes d’authentification

Méthodes d'authentification pour le code exécuté dans un AWS environnement

Si votre code continue à s'exécuter AWS, les informations d'identification peuvent être automatiquement mises à la disposition de votre application. Par exemple, si votre application est hébergée sur Amazon Elastic Compute Cloud et qu'un rôle IAM est associé à cette ressource, les informations d'identification sont automatiquement mises à la disposition de votre application. De même, si vous utilisez des conteneurs Amazon ECS ou Amazon EKS, les informations d'identification définies pour le rôle IAM peuvent être automatiquement obtenues par le code exécuté dans le conteneur via la chaîne de fournisseurs d'informations d'identification du SDK.

- Utilisation des rôles IAM pour authentifier les applications déployées sur Amazon EC2— Utilisez les rôles IAM pour exécuter votre application en toute sécurité sur une instance Amazon EC2.
- Vous pouvez interagir par programmation avec AWS IAM Identity Center de la manière suivante :
  - Permet AWS CloudShelld'exécuter AWS CLI des commandes depuis la console.
  - Pour essayer un espace de collaboration basé sur le cloud pour les équipes de développement de logiciels, pensez à utiliser Amazon CodeCatalyst.

Authentification via un fournisseur d'identité basé sur le Web - Applications Web mobiles ou basées sur le client

Si vous créez des applications mobiles ou des applications Web basées sur des clients qui nécessitent un accès à AWS, créez votre application de manière à ce qu'elle demande des informations d'identification de AWS sécurité temporaires de manière dynamique à l'aide de la fédération d'identité Web.

Lors de l'utilisation de la fédération d'identité web, il n'est pas nécessaire de créer de code de connexion personnalisé ni de gérer vos propres identités utilisateur. Les utilisateurs de l'application

peuvent plutôt se connecter à l'aide d'un fournisseur d'identité externe (IdP) connu, tel que Login with Amazon, Facebook, Google ou tout autre IdP compatible avec OpenID Connect (OIDC). Ils peuvent recevoir un jeton d'authentification, puis échanger ce jeton contre des informations d'identification de sécurité temporaires associées à un rôle IAM autorisé à utiliser les ressources de votre Compte AWS. AWS

Pour savoir comment le configurer pour votre SDK ou votre outil, consultezAssumer un rôle avec l'identité Web ou OpenID Connect pour l'authentification et les outils AWS SDKs.

Pour les applications mobiles, pensez à utiliser Amazon Cognito. Amazon Cognito agit en tant que courtier d'identité et effectue une grande partie du travail de fédération à votre place. Pour plus d'informations, consultez la section Utilisation d'Amazon Cognito pour les applications mobiles dans le guide de l'utilisateur IAM.

Méthodes d'authentification pour le code exécuté localement (pas dans AWS)

- Utilisation des informations d'identification de la console pour l'authentification AWS SDKs et des outils— Cette fonctionnalité fonctionne à la fois avec l'interface de ligne de AWS commande et avec les outils pour PowerShell et vous fournit des informations d'identification actualisables qui fonctionnent avec les outils de développement locaux tels que la AWS CLI, les outils pour PowerShell et. AWS
- Utilisation d'IAM Identity Center pour authentifier le AWS SDK et les outils— En tant que bonne pratique en matière de sécurité, nous vous recommandons AWS Organizations d'utiliser IAM Identity Center pour gérer l'accès de tous vos Comptes AWS utilisateurs. Vous pouvez créer des utilisateurs dans AWS IAM Identity Center, utiliser Microsoft Active Directory, utiliser un fournisseur d'identité (IdP) SAML 2.0 ou fédérer individuellement votre IdP avec. Comptes AWS Pour vérifier si votre région prend en charge le centre d'identité IAM, consultez la section AWS IAM Identity Center Points de terminaison et quotas dans le. Référence générale d'Amazon Web Services
- Utilisation d'IAM Roles Anywhere pour l'authentification et les outils AWS SDKs— Vous pouvez utiliser IAM Roles Anywhere pour obtenir des informations d'identification de sécurité temporaires dans IAM pour les charges de travail telles que les serveurs, les conteneurs et les applications qui s'exécutent en dehors de. AWS Pour utiliser IAM Roles Anywhere, vos charges de travail doivent utiliser des certificats X.509.
- Assumer un rôle avec des AWS informations d'identification pour l'authentification AWS SDKs et des outils— Vous pouvez assumer un rôle IAM pour accéder temporairement à AWS des ressources auxquelles vous n'auriez peut-être pas accès autrement.

- Utilisation de clés AWS d'accès pour l'authentification AWS SDKs et d'outils— D'autres options peuvent être moins pratiques ou augmenter les risques de sécurité pour vos AWS ressources.

Plus d'informations sur la gestion des accès

Le guide de l'utilisateur IAM contient les informations suivantes sur le contrôle sécurisé de l'accès aux AWS ressources :

- Identités IAM (utilisateurs, groupes d'utilisateurs et rôles) : comprenez les bases des identités dans AWS.
- Meilleures pratiques de sécurité en matière d'IAM : recommandations de sécurité à suivre lors du développement d' AWS applications selon le modèle de responsabilité partagée.

Référence générale d'Amazon Web ServicesIl contient des éléments de base sur les points suivants :

- Comprendre et obtenir vos AWS informations d'identification : accédez aux principales options et pratiques de gestion pour l'accès par console et par programmation.

Plug-in de propagation d'identité sécurisée (TIP) d'IAM Identity Center pour y accéder Services AWS

- Utiliser le plugin TIP pour accéder Services AWS— Si vous créez une application pour Amazon Q Business ou un autre service qui prend en charge la propagation fiable des identités et que vous utilisez le AWS SDK pour Java ou le AWS SDK pour JavaScript, vous pouvez utiliser le plug-in TIP pour une expérience d'autorisation rationalisée.

ID de constructeur AWS

Vos ID de constructeur AWS compléments à ceux Comptes AWS que vous possédez déjà ou que vous souhaitez créer. Alors qu'un Compte AWS agit comme un conteneur pour les AWS ressources que vous créez et fournit une limite de sécurité pour ces ressources, vous vous ID de constructeur AWS représente en tant qu'individu. Vous pouvez vous connecter ID de constructeur AWS pour accéder à des outils et services de développement tels qu'Amazon Q et Amazon CodeCatalyst.

- ID de constructeur AWS Connectez-vous avec le guide de l'Connexion à AWS utilisateur : découvrez comment créer et utiliser un Builder ID ID de constructeur AWS et découvrez ce que fournit le Builder ID.

- CodeCatalystconcepts - ID de constructeur AWS dans le guide de CodeCatalyst l'utilisateur Amazon - Découvrez comment CodeCatalyst utilise un ID de constructeur AWS.

Utilisation des informations d'identification de la console pour l'authentification AWS SDKs et des outils

L'utilisation des informations d'identification de console est la méthode recommandée pour fournir des AWS informations d'identification lors du développement d'une AWS application dans votre environnement local ou dans d'autres environnements de services non AWS informatiques. Si vous développez sur une AWS ressource, telle qu'Amazon Elastic Compute Cloud (Amazon EC2) AWS CloudShell, nous vous recommandons d'obtenir des informations d'identification auprès de ce service.

Vous pouvez également vous authentifier via IAM Identity Center. Utilisation d'IAM Identity Center pour authentifier le AWS SDK et les outils Cette option est un moyen courant pour les entreprises de gérer l'accès de leur personnel et nécessite l'activation d'Identity Center.

Fonctionnement

La connexion pour le développement AWS local à l'aide des informations d'identification de console vous permet d'utiliser vos informations de connexion existantes à la console de AWS gestion pour un accès programmatique aux AWS services. Après un flux d'authentification basé sur un navigateur, AWS génère des informations d'identification temporaires qui fonctionnent avec les outils de développement locaux tels que la AWS CLI, Tools for PowerShell et. AWS SDKs Cette fonctionnalité simplifie le processus de configuration et de gestion des informations d'identification de la AWS CLI, en particulier si vous préférez l'authentification interactive à la gestion des clés d'accès à long terme.

Ce processus vous permet de vous authentifier à l'aide des informations d'identification root créées lors de la configuration initiale du compte, des utilisateurs IAM ou d'une identité fédérée auprès de votre fournisseur d'identité.

Si vous l'utilisez SDKs pour le développement, les clients du SDK utiliseront les informations d'identification temporaires via leAWS SDKs et outils, fournisseurs d'accréditations standardisés. Vous pouvez également configurer leFournisseur d'identifiants de connexion.

L'authentification via la commande de connexion est prise en charge à la fois par la AWS CLI et par les outils pour PowerShell :

- Connectez-vous pour le développement AWS local à l'aide des informations d'identification de la console
- Connectez-vous à l'aide des informations d'identification de console indiquées dans le guide de Outils AWS pour PowerShell l'utilisateur

Utilisation d'IAM Identity Center pour authentifier le AWS SDK et les outils

AWS IAM Identity Center peut être utilisé pour fournir des AWS informations d'identification lors du développement d'une AWS application dans un environnement de service non AWS informatique. Si vous développez sur une AWS ressource, telle qu'Amazon Elastic Compute Cloud (Amazon EC2), nous vous recommandons plutôt d'obtenir AWS Cloud9 des informations d'identification auprès de ce service.

Utilisez l'authentification IAM Identity Center si vous utilisez déjà Identity Center pour accéder à un AWS compte ou si vous devez gérer l'accès pour une organisation.

Dans ce didacticiel, vous établissez l'accès à IAM Identity Center et vous le configurez pour votre SDK ou outil à l'aide du portail AWS d'accès et du. AWS CLI

- Le portail AWS d'accès est l'emplacement Web où vous vous connectez manuellement à l'IAM Identity Center. Le format de l'URL est `d-xxxxxxxxxx.awsapps.com/start` ou*`your_subdomain`*`.awsapps.com/start`. Lorsque vous êtes connecté au portail AWS d'accès, vous pouvez consulter Comptes AWS les rôles qui ont été configurés pour cet utilisateur. Cette procédure utilise le portail AWS d'accès pour obtenir les valeurs de configuration dont vous avez besoin pour le processus SDK/tool d'authentification.
- AWS CLI Il est utilisé pour configurer votre SDK ou votre outil afin d'utiliser l'authentification IAM Identity Center pour les appels d'API effectués par votre code. Ce processus unique met à jour votre AWS `config` fichier partagé, qui est ensuite utilisé par votre SDK ou votre outil lorsque vous exécutez votre code.

Conditions préalables

Avant de commencer cette procédure, vous devez avoir effectué les opérations suivantes :

- Si vous n'en avez pas Compte AWS, inscrivez-vous pour un Compte AWS.

- Si vous n'avez pas encore activé IAM Identity Center, activez IAM Identity Center en suivant les instructions du guide de l'AWS IAM Identity Center utilisateur.

Configuration de l'accès programmatique à l'aide d'IAM Identity Center

Étape 1 : établir l'accès et sélectionner l'ensemble d'autorisations approprié

Choisissez l'une des méthodes suivantes pour accéder à vos AWS informations d'identification.

Je ne dispose pas d'un accès établi via IAM Identity Center

1. Ajoutez un utilisateur et ajoutez des autorisations administratives en suivant la procédure de configuration de l'accès utilisateur avec le répertoire IAM Identity Center par défaut du guide de l'AWS IAM Identity Center utilisateur.
2. L'ensemble `AdministratorAccess` d'autorisations ne doit pas être utilisé pour le développement normal. Nous vous recommandons plutôt d'utiliser l'ensemble d'`PowerUserAccess`autorisations prédéfini, sauf si votre employeur a créé un ensemble d'autorisations personnalisé à cette fin.

   Suivez à nouveau la même procédure de configuration de l'accès utilisateur avec la procédure d'annuaire par défaut d'IAM Identity Center, mais cette fois :

   - Au lieu de créer le *`Admin team`* groupe, créez-en un *`Dev team`* et remplacez-le par la suite dans les instructions.
   - Vous pouvez utiliser l'utilisateur existant, mais celui-ci doit être ajouté au nouveau *`Dev team`* groupe.
   - Au lieu de créer l'ensemble d'*`AdministratorAccess`*autorisations, *`PowerUserAccess`* créez-en un et remplacez-le par la suite dans les instructions.

   Lorsque vous aurez terminé, vous devriez avoir les éléments suivants :

   - Un `Dev team` groupe.
   - Une `PowerUserAccess` autorisation attachée définie pour le `Dev team` groupe.
   - Votre utilisateur a été ajouté au `Dev team` groupe.
3. Quittez le portail et reconnectez-vous pour voir vos options Comptes AWS et celles pour `Administrator` ou`PowerUserAccess`. Sélectionnez `PowerUserAccess` lorsque vous travaillez avec votre outil/SDK.

J'y ai déjà accès AWS via un fournisseur d'identité fédéré géré par mon employeur (tel que Microsoft Entra ou Okta)

Connectez-vous AWS via le portail de votre fournisseur d'identité. Si votre administrateur cloud vous a accordé des autorisations `PowerUserAccess` (de développeur), vous voyez Comptes AWS celles auxquelles vous avez accès et votre ensemble d'autorisations. En regard du nom de votre jeu d'autorisations, vous pouvez voir des options permettant d'accéder aux comptes manuellement ou par programmation à l'aide de ce jeu d'autorisations.

Les implémentations personnalisées peuvent entraîner des expériences différentes, telles que des noms de jeux d'autorisations différents. Si vous avez des doutes sur le jeu d'autorisations à utiliser, contactez votre équipe informatique pour obtenir de l'aide.

J'y ai déjà accès AWS via le portail AWS d'accès géré par mon employeur

Connectez-vous AWS via le portail AWS d'accès. Si votre administrateur cloud vous a accordé des autorisations `PowerUserAccess` (développeur), vous pouvez voir les Comptes AWS auxquels vous avez accès et votre jeu d'autorisations. En regard du nom de votre jeu d'autorisations, vous pouvez voir des options permettant d'accéder aux comptes manuellement ou par programmation à l'aide de ce jeu d'autorisations.

J'y ai déjà accès AWS via un fournisseur d'identité personnalisé fédéré géré par mon employeur

Contactez votre équipe informatique pour obtenir de l'aide.

Étape 2 : Configuration SDKs et outils pour utiliser IAM Identity Center

1. Sur votre machine de développement, installez la dernière version AWS CLI.

    a. Consultez la section Installation ou mise à jour de la dernière version du AWS CLI dans le guide de AWS Command Line Interface l'utilisateur.

    b. (Facultatif) Pour vérifier que le AWS CLI fonctionne, ouvrez une invite de commande et exécutez la `aws --version` commande.

2. Connectez-vous au portail d' AWS accès. Votre employeur peut fournir cette URL ou vous pouvez l'obtenir par e-mail après l'étape 1 : Établir l'accès. Si ce n'est pas le cas, trouvez l'URL de votre portail d'AWS accès sur le tableau de bord de https://console.aws.amazon.com/singlesignon/.

    a. Dans le portail AWS d'accès, dans l'onglet Comptes, sélectionnez le compte individuel à gérer. Les rôles de votre utilisateur sont affichés. Choisissez les clés d'accès pour obtenir

les informations d'identification pour la ligne de commande ou l'accès programmatique pour l'ensemble d'autorisations approprié. Utilisez l'ensemble `PowerUserAccess` d'autorisations prédéfini, ou le jeu d'autorisations que vous ou votre employeur avez créé pour appliquer les autorisations du moindre privilège au développement.

b. Dans la boîte de dialogue Obtenir les informations d'identification, choisissez macOS et Linux ou Windows, selon votre système d'exploitation.

c. Choisissez la méthode d'identification IAM Identity Center pour obtenir les `SSO Region` valeurs `Issuer URL` et dont vous avez besoin pour l'étape suivante. Remarque : `SSO Start URL` peut être utilisé de manière interchangeable avec`Issuer URL`.

3. Dans l'invite de AWS CLI commande, exécutez la `aws configure sso` commande. Lorsque vous y êtes invité, entrez les valeurs de configuration que vous avez collectées à l'étape précédente. Pour plus de détails sur cette AWS CLI commande, voir Configurer votre profil avec l'`aws configure sso`assistant.

   a. Pour l'invite`SSO Start URL`, entrez la valeur que vous avez obtenue pour`Issuer URL`.

   b. Pour le nom du profil CLI, nous vous recommandons de le saisir *`default`* lorsque vous commencez. Pour plus d'informations sur la façon de définir des profils (nommés) autres que ceux par défaut et leur variable d'environnement associée, consultezProfils.

4. (Facultatif) Dans l'invite de AWS CLI commande, confirmez l'identité de la session active en exécutant la `aws sts get-caller-identity` commande. La réponse doit indiquer l'ensemble d'autorisations IAM Identity Center que vous avez configuré.

5. Si vous utilisez un AWS SDK, créez une application pour celui-ci dans votre environnement de développement.

   a. Pour certains SDKs, des packages supplémentaires tels que `SSO` et `SSOOIDC` doivent être ajoutés à votre application avant de pouvoir utiliser l'authentification IAM Identity Center. Pour plus de détails, consultez votre SDK spécifique.

   b. Si vous avez déjà configuré l'accès à AWS, passez en revue votre AWS `credentials` fichier partagé pour en détecter d'éventuelsAWS clés d'accès. Vous devez supprimer toutes les informations d'identification statiques avant que le SDK ou l'outil n'utilise les informations d'identification du centre d'identité IAM en raison de la Comprendre la chaîne des fournisseurs d'informations d'identification priorité.

Pour en savoir plus sur la façon dont les outils SDKs et utilisent et actualisent les informations d'identification à l'aide de cette configuration, consultezComment l'authentification IAM Identity Center est-elle résolue AWS SDKs et quels outils ?.

Pour configurer les paramètres du fournisseur IAM Identity Center directement dans le `config` fichier partagé, consultez Fournisseur d'identifiants IAM Identity Center ce guide.

Actualisation des sessions d'accès au portail

Votre accès finira par expirer et le SDK ou l'outil rencontrera une erreur d'authentification. La date d'expiration dépend de la durée de session que vous avez configurée. Pour actualiser à nouveau la session du portail d'accès en cas de besoin, utilisez la commande AWS CLI pour exécuter la `aws sso login` commande.

Vous pouvez prolonger à la fois la durée de session du portail d'accès IAM Identity Center et celle de la session d'ensemble d'autorisations. Cela allonge le délai pendant lequel vous pouvez exécuter le code avant de devoir vous reconnecter manuellement avec le AWS CLI. Pour plus d'informations, consultez les rubriques suivantes dans le AWS IAM Identity Center Guide de l'utilisateur :

- Durée de session IAM Identity Center : configurez la durée des sessions du portail d' AWS accès de vos utilisateurs
- Autoriser définir la durée de la session — Définir la durée de la session

Comment l'authentification IAM Identity Center est-elle résolue AWS SDKs et quels outils ?

Termes pertinents du centre d'identité IAM

Les termes suivants vous aident à comprendre le processus et la configuration sous-jacents AWS IAM Identity Center. La documentation du AWS SDK APIs utilise des noms différents de ceux d'IAM Identity Center pour certains de ces concepts d'authentification. Il est utile de connaître les deux noms.

Le tableau suivant montre comment les noms alternatifs sont liés les uns aux autres.

Nom du centre d'identité IAM	Nom de l'API du SDK	Description
Identity Center	`sso`	Bien que AWS Single Sign-On soit renommé, les espaces de noms de l'`sso`API conserver ont leur nom d'origine à des fins de rétrocompatibilité. Pour plus d'informations, consultez IAM Identity Center rename dans le Guide de l'utilisateur AWS IAM Identity Center .
Console IAM Identity Center Console d'administration		La console que vous utilisez pour configurer l'authent ification unique.
AWS URL du portail d'accès		Une URL propre à votre compte IAM Identity Center, par exemple`https://xxx.awsapps.com/start` . Vous vous connectez à ce portail à l'aide de vos identifiants de connexion IAM Identity Center.
Session sur le portail d'accès à l'IAM Identity Center	Session d'authentification	Fournit un jeton d'accès au porteur à l'appelant.
Session d'ensemble d'autoris ations		La session IAM que le SDK utilise en interne pour effectuer les Service AWS appels. Dans les discussions informelles, il est possible que cette session soit incorrect ement appelée « session de rôle ».

Nom du centre d'identité IAM	Nom de l'API du SDK	Description
Informations d'identification du jeu d'autorisations	AWS informations d'identification informations d'identification sigv4	Les informations d'identification que le SDK utilise réellement pour la plupart des Service AWS appels (en particulier, tous les Service AWS appels sigv4). Dans les discussions informelles, il est possible que cela soit appelé à tort « informations d'identification du rôle ».
Fournisseur d'identifiants IAM Identity Center	Fournisseur d'informations d'identification SSO	Comment obtenez-vous les informations d'identification, telles que la classe ou le module fournissant les fonctionnalités.

Comprendre la résolution des informations d'identification du SDK pour Services AWS

L'API IAM Identity Center échange les informations d'identification du jeton porteur contre des informations d'identification sigv4. La plupart Services AWS sont des sigv4 APIs, à quelques exceptions près comme Amazon CodeWhisperer et Amazon CodeCatalyst. Ce qui suit décrit le processus de résolution des informations d'identification permettant de prendre en charge la plupart des Service AWS appels visant à obtenir le code de votre application. AWS IAM Identity Center

Démarrer une session sur le portail AWS d'accès

- Commencez le processus en vous connectant à la session à l'aide de vos informations d'identification.
  - Utilisez la `aws sso login` commande dans le AWS Command Line Interface (AWS CLI). Cela démarre une nouvelle session IAM Identity Center si vous n'avez pas encore de session active.
- Lorsque vous démarrez une nouvelle session, vous recevez un jeton d'actualisation et un jeton d'accès de la part d'IAM Identity Center. AWS CLI Il met également à jour un fichier JSON de cache SSO avec un nouveau jeton d'accès et un nouveau jeton d'actualisation et le rend disponible pour utilisation par SDKs.

- Si vous avez déjà une session active, la AWS CLI commande réutilise la session existante et expirera chaque fois que la session existante expirera. Pour savoir comment définir la durée d'une session IAM Identity Center, voir Configurer la durée des sessions du portail d' AWS accès de vos utilisateurs dans le Guide de l'AWS IAM Identity Center utilisateur.
  - La durée maximale des sessions a été étendue à 90 jours afin de réduire le besoin de connexions fréquentes.

Comment le SDK obtient les informations d'identification pour les appels Service AWS

SDKs fournir un accès Services AWS lorsque vous instanciez un objet client par service. Lorsque le profil sélectionné du AWS `config` fichier partagé est configuré pour la résolution des informations d'identification IAM Identity Center, IAM Identity Center est utilisé pour résoudre les informations d'identification de votre application.

- Le processus de résolution des informations d'identification est terminé pendant l'exécution lorsqu'un client est créé.

Pour récupérer les informations d'identification pour sigv4 à APIs l'aide de l'authentification unique IAM Identity Center, le SDK utilise le jeton d'accès IAM Identity Center pour obtenir une session IAM. Cette session IAM est appelée session d'ensemble d'autorisations et permet d' AWS accéder au SDK en assumant un rôle IAM.

- La durée de session de l'ensemble d'autorisations est définie indépendamment de celle de la session IAM Identity Center.
  - Pour savoir comment définir la durée de session définie par les autorisations, voir Définir la durée de session dans le guide de AWS IAM Identity Center l'utilisateur.
- Sachez que les informations d'identification de l'ensemble d'autorisations sont également appelées informations AWS d'identification et informations d'identification sigv4 dans la plupart des documentations d'API du AWS SDK.

Les informations d'identification de l'ensemble d'autorisations sont renvoyées par un appel getRoleCredentialsde l'API IAM Identity Center au SDK. L'objet client du SDK utilise ce rôle IAM supposé pour effectuer des appels Service AWS, par exemple pour demander à Amazon S3 de répertorier les buckets de votre compte. L'objet client peut continuer à fonctionner en utilisant ces informations d'identification du jeu d'autorisations jusqu'à l'expiration de la session du jeu d'autorisations.

Expiration et actualisation de la session

Lorsque vous utilisez leConfiguration du fournisseur de jetons SSO, le jeton d'accès horaire obtenu auprès d'IAM Identity Center est automatiquement actualisé à l'aide du jeton d'actualisation.

- Si le jeton d'accès a expiré lorsque le SDK essaie de l'utiliser, le SDK utilise le jeton d'actualisation pour essayer d'obtenir un nouveau jeton d'accès. L'IAM Identity Center compare le jeton d'actualisation à la durée de la session de votre portail d'accès à l'IAM Identity Center. Si le jeton d'actualisation n'est pas expiré, le centre d'identité IAM répond avec un autre jeton d'accès.
- Ce jeton d'accès peut être utilisé soit pour actualiser la session d'ensemble d'autorisations des clients existants, soit pour résoudre les informations d'identification des nouveaux clients.

Toutefois, si la session du portail d'accès à l'IAM Identity Center est expirée, aucun nouveau jeton d'accès n'est accordé. Par conséquent, la durée définie d'autorisations ne peut pas être renouvelée. Il expirera (et l'accès sera perdu) chaque fois que la durée de session définie par le cache expirera pour les clients existants.

Tout code qui crée un nouveau client échouera à l'authentification dès l'expiration de la session IAM Identity Center. Cela est dû au fait que les informations d'identification de l'ensemble d'autorisations ne sont pas mises en cache. Votre code ne sera pas en mesure de créer un nouveau client et de terminer le processus de résolution des informations d'identification tant que vous ne disposerez pas d'un jeton d'accès valide.

Pour récapituler, lorsque le SDK a besoin de nouvelles informations d'identification d'un ensemble d'autorisations, le SDK vérifie d'abord les informations d'identification existantes valides et les utilise. Cela s'applique qu'il s'agisse d'un nouveau client ou d'un client existant dont les informations d'identification ont expiré. Si les informations d'identification ne sont pas trouvées ou si elles ne sont pas valides, le SDK appelle l'API IAM Identity Center pour obtenir de nouvelles informations d'identification. Pour appeler l'API, elle a besoin du jeton d'accès. Si le jeton d'accès est expiré, le SDK utilise le jeton d'actualisation pour essayer d'obtenir un nouveau jeton d'accès auprès du service IAM Identity Center. Ce jeton est accordé si votre session du portail d'accès à l'IAM Identity Center n'a pas expiré.

Utilisation d'IAM Roles Anywhere pour l'authentification et les outils AWS SDKs

Vous pouvez utiliser IAM Roles Anywhere pour obtenir des informations d'identification de sécurité temporaires dans IAM pour les charges de travail telles que les serveurs, les conteneurs et les applications qui s'exécutent en dehors de. AWS Pour utiliser IAM Roles Anywhere, vos charges de travail doivent utiliser des certificats X.509. Votre administrateur cloud doit fournir le certificat et la clé privée nécessaires pour configurer IAM Roles Anywhere en tant que fournisseur d'informations d'identification.

Étape 1 : configurer les rôles IAM n'importe où

IAM Roles Anywhere permet d'obtenir des informations d'identification temporaires pour une charge de travail ou un processus qui s'exécute en dehors de AWS. Une ancre de confiance est établie avec l'autorité de certification afin d'obtenir des informations d'identification temporaires pour le rôle IAM associé. Le rôle définit les autorisations dont bénéficiera votre charge de travail lorsque votre code s'authentifie auprès d'IAM Roles Anywhere.

Pour connaître les étapes de configuration de l'ancre de confiance, du rôle IAM et du profil IAM Roles Anywhere, consultez la section Création d'une ancre de confiance et d'un profil dans Gestion des identités et des accès AWS Roles Anywhere du guide de l'utilisateur d'IAM Roles Anywhere.

> **Note**
>
> Un profil figurant dans le guide de l'utilisateur d'IAM Roles Anywhere fait référence à un concept unique au sein du service IAM Roles Anywhere. Cela n'est pas lié aux profils contenus dans le AWS `config` fichier partagé.

Étape 2 : Utiliser les rôles IAM n'importe où

Pour obtenir des informations d'identification de sécurité temporaires auprès d'IAM Roles Anywhere, utilisez l'outil d'aide aux informations d'identification fourni par IAM Roles Anywhere. L'outil d'identification met en œuvre le processus de signature pour IAM Roles Anywhere.

Pour obtenir des instructions sur le téléchargement de l'outil d'aide aux informations d'identification, consultez la section Obtention d'informations d'identification de sécurité temporaires auprès de

Gestion des identités et des accès AWS Roles Anywhere dans le guide de l'utilisateur d'IAM Roles Anywhere.

Pour utiliser les informations d'identification de sécurité temporaires d'IAM Roles Anywhere avec AWS SDKs et le AWS CLI, vous pouvez configurer le `credential_process` paramètre dans le AWS `config` fichier partagé. Le AWS CLI support SDKs et un fournisseur d'informations d'identification de processus utilisé `credential_process` pour s'authentifier. Ce qui suit montre la structure générale à définir`credential_process`.

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

La `credential-process` commande de l'outil d'assistance renvoie des informations d'identification temporaires dans un format JSON standard compatible avec le `credential_process` paramètre. Notez que le nom de la commande contient un trait d'union mais que le nom du paramètre contient un trait de soulignement. La commande nécessite les paramètres suivants :

- `private-key`— Le chemin d'accès à la clé privée qui a signé la demande.
- `certificate`— Le chemin d'accès au certificat.
- `role-arn`— L'ARN du rôle pour lequel vous souhaitez obtenir des informations d'identification temporaires.
- `profile-arn`— L'ARN du profil qui fournit un mappage pour le rôle spécifié.
- `trust-anchor-arn`— L'ARN de l'ancre de confiance utilisée pour l'authentification.

Votre administrateur cloud doit fournir le certificat et la clé privée. Les trois valeurs ARN peuvent être copiées à partir du AWS Management Console. L'exemple suivant montre un `config` fichier partagé qui configure la récupération d'informations d'identification temporaires à partir de l'outil d'assistance.

```
[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
```

Pour les paramètres facultatifs et des informations supplémentaires sur les outils d'assistance, voir IAM Roles Anywhere Credential Helper on. GitHub

Pour plus de détails sur le paramètre de configuration du SDK lui-même et sur le fournisseur d'informations d'identification du processus, consultez ce Fournisseur d'identifiants de processus guide.

Assumer un rôle avec des AWS informations d'identification pour l'authentification AWS SDKs et des outils

Assumer un rôle implique l'utilisation d'un ensemble d'informations d'identification de sécurité temporaires pour accéder à AWS des ressources auxquelles vous n'auriez peut-être pas accès autrement. Ces informations d'identification temporaires incluent un ID de clé d'accès, une clé d'accès secrète et un jeton de sécurité. Pour en savoir plus sur les demandes d'API AWS Security Token Service (AWS STS), consultez la section Actions de la référence AWS Security Token Service d'API.

Pour configurer votre SDK ou votre outil afin qu'il assume un rôle, vous devez d'abord créer ou identifier un rôle spécifique à assumer. Les rôles IAM sont identifiés de manière unique par un rôle Amazon Resource Name (ARN). Les rôles établissent des relations de confiance avec une autre entité. L'entité de confiance qui utilise le rôle peut être une Service AWS ou une autre Compte AWS. Pour en savoir plus sur les rôles IAM, consultez la section Utilisation des rôles IAM dans le Guide de l'utilisateur IAM.

Une fois le rôle IAM identifié, si ce rôle vous fait confiance, vous pouvez configurer votre SDK ou votre outil pour utiliser les autorisations accordées par le rôle.

> **Note**
>
> Il est recommandé d' AWS utiliser des points de terminaison régionaux dans la mesure du possible et de configurer votreRégion AWS.

Assumez un rôle IAM

Lorsque vous assumez un rôle, AWS STS renvoie un ensemble d'informations d'identification de sécurité temporaires. Ces informations d'identification proviennent d'un autre profil ou de l'instance ou du conteneur dans lequel votre code est exécuté. Le plus souvent, ce type d'attribution de rôle est utilisé lorsque vous possédez les AWS informations d'identification d'un compte, mais que votre application doit accéder aux ressources d'un autre compte.

Étape 1 : configurer un rôle IAM

Pour configurer votre SDK ou votre outil afin qu'il assume un rôle, vous devez d'abord créer ou identifier un rôle spécifique à assumer. Les rôles IAM sont identifiés de manière unique à l'aide d'un ARN de rôle. Les rôles établissent des relations de confiance avec une autre entité, généralement au sein de votre compte ou pour un accès entre comptes. Pour configurer cela, consultez la section Création de rôles IAM dans le guide de l'utilisateur IAM.

Étape 2 : Configuration du SDK ou de l'outil

Configurez le SDK ou l'outil pour obtenir des informations d'identification auprès de `credential_source` ou`source_profile`.

`credential_source`À utiliser pour obtenir des informations d'identification à partir d'un conteneur Amazon ECS, d'une instance Amazon EC2 ou de variables d'environnement.

`source_profile`À utiliser pour obtenir des informations d'identification à partir d'un autre profil. `source_profile`prend également en charge le chaînage des rôles, qui consiste en des hiérarchies de profils dans lesquelles un rôle assumé est ensuite utilisé pour assumer un autre rôle.

Lorsque vous le spécifiez dans un profil, le SDK ou l'outil lance automatiquement l'appel d' AWS STS `AssumeRole`API correspondant pour vous. Pour récupérer et utiliser des informations d'identification temporaires en assumant un rôle, spécifiez les valeurs de configuration suivantes dans le AWS `config` fichier partagé. Pour plus de détails sur chacun de ces paramètres, consultez la Paramètres du fournisseur d'informations d'identification du rôle section.

- `role_arn`- À partir du rôle IAM que vous avez créé à l'étape 1
- Configurez `credential_source` soit `source_profile`
- (Facultatif) `duration_seconds`
- (Facultatif) `external_id`
- (Facultatif) `mfa_serial`
- (Facultatif) `role_session_name`

Les exemples suivants montrent la configuration des deux options d'attribution de rôle dans un `config` fichier partagé :

```
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
```

Pour plus de détails sur tous les paramètres du fournisseur d'informations d'identification d'assumer un rôle, consultez Assumer le rôle de fournisseur d'informations d'identification ce guide.

Assumer un rôle avec l'identité Web ou OpenID Connect pour l'authentification et les outils AWS SDKs

Assumer un rôle implique l'utilisation d'un ensemble d'informations d'identification de sécurité temporaires pour accéder à AWS des ressources auxquelles vous n'auriez peut-être pas accès autrement. Ces informations d'identification temporaires incluent un ID de clé d'accès, une clé d'accès secrète et un jeton de sécurité. Pour en savoir plus sur les demandes d'API AWS Security Token Service (AWS STS), consultez la section Actions de la référence AWS Security Token Service d'API.

Pour configurer votre SDK ou votre outil afin qu'il assume un rôle, vous devez d'abord créer ou identifier un rôle spécifique à assumer. Les rôles IAM sont identifiés de manière unique par un rôle Amazon Resource Name (ARN). Les rôles établissent des relations de confiance avec une autre entité. L'entité de confiance qui utilise le rôle peut être un fournisseur d'identité Web, OpenID Connect (OIDC) ou une fédération SAML. Pour en savoir plus sur les rôles IAM, consultez la section Méthodes pour assumer un rôle dans le Guide de l'utilisateur IAM.

Une fois le rôle IAM configuré dans votre SDK, s'il est configuré pour faire confiance à votre fournisseur d'identité, vous pouvez configurer davantage votre SDK pour qu'il assume ce rôle afin d'obtenir des informations d'identification temporaires. AWS

> **Note**
>
> Il est recommandé d' AWS utiliser des points de terminaison régionaux dans la mesure du possible et de configurer votreRégion AWS.

Fédérez avec l'identité Web ou OpenID Connect

Vous pouvez utiliser les jetons Web JSON (JWTs) provenant de fournisseurs d'identité publics, tels que Login With Amazon, Facebook, Google pour obtenir des AWS informations d'identification temporairesAssumeRoleWithWebIdentity. Selon la manière dont ils sont utilisés, ils JWTs peuvent être appelés jetons d'identification ou jetons d'accès. Vous pouvez également utiliser des JWTs documents émis par des fournisseurs d'identité (IdPs) compatibles avec le protocole de découverte de l'OIDC, tels que EntraId ou PingFederate.

Si vous utilisez Amazon Elastic Kubernetes Service, cette fonctionnalité permet de spécifier différents rôles IAM pour chacun de vos comptes de service dans un cluster Amazon EKS. Cette fonctionnalité de Kubernetes est distribuée JWTs à vos pods, qui sont ensuite utilisés par ce fournisseur d'informations d'identification pour obtenir des informations d'identification temporaires. AWS Pour plus d'informations sur cette configuration Amazon EKS, consultez la section Rôles IAM pour les comptes de service dans le guide de l'utilisateur Amazon EKS. Toutefois, pour une option plus simple, nous vous recommandons d'utiliser Amazon EKS Pod Identities à la place si votre SDK le prend en charge.

Étape 1 : configurer un fournisseur d'identité et un rôle IAM

Pour configurer la fédération avec un IdP externe, utilisez un fournisseur d'identité IAM pour fournir des AWS informations sur l'IdP externe et sa configuration. Cela établit la confiance entre votre Compte AWS IdP et l'IdP externe. Avant de configurer le SDK pour utiliser le jeton Web JSON (JWT) pour l'authentification, vous devez d'abord configurer le fournisseur d'identité (IdP) et le rôle IAM utilisé pour y accéder. Pour les configurer, consultez la section Création d'un rôle pour l'identité Web ou OpenID Connect Federation (console) dans le guide de l'utilisateur IAM.

Étape 2 : Configuration du SDK ou de l'outil

Configurez le SDK ou l'outil pour utiliser un jeton Web JSON (JWT) à des AWS STS fins d'authentification.

Lorsque vous le spécifiez dans un profil, le SDK ou l'outil lance automatiquement l'appel d' AWS STS `AssumeRoleWithWebIdentity`API correspondant pour vous. Pour récupérer et utiliser des informations d'identification temporaires à l'aide de la fédération d'identité Web, spécifiez les valeurs de configuration suivantes dans le AWS `config` fichier partagé. Pour plus de détails sur chacun de ces paramètres, consultez la Paramètres du fournisseur d'informations d'identification du rôle section.

