AWS SDKs and Tools

Reference Guide

AWS SDKs and Tools: Reference Guide

Table of Contents

AWS SDKs and Tools Reference Guide

Applicable to all SDKs and tools

AWS SDKs and Tools maintenance policy (p. 66) covers the maintenance policy and versioning for AWS Software Development Kits (SDKs) and Tools, including Mobile and IoT SDKs, and their underlying dependencies.

Applicable to some SDKs and tools

Many SDKs and tools share some common functionality – either through shared design specifications or through a shared library (in the case of the AWS Common Runtime (CRT)).

This AWS SDKs and Tools Reference Guide is intended to be a base of information that is applicable to multiple SDKs and/or tools. The specific SDK or tool guide for the SDK or tool you are using should be used in addition to any information presented here. The following SDKs and tools have, to at least some extent, additional information represented in this guide:

- AWS Cloud Development Kit (CDK) Developer Guide
- AWS Command Line Interface User Guide
- AWS Serverless Application Model Developer Guide
- AWS SDK for C++ Developer Guide
- AWS SDK for Go Developer Guide
- AWS SDK for Java Developer Guide
- AWS SDK for JavaScript Developer Guide
- AWS SDK for .NET Developer Guide
- AWS SDK for PHP Developer Guide
- AWS SDK for Python (Boto3) Getting Started
- AWS SDK for Ruby Developer Guide
- AWS Toolkit for Eclipse User Guide
- AWS Toolkit for JetBrains User Guide
- AWS Toolkit for Visual Studio User Guide
- AWS Toolkit for Visual Studio Code User Guide
- AWS Tools for Windows PowerShell User Guide

This guide includes information regarding:

- The shared config and credentials files (p. 3) – Explanation of the shared `config` and `credentials` files for the authentication and configuration of an AWS SDK or tool.
- Configuration and authentication settings reference (p. 11) – Reference for all settings available for authentication and configuration.

  Go right to the list of global settings. (p. 11)

Go right to the list of Amazon S3-specific settings. (p. 50)

- AWS Common Runtime (CRT) libraries (p. 65) – Overview of the AWS Common Runtime (CRT) libraries that are available to (almost) all SDKs.

The shared config and credentials files

AWS SDKs and other AWS developer tools, such as the AWS Command Line Interface (AWS CLI) enable you to interact with AWS service APIs. Before attempting that, however, you must configure the SDK or tool with the information it needs to perform the requested operation.

This information includes the following items:

- **Credentials information** that identifies who is calling the API. The credentials are used to encrypt the request to the AWS servers. Using this information, AWS confirms your identity and can retrieve permission policies associated with it. Then it can determine what actions you're allowed to perform.
- **Other configuration details** that enable you to tell the AWS CLI or SDK how to process the request, where to send the request (to which AWS service endpoint), and how to interpret or display the response.

About credential providers

Each SDK or tool can provide multiple methods, called *credential providers*, that you can use to supply the required credential and configuration information. Some credential providers are unique to the SDK or tool, and you must refer to the documentation for that tool or SDK for the details on how to use that method.

However, most of the AWS SDKs and tools share a few common credential providers for finding the required information. These methods are the subject of this guide.

- **Shared AWS config and credentials files** (p. 3) – The shared `config` and `credentials` files are the most common way that you can specify authentication and configuration to an AWS SDK or tool. These files enable you to store settings that your tools and applications can use. The primary file is `config`, and you can put all settings into it. However, by default and as a security best practice, sensitive values such as secret keys are stored in a separate `credentials` file. This enables you to separately protect those settings with different permissions. Together, these files enable you to configure multiple groups of settings. Each group of settings is called a *profile*. When you use an AWS tool to invoke a command or use an SDK to invoke an AWS API, you can specify which profile, and thus which configuration settings, to use for that action. One of the profiles is designated as the `default` profile and is used automatically when you don't explicitly specify a profile to use. The settings that you can store in these files are documented in this reference guide.
- **Environment variables** (p. 63) – Some of the settings can alternatively be stored in the environment variables of your operating system. While you can have only one set of environment variables in effect at a time, they are easily modified dynamically as your program runs and your requirements change.
- **Per-operation parameters** – A few settings can be set on a per-operation basis, and thus changed as needed for each operation you invoke. For the AWS CLI or AWS Tools for PowerShell, these take the form of parameters that you enter on the command line. For an SDK, they can take the form of a parameter that you set when you instantiate an AWS client session or service object, or sometimes when you call an individual API.

Precedence and credential provider order

When an AWS SDK or tool looks for credentials or a configuration setting, it invokes each credential provider in a certain order, and stops when it finds a value that it can use. Most AWS SDKs and tools check the credential providers in the following order:

1. Per-operation parameter
2. Environment variable
3. Shared `credentials` file
4. Shared `config` file

Note
If a setting exists in both the `config` file and the `credentials` file for the same profile, the value in the `credentials` file is used instead of the value in the `config` file.

Note
Some SDKs and tools might check in a different order. Also, some SDKs and tools support other methods of storing and retrieving parameters. For example, the AWS SDK for .NET supports an additional credential provider called the SDK Store. For more information about the credential provider order or credential providers that are unique to a SDK or tool, see the documentation for that SDK or tool (p. 8).

The order determines which methods take precedence and override others. For example, if you set up a `default` profile in the shared `config` file, it's only found and used after the SDK or tool checks the other credential providers first. This means that if you put a setting in the `credentials` file, it is used instead of one found in the `config` file. If you configure an environment variable with a setting and value, it would override that setting in both the `credentials` and `config` files. And finally, a setting on the individual operation (AWS CLI command-line parameter or API parameter), would override all other values for that one command.

Additional topics in this section

Location of the shared `config` and `credentials` files

The shared AWS `config` and `credentials` files are plaintext files that reside by default in a folder named `.aws` that is placed in the "home" folder on your computer.

On Linux and macOS, this is typically shown as `~/.aws`. On Windows it is `%USERPROFILE%\.aws`.

Operating system	Default location of files

Linux and macOS	`~/.aws/config`
	
	`~/.aws/credentials`

Operating system	Default location of files
Windows	`%USERPROFILE%\.aws\config` `%USERPROFILE%\.aws\credentials`

A `~/` or `~` followed by the file system's default path separator at the start of the path is resolved by checking, in order,

1. (All Platforms) The `HOME` environment variable.
2. (Windows Platforms) The `USERPROFILE` environment variable.
3. (Windows Platforms) The `HOMEDRIVE` environment variable prepended to the `HOMEPATH` environment variable (ie. `$HOMEDRIVE$HOMEPATH`).
4. (Optional per SDK or tool) An SDK or tool-specific home path resolution function or variable.

When possible, if a user's home directory is specified (e.g. `~username/`) at the start of the path, it is resolved to the requested username's home directory (e.g. `/home/username/.aws/config`).

Changing the default location of these files:

The following environment variables can be set to change the location or name of these files from the default to a custom value

- `config` file environment variable: **`AWS_CONFIG_FILE`**
- `credentials` file environment variable: **`AWS_SHARED_CREDENTIALS_FILE`**

Linux/macOS

You can specify an alternate location by running the following export commands on Linux or 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

You can specify an alternate location by running the following setx commands on 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
```

Format of the shared config and credentials files

The shared AWS `config` and `credentials` files contain a set of profiles. A profile is a set of configuration values that can be referenced from the SDK/tool using its profile name. Configuration values are attached to a profile in order to configure some aspect of the SDK/tool when that profile is used.

As a general rule, any value that you can place in the shared `credentials` file can alternatively be placed in the shared `config` file. The other way isn't true: only a few settings can be placed in the `credentials` file. However, as a security best practice, we recommend that you keep any sensitive values, such as access key IDs and secret keys, in the separate `credentials` file. This enables you to provide separate permissions for each file, if necessary.

The easiest way to obtain these files is to download them from the AWS Management Console by following the instructions for Managing access keys in the *IAM User Guide*

Both the shared `config` and `credentials` files are plaintext files that contain only ASCII characters (UTF-8 encoded). They take the form of what are generally referred to as INI files.

Format of the config file

The `config` file must be a plaintext file that uses the following format:

- Each section begins with the profile name in square brackets `[ ]`.
- All entries in a section take the general form of `setting-name=value`.

The following example shows a simple `config` file having a `[default]` profile. It sets the `region` (p. 37) and `output` (p. 35) global settings.

```
[default]
region = us-east-2
output = json
```

The `[default]` profile contains the values that are used by an SDK or tool operation if a specific named profile is not specified.

You can also create separate profiles that you can explicitly reference by name. To create a named profile in the `config` file, create a section with a new header, similar to the following example. You must use the word `profile` and follow it with a unique name. You can use letters, numbers, hyphens (–), and underscores (_), but no spaces.

```
[profile developers]
...settings...
```

Each named profile can have a different group of settings. You then reference the profile you want to use by name from your SDK code or your AWS CLI commands.

`[default]` is simply an unnamed profile. This profile is named "default" because it is the default profile used by the SDK if the user does not specify a profile. It does **not** provide inherited default values to other profiles, e.g. if you've set something in the `[default]` profile, and do not set it in a named profile, when you use the named profile the value will not be set.

Some settings have their own nested group of subsettings, such as the `s3` setting and subsettings in the following example. Associate the subsettings with the group by indenting them by one or more spaces.

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

Format of the credentials file

The following example shows a simple `credentials` file having a `[default]` profile. It sets the `aws_access_key_id` (p. 18) and `aws_secret_access_key` (p. 19) global settings.

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

The rules for the `credentials` file are generally identical to those for the `config` file, ***except***:

- The section names don't begin with the word `profile`. Use only the profile name itself between square brackets.

