Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.
# Table of Contents

Introduction ................................................................................................................................. 1

What's in This Guide .................................................................................................................. 2

Setting up ...................................................................................................................................... 3

Sign up for AWS ......................................................................................................................... 3

Create an IAM user ...................................................................................................................... 3

Getting started ........................................................................................................................... 5

Set up an Azure DevOps account ............................................................................................ 5

Install the AWS Toolkit for Azure DevOps extension ............................................................... 5

Establish AWS credentials for the AWS Toolkit for Azure DevOps ......................................... 5

Supply task credentials ............................................................................................................. 5

Supply task credentials using a service connection ................................................................. 6

Supply credentials through named variables in your build ....................................................... 6

Supply standard AWS environment variables in the build agent process ............................ 7

Supply credentials with Amazon EC2 build agents ................................................................. 7

Using the tools .......................................................................................................................... 8

Archiving build artifacts to AWS ............................................................................................ 8

Prerequisites .............................................................................................................................. 8

Archiving build artifacts with the AWS S3 Upload task .......................................................... 8

Deploying an ASP.NET web app to AWS ................................................................................ 14

Prerequisites ............................................................................................................................. 8

Deploying an ASP.NET application using the AWS Elastic Beanstalk Deploy Application task .... 14

Task reference ........................................................................................................................... 21

AWS CLI ....................................................................................................................................... 22

Synopsis .................................................................................................................................... 22

Description ............................................................................................................................... 22

Parameters ................................................................................................................................. 22

Task Permissions ...................................................................................................................... 23

AWS Tools for Windows PowerShell Script ............................................................................ 23

Synopsis .................................................................................................................................... 22

Description ............................................................................................................................... 22

Parameters ................................................................................................................................. 22

Task Permissions ...................................................................................................................... 23

AWS Shell Script ....................................................................................................................... 25

Synopsis .................................................................................................................................... 22

Description ............................................................................................................................... 22

Parameters ................................................................................................................................. 22

Task Permissions ...................................................................................................................... 23

AWS CloudFormation Create/Update Stack ............................................................................. 27

Synopsis .................................................................................................................................... 22

Description ............................................................................................................................... 22

Parameters ................................................................................................................................. 22

Task Permissions ...................................................................................................................... 23

AWS CloudFormation Delete Stack .......................................................................................... 33

Synopsis .................................................................................................................................... 22

Description ............................................................................................................................... 22

Parameters ................................................................................................................................. 22

Task Permissions ...................................................................................................................... 23

AWS CloudFormation Execute Change Set ............................................................................. 34

Synopsis .................................................................................................................................... 22

Description ............................................................................................................................... 22

Parameters ................................................................................................................................. 22

Task Permissions ...................................................................................................................... 23

AWS CodeDeploy Application Deployment ............................................................................... 36

Synopsis .................................................................................................................................... 22
AWS Toolkit for Microsoft Azure DevOps

AWS Toolkit for Microsoft Azure DevOps is an extension for Microsoft Azure DevOps (formerly known as Visual Studio Team Services or VSTS). It contains tasks you can use in build and release definitions in Azure DevOps and Microsoft Azure DevOps Server (previously named Visual Studio Team Foundation Server) to interact with AWS services. AWS Toolkit for Azure DevOps is available through the Visual Studio Marketplace. See the *Getting started (p. 5)* topic for more information.

You can use these tasks in an Azure DevOps project or in an on-premises Azure DevOps Server environment. The available AWS tasks include the following.

**Deployment tasks**

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS CodeDeploy Application Deployment (p. 36)</td>
<td>Deploys an application to Amazon EC2 instances.</td>
</tr>
<tr>
<td>AWS CloudFormation Create/Update Stack (p. 27)</td>
<td>Creates a new AWS CloudFormation stack or updates the stack if it exists.</td>
</tr>
<tr>
<td>AWS CloudFormation Delete Stack (p. 33)</td>
<td>Deletes an AWS CloudFormation stack.</td>
</tr>
<tr>
<td>AWS CloudFormation Execute Change Set (p. 34)</td>
<td>Executes an AWS CloudFormation change set to create or update a stack.</td>
</tr>
<tr>
<td>AWS Elastic Beanstalk Create Version (p. 41)</td>
<td>Creates a new version of an application.</td>
</tr>
<tr>
<td>AWS Elastic Beanstalk Deploy Application (p. 43)</td>
<td>Deploys a new version of an application to an Elastic Beanstalk environment.</td>
</tr>
<tr>
<td>Amazon ECR Push (p. 39)</td>
<td>Pushes a Docker image to the Amazon Elastic Container Registry (ECR).</td>
</tr>
<tr>
<td>AWS Lambda Deploy Function (p. 46)</td>
<td>Supports deployment of AWS Lambda functions for all supported Lambda language runtimes.</td>
</tr>
<tr>
<td>AWS Lambda .NET Core (p. 51)</td>
<td>Builds, packages, and deploys a .NET Core AWS Lambda function or serverless application.</td>
</tr>
<tr>
<td>AWS Lambda Invoke Function (p. 50)</td>
<td>Invokes an AWS Lambda function with a JSON payload.</td>
</tr>
</tbody>
</table>