- `role_arn`- À partir du rôle IAM que vous avez créé à l'étape 1
- `web_identity_token_file`- Depuis l'IdP externe
- (Facultatif) `duration_seconds`
- (Facultatif) `role_session_name`

Voici un exemple de configuration de `config` fichier partagé pour assumer un rôle avec une identité Web :

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

> **Note**
>
> Pour les applications mobiles, pensez à utiliser Amazon Cognito. Amazon Cognito agit en tant que courtier d'identité et effectue une grande partie du travail de fédération à votre place. Cependant, le fournisseur d'identité Amazon Cognito n'est pas inclus dans les bibliothèques principales de SDKs and tools comme les autres fournisseurs d'identité. Pour accéder à l'API Amazon Cognito, incluez le client du service Amazon Cognito dans la version ou les bibliothèques de votre SDK ou outil. Pour une utilisation avec AWS SDKs, consultez les exemples de code dans le manuel Amazon Cognito Developer Guide.

Pour plus de détails sur tous les paramètres du fournisseur d'informations d'identification d'assumer un rôle, consultez Assumer le rôle de fournisseur d'informations d'identification ce guide.

Utilisation de clés AWS d'accès pour l'authentification AWS SDKs et d'outils

L'utilisation de clés d' AWS accès est une option d'authentification lors de l'utilisation AWS SDKs d'outils.

Utiliser des identifiants à court terme

Nous vous recommandons de configurer votre SDK ou outil à utiliser Utilisation d'IAM Identity Center pour authentifier le AWS SDK et les outils pour utiliser les options de durée de session prolongée.

Toutefois, pour configurer directement les informations d'identification temporaires du SDK ou de l'outil, consultezUtilisation d'informations d'identification à court terme pour l'authentification AWS SDKs et d'outils.

Utilisez des identifiants à long terme

> **Warning**
>
> Afin d'éviter les risques de sécurité, n'employez pas les utilisateurs IAM pour l'authentification lorsque vous développez des logiciels spécialisés ou lorsque vous travaillez avec des données réelles. Préférez la fédération avec un fournisseur d'identité tel que AWS IAM Identity Center.

Gérez l'accès à travers Comptes AWS

En tant que bonne pratique en matière de sécurité, nous vous recommandons AWS Organizations d'utiliser IAM Identity Center pour gérer l'accès de tous vos Comptes AWS utilisateurs. Pour plus d'informations, consultez Bonnes pratiques de sécurité dans IAM dans le Guide de l'utilisateur IAM.

Vous pouvez créer des utilisateurs dans IAM Identity Center, utiliser Microsoft Active Directory, utiliser un fournisseur d'identité (IdP) SAML 2.0 ou fédérer individuellement votre IdP avec. Comptes AWS En utilisant l'une de ces approches, vous pouvez proposer une expérience d'authentification unique à vos utilisateurs. Vous pouvez également appliquer l'authentification multifactorielle (MFA) et utiliser des informations d'identification Compte AWS temporaires pour l'accès. Cela diffère d'un utilisateur IAM, qui est un identifiant à long terme qui peut être partagé et qui peut augmenter le risque de sécurité pour vos AWS ressources.

Création d'utilisateurs IAM pour les environnements sandbox uniquement

Si vous débutez dans ce domaine AWS, vous pouvez créer un utilisateur IAM de test, puis l'utiliser pour exécuter des didacticiels et découvrir ce qui AWS a à offrir. Vous pouvez utiliser ce type d'identifiant lorsque vous apprenez, mais nous vous recommandons d'éviter de l'utiliser en dehors d'un environnement sandbox.

Pour les cas d'utilisation suivants, il peut être judicieux de commencer avec les utilisateurs IAM dans AWS :

- Démarrage avec votre AWS SDK ou outil et exploration Services AWS dans un environnement sandbox.
- Exécution de scripts, de tâches et d'autres processus automatisés planifiés qui ne prennent pas en charge un processus de connexion assisté par un humain dans le cadre de votre apprentissage.

Si vous utilisez des utilisateurs IAM en dehors de ces cas d'utilisation, passez à IAM Identity Center ou fédérez votre fournisseur d'identité Comptes AWS dès que possible. Pour plus d'informations, consultez la section Fédération des identités dans AWS.

Clés d'accès utilisateur IAM sécurisées

Vous devez régulièrement alterner les clés d'accès utilisateur IAM. Suivez les instructions de la section Rotation des touches d'accès du guide de l'utilisateur IAM. Si vous pensez avoir accidentellement partagé vos clés d'accès utilisateur IAM, faites pivoter vos clés d'accès.

Les clés d'accès utilisateur IAM doivent être stockées dans le AWS `credentials` fichier partagé sur la machine locale. Ne stockez pas les clés d'accès utilisateur IAM dans votre code. N'incluez aucun fichier de configuration contenant vos clés d'accès utilisateur IAM dans un logiciel de gestion de code source. Des outils externes, tels que le projet open source git-secrets, peuvent vous aider à éviter de transférer par inadvertance des informations sensibles dans un dépôt Git. Pour plus d’informations, consultez Identités IAM (utilisateurs, groupes et rôles) dans le Guide de l’utilisateur IAM.

Pour configurer un utilisateur IAM pour qu'il démarre, voirUtilisation d'informations d'identification à long terme pour l'authentification AWS SDKs et d'outils.

Utilisation d'informations d'identification à court terme pour l'authentification AWS SDKs et d'outils

Nous vous recommandons de configurer votre AWS SDK ou votre outil pour l'utiliser Utilisation d'IAM Identity Center pour authentifier le AWS SDK et les outils avec des options de durée de session prolongée. Toutefois, vous pouvez copier et utiliser les informations d'identification temporaires disponibles sur le portail AWS d'accès. De nouvelles informations d’identification devront être copiées à leur expiration. Vous pouvez utiliser les informations d’identification temporaires dans un profil ou les utiliser comme valeurs pour les propriétés système et les variables d’environnement.

Bonne pratique : au lieu de gérer manuellement les clés d'accès et un jeton dans le fichier d'informations d'identification, nous recommandons que votre application utilise des informations d'identification temporaires fournies par :

- Un service de AWS calcul, tel que l'exécution de votre application sur Amazon Elastic Compute Cloud ou dans AWS Lambda.
- Une autre option de la chaîne des fournisseurs d'informations d'identification, telle queUtilisation d'IAM Identity Center pour authentifier le AWS SDK et les outils.
- Vous pouvez également utiliser le Fournisseur d'identifiants de processus pour récupérer des informations d'identification temporaires.

Configuration d'un fichier d'informations d'identification à l'aide d'informations d'identification à court terme extraites du portail AWS d'accès

1. Créez un fichier d'informations d'identification partagé.
2. Dans le fichier d'informations d'identification, collez le texte d'espace réservé suivant jusqu'à ce que vous y colliez des informations d'identification temporaires fonctionnelles.

```
[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. Enregistrez le fichier. Le fichier `~/.aws/credentials` devrait maintenant exister sur votre système de développement local. Ce fichier contient le profil [par défaut] utilisé par le SDK ou l'outil si aucun profil nommé spécifique n'est spécifié.
4. Connectez-vous au portail d' AWS accès.

5. Suivez ces instructions pour l'actualisation manuelle des informations d'identification afin de copier les informations d'identification du rôle IAM depuis le portail d' AWS accès.

   a. Pour l'étape 4 des instructions liées, choisissez le nom du rôle IAM qui accorde l'accès pour vos besoins de développement. Ce rôle porte généralement un nom tel que PowerUserAccess« Développeur ».

   b. Pour l'étape 7 des instructions liées, sélectionnez l'option Ajouter manuellement un profil à votre fichier AWS d'informations d'identification et copiez le contenu.

6. Collez les informations d'identification copiées dans votre `credentials` fichier local. Le nom de profil généré n'est pas nécessaire si vous utilisez le `default` profil. Votre fichier doit ressembler à ce qui suit.

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

7. Enregistrez le fichier `credentials`.

Lorsque le kit SDK crée un client de service, il accède à ces informations d'identification temporaires et les utilise pour chaque demande. Les paramètres du rôle IAM choisi à l'étape 5a déterminent la durée de validité des informations d'identification temporaires. La durée maximale est de douze heures.

Une fois les informations d'identification temporaires expirées, répétez les étapes 4 à 7.

Utilisation d'informations d'identification à long terme pour l'authentification AWS SDKs et d'outils

> **Warning**
>
> Afin d'éviter les risques de sécurité, n'employez pas les utilisateurs IAM pour l'authentification lorsque vous développez des logiciels spécialisés ou lorsque vous travaillez avec des données réelles. Préférez la fédération avec un fournisseur d'identité tel que AWS IAM Identity Center.

Si vous utilisez un utilisateur IAM pour exécuter votre code, le SDK ou l'outil de votre environnement de développement s'authentifie en utilisant les informations d'identification utilisateur IAM à long terme dans le fichier partagé. AWS `credentials` Consultez la rubrique Bonnes pratiques de sécurité dans IAM et passez à IAM Identity Center ou à d'autres informations d'identification temporaires dès que possible.

Avertissements et conseils importants concernant les informations d'identification

Avertissements concernant les informations d'identification

- N'utilisez PAS les informations d'identification root de votre compte pour accéder aux ressources AWS . Ces informations d’identification offrent un accès illimité au compte et sont difficiles à révoquer.
- N’incluez PAS de clés d’accès littérales ou d’informations d’identification dans vos fichiers d’application. Vous risqueriez en effet d’exposer accidentellement vos informations d’identification si, par exemple, vous chargiez le projet sur un référentiel public.
- N’incluez PAS de fichiers contenant des informations d’identification dans votre zone de projet.
- Sachez que toutes les informations d'identification stockées dans le AWS `credentials` fichier partagé sont stockées en texte brut.

Conseils supplémentaires pour gérer les informations d'identification en toute sécurité

Pour une discussion générale sur la manière de gérer les AWS informations d'identification en toute sécurité, consultez la section Meilleures pratiques pour la gestion des clés AWS d'accès dans le Références générales AWS. En plus de cette discussion, envisagez de prendre les mesures suivantes :

- Utilisez des rôles IAM pour les tâches Amazon Elastic Container Service (Amazon ECS).
- Utilisez des rôles IAM pour les applications qui s'exécutent sur des instances Amazon EC2.

Prérequis : créer un compte AWS

Pour utiliser un utilisateur IAM pour accéder aux AWS services, vous avez besoin d'un AWS compte et d' AWS informations d'identification.

1. Créez un compte.

Pour créer un AWS compte, consultez Commencer : êtes-vous un nouvel AWS utilisateur ? dans le guide Gestion de compte AWS de référence.

2. Créez un utilisateur administratif.

   Évitez d'utiliser votre compte utilisateur root (le compte initial que vous créez) pour accéder à la console de gestion et aux services. Au lieu de cela, créez un compte utilisateur administratif, comme décrit à la section Création d'un utilisateur administratif du Guide de l'utilisateur IAM.

   Après avoir créé le compte utilisateur administratif et enregistré les informations de connexion, veillez à vous déconnecter de votre compte utilisateur root et reconnectez-vous à l'aide du compte administratif.

Aucun de ces comptes n'est approprié pour le développement AWS ou l'exécution d'applications AWS. La meilleure pratique consiste à créer des utilisateurs, des ensembles d'autorisations ou des rôles de service adaptés à ces tâches. Pour en savoir plus, consultez Appliquer les autorisations de moindre privilège dans le Guide de l'utilisateur IAM.

Étape 1 : création de votre utilisateur IAM

- Créez votre utilisateur IAM en suivant la procédure Création d'utilisateurs IAM (console) du Guide de l'utilisateur IAM. Lors de la création de votre utilisateur IAM :
  - Nous vous recommandons de sélectionner Fournir un accès utilisateur au AWS Management Console. Cela vous permet d'afficher les Services AWS informations relatives au code que vous exécutez dans un environnement visuel, par exemple en consultant les journaux de AWS CloudTrail diagnostic ou en téléchargeant des fichiers sur Amazon Simple Storage Service, ce qui est utile lors du débogage de votre code.
  - Pour Définir les autorisations - Options d'autorisation, sélectionnez Joindre directement les politiques pour définir la manière dont vous souhaitez attribuer les autorisations à cet utilisateur.
    - La plupart des didacticiels du kit SDK « Démarrage » utilisent le service Amazon S3 comme exemple. Pour fournir à votre application un accès complet à Amazon S3, sélectionnez la politique `AmazonS3FullAccess` à attacher à cet utilisateur.
  - Vous pouvez ignorer les étapes facultatives de cette procédure concernant la définition des limites d'autorisation ou des balises.

Étape 2 : récupération de vos clés d'accès

1. Dans le volet de navigation de la console IAM, sélectionnez Utilisateurs, puis le **`User name`** de l'utilisateur que vous avez créé précédemment.
2. Sur la page de l'utilisateur, sélectionnez la page Informations d'identification de sécurité. Ensuite, sous Clés d'accès, sélectionnez Créer une clé d'accès.
3. Pour l'étape 1 de création d'une clé d'accès, choisissez soit l'interface de ligne de commande (CLI), soit le code local. Les deux options génèrent le même type de clé à utiliser à la fois avec le AWS CLI et le SDKs.
4. Pour Créer une clé d'accès – Étape 2, saisissez une balise facultative, puis cliquez sur Suivant.
5. Pour Créer une clé d'accès – Étape 3, sélectionnez Télécharger le fichier .csv afin d'enregistrer le fichier `.csv` contenant la clé d'accès et la clé d'accès secrète de l'utilisateur IAM. Vous aurez besoin de ces informations ultérieurement.

   > **Warning**
   >
   > Utilisez les mesures de sécurité appropriées pour protéger ces informations d'identification.

6. Sélectionnez Done (Terminé).

Étape 3 : mettre à jour le **`credentials`** fichier partagé

1. Créez ou ouvrez le fichier AWS `credentials` partagé. Ce fichier est `~/.aws/credentials` sous Linux et macOS, et `%USERPROFILE%\.aws\credentials` sous Windows. Pour plus d'informations, voir Emplacement des fichiers d'informations d'identification.
2. Ajoutez le texte suivant au fichier `credentials` partagé. Remplacez l'exemple de valeur d'ID et l'exemple de valeur de clé par les valeurs du `.csv` fichier que vous avez téléchargé précédemment.

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

3. Enregistrez le fichier.

Le `credentials` fichier partagé est le moyen le plus courant de stocker les informations d'identification. Elles peuvent également être définies en tant que variables d'environnement, voir AWS clés d'accès pour les noms des variables d'environnement. C'est une façon de démarrer, mais nous vous recommandons de passer à IAM Identity Center ou à d'autres informations d'identification temporaires dès que possible. Après avoir cessé d'utiliser des informations d'identification à long terme, pensez à supprimer ces informations d'identification du `credentials` fichier partagé.

Utilisation des rôles IAM pour authentifier les applications déployées sur Amazon EC2

Cet exemple décrit la configuration d'un Gestion des identités et des accès AWS rôle avec un accès Amazon S3 à utiliser dans votre application déployée sur une instance Amazon Elastic Compute Cloud.

Pour exécuter votre application AWS SDK sur une instance Amazon Elastic Compute Cloud, créez un rôle IAM, puis accordez à votre EC2 instance Amazon l'accès à ce rôle. Pour plus d'informations, consultez la section Rôles IAM pour Amazon EC2 dans le guide de l' EC2 utilisateur Amazon.

Créer un rôle IAM

L'application AWS SDK que vous développez accède probablement à au moins une application Service AWS pour effectuer des actions. Créez un rôle IAM qui accorde les autorisations nécessaires à l'exécution de votre application.

Cette procédure crée un rôle qui accorde un accès en lecture seule à Amazon S3, par exemple. La plupart des guides du AWS SDK contiennent des didacticiels de « mise en route » tirés d'Amazon S3.

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le volet de navigation, sélectionnez Rôles, puis sélectionnez Créer un rôle.
3. Pour Sélectionner une entité de confiance, sous Type d'entité fiable, sélectionnez Service AWS.
4. Sous Cas d'utilisation, choisissez Amazon EC2, puis sélectionnez Suivant.
5. Pour Ajouter des autorisations, cochez la case Amazon S3 Read Only Access dans la liste des politiques, puis sélectionnez Suivant.
6. Entrez un nom pour le rôle, puis sélectionnez Créer un rôle. N'oubliez pas ce nom car vous en aurez besoin lors de la création de votre EC2 instance Amazon.

Lancez une EC2 instance Amazon et spécifiez votre rôle IAM

Vous pouvez créer et lancer une EC2 instance Amazon à l'aide de votre rôle IAM en procédant comme suit :

- Suivez Lancer rapidement une instance dans le guide de EC2 l'utilisateur Amazon. Toutefois, avant l'étape finale de soumission, effectuez également les opérations suivantes :
  - Sous Détails avancés, pour le profil d'instance IAM, choisissez le rôle que vous avez créé à l'étape précédente.

Avec cette EC2 configuration IAM et Amazon, vous pouvez déployer votre application sur l' EC2 instance Amazon et votre application aura un accès en lecture au service Amazon S3.

Connect à l' EC2 instance

Connectez-vous à l' EC2 instance Amazon afin de pouvoir y transférer votre application, puis exécuter l'application. Vous aurez besoin du fichier contenant la partie privée de la paire de clés que vous avez utilisée sous Key pair (login) lorsque vous avez créé votre instance, c'est-à-dire le fichier PEM.

Pour ce faire, suivez les instructions correspondant à votre type d'instance : Connect to your Linux instance or Connect to your Windows instance. Lorsque vous vous connectez, faites-le de manière à pouvoir transférer des fichiers de votre machine de développement vers votre instance.

> **Note**
>
> Sur un terminal Linux ou macOS, vous pouvez utiliser la commande de copie sécurisée pour copier votre application. Pour l'utiliser `scp` avec une paire de clés, vous pouvez utiliser la commande suivante : `scp -i path/to/key file/to/copy ec2-user@ec2-xx-xx-xxx-xxx.compute.amazonaws.com:~`
> Pour plus d'informations sur Windows, voir Transférer des fichiers vers des instances Windows.

Si vous utilisez un AWS kit d'outils, vous pouvez souvent également vous connecter à l'instance à l'aide du kit d'outils. Pour plus d'informations, consultez le guide de l'utilisateur spécifique à la boîte à outils que vous utilisez.

Exécutez votre application sur l' EC2 instance

1. Copiez les fichiers de votre application depuis votre disque local vers votre EC2 instance Amazon.
2. Lancez l'application et vérifiez qu'elle s'exécute avec les mêmes résultats que sur votre machine de développement.
3. (Facultatif) Vérifiez que l'application utilise les informations d'identification fournies par le rôle IAM.

    a. Connectez-vous à la EC2 console Amazon AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/ec2/.

    b. Sélectionnez l’instance.

    c. Choisissez Actions, Sécurité, puis Modifier le rôle IAM.

    d. Pour le rôle IAM, détachez le rôle IAM en choisissant Aucun rôle IAM.

    e. Choisissez Mettre le rôle IAM à jour.

    f. Réexécutez l'application et vérifiez qu'elle renvoie une erreur d'autorisation.

Utiliser le plugin TIP pour accéder Services AWS

La propagation sécurisée des identités (TIP) est une fonctionnalité AWS IAM Identity Center qui permet aux administrateurs d' Services AWS accorder des autorisations en fonction des attributs de l'utilisateur tels que les associations de groupes. Avec la propagation d'identité sécurisée, le contexte d'identité est ajouté à un rôle IAM pour identifier l'utilisateur qui demande l'accès aux AWS ressources. Ce contexte est propagé à d'autres Services AWS.

Le contexte d'identité comprend les informations Services AWS utilisées pour prendre des décisions d'autorisation lorsqu'ils reçoivent des demandes d'accès. Ces informations incluent des métadonnées qui identifient le demandeur (par exemple, un utilisateur du IAM Identity Center), l'accès Service AWS auquel l'accès est demandé (par exemple, Amazon Redshift) et l'étendue de l'accès (par exemple, accès en lecture seule). Le destinataire Service AWS utilise ce contexte, ainsi que toutes les autorisations attribuées à l'utilisateur, pour autoriser l'accès à ses ressources. Pour plus d'informations, reportez-vous à la section Vue d'ensemble de la propagation des identités fiables dans le Guide de AWS IAM Identity Center l'utilisateur.

Le plugin TIP peut être utilisé avec un Services AWS support de propagation d'identité fiable. À titre de cas d'utilisation de référence, consultez la section Configuration d'une application Amazon Q Business AWS IAM Identity Centerà l'aide du guide de l'utilisateur Amazon Q Business.

> **Note**
>
> Si vous utilisez Amazon Q Business, consultez Configuration d'une application Amazon Q Business à l'aide AWS IAM Identity Center d'instructions spécifiques au service.

Conditions préalables à l'utilisation du plugin TIP

Les ressources suivantes sont nécessaires pour que le plugin fonctionne :

1. Vous devez utiliser le AWS SDK pour Java ou le AWS SDK pour JavaScript.
2. Vérifiez que le service que vous utilisez prend en charge la propagation d'identités fiables.

   Consultez la colonne Permet la propagation d'identités fiables via IAM Identity Center du tableau des applications AWS gérées qui s'intègrent à IAM Identity Center dans le guide de l'AWS IAM Identity Center utilisateur.
3. Activez le centre d'identité IAM et la propagation fiable des identités.

   Consultez les conditions préalables et considérations relatives au TIP dans le guide de AWS IAM Identity Center l'utilisateur.
4. Vous devez avoir une Identity-Center-integrated candidature.

   Consultez la section Applications AWS gérées ou Applications gérées par le client dans le Guide de AWS IAM Identity Center l'utilisateur.
5. Vous devez configurer un émetteur de jetons sécurisé (TTI) et connecter votre service à IAM Identity Center.

   Consultez les sections Conditions requises pour les émetteurs de jetons fiables et Tâches de configuration d'un émetteur de jetons de confiance dans le guide de l'AWS IAM Identity Center utilisateur.

Pour utiliser le plugin TIP dans votre code

1. Créez une instance du plugin de propagation d'identité sécurisé.

2. Créez une instance de client de service pour interagir avec votre client de service Service AWS et personnalisez-le en ajoutant le plug-in de propagation d'identité sécurisé.

Le plugin TIP prend les paramètres d'entrée suivants :

- **`webTokenProvider`**: fonction que le client implémente pour obtenir un jeton OpenID auprès de son fournisseur d'identité externe.
- **`accessRoleArn`**: L'ARN du rôle IAM à assumer par le plugin avec le contexte d'identité de l'utilisateur pour obtenir les informations d'identification améliorées.
- **`applicationArn`**: chaîne d'identifiant unique du client ou de l'application. Cette valeur est un ARN d'application pour lequel OAuth des autorisations sont configurées.
- **`ssoOidcClient`**: (Facultatif) Un client OIDC SSO, tel que `SsoOidcClient`pour Java ou `client-sso-oidc`pour JavaScript, avec des configurations définies par le client. S'il n'est pas fourni, un client OIDC utilisant `applicationRoleArn` sera instancié et utilisé.
- **`stsClient`**: (Facultatif) Un AWS STS client avec des configurations définies par le client, utilisé pour assumer le `accessRoleArn` contexte d'identité de l'utilisateur. S'il n'est pas fourni, un AWS STS client utilisant `applicationRoleArn` sera instancié et utilisé.
- **`applicationRoleArn`**: (Facultatif) L'ARN du rôle IAM à utiliser pour que l'OIDC et les AWS STS clients puissent être initialisés. `AssumeRoleWithWebIdentity`
  - S'ils ne sont pas fournis, les `stsClient` paramètres `ssoOidcClient` et doivent être fournis.
  - S'il est fourni, il ne `applicationRoleArn` peut pas s'agir de la même valeur que le `accessRoleArn` paramètre. `applicationRoleArn`est utilisé pour créer le STSClient, qui est utilisé pour assumer le rôle AccessRole. Si le même rôle est utilisé pour `applicationRole` les deux`accessRole`, cela signifierait utiliser un rôle pour s'assumer (hypothèse d'un rôle propre), ce qui est déconseillé par. AWS Consultez l'annonce pour plus de détails.

Considérations relatives à **`ssoOidcClient`**`stsClient`, et **`applicationRoleArn`** paramètres

Lors de la configuration du plug-in TIP, tenez compte des exigences d'autorisation suivantes en fonction des paramètres que vous fournissez :

- Si vous fournissez `ssoOidcClient` et `stsClient` :

  - Les informations d'identification figurant sur le `ssoOidcClient` doivent être `oauth:CreateTokenWithIAM` autorisées à appeler le centre d'identité afin d'obtenir le contexte utilisateur spécifique au centre d'identité.
  - Les informations d'identification `stsClient` devraient être `sts:AssumeRole` activées et `sts:SetContext` les autorisations activées`accessRole`. `accessRole`doit également être configuré avec une relation de confiance avec les informations d'identification activées`stsClient`.
- Si vous fournissez `applicationRoleArn` :
  - `applicationRole`doit disposer des `oauth:CreateTokenWithIAM` `sts:SetContext` autorisations `sts:AssumeRole` et des autorisations sur les ressources requises (instance Id`accessRole`) car elles seront utilisées pour créer des clients OIDC et STS.
  - `applicationRole`doit avoir une relation de confiance avec le fournisseur d'identité utilisé pour générer le`webToken`, car il `webToken` sera utilisé pour assumer le rôle d'application via l'AssumeRoleWithWebIdentityappel du plugin.

Exemple ApplicationRole de configuration :

Politique de confiance avec le fournisseur de jetons Web :

```
{
    "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"
                }
            }
        }
    ]
}
```

Politique d'autorisation :

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

Exemples de code utilisant TIP

Les exemples ci-dessous montrent comment implémenter le plugin TIP dans votre code à l'aide du AWS SDK pour Java ou du AWS SDK pour JavaScript.

Java

Pour utiliser le plugin TIP dans votre AWS SDK pour Java projet, vous devez le déclarer en tant que dépendance dans le `pom.xml` fichier de votre projet.

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

Dans votre code source, incluez l'instruction de package requise pour`software.amazon.awssdk.trustedidentitypropagation`.

Les exemples suivants montrent deux manières de créer une instance du plugin de propagation d'identité sécurisé et de l'ajouter à un client de service. Les deux exemples utilisent Amazon S3 en tant que service et sont utilisés `S3AccessGrantsPlugin` pour gérer les autorisations spécifiques aux utilisateurs, mais ils peuvent être appliqués à tout Service AWS ce qui prend en charge la propagation d'identité sécurisée (TIP).

> **Note**
>
> Pour ces exemples, vous devez configurer les autorisations spécifiques à l'utilisateur à partir de S3 Access Grants. Reportez-vous à la documentation relative aux subventions d'accès S3 pour plus de détails.

Option 1 : créer et transmettre des clients OIDC et STS

```
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());
```

Option 2 : Transférer applicationRoleArn et reporter la création du client au plugin

```
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());
```

Pour plus de détails et la source, voir trusted-identity-propagation-javaci-dessous GitHub.

JavaScript

Exécutez la commande suivante pour installer le package du plugin d'authentification TIP dans votre AWS SDK pour JavaScript projet :

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

La version finale `package.json` doit inclure une dépendance similaire à la suivante :

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

Dans votre code source, importez la `TrustedIdentityPropagationExtension` dépendance requise.

Les exemples suivants montrent deux manières de créer une instance du plugin de propagation d'identité sécurisé et de l'ajouter à un client de service. Les deux exemples utilisent Amazon S3 en tant que service et utilisent Amazon S3 Access Grants pour gérer les autorisations spécifiques aux utilisateurs, mais ils peuvent être appliqués à tout Service AWS ce qui prend en charge la propagation d'identité sécurisée (TIP).

> **Note**
>
> Pour ces exemples, vous devez configurer les autorisations spécifiques à l'utilisateur à partir d'Amazon S3 Access Grants. Consultez la documentation Amazon S3 Access Grants pour plus de détails.

Option 1 : créer et transmettre des clients OIDC et STS

```
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);
}
```

Option 2 : Transférer applicationRoleArn et reporter la création du client au plugin