```
[developers]
...settings...
```

- You can store only a subset of settings and values in the `credentials` file. Generally, it's only those with values that would be considered "secrets" or sensitive, such as access key IDs and secret keys. The page for each setting in this guide states whether it can be stored in the `credentials` file or if it can be stored only in the `config` file.

Example files

In summary, each profile can have some settings in each file. The majority of settings go in the `config` file, while the sensitive information settings go in the `credentials` file.

The following example shows three profiles stored in these two files:

- `default` profile – Provides access by using the long-term credentials of an AWS Identity and Access Management (IAM) user. Tools or code that use this profile send requests to the US West (Oregon) Region (`us-west-2`). AWS CLI commands invoked using this profile output the results as JSON.
- `dev-user` profile – Uses the long-term credentials of a different IAM user. Tools or code that use this profile send requests to the US West (N. California) Region (`us-west-1`). AWS CLI commands invoked using this profile output the results as simple text.
- `developers` profile – Uses short-term credentials from assuming the specified role. It uses the long-term credentials in the `dev-user` source profile only to assume the role and retrieve the short-term credentials for the role. Tools or code that use this profile send requests to the US West (Oregon) Region (`us-west-2`). AWS CLI commands invoked using this profile output the results as JSON. This profile doesn't store any of its values in the `credentials` file.

Contents of the `config` file

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

[profile dev-user]
region = us-west-1
output = text

[profile developers]
role_arn = arn:aws:iam::123456789012:role/developers
source_profile = dev-user
region = us-west-2
output = json
```

Contents of the `credentials` file

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

[dev-user]
aws_access_key_id = AKIAI44QH8DHBEXAMPLE
```

```
aws_secret_access_key = je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY
```

AWS SDKs and tools that use the shared config and credentials files

The following is a list of the AWS SDKs and other development tools that can use the shared `config` and `credentials` files to retrieve authentication and other configuration settings.

Each entry includes a link to that SDK's or tool's documentation where the use of these files is discussed in detail.

SDK or tool	Shared config and credentials topic
AWS Cloud Development Kit (CDK)	Getting started with AWS Cloud Development Kit (CDK)
AWS Command Line Interface (AWS CLI)	Configuring the AWS CLI
AWS Serverless Application Model	Setting up AWS credentials
AWS SDK for C++	Providing AWS credentials
AWS SDK for Go	Specifying credentials
AWS SDK for Java	Using credentials
AWS SDK for JavaScript	Getting your credentials
AWS SDK for .NET	Configure AWS credentials
AWS SDK for PHP	Using the AWS Credentials File and Credential Profiles
AWS SDK for Python (Boto3)	Configuration
AWS SDK for Ruby	Setting AWS Credentials
AWS Toolkit for Eclipse	Set up AWS credentials
AWS Toolkit for JetBrains	Setting AWS Credentials for the AWS Toolkit for JetBrains
AWS Toolkit for Visual Studio	Providing AWS Credentials
AWS Toolkit for Visual Studio Code	Setting Up Your AWS Credentials
AWS Tools for PowerShell	Getting Started with AWS Tools for PowerShell

Example uses for shared credentials

The topics in this section describe common authentication scenarios and how you might configure the shared AWS `config` and `credentials` files to support them.

Topics

Credentials for an IAM role assumed as an IAM user

This topic describes how to configure the shared AWS `config` and `credentials` files to support logging in to an AWS SDK or developer tool using an IAM role. The SDK or tool assumes the role using the credentials of a separate IAM user.

Scenario description

This scenario requires that you have two entities created in IAM:

- An IAM user, that in this example we call `UserAlpha`. This user has IAM policy permissions that enable it to perform only the `sts:AssumeRole` operation.

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

- An IAM role, that in this example we call `RoleBeta`. This role has a trust policy that enables all users in the account to assume the role.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::231179739868:root"
      },
      "Action": "sts:AssumeRole",
      "Condition": {}
    }
  ]
}
```

  It also has IAM permission policies that enable it to perform any task in Amazon S3.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": "*"
        }
    ]
}
```

How to configure the profile

The following example `config` and `credentials` files show how you can configure SDK or AWS developer tool access using `RoleBeta`.

Contents of config

```
[profile UserAlpha]
region = us-west-2
output = json

[profile RoleBeta]
source_profile = UserAlpha
role_arn = arn:aws:iam::123456789012:role/RoleBeta
```

Contents of credentials

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

How to use the profile

As an example, you can run the following AWS CLI command to list the Amazon S3 buckets available in the account. The AWS CLI sees that `RoleBeta` profile references the `source_profile` `UserAlpha`. It looks up the `UserAlpha` access key and secret key, and uses them to call the `sts:AssumeRole` operation on the Amazon Resource Name (ARN) of `RoleBeta`. That operation returns short-term credentials for `RoleBeta` that the AWS CLI uses to call the `s3:ListBuckets` operation.