**General purpose tasks**

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS CLI (p. 22)</td>
<td>Runs a command using the AWS CLI.</td>
</tr>
<tr>
<td>AWS Shell Script (p. 25)</td>
<td>Run a shell script using Bash with AWS credentials.</td>
</tr>
</tbody>
</table>
## Task Reference

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon S3 Download (p. 54)</td>
<td>Downloads file and folder content from an Amazon Simple Storage Service (S3) bucket.</td>
</tr>
<tr>
<td>Amazon S3 Upload (p. 57)</td>
<td>Uploads file and folder content to an Amazon Simple Storage Service (S3) bucket.</td>
</tr>
<tr>
<td>AWS Send SNS or SQS Message (p. 64)</td>
<td>Sends a message to an Amazon Simple Notification Service (SNS) topic or to an Amazon Simple Queue Service (SQS) queue.</td>
</tr>
<tr>
<td>AWS Secrets Manager Create/Update Secret (p. 60)</td>
<td>Updates a secret, optionally creating a secret if it does not exist.</td>
</tr>
<tr>
<td>AWS Secrets Manager Get Secret (p. 63)</td>
<td>Stores the value of a secret in AWS Secrets Manager into a secret build variable.</td>
</tr>
<tr>
<td>AWS SSM Get Parameter (p. 66)</td>
<td>Reads one or more values from Systems Manager Parameter Store into build variables.</td>
</tr>
<tr>
<td>AWS SSM Set Parameter (p. 68)</td>
<td>Creates or updates a parameter in Systems Manager Parameter Store.</td>
</tr>
<tr>
<td>AWS SSM Run Command (p. 70)</td>
<td>Runs a Systems Manager or user-provided Command on a fleet of EC2 instances.</td>
</tr>
</tbody>
</table>

---

## What's in This Guide


### Getting Started (p. 5)

How to set up an AWS account and install the AWS Toolkit for Azure DevOps. Also how to set up AWS credentials for use in the tasks, which can be accomplished using service endpoints, environment variables, or Amazon EC2 instance metadata (for build agents running on Amazon EC2 instances).

### Using the AWS Toolkit for Azure DevOps (p. 8)

Walk-through topics demonstrating how to use tasks in the AWS Toolkit for Azure DevOps in your build and release definitions.

### Task Reference (p. 21)

Describes the tasks included in the AWS Toolkit for Azure DevOps.
Setting up the AWS Toolkit for Azure DevOps

To use the AWS Toolkit for Azure DevOps to access AWS, you need an AWS account and AWS credentials. When build agents run the tasks contained in the tools, the tasks must be configured with, or have access to, those AWS credentials to enable them to call AWS service APIs. To increase the security of your AWS account, we recommend that you do not use your root account credentials. You should create an IAM user to provide access credentials to the tasks running in the build agent processes.

Topics
- Sign up for AWS (p. 3)
- Create an IAM user (p. 3)

Sign up for AWS

If you do not have an AWS account, complete the following steps to create one.

To sign up for an AWS account
2. Follow the online instructions.
   Part of the sign-up procedure involves receiving a phone call and entering a verification code on the phone keypad.