```
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);
}
```

Pour plus de détails et la source, voir trusted-identity-propagation-jsci-dessous GitHub.

AWS SDKs et référence des paramètres des outils

SDKs fournir des informations spécifiques à la langue APIs pour. Services AWS Ils prennent en charge certaines des tâches les plus lourdes nécessaires à la réussite des appels d'API, notamment l'authentification, le comportement des nouvelles tentatives, etc. Pour ce faire, SDKs ils disposent de stratégies flexibles pour obtenir des informations d'identification à utiliser pour vos demandes, pour maintenir les paramètres à utiliser avec chaque service et pour obtenir des valeurs à utiliser pour les paramètres globaux.

Vous trouverez des informations détaillées sur les paramètres de configuration dans les sections suivantes :

- AWS SDKs et outils, fournisseurs d'accréditations standardisés— Fournisseurs d'informations d'identification communs normalisés sur plusieurs SDKs.
- AWS SDKs et fonctionnalités standardisées des outils— Fonctionnalités communes standardisées sur plusieurs SDKs.

Création de clients de service

Pour y accéder par programmation Services AWS, SDKs utilisez un client class/object pour chacun d'entre eux. Service AWS Par exemple, si votre application doit accéder à Amazon EC2, elle crée un objet EC2 client Amazon pour interagir avec ce service. Vous utilisez ensuite le client du service pour y faire des demandes Service AWS. Dans la plupart des cas SDKs, un objet client de service est immuable. Vous devez donc créer un nouveau client pour chaque service auquel vous faites des demandes et pour envoyer des demandes au même service en utilisant une configuration différente.

Priorité des paramètres

Les paramètres globaux configurent les fonctionnalités, les fournisseurs d'informations d'identification et les autres fonctionnalités prises en charge par la plupart des utilisateurs SDKs et ayant un large impact sur l'ensemble Services AWS de la population. Tous SDKs ont une série de lieux (ou de sources) qu'ils vérifient afin de trouver une valeur pour les paramètres globaux. La définition de la priorité de recherche est la suivante :

1. Tout paramètre explicite défini dans le code ou sur un client de service lui-même a priorité sur tout autre paramètre.

- Certains paramètres peuvent être définis pour chaque opération et peuvent être modifiés selon les besoins pour chaque opération que vous invoquez. Pour le AWS CLI ou Outils AWS pour PowerShell, ils prennent la forme de paramètres par opération que vous entrez sur la ligne de commande. Pour un SDK, les attributions explicites peuvent prendre la forme d'un paramètre que vous définissez lorsque vous instanciez un Service AWS client ou un objet de configuration, ou parfois lorsque vous appelez une API individuelle.

2. Java/Kotlin uniquement : la propriété du système JVM pour le paramètre est vérifiée. Si elle est définie, cette valeur est utilisée pour configurer le client.
3. La variable d'environnement est contrôlée. Si elle est définie, cette valeur est utilisée pour configurer le client.
4. Le SDK vérifie le paramètre dans `credentials` le fichier partagé. S'il est défini, le client l'utilise.
5. Le `config` fichier partagé pour le paramètre. Si le paramètre est présent, le SDK l'utilise.
   - La variable d'`AWS_PROFILE`environnement ou la propriété du système `aws.profile` JVM peuvent être utilisées pour spécifier le profil chargé par le SDK.
6. Toute valeur par défaut fournie par le code source du SDK lui-même est utilisée en dernier.

> **Note**
>
> Certains SDKs outils peuvent être enregistrés dans un ordre différent. En outre, certains SDKs outils prennent en charge d'autres méthodes de stockage et de récupération des paramètres. Par exemple, il AWS SDK pour .NET prend en charge une source supplémentaire appelée SDK Store. Pour plus d'informations sur les fournisseurs spécifiques à un SDK ou à un outil, consultez le guide spécifique au SDK ou à l'outil que vous utilisez.

L'ordre détermine quelles méthodes ont priorité et remplacent les autres. Par exemple, si vous configurez un profil dans le `config` fichier partagé, il n'est trouvé et utilisé qu'une fois que le SDK ou l'outil a d'abord vérifié les autres emplacements. Cela signifie que si vous insérez un paramètre dans le `credentials` fichier, il est utilisé à la place de celui qui se trouve dans le `config` fichier. Si vous configurez une variable d'environnement avec un paramètre et une valeur, elle remplacera ce paramètre dans les `config` fichiers `credentials` et. Enfin, un réglage sur l'opération individuelle (paramètre de AWS CLI ligne de commande ou paramètre d'API) ou dans le code remplacerait toutes les autres valeurs de cette commande.

Comprendre les pages de paramètres de ce guide

Les pages de la section de référence des paramètres de ce guide détaillent les paramètres disponibles qui peuvent être définis par le biais de différents mécanismes. Les tableaux suivants répertorient les paramètres des fichiers de configuration et d'identification, les variables d'environnement et (pour Java et Kotlin SDKs) les paramètres JVM qui peuvent être utilisés en dehors de votre code pour configurer la fonctionnalité. Chaque rubrique liée dans chaque liste vous amène à la page de paramètres correspondante.

- `Config`liste des paramètres de fichier
- `Credentials`liste des paramètres de fichier
- Liste des variables d'environnement
- Liste des propriétés du système JVM

Chaque fournisseur d'informations d'identification ou fonctionnalité possède une page répertoriant les paramètres utilisés pour configurer cette fonctionnalité. Pour chaque paramètre, vous pouvez souvent définir la valeur soit en ajoutant le paramètre à un fichier de configuration, soit en définissant une variable d'environnement, soit (pour Java et Kotlin uniquement) en définissant une propriété système JVM. Chaque paramètre répertorie toutes les méthodes prises en charge pour définir la valeur dans un bloc situé au-dessus des détails de la description. Bien que la priorité varie, la fonctionnalité qui en résulte est la même, quelle que soit la manière dont vous la définissez.

La description inclura la valeur par défaut, le cas échéant, qui prend effet si vous ne faites rien. Il définit également la valeur valide pour ce paramètre.

Par exemple, examinons un paramètre de la page des Compression des demandes fonctionnalités.

Les informations de l'`disable_request_compression`exemple de paramètre documentent les éléments suivants :

- Il existe trois méthodes équivalentes pour contrôler la compression des demandes en dehors de votre base de code. Vous avez le choix entre les options suivantes :
  - Définissez-le dans votre fichier de configuration en utilisant `disable_request_compression`
  - Définissez-la comme variable d'environnement en utilisant `AWS_DISABLE_REQUEST_COMPRESSION`

- Ou, si vous utilisez le SDK Java ou Kotlin, définissez-le comme propriété du système JVM en utilisant `aws.disableRequestCompression`

> **Note**
>
> Il existe peut-être également un moyen de configurer la même fonctionnalité directement dans votre code, mais cette référence ne couvre pas ce point car il est unique à chaque SDK. Si vous souhaitez définir votre configuration dans le code lui-même, consultez le guide de votre SDK ou votre référence d'API spécifique.

- Si vous ne faites rien, la valeur par défaut sera`false`.
- Les seules valeurs valides pour ce paramètre booléen sont `true` et. `false`

Au bas de chaque page de fonctionnalité se trouve un tableau Support by AWS SDKs et Tools.

Ce tableau indique si votre SDK prend en charge les paramètres répertoriés sur la page. La `Supported` colonne indique le niveau de support avec les valeurs suivantes :

- `Yes`— Les paramètres sont entièrement pris en charge par le SDK tel qu'il est écrit.
- `Partial`— Certains paramètres sont pris en charge ou le comportement s'écarte de la description. En `Partial` effet, une note supplémentaire indique l'écart.
- No— Aucun des paramètres n'est pris en charge. Cela ne prétend pas que les mêmes fonctionnalités peuvent être obtenues dans le code ; cela indique simplement que les paramètres de configuration externes répertoriés ne sont pas pris en charge.

`Config`liste des paramètres de fichier

Les paramètres répertoriés dans le tableau suivant peuvent être attribués dans le AWS `config` fichier partagé. Elles sont mondiales et concernent tout le monde Services AWS. SDKs et les outils peuvent également prendre en charge des paramètres et des variables d'environnement uniques. Pour voir les paramètres et les variables d'environnement pris en charge uniquement par un SDK ou un outil individuel, consultez ce SDK ou ce guide d'outils spécifique.

Nom du paramètre	Détails
account_id_endpoint_mode	Points de terminaison basés sur des comptes
api_versions	Paramètres de configuration généraux
auth_scheme_preference	Schéma d'authentification
aws_access_key_id	AWS clés d'accès
aws_account_id	Points de terminaison basés sur des comptes
aws_secret_access_key	AWS clés d'accès
aws_session_token	AWS clés d'accès
ca_bundle	Paramètres de configuration généraux
credential_process	Fournisseur d'identifiants de processus
credential_source	Assumer le rôle de fournisseur d'informations d'identification
defaults_mode	Paramètres de configuration intelligents par défaut
disable_host_prefix_injection	Injection du préfixe hôte

Nom du paramètre	Détails
disable_request_compression	Compression des demandes
duration_seconds	Assumer le rôle de fournisseur d'informations d'identification
ec2_metadata_service_endpoint	fournisseur d'informations d'identification IMDS
ec2_metadata_service_endpoint_mode	fournisseur d'informations d'identification IMDS
ec2_metadata_v1_disabled	fournisseur d'informations d'identification IMDS
endpoint_discovery_enabled	Découverte des terminaux
endpoint_url	Points de terminaison spécifiques au service
external_id	Assumer le rôle de fournisseur d'informations d'identification
ignore_configured_endpoint_urls	Points de terminaison spécifiques au service
max_attempts	Comportement des nouvelles tentatives

Nom du paramètre	Détails	
`metadata_service_num_attempts`	Métadonnées de EC2 l'instance Amazon	
`metadata_service_timeout`	Métadonnées de EC2 l'instance Amazon	
`mfa_serial`	Assumer le rôle de fournisseur d'informations d'identification	
`output`	Paramètres de configuration généraux	
`parameter_validation`	Paramètres de configuration généraux	
`region`	Région AWS	
`request_checksum_calculation`	Protections de l'intégrité des données pour Amazon S3	
`request_min_compression_size_bytes`	Compression des demandes	
`response_checksum_validation`	Protections de l'intégrité des données pour Amazon S3	
`retry_mode`	Comportement des nouvelles tentatives	
`role_arn`	Assumer le rôle de fournisseur d'informations d'identification	

Nom du paramètre	Détails
`role_session_name`	Assumer le rôle de fournisseur d'informations d'identification
`s3_disable_express_session_auth`	Authentification de session S3 Express One Zone
`s3_disable_multiregion_access_points`	Amazon S3 Multi-Region Access Points
`s3_use_arn_region`	Points d’accès Amazon S3
`sdk_ua_app_id`	ID d'application
`sigv4a_signing_region_set`	Schéma d'authentification
`source_profile`	Assumer le rôle de fournisseur d'informations d'identification
`sso_account_id`	Fournisseur d'identifiants IAM Identity Center
`sso_region`	Fournisseur d'identifiants IAM Identity Center
`sso_registration_scopes`	Fournisseur d'identifiants IAM Identity Center
`sso_role_name`	Fournisseur d'identifiants IAM Identity Center
`sso_start_url`	Fournisseur d'identifiants IAM Identity Center
`sts_regional_endpoints`	AWS STS Points de terminaison régionaux

Nom du paramètre	Détails
`use_dualstack_endpoint`	Points de terminaison à double pile et FIPS
`use_fips_endpoint`	Points de terminaison à double pile et FIPS
`web_identity_token_file`	Assumer le rôle de fournisseur d'informations d'identification

`Credentials`liste des paramètres de fichier

Les paramètres répertoriés dans le tableau suivant peuvent être attribués dans le AWS `credentials` fichier partagé. Elles sont mondiales et concernent tout le monde Services AWS. SDKs et les outils peuvent également prendre en charge des paramètres et des variables d'environnement uniques. Pour voir les paramètres et les variables d'environnement pris en charge uniquement par un SDK ou un outil individuel, consultez ce SDK ou ce guide d'outils spécifique.

Nom du paramètre	Détails
`aws_access_key_id`	AWS clés d'accès
`aws_secret_access_key`	AWS clés d'accès
`aws_session_token`	AWS clés d'accès

Liste des variables d'environnement

Les variables d'environnement prises en charge par la plupart SDKs sont répertoriées dans le tableau suivant. Elles sont mondiales et concernent tout le monde Services AWS. SDKs et les outils peuvent également prendre en charge des paramètres et des variables d'environnement uniques. Pour voir

les paramètres et les variables d'environnement pris en charge uniquement par un SDK ou un outil individuel, consultez ce SDK ou ce guide d'outils spécifique.

Nom du paramètre	Détails	
AWS_ACCESS_KEY_ID	AWS clés d'accès	
AWS_ACCOUNT_ID	Points de terminaison basés sur des comptes	
AWS_ACCOUNT_ID_ENDPOINT_MODE	Points de terminaison basés sur des comptes	
AWS_AUTH_SCHEME_PREFERENCE	Schéma d'authentification	
AWS_CA_BUNDLE	Paramètres de configuration généraux	
AWS_CONFIG_FILE	Recherche et modification de l'emplacement du partage, des `credentials` fichiers `config`AWS SDKs et des outils	
AWS_CONTAINER_AUTHORIZATION_TOKEN	Fournisseur d'informations d'identification du conteneur	
AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE	Fournisseur d'informations d'identification du conteneur	
AWS_CONTAINER_CREDENTIALS_FULL_URI	Fournisseur d'informations d'identification du conteneur	

Nom du paramètre	Détails	
AWS_CONTAINER_CREDENTIALS_RELATIVE_URI	Fournisseur d'informations d'identification du conteneur	
AWS_DEFAULTS_MODE	Paramètres de configuration intelligents par défaut	
AWS_DISABLE_HOST_PREFIX_INJECTION	Injection du préfixe hôte	
AWS_DISABLE_REQUEST_COMPRESSION	Compression des demandes	
AWS_EC2_METADATA_DISABLED	fournisseur d'informations d'identification IMDS	
AWS_EC2_METADATA_SERVICE_ENDPOINT	fournisseur d'informations d'identification IMDS	
AWS_EC2_METADATA_SERVICE_ENDPOINT_MODE	fournisseur d'informations d'identification IMDS	
AWS_EC2_METADATA_V1_DISABLED	fournisseur d'informations d'identification IMDS	

Nom du paramètre	Détails	
AWS_ENABLE_ENDPOINT_DISCOVERY	Découverte des terminaux	
AWS_ENDPOINT_URL	Points de terminaison spécifiques au service	
AWS_ENDPOINT_URL_<SERVICE>	Points de terminaison spécifiques au service	
AWS_IGNORE_CONFIGURED_ENDPOINT_URLS	Points de terminaison spécifiques au service	
AWS_MAX_ATTEMPTS	Comportement des nouvelles tentatives	
AWS_METADATA_SERVICE_NUM_ATTEMPTS	Métadonnées de EC2 l'instance Amazon	
AWS_METADATA_SERVICE_TIMEOUT	Métadonnées de EC2 l'instance Amazon	
AWS_PROFILE	Utilisation du partage `config` et `credentials` des fichiers pour une configuration AWS SDKs et des outils globaux	
AWS_REGION	Région AWS	
AWS_REQUEST_CHECKSUM_CALCULATION	Protections de l'intégrité des données pour Amazon S3	

Nom du paramètre	Détails	
AWS_REQUEST_MIN_COMPRESSION_SIZE_BYTES	Compression des demandes	
AWS_RESPONSE_CHECKSUM_VALIDATION	Protections de l'intégrité des données pour Amazon S3	
AWS_RETRY_MODE	Comportement des nouvelles tentatives	
AWS_ROLE_ARN	Assumer le rôle de fournisseur d'informations d'identification	
AWS_ROLE_SESSION_NAME	Assumer le rôle de fournisseur d'informations d'identification	
AWS_S3_DISABLE_EXPRESS_SESSION_AUTH	Authentification de session S3 Express One Zone	
AWS_S3_DISABLE_MULTIREGION_ACCESS_POINTS	Amazon S3 Multi-Region Access Points	
AWS_S3_USE_ARN_REGION	Points d’accès Amazon S3	
AWS_SDK_UA_APP_ID	ID d'application	
AWS_SECRET_ACCESS_KEY	AWS clés d'accès	

Nom du paramètre	Détails	
`AWS_SESSION_TOKEN`	AWS clés d'accès	
`AWS_SHARED_CREDENTIALS_FILE`	Recherche et modification de l'emplacement du partage, des `credentials` fichiers `config`AWS SDKs et des outils	
`AWS_SIGV4A_SIGNING_REGION_SET`	Schéma d'authentification	
`AWS_STS_REGIONAL_ENDPOINTS`	AWS STS Points de terminaison régionaux	
`AWS_USE_DUALSTACK_ENDPOINT`	Points de terminaison à double pile et FIPS	
`AWS_USE_FIPS_ENDPOINT`	Points de terminaison à double pile et FIPS	
`AWS_WEB_IDENTITY_TOKEN_FILE`	Assumer le rôle de fournisseur d'informations d'identification	

Liste des propriétés du système JVM

Vous pouvez utiliser les propriétés du système JVM suivantes pour le AWS SDK pour Java et AWS SDK pour Kotlin (en ciblant la JVM). Consultez the section called “Comment définir les propriétés du système JVM” les instructions sur la façon de définir les propriétés du système JVM.

Nom du paramètre	Détails	
`aws.accessKeyId`	AWS clés d'accès	

Nom du paramètre	Détails
`aws.accountId`	Points de terminaison basés sur des comptes
`aws.accountIdEndpointMode`	Points de terminaison basés sur des comptes
`aws.authSchemePreference`	Schéma d'authentification
`aws.configFile`	Recherche et modification de l'emplacement du partage, des `credentials` fichiers `config`AWS SDKs et des outils
`aws.defaultsMode`	Paramètres de configuration intelligents par défaut
`aws.disableEc2MetadataV1`	fournisseur d'informations d'identification IMDS
`aws.disableHostPrefixInjection`	Injection du préfixe hôte
`aws.disableRequestCompression`	Compression des demandes
`aws.disableS3ExpressAuth`	Authentification de session S3 Express One Zone
`aws.ec2MetadataServiceEndpoint`	fournisseur d'informations d'identification IMDS

Nom du paramètre	Détails	
`aws.ec2MetadataServiceEndpointMode`	fournisseur d'informations d'identification IMDS	
`aws.endpointDiscoveryEnabled`	Découverte des terminaux	
`aws.endpointUrl`	Points de terminaison spécifiques au service	
`aws.endpointUrl<ServiceName>`	Points de terminaison spécifiques au service	
`aws.ignoreConfiguredEndpointUrls`	Points de terminaison spécifiques au service	
`aws.maxAttempts`	Comportement des nouvelles tentatives	
`aws.profile`	Utilisation du partage `config` et `credentials` des fichiers pour une configuration AWS SDKs et des outils globaux	
`aws.region`	Région AWS	
`aws.requestChecksumCalculation`	Protections de l'intégrité des données pour Amazon S3	
`aws.requestMinCompressionSizeBytes`	Compression des demandes	

Nom du paramètre	Détails	
`aws.responseChecksumValidation`	Protections de l'intégrité des données pour Amazon S3	
`aws.retryMode`	Comportement des nouvelles tentatives	
`aws.roleArn`	Assumer le rôle de fournisseur d'informations d'identification	
`aws.roleSessionName`	Assumer le rôle de fournisseur d'informations d'identification	
`aws.s3DisableMultiRegionAccessPoints`	Amazon S3 Multi-Region Access Points	
`aws.s3UseArnRegion`	Points d’accès Amazon S3	
`aws.secretAccessKey`	AWS clés d'accès	
`aws.sessionToken`	AWS clés d'accès	
`aws.sharedCredentialsFile`	Recherche et modification de l'emplacement du partage, des `credentials` fichiers `config`AWS SDKs et des outils	
`aws.useDualstackEndpoint`	Points de terminaison à double pile et FIPS	
`aws.useFipsEndpoint`	Points de terminaison à double pile et FIPS	

Nom du paramètre	Détails
`aws.webId entityTok enFile`	Assumer le rôle de fournisseur d'informations d'identification
`sdk.ua.appId`	ID d'application

AWS SDKs et outils, fournisseurs d'accréditations standardisés

De nombreux fournisseurs d'informations d'identification ont été normalisés selon des valeurs par défaut cohérentes et fonctionnent de la même manière pour de nombreux fournisseurs. SDKs Cette cohérence augmente la productivité et la clarté lors du codage sur plusieurs SDKs. Tous les paramètres peuvent être remplacés dans le code. Pour plus de détails, consultez l'API de votre SDK spécifique.

> **Important**
>
> Tous ne prennent pas SDKs en charge tous les fournisseurs, ni même tous les aspects d'un fournisseur.

Rubriques

- Comprendre la chaîne des fournisseurs d'informations d'identification
- Chaînes de fournisseurs d'informations d'identification spécifiques au SDK et aux outils
- AWS clés d'accès
- Fournisseur d'identifiants de connexion
- Assumer le rôle de fournisseur d'informations d'identification
- Fournisseur d'informations d'identification du conteneur
- Fournisseur d'identifiants IAM Identity Center
- fournisseur d'informations d'identification IMDS
- Fournisseur d'identifiants de processus

Comprendre la chaîne des fournisseurs d'informations d'identification

Tous SDKs ont une série de lieux (ou de sources) qu'ils vérifient afin de trouver des informations d'identification valides à utiliser pour faire une demande à un Service AWS. Une fois les informations d'identification valides trouvées, la recherche s'arrête. Cette recherche systématique est appelée chaîne de fournisseurs d'informations d'identification.

Lorsque vous utilisez l'un des fournisseurs d'informations d'identification standardisés, ils essaient AWS SDKs toujours de renouveler automatiquement les informations d'identification lorsqu'elles expirent. La chaîne de fournisseurs d'informations d'identification intégrée permet à votre application d'actualiser vos informations d'identification quel que soit le fournisseur que vous utilisez dans la chaîne. Aucun code supplémentaire n'est requis pour que le SDK puisse effectuer cette opération.

Bien que la chaîne distincte utilisée par chaque SDK varie, ils incluent le plus souvent des sources telles que les suivantes :

Fournisseur d'informations d'identification	Description
AWS clés d'accès	AWS clés d'accès pour un utilisateur IAM (telles que`AWS_ACCESS_KEY_ID` , et`AWS_SECRET_ACCESS_KEY`).
Fédérez avec l'identité Web ou OpenID Connect- Assumer le rôle de fournisseur d'informations d'identification	Connectez-vous à l'aide d'un fournisseur d'identité externe (IdP) connu, tel que Login with Amazon, Facebook, Google ou tout autre IdP compatible avec OpenID Connect (OIDC). Assumez les autorisations d'un rôle IAM à l'aide d'un jeton Web JSON (JWT) provenant de AWS Security Token Service (AWS STS).
Fournisseur d'identifiants de connexion	Obtenez les informations d'identification pour une session de console nouvelle ou existante à laquelle vous êtes connecté.
Fournisseur d'identifiants IAM Identity Center	Obtenez des informations d'identification auprès de AWS IAM Identity Center.

Fournisseur d'informations d'identification	Description
Assumer le rôle de fournisseur d'informations d'identification	Accédez à d'autres ressources en assumant les autorisations d'un rôle IAM. (Récupérez puis utilisez les informations d'identification temporaires pour un rôle).
Fournisseur d'informations d'identification du conteneur	Informations d'identification Amazon Elastic Container Service (Amazon ECS) et Amazon Elastic Kubernetes Service (Amazon EKS). Le fournisseur d'informations d'identification du conteneur récupère les informations d'identification de l'application conteneurisée du client.
Fournisseur d'identifiants de processus	Fournisseur d'informations d'identification personnalisé. Obtenez vos informations d'identification auprès d'une source ou d'un processus externe, notamment IAM Roles Anywhere.
fournisseur d'informations d'identification IMDS	Informations d'identification du profil d'instance Amazon Elastic Compute Cloud (Amazon EC2). Associez un rôle IAM à chacune de vos instances EC2. Les informations d'identification temporaires pour ce rôle sont mises à la disposition du code exécuté dans l'instance. Les informations d'identification sont fournies au moyen du service de métadonnées Amazon EC2.

Pour chaque étape de la chaîne, il existe plusieurs manières d'attribuer des valeurs de réglage. La définition des valeurs spécifiées dans le code est toujours prioritaire. Cependant, il y a aussi Variables d’environnement et lesUtilisation du partage `config` et `credentials` des fichiers pour une configuration globale AWS SDKs et des outils. Pour de plus amples informations, veuillez consulter Priorité des paramètres.

Chaînes de fournisseurs d'informations d'identification spécifiques au SDK et aux outils

Pour accéder directement aux détails de la chaîne de fournisseurs d'informations d'identification spécifique à votre SDK ou à votre outil, choisissez votre SDK ou outil parmi les options suivantes :

- AWS CLI
- SDK pour C++
- SDK pour Go
- SDK pour Java
- SDK pour JavaScript
- SDK pour Kotlin
- SDK pour .NET
- SDK pour PHP
- SDK pour Python (Boto3)
- SDK pour Ruby
- SDK pour Rust
- SDK pour Swift
- Outils pour PowerShell

AWS clés d'accès

> **Warning**
>
> Afin d'éviter les risques de sécurité, n'employez pas les utilisateurs IAM pour l'authentification lorsque vous développez des logiciels spécialisés ou lorsque vous travaillez avec des données réelles. Préférez la fédération avec un fournisseur d'identité tel que AWS IAM Identity Center.

AWS les clés d'accès d'un utilisateur IAM peuvent être utilisées comme AWS informations d'identification. Le AWS SDK utilise automatiquement ces AWS informations d'identification pour signer les demandes d'API AWS, afin que vos charges de travail puissent accéder à vos AWS ressources et à vos données de manière sûre et pratique. Il est recommandé de toujours utiliser le `aws_session_token` afin que les informations d'identification soient temporaires et ne soient plus valides après leur expiration. L'utilisation d'informations d'identification à long terme n'est pas recommandée.

> **Note**
>
> Si vous AWS ne parvenez pas à actualiser ces informations d'identification temporaires, cela AWS peut prolonger la validité des informations d'identification afin que vos charges de travail ne soient pas affectées.

Le AWS `credentials` fichier partagé est l'emplacement recommandé pour stocker les informations d'identification, car il se trouve en toute sécurité en dehors des répertoires des sources de l'application et distinct des paramètres spécifiques au SDK du fichier partagé`config`.

Pour en savoir plus sur les informations AWS d'identification et l'utilisation des clés d'accès, consultez les AWS sections Identifiants de sécurité et Gestion des clés d'accès pour les utilisateurs IAM dans le Guide de l'utilisateur IAM.

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`aws_access_key_id`- réglage AWS **`config`** du fichier partagé, **`aws_access_key_id`**- réglage AWS **`credentials`** du fichier partagé (méthode recommandée), **`AWS_ACCESS_KEY_ID`**- variable d'environnement, **`aws.accessKeyId`**- Propriété du système JVM : uniquement Java/Kotlin

Spécifie la clé AWS d'accès utilisée dans le cadre des informations d'identification pour authentifier l'utilisateur.

`aws_secret_access_key`- réglage AWS **`config`** du fichier partagé, **`aws_secret_access_key`**- réglage AWS **`credentials`** du fichier partagé (méthode recommandée), **`AWS_SECRET_ACCESS_KEY`**- variable d'environnement, **`aws.secretAccessKey`**- Propriété du système JVM : uniquement Java/Kotlin

Spécifie la clé AWS secrète utilisée dans le cadre des informations d'identification pour authentifier l'utilisateur.

`aws_session_token`- réglage AWS **`config`** du fichier partagé, **`aws_session_token`**- réglage AWS **`credentials`** du fichier partagé (méthode recommandée), **`AWS_SESSION_TOKEN`**- variable d'environnement, **`aws.sessionToken`**- Propriété du système JVM : uniquement Java/Kotlin

Spécifie un jeton de AWS session utilisé dans le cadre des informations d'identification pour authentifier l'utilisateur. Vous recevez cette valeur dans le cadre des informations d'identification temporaires renvoyées en cas de réussite des demandes d'attribution d'un rôle. Un jeton de session n'est nécessaire que si vous spécifiez manuellement des informations d'identification de sécurité temporaires. Cependant, nous vous recommandons de toujours utiliser des informations

de sécurité temporaires plutôt que des informations d'identification à long terme. Pour les recommandations en matière de sécurité, consultez la section Meilleures pratiques en matière de sécurité dans IAM.

Pour obtenir des instructions sur la façon d'obtenir ces valeurs, reportez-vous àUtilisation d'informations d'identification à court terme pour l'authentification AWS SDKs et d'outils.

Exemple de définition de ces valeurs obligatoires dans le `credentials` fichier `config` or :

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

Exemple Linux/macOS de définition de variables d'environnement via la ligne de commande :

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

Exemple Windows de définition de variables d'environnement via la ligne de commande :

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

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
AWS CLI v2	Oui	

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
SDK pour C++	Oui	`config`le fichier partagé n'est pas pris en charge.
SDK pour Go V2 (1.x)	Oui	
SDK pour Go 1.x (V1)	Oui	Pour utiliser les paramètres des `config` fichiers partagés, vous devez activer le chargement à partir du fichier de configuration ; voir Sessions.
SDK pour Java 2.x	Oui	
SDK pour Java 1.x	Oui	
SDK pour 3.x JavaScript	Oui	
SDK pour 2.x JavaScript	Oui	
SDK pour Kotlin	Oui	
SDK pour .NET 4.x	Oui	
SDK pour .NET 3.x	Oui	
SDK pour PHP 3.x	Oui	
SDK pour Python (Boto3)	Oui	
SDK pour Ruby 3.x	Oui	
SDK pour Rust	Oui	
SDK pour Swift	Oui	
Outils pour PowerShell V5	Oui	
Outils pour PowerShell V4	Oui	Les variables d'environnement ne sont pas prises en charge.

Fournisseur d'identifiants de connexion

Vous pouvez utiliser vos informations de connexion à AWS la console de gestion existantes pour obtenir des informations d'identification à court terme qui peuvent être utilisées pour un accès par programmation. Une fois que vous avez terminé le flux d'authentification basé sur le navigateur, AWS génère des informations d'identification temporaires qui fonctionnent avec les outils de développement locaux tels que la AWS CLI, AWS Tools for PowerShell et. AWS SDKs

Pour générer ces informations d'identification, exécutez la `aws login` commande dans la AWS CLI ou l'`Invoke-AWSLogin`applet de commande dans AWS Tools for. PowerShell Les informations d'identification à court terme qui en résultent seront mises en cache localement, où elles pourront être réutilisées par le. AWS SDKs Les informations d'identification à court terme expirent au bout de 15 minutes, mais la CLI les actualise SDKs automatiquement selon les besoins jusqu'à 12 heures. Lorsque le jeton d'actualisation expire, vous êtes invité à vous reconnecter via la CLI ou PowerShell.

La commande de connexion met à jour le profil que vous spécifiez avec le `login_session` paramètre, qui stocke l'identité de la session de console de gestion que vous avez sélectionnée pendant le flux de travail de connexion.

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

Par défaut, les informations d'identification à court terme et le jeton d'actualisation sont stockés dans un fichier JSON du `~/.aws/login/cache` répertoire sous Linux et macOS, ou `%USERPROFILE%\.aws\login\cache` sous Windows. Le nom du fichier est basé sur le nom de la session de connexion. Vous pouvez remplacer le répertoire en définissant la variable d'`AWS_LOGIN_CACHE_DIRECTORY`environnement.

Paramètres du fournisseur de connexion

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`AWS_LOGIN_CACHE_DIRECTORY`- variable d'environnement

Répertoire alternatif dans lequel la CLI SDKs stockera les informations d'identification mises en cache qui correspondent à un profil de session de connexion.

Valeur par défaut : `~/.aws/login/cache` sous Linux et macOS, ou `%USERPROFILE%\.aws\login\cache` sous Windows.

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
AWS CLI v2	Oui	
SDK pour C++	Oui	
SDK pour Go V2 (1.x)	Nor	
SDK pour Go 1.x (V1)	Oui	
SDK pour Java 2.x	Oui	
SDK pour Java 1.x	Nor	
SDK pour 3.x JavaScript	Oui	
SDK pour 2.x JavaScript	Nor	
SDK pour Kotlin	Oui	
SDK pour .NET 4.x	Oui	
SDK pour .NET 3.x	Oui	
SDK pour PHP 3.x	Oui	
SDK pour Python (Boto3)	Oui	Nécessite un tube cathodique
SDK pour Ruby 3.x	Oui	
SDK pour Rust	Oui	
Outils pour PowerShell V5	Oui	

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
Outils pour PowerShell V4	Nor	

Assumer le rôle de fournisseur d'informations d'identification

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

Assumer un rôle implique l'utilisation d'un ensemble d'informations d'identification de sécurité temporaires pour accéder à AWS des ressources auxquelles vous n'auriez peut-être pas accès autrement. Ces informations d'identification temporaires incluent un ID de clé d'accès, une clé d'accès secrète et un jeton de sécurité.

Pour configurer votre SDK ou votre outil afin qu'il assume un rôle, vous devez d'abord créer ou identifier un rôle spécifique à assumer. Les rôles IAM sont identifiés de manière unique par un rôle Amazon Resource Name (ARN). Les rôles établissent des relations de confiance avec une autre entité. L'entité de confiance qui utilise le rôle peut être un Service AWS fournisseur d'identité Web Compte AWS, une fédération OIDC ou SAML.

Une fois le rôle IAM identifié, si ce rôle vous fait confiance, vous pouvez configurer votre SDK ou votre outil pour utiliser les autorisations accordées par le rôle. Pour ce faire, utilisez les paramètres suivants.

Pour savoir comment commencer à utiliser ces paramètres, consultez Assumer un rôle avec des AWS informations d'identification pour l'authentification AWS SDKs et des outils ce guide.

Paramètres du fournisseur d'informations d'identification du rôle

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`credential_source`- réglage AWS **`config`** du fichier partagé

Utilisé dans les instances Amazon EC2 ou les conteneurs Amazon Elastic Container Service pour spécifier où le SDK ou l'outil peut trouver les informations d'identification autorisées à assumer le rôle que vous spécifiez avec le paramètre. `role_arn`

Valeur par défaut : Aucune

Valeurs valides:

- Environnement — Spécifie que le SDK ou l'outil doit récupérer les informations d'identification de la source à partir des variables d'environnement `AWS_ACCESS_KEY_IDet AWS_SECRET_ACCESS_KEY`.
- Ec2 InstanceMetadata — Spécifie que le SDK ou l'outil doit utiliser le rôle IAM attaché au profil d'instance EC2 pour obtenir les informations d'identification de la source.
- EcsContainer— Spécifie que le SDK ou l'outil doit utiliser le rôle IAM attaché au conteneur Amazon ECS ou le rôle IAM attaché au conteneur Amazon EKS pour obtenir les informations d'identification de la source.

Vous ne pouvez pas spécifier à la fois `credential_source` et `source_profile` dans le même profil.

Exemple de définition de ce paramètre dans un `config` fichier pour indiquer que les informations d'identification doivent provenir d'Amazon EC2 :

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

`duration_seconds`- réglage AWS **`config`** du fichier partagé

Spécifie la durée maximale de la session de rôle, en secondes.

Ce paramètre s'applique uniquement lorsque le profil indique d'assumer un rôle.

Valeur par défaut : 3 600 secondes (une heure)

Valeurs valides : La valeur peut aller de 900 secondes (15 minutes) à la durée maximale de session configurée pour le rôle (qui peut être de 43 200 secondes ou 12 heures au maximum). Pour plus d'informations, voir Afficher le paramètre de durée maximale de session pour un rôle dans le guide de l'utilisateur IAM.

Exemple de configuration de ce paramètre dans un `config` fichier :

```
duration_seconds = 43200
```

`external_id`- réglage AWS **`config`** du fichier partagé

Identifiant unique utilisé par des tiers pour assumer un rôle dans les comptes de leurs clients.

Ce paramètre s'applique uniquement lorsque le profil indique d'assumer un rôle et que la politique de confiance associée à ce rôle nécessite une valeur pour`ExternalId`. La valeur correspond au `ExternalId` paramètre transmis à l'`AssumeRole`opération lorsque le profil spécifie un rôle.

Valeur par défaut : Aucune.

Valeurs valides : consultez la section Comment utiliser un identifiant externe lorsque vous accordez l'accès à vos AWS ressources à un tiers dans le guide de l'utilisateur IAM.

Exemple de configuration de ce paramètre dans un `config` fichier :

```
external_id = unique_value_assigned_by_3rd_party
```

`mfa_serial`- réglage AWS **`config`** du fichier partagé

Spécifie l'identification ou le numéro de série d'un dispositif d'authentification multifactorielle (MFA) que l'utilisateur doit utiliser lorsqu'il assume un rôle.

Obligatoire lorsque vous assumez un rôle où la politique de confiance associée à ce rôle inclut une condition nécessitant une authentification MFA. Pour plus d'informations sur l'authentification multifactorielle, consultez la section Authentification AWS multifactorielle dans le guide de l'utilisateur IAM.

Valeur par défaut : Aucune.

Valeurs valides : la valeur peut être soit un numéro de série pour un périphérique matériel (tel que`GAHT12345678`), soit un Amazon Resource Name (ARN) pour un périphérique MFA virtuel. Le format de l'ARN est le suivant : `arn:aws:iam::`*`account-id`*`:mfa/`*`mfa-device-name`*

Exemple de configuration de ce paramètre dans un `config` fichier :

Cet exemple suppose qu'un périphérique MFA virtuel, appelé`MyMFADevice`, a été créé pour le compte et activé pour un utilisateur.

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

`role_arn`- réglage AWS **`config`** du fichier partagé, **`AWS_ROLE_ARN`**- variable d'environnement, **`aws.roleArn`**- Propriété du système JVM : uniquement Java/Kotlin

Spécifie le nom de ressource Amazon (ARN) d'un rôle IAM que vous souhaitez utiliser pour effectuer les opérations demandées à l'aide de ce profil.

Valeur par défaut : Aucune.

Valeurs valides : la valeur doit être l'ARN d'un rôle IAM, formaté comme suit : `arn:aws:iam::`*`account-id`*`:role/`*`role-name`*

En outre, vous devez également définir l'un des paramètres suivants :

- `source_profile`— Pour identifier un autre profil à utiliser pour trouver les informations d'identification autorisées à assumer le rôle dans ce profil.
- `credential_source`— Pour utiliser les informations d'identification identifiées par les variables d'environnement actuelles ou les informations d'identification associées à un profil d'instance Amazon EC2 ou à une instance de conteneur Amazon ECS.
- `web_identity_token_file`— Utiliser des fournisseurs d'identité publics ou tout autre fournisseur d'identité compatible avec OpenID Connect (OIDC) pour les utilisateurs authentifiés dans une application mobile ou Web.

`role_session_name`- réglage AWS **`config`** du fichier partagé, **`AWS_ROLE_SESSION_NAME`**- variable d'environnement, **`aws.roleSessionName`**- Propriété du système JVM : uniquement Java/Kotlin

Spécifie le nom à attacher à la session de rôle. Ce nom apparaît dans AWS CloudTrail les journaux des entrées associées à cette session, ce qui peut être utile lors d'un audit. Pour plus de détails, consultez l'élément CloudTrail UserIdentity dans le guide de l'AWS CloudTrail utilisateur.

Valeur par défaut : paramètre facultatif. Si vous ne fournissez pas cette valeur, un nom de session est généré automatiquement si le profil assume un rôle.

Valeurs valides : fournies au `RoleSessionName` paramètre lorsque l' AWS API AWS CLI or appelle l'`AssumeRole`opération (ou des opérations telles que l'`AssumeRoleWithWebIdentity`opération) en votre nom. La valeur fait partie du rôle assumé par l'utilisateur Amazon Resource Name (ARN) que vous pouvez interroger et apparaît dans les entrées du CloudTrail journal pour les opérations invoquées par ce profil.

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

Exemple de configuration de ce paramètre dans un `config` fichier :

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

`source_profile`- réglage AWS **`config`** du fichier partagé

Spécifie un autre profil dont les informations d'identification sont utilisées pour assumer le rôle spécifié par le `role_arn` paramètre du profil d'origine. Pour comprendre comment les profils sont utilisés dans le partage AWS `config` et `credentials` les fichiers, consultezPartage `config` et `credentials` fichiers.

Si vous spécifiez un profil qui est également un profil d'acceptation de rôle, chaque rôle sera assumé dans un ordre séquentiel afin de résoudre complètement les informations d'identification. Cette chaîne est arrêtée lorsque le SDK rencontre un profil avec des informations d'identification. Le chaînage des rôles limite votre session de rôle AWS CLI ou celle de l' AWS API à un maximum d'une heure et ne peut pas être augmenté. Pour plus d'informations, consultez la section Termes et concepts relatifs aux rôles dans le guide de l'utilisateur d'IAM.

Valeur par défaut : Aucune.

Valeurs valides : chaîne de texte composée du nom d'un profil défini dans les `credentials` fichiers `config` et. Vous devez également spécifier une valeur pour `role_arn` dans le profil actuel.

Vous ne pouvez pas spécifier à la fois `credential_source` et `source_profile` dans le même profil.

Exemple de configuration dans un fichier de configuration :

```
[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
```

Dans l'exemple précédent, le `A` profil indique au SDK ou à l'outil de rechercher automatiquement les informations d'identification du `B` profil lié. Dans ce cas, le `B` profil utilise l'outil d'aide aux

informations d'identification fourni par pour Utilisation d'IAM Roles Anywhere pour l'authentification et les outils AWS SDKs obtenir les informations d'identification du AWS SDK. Ces informations d'identification temporaires sont ensuite utilisées par votre code pour accéder aux AWS ressources. Le rôle spécifié doit être associé à des politiques d'autorisation IAM autorisant l'exécution du code demandé, par exemple la commande ou la méthode d'API. Service AWS Le nom de session du rôle est inclus dans les CloudTrail journaux pour chaque action entreprise par profilA.

Pour un deuxième exemple de chaînage de rôles, la configuration suivante peut être utilisée si vous avez une application sur une instance Amazon Elastic Compute Cloud et que vous souhaitez que cette application assume un autre rôle.

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

[profile B]
credential_source=Ec2InstanceMetadata
```

Le profil A utilisera les informations d'identification de l'instance Amazon EC2 pour assumer le rôle spécifié et les renouvellera automatiquement.

`web_identity_token_file`- réglage AWS **`config`** du fichier partagé, **`AWS_WEB_IDENTITY_TOKEN_FILE`**- variable d'environnement, **`aws.webIdentityTokenFile`**- Propriété du système JVM : uniquement Java/Kotlin

Spécifie le chemin d'accès à un fichier contenant un jeton d'accès provenant d'un fournisseur OAuth 2.0 compatible ou d'un fournisseur d'identité OpenID Connect ID.

Ce paramètre active l'authentification en utilisant des fournisseurs de fédération d'identité Web, tels que Google, Facebook et Amazon, entre autres. Le SDK ou l'outil de développement charge le contenu de ce fichier et le transmet comme `WebIdentityToken` argument lorsqu'il appelle l'`AssumeRoleWithWebIdentity`opération en votre nom.

Valeur par défaut : Aucune.

Valeurs valides : cette valeur doit être un chemin et un nom de fichier. Le fichier doit contenir un jeton d'accès OAuth 2.0 ou un jeton OpenID Connect qui vous a été fourni par un fournisseur d'identité. Les chemins relatifs sont traités comme relatifs au répertoire de travail du processus.

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
AWS CLI v2	Oui	
SDK pour C++	Partie	`credential_source` non pris en charge. `duration_seconds` non pris en charge. `mfa_serial` non pris en charge.
SDK pour Go V2 (1.x)	Oui	
SDK pour Go 1.x (V1)	Oui	Pour utiliser les paramètres des `config` fichiers partagés, vous devez activer le chargement à partir du fichier de configuration ; voir Sessions.
SDK pour Java 2.x	Partie	`mfa_serial` non pris en charge. `duration_seconds` non pris en charge.
SDK pour Java 1.x	Partie	`credential_source` non pris en charge. `mfa_serial` non pris en charge. Les propriétés du système JVM ne sont pas prises en charge.
SDK pour 3.x JavaScript	Oui	
SDK pour 2.x JavaScript	Partie	`credential_source` non pris en charge.
SDK pour Kotlin	Oui	
SDK pour .NET 4.x	Oui	
SDK pour .NET 3.x	Oui	
SDK pour PHP 3.x	Oui	

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
SDK pour Python (Boto3)	Oui	
SDK pour Ruby 3.x	Oui	
SDK pour Rust	Oui	
SDK pour Swift	Oui	
Outils pour PowerShell V5	Oui	
Outils pour PowerShell V4	Oui	

Fournisseur d'informations d'identification du conteneur

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

Le fournisseur d'informations d'identification du conteneur récupère les informations d'identification de l'application conteneurisée du client. Ce fournisseur d'informations d'identification est utile pour les clients d'Amazon Elastic Container Service (Amazon ECS) et d'Amazon Elastic Kubernetes Service (Amazon EKS). SDKs tente de charger les informations d'identification depuis le point de terminaison HTTP spécifié via une requête GET.

Si vous utilisez Amazon ECS, nous vous recommandons d'utiliser un rôle IAM de tâche pour améliorer l'isolation, l'autorisation et l'auditabilité des informations d'identification. Une fois configuré, Amazon ECS définit la variable d'`AWS_CONTAINER_CREDENTIALS_RELATIVE_URI`environnement que les outils SDKs et utilisent pour obtenir des informations d'identification. Pour configurer Amazon ECS pour cette fonctionnalité, consultez le rôle de Task IAM dans le manuel Amazon Elastic Container Service Developer Guide.

Si vous utilisez Amazon EKS, nous vous recommandons d'utiliser Amazon EKS Pod Identity pour améliorer l'isolation des informations d'identification, le moindre privilège, l'auditabilité, le fonctionnement indépendant, la réutilisabilité et l'évolutivité. Votre Pod et un rôle IAM sont associés à un compte de service Kubernetes pour gérer les informations d'identification de vos applications. Pour en savoir plus sur Amazon EKS Pod Identity, consultez Amazon EKS Pod Identities dans le guide de l'utilisateur Amazon EKS. Une fois configuré, Amazon EKS définit les variables d'`AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE`environnement `AWS_CONTAINER_CREDENTIALS_FULL_URI` et que les outils SDKs et utilisent pour obtenir des informations d'identification. Pour plus d'informations sur la configuration, consultez Configuration de l'agent Amazon EKS Pod Identity dans le guide de l'utilisateur Amazon EKS ou Amazon EKS Pod Identity simplifie les autorisations IAM pour les applications sur les clusters Amazon EKS sur le site Web du AWS blog.

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`AWS_CONTAINER_CREDENTIALS_FULL_URI`- variable d'environnement

Spécifie le point de terminaison de l'URL HTTP complet que le SDK doit utiliser lors d'une demande d'informations d'identification. Cela inclut à la fois le schéma et l'hôte.

Valeur par défaut : Aucune.

Valeurs valides : URI valide.

Remarque : Ce paramètre est une alternative *`AWS_CONTAINER_CREDENTIALS_RELATIVE_URI`* et ne sera utilisé que s'il n'*`AWS_CONTAINER_CREDENTIALS_RELATIVE_URI`*est pas défini.

Exemple Linux/macOS de définition de variables d'environnement via la ligne de commande :

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

or

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

`AWS_CONTAINER_CREDENTIALS_RELATIVE_URI`- variable d'environnement

Spécifie le point de terminaison de l'URL HTTP relative que le SDK doit utiliser lors d'une demande d'informations d'identification. La valeur est ajoutée au nom d'hôte Amazon ECS par défaut de. `169.254.170.2`

Valeur par défaut : Aucune.

Valeurs valides : URI relative valide.

Exemple Linux/macOS de définition de variables d'environnement via la ligne de commande :

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

`AWS_CONTAINER_AUTHORIZATION_TOKEN`- variable d'environnement

Spécifie un jeton d'autorisation en texte brut. Si cette variable est définie, le SDK définira l'en-tête d'autorisation de la requête HTTP avec la valeur de la variable d'environnement.

Valeur par défaut : Aucune.

Valeurs valides : chaîne.

Remarque : Ce paramètre est une alternative *`AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE`* et ne sera utilisé que s'il n'*`AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE`*est pas défini.

Exemple Linux/macOS de définition de variables d'environnement via la ligne de commande :

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

`AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE`- variable d'environnement

Spécifie un chemin de fichier absolu vers un fichier contenant le jeton d'autorisation en texte brut.

Valeur par défaut : Aucune.

Valeurs valides : chaîne.