```
$ aws s3api list-buckets --profile RoleBeta
{
    "Buckets": [
        {
            "Name": "my-first-bucket",
            "CreationDate": "2018-08-31T07:46:02+00:00"
        },
        {
            "Name": "my-second-bucket",
            "CreationDate": "2019-09-17T19:17:31+00:00"
        },
        {
            "Name": "my-third-bucket",
            "CreationDate": "2018-06-12T23:18:08+00:00"
        }
    ],
    "Owner": { ...truncated... }
}
```

Configuration and authentication settings reference

- Global settings for config and credentials files (p. 11) – List of all settings available for shared AWS `config` and `credentials` files for the configuration and authentication of an AWS SDK or tool.
- SDK and tool settings for Amazon S3 APIs (p. 50) – Settings that specifically affect Amazon S3
- Supported environment variables (p. 63) – Some of the settings can alternatively be stored in the environment variables of your operating system. While you can have only one set of environment variables in effect at a time, they are easily modified dynamically as your program runs and your requirements change.

Global settings for config and credentials files

The topics in this section describe the settings that can be stored in the shared `config` and `credentials` files. For information about these files, see The shared config and credentials files (p. 3).

The following settings are global and affect all services. Choose the name of the setting to see its details page.

Important
Not all SDKs and Toolkits support all of the settings listed below. Choose a setting to see the details page that includes which SDKs and Toolkits support that setting.

Setting	Summary	Config file	Credentials file	Env variable	AWS CLI parameter
`api_versions` (p. 17)	Specifies the version of the AWS service's API to use.	Yes	-	-	-
`aws_access_key_id` (p. 18)	Access key ID to use to authenticate the user.	Yes	Yes	Yes	-
`aws_secret_access_key` (p. 19)	Secret key to use to authenticate the user.	Yes	Yes	Yes	-
`aws_session_token` (p. 20)	Session token for temporary credentials to use to authenticate the user.	Yes	Yes	Yes	-
`ca_bundle` (p. 21)	Specifies a Certificate Authority (CA)	Yes	-	Yes	Yes

Setting	Summary	Config file	Credentials file	Env variable	AWS CLI parameter
	certificate bundle (a file with the `pem` extension) that is used to verify SSL certificates.				
`cli_binary_format` (p. 22)	*(Available in the AWS CLI V2 only.)* Specifies how the AWS CLI handles binary input parameters.	Yes	-	-	Yes
`cli_follow_urlparam` (p. 24)	*(Available in the AWS CLI V1 only.)* Specifies whether the AWS CLI attempts to follow URL links in command line parameters that begin with `http://` or `https://`.	Yes	-	-	-
`cli_timestamp_format` (p. 25)	*(Available in the AWS CLI only.)* Specifies how the AWS CLI formats timestamp output values.	Yes	-	-	-
`credential_process` (p. 26)	Specifies an external command that the SDK or tool runs on your behalf to generate or retrieve authentication credentials to use.	Yes	-	-	-

Setting	Summary	Config file	Credentials file	Env variable	AWS CLI parameter
`credential_source` (p. 28)	Used with Amazon EC2 instances or containers. Specifies where the SDK or tool can find credentials that can be used to assume the role identified by the `role_arn` setting.	Yes	-	-	-
`duration_seconds` (p. 29)	Specifies the maximum duration of an assumed role session.	Yes	-	-	-
`external_id` (p. 30)	Specifies a unique identifier that is used by third parties to assume a role in their customers' accounts.	Yes	-	-	-
`max_attempts` (p. 31)	Specifies the maximum number attempts to make on a request.	Yes	-	Yes	-
`metadata_service_num_attempts` (p. 32)	Specifies the number of attempts to make before giving up when attempting to retrieve data from the instance metadata service.	Yes	-	-	-

Setting	Summary	Config file	Credentials file	Env variable	AWS CLI parameter
`metadata_service_timeout` (p. 33)	Specifies the number of seconds before timing out when attempting to retrieve data from the instance metadata service.	Yes	-	-	-
`mfa_serial` (p. 34)	Specifies the identification number of a multi-factor authentication (MFA) device to use when assuming a role.	Yes	-	-	-
`output` (p. 35)	*(Available in the AWS CLI only.)* Specifies the output format for AWS CLI commands.	Yes	-	Yes	Yes
`parameter_validation` (p. 36)	*(Available in the AWS CLI only.)* Specifies whether the AWS CLI client attempts to validate parameters before sending them to the AWS service endpoint.	Yes	-	-	-
`region` (p. 37)	Specifies the AWS Region to send requests to for commands requested using this profile.	Yes	-	Yes	Yes

Setting	Summary	Config file	Credentials file	Env variable	AWS CLI parameter
`retry_mode` (p. 38)	Specifies how the SDK attempts retries.	Yes	-	Yes	-
`role_arn` (p. 39)	Specifies the Amazon Resource Name (ARN) of an IAM role that you want to use for operations requested using this profile.	Yes	-	-	-
`role_session_name` (p. 41)	Specifies the name to attach to the role session. This name appears in AWS CloudTrail logs for entries associated with this session.	Yes	-	-	-
`source_profile` (p. 42)	Specifies another profile whose credentials are used to assume the role specified by this profile.	Yes	-	-	-
`sso_account_id` (p. 43)	*(Available in the AWS CLI V2 only.)* Specifies the AWS account ID that contains the IAM role with the permission that you want to grant to the associated AWS SSO user.	Yes	-	-	-

Setting	Summary	Config file	Credentials file	Env variable	AWS CLI parameter
`sso_region` (p. 45)	*(Available in the AWS CLI V2 only.)* Specifies the AWS Region that contains the AWS SSO portal host.	Yes	-	-	-
`sso_role_name` (p. 45)	*(Available in the AWS CLI V2 only.)* Specifies the friendly name of the IAM role that defines the user's permissions when using this profile.	Yes	-	-	-
`sso_start_url` (p. 46)	*(Available in the AWS CLI V2 only.)* Specifies the URL that points to the organization's AWS SSO user portal. The AWS CLI uses this URL to establish a session with the AWS SSO service to authenticate its users.	Yes	-	-	-
`sts_regional_endpoints` (p. 47)	Specifies how the AWS CLI determines the AWS service endpoint that the AWS CLI client uses to talk to the AWS Security Token Service (AWS STS).	Yes	-	-	-

Setting	Summary	Config file	Credentials file	Env variable	AWS CLI parameter
tcp_keepalive (p. 49)	Specifies whether the AWS CLI client uses TCP keep-alive packets.	Yes	-	-	-
web_identity_token_file (p. 49)	Specifies the path to a file that contains an access token from an OAuth 2.0 or OpenID Connect identity provider. This enables authentication by using Web Identity federation.	Yes	-	-	-

api_versions

Some AWS services maintain multiple API versions to support backward compatibility. By default, SDK and AWS CLI operations use the latest available API version. To require a specific API version to use for your requests, include the api_versions setting in your profile.

Details

This is a "nested" setting that's followed by one or more indented lines that each identify one AWS service and the API version to use. See the documentation for each service to understand which API versions are available.

Ways to set this value

The example shows how to specify an API version for two AWS services. These API versions are used only for commands that run under the profile that contains these settings. Commands for any other service use the default (latest) version of that service's API.

Location	Supported	Example

config file	Yes	`api_versions =`
		`    ec2 = 2015-03-01`
		`    cloudfront = 2015-09-017`

credentials file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	-	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	-	
AWS Tools for PowerShell	-	

`aws_access_key_id`

Specifies the AWS access key used as part of the credentials to authenticate the user.

Details

Although this setting can be stored in the config file, for security reasons, we recommend that you store this in the credentials file.

Ways to set this value

Location	Supported	Example
`config` file	Yes *Not recommended*	`aws_access_key_id = AKIAIOSFODNN7EXAMPLE`
`credentials` file	Yes ***Recommended***	`aws_access_key_id = AKIAIOSFODNN7EXAMPLE`

Environment variable	Yes	Linux/macOS
		`export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE`
		Windows
		`setx AWS_ACCESS_KEY_ID AKIAIOSFODNN7EXAMPLE`

AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	Yes	
AWS SDK for PHP	Yes	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	Yes	This SDK/Tool doesn't support the environment variable equivalent: `AWS_ACCESS_KEY_ID`
AWS Toolkit for Visual Studio	Yes	This SDK/Tool doesn't support the environment variable equivalent: `AWS_ACCESS_KEY_ID`
AWS Toolkit for Visual Studio Code	Yes	This SDK/Tool doesn't support the environment variable equivalent: `AWS_ACCESS_KEY_ID`
AWS Tools for PowerShell	Yes	

aws_secret_access_key

Specifies the AWS secret key used as part of the credentials to authenticate the user.

Details

Although this setting can be stored in the config file, for security reasons, we recommend that you store this in the credentials file.

Ways to set this value

Location	Supported	Example
`config` file	Yes *Not recommended*	`aws_secret_access_key = wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`
`credentials` file	Yes ***Recommended***	`aws_secret_access_key = wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`
Environment variable	Yes	Linux/macOS `export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY` Windows

Location	Supported	Example
		`setx AWS_SECRET_ACCESS_KEY wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	Yes	
AWS SDK for PHP	Yes	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	Yes	This SDK/Tool doesn't support the environment variable equivalent: `AWS_SECRET_ACCESS_KEY`
AWS Toolkit for Visual Studio	Yes	This SDK/Tool doesn't support the environment variable equivalent: `AWS_SECRET_ACCESS_KEY`
AWS Toolkit for Visual Studio Code	Yes	This SDK/Tool doesn't support the environment variable equivalent: `AWS_SECRET_ACCESS_KEY`
AWS Tools for PowerShell	Yes	

`aws_session_token`

Specifies an AWS session token used as part of the credentials to authenticate the user. A session token is required only if you manually specify temporary security credentials.

Details

Although this setting can be stored in the config file, for security reasons, we recommend that you store this in the credentials file.

You receive this value as part of the temporary credentials returned by successful requests to assume a role.

Ways to set this value

Location	Supported	Example
`config` file	Yes - *Not recommended*	`aws_session_token = AQoEXAMPLEH4aoAH0gNCAPy...truncated...zr`

Location	Supported	Example
`credentials` file	Yes - ***Recommended***	`aws_session_token = AQoEXAMPLEH4aoAH0gNCAPy...truncated...zr`

Environment variable	Yes	Linux/macOS
		`export AWS_SESSION_TOKEN=AQoEXAMPLEH4aoAH0gNCAP`
		Windows
		`setx AWS_SESSION_TOKEN AQoEXAMPLEH4aoAH0gNCAPy...truncated...zr`

AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	Yes	
AWS SDK for PHP	Yes	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	Yes	This SDK/Tool doesn't support the environment variable equivalent: `AWS_SESSION_TOKEN`
AWS Toolkit for Visual Studio	Yes	This SDK/Tool doesn't support the environment variable equivalent: `AWS_SESSION_TOKEN`
AWS Toolkit for Visual Studio Code	Yes	This SDK/Tool doesn't support the environment variable equivalent: `AWS_SESSION_TOKEN`
AWS Tools for PowerShell	Yes	

`ca_bundle`

Specifies the path to a custom certificate bundle (a file with a `.pem` extension) to use when establishing SSL/TLS connections.

Details

The AWS SDKs and tools that support this setting include a CA bundle that they will use by default. But you can set this value to use a different CA bundle.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`ca_bundle = dev/apps/ca-certs/cabundle-2019mar05.pem`
`credentials` file	-	

Environment variable	Yes	Linux/macOS
		`export AWS_CA_BUNDLE=/dev/apps/ca-certs/cabundle-2019mar05.pem`
		Windows
		`setx AWS_CA_BUNDLE C:\dev\apps\ca-certs\cabundle-2019mar05.pem`

AWS CLI parameter	Yes	`--ca-bundle /dev/apps/ca-certs/cabundle-2019mar05.pem`

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	-	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	-	
AWS Tools for PowerShell	-	

cli_binary_format

This setting applies only to AWS CLI commands.

Specifies how the AWS CLI version 2 interprets binary input parameters.

Details

Default value:

- AWS CLI version 1 – `raw-in-base64-out`. *In AWS CLI version 1 you can't change this setting.*
- AWS CLI version 2 – `base64`

Valid values:

- **base64** – An input parameter that is typed as a binary large object (BLOB) that accepts a base64-encoded string. To pass true binary content, put the content in a file and provide the file's path and name with the `fileb://` prefix as the parameter's value. To pass base64-encoded text contained in a file, provide the file's path and name with the `file://` prefix as the parameter's value.
- **raw-in-base64-out** – Provides backward compatibility with the AWS CLI version 1 behavior where binary values must be passed literally.

This setting also affects how the AWS CLI version 2 handles input files that contain information for a binary parameter.

- If you specify a binary value by referencing a file using the `fileb://` prefix notation, the AWS CLI (both versions 1 and 2) *always* handles the file as raw binary content and doesn't attempt to convert the value.
- If you specify a binary value by referencing a file using the `file://` prefix notation, the AWS CLI version 2 handles the file according to the current `cli_binary_format` setting. If that setting's value is `base64` (the default when not explicitly set), the CLI handles the file as base64-encoded text. If that setting's value is `raw-in-base64-out`, the AWS CLI version 2 handles the file as raw binary content. The AWS CLI version 1 *always* handles a file referenced by `file://` as raw binary content.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`cli_binary_format = raw-in-base64-out`
`credentials` file	-	
Environment variable	-	
AWS CLI parameter	Yes	`--cli-binary-format raw-in-base64-out`

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	-	

SDK or tool	Su	Notes or more information
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	-	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	-	
AWS Tools for PowerShell	-	

`cli_follow_urlparam`

This setting applies **only** to AWS CLI commands.

Specifies how the AWS CLI interprets input parameters that begin with `http:` or `https://`.

Details

Default value:

- AWS CLI version 1: `true`.
- AWS CLI version 2: `false`. *In the AWS CLI version 2 you can't change this setting.*

Valid values:

- **true** – If specified, any string parameters that begin with `http://` or `https://` are retrieved from the internet, and any downloaded content is used as the parameter value for the command.
- **false** – If specified, the CLI doesn't treat parameter string values that begin with `http://` or `https://` differently from other strings.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`cli_follow_urlparam = false`
`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	-	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	-	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	-	
AWS Tools for PowerShell	-	

`cli_timestamp_format`

This setting applies **only** to AWS CLI commands.

Specifies the format of timestamp values included in the output of AWS CLI commands.

Details

Default value:

- AWS CLI version 1 – `wire`
- AWS CLI version 2 – `iso8601`

Valid values:

- **iso8601** – If specified, the AWS CLI reformats all timestamps according to ISO 8601.
- **wire** – If specified, the AWS CLI displays all timestamp values exactly as received in the HTTP query response from the AWS service.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`cli_timestamp_format = iso8601`
`credentials` file	-	
Environment variable	-	

Location	Supported	Example
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	-	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	-	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	-	
AWS Tools for PowerShell	-	

`credential_process`

Specifies an external command that the SDK or tool runs on your behalf to generate or retrieve authentication credentials to use.

Details

When the SDK or developer tool you're using requires credentials, and you specify a profile that contains this setting, the SDK or tool runs the specified program on your behalf in the background. It must return information in the specified format. That information contains the credentials that the SDK or tool can use to authenticate you.

Specifying the path to the credentials program

The setting's value is a string that contains a path to a program that the SDK or development tool runs on your behalf:

- The path and file name can consist of only these characters: A-Z, a-z, 0-9, hyphen (-), underscore (_), period (.), and space.
- If the path or file name contains a space, surround the complete path and file name with double-quotation marks (" ").
- If a parameter name or a parameter value contains a space, surround that element with double-quotation marks (" "). Surround only the name or value, not the pair.
- Don't include any environment variables in the strings. For example, don't include `$HOME` or `%USERPROFILE%`.
- Don't specify the home folder as ~. You must specify the full path.

For examples, see Ways to set this value (p. 27), later in this topic.

Expected output from the credentials program

The AWS CLI runs the command as specified in the profile and then reads data from the standard output stream. The command you specify, whether a script or binary program, must generate JSON output on `STDOUT` that matches the following syntax.