Exemple Linux/macOS de définition de variables d'environnement via la ligne de commande :

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

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
AWS CLI v2	Oui	
SDK pour C++	Oui	
SDK pour Go V2 (1.x)	Oui	
SDK pour Go 1.x (V1)	Oui	
SDK pour Java 2.x	Oui	Lorsque Lambda SnapStart est activé, `AWS_CONTAINER_CREDENTIALS_FULL_URI` il est automatiquement utilisé pour l'authentification. `AWS_CONTAINER_AUTHORIZATION_TOKEN`
SDK pour Java 1.x	Oui	Lorsque Lambda SnapStart est activé, `AWS_CONTAINER_CREDENTIALS_FULL_URI` il est automatiquement utilisé pour l'authentification. `AWS_CONTAINER_AUTHORIZATION_TOKEN`
SDK pour 3.x JavaScript	Oui	
SDK pour 2.x JavaScript	Oui	
SDK pour Kotlin	Oui	
SDK pour .NET 4.x	Oui	Lorsque Lambda SnapStart est activé, `AWS_CONTAINER_CREDENTIALS_FULL_URI` il est automatiquement utilisé pour l'authentification. `AWS_CONTAINER_AUTHORIZATION_TOKEN`

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
SDK pour .NET 3.x	Oui	Lorsque Lambda SnapStart est activé, `AWS_CONTAINER_CREDENTIALS_FULL_URI` il est automatiquement utilisé pour l'authentification. `AWS_CONTAINER_AUTHORIZATION_TOKEN`
SDK pour PHP 3.x	Oui	
SDK pour Python (Boto3)	Oui	Lorsque Lambda SnapStart est activé, `AWS_CONTAINER_CREDENTIALS_FULL_URI` il est automatiquement utilisé pour l'authentification. `AWS_CONTAINER_AUTHORIZATION_TOKEN`
SDK pour Ruby 3.x	Oui	
SDK pour Rust	Oui	
SDK pour Swift	Oui	
Outils pour PowerShell V5	Oui	
Outils pour PowerShell V4	Oui	

Fournisseur d'identifiants IAM Identity Center

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

Ce mécanisme d'authentification permet AWS IAM Identity Center d'obtenir un accès par authentification unique (SSO) Services AWS à votre code.

> **Note**
>
> Dans la documentation de l'API du AWS SDK, le fournisseur d'informations d'identification IAM Identity Center est appelé fournisseur d'informations d'identification SSO.

Après avoir activé IAM Identity Center, vous définissez un profil pour ses paramètres dans votre AWS `config` fichier partagé. Ce profil est utilisé pour se connecter au portail d'accès IAM Identity Center. Lorsqu'un utilisateur s'authentifie avec succès auprès d'IAM Identity Center, le portail renvoie des informations d'identification à court terme pour le rôle IAM associé à cet utilisateur. Pour savoir comment le SDK obtient des informations d'identification temporaires à partir de la configuration et les utilise pour les Service AWS demandes, consultezComment l'authentification IAM Identity Center est-elle résolue AWS SDKs et quels outils ?.

Il existe deux manières de configurer IAM Identity Center via le `config` fichier :

- Configuration du fournisseur de jetons SSO (recommandée) — Durées de session prolongées. Inclut la prise en charge de durées de session personnalisées.
- Configuration ancienne non actualisable : utilise une session fixe de huit heures.

Dans les deux configurations, vous devez vous reconnecter à l'expiration de votre session.

Les deux guides suivants contiennent des informations supplémentaires sur IAM Identity Center :

- AWS IAM Identity Center Guide de l'utilisateur
- AWS IAM Identity Center Référence de l'API du portail

Pour en savoir plus sur la façon dont les outils SDKs et utilisent et actualisent les informations d'identification à l'aide de cette configuration, consultezComment l'authentification IAM Identity Center est-elle résolue AWS SDKs et quels outils ?.

Conditions préalables

Vous devez d'abord activer IAM Identity Center. Pour plus de détails sur l'activation de l'authentification IAM Identity Center, consultez la section Activation AWS IAM Identity Center dans le guide de AWS IAM Identity Center l'utilisateur.

> **Note**
>
> Sinon, pour connaître les prérequis complets et la configuration de `config` fichiers partagés nécessaire, détaillés sur cette page, consultez les instructions de configurationUtilisation d'IAM Identity Center pour authentifier le AWS SDK et les outils.

Configuration du fournisseur de jetons SSO

Lorsque vous utilisez la configuration du fournisseur de jetons SSO, votre AWS SDK ou outil actualise automatiquement votre session jusqu'à votre période de session prolongée. Pour plus d'informations sur la durée de session et la durée maximale, voir Configurer la durée de session du portail d' AWS accès et des applications intégrées d'IAM Identity Center dans le guide de l'AWS IAM Identity Center utilisateur.

La `sso-session` section du `config` fichier est utilisée pour regrouper les variables de configuration permettant d'acquérir des jetons d'accès SSO, qui peuvent ensuite être utilisés pour acquérir des AWS informations d'identification. Pour plus de détails sur cette section d'un `config` fichier, consultezFormat du fichier de configuration.

L'exemple de `config` fichier partagé suivant configure le SDK ou l'outil à l'aide d'un `dev` profil pour demander les informations d'identification IAM Identity Center.

```
[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
```

Les exemples précédents montrent que vous définissez une `sso-session` section et que vous l'associez à un profil. Généralement, `sso_account_id` et `sso_role_name` doit être défini dans la `profile` section afin que le SDK puisse demander des AWS informations d'identification. `sso_regionsso_start_url`, et `sso_registration_scopes` doit être défini dans la `sso-session` section.

`sso_account_id`et `sso_role_name` ne sont pas obligatoires pour tous les scénarios de configuration de jetons SSO. Si votre application utilise uniquement Services AWS cette authentification par support, les informations d' AWS identification traditionnelles ne sont pas nécessaires. L'authentification par jeton de porteur est un schéma d'authentification HTTP qui utilise des jetons de sécurité appelés jetons de porteur. Dans ce scénario, `sso_account_id` et `sso_role_name` ne sont pas obligatoires. Consultez le Service AWS guide individuel pour déterminer si le service prend en charge l'autorisation par jeton au porteur.

Les étendues d'enregistrement sont configurées dans le cadre d'un`sso-session`. Scope est un mécanisme permettant OAuth 2.0 de limiter l'accès d'une application au compte d'un utilisateur. L'exemple précédent vise `sso_registration_scopes` à fournir l'accès nécessaire pour répertorier les comptes et les rôles.

L'exemple suivant montre comment vous pouvez réutiliser la même `sso-session` configuration sur plusieurs profils.

```
[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
```

Le jeton d'authentification est mis en cache sur le disque sous le `~/.aws/sso/cache` répertoire avec un nom de fichier basé sur le nom de session.

Ancienne configuration non actualisable

L'actualisation automatique des jetons n'est pas prise en charge avec la configuration non actualisable héritée. Nous vous recommandons d'utiliser le à la Configuration du fournisseur de jetons SSO place.

Pour utiliser l'ancienne configuration non actualisable, vous devez spécifier les paramètres suivants dans votre profil :

- `sso_start_url`
- `sso_region`
- `sso_account_id`
- `sso_role_name`

Vous spécifiez le portail utilisateur d'un profil avec les `sso_region` paramètres `sso_start_url` et. Vous spécifiez les autorisations à l'aide `sso_account_id` des `sso_role_name` paramètres et.

L'exemple suivant définit les quatre valeurs requises dans le `config` fichier.

```
[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
```

Le jeton d'authentification est mis en cache sur le disque sous le `~/.aws/sso/cache` répertoire avec un nom de fichier basé sur le`sso_start_url`.

Paramètres du fournisseur d'informations d'identification IAM Identity Center

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`sso_start_url`- réglage AWS **`config`** du fichier partagé

URL pointant vers l'URL de l'émetteur du centre d'identité IAM ou l'URL du portail d'accès de votre organisation. Pour plus d'informations, consultez la section Utilisation du portail AWS d'accès dans le guide de AWS IAM Identity Center l'utilisateur.

Pour trouver cette valeur, ouvrez la console IAM Identity Center, consultez le tableau de bord, recherchez l'URL du portail AWS d'accès.

- À partir de la version 2.22.0 du AWS CLI, vous pouvez également utiliser la valeur de l'URL de l'AWS émetteur.

`sso_region`- réglage AWS **`config`** du fichier partagé

Celui Région AWS qui contient l'hôte de votre portail IAM Identity Center, c'est-à-dire la région que vous avez sélectionnée avant d'activer IAM Identity Center. Elle est indépendante de votre AWS région par défaut et peut être différente.

Pour une liste complète des Régions AWS et de leurs codes, consultez la section Points de terminaison régionaux dans le Référence générale d'Amazon Web Services. Pour trouver cette valeur, ouvrez la console IAM Identity Center, consultez le tableau de bord et recherchez Region.

`sso_account_id`- réglage AWS **`config`** du fichier partagé

L'ID numérique du Compte AWS qui a été ajouté via le AWS Organizations service à utiliser pour l'authentification.

Pour consulter la liste des comptes disponibles, accédez à la console IAM Identity Center et ouvrez la Comptes AWSpage. Vous pouvez également consulter la liste des comptes disponibles à l'aide de la méthode ListAccountsAPI dans le manuel de référence des API du AWS IAM Identity Center portail. Par exemple, vous pouvez appeler la AWS CLI méthode list-accounts.

`sso_role_name`- réglage AWS **`config`** du fichier partagé

Nom d'un ensemble d'autorisations fourni en tant que rôle IAM qui définit les autorisations obtenues par l'utilisateur. Le rôle doit exister dans le champ Compte AWS spécifié par`sso_account_id`. Utilisez le nom du rôle, et non le nom Amazon Resource Name (ARN) du rôle.

Les ensembles d'autorisations sont associés à des politiques IAM et à des politiques d'autorisations personnalisées et définissent le niveau d'accès des utilisateurs à ce qui leur est attribué Comptes AWS.

Pour voir la liste des ensembles d'autorisations disponibles par Compte AWS, accédez à la console IAM Identity Center et ouvrez la Comptes AWSpage. Choisissez le nom du jeu d'autorisations correct répertorié dans le Comptes AWS tableau. Vous pouvez également consulter la liste des ensembles d'autorisations disponibles à l'aide de la méthode ListAccountRolesAPI dans le manuel de référence des API du AWS IAM Identity Center portail. Par exemple, vous pouvez appeler la AWS CLI méthode list-account-roles.

`sso_registration_scopes`- réglage AWS **`config`** du fichier partagé

Une liste séparée par des virgules de chaînes de portée valides à autoriser pour. `sso-session` Une application peut demander une ou plusieurs portées, et le jeton d'accès émis pour l'application est limité aux portées accordées. Une portée minimale de `sso:account:access`

doit être accordée pour récupérer un jeton d'actualisation auprès du service IAM Identity Center. Pour obtenir la liste des options d'étendue d'accès disponibles, voir Étendue d'accès dans le Guide de l'AWS IAM Identity Center utilisateur.

Ces portées définissent les autorisations demandées à accorder au client OIDC enregistré ainsi que les jetons d'accès que ce client obtient. Les portées autorisent l'accès aux points de terminaison autorisés par le jeton de porteur IAM Identity Center.

Ce paramètre ne s'applique pas à l'ancienne configuration non actualisable. Les jetons émis à l'aide de l'ancienne configuration sont `sso:account:access` implicitement limités à leur portée.

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
AWS CLI v2	Oui	
SDK pour C++	Oui	
SDK pour Go V2 (1.x)	Oui	
SDK pour Go 1.x (V1)	Oui	Pour utiliser les paramètres des `config` fichiers partagés, vous devez activer le chargement à partir du fichier de configuration ; voir Sessions.
SDK pour Java 2.x	Oui	Les valeurs de configuration sont également prises en charge dans `credentials` le fichier.
SDK pour Java 1.x	Nor	
SDK pour 3.x JavaScript	Oui	
SDK pour 2.x JavaScript	Oui	

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
SDK pour Kotlin	Oui	
SDK pour .NET 4.x	Oui	
SDK pour .NET 3.x	Oui	
SDK pour PHP 3.x	Oui	
SDK pour Python (Boto3)	Oui	
SDK pour Ruby 3.x	Oui	
SDK pour Rust	Partie	Configuration héritée non actualisable uniquement.
SDK pour Swift	Oui	
Outils pour PowerShell V5	Oui	
Outils pour PowerShell V4	Oui	

fournisseur d'informations d'identification IMDS

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

Le service de métadonnées d'instance (IMDS) fournit des données sur votre instance que vous pouvez utiliser pour configurer ou gérer l'instance en cours d'exécution. Pour plus d'informations sur les données disponibles, consultez la section Utilisation des métadonnées d'instance dans le guide de l'utilisateur Amazon EC2. Amazon EC2 fournit un point de terminaison local à la disposition des instances qui peut fournir différentes informations à l'instance. Si un rôle est attaché à l'instance, elle peut fournir un ensemble d'informations d'identification valides pour ce rôle. Ils SDKs peuvent utiliser

ce point de terminaison pour résoudre les informations d'identification dans le cadre de leur chaîne de fournisseurs d'informations d'identification par défaut. Le service de métadonnées d'instance version 2 (IMDSv2), une version plus sécurisée d'IMDS qui utilise un jeton de session, est utilisé par défaut. Si cela échoue en raison d'une condition non réessayable (codes d'erreur HTTP 403, 404, 405), il IMDSv1 est utilisé comme solution de secours.

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`AWS_EC2_METADATA_DISABLED`- variable d'environnement

Essayer ou non d'utiliser le service de métadonnées d'instance Amazon EC2 (IMDS) pour obtenir des informations d'identification.

Valeur par défaut : `false`.

Valeurs valides:

- **`true`**— N'utilisez pas l'IMDS pour obtenir des informations d'identification.
- **`false`**— Utilisez IMDS pour obtenir des informations d'identification.

`ec2_metadata_v1_disabled`- réglage AWS **`config`** du fichier partagé, **`AWS_EC2_METADATA_V1_DISABLED`**- variable d'environnement, **`aws.disableEc2MetadataV1`**- Propriété du système JVM : uniquement Java/Kotlin

S'il faut ou non utiliser le service de métadonnées d'instance version 1 (IMDSv1) comme solution de secours en cas d' IMDSv2 échec.

> **Note**
>
> Les nouveautés SDKs ne sont pas compatibles avec ce paramètre IMDSv1 et ne le prennent donc pas en charge. Pour plus de détails, voir le tableauSupport par AWS SDKs et outils.

Valeur par défaut : `false`.

Valeurs valides:

- **`true`**— Ne pas utiliser IMDSv1 comme solution de secours.
- **`false`**— À utiliser IMDSv1 comme solution de secours.

`ec2_metadata_service_endpoint`- réglage AWS **`config`** du fichier partagé, **`AWS_EC2_METADATA_SERVICE_ENDPOINT`**- variable d'environnement, **`aws.ec2MetadataServiceEndpoint`**- Propriété du système JVM : uniquement Java/Kotlin

Le point final de l'IMDS. Cette valeur remplace l'emplacement par défaut dans lequel les AWS SDK et les outils rechercheront les métadonnées des instances Amazon EC2.

Valeur par défaut : si elle `ec2_metadata_service_endpoint_mode` est égale`IPv4`, le point de terminaison par défaut est`http://169.254.169.254`. Si `ec2_metadata_service_endpoint_mode` égal`IPv6`, le point de terminaison par défaut est`http://[fd00:ec2::254]`.

Valeurs valides : URI valide.

`ec2_metadata_service_endpoint_mode`- réglage AWS **`config`** du fichier partagé, **`AWS_EC2_METADATA_SERVICE_ENDPOINT_MODE`**- variable d'environnement, **`aws.ec2MetadataServiceEndpointMode`**- Propriété du système JVM : uniquement Java/Kotlin

Le mode endpoint de l'IMDS.

Valeur par défaut :`IPv4`.

Valeurs valides :`IPv4`,`IPv6`.

> **Note**
>
> Le fournisseur d'informations d'identification IMDS fait partie du. Comprendre la chaîne des fournisseurs d'informations d'identification Cependant, le fournisseur d'informations d'identification IMDS n'est vérifié qu'après plusieurs autres fournisseurs de cette série. Par conséquent, si vous souhaitez que votre programme utilise les informations d'identification de ce fournisseur, vous devez supprimer les autres fournisseurs d'informations d'identification valides de votre configuration ou utiliser un profil différent. Sinon, au lieu de vous fier à la chaîne de fournisseurs d'informations d'identification pour découvrir automatiquement quel fournisseur renvoie des informations d'identification valides, spécifiez l'utilisation du fournisseur d'informations d'identification IMDS dans le code. Vous pouvez spécifier les sources d'informations d'identification directement lorsque vous créez des clients de service.

Sécurité des informations d'identification IMDS

Par défaut, lorsque le AWS SDK n'est pas configuré avec des informations d'identification valides, le SDK tente d'utiliser le service de métadonnées d'instance Amazon EC2 (IMDS) pour récupérer les informations d'identification d'un rôle. AWS Ce comportement peut être désactivé en définissant la variable d'`AWS_EC2_METADATA_DISABLED`environnement sur`true`. Cela empêche toute activité réseau inutile et améliore la sécurité sur les réseaux non fiables sur lesquels le service de métadonnées d'instance Amazon EC2 peut être usurpé.

> **Note**
>
> AWS Les clients du SDK configurés avec des informations d'identification valides n'utiliseront jamais IMDS pour récupérer des informations d'identification, quels que soient ces paramètres.

Désactivation de l'utilisation des informations d'identification Amazon EC2 IMDS

La façon dont vous définissez cette variable d'environnement dépend du système d'exploitation utilisé et du fait que vous souhaitez ou non que la modification soit persistante.

Linux et macOS

Les clients utilisant Linux ou macOS peuvent définir cette variable d'environnement à l'aide de la commande suivante :

```
$ export AWS_EC2_METADATA_DISABLED=true
```

Si vous souhaitez que ce paramètre soit permanent pendant plusieurs sessions et redémarrages du système, vous pouvez ajouter la commande ci-dessus à votre fichier de profil shell, telle que `.bash_profile`.`zsh_profile`, ou`.profile`.

Windows

Les clients utilisant Windows peuvent définir cette variable d'environnement à l'aide de la commande suivante :

```
$ set AWS_EC2_METADATA_DISABLED=true
```

Si vous souhaitez que ce paramètre soit persistant sur plusieurs sessions shell et redémarrages du système, vous pouvez utiliser la commande suivante à la place :

```
$ setx AWS_EC2_METADATA_DISABLED=true
```

> **Note**
>
> La setx commande n'applique pas la valeur à la session shell en cours. Vous devrez donc recharger ou rouvrir le shell pour que la modification soit prise en compte.

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
AWS CLI v2	Oui	
SDK pour C++	Oui	
SDK pour Go V2 (1.x)	Oui	
SDK pour Go 1.x (V1)	Oui	Pour utiliser les paramètres des `config` fichiers partagés, vous devez activer le chargement à partir du fichier de configuration ; voir Sessions.
SDK pour Java 2.x	Oui	
SDK pour Java 1.x	Partie	Propriétés du système JVM : à utiliser à la `com.amazonaws.sdk.disableEc2MetadataV1` place de `aws.disableEc2MetadataV1` ; `aws.ec2MetadataServiceEndpoint` et `aws.ec2MetadataServiceEndpointMode` non prises en charge.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
SDK pour 3.x JavaScript	Oui	
SDK pour 2.x JavaScript	Oui	
SDK pour Kotlin	Oui	N'utilise pas de solution de IMDSv1 secours.
SDK pour .NET 4.x	Oui	
SDK pour .NET 3.x	Oui	
SDK pour PHP 3.x	Oui	
SDK pour Python (Boto3)	Oui	
SDK pour Ruby 3.x	Oui	
SDK pour Rust	Oui	N'utilise pas de solution de IMDSv1 secours.
SDK pour Swift	Oui	
Outils pour PowerShell V5	Oui	Vous pouvez désactiver le IMDSv1 repli de manière explicite dans le code en utilisant`[Amazon.Util.EC2InstanceMetadata]::EC2MetadataV1Disabled = $true`.
Outils pour PowerShell V4	Oui	Vous pouvez désactiver le IMDSv1 repli de manière explicite dans le code en utilisant`[Amazon.Util.EC2InstanceMetadata]::EC2MetadataV1Disabled = $true`.

Fournisseur d'identifiants de processus

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

SDKs fournir un moyen d'étendre la chaîne de fournisseurs d'informations d'identification pour les cas d'utilisation personnalisés. Ce fournisseur peut être utilisé pour fournir des implémentations personnalisées, telles que la récupération d'informations d'identification à partir d'un magasin d'informations d'identification local ou l'intégration à votre fournisseur d'identité local.

Par exemple, IAM Roles Anywhere permet `credential_process` d'obtenir des informations d'identification temporaires au nom de votre application. `credential_process`Pour configurer cet usage, voirUtilisation d'IAM Roles Anywhere pour l'authentification et les outils AWS SDKs.

> **Note**
>
> Ce qui suit décrit une méthode d'obtention d'informations d'identification auprès d'un processus externe qui peut être utilisée si vous exécutez un logiciel en dehors de AWS. Si vous utilisez une ressource de AWS calcul, utilisez d'autres fournisseurs d'informations d'identification. Si vous utilisez cette option, vous devez vous assurer que le fichier de configuration est aussi verrouillé que possible conformément aux meilleures pratiques de sécurité de votre système d'exploitation. Vérifiez que votre outil d'identification personnalisé n'écrit aucune information secrèteStdErr, car le SDKs and AWS CLI peut capturer et enregistrer ces informations, les exposant potentiellement à des utilisateurs non autorisés.

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`credential_process`- réglage AWS **`config`** du fichier partagé

Spécifie une commande externe que le SDK ou l'outil exécute en votre nom pour générer ou récupérer les informations d'authentification à utiliser. Le paramètre spécifie le nom d'un fichier program/command que le SDK invoquera. Lorsque le SDK appelle le processus, il attend que celui-ci y écrive des données JSON. `stdout` Le fournisseur personnalisé doit renvoyer

les informations dans un format spécifique. Ces informations contiennent les informations d'identification que le SDK ou l'outil peut utiliser pour vous authentifier.

> **Note**
>
> Le fournisseur d'informations d'identification du processus fait partie duComprendre la chaîne des fournisseurs d'informations d'identification. Cependant, le fournisseur d'identifiants de processus n'est vérifié qu'après plusieurs autres fournisseurs de cette série. Par conséquent, si vous souhaitez que votre programme utilise les informations d'identification de ce fournisseur, vous devez supprimer les autres fournisseurs d'informations d'identification valides de votre configuration ou utiliser un profil différent. Sinon, au lieu de vous fier à la chaîne de fournisseurs d'informations d'identification pour découvrir automatiquement quel fournisseur renvoie des informations d'identification valides, spécifiez l'utilisation du fournisseur d'informations d'identification de processus dans le code. Vous pouvez spécifier les sources d'informations d'identification directement lorsque vous créez des clients de service.

Spécification du chemin d'accès au programme d'identification

La valeur du paramètre est une chaîne qui contient le chemin d'accès à un programme que le SDK ou l'outil de développement exécute en votre nom :

- Le chemin et le nom du fichier ne peuvent être composés que des caractères suivants : A-Z, a-z, 0-9, tiret (-), trait de soulignement (_), point (.), barre oblique (/), barre oblique inverse (\) et espace.
- Si le chemin d'accès ou le nom du fichier contient un espace, entourez le chemin d'accès complet et le nom du fichier de guillemets doubles (" ").
- Si un nom de paramètre ou une valeur de paramètre contient un espace, entourez cet élément de guillemets doubles (" "). Entourez uniquement le nom ou la valeur, pas la paire.
- N'incluez aucune variable d'environnement dans les chaînes. Par exemple, n'incluez pas `$HOME` ou`%USERPROFILE%`.
- Ne spécifiez pas le dossier de base sous la forme~. * Vous devez spécifier le chemin complet ou le nom du fichier de base. S'il existe un nom de fichier de base, le système tente de trouver le programme dans les dossiers spécifiés par la variable d'`PATH`environnement. Le chemin varie en fonction du système d'exploitation :

L'exemple suivant montre comment définir credential_process dans le `config` fichier partagé sous Linux/macOS.

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

L'exemple suivant montre comment définir credential_process dans le `config` fichier partagé sous Windows.

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

- Peut être spécifié dans un profil dédié :

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

Sortie valide du programme d'identification

Le SDK exécute la commande comme indiqué dans le profil, puis lit les données à partir du flux de sortie standard. La commande que vous spécifiez, qu'il s'agisse d'un script ou d'un programme binaire, doit générer une sortie JSON `STDOUT` correspondant à la syntaxe suivante.

```
{
    "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**
>
> À la date de cette publication, la clé `Version` doit être définie sur `1`. Cela pourrait augmenter avec le temps à mesure que la structure évolue.

La `Expiration` clé est un RFC3339 horodatage formaté. Si la `Expiration` clé n'est pas présente dans la sortie de l'outil, le SDK suppose que les informations d'identification sont des informations d'identification à long terme qui ne sont pas actualisées. Dans le cas contraire, les informations d'identification sont considérées comme des informations d'identification temporaires et elles sont automatiquement actualisées en réexécutant la `credential_process` commande avant leur expiration.

> **Note**
>
> Le SDK ne met pas en cache les informations d'identification des processus externes de la même manière qu'il assume les informations d'identification des rôles. Si la mise en cache est obligatoire, vous devez la mettre en œuvre dans le processus externe.

Le processus externe peut renvoyer un code de retour non nul pour indiquer qu'une erreur s'est produite lors de la récupération des informations d'identification.

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
AWS CLI v2	Oui	
SDK pour C++	Oui	
SDK pour Go V2 (1.x)	Oui	
SDK pour Go 1.x (V1)	Oui	Pour utiliser les paramètres des `config` fichiers partagés, vous devez activer le chargement à partir du fichier de configuration ; voir Sessions.
SDK pour Java 2.x	Oui	

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
SDK pour Java 1.x	Oui	
SDK pour 3.x JavaScript	Oui	
SDK pour 2.x JavaScript	Oui	
SDK pour Kotlin	Oui	
SDK pour .NET 4.x	Oui	
SDK pour .NET 3.x	Oui	
SDK pour PHP 3.x	Oui	
SDK pour Python (Boto3)	Oui	
SDK pour Ruby 3.x	Oui	
SDK pour Rust	Oui	
SDK pour Swift	Oui	
Outils pour PowerShell V5	Oui	
Outils pour PowerShell V4	Oui	

AWS SDKs et fonctionnalités standardisées des outils

De nombreuses fonctionnalités ont été standardisées selon des valeurs par défaut cohérentes et pour fonctionner de la même manière pour de nombreuses SDKs fonctionnalités. Cette cohérence augmente la productivité et la clarté lors du codage sur plusieurs SDKs. Tous les paramètres peuvent être remplacés dans le code. Consultez l'API de votre SDK spécifique pour plus de détails.

> **Important**
>
> Tous ne prennent pas SDKs en charge toutes les fonctionnalités, ni même tous les aspects d'une fonctionnalité.

Rubriques

- Points de terminaison basés sur un compte
- ID d'application
- Métadonnées d'instance Amazon EC2.
- Points d’accès Amazon S3
- Amazon S3 Multi-Region Access Points
- Authentification de session S3 Express One Zone
- Schéma d'authentification
- Région AWS
- AWS STS Points de terminaison régionaux
- Protections de l'intégrité des données pour Amazon S3
- Points de terminaison à double pile et FIPS
- Découverte du points de terminaison
- Paramètres de configuration généraux
- Injection du préfixe hôte
- Client IMDS
- Comportement de nouvelle tentative
- Compression des demandes
- Points de terminaison spécifiques au service
- Paramètres de configuration intelligents par défaut

Points de terminaison basés sur un compte

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

Les terminaux basés sur des comptes contribuent à garantir des performances et une évolutivité élevées en utilisant votre Compte AWS identifiant pour acheminer les demandes de services prenant en charge cette fonctionnalité. Lorsque vous utilisez un AWS SDK et un service qui prennent en charge les points de terminaison basés sur des comptes, le client du SDK construit et utilise un point de terminaison basé sur un compte plutôt qu'un point de terminaison régional. Si l'identifiant du compte n'est pas visible pour le client du SDK, celui-ci utilisera le point de terminaison régional. Les points de terminaison basés sur un compte prennent la forme`https://<account-id>.ddb.<region>.amazonaws.com`, où *`<account-id>`* et *`<region>`* sont votre Compte AWS identifiant et. Région AWS

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`aws_account_id`- réglage AWS **`config`** du fichier partagé, **`AWS_ACCOUNT_ID`**- variable d'environnement, **`aws.accountId`**- Propriété du système JVM : uniquement Java/Kotlin

La Compte AWS pièce d'identité. Utilisé pour le routage des terminaux basé sur un compte. Un Compte AWS identifiant a un format tel que 111122223333.

Le routage des points de terminaison basé sur le compte améliore les performances des demandes pour certains services.

`account_id_endpoint_mode`- réglage AWS **`config`** du fichier partagé, **`AWS_ACCOUNT_ID_ENDPOINT_MODE`**- variable d'environnement, **`aws.accountIdEndpointMode`**- Propriété du système JVM : uniquement Java/Kotlin

Ce paramètre est utilisé pour désactiver le routage des points de terminaison basé sur le compte si nécessaire et pour contourner les règles basées sur le compte.

Valeur par défaut : `preferred`

Valeurs valides:

- **`preferred`**— Le point de terminaison doit inclure l'identifiant du compte s'il est disponible.
- **`disabled`** : un point de terminaison résolu n'inclut pas d'ID de compte.
- **`required`** : le point de terminaison doit inclure un ID de compte. Si l'ID de compte n'est pas disponible, le kit SDK génère une erreur.

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pris en char	Publié en version SDK	Remarques ou informations supplémentaires
AWS CLI v2	Oui	2.25.0	
AWS CLI v1	Oui	1,38,0	
SDK pour C++	Non		
SDK pour Go V2 (1.x)	Oui	v1.35.0	
SDK pour Go 1.x (V1)	Non		
SDK pour Java 2.x	Oui	v2.28.4	
SDK pour Java 1.x	Oui	v1.12.771	
SDK pour 3.x JavaScript	Oui	v3.656.0	
SDK pour 2.x JavaScript	Non		
SDK pour Kotlin	Oui	v1.3.37	

Kit SDK	Pris en char	Publié en version SDK	Remarques ou informations supplémentaires
SDK pour .NET 4.x	Oui	4.0.0	
SDK pour .NET 3.x	Non		
SDK pour PHP 3.x	Oui	v3.318.0	
SDK pour Python (Boto3)	Oui	1.37.0	
SDK pour Ruby 3.x	Oui	v1.123.0	
SDK pour Rust	Oui	publique-2025-04-24	
SDK pour Swift	Oui	1.2.0	
Outils pour PowerShell V5	Non		
Outils pour PowerShell V4	Non		

ID d'application

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau « Support par AWS les SDK et les outils » qui suit, consultezComprendre les pages de paramètres de ce guide.

Un seul Compte AWS peut être utilisé par plusieurs applications clients pour passer des appels à Services AWS. L'ID d'application permet aux clients d'identifier quelle application source a effectué

un ensemble d'appels à l'aide d'un Compte AWS. AWS Les SDK et les services n'utilisent ni n'interprètent cette valeur autrement que pour la réintégrer dans les communications avec les clients. Par exemple, cette valeur peut être incluse dans les e-mails opérationnels ou dans le Tableau de bord AWS Health pour identifier de manière unique laquelle de vos applications est associée à la notification.

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`sdk_ua_app_id`- réglage AWS **`config`** du fichier partagé, **`AWS_SDK_UA_APP_ID`**- variable d'environnement, **`sdk.ua.appId`**- Propriété du système JVM : uniquement Java/Kotlin

Ce paramètre est une chaîne unique que vous attribuez à votre application pour identifier les applications auxquelles une Compte AWS application donnée appelle AWS.

Valeur par défaut : `None`

Valeurs valides : chaîne d'une longueur maximale de 50. Les lettres, les chiffres et les caractères spéciaux suivants sont autorisés : `! # $%,&,',*,+,-,.,^,_,` ,|,~`.

Exemple de définition de cette valeur dans le `config` fichier :

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

Linux/macOS exemple de définition de variables d'environnement via la ligne de commande :

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

Exemple Windows de définition de variables d'environnement via la ligne de commande :

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

Si vous incluez des symboles qui ont une signification particulière pour le shell utilisé, évitez la valeur le cas échéant.

Support par AWS Kits SDK et outils

Les SDK suivants prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
AWS CLI v2	Oui	
SDK pour C++	Oui	`config`le fichier partagé n'est pas pris en charge.
SDK pour Go V2 (1.x)	Oui	
SDK pour Go 1.x (V1)	Non	
SDK pour Java 2.x	Oui	
SDK pour Java 1.x	Non	
SDK pour 3.x JavaScript	Oui	
SDK pour 2.x JavaScript	Non	
SDK pour Kotlin	Oui	La propriété du système JVM est`aws.userAgentAppId` .
SDK pour .NET 4.x	Oui	
SDK pour .NET 3.x	Oui	
SDK pour PHP 3.x	Oui	
SDK pour Python (Boto3)	Oui	
SDK pour Ruby 3.x	Oui	
SDK pour Rust	Oui	
SDK pour Swift	Oui	

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
Outils pour PowerShell V5	Oui	
Outils pour PowerShell V4	Oui	

Métadonnées d'instance Amazon EC2.

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

Amazon EC2 fournit un service sur les instances appelé Instance Metadata Service (IMDS). Pour en savoir plus sur ce service, consultez la section Utilisation des métadonnées d'instance dans le guide de l'utilisateur Amazon EC2. Lorsque vous tentez de récupérer des informations d'identification sur une instance Amazon EC2 configurée avec un rôle IAM, la connexion au service de métadonnées de l'instance est ajustable.

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`metadata_service_num_attempts`- réglage AWS **`config`** du fichier partagé, **`AWS_METADATA_SERVICE_NUM_ATTEMPTS`**- variable d'environnement

Ce paramètre indique le nombre total de tentatives à effectuer avant d'abandonner lors de la tentative de récupération de données à partir du service de métadonnées d'instance.

Valeur par défaut : 1

Valeurs valides : nombre supérieur ou égal à 1.

`metadata_service_timeout`- réglage AWS **`config`** du fichier partagé, **`AWS_METADATA_SERVICE_TIMEOUT`**- variable d'environnement

Spécifie le nombre de secondes avant l'expiration du délai lorsque vous tentez de récupérer des données à partir du service de métadonnées d'instance.

Valeur par défaut : 1

Valeurs valides : nombre supérieur ou égal à 1.

Exemple de définition de ces valeurs dans le `config` fichier :

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

Exemple Linux/macOS de définition de variables d'environnement via la ligne de commande :

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

Exemple Windows de définition de variables d'environnement via la ligne de commande :

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

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
AWS CLI v2	Oui	
SDK pour C++	Nor	
SDK pour Go V2 (1.x)	Nor	
SDK pour Go 1.x (V1)	Nor	

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
SDK pour Java 2.x	Partie	Seule la clause `AWS_METADATA_SERVICE_TIMEOUT` est prise en charge.
SDK pour Java 1.x	Partie	Seule la clause `AWS_METADATA_SERVICE_TIMEOUT` est prise en charge.
SDK pour 3.x JavaScript	Nor	
SDK pour 2.x JavaScript	Nor	
SDK pour Kotlin	Nor	
SDK pour .NET 4.x	Nor	
SDK pour .NET 3.x	Nor	
SDK pour PHP 3.x	Oui	
SDK pour Python (Boto3)	Oui	
SDK pour Ruby 3.x	Nor	
SDK pour Rust	Nor	
SDK pour Swift	Nor	
Outils pour PowerShell V5	Nor	
Outils pour PowerShell V4	Nor	

Points d’accès Amazon S3

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

Le service Amazon S3 fournit des points d'accès comme autre moyen d'interagir avec les compartiments Amazon S3. Les points d'accès ont des politiques et des configurations uniques qui peuvent leur être appliquées plutôt que directement au bucket. Avec AWS SDKs, vous pouvez utiliser le point d'accès Amazon Resource Names (ARNs) dans le champ du bucket pour les opérations d'API au lieu de spécifier explicitement le nom du bucket. Ils sont utilisés pour des opérations spécifiques telles que l'utilisation d'un point d'accès ARN `GetObject`pour récupérer un objet dans un bucket, ou l'utilisation d'un ARN de point d'accès avec `PutObject`pour ajouter un objet à un bucket.

Pour en savoir plus sur les points d'accès Amazon S3 et ARNs consultez la section Utilisation des points d'accès dans le guide de l'utilisateur Amazon S3.

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`s3_use_arn_region`- réglage AWS **`config`** du fichier partagé, **`AWS_S3_USE_ARN_REGION`**- variable d'environnement, **`aws.s3UseArnRegion`**- Propriété du système JVM : uniquement Java/Kotlin , Pour configurer la valeur directement dans le code, consultez directement votre SDK spécifique.

Ce paramètre contrôle si le SDK utilise l'ARN du point d'accès Région AWS pour créer le point de terminaison régional pour la demande. Le SDK confirme que l'ARN Région AWS est servi par la même AWS partition que celle configurée par le client afin Région AWS d'empêcher les appels entre partitions susceptibles d'échouer. S'il est défini par multiplicateur, le paramètre configuré par le code est prioritaire, suivi du paramètre de variable d'environnement.

Valeur par défaut : `false`

Valeurs valides:

- **`true`**— Le SDK utilise les ARN Région AWS lors de la construction du point de terminaison plutôt que celui configuré Région AWS par le client. Exception : si la configuration du client

Région AWS est une norme FIPS Région AWS, elle doit correspondre à l' Région AWS ARN. Si vous ne le faites pas, une erreur se produit.

- **`false`**— Le SDK utilise la configuration du client Région AWS lors de la construction du point de terminaison.

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
AWS CLI v2	Oui	
SDK pour C++	Oui	
SDK pour Go V2 (1.x)	Oui	
SDK pour Go 1.x (V1)	Oui	Pour utiliser les paramètres des `config` fichiers partagés, vous devez activer le chargement à partir du fichier de configuration ; voir Sessions.
SDK pour Java 2.x	Oui	
SDK pour Java 1.x	Oui	La propriété du système JVM n'est pas prise en charge.
SDK pour 3.x JavaScript	Oui	
SDK pour 2.x JavaScript	Oui	
SDK pour Kotlin	Oui	
SDK pour .NET 4.x	Oui	
SDK pour .NET 3.x	Oui	Ne suit pas la priorité standard ; la valeur `config` du fichier partagé a priorité sur la variable d'environnement.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
SDK pour PHP 3.x	Oui	
SDK pour Python (Boto3)	Oui	
SDK pour Ruby 3.x	Oui	
SDK pour Rust	Nor	
SDK pour Swift	Nor	
Outils pour PowerShell V5	Oui	Ne suit pas la priorité standard ; la valeur `config` du fichier partagé a priorité sur la variable d'environnement.
Outils pour PowerShell V4	Oui	Ne suit pas la priorité standard ; la valeur `config` du fichier partagé a priorité sur la variable d'environnement.

Amazon S3 Multi-Region Access Points

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

Les points d'accès multirégionaux Amazon S3 fournissent un point de terminaison global que les applications peuvent utiliser pour traiter les demandes provenant de compartiments Amazon S3 situés dans plusieurs compartiments. Régions AWS Vous pouvez utiliser des points d'accès multirégionaux pour créer des applications multirégionales avec la même architecture que celle utilisée dans une seule région, puis exécuter ces applications n'importe où dans le monde.

Pour en savoir plus sur les points d'accès multirégionaux, consultez la section Points d'accès multirégionaux dans Amazon S3 dans le guide de l'utilisateur d'Amazon S3.

Pour en savoir plus sur les points d'accès multirégionaux Amazon Resource Names (ARNs), consultez la section Faire des demandes à l'aide d'un point d'accès multirégional dans le guide de l'utilisateur Amazon S3.

Pour en savoir plus sur la création de points d'accès multirégionaux, consultez la section Gestion des points d'accès multirégionaux dans le guide de l'utilisateur Amazon S3.

L'algorithme SigV4A est l'implémentation de signature utilisée pour signer les demandes régionales globales. Cet algorithme est obtenu par le SDK via une dépendance à l'égard duAWS bibliothèques CRT (Common Runtime).

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`s3_disable_multiregion_access_points`- réglage AWS **`config`** du fichier partagé, **`AWS_S3_DISABLE_MULTIREGION_ACCESS_POINTS`**- variable d'environnement, **`aws.s3DisableMultiRegionAccessPoints`**- Propriété du système JVM : uniquement Java/ Kotlin , Pour configurer la valeur directement dans le code, consultez directement votre SDK spécifique.

Ce paramètre détermine si le SDK est susceptible de tenter des requêtes interrégionales. Si le paramètre multiplicateur est défini, le paramètre configuré par le code est prioritaire, suivi du paramètre de variable d'environnement.

Valeur par défaut : `false`

Valeurs valides:

- **`true`**— Arrête l'utilisation des demandes interrégionales.
- **`false`**— Active les demandes interrégionales à l'aide de points d'accès multirégionaux.

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
AWS CLI v2	Oui	
SDK pour C++	Oui	
SDK pour Go V2 (1.x)	Oui	
SDK pour Go 1.x (V1)	Nor	
SDK pour Java 2.x	Oui	
SDK pour Java 1.x	Nor	
SDK pour 3.x JavaScript	Oui	
SDK pour 2.x JavaScript	Nor	
SDK pour Kotlin	Oui	
SDK pour .NET 4.x	Oui	
SDK pour .NET 3.x	Oui	
SDK pour PHP 3.x	Oui	
SDK pour Python (Boto3)	Oui	
SDK pour Ruby 3.x	Oui	
SDK pour Rust	Oui	
SDK pour Swift	Nor	
Outils pour PowerShell V5	Oui	
Outils pour PowerShell V4	Oui	

Authentification de session S3 Express One Zone

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

S3 Express One Zone est la classe de stockage hautes performances d'Amazon S3 qui fournit une latence d'un chiffre en millisecondes pour les données fréquemment consultées. Lorsque vous utilisez des compartiments S3 Express One Zone, AWS SDKs les outils utilisent automatiquement une authentification basée sur les sessions, optimisée pour l'autorisation à faible latence des demandes de données. Vous utilisez des jetons de session avec des opérations zonales (au niveau de l'objet) pour répartir la latence associée à l'autorisation sur un certain nombre de demandes au cours d'une session, réduisant ainsi la charge d'authentification et améliorant les performances globales des demandes.

Les compartiments S3 Express One Zone utilisent un format de dénomination spécifique qui inclut l'ID de zone de disponibilité, tel que`bucket-name--usw2-az1--x-s3`. Lorsque le SDK détecte ce modèle de dénomination, il achemine automatiquement les demandes vers les points de terminaison S3 Express One Zone appropriés et applique le flux d'authentification optimisé. L'authentification de session crée des informations d'identification temporaires spécifiques au compartiment qui fournissent un accès à faible latence à votre compartiment et sont mises en cache et actualisées automatiquement par le SDK. Consultez S3 Express One Zone dans le guide de l'utilisateur Amazon S3 pour en savoir plus.

Par défaut, l'authentification de session est activée pour les compartiments S3 Express One Zone.

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`s3_disable_express_session_auth`- réglage AWS **`config`** du fichier partagé, **`AWS_S3_DISABLE_EXPRESS_SESSION_AUTH`**- variable d'environnement, **`aws.disableS3ExpressAuth`**- Propriété du système JVM : uniquement Java/Kotlin

Contrôle si l'authentification de session S3 Express One Zone est désactivée. Lorsqu'il est défini sur`true`, le SDK utilise l'authentification Sigv4 standard pour les compartiments S3 Express One Zone au lieu de l'authentification de session.

Valeur par défaut : `false`

Valeurs valides:

- **`true`**— Désactive l'authentification de session S3 Express One Zone.
- **`false`**— Activez l'authentification de session S3 Express One Zone.

Exemple de définition de cette valeur dans le `config` fichier :

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

Exemple Linux/macOS de définition de variables d'environnement via la ligne de commande :

```
export AWS_S3_DISABLE_EXPRESS_SESSION_AUTH=true
```

Exemple Windows de définition de variables d'environnement via la ligne de commande :

```
setx AWS_S3_DISABLE_EXPRESS_SESSION_AUTH true
```

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pris en charg	Remarques ou informations supplémentaires
AWS CLI v2	Oui	
AWS CLI v1	Oui	
SDK pour C++	Oui	
SDK pour Go V2 (1.x)	Oui	

Kit SDK	Pris en charg	Remarques ou informations supplémentaires
SDK pour Go 1.x (V1)	Non	Pour utiliser les paramètres des `config` fichiers partagés, vous devez activer le chargement à partir du fichier de configuration ; voir Sessions.
SDK pour Java 2.x	Oui	
SDK pour Java 1.x	Non	
SDK pour 3.x JavaScript	Oui	
SDK pour 2.x JavaScript	Non	
SDK pour Kotlin	Oui	La propriété du système JVM est`aws.s3DisableExpressSessionAuth` .
SDK pour .NET 4.x	Oui	
SDK pour .NET 3.x	Oui	
SDK pour PHP 3.x	Oui	
SDK pour Python (Boto3)	Oui	
SDK pour Ruby 3.x	Oui	
SDK pour Rust	Oui	
SDK pour Swift	Oui	
Outils pour PowerShell V5	Oui	

Kit SDK	Pris en charg	Remarques ou informations supplémentaires
Outils pour PowerShell V4	Oui	

Schéma d'authentification

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

AWS les services prennent en charge plusieurs schémas d'authentification, tels que AWS Signature Version 4 (SigV4) et AWS Signature Version 4a (SigV4a). Par défaut, SDKs sélectionnez les schémas d'authentification en fonction des définitions des modèles de service et priorisez les schémas offrant la meilleure compatibilité. Cependant, vous pouvez configurer votre schéma d'authentification préféré pour l'optimiser en fonction de besoins spécifiques.

Contrairement à SigV4, les demandes signées avec SigV4a sont valides en plusieurs exemplaires. Régions AWS Le SigV4a améliore la disponibilité grâce à la signature des demandes entre régions, ce qui permet le basculement automatique vers les régions de sauvegarde en cas de perturbations régionales. Cela est particulièrement avantageux pour les services internationaux tels Gestion des identités et des accès AWS qu'Amazon CloudFront.

Pour plus d'informations sur ces deux schémas d'authentification, consultez la version 4 de AWS Signature pour les demandes d'API dans le guide de l'utilisateur IAM.

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`auth_scheme_preference`- réglage AWS **`config`** du fichier partagé, **`AWS_AUTH_SCHEME_PREFERENCE`**- variable d'environnement, **`aws.authSchemePreference`**- Propriété du système JVM : uniquement Java/Kotlin

Spécifie une liste de schémas d'authentification préférés séparés par des virgules par ordre de priorité. Lorsqu'un service prend en charge plusieurs schémas d'authentification, le SDK tente d'utiliser les schémas de cette liste dans l'ordre indiqué, en revenant au comportement par défaut si aucun des schémas préférés n'est disponible.

Valeur par défaut : Aucune.

Valeurs valides : liste séparée par des virgules d'un ou de plusieurs des éléments suivants :

- **`sigv4`**— Signature Version 4 (performance la plus rapide, région unique)
- **`sigv4a`**— Signature Version 4a (disponibilité améliorée, prise en charge interrégionale, performances de signature plus lentes que celles de SigV4)
- **`httpBearerAuth`**— Authentification par jeton HTTP Bearer

Les espaces et les tabulations entre les noms de schéma sont ignorés.

Exemple de définition de cette valeur dans le `config` fichier pour préférer SigV4a :

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

`sigv4a_signing_region_set`- réglage AWS **`config`** du fichier partagé, **`AWS_SIGV4A_SIGNING_REGION_SET`**- variable d'environnement

Spécifie une liste séparée par des virgules Régions AWS pour la signature multirégionale SIGv4a. Elle est utilisée comme région par défaut définie pour la demande si SigV4a est le schéma d'authentification sélectionné.

Valeur par défaut : déterminée par la demande.

Valeurs valides : liste séparée par des virgules de. Régions AWS Les espaces et les tabulations entre les régions ne sont pas pris en compte.

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
AWS CLI v2	Oui	
SDK pour C++	Oui	
SDK pour Go V2 (1.x)	Oui	
SDK pour Go 1.x (V1)	Non	
SDK pour Java 2.x	Oui	
SDK pour Java 1.x	Non	
SDK pour 3.x JavaScript	Oui	
SDK pour 2.x JavaScript	Non	
SDK pour Kotlin	Oui	
SDK pour .NET 4.x	Oui	
SDK pour .NET 3.x	Non	
SDK pour PHP 3.x	Oui	
SDK pour Python (Boto3)	Oui	
SDK pour Ruby 3.x	Oui	
SDK pour Rust	Oui	
SDK pour Swift	Oui	

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
Outils pour PowerShell V5	Oui	
Outils pour PowerShell V4	Non	

Région AWS

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

Régions AWS sont un concept important à comprendre lorsque vous travaillez avec Services AWS.

Avec Régions AWS, vous pouvez accéder à Services AWS ceux qui résident physiquement dans une zone géographique spécifique. Cela peut être utile pour que vos données et applications fonctionnent à proximité de l'endroit où vous et vos utilisateurs y accéderez. Les régions fournissent une tolérance aux pannes, une stabilité et une résilience, et peuvent également réduire la latence. Avec les régions, vous pouvez créer des ressources redondantes qui restent disponibles et qui ne sont pas affectées par une panne régionale.

La plupart des Service AWS demandes sont associées à une région géographique particulière. Les ressources que vous créez dans une région n'existent dans aucune autre région, sauf si vous utilisez explicitement une fonctionnalité de réplication proposée par un Service AWS. Par exemple, Amazon S3 et Amazon EC2 prennent en charge la réplication entre régions. Certains services, tels que IAM, ne disposent pas de ressources régionales.

Références générales AWSIl contient des informations sur les points suivants :

- Pour comprendre la relation entre les régions et les points de terminaison, et pour consulter la liste des points de terminaison régionaux existants, voir Points de terminaison de AWS service.
- Pour consulter la liste actuelle de toutes les régions et points de terminaison pris en charge pour chacune d'entre elles Service AWS, voir Points de terminaison et quotas de service.

Création de clients de service

Pour y accéder par programmation Services AWS, SDKs utilisez un client class/object pour chacun d'entre eux. Service AWS Si votre application doit accéder à Amazon EC2, par exemple, elle créera un objet client Amazon EC2 pour établir une interface avec ce service.

Si aucune région n'est explicitement spécifiée pour le client dans le code lui-même, le client utilise par défaut la région définie par le biais du `region` paramètre suivant. Cependant, la région active d'un client peut être définie explicitement pour n'importe quel objet client individuel. La définition de la région de cette manière a priorité sur tout paramètre global pour ce client de service en particulier. La région alternative est spécifiée lors de l'instanciation de ce client, en fonction de votre SDK (consultez votre guide SDK spécifique ou la base de code de votre SDK).

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`region`- réglage AWS **`config`** du fichier partagé, **`AWS_REGION`**- variable d'environnement, **`aws.region`**- Propriété du système JVM : uniquement Java/Kotlin

Spécifie la valeur par défaut Région AWS à utiliser pour les AWS demandes. Cette région est utilisée pour les demandes de service du SDK qui ne sont pas fournies avec une région spécifique à utiliser.

Valeur par défaut : Aucune. Vous devez spécifier cette valeur de manière explicite.

Valeurs valides:

- N'importe quel code de région disponible pour le service choisi, tel qu'indiqué dans les points de terminaison du AWS service dans la référence AWS générale. Par exemple, la valeur `us-east-1` définit le point de terminaison sur l'est des Région AWS États-Unis (Virginie du Nord).
- `aws-global`spécifie le point de terminaison global pour les services qui prennent en charge un point de terminaison mondial distinct en plus des points de terminaison régionaux, tels que AWS Security Token Service (AWS STS) et Amazon Simple Storage Service (Amazon S3).

Exemple de définition de cette valeur dans le `config` fichier :

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

Exemple Linux/macOS de définition de variables d'environnement via la ligne de commande :

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

Exemple Windows de définition de variables d'environnement via la ligne de commande :

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

La plupart SDKs disposent d'un objet de « configuration » qui permet de définir la région par défaut à partir du code de l'application. Pour plus de détails, consultez le guide du développeur de votre AWS SDK spécifique.

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
AWS CLI v2	Oui	AWS CLI v2 utilise n'importe quelle valeur in `AWS_REGION` avant toute valeur in `AWS_DEFAULT_REGION` (les deux variables sont cochées).
AWS CLI v1	Oui	AWS CLI v1 utilise une variable d'environnement nommée `AWS_DEFAULT_REGION` à cette fin.
SDK pour C++	Oui	
SDK pour Go V2 (1.x)	Oui	
SDK pour Go 1.x (V1)	Oui	Pour utiliser les paramètres des `config` fichiers partagés, vous devez activer le chargement à partir du fichier de configuration ; voir Sessions.
SDK pour Java 2.x	Oui	
SDK pour Java 1.x	Oui	

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
SDK pour 3.x JavaScript	Oui	
SDK pour 2.x JavaScript	Oui	
SDK pour Kotlin	Oui	
SDK pour .NET 4.x	Oui	
SDK pour .NET 3.x	Oui	
SDK pour PHP 3.x	Oui	
SDK pour Python (Boto3)	Oui	Ce SDK utilise une variable d'environnement nommée `AWS_DEFAULT_REGION` à cette fin.
SDK pour Ruby 3.x	Oui	
SDK pour Rust	Oui	
SDK pour Swift	Oui	
Outils pour PowerShell V5	Oui	
Outils pour PowerShell V4	Oui	

AWS STS Points de terminaison régionaux

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

AWS Security Token Service (AWS STS) est disponible à la fois en tant que service mondial et régional. Certains d'entre AWS SDKs eux CLIs utilisent le point de terminaison de service global (`https://sts.amazonaws.com`) par défaut, tandis que d'autres utilisent les points de terminaison de service régionaux (`https://sts.{region_identifier}.{partition_domain}`). Dans les régions activées par défaut, les demandes adressées au point de terminaison AWS STS global sont automatiquement traitées dans la même région d'où provient la demande. Dans les régions optionnelles, les demandes adressées au point de terminaison AWS STS mondial sont traitées par une seule entité Région AWS, celle de l'est des États-Unis (Virginie du Nord). Pour plus d'informations sur les AWS STS points de terminaison, consultez la section Points de terminaison dans la référence d'AWS Security Token Service API ou Gérer AWS STS dans un Région AWS dans le guide de l'Gestion des identités et des accès AWS utilisateur.

Il est recommandé d' AWS utiliser des points de terminaison régionaux dans la mesure du possible et de configurer votreRégion AWS. Les clients résidant dans des partitions autres que commerciales doivent utiliser des points de terminaison régionaux. Tous SDKs les outils ne prennent pas en charge ce paramètre, mais ils ont tous défini un comportement en ce qui concerne les points de terminaison mondiaux et régionaux. Consultez la section suivante pour plus d'informations.

> **Note**
>
> AWS a apporté des modifications au AWS Security Token Service point de terminaison global (`https://sts.amazonaws.com`) dans les régions activées par défaut afin d'améliorer sa résilience et ses performances.AWS STS AWS STS les demandes adressées au point de terminaison global sont automatiquement traitées au même Région AWS titre que vos charges de travail. Ces modifications ne seront pas déployées vers des régions d'adhésion. Nous vous recommandons d'utiliser les points de terminaison AWS STS régionaux appropriés. Pour plus d'informations, voir les modifications AWS STS globales des terminaux dans le Guide de Gestion des identités et des accès AWS l'utilisateur.

Pour SDKs les outils qui prennent en charge ce paramètre, les clients peuvent configurer la fonctionnalité à l'aide des méthodes suivantes :

`sts_regional_endpoints`- réglage AWS **`config`** du fichier partagé,
`AWS_STS_REGIONAL_ENDPOINTS`- variable d'environnement

Ce paramètre indique comment le SDK ou l'outil détermine le Service AWS point de terminaison qu'il utilise pour communiquer avec le AWS Security Token Service (AWS STS).

Valeur par défaut :`regional`, voir les exceptions dans le tableau suivant.

> **Note**
>
> Toutes les nouvelles versions majeures du SDK publiées après juillet 2022 seront par défaut définies sur`regional`. Les nouvelles versions majeures du SDK peuvent supprimer ce paramètre et utiliser `regional` le comportement. Pour réduire l'impact futur de cette modification, nous vous recommandons de commencer à l'utiliser `regional` dans votre application dès que possible.

Valeurs valides : (Valeur recommandée :*`regional`*)

- **`legacy`**— Utilise le point de AWS STS terminaison global,`sts.amazonaws.com`.
- **`regional`**— Le SDK ou l'outil utilise toujours le AWS STS point de terminaison de la région actuellement configurée. Par exemple, si le client est configuré pour être utiliséus-west-2, tous les appels AWS STS sont effectués vers le point de terminaison régionals`ts.us-west-2.amazonaws.com`, plutôt que vers le point de `sts.amazonaws.com` terminaison global. Pour envoyer une demande au point de terminaison global lorsque ce paramètre est activé, vous pouvez définir la région sur `aws-global`.

Exemple de définition de ces valeurs dans le `config` fichier :

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

Exemple Linux/macOS de définition de variables d'environnement via la ligne de commande :

```
export AWS_STS_REGIONAL_ENDPOINTS=regional
```

Exemple Windows de définition de variables d'environnement via la ligne de commande :

```
setx AWS_STS_REGIONAL_ENDPOINTS regional
```

Support par AWS SDKs et outils

> **Note**
>
> Il est recommandé d' AWS utiliser des points de terminaison régionaux dans la mesure du possible et de configurer votreRégion AWS.

Le tableau ci-dessous récapitule, pour votre SDK ou outil :

- Paramètre pris en charge : si la variable de `config` fichier partagé et la variable d'environnement pour les points de terminaison régionaux STS sont prises en charge.
- Valeur du paramètre par défaut : valeur par défaut du paramètre s'il est pris en charge.
- Point de terminaison STS cible du client de service par défaut : quel point de terminaison par défaut est utilisé par le client même si le paramètre permettant de le modifier n'est pas disponible.
- Comportement de remplacement du client de service : ce que fait le SDK lorsqu'il est censé utiliser un point de terminaison régional mais qu'aucune région n'a été configurée. C'est le comportement, qu'il utilise un point de terminaison régional en raison d'une valeur par défaut ou parce qu'il `regional` a été sélectionné par le paramètre.

Le tableau utilise également les valeurs suivantes :

- Point de terminaison global :`https://sts.amazonaws.com`.
- Point de terminaison régional : basé sur la configuration Région AWS utilisée par votre application.
- **`us-east-1`**(Régional) : utilise le point de terminaison `us-east-1` régional mais avec des jetons de session plus longs que les demandes globales classiques.

Kit SDK	Valeur de réglage par défaut	Client de service par défaut cible STS Endpoint	Comportement de secours du client de service	Remarques ou informations supplémentaires
AWS CLI v2	N/A	Point de terminaison régional	Point de terminaison mondial	
AWS CLI v1	`legacy`	Point de terminaison mondial	Point de terminaison mondial	
SDK pour C++	N/A	Point de terminaison régional	`us-east-1` (Régional)	
SDK pour Go V2 (1.x)	N/A	Point de terminaison régional	Echec de la demande	
SDK pour Go 1.x (V1)	`legacy`	Point de terminaison mondial	Point de terminaison mondial	Pour utiliser les paramètres des `config` fichiers partagés, vous devez activer le chargement à partir du fichier de configuration ; voir Sessions.
SDK pour Java 2.x	N/A	Point de terminaison régional	Echec de la demande	Si aucune région n'est configurée, le `AssumeRole` et `AssumeRoleWithWebIdentity` utilisera le point de terminaison STS global.
SDK pour Java 1.x	`legacy`	Point de terminaison mondial	Point de terminaison mondial	

Kit SDK		Valeur de réglage par défaut	Client de service par défaut cible STS Endpoint	Comportement de secours du client de service	Remarques ou informations supplémentaires
SDK pour 3.x JavaScript	Nc	N/A	Point de terminaison régional	`us-east-1` (Régional)	
SDK pour 2.x JavaScript	Ou	`legacy`	Point de terminaison mondial	Point de terminaison mondial	
SDK pour Kotlin	Nc	N/A	Point de terminaison régional	Point de terminaison mondial	
SDK pour .NET 4.x	Nc	N/A	Point de terminaison régional	`us-east-1` (Régional)	
SDK pour .NET 3.x	Ou	`regional`	Point de terminaison mondial	Point de terminaison mondial	
SDK pour PHP 3.x	Ou	`regional`	Point de terminaison mondial	Echec de la demande	
SDK pour Python (Boto3)	Ou	`regional`	Point de terminaison mondial	Point de terminaison mondial	
SDK pour Ruby 3.x	Ou	`regional`	Point de terminaison régional	Echec de la demande	

Kit SDK	Valeur de réglage par défaut	Client de service par défaut cible STS Endpoint	Comportement de secours du client de service	Remarques ou informations supplémentaires
SDK pour Rust	N/A	Point de terminaison régional	Echec de la demande	
SDK pour Swift	N/A	Point de terminaison régional	Echec de la demande	
Outils pour PowerShell V5	`regional`	Point de terminaison mondial	Point de terminaison mondial	
Outils pour PowerShell V4	`regional`	Point de terminaison mondial	Point de terminaison mondial	

Protections de l'intégrité des données pour Amazon S3

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

Depuis un certain temps, AWS SDKs nous prenons en charge les contrôles d'intégrité des données lors du chargement de données vers ou du téléchargement de données depuis Amazon Simple Storage Service. Auparavant, ces vérifications étaient facultatives. Nous avons maintenant activé ces vérifications par défaut, en utilisant des algorithmes basés sur le CRC tels que NVME CRC32 . CRC64 Bien que chaque SDK ou outil possède un algorithme par défaut, vous pouvez en choisir un autre. Vous pouvez également continuer à fournir manuellement une somme de contrôle précalculée

pour les téléchargements si vous le souhaitez. Un comportement cohérent entre les chargements, les chargements partitionnés, les téléchargements et les modes de chiffrement simplifie les contrôles d'intégrité côté client.

Les dernières versions de notre outil AWS SDKs calculent AWS CLI automatiquement une somme de contrôle basée sur le contrôle de redondance cyclique (CRC) pour chaque téléchargement et l'envoie à Amazon S3. Amazon S3 calcule indépendamment une somme de contrôle côté serveur et la valide par rapport à la valeur fournie avant de stocker durablement l'objet et sa somme de contrôle dans les métadonnées de l'objet. En stockant la somme de contrôle dans les métadonnées à côté de l'objet, lorsque l'objet est téléchargé, la même somme de contrôle peut être automatiquement renvoyée et utilisée également pour valider les téléchargements. Vous pouvez également vérifier le checksum enregistré dans les métadonnées de l'objet à tout moment.

Pour en savoir plus sur les opérations de somme de contrôle, les téléchargements partitionnés ou la liste des algorithmes de somme de contrôle pris en charge, consultez la section Vérification de l'intégrité des objets dans Amazon S3 dans le guide de l'utilisateur d'Amazon Simple Storage Service.

Téléchargements partitionnés :

Amazon S3 fournit également aux développeurs des totaux d'objets complets cohérents pour les téléchargements partiels ou partitionnés.

Lorsque vous téléchargez des fichiers en plusieurs parties, les sommes de contrôle sont SDKs calculées pour chaque partie. Amazon S3 utilise ces sommes de contrôle pour vérifier l'intégrité de chaque partie par le biais de l'`UploadPart`API. En outre, Amazon S3 valide la taille complète du fichier et la somme de contrôle lorsque vous appelez l'`CompleteMultipartUpload`API.

Si votre SDK dispose d'un gestionnaire de transfert Amazon S3 pour faciliter les téléchargements partitionnés, les sommes de contrôle sont validées pour les parties à l'aide de l'algorithme par défaut spécifique au SDK figurant dans le tableau. Support par AWS SDKs et outils Vous pouvez opter pour une somme de contrôle complète de l'objet en réglant le paramètre `checksum_type` sur `FULL_OBJECT` ou en choisissant d'utiliser l'algorithme CRC64 NVME.

Si vous utilisez une ancienne version du SDK ou AWS CLI si vous :

Si votre application utilise une version antérieure à décembre 2024 du SDK ou de l'outil, Amazon S3 calcule toujours une somme de contrôle CRC64 NVME sur les nouveaux objets et la stocke dans les métadonnées des objets pour référence future. Vous pouvez ensuite comparer le CRC enregistré avec un CRC calculé de votre côté et vérifier que la transmission réseau était correcte. En outre, vous pouvez toujours étendre manuellement la protection de l'intégrité en fournissant vos propres

sommes de contrôle précalculées avec vos `UploadPart`demandes `PutObject`OR, ce qui est la technique standard pour résoudre ce problème dans les anciennes versions.

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`request_checksum_calculation`- réglage AWS **`config`** du fichier partagé, **`AWS_REQUEST_CHECKSUM_CALCULATION`**- variable d'environnement, **`aws.requestChecksumCalculation`**- Propriété du système JVM : uniquement Java/Kotlin

Par défaut, les utilisateurs sont autorisés à calculer la somme de contrôle d'une demande lors de l'envoi d'une demande. L'utilisateur peut choisir l'un des algorithmes de somme de contrôle disponibles dans le cadre de la création de la demande. Dans le cas contraire, un algorithme par défaut spécifique au SDK est utilisé. Consultez le Support par AWS SDKs et outils tableau de l'algorithme par défaut pour chaque SDK ou outil.

Valeur par défaut : `WHEN_SUPPORTED`

Valeurs valides:

- **`WHEN_SUPPORTED`**— La validation de la somme de contrôle est effectuée sur toutes les charges utiles des demandes lorsqu'elles sont prises en charge par l'opération d'API, telles que les transferts de données vers Amazon S3.
- **`WHEN_REQUIRED`**— La validation de la somme de contrôle est effectuée uniquement lorsque l'opération de l'API l'exige.

`response_checksum_validation`- réglage AWS **`config`** du fichier partagé, **`AWS_RESPONSE_CHECKSUM_VALIDATION`**- variable d'environnement, **`aws.responseChecksumValidation`**- Propriété du système JVM : uniquement Java/Kotlin

Par défaut, les utilisateurs sont autorisés à valider la somme de contrôle des réponses lorsqu'ils envoient une demande. Une somme de contrôle est calculée pour la charge utile de réponse et comparée à l'en-tête de réponse de somme de contrôle. Si la validation de la somme de contrôle échoue, une erreur est signalée à l'utilisateur lors de la lecture de la charge utile.

L'en-tête de réponse à la somme de contrôle indique également l'algorithme de la somme de contrôle. Le client Amazon S3 tente de valider les sommes de contrôle des réponses pour toutes les opérations d'API Amazon S3 qui prennent en charge les sommes de contrôle. Toutefois, si le SDK n'a pas implémenté l'algorithme de somme de contrôle spécifié, cette validation est ignorée.

Valeur par défaut : `WHEN_SUPPORTED`

Valeurs valides:

- **`WHEN_SUPPORTED`**— La validation de la somme de contrôle est effectuée sur toutes les charges utiles de réponse lorsqu'elles sont prises en charge par l'opération d'API, telles que les transferts de données vers Amazon S3.
- **`WHEN_REQUIRED`**— La validation de la somme de contrôle est effectuée uniquement si elle est prise en charge par l'opération d'API et si l'appelant a explicitement activé la somme de contrôle pour l'opération. Par exemple, lorsque l'`GetObject`API Amazon S3 est appelée et que le `ChecksumMode` paramètre est défini sur Activé.

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

> **Note**
>
> Dans le tableau suivant, « CRT » fait référence à AWS bibliothèques CRT (Common Runtime) et peut nécessiter l'ajout d'une dépendance supplémentaire à votre projet.

Kit SDK	Pris en charg	Algorithme de somme de contrôle par défaut	Algorithmes de somme de contrôle compatibles	Remarques ou informations supplémentaires
AWS CLI v2	Oui	CRC64NVME	CRC64NVME, CRC32C, CRC32, xxHash3 SHA1 SHA256, xxHash64, xxHash128, SHA512	Pour la AWS CLI version 1, l'algorithme par défaut et les algorithmes pris en charge seront identiques à ceux de Python (Boto3).
SDK pour C++	Oui	CRC64NVME	CRC64NVME, CRC32C, CRC32, xxHash3 SHA1 SHA256,	

Kit SDK	Pris en charg	Algorithme de somme de contrôle par défaut	Algorithmes de somme de contrôle compatibles	Remarques ou informations supplémentaires
			xxHash64, xxHash128, SHA512	
SDK pour Go V2 (1.x)	Oui	CRC32	CRC64NVME CRC32, CRC32 C, SHA1 SHA256	
SDK pour Go 1.x (V1)	Non			
SDK pour Java 2.x	Oui	CRC32	CRC32, CRC32 C SHA1, SHA256 Par CRT uniquement : CRC64 NVME, xxHash3, xxHash64, xxHash128, SHA512	
SDK pour Java 1.x	Non			
SDK pour 3.x JavaScript	Oui	CRC32	CRC32, CRC32 C SHA1, SHA256	
SDK pour 2.x JavaScript	Non			
SDK pour Kotlin	Oui	CRC32	CRC32, CRC32 C SHA1, SHA256	
SDK pour .NET 4.x	Oui	CRC32	CRC32, CRC32 C SHA1, SHA256, SHA512	

Kit SDK	Pris en charg	Algorithme de somme de contrôle par défaut	Algorithmes de somme de contrôle compatibles	Remarques ou informations supplémentaires
SDK pour .NET 3.x	Oui	CRC32	CRC32, CRC32 C SHA1, SHA256, SHA512	
SDK pour PHP 3.x	Oui	CRC32	CRC32, SHA1, SHA256 Via CRT uniquement : C CRC32	`awscrt`une extension est requise pour utiliser CRC32 C.
SDK pour Python (Boto3)	Oui	CRC32	CRC32, SHA1, SHA256 Par CRT uniquement : CRC32 C, CRC64 NVME, xxHash3, xxHash64, xxHash128, SHA512	
SDK pour Ruby 3.x	Oui	CRC32	CRC32, SHA1, SHA256 Par CRT uniquement : CRC64 NVME, C CRC32	
SDK pour Rust	Oui	CRC32	CRC64NVME CRC32, CRC32 C, SHA1 SHA256	
SDK pour Swift	Oui	CRC32	CRC64NVME CRC32, CRC32 C, SHA1 SHA256	La dépendance au CRT est requise pour tous les algorithmes.

Kit SDK	Pris en charg	Algorithme de somme de contrôle par défaut	Algorithmes de somme de contrôle compatibles	Remarques ou informations supplémentaires
Outils pour PowerShell V5	Oui	CRC32	CRC32, CRC32C, SHA1 SHA256, XXHash3, SHA512	
Outils pour PowerShell V4	Oui	CRC32	CRC32, CRC32 C SHA1, SHA256, SHA512	

Points de terminaison à double pile et FIPS

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`use_dualstack_endpoint`- réglage AWS **`config`** du fichier partagé, **`AWS_USE_DUALSTACK_ENDPOINT`**- variable d'environnement, **`aws.useDualstackEndpoint`**- Propriété du système JVM : uniquement Java/Kotlin

Active ou désactive l'envoi de demandes par le SDK aux points de terminaison à double pile. Pour en savoir plus sur les points de terminaison à double pile, qui prennent en charge à la fois le IPv6 trafic IPv4 et le trafic, consultez la section Utilisation des points de terminaison à double pile Amazon S3 dans le guide de l'utilisateur d'Amazon Simple Storage Service. Les points de terminaison à double pile sont disponibles pour certains services dans certaines régions.

Valeur par défaut : `false`

Valeurs valides:

- **`true`**— Le SDK ou l'outil tentera d'utiliser des points de terminaison à double pile pour effectuer des requêtes réseau. S'il n'existe pas de point de terminaison à double pile pour le service et/ou Région AWS la demande échouera.
- **`false`**— Le SDK ou l'outil n'utilisera pas de points de terminaison à double pile pour effectuer des requêtes réseau.

`use_fips_endpoint`- réglage AWS **`config`** du fichier partagé, **`AWS_USE_FIPS_ENDPOINT`**- variable d'environnement, **`aws.useFipsEndpoint`**- Propriété du système JVM : uniquement Java/Kotlin

Active ou désactive l'envoi de demandes par le SDK ou l'outil à des points de terminaison conformes à la norme FIPS. Les normes fédérales de traitement de l'information (FIPS) sont un ensemble d'exigences de sécurité du gouvernement américain relatives aux données et à leur cryptage. Les agences gouvernementales, les partenaires et ceux qui souhaitent faire affaire avec le gouvernement fédéral sont tenus de respecter les directives FIPS. Contrairement aux points de AWS terminaison standard, les points de terminaison FIPS utilisent une bibliothèque logicielle TLS validée par rapport à la norme FIPS 140. Si ce paramètre est activé et qu'il n'existe pas de point de terminaison FIPS pour le service dans votre ordinateur Région AWS, l' AWS appel risque d'échouer. Points de terminaison spécifiques au serviceet l'`--endpoint-url`option permettant de AWS Command Line Interface remplacer ce paramètre.

Pour en savoir plus sur les autres méthodes de spécification des points de terminaison FIPS par Région AWS, consultez la section Points de terminaison FIPS par service. Pour plus d'informations sur les points de terminaison du service Amazon Elastic Compute Cloud, consultez la section Points de terminaison à double pile (IPv4 et IPv6) du manuel Amazon EC2 API Reference.

Valeur par défaut : `false`

Valeurs valides:

- **`true`**— Le SDK ou l'outil enverra des demandes aux terminaux conformes à la norme FIPS.
- **`false`**— Le SDK ou l'outil n'enverra pas de demandes aux terminaux conformes à la norme FIPS.

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
AWS CLI v2	Oui	
SDK pour C++	Oui	
SDK pour Go V2 (1.x)	Oui	
SDK pour Go 1.x (V1)	Oui	Pour utiliser les paramètres des `config` fichiers partagés, vous devez activer le chargement à partir du fichier de configuration ; voir Sessions.
SDK pour Java 2.x	Oui	
SDK pour Java 1.x	Nor	
SDK pour 3.x JavaScript	Oui	
SDK pour 2.x JavaScript	Oui	
SDK pour Kotlin	Oui	
SDK pour .NET 4.x	Oui	
SDK pour .NET 3.x	Oui	
SDK pour PHP 3.x	Oui	
SDK pour Python (Boto3)	Oui	
SDK pour Ruby 3.x	Oui	
SDK pour Rust	Oui	

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
SDK pour Swift	Oui	
Outils pour PowerShell V5	Oui	
Outils pour PowerShell V4	Oui	

Découverte du points de terminaison

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

SDKs utilisez la découverte des points de terminaison pour accéder aux points de terminaison des services (URLs pour accéder à diverses ressources), tout en conservant la flexibilité nécessaire AWS pour les modifier URLs en fonction des besoins. De cette façon, votre code peut détecter automatiquement les nouveaux points de terminaison. Il n'existe aucun point de terminaison fixe pour certains services. Au lieu de cela, vous obtenez les points de terminaison disponibles pendant l'exécution en faisant d'abord une demande pour obtenir les points de terminaison. Après avoir récupéré les points de terminaison disponibles, le code utilise le point de terminaison pour accéder à d'autres opérations. Par exemple, pour Amazon Timestream, le SDK fait `DescribeEndpoints` une demande pour récupérer les points de terminaison disponibles, puis utilise ces points de terminaison pour effectuer des opérations spécifiques telles que ou. `CreateDatabase` `CreateTable`

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`endpoint_discovery_enabled`- réglage AWS **`config`** du fichier partagé, **`AWS_ENABLE_ENDPOINT_DISCOVERY`**- variable d'environnement, **`aws.endpointDiscoveryEnabled`**- Propriété du système JVM : uniquement Java/Kotlin , Pour configurer la valeur directement dans le code, consultez directement votre SDK spécifique.

Active ou désactive la découverte des points de terminaison pour DynamoDB.

La découverte des terminaux est obligatoire dans Timestream et facultative dans Amazon DynamoDB. Ce paramètre est défini par défaut sur l'un `true` ou l'autre ou `false` selon que le service nécessite ou non la découverte des points de terminaison. Les requêtes Timestream sont par défaut et les `true` requêtes Amazon DynamoDB sont définies par défaut sur. `false`

Valeurs valides:

- **`true`**— Le SDK doit automatiquement tenter de découvrir un point de terminaison pour les services où la découverte du point de terminaison est facultative.
- **`false`**— Le SDK ne doit pas tenter automatiquement de découvrir un point de terminaison pour les services où la découverte de point de terminaison est facultative.

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
AWS CLI v2	Oui	
SDK pour C++	Oui	
SDK pour Go V2 (1.x)	Oui	
SDK pour Go 1.x (V1)	Oui	Pour utiliser les paramètres des `config` fichiers partagés, vous devez activer le chargement à partir du fichier de configuration ; voir Sessions.
SDK pour Java 2.x	Oui	Le SDK pour Java 2.x `AWS_ENDPOINT_DISCOVERY_ENABLED` utilise comme nom de variable d'environnement.
SDK pour Java 1.x	Partie	La propriété du système JVM n'est pas prise en charge.
SDK pour 3.x JavaScript	Oui	

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
SDK pour 2.x JavaScript	Oui	
SDK pour Kotlin	Oui	
SDK pour .NET 4.x	Oui	
SDK pour .NET 3.x	Oui	
SDK pour PHP 3.x	Oui	
SDK pour Python (Boto3)	Oui	
SDK pour Ruby 3.x	Oui	
SDK pour Rust	Partie	Pris en charge uniquement pour Timestream.
SDK pour Swift	Nor	
Outils pour PowerShell V5	Oui	
Outils pour PowerShell V4	Oui	

Paramètres de configuration généraux

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

SDKs prend en charge certains paramètres généraux qui configurent les comportements généraux du SDK.

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`api_versions`- réglage AWS **`config`** du fichier partagé

Certains AWS services gèrent plusieurs versions d'API afin de garantir la rétrocompatibilité. Par défaut, le SDK et les AWS CLI opérations utilisent la dernière version d'API disponible. Pour exiger l'utilisation d'une version d'API spécifique pour vos demandes, incluez le `api_versions` paramètre dans votre profil.

Valeur par défaut : Aucune. (La dernière version de l'API est utilisée par le SDK.)

Valeurs valides : il s'agit d'un paramètre imbriqué suivi d'une ou de plusieurs lignes en retrait identifiant chacune un AWS service et la version d'API à utiliser. Consultez la documentation du AWS service pour savoir quelles versions d'API sont disponibles.

L'exemple définit une version d'API spécifique pour deux AWS services du `config` fichier. Ces versions de l'API ne sont utilisées que pour les commandes qui s'exécutent sous le profil qui contient ces paramètres. Les commandes de tout autre service utilisent la dernière version de l'API de ce service.