```
{
    "Version": 1,
    "AccessKeyId": "an AWS access key",
    "SecretAccessKey": "your AWS secret access key",
    "SessionToken": "the AWS session token for temporary credentials",
    "Expiration": "ISO8601 timestamp for when the credentials expire"
}
```

Note
As of this writing, the `Version` key must be set to `1`. This might increment over time as the structure evolves.

The `Expiration` key is an ISO8601 formatted timestamp. If the `Expiration` key isn't present in the tool's output, the CLI assumes that the credentials are long-term credentials that don't refresh. Otherwise, the credentials are considered temporary credentials and are refreshed automatically by rerunning the `credential_process` command before they expire.

Note
The AWS CLI does ***not*** cache external process credentials the way it does assume-role credentials. If caching is required, you must implement it in the external process.

The external process can return a non-zero return code to indicate that an error occurred while retrieving the credentials.

Ways to set this value

Location	Supported	Example

`config` file	Yes	Linux or macOS
		`credential_process = "/path/to/credentials.sh" parameterWithoutSpaces "parameter with spaces"`
		Windows
		`credential_process = "C:\Path\To\credentials.cmd" parameterWithoutSpaces "parameter with spaces"`

`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	Yes	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	Yes	
AWS Toolkit for Visual Studio	Yes	
AWS Toolkit for Visual Studio Code	Yes	
AWS Tools for PowerShell	Yes	

`credential_source`

Used within Amazon EC2 instances or containers to specify where the SDK or development tool can find credentials that have permission to assume the role you specify with the `role_arn` parameter.

Details

Default value: None

Valid values:

- **Environment** – Specifies that the SDK or tool is to retrieve source credentials from the environment variables `AWS_ACCESS_KEY_ID` (p. 18) and `AWS_SECRET_ACCESS_KEY` (p. 19).
- **Ec2InstanceMetadata** – Specifies that the SDK or tool is to use the IAM role attached to the EC2 instance profile to get source credentials.
- **EcsContainer** – Specifies that the SDK or tool is to use the IAM role attached to the ECS container to get source credentials.

You cannot specify both `credential_source` and `source_profile` in the same profile.

Ways to set this value

In the following example, apps that use the AWS SDKs or other AWS tools retrieve credentials for the instance's assigned role from the instance's metadata. The app or tool then uses those credentials to assume the role named `my-role-name`.

Location	Supported	Example
`config` file	Yes	`credential_source = Ec2InstanceMetadata`

Location	Supported	Example
		`role_arn = arn:aws:iam::123456789012:role/my-role-name`
`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	Yes	
AWS SDK for PHP	Yes	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	-	
AWS Tools for PowerShell	Yes	

`duration_seconds`

Specifies the maximum duration of the role session, in seconds.

Details

This setting applies only when the profile specifies to assume a role.

Default value: 3600 seconds (one hour)

Valid values: The value can range from 900 seconds (15 minutes) up to the maximum session duration setting configured for the role (which can be a maximum of 43200 seconds, or 12 hours). For more information, see View the Maximum Session Duration Setting for a Role in the *IAM User Guide*.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`duration_seconds = 43200`

Location	Supported	Example
`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	-	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	-	
AWS Tools for PowerShell	-	

`external_id`

Specifies a unique identifier that is used by third parties to assume a role in their customers' accounts.

Details

This setting applies only when the profile specifies to assume a role and the trust policy for the role requires a value for `ExternalId`. The value maps to the `ExternalId` parameter that is passed to the `AssumeRole` operation when the profile specifies a role. For more information, see How to use an External ID When Granting Access to Your AWS Resources to a Third Party in the *IAM User Guide*.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`external_id = unique_value_assigned_by_3rd_party`
`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	Yes	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	Yes	
AWS Toolkit for Visual Studio	Yes	
AWS Toolkit for Visual Studio Code	Yes	
AWS Tools for PowerShell	Yes	

`max_attempts`

Specifies the maximum number attempts to make on a request.

Details

If this value is not specified, it defaults to a value based on the current value of the `retry_mode (p. 39)` setting.

- `legacy` – Uses the existing `max_attempts` default specific to your SDK (check the Guide for your specific SDK or your SDK's code base)
- `standard` – 3 attempts
- `adaptive` – 3 attempts

Ways to set this value

Location	Supported	Example
`config` file	Yes	`max_attempts = 4`
`credentials` file	-	
Environment variable	Yes	`AWS_MAX_ATTEMPTS`
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	-	

`metadata_service_num_attempts`

Specifies the number of attempts to make before giving up when attempting to retrieve data from the instance metadata service.

Details

When attempting to retrieve credentials on an Amazon EC2 instance that has been configured with an IAM role, the connection to the instance metadata service is attempted only once by default. If you know you're running on an Amazon EC2 instance with an IAM role configured, you can increase this value so that any intermittent failures are retried.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`metadata_service_num_attempts=10`
`credentials` file	-	

Environment variable	Yes	Linux/macOS
		`export AWS_METADATA_SERVICE_NUM_ATTEMPTS=10`
		Windows
		`setx AWS_METADATA_SERVICE_NUM_ATTEMPTS 10`

AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	-	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	-	
AWS Tools for PowerShell	-	

`metadata_service_timeout`

Specifies the number of seconds before timing out when attempting to retrieve data from the instance metadata service.

Details

When attempting to retrieve credentials on an Amazon EC2 instance that has been configured with an IAM role, the connection to the instance metadata service times out after one second by default. If you know you're running on an Amazon EC2 instance with an IAM role configured, you can increase this value, if needed.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`metadata_service_timeout=10`
`credentials` file	-	
Environment variable	Yes	Linux/macOS: `export AWS_METADATA_SERVICE_TIMEOUT=10` Windows: `setx AWS_METADATA_SERVICE_TIMEOUT 10`
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	-	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	-	
AWS Tools for PowerShell	-	

`mfa_serial`

Specifies the identification or serial number of an multi-factor authentication (MFA) device that the user must use when assuming a role.

Details

This setting is required only when the profile specifies to assume a role and the trust policy for that role includes a condition that requires MFA authentication.

Default value: None. This value must be explicitly set.

Valid values: The value can be either a serial number for a hardware device (such as `GAHT12345678`), or an Amazon Resource Name (ARN) for a virtual MFA device. For more information about MFA, see Configuring MFA-Protected API Access in the *IAM User Guide*.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`mfa_serial = arn:aws:iam::123456789012:mfa/my-user-name`
`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	Yes	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	Yes	
AWS Toolkit for Visual Studio	Yes	
AWS Toolkit for Visual Studio Code	Yes	
AWS Tools for PowerShell	Yes	

`output`

(Available in the AWS CLI only.)

Specifies the default output format for AWS CLI commands requested using this profile.

Details

Default value: `json`

Valid values:

- **`json`** – Output is formatted as a JSON string.
- **`yaml`** – Output is formatted as a YAML string. *(Available in the AWS CLI version 2 only.)*
- **`text`** – Output is formatted as multiple lines of tab-separated string values. This can be useful to pass the output to a text processor, like `grep`, `sed`, or `awk`.
- **`table`** – Output is formatted as a table using the characters +|- to form the cell borders. It typically presents the information in a "human-friendly" format that is much easier to read than the others, but not as programmatically useful.

For more information, see Controlling Command Output from the AWS CLI in the *AWS Command Line Interface User Guide*.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`output = json`
`credentials` file	-	

Location	Supported	Example
Environment variable	Yes	Linux/macOS: `export AWS_DEFAULT_OUTPUT=json` Windows: `setx AWS_DEFAULT_OUTPUT json`
AWS CLI parameter	Yes	`--output json`

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	-	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	-	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	-	
AWS Tools for PowerShell	-	

`parameter_validation`

Specifies whether the SDK or tool attempts to validate command line parameters before sending them to the AWS service endpoint.

Details

Default: `true`

Valid values:

- **`true`** – The default. The SDK or tool performs client-side validation of command line parameters. This enables the SDK or tool to ensure that parameters are valid and catch some errors. The SDK or tool can reject invalid requests before sending requests to the AWS service endpoint.
- **`false`** – The SDK or tool doesn't validate command line parameters before sending them to the AWS service endpoint. The AWS service endpoint is responsible for validating all requests and rejecting invalid requests.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`parameter_validation = false`
`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	-	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	-	
AWS Tools for PowerShell	-	

`region`

Specifies the AWS Region to send requests to for commands requested using this profile.

Details

Default value: None. You must specify this value explicitly.

Valid values:

- Any of the Region codes available for the chosen service as listed in AWS Regions and Endpoints in the *AWS General Reference*. For example, the value `us-east-1` sets the endpoint to the AWS Region US East (N. Virginia).
- `aws_global`, to specify the global endpoint for services that support a separate global endpoint in addition to regional endpoints, such as AWS Security Token Service (AWS STS) and Amazon Simple Storage Service (Amazon S3).

Ways to set this value

Location	Supported	Example
`config` file	Yes	`region = us-west-2`
`credentials` file	-	
Environment variable	Yes	Linux/macOS: `export AWS_DEFAULT_REGION=us-west-2` Windows: `setx AWS_DEFAULT_REGION us-west-2`
AWS CLI parameter	Yes	`--region us-west-2`

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	Yes	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	Yes	Supports this setting in only the `[default]` profile. You can change the profile recognized as default by using the environment variable `AWS_PROFILE`. This SDK/Tool doesn't support the environment variable equivalent: `AWS_DEFAULT_REGION` For more information, see Setting an AWS Region in AWS Toolkit for JetBrains.
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	Yes	
AWS Tools for PowerShell	Yes	

retry_mode

Specifies how the SDK or developer tool attempts retries.

Details

Default value: `legacy`

Valid values:

- `legacy` – The current preexisting retry behavior.
- `standard` – The standard set of retry rules across AWS SDKs. This mode includes a standard set of errors that are retried, and support for retry quotas. The default maximum number of attempts with this mode is three, unless `max_attempts (p. 31)` is explicitly configured.
- `adaptive` – An experimental retry mode that includes the functionality of standard mode but includes automatic client-side throttling. Because this mode is experimental, it might change behavior in the future.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`retry_mode=standard`
`credentials` file	-	
Environment variable	Yes	`AWS_RETRY_MODE`
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	-	

role_arn

Specifies the Amazon Resource Name (ARN) of an IAM role that you want to use to perform operations requested using this profile.

Details

Default value: None. You must specify this value explicitly.