```
api_versions =
    ec2 = 2015-03-01
    cloudfront = 2015-09-017
```

`ca_bundle`- réglage AWS **`config`** du fichier partagé, **`AWS_CA_BUNDLE`**- variable d'environnement

Spécifie le chemin d'accès à un ensemble de certificats personnalisé (un fichier avec une `.pem` extension) à utiliser lors de l'établissement de SSL/TLS connexions.

Valeur par défaut : aucune

Valeurs valides : Spécifiez le chemin complet ou le nom du fichier de base. S'il existe un nom de fichier de base, le système tente de trouver le programme dans les dossiers spécifiés par la variable d'PATHenvironnement.

Exemple de définition de cette valeur dans le `config` fichier :

```
[default]
ca_bundle = dev/apps/ca-certs/cabundle-2019mar05.pem
```

En raison des différences dans la façon dont les systèmes d'exploitation gèrent les chemins et l'absence de caractères de chemin, voici un exemple de définition de cette valeur dans le `config` fichier sous Windows :

```
[default]
ca_bundle = C:\\Users\\username\\.aws\\aws-custom-bundle.pem
```

Exemple Linux/macOS de définition de variables d'environnement via la ligne de commande :

```
export AWS_CA_BUNDLE=/dev/apps/ca-certs/cabundle-2019mar05.pem
```

Exemple Windows de définition de variables d'environnement via la ligne de commande :

```
setx AWS_CA_BUNDLE C:\dev\apps\ca-certs\cabundle-2019mar05.pem
```

`output`- réglage AWS **`config`** du fichier partagé

Spécifie la manière dont les résultats sont formatés dans les outils AWS CLI et dans les autres AWS SDKs outils.

Valeur par défaut : `json`

Valeurs valides:

- **`json`** : la sortie est au format d'une chaîne JSON.
- **`yaml`** : la sortie est au format d'une chaîne YAML.
- **`yaml-stream`** : la sortie est diffusée et au format d'une chaîne YAML. La diffusion permet de traiter plus rapidement de gros types de données.
- **`text`** : la sortie a le format de plusieurs lignes de valeurs de chaîne séparées par des tabulations. Cela peut être utile pour transmettre la sortie à un processeur de texte, comme `grep`, `sed` ou `awk`.
- **`table`** : la sortie est au format d'un tableau utilisant les caractères +|- pour délimiter les bordures des cellules. La présentation des informations est dans un format beaucoup plus lisible par l'utilisateur que les autres, mais peu pratique du point de vue programmation.

`parameter_validation`- réglage AWS **`config`** du fichier partagé

Spécifie si le SDK ou l'outil tente de valider les paramètres de ligne de commande avant de les envoyer au point de terminaison du AWS service.

Valeur par défaut : `true`

Valeurs valides:

- **`true`** – Valeur par défaut Le SDK ou l'outil effectue la validation côté client des paramètres de ligne de commande. Cela permet au SDK ou à l'outil de confirmer que les paramètres sont valides et de détecter certaines erreurs. Le SDK ou l'outil peut rejeter les demandes non valides avant de les envoyer au point de terminaison du AWS service.
- **`false`**— Le SDK ou l'outil ne valide pas les paramètres de ligne de commande avant de les envoyer au point de terminaison du AWS service. Le point AWS de terminaison du service est chargé de valider toutes les demandes et de rejeter les demandes non valides.

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
AWS CLI v2	Partie	`api_versions` non pris en charge.
SDK pour C++	Oui	
SDK pour Go V2 (1.x)	Partie	`api_versions` et `parameter_validation` non pris en charge.
SDK pour Go 1.x (V1)	Partie	`api_versions` et `parameter_validation` non pris en charge. Pour utiliser les paramètres des `config` fichiers partagés, vous devez activer le chargement à partir du fichier de configuration ; voir Sessions.
SDK pour Java 2.x	Nor	
SDK pour Java 1.x	Nor	
SDK pour 3.x JavaScript	Oui	
SDK pour 2.x JavaScript	Oui	

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
SDK pour Kotlin	Nor	
SDK pour .NET 4.x	Nor	
SDK pour .NET 3.x	Nor	
SDK pour PHP 3.x	Oui	
SDK pour Python (Boto3)	Oui	
SDK pour Ruby 3.x	Oui	
SDK pour Rust	Nor	
SDK pour Swift	Nor	
Outils pour PowerShell V5	Nor	
Outils pour PowerShell V4	Nor	

Injection du préfixe hôte

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

L'injection de préfixe d'hôte est une fonctionnalité qui permet d'ajouter AWS SDKs automatiquement un préfixe au nom d'hôte des points de terminaison de service pour certaines opérations d'API. Ce préfixe peut être une chaîne statique ou une valeur dynamique qui inclut les données des paramètres de votre demande.

Par exemple, lorsque vous utilisez Amazon Simple Storage Service pour effectuer des actions sur des objets ou des compartiments Amazon S3, le SDK remplace le nom et l' Compte AWS ID de votre compartiment dans le point de terminaison final de l'API.

Bien que ce comportement soit requis pour les points de terminaison de AWS service normaux, il peut poser des problèmes lors de l'utilisation de points de terminaison personnalisés tels que des points de terminaison VPC ou des outils de test locaux. Dans ces cas, vous devrez peut-être désactiver l'injection de préfixe d'hôte.

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`disable_host_prefix_injection`- réglage AWS **`config`** du fichier partagé, **`AWS_DISABLE_HOST_PREFIX_INJECTION`**- variable d'environnement, **`aws.disableHostPrefixInjection`**- Propriété du système JVM : uniquement Java/Kotlin

Ce paramètre détermine si le SDK ou l'outil modifiera le nom d'hôte du point de terminaison en ajoutant un préfixe d'hôte tel que défini dans l'objet ou la variable client de votre SDK.

Valeur par défaut : `false`

Valeurs valides:

- **`true`**— Désactive l'injection de préfixe d'hôte. Le SDK ne modifiera pas le nom d'hôte du point de terminaison.
- **`false`**— Active l'injection de préfixe d'hôte. Le SDK ajoutera le préfixe d'hôte au nom d'hôte du point de terminaison.

Exemple de définition de cette valeur dans le `config` fichier :

```
[default]
disable_host_prefix_injection = true
```

Exemple Linux/macOS de définition de variables d'environnement via la ligne de commande :

```
export AWS_DISABLE_HOST_PREFIX_INJECTION=true
```

Exemple Windows de définition de variables d'environnement via la ligne de commande :

```
setx AWS_DISABLE_HOST_PREFIX_INJECTION true
```

Exemples d'injection de préfixes d'hôte

Le tableau d'exemples suivant montre comment SDKs modifier le point de terminaison final lorsque l'injection de préfixe d'hôte est activée ou désactivée.

- Préfixe d'hôte : modèle de chaîne de propriété de préfixe d'hôte définie dans le code sur l'objet ou la variable client du SDK.
- Entrées : entrées supplémentaires définies sur l'objet client ou la variable du SDK dans le code.
- Point de terminaison du client : point de terminaison dérivé du client.
- Valeur de réglage : valeur résolue pour le paramètre précédent.
- Point de terminaison résultant : point de terminaison résultant que le client du SDK utilise pour effectuer l'appel d'API.

Préfixe de l'hôte	Inputs	Point de terminaison client	Valeur de réglage	Point final résultant
« données ».	{}	"https://service.us-west-2.amazonaws.com"	false	"https://data.service.us-west-2.amazonaws.com"
« {Seau} - {AccountId}. »	Seau : « amzn-s3-demo-bucket1", :" 123456789012" AccountId	"https://service.us-west-2.amazonaws.com"	false	"https://amzn-s3-demo-bucket1-123456789012.service.us-west-2.amazonaws.com"
« données ».	{}	"https://override.us-west-2.amazonaws.com"(en tant que point de	true	"https://override.us-west-2.amazonaws.com"

Préfixe de l'hôte	Inputs	Point de terminaison client terminaison de remplacement)	Valeur de réglage	Point final résultant

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
AWS CLI v2	Oui	
SDK pour C++	Nor	Le paramètre n'est pas pris en charge, mais peut être configuré dans le code sur le client en utilisant : `enableHostPrefixInjection` .
SDK pour Go V2 (1.x)	Nor	Peut être désactivé à l'aide d'un intergiciel.
SDK pour Go 1.x (V1)	Nor	
SDK pour Java 2.x	Nor	Le paramètre n'est pas pris en charge, mais peut être configuré dans le code sur le client en utilisant : `SdkAdvancedClientOption.DISABLE_HOST_PREFIX_INJECTION` .
SDK pour Java 1.x	Nor	Le paramètre n'est pas pris en charge, mais peut être configuré dans le code sur le client en utilisant : `withDisableHostPrefixInjection` .

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
SDK pour 3.x JavaScript	Nor	Le paramètre n'est pas pris en charge, mais peut être configuré dans le code sur le client en utilisant : `disableHostPrefix` .
SDK pour 2.x JavaScript	Nor	Le paramètre n'est pas pris en charge, mais peut être configuré dans le code sur le client en utilisant : `hostPrefixEnabled` .
SDK pour Kotlin	Nor	
SDK pour .NET 4.x	Nor	Le paramètre n'est pas pris en charge, mais peut être configuré dans le code sur le client en utilisant : `DisableHostPrefixInjection` .
SDK pour .NET 3.x	Nor	Le paramètre n'est pas pris en charge, mais peut être configuré dans le code sur le client en utilisant : `DisableHostPrefixInjection` .
SDK pour PHP 3.x	Nor	Le paramètre n'est pas pris en charge, mais peut être configuré dans le code sur le client en utilisant : `disable_host_prefix_injection` .
SDK pour Python (Boto3)	Oui	Peut être configuré en code sur le client en utilisant : `inject_host_prefix` .
SDK pour Ruby 3.x	Nor	Le paramètre n'est pas pris en charge, mais peut être configuré dans le code sur le client en utilisant : `disable_host_prefix_injection` .
SDK pour Rust	Nor	
SDK pour Swift	Nor	

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
Outils pour PowerShell V5	Nor	Le paramètre n'est pas pris en charge, mais peut être inclus dans des applets de commande spécifiques à l'aide de paramètres. `-ClientConfig @{DisableHostPrefixInjection = $true}`
Outils pour PowerShell V4	Nor	Le paramètre n'est pas pris en charge, mais peut être inclus dans des applets de commande spécifiques à l'aide de paramètres. `-ClientConfig @{DisableHostPrefixInjection = $true}`

Client IMDS

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

SDKs implémentez un client Instance Metadata Service Version 2 (IMDSv2) à l'aide de requêtes orientées session. Pour plus d'informations IMDSv2, consultez la section Utilisation IMDSv2 dans le guide de l'utilisateur Amazon EC2. Le client IMDS est configurable via un objet de configuration client disponible dans la base de code du SDK.

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`retries`- membre de l'objet de configuration client

Le nombre de nouvelles tentatives pour chaque demande ayant échoué.

Valeur par défaut : 3

Valeurs valides : nombre supérieur à 0.

`port`- membre de l'objet de configuration client

Le port du point de terminaison.

Valeur par défaut : 80

Valeurs valides : Nombre.

`token_ttl`- membre de l'objet de configuration client

Le TTL du jeton.

Valeur par défaut : 21 600 secondes (6 heures, durée maximale allouée).

Valeurs valides : Nombre.

`endpoint`- membre de l'objet de configuration client

Le point final de l'IMDS.

Valeur par défaut : si elle `endpoint_mode` est égale`IPv4`, le point de terminaison par défaut est`http://169.254.169.254`. Si `endpoint_mode` égal`IPv6`, le point de terminaison par défaut est`http://[fd00:ec2::254]`.

Valeurs valides : URI valide.

Les options suivantes sont prises en charge par la plupart SDKs. Consultez la base de code de votre SDK spécifique pour plus de détails.

`endpoint_mode`- membre de l'objet de configuration client

Le mode endpoint de l'IMDS.

Valeur par défaut : `IPv4`

Valeurs valides : `IPv4`, `IPv6`

`http_open_timeout`- membre de l'objet de configuration client (le nom peut varier)

Le nombre de secondes à attendre avant l'ouverture de la connexion.

Valeur par défaut : 1 seconde.

Valeurs valides : nombre supérieur à 0.

`http_read_timeout`- membre de l'objet de configuration client (le nom peut varier)

Le nombre de secondes nécessaires à la lecture d'un bloc de données.

Valeur par défaut : 1 seconde.

Valeurs valides : nombre supérieur à 0.

`http_debug_output`- membre de l'objet de configuration client (le nom peut varier)

Définit un flux de sortie pour le débogage.

Valeur par défaut : Aucune.

Valeurs valides : un I/O flux valide, comme STDOUT.

`backoff`- membre de l'objet de configuration client (le nom peut varier)

Le nombre de secondes passées en veille entre deux tentatives ou le nombre de secondes qu'un client a fourni une fonction de temporisation pour appeler. Cela remplace la stratégie de ralentissement exponentiel par défaut.

Valeur par défaut : varie selon le SDK.

Valeurs valides : varient selon le SDK. Il peut s'agir d'une valeur numérique ou d'un appel à une fonction personnalisée.

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
AWS CLI v2	Oui	
SDK pour C++	Nor	
SDK pour Go V2 (1.x)	Oui	

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
SDK pour Go 1.x (V1)	Oui	
SDK pour Java 2.x	Oui	
SDK pour Java 1.x	Oui	
SDK pour 3.x JavaScript	Oui	
SDK pour 2.x JavaScript	Oui	
SDK pour Kotlin	Nor	
SDK pour .NET 4.x	Oui	
SDK pour .NET 3.x	Oui	
SDK pour PHP 3.x	Oui	
SDK pour Python (Boto3)	Oui	
SDK pour Ruby 3.x	Oui	
SDK pour Rust	Oui	
SDK pour Swift	Oui	
Outils pour PowerShell V5	Oui	
Outils pour PowerShell V4	Oui	

Comportement de nouvelle tentative

> **Important**
>
> Le comportement décrit sur cette page nécessite une activation jusqu'à ce qu'il devienne le comportement par défaut. `AWS_NEW_RETRIES_2026=true`Installez-le dans votre

environnement. Sans ce paramètre, votre SDK utilise le comportement des nouvelles tentatives antérieur à 2026, qui diffère en termes de temps d'attente, de coûts de quota de nouvelles tentatives et de valeurs par défaut spécifiques au service. Pour plus de détails, consultez le billet de blog consacré à l'annonce.

Lorsqu'une demande Service AWS échoue en raison d'une erreur transitoire ou d'un ralentissement, le SDK peut automatiquement réessayer la demande. Cette page explique comment configurer les nouvelles tentatives et comment elles fonctionnent en interne.

- Configuration des nouvelles tentatives: Choisissez un mode de nouvelle tentative, définissez le nombre maximum de tentatives et comprenez la priorité de configuration.
- Comment fonctionnent les nouvelles tentatives: flux de nouvelles tentatives, classification des erreurs, formule de temporisation, mécanisme des quotas de nouvelles tentatives et comportement spécifique au service.

Configuration des nouvelles tentatives

Vous contrôlez la stratégie de nouvelles tentatives utilisée par le SDK et le nombre de nouvelles tentatives.

Choix d'un mode de nouvelle tentative

Le mode nouvelle tentative détermine le comportement du SDK en cas d'échec d'une demande. Trois modes sont disponibles : standard, adaptatif et traditionnel.

	Standard	Adaptatif	Héritée
Quota de nouvelles tentatives	Oui	Oui	Varie selon le SDK
Peut retarder la demande initiale	Non	Oui	Non
Error-type-specific reculer	Oui	Oui	Varie selon le SDK
Standardisé pour tous les SDK	Oui	Oui	Non

	Standard	Adaptatif	Héritée
Recommendation	Par défaut pour toutes les charges de travail	Single-resource, puissant en termes d'étranglement, tolérant à la latence	Rétrocompatibilité uniquement

Mode standard (par défaut)

Le mode standard réessaie les demandes qui ont échoué en utilisant un retard exponentiel avec gigue. Il utilise des délais plus courts pour les erreurs transitoires (telles que les délais d'attente du réseau) et des délais plus longs pour les erreurs de régulation (telles que). `ThrottlingException`

Le mode standard inclut un quota de nouvelles tentatives, un bucket de jetons qui déduit les jetons à chaque nouvelle tentative et réapprovisionne les jetons lorsque les demandes aboutissent. Lorsque les jetons disponibles sont épuisés, le SDK renvoie l'erreur sans réessayer, de sorte que votre application échoue rapidement au lieu d'attendre des tentatives qui ont peu de chances de réussir. Cela permet également de résoudre plus rapidement les interruptions de service en réduisant le nombre de nouvelles tentatives. Pendant le fonctionnement normal, le quota reste plein et n'a aucun effet. Le quota de nouvelles tentatives ne retarde ni ne bloque jamais la demande initiale. Seules les nouvelles tentatives sont concernées. Pour en savoir plus, consultez Quota de nouvelles tentatives (bucket de jetons).

Utilisez le mode standard sauf si vous avez une raison précise de choisir un autre mode.

Mode adaptatif

Le mode adaptatif inclut tout le mode standard, plus un limiteur de débit côté client. Le limiteur de débit suit les réponses de régulation et ajuste le débit auquel le SDK envoie les demandes. Contrairement au mode standard, le mode adaptatif peut retarder ou bloquer la demande initiale, et pas simplement les nouvelles tentatives, lorsqu'un étranglement est détecté.

Le limiteur de débit fonctionne par instance client du SDK. Toutes les demandes d'un client partagent la même limite de débit, quelle que soit l'opération d'API ou la ressource qu'elles ciblent.

Quand utiliser le mode adaptatif :

- Votre client cible une seule ressource (par exemple, une table DynamoDB) et vous vous attendez à des réponses de régulation fréquentes. Cela est courant dans les flux de travail automatisés, les processeurs par lots ou les charges de travail basées sur l'IA qui font appel à une seule opération d'API à volume élevé.
- Vous souhaitez que le SDK ralentisse automatiquement lorsque le service signale une limitation.

Quand ne pas utiliser le mode adaptatif :

- Votre client envoie des demandes à plusieurs ressources ou dessert plusieurs locataires. La limitation d'une ressource entraîne le ralentissement par le limiteur de débit de toutes les demandes provenant de ce client, y compris les demandes adressées aux ressources non affectées.
- Vous avez besoin d'une latence prévisible lors de la demande initiale.

Le mode adaptatif n'est pas recommandé par défaut.

Mode Héritage

Le mode Legacy est le comportement de nouvelle tentative utilisé par chaque SDK avant l'introduction du mode standard. Il n'inclut pas de quota de nouvelles tentatives normalisé. Certains SDK (tels que Java) avaient leurs propres implémentations de quotas de nouvelles tentatives en mode ancien, mais le comportement n'est pas uniforme d'un SDK à l'autre. Sans quota standardisé, un client continue de réessayer à plein régime en cas d'interruption de service. Cela bloque les threads et les connexions lorsque les demandes ont peu de chances d'aboutir, tout en ajoutant une charge susceptible de retarder le rétablissement du service.

Le mode Legacy varie selon les SDK. Le nombre de tentatives, le délai d'attente, les ensembles d'erreurs réessayables et le comportement de limitation varient selon les langues. Le code qui dépend du comportement de nouvelle tentative existant peut se comporter différemment lorsqu'il est déplacé entre les SDK.

Disponible en : Java, Python, Ruby, PHP, C++, CLI

Non disponible dans : .NET, Go, Kotlin, Rust, Swift, JavaScript

Le mode Legacy existe pour des raisons de rétrocompatibilité. Si vous utilisez actuellement le mode traditionnel, passez en mode standard.

Réglages de nouvelle tentative

Les paramètres suivants contrôlent le comportement des nouvelles tentatives. Vous pouvez les définir via des variables d'environnement, le fichier de configuration partagé (`~/.aws/config`) ou la configuration du client dans le code.

Paramètre	Ce qu'il contrôle	Variable d'environnement	Clé du fichier de configuration	Par défaut
Mode nouvelle tentative	Quelle stratégie de nouvelle tentative utiliser	`AWS_RETRY_MODE`	`retry_mode`	`standard`
Nombre maximum de tentatives	Nombre total de tentatives, demande initiale comprise	`AWS_MAX_ATTEMPTS`	`max_attempts`	3(voir notes)

Une valeur maximale de tentatives de 3 signifie que le SDK effectue une demande initiale et jusqu'à deux nouvelles tentatives. Définissez le nombre maximum de tentatives sur 1 pour désactiver complètement les nouvelles tentatives.

> **Note**
>
> Les clients DynamoDB et DynamoDB Streams utilisent par défaut le nombre maximum de tentatives. 4 Ces services utilisent un délai de temporisation de base plus court (25 ms au lieu de 50 ms) pour correspondre à leur profil de faible latence. La tentative supplémentaire permet de maintenir le retard maximal de la dernière tentative comparable à celui des autres services. Vous pouvez le remplacer par les mêmes paramètres que ceux indiqués dans le tableau précédent.

Ordre de priorité de configuration

Lorsque vous spécifiez le même paramètre à plusieurs endroits, le SDK résout la valeur en utilisant la priorité suivante, de la plus élevée à la plus faible :

1. Configuration client explicite dans le code. Une valeur définie directement sur le client du SDK ou son objet de configuration.
2. Variable d'environnement. Par exemple, `AWS_RETRY_MODE` ou `AWS_MAX_ATTEMPTS`.
3. Fichier de configuration partagé. Entrez la `max_attempts` touche `retry_mode` OR`~/.aws/config`.
4. SDK par défaut. La valeur par défaut intégrée pour le paramètre.

Cela suit la priorité de configuration standard du AWS SDK. Une valeur définie à un niveau supérieur remplace toujours une valeur définie à un niveau inférieur. Par exemple, si vous définissez `AWS_RETRY_MODE=adaptive` en tant que variable d'environnement et `retry_mode=standard` en tant que variable d'environnement`~/.aws/config`, le SDK utilise le mode adaptatif.

Language-specific configuration

Les paramètres inter-SDK décrits sur cette page (`retry_modeetmax_attempts`) fonctionnent dans tous les SDK. Cependant, l'API permettant de configurer les nouvelles tentatives dans le code varie selon la langue. Consultez le guide du développeur de votre SDK pour connaître les options de configuration spécifiques au langage, telles que les stratégies de latence personnalisées, les erreurs supplémentaires susceptibles d'être réessayées et le réglage du quota de nouvelles tentatives.

Comment fonctionnent les nouvelles tentatives

Cette section décrit la manière dont AWS les SDK traitent les demandes ayant échoué : quelles erreurs déclenchent de nouvelles tentatives, combien de temps le SDK attend entre les tentatives et quand il arrête de réessayer.

Que se passe-t-il lorsqu'une demande échoue

Lorsque vous effectuez un appel d'API via un AWS SDK, celui-ci suit la séquence suivante :

1. Mode adaptatif uniquement : le SDK vérifie le limiteur de débit côté client. Si un étranglement a été détecté, le SDK peut retarder ou bloquer la demande avant de l'envoyer.
2. Le SDK envoie la demande au point de Service AWS terminaison.
3. Si le service renvoie une réponse positive, le SDK renvoie le résultat à votre code.
4. Si la demande échoue, le SDK classe l'erreur comme transitoire, limitée ou non réessayable. Consultez Quelles erreurs sont réessayées.
5. Si l'erreur ne peut pas être réessayée, le SDK renvoie immédiatement l'erreur à votre code. Aucune nouvelle tentative n'est tentée.

6. Si l'erreur est réessayable, le SDK vérifie si le nombre maximum de tentatives a été atteint. Si tel est le cas, il renvoie l'erreur à votre code.
7. Le SDK vérifie le. Quota de nouvelles tentatives (bucket de jetons) Si le budget des jetons est épuisé, le SDK ne réessaie pas et renvoie l'erreur à votre code. Exception : Long-polling opérations car le SDK applique toujours un délai d'attente avant de renvoyer l'erreur.
8. Le SDK calcule un délai d'attente en fonction du type d'erreur et du nombre de nouvelles tentatives. Consultez Combien de temps le SDK attend-il.
9. Le SDK attend le délai calculé, puis envoie à nouveau la demande à partir de l'étape 2.

Le SDK répète cette boucle jusqu'à ce que la demande aboutisse, que le nombre maximal de tentatives soit atteint, que le quota de nouvelles tentatives soit épuisé ou qu'une erreur non réessayable se produise. L'ensemble du processus est automatique. Votre application reçoit soit une réponse positive, soit une dernière erreur.

Quelles erreurs sont réessayées

Le SDK classe chaque demande ayant échoué dans l'une des trois catégories suivantes : transitoire, limitée ou non réessayable. Cette classification détermine si le SDK tente à nouveau la demande et combien de temps il attend avant de réessayer.

La classification est basée sur le code d'erreur et le code d'état HTTP figurant dans la réponse du service. Par exemple, un HTTP 400 avec le code d'erreur `RequestTimeout` est classé comme transitoire et réessayé. Un HTTP 400 avec `ValidationException` est classé comme non réessayable et renvoyé immédiatement.

Classification des erreurs

Les erreurs transitoires sont réessayées avec un court délai de base (50 ms) :

Code d’erreur
`RequestTimeout`
`RequestTimeoutException`
`InternalError`
`IDPCommunicationError`

Code d’erreur
I/O Échec (réinitialisation de la connexion, échec de la résolution DNS, délai d'expiration du socket)
(tout HTTP 500, 502, 503 ou 504 sans code d'erreur reconnu)

Les erreurs de régulation sont réessayées avec un délai de base plus long (1 000 ms) :

Code d’erreur
`Throttling`
`ThrottlingException`
`ThrottledException`
`RequestThrottledException`
`TooManyRequestsException`
`ProvisionedThroughputExceededException`
`TransactionInProgressException`
`LimitExceededException`
`PriorRequestNotComplete`
`RequestThrottled`
`EC2ThrottledException`
`RequestLimitExceeded`
`SlowDown`
`BandwidthLimitExceeded`

Non-retryable les erreurs (telles que`AccessDeniedException`,`ValidationException`,`ResourceNotFoundException`) sont immédiatement renvoyées à votre code.

> **Note**
>
> Un HTTP 5XX avec un code d'erreur de régulation est classé comme une erreur de régulation et non comme une erreur transitoire, même si les erreurs 5XX sont normalement transitoires. Le SDK correspond d'abord au code d'erreur, puis revient au code d'état HTTP.

Les erreurs de régulation signifient que le service a activement rejeté votre demande en raison de limites de débit. Le SDK attend donc plus longtemps avant de réessayer afin de donner au service le temps de récupérer sa capacité. Consultez Combien de temps le SDK attend-il les délais spécifiques.

Combien de temps le SDK attend-il

Le SDK utilise un ralentissement exponentiel avec une instabilité totale. En moyenne, chaque nouvelle tentative attend plus longtemps que la précédente, et la randomisation permet de répartir les demandes provenant de plusieurs clients.

Retards de base par type d'erreur

Le délai de base varie selon que l'erreur est transitoire ou limitée :

Error type (Type d'erreur)	Délai de base	Justification
Transitoire (sans étranglement)	50 millisecondes	Les erreurs transitoires sont généralement résolues en quelques millisecondes. Un court délai de base permet une reprise rapide.
étranglement	1 000 ms	Le service a limité le débit de la demande. Un délai de base plus long donne le temps de récupérer la capacité.

Formule Backoff

Le SDK calcule le délai de chaque nouvelle tentative à l'aide de la formule suivante :

```
delay = random(0, 1) × min(20,000 ms, base_delay × 2^retry)
```

Où :

- `random(0, 1)`renvoie une valeur uniformément répartie entre 0 et 1
- `base_delay`est de 50 ms pour les erreurs transitoires ou de 1 000 ms pour les erreurs de régulation
- `retry`commence à 0 pour la première tentative (la deuxième tentative globale de demande)

La durée maximale d'attente est de 20 secondes. Aucun délai individuel ne dépasse 20 secondes, quel que soit le nombre de tentatives effectuées.

Exemples réussis

Exemple 1 : erreur transitoire, 3 tentatives maximum

Step (Étape)	Que se passe-t-il	Delay
Tentative 1	Demande initiale. Le service renvoie HTTP 503.	(aucun)
Tentative 2	Le SDK attend aléatoirement (0, 50 ms). La nouvelle tentative échoue avec 503.	0 à 50 ms (moyenne ~25 ms)
Tentative 3	Le SDK attend aléatoirement (0, 100 ms). La nouvelle tentative réussit.	0 à 100 ms (moyenne ~50 ms)

La latence ajoutée totale est en moyenne d'environ 75 ms lors des deux tentatives.

Exemple 2 : erreur de limitation, 3 tentatives maximum

Step (Étape)	Que se passe-t-il	Delay
Tentative 1	Demande initiale. Les retours de service 429`Throttling` .	(aucun)

Step (Étape)	Que se passe-t-il	Delay
Tentative 2	Le SDK attend aléatoirement (0, 1 000 ms). Retry renvoie 429.	0 à 1 000 ms (moyenne ~500 ms)
Tentative 3	Le SDK attend aléatoirement (0, 2 000 ms). La nouvelle tentative réussit.	0 à 2 000 ms (moyenne ~1 000 ms)

La latence ajoutée totale est en moyenne d'environ 1 500 ms lors des deux tentatives.

Exemple 3 : erreur transitoire, atteinte du plafond de réduction

Avec un délai de base de 50 ms, le délai calculé avant le plafonnement serait :

Tentative de nouvelle tentative	Délai maximal calculé	Après un plafond de 20 s
1	50 millisecondes	50 millisecondes
2	100 millisecondes	100 millisecondes
5	800 millisecondes	800 millisecondes
9	12 800 ms	12 800 ms
10	25 600 ms	20 000 ms

Le plafond prend effet à la 10e tentative (11e tentative) pour les erreurs transitoires. Pour les erreurs de régulation avec une base de 1 000 ms, le plafond prend effet à la 6e tentative.

> **Note**
>
> Avec la valeur par défaut de 3 tentatives maximum (1 demande initiale + 2 nouvelles tentatives), le délai maximal n'est jamais atteint. Ce tableau illustre ce qui se passe si vous augmentez `max_attempts` bien au-delà de la valeur par défaut.

Pourquoi la nervosité est importante

Le multiplicateur aléatoire est appelé full jitter. Sans cela, tous les clients qui rencontraient une erreur en même temps réessaieraient en même temps, ce qui créerait un pic de trafic de nouvelles tentatives (le problème du « tonnerre »). L'instabilité totale répartit les nouvelles tentatives de manière uniforme sur l'ensemble de la fenêtre d'attente, de sorte que le service reçoit un flot constant de demandes au lieu de pics synchronisés.

Supposons, par exemple, que 1 000 clients reçoivent tous un 503 au même moment. Full Jitter répartit leurs premières tentatives de manière uniforme sur une fenêtre de 50 ms au lieu de les faire toutes les 1 000 tentatives exactement à 50 ms.

Server-directed réessayer le chronométrage

Certains Services AWS incluent un `x-amz-retry-after` en-tête dans les réponses aux erreurs. La valeur de l'en-tête est un délai en millisecondes. Lorsque cet en-tête est présent, le SDK utilise le délai spécifié par le serveur, limité au minimum au délai de temporisation calculé et au maximum au délai de temporisation calculé majoré de 5 000 ms. Le délai calculé étant lui-même plafonné à 20 secondes, le délai maximal effectif dirigé par le serveur est de 25 secondes. Le SDK n'applique pas de gigue à cette valeur, car le service est censé la modifier. Cela permet au service de communiquer exactement au moment où il prévoit disposer de la capacité disponible.

Quota de nouvelles tentatives (bucket de jetons)

Le SDK gère un budget interne de jetons qui permet de suivre le ratio entre les demandes réussies et les échecs. Lorsque les défaillances sont généralisées, le budget s'épuise et le SDK renvoie directement les erreurs. Votre application échoue rapidement au lieu d'attendre de nouvelles tentatives qui ont peu de chances de réussir. Cela réduit également le trafic de nouvelles tentatives, ce qui permet de résoudre plus rapidement les interruptions de service.

Comment fonctionne le quota de nouvelles tentatives

Le budget symbolique commence au point de départ plein. Chaque nouvelle tentative déduit des jetons. Lorsqu'une nouvelle tentative réussit, le SDK restaure les jetons consommés par cette nouvelle tentative. Lorsqu'une demande aboutit au premier essai (aucune nouvelle tentative n'est nécessaire), le SDK restaure 1 jeton. Lorsque le budget atteint zéro, le SDK arrête de réessayer et renvoie les erreurs directement dans votre code.

Paramètre	Value
Capacité budgétaire	500 jetons
Coût par nouvelle tentative transitoire (sans limitation)	14 jetons
Coût par nouvelle tentative de limitation	5 jetons
Tokens restaurés en cas de succès après une nouvelle tentative	Quantité consommée lors de la dernière tentative (14 ou 5)
Tokens restaurés en cas de succès sans nouvelle tentative	1 jeton

Le coût plus élevé des tentatives transitoires reflète leur schéma d'échec différent. Les erreurs transitoires telles que les 500 et les défaillances de connexion indiquent souvent un problème à l'échelle du service. Dans ces situations, il est peu probable que la poursuite des tentatives aboutisse. Cela augmente la latence de vos appels, bloque les ressources des clients et peut retarder le rétablissement pour tout le monde. Les erreurs de régulation indiquent que le service a besoin de plus de temps avant que la demande puisse aboutir. Le SDK attend plus longtemps entre les tentatives afin d'améliorer les chances de réussite.

Quand le bloc de quotas réessaie-t-il ?

Le quota de nouvelles tentatives permet de suivre les jetons à tout moment, mais ne bloque les nouvelles tentatives que lorsque le budget est épuisé. Pendant le fonctionnement normal, presque toutes les demandes aboutissent et le budget reste plein. Le quota n'a aucun effet observable sur les nouvelles tentatives.

Une nouvelle tentative réussie rétablit uniquement son propre coût en jetons (14 ou 5 jetons), et non le coût des tentatives infructueuses précédentes dans le cadre de la même demande. Par exemple, si la première tentative échoue et que la seconde réussit, le budget perd 14 jetons nets. Le budget s'épuise le plus rapidement lorsque toutes les tentatives sont épuisées sans succès, mais il s'épuise également progressivement lorsque les demandes nécessitent plusieurs tentatives avant d'aboutir.

Avec la valeur par défaut de 3 tentatives maximum, le quota commence à s'épuiser lorsque plus d'environ 22 % des demandes entraînent des échecs transitoires prolongés, ou plus d'environ 32 %

en cas d'erreurs de limitation. En dessous de ces taux, les demandes réussies réapprovisionnent le budget plus rapidement que les tentatives infructueuses ne l'épuisent.

Le solde de départ de 500 jetons du budget fournit une marge de manœuvre qui permet d'absorber les défaillances de courte durée. Une brève augmentation du nombre d'erreurs, même grave, ne bloque pas les nouvelles tentatives, sauf si elle persiste suffisamment longtemps pour épuiser la mémoire tampon.

Implications pratiques

- Faibles taux d'échec : le quota n'a aucun effet. Le budget reste à pleine capacité ou presque.
- En cas d'interruption de service : si un pourcentage élevé de vos demandes échouent pendant une période prolongée, le quota est épuisé et votre client reçoit immédiatement des erreurs au lieu d'attendre de nouvelles tentatives. Cela réduit la latence côté client, libère des threads et des connexions, et permet au service de se rétablir plus rapidement.
- Restauration : à mesure que le service se rétablit et que les demandes recommencent à aboutir, les tentatives réussies rétablissent le coût total du jeton et les tentatives réussies au premier essai restaurent 1 jeton. Le budget se recharge progressivement et tente à nouveau de reprendre automatiquement.
- Champ d'application : le budget des jetons est généralement limité à une seule instance client du SDK. La portée exacte peut varier selon le SDK. Il n'est pas partagé entre les processus ou les hôtes.

Service-specific comportement

DynamoDB

Les clients DynamoDB utilisent des paramètres par défaut optimisés pour le profil de faible latence de DynamoDB :

Paramètre	Valeur par défaut générale	DynamoDB par défaut
Retard de base transitoire (sans étranglement)	50 millisecondes	25 millisecondes
Atténuation du délai de base	1 000 ms	1 000 ms
Nombre maximum de tentatives	3	4

Ces valeurs par défaut s'appliquent à Amazon DynamoDB et à DynamoDB Streams.

Long-polling opérations

Certaines AWS opérations utilisent de longs sondages. Ils peuvent maintenir une connexion ouverte en attendant l'arrivée des travaux. Ces opérations font l'objet d'un traitement spécial pour les nouvelles tentatives :

- `SQS.ReceiveMessage`
- `SFN.GetActivityTask`
- `SWF.PollForActivityTask`
- `SWF.PollForDecisionTask`

Comportement spécial : lorsque le quota de nouvelles tentatives est épuisé et que les nouvelles tentatives sont bloquées (étape 7 plus loinQue se passe-t-il lorsqu'une demande échoue), le SDK applique toujours un délai avant de renvoyer l'erreur à votre code.

Cela est important car les longues opérations de sondage sont généralement organisées en boucle étroite. Votre code appelle`ReceiveMessage`, traite tous les messages, puis appelle `ReceiveMessage` à nouveau immédiatement. Sans ce recul forcé, un budget de jetons épuisé entraînerait le renvoi d'erreurs par le SDK sans délai. Votre boucle de sondage enverra alors immédiatement la demande suivante, ce qui augmentera l'utilisation du processeur du client et générera du trafic supplémentaire. Le délai d'attente forcé met fin à ce cycle, ce qui permet de gérer l'utilisation des ressources des clients et le taux d'interrogation en cas de panne.

Support par AWS Kits SDK et outils

Le tableau suivant indique la disponibilité du comportement de nouvelle tentative mis à jour dans chaque SDK. Pour SDK-specific plus de détails, notamment la version minimale, les valeurs par défaut avant/après et des exemples de code, consultez le GitHub problème de suivi.

Kit SDK	Pris en charge	GitHub problème de suivi
SDK pour Java 2.x	Oui	Problème de suivi
SDK pour Python (Boto3)	Oui	Problème de suivi
SDK pour .NET 4.x	Oui	Problème de suivi

Kit SDK	Pris en charge	GitHub problème de suivi
Outils pour PowerShell V5	Oui	Problème de suivi
SDK pour 3.x JavaScript	Oui	Problème de suivi
SDK pour PHP 3.x	Oui	Problème de suivi
SDK pour Kotlin	Oui	Problème de suivi
SDK pour Rust	Oui	Problème de suivi
SDK pour Swift	Voir le problème de suivi	Problème de suivi
SDK pour Ruby 3.x	Voir le problème de suivi	Problème de suivi
SDK pour Go V2 (1.x)	Voir le problème de suivi	Problème de suivi
SDK pour C++	Voir le problème de suivi	Problème de suivi
AWS CLI v2	Voir le problème de suivi	Problème de suivi

Compression des demandes

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

AWS SDKs et les outils peuvent automatiquement compresser les charges utiles lors de l'envoi de demandes à Services AWS ce support recevant des charges utiles compressées. La compression de la charge utile du client avant de l'envoyer à un service peut réduire le nombre total de demandes

et la bande passante nécessaires pour envoyer des données au service, ainsi que le nombre de demandes infructueuses en raison des limites du service relatives à la taille de la charge utile. Pour la compression, le SDK ou l'outil sélectionne un algorithme de codage pris en charge à la fois par le service et par le SDK. Cependant, la liste actuelle des encodages possibles se compose uniquement de gzip, mais elle pourrait s'étendre à l'avenir.

La compression des demandes peut être particulièrement utile si votre application utilise Amazon CloudWatch. CloudWatch est un service de surveillance et d'observabilité qui collecte des données opérationnelles et de surveillance sous forme de journaux, de métriques et d'événements. La méthode PutMetricDataAPI est un exemple d'opération de service qui prend en charge CloudWatch la compression.

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`disable_request_compression`- réglage AWS **`config`** du fichier partagé, **`AWS_DISABLE_REQUEST_COMPRESSION`**- variable d'environnement, **`aws.disableRequestCompression`**- Propriété du système JVM : uniquement Java/Kotlin

Active ou désactive la compression d'une charge utile par le SDK ou l'outil avant d'envoyer une demande.

Valeur par défaut : `false`

Valeurs valides:

- **`true`**— Désactive la compression des demandes.
- **`false`**— Utilisez la compression des demandes lorsque cela est possible.

`request_min_compression_size_bytes`- réglage AWS **`config`** du fichier partagé, **`AWS_REQUEST_MIN_COMPRESSION_SIZE_BYTES`**- variable d'environnement, **`aws.requestMinCompressionSizeBytes`**- Propriété du système JVM : uniquement Java/Kotlin

Définit la taille minimale en octets du corps de la demande que le SDK ou l'outil doit compresser. Les petites charges utiles peuvent devenir plus longues lorsqu'elles sont compressées. Il existe donc une limite inférieure à laquelle il est judicieux de procéder à la compression. Cette valeur est inclusive, une taille de demande supérieure ou égale à la valeur est compressée.

Valeur par défaut : 10240 octets

Valeurs valides : valeur entière comprise entre 0 et 10485760 octets inclus.

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
AWS CLI v2	Oui	
SDK pour C++	Oui	
SDK pour Go V2 (1.x)	Oui	
SDK pour Go 1.x (V1)	Non	
SDK pour Java 2.x	Oui	
SDK pour Java 1.x	Non	
SDK pour 3.x JavaScript	Oui	
SDK pour 2.x JavaScript	Non	
SDK pour Kotlin	Oui	
SDK pour .NET 4.x	Oui	
SDK pour .NET 3.x	Oui	
SDK pour PHP 3.x	Oui	
SDK pour Python (Boto3)	Oui	
SDK pour Ruby 3.x	Oui	
SDK pour Rust	Oui	
SDK pour Swift	Non	

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
Outils pour PowerShell V5	Oui	
Outils pour PowerShell V4	Oui	

Points de terminaison spécifiques au service

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

La configuration du point de terminaison spécifique au service offre la possibilité d'utiliser un point de terminaison de votre choix pour les demandes d'API et de conserver ce choix. Ces paramètres offrent la possibilité de prendre en charge les points de terminaison locaux, les points de terminaison d'un VPC et les environnements de développement AWS locaux tiers. Différents points de terminaison peuvent être utilisés pour les environnements de test et de production. Vous pouvez indiquer une URL de point de terminaison pour des Services AWS individuels.

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`endpoint_url`- réglage AWS **`config`** du fichier partagé, **`AWS_ENDPOINT_URL`**- variable d'environnement, **`aws.endpointUrl`**- Propriété du système JVM : uniquement Java/Kotlin

Lorsqu'il est spécifié directement dans un profil ou en tant que variable d'environnement, ce paramètre indique le point de terminaison utilisé pour toutes les demandes de service. Ce point de terminaison est remplacé par tout point de terminaison spécifique au service configuré.

Vous pouvez également utiliser ce paramètre dans une `services` section d'un AWS `config` fichier partagé pour définir un point de terminaison personnalisé pour un service spécifique. Pour obtenir la liste de toutes les clés d'identification de service à utiliser pour les sous-sections de `services` cette section, consultezIdentifiants pour les points de terminaison spécifiques au service.

Valeur par défaut : none

Valeurs valides : URL incluant le schéma et l'hôte du point de terminaison. L'URL peut éventuellement contenir un composant de chemin contenant un ou plusieurs segments de chemin.

AWS_ENDPOINT_URL_<SERVICE>- variable d'environnement, **aws.endpointUrl<ServiceName>**- Propriété du système JVM : uniquement Java/Kotlin

AWS_ENDPOINT_URL_<SERVICE>, où <SERVICE> est l' Service AWS identifiant, définit un point de terminaison personnalisé pour un service spécifique. Pour obtenir la liste de toutes les variables d'environnement spécifiques aux services, consultez. Identifiants pour les points de terminaison spécifiques au service

Ce point de terminaison spécifique au service remplace tout point de terminaison global défini dans. AWS_ENDPOINT_URL

Valeur par défaut : none

Valeurs valides : URL incluant le schéma et l'hôte du point de terminaison. L'URL peut éventuellement contenir un composant de chemin contenant un ou plusieurs segments de chemin.

ignore_configured_endpoint_urls- réglage AWS **config** du fichier partagé, **AWS_IGNORE_CONFIGURED_ENDPOINT_URLS**- variable d'environnement, **aws.ignoreConfiguredEndpointUrls**- Propriété du système JVM : uniquement Java/Kotlin

Ce paramètre est utilisé pour ignorer toutes les configurations de points de terminaison personnalisées.

Notez que tout point de terminaison explicite défini dans le code ou sur un client de service lui-même est utilisé quel que soit ce paramètre. Par exemple, l'inclusion du paramètre de ligne de `--endpoint-url` commande dans une AWS CLI commande ou la transmission d'une URL de point de terminaison à un constructeur client prendra toujours effet.

Valeur par défaut : `false`

Valeurs valides:

- **true**— Le SDK ou l'outil ne lit aucune option de configuration personnalisée à partir du `config` fichier partagé ou des variables d'environnement pour définir l'URL d'un point de terminaison.

- **`false`**— Le SDK ou l'outil utilise tous les points de terminaison disponibles fournis par l'utilisateur à partir du `config` fichier partagé ou des variables d'environnement.

Configuration des points de terminaison à l'aide de variables d'environnement

Pour acheminer les demandes de tous les services vers une URL de point de terminaison personnalisée, définissez la variable d'environnement `AWS_ENDPOINT_URL` globale.