Valid values: The value must be the ARN of an IAM role. See the example in the following table.

In addition, you must also specify **one** of the following settings:

- `source_profile` (p. 42) – To identify another profile to use to find credentials that have permission to assume the role in this profile.
- `credential_source` (p. 28) – To use either credentials identified by the current environment variables or credentials attached to an Amazon EC2 instance profile or an Amazon ECS container instance.

Ways to set this value

Location	Supported	Example

`config` file	Yes	`role_arn = arn:aws:iam::123456789012:role/my-role-name`
		`source_profile = profile-with-user-that-can-assume-role`
		
		`role_arn = arn:aws:iam::123456789012:role/my-role-name`
		`credential_source = Ec2InstanceMetadata`

`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	Yes	
AWS SDK for PHP	Yes	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	Yes	
AWS Toolkit for Visual Studio	Yes	

SDK or tool	Su	Notes or more information
AWS Toolkit for Visual Studio Code	Yes	
AWS Tools for PowerShell	Yes	

role_session_name

Specifies the name to attach to the role session. This name appears in AWS CloudTrail logs for entries associated with this session.

Details

Default value: An optional parameter. If you don't provide this value, a session name is generated automatically if the profile assumes a role.

Valid values: Provided to the `RoleSessionName` parameter when the AWS CLI calls the `AssumeRole` operation on your behalf. The value becomes part of the assumed role user Amazon Resource Name (ARN) that you can query, and shows up as part of the CloudTrail log entries for operations invoked by this profile.

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

Ways to set this value

Location	Supported	Example
`config` file	Yes	`role_session_name = my-role-session-name`
`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	Yes	
AWS SDK for PHP	Yes	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	Yes	
AWS Toolkit for Visual Studio	Yes	

SDK or tool	Su	Notes or more information
AWS Toolkit for Visual Studio Code	Yes	
AWS Tools for PowerShell	Yes	

source_profile

Specifies another profile whose credentials are used to assume the role specified by this `role_arn` setting in this profile.

Details

Default value: None.

Valid values: A text string that consists of the name of a profile defined in the `config` and `credentials` files. You must also specify a value for `role_arn` (p. 39) in the current profile.

> **Note**
> This setting is an alternative to `credential_source` (p. 28). You can't specify both `source_profile` and `credential_source` in the same profile.

Ways to set this value

Location	Supported	Example

`config` file	Yes	`source_profile = user-who-can-assume-roleA`
		`role_arn = arn-of-roleA`

`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	Yes	
AWS SDK for PHP	Yes	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	Yes	

SDK or tool	Su	Notes or more information
AWS Toolkit for Visual Studio	Yes	
AWS Toolkit for Visual Studio Code	Yes	
AWS Tools for PowerShell	Yes	

sso_account_id

Specifies the AWS account ID that contains the IAM role with the permission that you want to grant to the associated AWS Single Sign-On (AWS SSO) user.

Details

When the AWS SSO settings are specified in a profile, the SDK or tool connects to the AWS SSO portal specified in `sso_start_url` (p. 46) and `sso_region` (p. 44). After the user successfully authenticates with AWS SSO, the portal returns short-term credentials for the IAM role specified by the `sso_account_id` (p. 43) and `sso_role_name` (p. 45).

Default value: None. You must set this value explicitly if you want to use AWS SSO.

Valid values: Must be a valid AWS account ID number.

If you set this value, you must also set `sso_region` (p. 44), `sso_role_name` (p. 45), and `sso_start_url` (p. 46) in the same profile.

Ways to set this value

Location	Supported	Example

`config` file	Yes	`sso_account_id = 123456789012`
		`sso_start_url = https://my-sso-portal.awsapps.com/start`
		`sso_region = us-east-1`
		`sso_role_name = SSOReadOnlyRole`

`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and Tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	*(Available in the AWS CLI V2 only.)*

SDK or tool	Su	Notes or more information
		Configuring the AWS CLI to use AWS Single Sign-On
AWS SDK for .NET	-	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	-	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	-	
AWS Tools for PowerShell	-	

`sso_region`

Specifies the AWS Region that contains the AWS Single Sign-On (AWS SSO) portal host. This is separate from and can be a different AWS Region than that specified by the default `region` (p. 37) parameter.

Details

When the AWS SSO settings are specified in a profile, the SDK or tool connects to the AWS SSO portal specified in `sso_start_url` (p. 46) and `sso_region` (p. 44). After the user successfully authenticates with AWS SSO, the portal returns short-term credentials for the IAM role specified by the `sso_account_id` (p. 43) and `sso_role_name` (p. 45).

Default value: None. You must set this value explicitly if you want to use AWS SSO.

Valid values: This value must be a valid code for an AWS Region. For example, `us-west-2`.

For a complete list of the AWS Regions and their codes, see Regional Endpoints in the *Amazon Web Services General Reference*.

If you set this value, you must also set `sso_account_id` (p. 43), `sso_role_name` (p. 45), and `sso_start_url` (p. 46) in the same profile.

Ways to set this value

Location	Supported	Example

`config` file	Yes	`sso_region = us-east-1`
		`sso_account_id`
		` = 123456789012`
		`sso_start_url = https://my-`
		`sso-portal.awsapps.com/start`
		`sso_role_name`
		` = SSOReadOnlyRole`

`credentials` file	-	

Location	Supported	Example
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	*(Available in the AWS CLI V2 only.)* Configuring the AWS Command Line Interface to use AWS Single Sign-On
AWS SDK for .NET	-	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	-	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	-	
AWS Tools for PowerShell	-	

`sso_role_name`

Specifies the friendly name of the IAM role that defines the user's permissions when using this profile to get credentials through AWS SSO.

Details

When the AWS SSO settings are specified in a profile, the SDK or tool connects to the AWS SSO portal specified in `sso_start_url` (p. 46) and `sso_region` (p. 44). After the user successfully authenticates with AWS SSO, the portal returns short-term credentials for the IAM role specified by the `sso_account_id` (p. 43) and `sso_role_name` (p. 45).

Default value: None. You must set this value explicitly if you want to use AWS SSO.

Valid values: This value must be the friendly name for an IAM role, not the Amazon Resource Name (ARN). The role must exist in the AWS account specified by `sso_account_id` (p. 43).

If you set this value, you must also set `sso_account_id` (p. 43), `sso_region` (p. 44), and `sso_start_url` (p. 46) in the same profile. For more information about using AWS SSO and the AWS CLI, see Configuring the AWS CLI to use AWS Single Sign-On in the *AWS Command Line Interface User Guide*.

Ways to set this value

Location	Supported	Example

`config` file	Yes	`sso_role_name = SSOReadOnlyRole`
		`sso_region = us-east-1`
		`sso_account_id = 123456789012`
		`sso_start_url = https://my-sso-portal.awsapps.com/start`

`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	*(Available in the AWS CLI V2 only.)* Configuring the AWS CLI to use AWS Single Sign-On
AWS SDK for .NET	-	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	-	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	-	
AWS Tools for PowerShell	-	

`sso_start_url`

Specifies the URL that points to the organization's AWS Single Sign-On (AWS SSO) user portal.

Details

When the AWS SSO settings are specified in a profile, the SDK or tool connects to the AWS SSO portal specified in `sso_start_url` (p. 46) and `sso_region` (p. 44). After the user successfully authenticates with AWS SSO, the portal returns short-term credentials for the IAM role specified by the `sso_account_id` (p. 43) and `sso_role_name` (p. 45).

Default value: None. You must set this value explicitly if you want to use AWS SSO.

Valid values: Must be a valid URL that points to your organization's AWS SSO portal.

If you set this value, you must also set `sso_account_id` (p. 43), `sso_region` (p. 44), and `sso_role_name` (p. 45) in the same profile.

Ways to set this value

Location	Supported	Example

`config` file	Yes	`sso_start_url = https://my-sso-portal.awsapps.com/start`
		`sso_role_name = SSOReadOnlyRole`
		`sso_region = us-east-1`
		`sso_account_id = 123456789012`

`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	*(Available in the AWS CLI V2 only.)* Configuring the AWS CLI to use AWS Single Sign-On
AWS SDK for .NET	-	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	-	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	-	
AWS Tools for PowerShell	-	

`sts_regional_endpoints`

Specifies how the SDK or tool determines the AWS service endpoint that it uses to talk to the AWS Security Token Service (AWS STS).

Details

Default value: `legacy` (exception: **AWS CLI version 2** uses `regional` as the default value)

Valid values:

- **`legacy`** – Uses the global AWS STS endpoint, `sts.amazonaws.com`, for the following AWS Regions: `ap-northeast-1`, `ap-south-1`, `ap-southeast-1`, `ap-southeast-2`, `aws-global`, `ca-central-1`, `eu-central-1`, `eu-north-1`, `eu-west-1`, `eu-west-2`, `eu-west-3`, `sa-east-1`, `us-east-1`, `us-east-2`, `us-west-1`, and `us-west-2`. All other Regions automatically use their respective regional endpoint.

- **`regional`** – The SDK or tool always uses the AWS STS endpoint for the currently configured Region. For example, if the client is configured to use `us-west-2`, all calls to AWS STS are made to the regional endpoint `sts.us-west-2.amazonaws.com`, instead of the global `sts.amazonaws.com` endpoint. To send a request to the global endpoint while this setting is enabled, you can set the Region to `aws-global`.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`sts_regional_endpoints = regional`
`credentials` file	-	
Environment variable	Yes	Linux/macOS: `export AWS_STS_REGIONAL_ENDPOINTS=regional` Windows: `setx AWS_STS_REGIONAL_ENDPOINTS regional`
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	Default behavior for version 2 changed value to `regional`.
AWS SDK for .NET	Yes	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	-	
AWS Tools for PowerShell	Yes	

tcp_keepalive

Specifies whether the SDK or development tool uses the TCP Keep-Alive socket option when creating connections.

Details

Default value: `true`

Valid values:

- **`true`** – The default. The SDK or tool uses TCP keepalive packets.
- **`false`** – The SDK or tool doesn't use TCP keepalive packets.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`tcp_keepalive = false`
`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	-	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	-	
AWS Tools for PowerShell	-	

web_identity_token_file

Specifies the path to a file that contains an access token from a supported OAuth 2.0 provider or OpenID Connect ID identity provider.

Details

This setting enables authentication by using Web Identity Federation providers, such as Google, Facebook, and Amazon, among many others. The SDK or developer tool loads the contents of this file and passes it as the `WebIdentityToken` argument when it calls the `AssumeRoleWithWebIdentity` operation on your behalf.

Default value: None. You must specify this value explicitly.

Valid values: This value must be a path and file name. The file must contain an OAuth 2.0 access token or an OpenID Connect token that was provided to you by an identity provider.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`web_identity_token_file = ~/file/token.txt`
`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	Yes	
AWS SDK for PHP	-	
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code	-	
AWS Tools for PowerShell	Yes	

SDK and tool settings for Amazon S3 APIs

The topics in this section describe the Amazon S3 service-specific settings that can be stored in the shared AWS `config` file.

For information about the shared `config` and `credentials` files, see The shared config and credentials files (p. 3).

AWS SDKs and tools that support these settings have several options that configure how those SDKs and tools perform and optimize Amazon S3 operations.

You can configure all of these options by specifying the following nested setting in your `config` file.