```
export AWS_ENDPOINT_URL=http://localhost:4567
```

Pour acheminer les demandes d'une URL de point de terminaison spécifique Service AWS vers une URL de point de terminaison personnalisée, utilisez la variable d'`AWS_ENDPOINT_URL_<SERVICE>`environnement. Amazon DynamoDB a un `serviceId` de DynamoDB. Pour ce service, la variable d'environnement de l'URL du point de terminaison est `AWS_ENDPOINT_URL_DYNAMODB`. Ce point de terminaison a priorité sur le point de terminaison global défini `AWS_ENDPOINT_URL` pour ce service.

```
export AWS_ENDPOINT_URL_DYNAMODB=http://localhost:5678
```

Comme autre exemple, AWS Elastic Beanstalk possède un `serviceId` de `Elastic Beanstalk`. L' Service AWS identifiant est basé sur le modèle d'API `serviceId` en remplaçant tous les espaces par des traits de soulignement et en majuscules toutes les lettres. Pour définir le point de terminaison de ce service, la variable d'environnement correspondante est`AWS_ENDPOINT_URL_ELASTIC_BEANSTALK`. Pour obtenir la liste de toutes les variables d'environnement spécifiques aux services, consultez. Identifiants pour les points de terminaison spécifiques au service

```
export AWS_ENDPOINT_URL_ELASTIC_BEANSTALK=http://localhost:5567
```

Configuration des points de terminaison à l'aide du fichier partagé **`config`**

Dans le `config` fichier partagé, `endpoint_url` il est utilisé à différents endroits pour différentes fonctionnalités.

- `endpoint_url`spécifié directement dans a `profile` fait de ce point de terminaison le point de terminaison global.
- `endpoint_url`imbriqué sous une clé d'identification de service dans une `services` section, ce point de terminaison s'applique aux demandes adressées uniquement à ce service. Pour plus

de détails sur la définition d'une section `services` dans votre fichier `config` partagé, consultez Format du fichier de configuration.

L'exemple suivant utilise une `services` définition pour configurer une URL de point de terminaison spécifique au service à utiliser pour Amazon S3 et un point de terminaison global personnalisé à utiliser pour tous les autres services :

```
[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
```

Un profil unique peut configurer des points de terminaison pour plusieurs services. Cet exemple montre comment définir le point de terminaison spécifique au service URLs pour Amazon S3 et AWS Elastic Beanstalk dans le même profil. AWS Elastic Beanstalk a un `serviceId` de `Elastic Beanstalk`. L' Service AWS identifiant est basé sur le modèle d'API `serviceId` en remplaçant tous les espaces par des traits de soulignement et en minuscules toutes les lettres. Ainsi, la clé d'identification du service devient `elastic_beanstalk` et les paramètres de ce service commencent sur la ligne`elastic_beanstalk =` . Pour obtenir la liste de toutes les clés d'identification de service à utiliser dans la section `services`, consultez Identifiants pour les points de terminaison spécifiques au service.

```
[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
```

La section de configuration du service peut être utilisée à partir de plusieurs profils. Par exemple, deux profils peuvent utiliser la même `services` définition tout en modifiant d'autres propriétés de profil :

```
[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
```

Configurer les points de terminaison dans les profils à l'aide d'informations d'identification basées sur les rôles

Si votre profil comporte des informations d'identification basées sur un rôle configurées via un paramètre `source_profile` pour la fonctionnalité de rôle de responsable IAM, le kit SDK utilise uniquement les configurations de service pour le profil spécifié. Il n'utilise pas de profils auxquels des rôles sont liés. Par exemple, en utilisant le fichier `config` partagé suivant :

```
[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
```

Si vous utilisez le profil `B` et que vous effectuez un appel dans votre code à Amazon EC2, le point de terminaison est résolu en `https://profile-b-ec2-endpoint.aws`. Si votre code envoie une demande à un autre service, la résolution du point de terminaison ne suivra aucune logique personnalisée. Le point de terminaison n'est pas résolu en point de terminaison global défini dans le profil `A`. Pour qu'un point de terminaison global prenne effet pour le profil `B`, vous devez définir `endpoint_url` directement dans le profil `B`. Pour plus d'informations sur le paramètre `source_profile`, consultez Assumer le rôle de fournisseur d'informations d'identification.

Priorité des paramètres

Les paramètres de cette fonctionnalité peuvent être utilisés simultanément, mais une seule valeur sera prioritaire par service. Pour les appels d'API effectués vers une valeur donnée Service AWS, l'ordre suivant est utilisé pour sélectionner une valeur :

1. Tout paramètre explicite défini dans le code ou sur un client de service lui-même a priorité sur tout autre paramètre.
   - Pour le AWS CLI, il s'agit de la valeur fournie par le paramètre de ligne de `--endpoint-url` commande. Pour un SDK, les attributions explicites peuvent prendre la forme d'un paramètre que vous définissez lorsque vous instanciez un Service AWS client ou un objet de configuration.
2. La valeur fournie par une variable d'environnement spécifique au service, telle que. `AWS_ENDPOINT_URL_DYNAMODB`
3. La valeur fournie par la variable d'environnement `AWS_ENDPOINT_URL` globale du point de terminaison.
4. Valeur fournie par le `endpoint_url` paramètre imbriqué sous une clé d'identification de service dans une `services` section du `config` fichier partagé.
5. La valeur fournie par le `endpoint_url` paramètre spécifié directement dans un `profile config` fichier partagé.
6. Toute URL de point de terminaison par défaut pour le point de terminaison correspondant Service AWS est utilisée en dernier.

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
AWS CLI v2	Oui	
SDK pour C++	Oui	

Kit SDK	Pr er ch	Remarques ou informations supplémentaires
SDK pour Go V2 (1.x)	Oui	
SDK pour Go 1.x (V1)	Nor	
SDK pour Java 2.x	Oui	
SDK pour Java 1.x	Nor	
SDK pour 3.x JavaScript	Oui	
SDK pour 2.x JavaScript	Nor	
SDK pour Kotlin	Oui	
SDK pour .NET 4.x	Oui	
SDK pour .NET 3.x	Oui	
SDK pour PHP 3.x	Oui	
SDK pour Python (Boto3)	Oui	
SDK pour Ruby 3.x	Oui	
SDK pour Rust	Oui	
SDK pour Swift	Oui	
Outils pour PowerShell V5	Oui	
Outils pour PowerShell V4	Oui	

Identifiants pour les points de terminaison spécifiques au service

Pour plus d'informations sur comment et où utiliser les identifiants du tableau suivant, reportez-vous Points de terminaison spécifiques au service à.

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
AccessAnalyzer	ac ly	AWS_ENDPOINT_URL_ACCESSANALYZER
Account	ac	AWS_ENDPOINT_URL_ACCOUNT
ACM	ac	AWS_ENDPOINT_URL_ACM
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

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
ApiGatewayManagementApi	ap yr nt	AWS_ENDPOINT_URL_APIGATEWAYMANAGEMENTAPI
ApiGatewayV2	ap y	AWS_ENDPOINT_URL_APIGATEWAYV2
AppConfig	ap	AWS_ENDPOINT_URL_APPCONFIG
AppConfigData	ap da	AWS_ENDPOINT_URL_APPCONFIGDATA
AppFabric	ap	AWS_ENDPOINT_URL_APPFABRIC
Appflow	ap	AWS_ENDPOINT_URL_APPFLOW
AppIntegrations	ap at	AWS_ENDPOINT_URL_APPINTEGRATIONS
Application Auto Scaling	ap or ca	AWS_ENDPOINT_URL_APPLICATION_AUTO_SCALING

serviceId	Cl d' ic de se pc A\ **cc** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
Application Insights	ap or ts	AWS_ENDPOINT_URL_APPLICATION_INSIGHTS
ApplicationCostProfiler	ap or f:	AWS_ENDPOINT_URL_APPLICATIONCOSTPROFILER
App Mesh	ap	AWS_ENDPOINT_URL_APP_MESH
AppRunner	ap	AWS_ENDPOINT_URL_APPRUNNER
AppStream	ap	AWS_ENDPOINT_URL_APPSTREAM
AppSync	ap	AWS_ENDPOINT_URL_APPSYNC
ARC Zonal Shift	a: _s	AWS_ENDPOINT_URL_ARC_ZONAL_SHIFT
Artifact	a:	AWS_ENDPOINT_URL_ARTIFACT
Athena	at	AWS_ENDPOINT_URL_ATHENA
AuditManager	au ge	AWS_ENDPOINT_URL_AUDITMANAGER

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
Auto Scaling	au ir	AWS_ENDPOINT_URL_AUTO_SCALING
Auto Scaling Plans	au ir	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
BCM Data Exports	bc ex	AWS_ENDPOINT_URL_BCM_DATA_EXPORTS
Bedrock	be	AWS_ENDPOINT_URL_BEDROCK
Bedrock Agent	be ge	AWS_ENDPOINT_URL_BEDROCK_AGENT

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
Bedrock Agent Runtime	be ge ir	AWS_ENDPOINT_URL_BEDROCK_AGENT_RUNTIME
Bedrock Runtime	be ur	AWS_ENDPOINT_URL_BEDROCK_RUNTIME
billingconductor	b: nc	AWS_ENDPOINT_URL_BILLINGCONDUCTOR
Braket	b:	AWS_ENDPOINT_URL_BRAKET
Budgets	bu	AWS_ENDPOINT_URL_BUDGETS
Cost Explorer	cc 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

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
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
Chime SDK Voice	ch _v	AWS_ENDPOINT_URL_CHIME_SDK_VOICE
CleanRooms	cl s	AWS_ENDPOINT_URL_CLEANROOMS
CleanRoomsML	cl sr	AWS_ENDPOINT_URL_CLEANROOMSML
Cloud9	cl	AWS_ENDPOINT_URL_CLOUD9
CloudControl	cl rc	AWS_ENDPOINT_URL_CLOUDCONTROL

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
CloudDirectory	c ct	AWS_ENDPOINT_URL_CLOUDDIRECTORY
CloudFormation	c at	AWS_ENDPOINT_URL_CLOUDFORMATION
CloudFront	c t	AWS_ENDPOINT_URL_CLOUDFRONT
CloudFront KeyValueStore	c t_ es	AWS_ENDPOINT_URL_CLOUDFRONT_KEYVALUESTORE
CloudHSM	c	AWS_ENDPOINT_URL_CLOUDHSM
CloudHSM V2	c v2	AWS_ENDPOINT_URL_CLOUDHSM_V2
CloudSearch	c ch	AWS_ENDPOINT_URL_CLOUDSEARCH
CloudSearch Domain	c ch	AWS_ENDPOINT_URL_CLOUDSEARCH_DOMAIN

serviceId	Cl d' ic de se pc A **cc** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
CloudTrail	cl	AWS_ENDPOINT_URL_CLOUDTRAIL
CloudTrail Data	cl_	AWS_ENDPOINT_URL_CLOUDTRAIL_DATA
CloudWatch	ch	AWS_ENDPOINT_URL_CLOUDWATCH
codeartifact	ca	AWS_ENDPOINT_URL_CODEARTIFACT
CodeBuild	co	AWS_ENDPOINT_URL_CODEBUILD
CodeCatalyst	co ys	AWS_ENDPOINT_URL_CODECATALYST
CodeCommit	co t	AWS_ENDPOINT_URL_CODECOMMIT
CodeDeploy	co y	AWS_ENDPOINT_URL_CODEDEPLOY
CodeGuru Reviewer	co re	AWS_ENDPOINT_URL_CODEGURU_REVIEWER

serviceId	Cl d' ic de se pc A **cc** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
CodeGuru Security	cc se	AWS_ENDPOINT_URL_CODEGURU_SECURITY
CodeGuruProfiler	cc rc	AWS_ENDPOINT_URL_CODEGURUPROFILER
CodePipeline	cc ir	AWS_ENDPOINT_URL_CODEPIPELINE
CodeStar	cc	AWS_ENDPOINT_URL_CODESTAR
CodeStar connections	cc cc ns	AWS_ENDPOINT_URL_CODESTAR_CONNECTIONS
codestar notifications	cc nc ic	AWS_ENDPOINT_URL_CODESTAR_NOTIFICATIONS
Cognito Identity	cc de	AWS_ENDPOINT_URL_COGNITO_IDENTITY
Cognito Identity Provider	cc de rc	AWS_ENDPOINT_URL_COGNITO_IDENTITY_PROVIDER

serviceId	Cl d' ic: de se pc A\ **cc** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
Cognito Sync	cc yr	AWS_ENDPOINT_URL_COGNITO_SYNC
Comprehend	cc d	AWS_ENDPOINT_URL_COMPREHEND
ComprehendMedical	cc dr	AWS_ENDPOINT_URL_COMPREHENDMEDICAL
Compute Optimizer	cc pt	AWS_ENDPOINT_URL_COMPUTE_OPTIMIZER
Config Service	cc rv	AWS_ENDPOINT_URL_CONFIG_SERVICE
Connect	cc	AWS_ENDPOINT_URL_CONNECT
Connect Contact Lens	cc or ns	AWS_ENDPOINT_URL_CONNECT_CONTACT_LENS
ConnectCampaigns	cc mp	AWS_ENDPOINT_URL_CONNECTCAMPAIGNS
ConnectCases	cc se	AWS_ENDPOINT_URL_CONNECTCASES

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
ConnectParticipant	cc rt	AWS_ENDPOINT_URL_CONNECTPARTICIPANT
ControlTower	cc we	AWS_ENDPOINT_URL_CONTROLTOWER
Cost Optimization Hub	cc m hu	AWS_ENDPOINT_URL_COST_OPTIMIZATION_HUB
Cost and Usage Report Service	cc us o ce	AWS_ENDPOINT_URL_COST_AND_USAGE_REPORT_SERVICE
Customer Profiles	cu p	AWS_ENDPOINT_URL_CUSTOMER_PROFILES
DataBrew	da	AWS_ENDPOINT_URL_DATABREW
DataExchange	da ng	AWS_ENDPOINT_URL_DATAEXCHANGE
Data Pipeline	da l	AWS_ENDPOINT_URL_DATA_PIPELINE

serviceId	Cl d' ic de se pc A **ce** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
DataSync	da	AWS_ENDPOINT_URL_DATASYNC
DataZone	da	AWS_ENDPOINT_URL_DATAZONE
DAX	da	AWS_ENDPOINT_URL_DAX
Detective	de	AWS_ENDPOINT_URL_DETECTIVE
Device Farm	de rn	AWS_ENDPOINT_URL_DEVICE_FARM
DevOps Guru	de ru	AWS_ENDPOINT_URL_DEVOPS_GURU
Direct Connect	di nn	AWS_ENDPOINT_URL_DIRECT_CONNECT
Application Discovery Service	ap or e: ce	AWS_ENDPOINT_URL_APPLICATION_DISCOVERY_SERVICE
DLM	dl	AWS_ENDPOINT_URL_DLM

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
Database Migration Service	da m _	AWS_ENDPOINT_URL_DATABASE_MIGRATION_SERVICE
DocDB	dc	AWS_ENDPOINT_URL_DOCDB
DocDB Elastic	dc st	AWS_ENDPOINT_URL_DOCDB_ELASTIC
drs	dr	AWS_ENDPOINT_URL_DRS
Directory Service	di _s	AWS_ENDPOINT_URL_DIRECTORY_SERVICE
DynamoDB	dy	AWS_ENDPOINT_URL_DYNAMODB
DynamoDB Streams	dy st	AWS_ENDPOINT_URL_DYNAMODB_STREAMS
EBS	eb	AWS_ENDPOINT_URL_EBS
EC2	ec	AWS_ENDPOINT_URL_EC2
EC2 Instance Connect	ec ne ct	AWS_ENDPOINT_URL_EC2_INSTANCE_CONNECT

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
ECR	ec	AWS_ENDPOINT_URL_ECR
ECR PUBLIC	ec c	AWS_ENDPOINT_URL_ECR_PUBLIC
ECS	ec	AWS_ENDPOINT_URL_ECS
EFS	ef	AWS_ENDPOINT_URL_EFS
EKS	el	AWS_ENDPOINT_URL_EKS
EKS Auth	el	AWS_ENDPOINT_URL_EKS_AUTH
Elastic Inference	el nf	AWS_ENDPOINT_URL_ELASTIC_INFERENCE
ElastiCache	el he	AWS_ENDPOINT_URL_ELASTICACHE
Elastic Beanstalk	el ea	AWS_ENDPOINT_URL_ELASTIC_BEANSTALK
Elastic Transcoder	el ra	AWS_ENDPOINT_URL_ELASTIC_TRANSCODER

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_ SERVICE>**
Elastic Load Balancing	e o c	AWS_ENDPOINT_URL_ELASTIC_LOAD_BALANCING
Elastic Load Balancing v2	e o c	AWS_ENDPOINT_URL_ELASTIC_LOAD_BALANCING_V2
EMR	er	AWS_ENDPOINT_URL_EMR
EMR containers	er ir	AWS_ENDPOINT_URL_EMR_CONTAINERS
EMR Serverless	er r	AWS_ENDPOINT_URL_EMR_SERVERLESS
EntityResolution	er o	AWS_ENDPOINT_URL_ENTITYRESOLUTION
Elasticsearch Service	e a i	AWS_ENDPOINT_URL_ELASTICSEARCH_SERVICE
EventBridge	ev ge	AWS_ENDPOINT_URL_EVENTBRIDGE

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
Evidently	ev	AWS_ENDPOINT_URL_EVIDENTLY
finspace	fi	AWS_ENDPOINT_URL_FINSPACE
finspace data	fi da	AWS_ENDPOINT_URL_FINSPACE_DATA
Firehose	fi	AWS_ENDPOINT_URL_FIREHOSE
fis	fi	AWS_ENDPOINT_URL_FIS
FMS	fr	AWS_ENDPOINT_URL_FMS
forecast	fo	AWS_ENDPOINT_URL_FORECAST
forecastquery	fo ue	AWS_ENDPOINT_URL_FORECASTQUERY
FraudDetector	fi ct	AWS_ENDPOINT_URL_FRAUDDETECTOR
FreeTier	fi	AWS_ENDPOINT_URL_FREETIER
FSx	fs	AWS_ENDPOINT_URL_FSX
GameLift	ga	AWS_ENDPOINT_URL_GAMELIFT

serviceId	Cl d' ic de se pc A **ce** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
Glacier	gl	AWS_ENDPOINT_URL_GLACIER
Global Accelerator	gl ce	AWS_ENDPOINT_URL_GLOBAL_ACCELERATOR
Glue	gl	AWS_ENDPOINT_URL_GLUE
grafana	gr	AWS_ENDPOINT_URL_GRAFANA
Greengrass	gr s	AWS_ENDPOINT_URL_GREENGRASS
GreengrassV2	gr sv	AWS_ENDPOINT_URL_GREENGRASSV2
GroundStation	gr t:	AWS_ENDPOINT_URL_GROUNDSTATION
GuardDuty	gu	AWS_ENDPOINT_URL_GUARDDUTY
Health	he	AWS_ENDPOINT_URL_HEALTH
HealthLake	he e	AWS_ENDPOINT_URL_HEALTHLAKE

serviceId	Cl d' ic de se pc A\ **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
Honeycode	ho	AWS_ENDPOINT_URL_HONEYCODE
IAM	ia	AWS_ENDPOINT_URL_IAM
identitystore	id to	AWS_ENDPOINT_URL_IDENTITYSTORE
imagebuilder	im de	AWS_ENDPOINT_URL_IMAGEBUILDER
ImportExport	im ox	AWS_ENDPOINT_URL_IMPORTEXPORT
Inspector	in	AWS_ENDPOINT_URL_INSPECTOR
Inspector Scan	in _s	AWS_ENDPOINT_URL_INSPECTOR_SCAN
Inspector2	in 2	AWS_ENDPOINT_URL_INSPECTOR2
InternetMonitor	in or	AWS_ENDPOINT_URL_INTERNETMONITOR
IoT	io	AWS_ENDPOINT_URL_IOT

serviceId	Cl d' ic de se pc A **cc** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
IoT Data Plane	ic p.	AWS_ENDPOINT_URL_IOT_DATA_PLANE
IoT Jobs Data Plane	ic da e	AWS_ENDPOINT_URL_IOT_JOBS_DATA_PLANE
IoT 1Click Devices Service	ic k_ _S	AWS_ENDPOINT_URL_IOT_1CLICK_DEVICES_SERVICE
IoT 1Click Projects	ic k_ s	AWS_ENDPOINT_URL_IOT_1CLICK_PROJECTS
IoTAnalytics	ic ic	AWS_ENDPOINT_URL_IOTANALYTICS
IotDeviceAdvisor	ic ad	AWS_ENDPOINT_URL_IOTDEVICEADVISOR
IoT Events	ic s	AWS_ENDPOINT_URL_IOT_EVENTS
IoT Events Data	ic s_	AWS_ENDPOINT_URL_IOT_EVENTS_DATA

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
IoTFleetHub	ic ul	AWS_ENDPOINT_URL_IOTFLEETHUB
IoTFleetWise	ic is	AWS_ENDPOINT_URL_IOTFLEETWISE
IoTSecureTunneling	ic tu	AWS_ENDPOINT_URL_IOTSECURETUNNELING
IoTSiteWise	ic se	AWS_ENDPOINT_URL_IOTSITEWISE
IoTThingsGraph	ic g:	AWS_ENDPOINT_URL_IOTTHINGSGRAPH
IoTTwinMaker	ic ke	AWS_ENDPOINT_URL_IOTTWINMAKER
IoT Wireless	ic es	AWS_ENDPOINT_URL_IOT_WIRELESS
ivs	iv	AWS_ENDPOINT_URL_IVS
IVS RealTime	iv ir	AWS_ENDPOINT_URL_IVS_REALTIME

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
ivschat	iv	AWS_ENDPOINT_URL_IVSCHAT
Kafka	ka	AWS_ENDPOINT_URL_KAFKA
KafkaConnect	ka ec	AWS_ENDPOINT_URL_KAFKACONNECT
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 id iv a	AWS_ENDPOINT_URL_KINESIS_VIDEO_ARCHIVED_MEDIA
Kinesis Video Media	ki id a	AWS_ENDPOINT_URL_KINESIS_VIDEO_MEDIA

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
Kinesis Video Signaling	k ic a	AWS_ENDPOINT_URL_KINESIS_VIDEO_SIGNALING
Kinesis Video WebRTC Storage	k ic t e	AWS_ENDPOINT_URL_KINESIS_VIDEO_WEBRTC_STORAGE
Kinesis Analytics	k na	AWS_ENDPOINT_URL_KINESIS_ANALYTICS
Kinesis Analytics V2	k na v	AWS_ENDPOINT_URL_KINESIS_ANALYTICS_V2
Kinesis Video	k ic	AWS_ENDPOINT_URL_KINESIS_VIDEO
KMS	kr	AWS_ENDPOINT_URL_KMS
LakeFormation	la t	AWS_ENDPOINT_URL_LAKEFORMATION
Lambda	la	AWS_ENDPOINT_URL_LAMBDA

serviceId	Cl d' ic de se pc A **ce** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
Launch Wizard	la za	AWS_ENDPOINT_URL_LAUNCH_WIZARD
Lex Model Building Service	le _l _s	AWS_ENDPOINT_URL_LEX_MODEL_BUILDING_SERVICE
Lex Runtime Service	le me e	AWS_ENDPOINT_URL_LEX_RUNTIME_SERVICE
Lex Models V2	le s_	AWS_ENDPOINT_URL_LEX_MODELS_V2
Lex Runtime V2	le me	AWS_ENDPOINT_URL_LEX_RUNTIME_V2
License Manager	li ar	AWS_ENDPOINT_URL_LICENSE_MANAGER
License Manager Linux Subscriptions	li ar nu r:	AWS_ENDPOINT_URL_LICENSE_MANAGER_LINUX_SUBSCRIPTIONS

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
License Manager User Subscriptions	l a e i	AWS_ENDPOINT_URL_LICENSE_MANAGER_USER_SUBSCRIPTIONS
Lightsail	l	AWS_ENDPOINT_URL_LIGHTSAIL
Location	l	AWS_ENDPOINT_URL_LOCATION
CloudWatch Logs	c h	AWS_ENDPOINT_URL_CLOUDWATCH_LOGS
LookoutEquipment	l u	AWS_ENDPOINT_URL_LOOKOUTEQUIPMENT
LookoutMetrics	l t	AWS_ENDPOINT_URL_LOOKOUTMETRICS
LookoutVision	l s	AWS_ENDPOINT_URL_LOOKOUTVISION
m2	m	AWS_ENDPOINT_URL_M2
Machine Learning	m e	AWS_ENDPOINT_URL_MACHINE_LEARNING

serviceId	Cl d' ic de se pc A **ce** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
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
Marketplace Catalog	ma ce g	AWS_ENDPOINT_URL_MARKETPLACE_CATALOG
Marketplace Deployment	ma ce me	AWS_ENDPOINT_URL_MARKETPLACE_DEPLOYMENT
Marketplace Entitlement Service	ma ce er v:	AWS_ENDPOINT_URL_MARKETPLACE_ENTITLEMENT_SERVICE

serviceId	Cl d' ic de se pc A **ce** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
Marketplace Commerce Analytics	ma ce ce ic	AWS_ENDPOINT_URL_MARKETPLACE_COMMERCE_ANALYTICS
MediaConnect	me ec	AWS_ENDPOINT_URL_MEDIACONNECT
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

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
MediaStore Data	me e_	AWS_ENDPOINT_URL_MEDIASTORE_DATA
MediaTailor	me o:	AWS_ENDPOINT_URL_MEDIATAILOR
Medical Imaging	me ma	AWS_ENDPOINT_URL_MEDICAL_IMAGING
MemoryDB	me	AWS_ENDPOINT_URL_MEMORYDB
Marketplace Metering	ma ce ng	AWS_ENDPOINT_URL_MARKETPLACE_METERING
Migration Hub	m: _I	AWS_ENDPOINT_URL_MIGRATION_HUB
mgn	mg	AWS_ENDPOINT_URL_MGN
Migration Hub Refactor Spaces	m: _I ct es	AWS_ENDPOINT_URL_MIGRATION_HUB_REFAC TOR_SPACES

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
MigrationHub Config	m hu g	AWS_ENDPOINT_URL_MIGRATIONHUB_CONFIG
MigrationHubOrches trator	m hu t	AWS_ENDPOINT_URL_MIGRATIONHUBORCHESTRATOR
MigrationHubStrategy	m hu gy	AWS_ENDPOINT_URL_MIGRATIONHUBSTRATEGY
Mobile	mo	AWS_ENDPOINT_URL_MOBILE
mq	mo	AWS_ENDPOINT_URL_MQ
MTurk	mt	AWS_ENDPOINT_URL_MTURK
MWAA	mw	AWS_ENDPOINT_URL_MWAA
Neptune	ne	AWS_ENDPOINT_URL_NEPTUNE
Neptune Graph	ne ra	AWS_ENDPOINT_URL_NEPTUNE_GRAPH
neptunedata	ne ta	AWS_ENDPOINT_URL_NEPTUNEDATA

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
Network Firewall	ne 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
OpsWorksCM	op m	AWS_ENDPOINT_URL_OPSWORKSCM

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
Organizations	o ic	AWS_ENDPOINT_URL_ORGANIZATIONS
OSIS	os	AWS_ENDPOINT_URL_OSIS
Outposts	ou	AWS_ENDPOINT_URL_OUTPOSTS
p8data	p8	AWS_ENDPOINT_URL_P8DATA
p8data	p8	AWS_ENDPOINT_URL_P8DATA
Panorama	pa	AWS_ENDPOINT_URL_PANORAMA
Payment Cryptography	pa ry hy	AWS_ENDPOINT_URL_PAYMENT_CRYPTOGRAPHY
Payment Cryptography Data	pa ry hy	AWS_ENDPOINT_URL_PAYMENT_CRYPTOGRAPHY_DATA
Pca Connector Ad	pc ct	AWS_ENDPOINT_URL_PCA_CONNECTOR_AD
Personalize	pe ze	AWS_ENDPOINT_URL_PERSONALIZE

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
Personalize Events	pe ze	AWS_ENDPOINT_URL_PERSONALIZE_EVENTS
Personalize Runtime	pe ze e	AWS_ENDPOINT_URL_PERSONALIZE_RUNTIME
PI	p:	AWS_ENDPOINT_URL_PI
Pinpoint	p:	AWS_ENDPOINT_URL_PINPOINT
Pinpoint Email	p: er	AWS_ENDPOINT_URL_PINPOINT_EMAIL
Pinpoint SMS Voice	p: sr	AWS_ENDPOINT_URL_PINPOINT_SMS_VOICE
Pinpoint SMS Voice V2	p: sr _	AWS_ENDPOINT_URL_PINPOINT_SMS_VOICE_V2
Pipes	p:	AWS_ENDPOINT_URL_PIPES
Polly	pc	AWS_ENDPOINT_URL_POLLY

serviceId	Cl d' ic de se pc A\ **cc** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
Pricing	p:	AWS_ENDPOINT_URL_PRICING
PrivateNetworks	p: tv	AWS_ENDPOINT_URL_PRIVATENETWORKS
Proton	p:	AWS_ENDPOINT_URL_PROTON
QBusiness	qb	AWS_ENDPOINT_URL_QBUSINESS
QConnect	qc	AWS_ENDPOINT_URL_QCONNECT
QLDB	q:	AWS_ENDPOINT_URL_QLDB
QLDB Session	q: ic	AWS_ENDPOINT_URL_QLDB_SESSION
QuickSight	qu t	AWS_ENDPOINT_URL_QUICKSIGHT
RAM	ra	AWS_ENDPOINT_URL_RAM
rbin	rb	AWS_ENDPOINT_URL_RBIN
RDS	rc	AWS_ENDPOINT_URL_RDS
RDS Data	rc	AWS_ENDPOINT_URL_RDS_DATA

serviceId	Cl d' ic de se pc A **ce** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
Redshift	re	AWS_ENDPOINT_URL_REDSHIFT
Redshift Data	re da	AWS_ENDPOINT_URL_REDSHIFT_DATA
Redshift Serverless	re se s	AWS_ENDPOINT_URL_REDSHIFT_SERVERLESS
Rekognition	re on	AWS_ENDPOINT_URL_REKOGNITION
repostspace	re ce	AWS_ENDPOINT_URL_REPOSTSPACE
resiliencehub	re eh	AWS_ENDPOINT_URL_RESILIENCEHUB
Resource Explorer 2	re ex 2	AWS_ENDPOINT_URL_RESOURCE_EXPLORER_2
Resource Groups	re gr	AWS_ENDPOINT_URL_RESOURCE_GROUPS

serviceId	Cl d' ic de se pc A\ **cc** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
Resource Groups Tagging API	re g: g	AWS_ENDPOINT_URL_RESOURCE_GROUPS_TAGGING_API
RoboMaker	rc	AWS_ENDPOINT_URL_ROBOMAKER
RolesAnywhere	rc he	AWS_ENDPOINT_URL_ROLESANYWHERE
Route 53	rc	AWS_ENDPOINT_URL_ROUTE_53
Route53 Recovery Cluster	rc ec lu	AWS_ENDPOINT_URL_ROUTE53_RECOVERY_CLUSTER
Route53 Recovery Control Config	rc ec or n:	AWS_ENDPOINT_URL_ROUTE53_RECOVERY_CONTROL_CONFIG
Route53 Recovery Readiness	rc ec ea	AWS_ENDPOINT_URL_ROUTE53_RECOVERY_READINESS

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
Route 53 Domains	rc dc	AWS_ENDPOINT_URL_ROUTE_53_DOMAINS
Route53Resolver	rc sc	AWS_ENDPOINT_URL_ROUTE53RESOLVER
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	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
SageMaker FeatureSt ore Runtime	sa _ t ir	AWS_ENDPOINT_URL_SAGEMAKER_FEATUREST ORE_RUNTIME
SageMaker Geospatial	sa _ a	AWS_ENDPOINT_URL_SAGEMAKER_GEOSPATIAL
SageMaker Metrics	sa _	AWS_ENDPOINT_URL_SAGEMAKER_METRICS
SageMaker Runtime	sa _	AWS_ENDPOINT_URL_SAGEMAKER_RUNTIME
savingsplans	sa a	AWS_ENDPOINT_URL_SAVINGSPLANS
Scheduler	s	AWS_ENDPOINT_URL_SCHEDULER
schemas	s	AWS_ENDPOINT_URL_SCHEMAS
SimpleDB	s	AWS_ENDPOINT_URL_SIMPLEDB

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
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 ic to	AWS_ENDPOINT_URL_SERVERLESSAPPLICATIONREPOSITORY
Service Quotas	se uc	AWS_ENDPOINT_URL_SERVICE_QUOTAS
Service Catalog	se at	AWS_ENDPOINT_URL_SERVICE_CATALOG
Service Catalog AppRegistry	se at p:	AWS_ENDPOINT_URL_SERVICE_CATALOG_APPREGISTRY

serviceId	Cl d' ic de se pc A **ce** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
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
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

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
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
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

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
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
Timestream InfluxDB	ti m_ b	AWS_ENDPOINT_URL_TIMESTREAM_INFLUXDB
Timestream Query	ti m_	AWS_ENDPOINT_URL_TIMESTREAM_QUERY
Timestream Write	ti m_	AWS_ENDPOINT_URL_TIMESTREAM_WRITE
tnb	tn	AWS_ENDPOINT_URL_TNB
Transcribe	ti e	AWS_ENDPOINT_URL_TRANSCRIBE
Transfer	ti	AWS_ENDPOINT_URL_TRANSFER

serviceId	Cl d' ic de se pc A **ce** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
Translate	t:	AWS_ENDPOINT_URL_TRANSLATE
TrustedAdvisor	t: v:	AWS_ENDPOINT_URL_TRUSTEDADVISOR
VerifiedPermissions	ve e: s	AWS_ENDPOINT_URL_VERIFIEDPERMISSIONS
Voice ID	vc	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	w:	AWS_ENDPOINT_URL_WISDOM

serviceId	Cl d' ic de se pc A **c** le fic pa	variable d'environnement **AWS_ENDPOINT_URL_<SERVICE>**
WorkDocs	wc	AWS_ENDPOINT_URL_WORKDOCS
WorkLink	wc	AWS_ENDPOINT_URL_WORKLINK
WorkMail	wc	AWS_ENDPOINT_URL_WORKMAIL
WorkMailMessageFlow	wc es w	AWS_ENDPOINT_URL_WORKMAILMESSAGEFLOW
WorkSpaces	wc s	AWS_ENDPOINT_URL_WORKSPACES
WorkSpaces Thin Client	wc s_ ie	AWS_ENDPOINT_URL_WORKSPACES_THIN_CLIENT
WorkSpaces Web	wc s_	AWS_ENDPOINT_URL_WORKSPACES_WEB
XRay	x:	AWS_ENDPOINT_URL_XRAY

Paramètres de configuration intelligents par défaut

> **Note**
>
> Pour vous aider à comprendre la mise en page des pages de paramètres ou à interpréter le tableau Support by AWS SDKs et outils ci-dessous, voirComprendre les pages de paramètres de ce guide.

Grâce à la fonctionnalité de configuration intelligente par défaut, AWS SDKs vous pouvez fournir des valeurs par défaut prédéfinies et optimisées pour d'autres paramètres de configuration.

Configurez cette fonctionnalité à l'aide des méthodes suivantes :

`defaults_mode`- réglage AWS **`config`** du fichier partagé, **`AWS_DEFAULTS_MODE`**- variable d'environnement, **`aws.defaultsMode`**- Propriété du système JVM : uniquement Java/Kotlin

Avec ce paramètre, vous pouvez choisir un mode qui s'aligne sur l'architecture de votre application, qui fournit ensuite des valeurs par défaut optimisées pour votre application. Si une valeur est explicitement définie pour un paramètre du AWS SDK, cette valeur est toujours prioritaire. Si aucune valeur n'est définie explicitement pour un paramètre du AWS SDK et qu'`defaults_mode`il n'est pas égal à l'ancien paramètre, cette fonctionnalité peut fournir des valeurs par défaut différentes pour différents paramètres optimisés pour votre application. Les paramètres peuvent inclure les éléments suivants : les paramètres de communication HTTP, le comportement des nouvelles tentatives, les paramètres du point de terminaison régional du service et, éventuellement, toute configuration liée au SDK. Les clients qui utilisent cette fonctionnalité peuvent obtenir de nouvelles configurations par défaut adaptées aux scénarios d'utilisation courants. Si vous `defaults_mode` n'êtes pas égal à`legacy`, nous vous recommandons de tester votre application lors de la mise à niveau du SDK, car les valeurs par défaut fournies peuvent changer en fonction de l'évolution des meilleures pratiques.

Valeur par défaut : `legacy`

Remarque : les nouvelles versions majeures de SDKs seront définies par défaut sur`standard`.

Valeurs valides:

- `legacy`— Fournit des paramètres par défaut qui varient selon le SDK et qui existaient avant la création de`defaults_mode`.

- `standard`— Fournit les dernières valeurs par défaut recommandées qui devraient pouvoir être exécutées en toute sécurité dans la plupart des scénarios.
- `in-region`— S'appuie sur le mode standard et inclut une optimisation adaptée aux applications qui appellent Services AWS depuis le même mode Région AWS.
- `cross-region`— S'appuie sur le mode standard et inclut une optimisation adaptée aux applications faisant appel Services AWS à une région différente.
- `mobile`— S'appuie sur le mode standard et inclut une optimisation adaptée aux applications mobiles.
- `auto` : s'appuie sur le mode standard et inclut des fonctionnalités expérimentales. Le kit SDK tente de découvrir l'environnement d'exécution afin de déterminer automatiquement les paramètres appropriés. La détection automatique est basée sur l'heuristique et ne fournit pas une précision de 100 %. Si l'environnement d'exécution ne peut pas être déterminé, `standard` le mode est utilisé. La détection automatique peut interroger les métadonnées de l'instance, ce qui peut introduire de la latence. Si la latence de démarrage est essentielle pour votre application, nous vous recommandons de choisir plutôt un `defaults_mode` explicite.

Exemple de définition de cette valeur dans le `config` fichier :

```
[default]
defaults_mode = standard
```

Les paramètres suivants peuvent être optimisés en fonction de la sélection de `defaults_mode` :

- `retryMode`— Spécifie la manière dont le SDK tente de réessayer. Consultez Comportement de nouvelle tentative.
- `stsRegionalEndpoints`— Spécifie la manière dont le SDK détermine le Service AWS point de terminaison qu'il utilise pour communiquer avec le AWS Security Token Service (AWS STS). Consultez AWS STS Points de terminaison régionaux.
- `s3UsEast1RegionalEndpoints`— Spécifie la manière dont le SDK détermine le point de terminaison du AWS service qu'il utilise pour communiquer avec Amazon S3 pour la `us-east-1` région.
- `connectTimeoutInMillis`— Après avoir effectué une première tentative de connexion sur un socket, délai avant l'expiration du délai imparti. Si le client ne reçoit pas de confirmation de connexion terminée, il abandonne l'opération et échoue.

- `tlsNegotiationTimeoutInMillis`— Le temps maximum qu'une poignée de main TLS peut prendre entre le moment où le message CLIENT HELLO est envoyé et le moment où le client et le serveur ont entièrement négocié les chiffrements et échangé les clés.

La valeur par défaut de chaque paramètre change en fonction de `defaults_mode` celui sélectionné pour votre application. Ces valeurs sont actuellement définies comme suit (sous réserve de modifications) :

Paramètre	Mode **`standard`**	Mode **`in-region`**	Mode **`cross-region`**	Mode **`mobile`**
`retryMode`	`standard`	`standard`	`standard`	`standard`
`stsRegionalEndpoints`	`regional`	`regional`	`regional`	`regional`
`s3UsEast1RegionalEndpoints`	`regional`	`regional`	`regional`	`regional`
`connectTimeoutInMillis`	3100	1100	3100	30 000
`tlsNegotiationTimeoutInMillis`	3100	1100	3100	30 000

Par exemple, si `defaults_mode` vous avez sélectionné l'était`standard`, la valeur de `standard` serait attribuée à `retry_mode` (à partir des `retry_mode` options valides) et la valeur de `regional` serait affectée à `stsRegionalEndpoints` (à partir des `stsRegionalEndpoints` options valides).

Support par AWS SDKs et outils

Les éléments suivants SDKs prennent en charge les fonctionnalités et les paramètres décrits dans cette rubrique. Toute exception partielle est notée. Tous les paramètres de propriété du système JVM sont pris en charge par le AWS SDK pour Java et le AWS SDK pour Kotlin seul.

Kit SDK	Pris en charge	Remarques ou informations supplémentaires
AWS CLI v2	Non	
SDK pour C++	Oui	Paramètres non optimisés : `stsRegionalEndpoints`, `s3UsEast1RegionalEndpoints`, `tlsNegotiationTimeoutInMillis`.
SDK pour Go V2 (1.x)	Oui	Paramètres non optimisés : `retryMode`, `stsRegionalEndpoints`, `s3UsEast1RegionalEndpoints`.
SDK pour Go 1.x (V1)	Non	
SDK pour Java 2.x	Oui	Paramètres non optimisés : `stsRegionalEndpoints`.
SDK pour Java 1.x	Non	
SDK pour 3.x JavaScript	Oui	Paramètres non optimisés : `stsRegionalEndpoints`, `s3UsEast1RegionalEndpoints`, `tlsNegotiationTimeoutInMill`

Kit SDK	Pris en charge	Remarques ou informations supplémentaires
		`is .connectTimeoutInMillis` est appelé `connectionTimeout .`
SDK pour 2.x JavaScript	Non	
SDK pour Kotlin	Non	
SDK pour .NET 4.x	Oui	Paramètres non optimisés : `connectTimeoutInMillis` , `tlsNegotiationTimeoutInMillis .`
SDK pour .NET 3.x	Oui	Paramètres non optimisés : `connectTimeoutInMillis` , `tlsNegotiationTimeoutInMillis .`
SDK pour PHP 3.x	Oui	Paramètres non optimisés : `tlsNegotiationTimeoutInMillis .`
SDK pour Python (Boto3)	Oui	Paramètres non optimisés : `tlsNegotiationTimeoutInMillis .`
SDK pour Ruby 3.x	Oui	
SDK pour Rust	Non	

Kit SDK	Pris en charge	Remarques ou informations supplémentaires
SDK pour Swift	Non	
Outils pour PowerShell V5	Oui	Paramètres non optimisés : `connectTimeoutInMillis`, `tlsNegotiationTimeoutInMillis`.
Outils pour PowerShell V4	Oui	Paramètres non optimisés : `connectTimeoutInMillis`, `tlsNegotiationTimeoutInMillis`.

AWS Paramètres de sécurité des SDK et des outils

Cette section traite de certains paramètres de sécurité communs à tous les AWS SDK. Toutefois, pour obtenir des informations complètes sur la sécurité des différents AWS SDK, reportez-vous au chapitre consacré à la sécurité dans le guide du développeur du AWS SDK correspondant :

- AWS Chapitre sur la sécurité du SDK for C++
- AWS Chapitre sur le SDK for Go Security
- AWS Chapitre sur la sécurité du SDK for Java
- AWS Chapitre du SDK pour JavaScript la sécurité
- AWS Chapitre du SDK pour Kotlin Security
- AWS Chapitre consacré à la sécurité du SDK for .NET
- AWS Chapitre sur la sécurité du SDK for PHP
- AWS Chapitre sur la sécurité du SDK pour Python
- AWS Chapitre sur le SDK pour Ruby Security
- AWS Chapitre du SDK pour Rust Security
- AWS Chapitre du SDK pour Swift Security

Rubriques

Activer le TLS post-quantique hybride

AWS Les SDK et les outils ont des capacités et une configuration cryptographiques qui diffèrent selon le langage et le moteur d'exécution. Un AWS SDK ou un outil fournit actuellement le support PQ TLS de trois manières :

Rubriques

- AWS Les SDK et outils ne prévoient pas de prendre en charge le protocole PQ TLS

SDK avec PQ TLS activé par défaut

> **Note**
>
> À partir de 6Nov-2025, le AWS SDK et ses bibliothèques CRT sous-jacentes pour macOS et Windows utilisent des bibliothèques système pour TLS. Les capacités PQ TLS sur ces plateformes sont donc généralement déterminées par le support au niveau du système.

AWS SDK pour Go

Le AWS SDK pour Go utilise la propre implémentation TLS de Golang fournie par sa bibliothèque standard. Golang prend en charge et préfère PQ TLS à partir de la version v1.24. Les utilisateurs du SDK for AWS Go peuvent donc activer PQ TLS en mettant simplement à niveau Golang vers la version v1.24

AWS SDK pour JavaScript (navigateur)

Le AWS SDK pour JavaScript (navigateur) utilise la pile TLS du navigateur. Le SDK négociera donc le protocole TLS PQ si le moteur d'exécution du navigateur le prend en charge et le préfère. Firefox a lancé le support de PQ TLS dans la version 132.0. Chrome a annoncé la prise en charge de PQ TLS dans la version 131. Edge prend en charge le protocole PQ TLS opt-in dans la version 120 pour ordinateur de bureau et dans la version 140 pour Android.

AWS SDK pour Node.js

À partir de la Node.js version 22.20 (LTS) et de la version 24.9.0, OpenSSL 3.5 est Node.js lié et regroupe statiquement. Cela signifie que PQ TLS est activé et préféré par défaut pour ces versions et les suivantes.

AWS SDK pour Kotlin

Le SDK Kotlin prend en charge et préfère PQ TLS sous Linux à partir de la v1.5.78. Étant donné que le AWS SDK pour le CRT-based client de Kotlin repose sur des bibliothèques système pour TLS sous macOS et Windows, la prise en charge de PQ TLS dépendra de ces bibliothèques système sous-jacentes.

AWS SDK pour Rust

Le AWS SDK pour Rust distribue des packages distincts (appelés « caisses » dans l'écosystème Rust) pour chaque client de service. Ils sont tous gérés dans un GitHub référentiel consolidé, mais chaque client de service suit sa propre version et sa propre cadence de publication. Le SDK consolidé a publié la préférence PQ TLS sur 8/29 /25, de sorte que toute version de client de service individuelle publiée après cette date prendra en charge et préférera PQ TLS par défaut.

Vous pouvez déterminer la version minimale supportant le protocole PQ TLS pour un client de service donné en accédant à l'URL de la version de crates.io correspondante (par exemple, Crédit promotionnel AWS elle se trouve ici) et en recherchant la première version publiée après 29-. Aug-25 Toute version du client de service publiée après le 29- Aug-25 aura PQ TLS activé et préféré par défaut.

Opt-in Prise en charge du protocole PQ TLS

AWS SDK pour C++

Par défaut, le SDK C++ utilise des clients natifs de la plateforme tels que libcurl et. WinHttp Libcurl s'appuie généralement sur le système OpenSSL pour TLS, donc PQ TLS n'est activé par défaut que si le système OpenSSL est ≥ v3.5. Vous pouvez remplacer cette valeur par défaut dans le SDK C++ v1.11.673 ou version ultérieure, et opter pour le logiciel AwsCrtHttpClient qui prend en charge et active PQ TLS par défaut.

Remarques sur la création de TLS pour Opt-In PQ Vous pouvez récupérer les dépendances CRT du SDK à l'aide de ce script. La création du SDK à partir des sources est décrite ici et ici, mais notez que vous aurez peut-être besoin de quelques indicateurs CMake supplémentaires :

```
-DUSE_CRT_HTTP_CLIENT=ON \
-DUSE_TLS_V1_2=OFF \
-DUSE_TLS_V1_3=ON \
-DUSE_OPENSSL=OFF \
```

AWS SDK pour Java

Depuis la version 2, le AWS SDK for Java fournit AWS un client HTTP CRT AWS (Common Runtime) qui peut être configuré pour exécuter le protocole PQ TLS. Depuis la version 2.35.11, le AwsCrtHttpClient protocole TLS PQ est activé et préféré par défaut partout où il est utilisé.

SDK qui s'appuient sur le système OpenSSL

Plusieurs AWS kits de développement logiciel et outils dépendent de la libcrypto/libssl bibliothèque TLS du système. La bibliothèque système la plus souvent utilisée est OpenSSL. OpenSSL a activé le support PQ TLS dans la version 3.5. Le moyen le plus simple de configurer ces SDK et outils pour PQ TLS est donc de les utiliser sur une distribution de système d'exploitation sur laquelle OpenSSL 3.5 est installé au moins.

Vous pouvez également configurer un conteneur Docker pour utiliser OpenSSL 3.5 afin d'activer PQ TLS sur n'importe quel système prenant en charge Docker. Voir Post-quantum TLS en Python pour un exemple de configuration pour Python.

AWS INTERFACE DE LIGNE DE COMMANDE (CLI)

Depuis la version 2.34.54, le programme d'installation de la AWS CLI pour Linux intègre OpenSSL 3.5.6. PQ TLS est donc activé et préféré par défaut pour cette version et les versions suivantes sous Linux. AWS Les utilisateurs de la CLI sous Linux peuvent activer PQ TLS en effectuant une mise à niveau vers la version 2.34.54 ou ultérieure de la AWS CLI.

Pour macOS, installez la AWS CLI via Homebrew et assurez-vous que votre Homebrew-vended openSSL est mis à niveau vers la version 3.5+. Vous pouvez le faire avec « brew install openssl @3 .6 » et valider avec « brew list | grep openssl ».

Si vous ne pouvez pas effectuer la mise à niveau vers AWS CLI v2.34.54 ou version ultérieure sur Ubuntu ou Debian Linux, assurez-vous qu'OpenSSL 3.5+ est installé sur la distribution Linux que vous utilisez en tant que système OpenSSL. Ensuite, installez la AWS CLI en utilisant apt ou PyPI. Avec ces prérequis, la AWS CLI fournie par apt ou PyPI sera configurée pour négocier. PQ-TLS Pour obtenir des instructions étape par étape pour valider l'installation, consultez le référentiel github et le billet de blog qui l'accompagne.

AWS Kit SDK pour PHP

Le AWS SDK pour PHP repose sur le libssl/libcrypto système. Pour utiliser PQ TLS, utilisez ce SDK sur une distribution de système d'exploitation sur laquelle OpenSSL 3.5 est installé au moins.

AWS Kit SDK for Python (Boto3)

Le AWS SDK pour Python (Boto3) repose sur le système. libssl/libcrypto Pour utiliser PQ TLS, utilisez ce SDK sur une distribution de système d'exploitation sur laquelle OpenSSL 3.5 est installé au moins.

AWS Kit SDK pour Ruby

Le AWS SDK pour Ruby repose sur le libssl/libcrypto système. Pour utiliser PQ TLS, utilisez ce SDK sur une distribution de système d'exploitation sur laquelle OpenSSL 3.5 est installé au moins.

AWS SDK pour .NET

Sous Linux, le AWS SDK pour .NET repose sur le libssl/libcrypto système. Pour utiliser PQ TLS, utilisez ce SDK sur une distribution de système d'exploitation sur laquelle OpenSSL 3.5 est installé au moins. Sous Windows et macOS, PQ TLS est disponible à partir de .NET 10 et Windows 11. Sur macOS, la prise en charge du protocole TLS 1.3 (condition préalable au protocole PQ TLS) peut être activée en optant pour le protocole Apple, comme décrit ici. Network.framework En supposant une version .NET minimale de 10, PQ TLS devrait alors être activé.

AWS Les SDK et outils ne prévoient pas de prendre en charge le protocole PQ TLS

Il n'est actuellement pas prévu de prendre en charge les SDK et outils linguistiques suivants :

- AWS SDK pour SAP
- AWS SDK pour Swift
- AWS Outils pour Windows PowerShell

AWS bibliothèques CRT (Common Runtime)

Les bibliothèques AWS Common Runtime (CRT) sont une bibliothèque de base du SDKs. Le CRT est une famille modulaire de packages indépendants, écrits en C. Chaque package fournit de bonnes performances et un encombrement minimal pour les différentes fonctionnalités requises. Ces fonctionnalités sont communes et partagées par tous, ce SDKs qui permet une meilleure réutilisation, optimisation et précision du code. Les packages sont les suivants :

- `awslabs/aws-c-auth`: authentification AWS côté client (fournisseurs d'informations d'identification standard et signature (sigv4))
- `awslabs/aws-c-cal`: types primitifs cryptographiques, hachages (MD5,, SHA256 HMAC) SHA256, signataires, AES
- `awslabs/aws-c-common`: structures de données de base, types threading/synchronization primitifs, gestion de la mémoire tampon, fonctions liées à stdlib
- `awslabs/aws-c-compression`: Algorithmes de compression (codage/décodage Huffman)
- `awslabs/aws-c-event-stream`: traitement des messages des flux d'événements (en-têtes, prélude, charge utile, crc/trailer), implémentation des appels de procédure à distance (RPC) sur les flux d'événements
- `awslabs/aws-c-http`: Implémentation C99 des spécifications HTTP/1.1 et HTTP/2
- `awslabs/aws-c-io`: sockets (TCP, UDP), DNS, canaux, boucles d'événements, canaux, SSL/TLS
- `awslabs/aws-c-iot`: C99 Mise en œuvre de l'intégration des services cloud AWS IoT aux appareils
- `awslabs/aws-c-mqtt`: protocole de messagerie standard et léger pour l'Internet des objets (IoT)
- `awslabs/aws-c-s3`: implémentation de la bibliothèque C99 pour communiquer avec le service Amazon S3, conçue pour optimiser le débit sur les instances Amazon à bande passante élevée EC2
- `awslabs/aws-c-sdkutils`: bibliothèque d'utilitaires pour l'analyse et la gestion des profils AWS
- `awslabs/aws-checksums`: accélération matérielle multiplateforme et repli sur CRC32c des CRC32 implémentations logicielles efficaces
- `awslabs/aws-lc`: bibliothèque cryptographique à usage général gérée par l'équipe de AWS cryptographie pour AWS et ses clients, basée sur le code du projet Google BoringSSL et du projet OpenSSL

- `awslabs/s2n`: Implémentation C99 des protocoles TLS/SSL, conçus pour être petits et rapides avec la sécurité comme priorité

Le CRT est disponible partout SDKs sauf Go et Rust.

Dépendances CRT

Les bibliothèques CRT forment un réseau complexe de relations et de dépendances. Connaître ces relations est utile si vous devez créer le CRT directement à partir de la source. Cependant, la plupart des utilisateurs accèdent aux fonctionnalités CRT par le biais de leur SDK linguistique (tel que le AWS SDK pour C++ ou le SDK pour AWS Java) ou du SDK de leur appareil IoT (tel que le SDK IoT pour AWS C++ ou le SDK IoT pour Java). AWS Dans le schéma suivant, la zone Language CRT Bindings fait référence au package qui contient les bibliothèques CRT pour un SDK de langage spécifique. Il s'agit d'une collection de packages du format`aws-crt-*`, où « * » est un langage du SDK (tel que `aws-crt-cpp`ou `aws-crt-java`).

Voici une illustration des dépendances hiérarchiques des bibliothèques CRT.
Schéma de dépendance CRT montrant comment les bibliothèques CRT individuelles interagissent les unes avec les autres.

AWS SDKs et politique de maintenance des outils

Présentation de

Ce document décrit la politique de maintenance des kits de développement AWS logiciel (SDKs) et des outils, y compris le mobile et l'IoT SDKs, ainsi que leurs dépendances sous-jacentes. AWS fournit régulièrement aux outils AWS SDKs et aux outils des mises à jour qui peuvent inclure la prise en charge de nouvelles fonctionnalités ou mises à jour AWS APIs, de nouvelles fonctionnalités, d'améliorations, de corrections de bogues, de correctifs de sécurité ou de mises à jour de documentation. Les mises à jour peuvent également prendre en compte les modifications liées aux dépendances, aux environnements d'exécution des langues et aux systèmes d'exploitation. AWS Les versions du SDK sont publiées dans les gestionnaires de packages (par exemple Maven, NuGet PyPI) et sont disponibles sous forme de code source sur. GitHub

Nous recommandons aux utilisateurs de s'en tenir up-to-date aux versions du SDK pour rester au fait des dernières fonctionnalités, des mises à jour de sécurité et des dépendances sous-jacentes. L'utilisation continue d'une version du SDK non prise en charge n'est pas recommandée et est laissée à la discrétion de l'utilisateur.

Gestion des versions

Les versions du AWS SDK se présentent sous la forme de X.Y.Z où X représente la version majeure. L'augmentation de la version principale d'un SDK indique que ce SDK a subi des modifications importantes et substantielles pour prendre en charge les nouvelles expressions idiomatiques et les nouveaux modèles du langage. Les versions majeures sont introduites lorsque les interfaces publiques (par exemple, les classes, les méthodes, les types, etc.), les comportements ou la sémantique ont changé. Les applications doivent être mises à jour pour fonctionner avec la dernière version du SDK. Il est important de mettre à jour les versions majeures avec soin et conformément aux directives de mise à niveau fournies par AWS.

Cycle de vie des versions majeures du SDK

Le cycle de vie des versions majeures SDKs et des versions Tools comprend 5 phases, décrites ci-dessous.

- Version préliminaire pour les développeurs (phase 0) : pendant cette phase, SDKs ils ne sont pas pris en charge, ne doivent pas être utilisés dans des environnements de production et sont

uniquement destinés à un accès anticipé et à des fins de commentaires. Il est possible que les futures versions introduisent des modifications majeures. Une fois AWS qu'une version a été identifiée comme étant un produit stable, elle peut la marquer comme Release Candidate. Les versions candidates sont prêtes pour la version GA à moins que des bogues importants ne surviennent, et bénéficieront d'un AWS support complet.

- Disponibilité générale (GA) (phase 1) - Pendant cette phase, SDKs ils sont entièrement pris en charge. AWS fournira des versions régulières du SDK qui incluent la prise en charge de nouveaux services, des mises à jour d'API pour les services existants, ainsi que des correctifs de bogues et de sécurité. Pour les outils, AWS fournira des versions régulières incluant de nouvelles mises à jour de fonctionnalités et des corrections de bogues. AWS prendra en charge la version GA d'un SDK pendant au moins 24 mois.
- Annonce de maintenance (phase 2) : AWS fera une annonce publique au moins 6 mois avant qu'un SDK ne passe en mode maintenance. Pendant cette période, le SDK continuera d'être entièrement pris en charge. Généralement, le mode maintenance est annoncé au moment où la prochaine version majeure passe en GA.
- Maintenance (phase 3) - Pendant le mode maintenance, AWS limite les versions du SDK pour résoudre les corrections de bogues critiques et les problèmes de sécurité uniquement. Un SDK ne recevra pas de mises à jour d'API pour les services nouveaux ou existants, ni ne sera mis à jour pour prendre en charge de nouvelles régions. Le mode maintenance a une durée par défaut de 12 mois, sauf indication contraire.
- End-of-Support (Phase 4) - Lorsqu'un SDK arrive à la fin du support, il ne reçoit plus de mises à jour ni de versions. Les versions précédemment publiées continueront d'être disponibles via les gestionnaires de packages publics et le code restera activé GitHub. Le GitHub référentiel peut être archivé. L'utilisation d'un SDK atteint end-of-support est laissée à la discrétion de l'utilisateur. Nous recommandons aux utilisateurs de passer à la nouvelle version majeure.

Voici une illustration visuelle du cycle de vie des versions majeures du SDK. Veuillez noter que les délais indiqués ci-dessous sont illustratifs et ne sont pas contraignants.
Délais de la politique de maintenance

Cycle de vie des dépendances

La plupart AWS SDKs ont des dépendances sous-jacentes, telles que des environnements d'exécution de langage, des systèmes d'exploitation ou des bibliothèques et frameworks tiers. Ces dépendances sont généralement liées à la communauté linguistique ou au fournisseur propriétaire

du composant en question. Chaque communauté ou fournisseur publie son propre end-of-support calendrier pour son produit.

Les termes suivants sont utilisés pour classer les dépendances tierces sous-jacentes :

- Système d'exploitation (OS) : les exemples incluent Amazon Linux AMI, Amazon Linux 2, Windows 2008, Windows 2012, Windows 2016, etc.
- Language Runtime : les exemples incluent Java 7, Java 8, Java 11, .NET Core, .NET Standard, .NET PCL, etc.
- Bibliothèque/framework tiers : les exemples incluent OpenSSL, .NET Framework 4.5, Java EE, etc.

Notre politique est de continuer à prendre en charge les dépendances du SDK pendant au moins 6 mois après que la communauté ou le fournisseur a mis fin au support de ces dépendances. Cette politique peut toutefois varier en fonction de la dépendance spécifique.

> **Note**
>
> AWS se réserve le droit de mettre fin à la prise en charge d'une dépendance sous-jacente sans augmenter la version majeure du SDK

Méthodes de communication

Les annonces de maintenance sont communiquées de plusieurs manières :

- Un e-mail d'annonce est envoyé aux comptes concernés, annonçant notre intention de mettre fin au support pour la version spécifique du SDK. L'e-mail indiquera le chemin à suivre end-of-support, précisera le calendrier de la campagne et fournira des conseils de mise à niveau.
- AWS La documentation du SDK, telle que la documentation de référence des API, les guides de l'utilisateur, les pages marketing des produits du SDK et les fichiers GitHub readme, est mise à jour pour indiquer le calendrier de la campagne et fournir des conseils sur la mise à niveau des applications concernées.
- Un article de AWS blog est publié qui décrit le chemin à end-of-support suivre et réitère le calendrier de la campagne.
- Des avertissements d'obsolescence sont ajoutés au SDKs, décrivant le chemin d'accès à la documentation du SDK end-of-support et contenant des liens vers celle-ci.

Pour consulter la liste des versions majeures disponibles de AWS SDKs et Tools et leur état d'avancement dans leur cycle de maintenance, voirCycle de vie des versions.

AWS SDKs et cycle de vie des versions des outils

Le tableau ci-dessous présente la liste des versions majeures du kit de développement AWS logiciel (SDK) disponibles et indique où elles en sont dans le cycle de maintenance, ainsi que les délais associés. Pour des informations détaillées sur le cycle de vie des versions principales de AWS SDKs and Tools et de leurs dépendances sous-jacentes, consultezPolitique de maintenance.

Kit SDK	Version majeure	Phase en cours	Date de disponibilité générale	Remarques
AWS CLI	1.x	Annonce de maintenance	02/09/2013	Voir l'annonce pour les détails et les dates
AWS CLI	2.x	Disponibilité générale	10/02/2020	
SDK pour C++	1.x	Disponibilité générale	02/09/2015	
SDK pour Go V2	V2.1.x	Disponibilité générale	19/01/2021	
SDK pour Go	1.x	Fin du support	19/11/2015	
SDK pour Java	1.x	Fin du support	25/03/2010	
SDK pour Java	2.x	Disponibilité générale	11-20-2018	
SDK pour JavaScript	1.x	Fin du support	06/05/2013	
SDK pour JavaScript	2.x	Fin du support	19/06/2014	

Kit SDK	Version majeure	Phase en cours	Date de disponibilité générale	Remarques
SDK pour JavaScript	3.x	Disponibilité générale	15 décembre 2020	
SDK pour Kotlin	1.x	Disponibilité générale	27/11/2023	
SDK pour .NET	1.x	Fin du support	11/2009	
SDK pour .NET	2.x	Fin du support	08/11/2013	
SDK pour .NET	3.x	Disponibilité générale	28/07/2015	
SDK pour .NET	4. x	Disponibilité générale	28/04/2025	
SDK pour PHP	2.x	Fin du support	02/11/2012	
SDK pour PHP	3.x	Disponibilité générale	27/05/2015	
SDK pour Python (Boto2)	1.x	Fin du support	13/07/2011	
SDK pour Python (Boto3)	1.x	Disponibilité générale	22/06/2015	
SDK pour Python (Botocore)	1.x	Disponibilité générale	22/06/2015	
SDK pour Ruby	1.x	Fin du support	14/07/2011	
SDK pour Ruby	2.x	Fin du support	15/02/2015	

Kit SDK	Version majeure	Phase en cours	Date de disponibilité générale	Remarques
SDK pour Ruby	3.x	Disponibilité générale	29/08/2017	
SDK pour Rust	1.x	Disponibilité générale	27/11/2023	
SDK pour Swift	1.x	Disponibilité générale	17/09/2024	
Outils pour PowerShell	2.x	Fin du support	08/11/2013	
Outils pour PowerShell	3.x	Fin du support	29/07/2015	
Outils pour PowerShell	4. x	Disponibilité générale	21/11/2019	
Outils pour PowerShell	5.x	Disponibilité générale	23/06/2025	

Vous recherchez un SDK ou un outil non mentionné ? Le chiffrement SDKs, les appareils SDKs IoT et les appareils mobiles SDKs, par exemple, ne sont pas inclus dans ce guide. Pour trouver de la documentation sur ces autres outils, consultez la section Outils sur lesquels vous pouvez vous appuyer AWS.

Historique du document AWS SDKs et guide de référence sur les outils

Le tableau suivant décrit les ajouts et mises à jour importants apportés au guide de référence AWS SDKs and Tools. Pour recevoir les notifications sur les mises à jour de cette documentation, vous pouvez vous abonner au Flux RSS.

Modification	Description	Date
Ajout d'un nouveau paramètre S3 Express One Zone	Ajout d'un nouveau paramètre S3 Express One Zone pour désactiver l'authentification de session.	13 octobre 2025
Ajout d'un nouvel arbre de décision en matière d'authentification	Ajout d'un nouvel arbre de décision pour faciliter les décisions d'authentification entre les options.	23 septembre 2025
Ajout d'une nouvelle fonctionnalité de schéma d'authentification	Ajout d'une nouvelle fonctionnalité de schéma d'authentification. Mises à jour des points de terminaison AWS STS régionaux.	18 août 2025
Ajout d'une nouvelle version de Tools for PowerShell	Ajout de la dernière version des outils d' PowerShell l assistance à tous les paramètres de référence . Compatibilité avec AWS SDKs les tables. Ajout de la fonctionnalité d'injection du préfixe hôte.	23 juin 2025

Mise à jour du titre de page	Plus de titres, de titres de tableaux, de résumés et de mises à jour SEO.	5 mars 2025
Mise à jour du titre de page	Mettre à jour le contenu pour utiliser des titres plus descripti fs.	24 février 2025
Ajout du SDK Swift à la référence des paramètres	Ajout de la prise en charge du SDK Swift à tous les paramètres de référence. Compatibilité avec AWS SDKs les tables.	17 septembre 2024
Propriétés du système SDK for Java 1.x	Ajoutez des détails sur les paramètres de configura tion du système JVM pris en charge par la version AWS SDK pour Java 1.x.	30 mai 2024
Mise à jour des paramètres	Ajoutez les paramètres de configuration du système JVM.	27 mars 2024
Mise à jour du tableau de compatibilité	Mises à jour de compatibilité pour le support du SDK, mises à jour des procédures IAM Identity Center.	20 février 2024
Mise à jour des informations d'identification du conteneur. Mise à jour IMDS.	Ajout de la prise en charge d'Amazon EKS. Ajout d'un paramètre pour désactiver le IMDSv1 repli.	29 décembre 2023
Compression des demandes	Ajout de paramètres pour la fonction de compression des demandes.	27 décembre 2023

Tableaux de compatibilité	Les tableaux de compatibilité pour les fonctionnalités du SDK et des outils ont été mis à jour pour inclure le SDK pour Kotlin, le SDK pour Rust et. Outils AWS pour PowerShell	10 décembre 2023
Mises à jour de	Mises à jour des méthodes d'authentification SDKs et des outils pris en charge.	1er juillet 2023
Mises à jour des bonnes pratiques IAM	Mise à jour du guide s'aligner sur les bonnes pratiques IAM. Pour plus d'informations, consultez Bonnes pratiques de sécurité dans IAM.	27 février 2023
Mises à jour du SSO	Mises à jour des informations d'identification SSO pour la nouvelle configuration du jeton SSO.	19 novembre 2022
Mise à jour des paramètres	Mises à jour du tableau de support pour la configuration générale et pour les points d'accès multirégionaux Amazon S3.	17 novembre 2022
Mise à jour des paramètres	Mises à jour visant à clarifier les informations d'identification du client IMDS et de l'IMDS. Mises à jour des variables d'environnement.	4 novembre 2022
Mettre à jour la page d'accueil	Annonce d'Amazon CodeWhisperer.	22 septembre 2022

Changement de nom de service pour l'authentification unique	Mises à jour pour refléter le fait que le AWS SSO est désormais appelé. AWS IAM Identity Center	26 juillet 2022
Mise à jour des paramètres	Mises à jour mineures des détails du fichier de configuration et des paramètres pris en charge.	15 juin 2022
Mettre à jour	Mise à jour massive de presque toutes les parties de ce guide.	1er février 2022
Première version	La première version de ce guide est mise à la disposition du public.	13 mars 2020

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.