```
[profile my-profile-name]
... global settings ...
s3 =
    ... individual s3 settings ...
```

Each Amazon S3 setting is then indented by one or more spaces on its own line. The following example sets a variety of settings.

```
[profile development]
s3 =
    max_concurrent_requests = 20
    max_queue_size = 10000
    multipart_threshold = 64MB
    multipart_chunksize = 16MB
    max_bandwidth = 50MB/s
    use_accelerate_endpoint = true
    addressing_style = path
```

Note
These settings are entirely optional. You should be able to successfully use the Amazon S3 operations without configuring any of these settings. The Amazon S3 settings are provided to enable you to tune for performance or to account for the specific environment where you are running these operations.

Setting	Summary	Config file	Credentials file	Env variable	Per operation
`addressing_style` (p. 53)	Specifies whether the SDK or tool recognizes the Amazon S3 bucket name is part of the hostname or as part of the URL path.	Yes	-	-	-
`max_bandwidth` (p. 54)	Specifies the maximum bandwidth that the SDK or tool can consume to upload and download data to and from Amazon S3.	Yes	-	-	-

Setting	Summary	Config file	Credentials file	Env variable	Per operation
`max_concurrent_requests` (p. 55)	Specifies the maximum number of concurrent Amazon S3 transfer requests that the SDK or tool can run at the same time.	Yes	-	-	-
`max_queue_size` (p. 56)	Specifies the maximum number of transfer tasks in the SDK or tool task queue.	Yes	-	-	-
`multipart_chunksize` (p. 57)	Specifies the chunk size that the SDK or tool uses for Amazon S3 multipart transfers of individual files.	Yes	-	-	-
`multipart_threshold` (p. 59)	Specifies the size threshold the SDK or tool uses to trigger Amazon S3 multipart transfers of individual files.	Yes	-	-	-
`payload_signing_enabled` (p. 60)	Specifies whether the SDK or tool uses SHA256 to sign `sigv4` payloads for Amazon S3 commands.	Yes	-	-	-
`use_accelerate_endpoint` (p. 61)	Specifies that the SDK or tool use the Amazon S3 Accelerate endpoint for all Amazon S3 operations.	Yes	-	-	-

Setting	Summary	Config file	Credentials file	Env variable	Per operation
use_dualstack_endpoint (p. 62)	Specifies that the SDK or tool use the Amazon S3 dual IPv4/IPv6 endpoint for all Amazon S3 operations.	Yes	-	-	-

addressing_style

Specifies which addressing style to use. This controls whether the bucket name is in the hostname or is part of the URL.

Details

Default value: auto

Valid values:

- **path** – Treats the bucket name as if it's a path in the URI; for example, `https://s3.amazonaws.com/bucketname`. If you set the addressing style to path, you must ensure that the AWS Region you configured in the SDK or tool matches the Region of your bucket.
- **virtual** – Includes the bucket name as part of the hostname; for example, `https://bucketname.s3.amazonaws.com`.
- **auto** – The default value. The SDK or tool attempts to use the `virtual` style where it can, but will fall back to `path` style when required. For example, if your bucket name is not DNS compatible, the bucket name cannot be part of the hostname and must be in the path. With `auto`, the SDK or tool detects this condition and automatically switches to `path` style for you.

Ways to set this value

Location	Supported	Example

`config` file	Yes	`s3 =`
		`    addressing_style = path`

`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	-	
AWS SDK for PHP		
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code		
AWS Tools for PowerShell	-	

`max_bandwidth`

This setting applies only to AWS CLI commands in the `s3` namespace.

Specifies the maximum bandwidth that can be consumed for uploading data to and downloading data from Amazon S3.

Details

This setting limits the maximum bandwidth that the SDK or tool can use to transfer data to and from Amazon S3. This value applies to only uploads and downloads; it doesn't apply to copies or deletes.

Default value: The default value is no limit.

Valid values: The value is expressed as bytes per second. The value can be specified as:

- An integer. For example, `1048576` sets the maximum bandwidth usage to 1 megabyte per second.
- An integer followed by a rate suffix. You can specify rate suffixes using: `KB/s`, `MB/s`, or `GB/s`; for example, `300KB/s` or `10MB/s`.

In general, we recommend that you first try to lower bandwidth consumption by lowering `max_concurrent_requests`. If that doesn't adequately limit bandwidth consumption to the desired rate, you can use the `max_bandwidth` setting to further limit bandwidth consumption. This is because `max_concurrent_requests` controls how many threads are currently running. If you instead first lower `max_bandwidth` but leave a high `max_concurrent_requests` setting, it can result in threads having to wait unnecessarily. This can lead to excess resource consumption and connection timeouts.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`s3 =`

Location	Supported	Example
		`max_bandwidth = 5MB/s`
`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	Applies **only** to AWS CLI commands in the `s3` namespace.
AWS SDK for .NET	-	
AWS SDK for PHP		
AWS SDK for Python (Boto3)	-	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code		
AWS Tools for PowerShell	-	

max_concurrent_requests

This setting applies only to AWS CLI commands in the `s3` namespace.

Specifies the maximum number of concurrent Amazon S3 transfer requests that can run at the same time.

Details

The `aws s3` transfer commands are multithreaded. At any given time, multiple Amazon S3 requests can be running. For example, when you use the command:

```
$ aws s3 cp some/local/dir s3://bucket/ --recursive
```

to upload files to an Amazon S3 bucket, the AWS CLI can upload the files `some/local/dir/file1`, `some/local/dir/file2`, and `some/local/dir/file3` in parallel. This setting limits the number of transfer operations that can run at the same time.

Default value: `10`

You might need to change this value for a few reasons:

- Decreasing this value – On some environments, the default of 10 concurrent requests can overwhelm a system. This can cause connection timeouts or slow the responsiveness of the system. Lowering this value makes the Amazon S3 transfer commands less resource intensive. The tradeoff is that Amazon S3 transfers can take longer to complete. Lowering this value might be necessary if your environment includes a tool that limits your available bandwidth.

- Increasing this value – In some scenarios, you might want the Amazon S3 transfers to complete as quickly as possible, using as much network bandwidth as necessary. In this scenario, the default number of concurrent requests might not be enough to use all of the available network bandwidth. Increasing this value can improve the time it takes to complete an Amazon S3 transfer.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`s3 = max_concurrent_requests = 20`
`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or Tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	Applies **only** to commands in the `s3` namespace.
AWS SDK for .NET	-	
AWS SDK for PHP		
AWS SDK for Python (Boto3)	-	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code		
AWS Tools for PowerShell	-	

`max_queue_size`

This setting applies only to AWS CLI commands in the `s3` names*pace.*

Specifies the maximum number of transfer tasks in the Amazon S3 task queue.

Details

The AWS CLI internally uses a model where it queues up Amazon S3 tasks that are then executed by consumers whose numbers are limited by `max_concurrent_requests`. A task generally maps to a single Amazon S3 operation. For example, a task could be a `PutObjectTask`, `GetObjectTask`, or `UploadPartTask`. The rate at which tasks are added to the queue can be much faster than the rate at which consumers finish the tasks. To avoid unbounded growth, the task queue size is capped to a specific size. This setting changes the value of that maximum number.

Default value: 1000

You generally don't need to change this setting. This setting also corresponds to the number of tasks that the AWS CLI is aware of that need to run. This means that by default the AWS CLI can only see 1000 tasks ahead. Increasing this value means that the CLI can more quickly know the total number of tasks needed, assuming that the queuing rate is quicker than the rate of task completion. The tradeoff is that a larger `max_queue_size` requires more memory.

Ways to set this value

Location	Supported	Example

`config` file	Yes	`s3 =`
		`    max_queue_size = 2000`

`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	Applies **only** to commands in the `s3` namespace.
AWS SDK for .NET	-	
AWS SDK for PHP		
AWS SDK for Python (Boto3)	-	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code		
AWS Tools for PowerShell	-	

`multipart_chunksize`

This setting applies only to AWS CLI commands in the `s3` namespace.

Specifies the chunk size that the AWS CLI uses for Amazon S3 multipart transfers of individual files.

Details

When the AWS CLI performs an Amazon S3 file transfer where the overall file size exceeds the `multipart_threshold` size, the AWS CLI divides the file into chunks of this setting's size. This value can be specified either as the number of bytes as an integer, or by using a size and a suffix.

Default value: 8MB

Valid values: The value is expressed as the number of bytes in the chunk. The value can be specified as:

- An integer. For example, `1048576` sets the chunk size to 1 megabyte.
- An integer followed by a size suffix. You can specify rate suffixes using the following: `KB`, `MB`, `GB`, or `TB`. For example, `300KB` or `10MB`.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`s3 = multipart_chunksize = 5MB`
`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	Applies **only** to commands in the `s3` namespace.
AWS SDK for .NET	-	
AWS SDK for PHP		
AWS SDK for Python (Boto3)	-	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code		
AWS Tools for PowerShell	-	

`multipart_threshold`

This setting applies only to AWS CLI commands in the `s3` namespace.

Specifies the size threshold that the AWS CLI uses to trigger Amazon S3 multipart transfers of individual files.

Details

When uploading, downloading, or copying a file, the `aws s3` commands switch to multipart operations if the file exceeds this size. This value can be specified either as the number of bytes as an integer, or by using a size and a suffix.

Default value: 8MB

Valid values: The value is expressed as the number of bytes in the chunk. The value can be specified as:

- An integer. For example, `1048576` sets the chunk size to 1 megabyte.
- An integer followed by a size suffix. You can specify rate suffixes using: `KB`, `MB`, `GB`, or `TB`. For example, `300KB` or `10MB`.

Note
Amazon S3 can impose limits on valid values that can be used for multipart operations. For more information, see the S3 Multipart Upload documentation in the *Amazon Simple Storage Service Developer Guide*.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`s3 = multipart_threshold = 5GB`
`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or Tool	Su	Notes / More information
AWS Command Line Interface (AWS CLI)	Yes	Applies **only** to commands in the `s3` namespace.
AWS SDK for .NET	-	
AWS SDK for PHP		
AWS SDK for Python (Boto3)	-	

SDK or Tool	Su	Notes / More information
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code		
AWS Tools for PowerShell	-	

payload_signing_enabled

Specifies whether to SHA256 sign sigv4 payloads for Amazon S3 operations.

Details

Default value: `false`

Valid values:

- **true** – Amazon S3 requests receive additional content validation in the form of an SHA256 checksum, which is calculated for you and included in the request signature. If set to `false`, the checksum isn't calculated. Disabling this can be useful to reduce the performance overhead created by the checksum calculation.

- **false** – This is the default value. Specifies that the SDK or tool should not calculate the checksum for streaming uploads (`UploadPart` and `PutObject`) when using HTTPS and a `ContentMD5` is present.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`s3 = payload_signing_enabled = true`
`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or Tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	-	

SDK or Tool	Su	Notes or more information
AWS SDK for PHP		
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code		
AWS Tools for PowerShell	-	

use_accelerate_endpoint

Details

Default value: `false`

Valid values:

- **`false`** –The default value. The SDK or tool directs all Amazon S3 requests to the standard endpoint for the configured Region.
- **`true`** – The SDK or tool directs all Amazon S3 requests to the `S3 Accelerate` endpoint at `s3-accelerate.amazonaws.com`. To use this endpoint, you must enable your bucket to use `S3 Accelerate`. All requests are sent using the virtual style of bucket addressing: *`my-bucket`*`.s3-accelerate.amazonaws.com`. Note that `ListBuckets`, `CreateBucket`, and `DeleteBucket` requests aren't sent to the S3 Accelerate endpoint because that endpoint doesn't support those operations. This behavior can also be set if the `--endpoint-url` command-line parameter is set to `https://s3-accelerate.amazonaws.com` or `http://s3-accelerate.amazonaws.com` for any `s3` or `s3api` command.

Note
You can't set both `use_accelerate_endpoint` and `use_dualstack_endpoint (p. 62)` to `true`. They are mutually exclusive.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`s3 = use_accelerate_endpoint = true`
`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or Tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	-	
AWS SDK for PHP		
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code		
AWS Tools for PowerShell	-	

`use_dualstack_endpoint`

Details

Default value: `false`

Valid values:

- **`false`** – The default value. The SDK or tool directs all Amazon S3 requests to the standard endpoint for the configured Region.
- **`true`** – The SDK or tool directs all Amazon S3 requests to the dual IPv4/IPv6 endpoint for the configured Region.

Note
You can't set both `use_dualstack_endpoint` and `use_accelerate_endpoint` to `true`. They are mutually exclusive.

Ways to set this value

Location	Supported	Example
`config` file	Yes	`s3 = use_dualstack_endpoint = true`
`credentials` file	-	
Environment variable	-	
AWS CLI parameter	-	

Compatibility with AWS SDKS and tools

SDK or tool	Su	Notes or more information
AWS Command Line Interface (AWS CLI)	Yes	
AWS SDK for .NET	-	This can be set in code by using the **`Config`** property of each `Client` class. See specifically the **`UseDualstackEndpoint`** property, which is inherited from Amazon.Runtime.IClientConfig.
AWS SDK for PHP		
AWS SDK for Python (Boto3)	Yes	
AWS Toolkit for JetBrains	-	
AWS Toolkit for Visual Studio	-	
AWS Toolkit for Visual Studio Code		
AWS Tools for PowerShell	-	

Supported environment variables

This topic lists the supported environment variables that some SDKs and tools enable you to use in place of settings in the shared `config` and `credentials` files. For information about these files, see The shared config and credentials files (p. 3).

> **Note**
> Not all SDKs and Tools that support a Config setting support the equivalent environment variable setting. See the details page for a Config setting for more information.

The listing for each variable links back to the Settings (p. 11) page for details about what that variable does.

Environment variable	Description or link to `Config` setting and reference
`AWS_ACCESS_KEY_ID`	`aws_access_key_id` (p. 18)
`AWS_CA_BUNDLE`	`ca_bundle` (p. 21)
`AWS_DEFAULT_OUTPUT`	`output` (p. 35)
`AWS_CONFIG_FILE`	Specifies the path to the shared `config` file. The default value is `~/.aws/config`. Change this only if you want to move the file to a different location.
`AWS_DEFAULT_REGION`	`region` (p. 37)
`AWS_MAX_ATTEMPTS`	`max_attempts` (p. 31)
`AWS_METADATA_SERVICE_NUM_ATTEMPTS`	`metadata_service_num_attempts` (p. 32)

Environment variable	Description or link to `Config` setting and reference
`AWS_METADATA_SERVICE_TIMEOUT`	`metadata_service_timeout` (p. 33)
`AWS_PROFILE`	Specifies which profile is the default profile. The default value is `[default]`.
`AWS_RETRY_MODE`	`retry_mode` (p. 39)
`AWS_SECRET_ACCESS_KEY`	`aws_secret_access_key` (p. 19)
`AWS_SESSION_TOKEN`	`aws_session_token` (p. 20)
`AWS_SHARED_CREDENTIALS_FILE`	Specifies the path to the shared `credentials` file. The default value is `~/.aws/credentials`. Change this only if you want to move the file to a different location.
`AWS_STS_REGIONAL_ENDPOINTS`	`sts_regional_endpoints` (p. 47)
`CA_BUNDLE`	`ca_bundle` (p. 21)

AWS Common Runtime (CRT) libraries

The AWS Common Runtime (CRT) libraries are a new base library of the SDKs. The CRT is a modular family of nine independent packages, written in C, that each provide good performance and minimal footprint for different required functionalities. These functionalities are common and shared across all SDKs allowing for better code reuse, optimization, and accuracy. The packages are:

- `awslabs/aws-c-common`: Basic data structures, threading/synchronization primitives, buffer management, stdlib-related functions
- `awslabs/aws-c-io`: Sockets (TCP, UDP), DNS, pipes, event loops, channels, SSL/TLS
- `awslabs/aws-c-mqtt`: MQTT is a standard, lightweight messaging protocol for the Internet of Things (IoT)
- `awslabs/aws-c-http`: HTTP implementation
- `awslabs/aws-c-cal`: Cryptographic primitives, hashes (MD5, SHA256, SHA256 HMAC), signers, AES
- `awslabs/aws-c-auth`: AWS client-side authentication (standard credential providers and signing (sigv4))
- `awslabs/aws-c-compression`: Compression algorithms (huffman encoding/decoding)
- `awslabs/aws-c-event-stream`: Event stream message processing (headers, prelude, payload, crc/trailer), remote procedure call (RPC) implementation over event streams
- `awslabs/aws-checksums`: Cross-platform hardware-accelerated CRC32c and CRC32 with fallback to efficient software implementations

The CRT is being adopted by all SDKs except Go.

Maintenance and support

For an overview of tools that can help you develop applications on AWS, see Tools to Build on AWS. For information on support, see the AWS Knowledge Center.

The following topics cover the maintenance and version support policies for AWS SDKs.

Topics

AWS SDKs and Tools maintenance policy

Overview

This document outlines the maintenance policy for AWS Software Development Kits (SDKs) and Tools, including Mobile and IoT SDKs, and their underlying dependencies. AWS regularly provides the AWS SDKs and Tools with updates that may contain support for new or updated AWS APIs, new features, enhancements, bug fixes, security patches, or documentation updates. Updates may also address changes with dependencies, language runtimes, and operating systems. AWS SDK releases are published to package managers (e.g. Maven, NuGet, PyPI), and are available as source code on GitHub.

We recommend users to stay up-to-date with SDK releases to keep up with the latest features, security updates, and underlying dependencies. Continued use of an unsupported SDK version is not recommended and is done at the user's discretion.

Versioning

The AWS SDK release versions are in the form of X.Y.Z where X represents the major version. Increasing the major version of an SDK indicates that this SDK underwent significant and substantial changes to support new idioms and patterns in the language. Major versions are introduced when public interfaces (e.g. classes, methods, types, etc.), behaviors, or semantics have changed. Applications need to be updated in order for them to work with the newest SDK version. It is important to update major versions carefully and in accordance with the upgrade guidelines provided by AWS.

SDK major version life-cycle

The life-cycle for major SDKs and Tools versions consists of 5 phases, which are outlined below.

- *Developer Preview (Phase 0)* - During this phase, SDKs are not supported, should not be used in production environments, and are meant for early access and feedback purposes only. It is possible for future releases to introduce breaking changes. Once AWS identifies a release to be a stable product, it may mark it as a Release Candidate. Release Candidates are ready for GA release unless significant bugs emerge, and will receive full AWS support.
- *General Availability (GA) (Phase 1)* - During this phase, SDKs are fully supported. AWS will provide regular SDK releases that include support for new services, API updates for existing services, as well as bug and security fixes. For Tools, AWS will provide regular releases that include new feature updates and bug fixes. AWS will support the GA version of an SDK for *at least 24 months*.

- *Maintenance Announcement (Phase 2)* - AWS will make a public announcement at least 6 months before an SDK enters maintenance mode. During this period, the SDK will continue to be fully supported. Typically, maintenance mode is announced at the same time as the next major version is transitioned to GA.
- *Maintenance (Phase 3)* - During the maintenance mode, AWS limits SDK releases to address critical bug fixes and security issues only. An SDK will not receive API updates for new or existing services, or be updated to support new regions. Maintenance mode has a *default duration of 12 months*, unless otherwise specified.
- *End-of-Support (Phase 4)* - When an SDK reaches end-of support, it will no longer receive updates or releases. Previously published releases will continue to be available via public package managers and the code will remain on GitHub. The GitHub repository may be archived. Use of an SDK which has reached end-of-support is done at the user's discretion. We recommend users upgrade to the new major version.

The following is a visual illustration of the SDK major version life-cycle. Please note that the timelines shown below are illustrative and not binding.

Dependency life-cycle

Most AWS SDKs have underlying dependencies, such as language runtimes, operating systems, or third party libraries and frameworks. These dependencies are typically tied to the language community or the vendor who owns that particular component. Each community or vendor publishes their own end-of-support schedule for their product.

The following terms are used to classify underlying third party dependencies:

- *Operating System (OS):* Examples include Amazon Linux AMI, Amazon Linux 2, Windows 2008, Windows 2012, Windows 2016, etc.
- **Language Runtime:** Examples include Java 7, Java 8, Java 11, .NET Core, .NET Standard, .NET PCL, etc.
- **Third party Library / Framework:* Examples include OpenSSL, .NET Framework 4.5, Java EE, etc.

Our policy is to continue supporting SDK dependencies for* at least 6 months* after the community or vendor ends support for the dependency. This policy, however, could vary depending on the specific dependency.

> **Note**
> AWS reserves the right to stop support for an underlying dependency without increasing the major SDK version

Communication methods

Maintenance announcements are communicated in several ways:

- An email announcement is sent to affected accounts, announcing our plans to end support for the specific SDK version. The email will outline the path to end-of-support, specify the campaign timelines, and provide upgrade guidance.
- AWS SDK documentation, such as API reference documentation, user guides, SDK product marketing pages, and GitHub readme(s) are updated to indicate the campaign timeline and provide guidance on upgrading affected applications.
- An AWS blog post is published that outlines the path to end-of-support, as well as reiterates the campaign timelines.
- Deprecation warnings are added to the SDKs, outlining the path to end-of-support and linking to the SDK documentation.

To see the list of available major versions of AWS SDKs and Tools and where they are in their maintenance life cycle, see the section called "Version support matrix" (p. 68).

AWS SDKs and Tools version support matrix

The matrix below shows the list of available AWS SDK major versions and where they are in the maintenance life cycle with associated timelines. For detailed information on the life-cycle for the major versions of Software Development Kits (SDKs) and Tools and their underlying dependencies see the section called "Maintenance policy" (p. 66)

SDK	Major version	General Availability Date	Current Phase	Next Phase
SDK for Java	1.x	3/25/2010	Full Support	
SDK for Java	2.x	11/20/2018	Full Support	
SDK for .NET	1.x	11/2009	End of Support	
SDK for .NET	2.x	11/8/2013	End of support	
SDK for .NET	3.x	7/28/2015	Full Support	
AWS CLI	1.x	9/2/2013	Full Support	
AWS CLI	2.x	2/10/2020	Full Support	
SDK for JavaScript	1.x	5/6/2013	Full Support	Maintenance Announcement
SDK for JavaScript	2.x	6/19/2014	Full Support	
SDK for JavaScript	3.x	12/15/2020	Full Support	
SDK for PHP	2.x	11/2/2012	End of Support	
SDK for PHP	3.x	5/27/2015	Full Support	
SDK for Ruby	1.x	7/14/2011	End of Support	
SDK for Ruby	2.x	2/15/2015	Full Support	Maintenance Announcement
SDK for Ruby	3.x	8/29/2017	Full Support	
SDK for Python (Boto2)	1.x	7/13/2011	End of Support	
SDK for Python (Boto3)	1.x	6/22/2015	Full Support	
SDK for Python (Botocore)	1.x	6/22/2015	Full Support	
SDK for Go v2	2.x	1/19/2021	Full Support	
SDK for Go	1.x	11/19/2015	Full Support	
SDK for C++	1.x	9/2/2015	Full Support	

SDK	Major version	General Availability Date	Current Phase	Next Phase
Tools for PowerShell	2.x	11/8/2013	Full Support	Maintenance Announcement
Tools for PowerShell	3.x	7/29/2015	Full Support	
Tools for PowerShell	4.x	11/21/2019	Full Support	

Document history for AWS SDKs and Tools Reference Guide

The following table describes important additions and updates to the *AWS SDKs and Tools Reference Guide*. For notification about updates to this documentation, you can subscribe to the RSS feed.

update-history-change	update-history-description	update-history-date
Initial release	The first release of this guide is released to the public.	March 13, 2020

AWS glossary

For the latest AWS terminology, see the AWS glossary in the *AWS General Reference*.