Create an IAM user

To create an administrator user for yourself and add the user to an administrators group (console)
1. Sign in to the IAM console as the account owner by choosing Root user and entering your AWS account email address. On the next page, enter your password.
   Note
   We strongly recommend that you adhere to the best practice of using the Administrator IAM user that follows and securely lock away the root user credentials. Sign in as the root user only to perform a few account and service management tasks.
2. In the navigation pane, choose Users and then choose Add users.
3. For User name, enter Administrator.
4. Select the check box next to AWS Management Console access. Then select Custom password, and then enter your new password in the text box.
5. (Optional) By default, AWS requires the new user to create a new password when first signing in. You can clear the check box next to User must create a new password at next sign-in to allow the new user to reset their password after they sign in.
6. Choose Next: Permissions.
7. Under Set permissions, choose Add user to group.
8. Choose **Create group**.
9. In the **Create group** dialog box, for **Group name** enter **Administrators**.
10. Choose **Filter policies**, and then select **AWS managed - job function** to filter the table contents.
11. In the policy list, select the check box for **AdministratorAccess**. Then choose **Create group**.
   **Note**
   You must activate IAM user and role access to Billing before you can use the 
   AdministratorAccess permissions to access the AWS Billing and Cost Management 
   console. To do this, follow the instructions in step 1 of the tutorial about delegating access 
   to the billing console.
12. Back in the list of groups, select the check box for your new group. Choose **Refresh** if necessary to 
   see the group in the list.
13. Choose **Next: Tags**.
14. (Optional) Add metadata to the user by attaching tags as key-value pairs. For more information 
   about using tags in IAM, see **Tagging IAM entities** in the **IAM User Guide**.
15. Choose **Next: Review** to see the list of group memberships to be added to the new user. When you 
   are ready to proceed, choose **Create user**.

You can use this same process to create more groups and users and to give your users access to your AWS 
account resources. To learn about using policies that restrict user permissions to specific AWS resources, 
see **Access management** and **Example policies**.

**Create an IAM user and download its credentials**

After you've created an IAM user, copy its credentials. To use the AWS Toolkit for Azure DevOps, you 
must have a set of valid AWS credentials, which consist of an access key and a secret key. These keys are 
used to sign programmatic web service requests and enable AWS to verify that the request comes from 
an authorized source.

**Warning**
Do not copy your root account credentials for use with AWS Toolkit for Azure DevOps.
Getting started

This section provides information about how to install, set up, and use the AWS Toolkit for Microsoft Azure DevOps.

Topics
• Set up an Azure DevOps account (p. 5)
• Install the AWS Toolkit for Azure DevOps extension (p. 5)
• Establish AWS credentials for the AWS Toolkit for Azure DevOps (p. 5)
• Supply task credentials (p. 5)

Set up an Azure DevOps account

To use Azure DevOps, you will first need to sign up for an Azure DevOps account.

Install the AWS Toolkit for Azure DevOps extension

This procedure outlines how to install the AWS Toolkit for Azure DevOps extension.

2. Choose Get it free and sign in to your Azure DevOps account, if prompted.
3. Choose Install to install the toolkit into your Azure DevOps account, or choose Download to install it on an on-premises server.

Establish AWS credentials for the AWS Toolkit for Azure DevOps

To allow the AWS Toolkit for Azure DevOps to access AWS services, you need an AWS account and AWS credentials. When build agents run the tasks contained in the tools, the tasks must be configured with, or have access to, those AWS credentials to enable them to call AWS service APIs.

To increase the security of your AWS account, we recommend that you don't use your root account credentials. You should create an IAM user to provide access credentials to the tasks running in the build agent processes.

For more information about creating an IAM user in your account, and copying that user's credentials, see Setting up the AWS Toolkit for Azure DevOps (p. 3).

Supply task credentials

After you create an AWS account and IAM user, and you've made a copy of the access key and secret access key for that user, you can supply credentials to the tasks in the following ways.
Supply task credentials using a service connection

You can create a link to your AWS subscription by using the Service connections section of the Project settings for your project. Note that a service connection expects long-lived AWS credentials consisting of an access-key and secret-key pair. You can also define Assume Role credentials to scope down the access. Service connections also support the use of a session token variable. You can rotate session tokens from the service connections. For more information, see Use personal access tokens in the Microsoft Azure DevOps online documentation.

To set up a service connection

1. Open Azure DevOps and access the project that you want to add a service connection to.
2. Choose the settings icon in the lower-left side of the screen, and then choose Service connections.
3. From New AWS service connection, choose AWS. This opens the Add AWS service connection form.
4. Provide a Connection name, Access key ID, and Secret key ID, and complete any other fields you want.
5. When you've completed the required and any optional fields in the form, choose OK.

You can test your credentials by creating a new AWS Toolkit for Azure DevOps task in an existing build pipeline and using the connection name you defined in the Add AWS service connection form.

Supply credentials through named variables in your build

You can specify credentials by using named variables. You can set these variables using values from previous jobs in the pipeline, or set them globally. Named variables can be used to get credentials from a custom credentials store.

The following are all the supported named variables:

- `AWS.AccessKeyId` – IAM access key ID.
- `AWS.SecretAccessKey` – IAM secret access key.
- `AWS.SessionToken` – IAM session token.
- `AWS.Region` – AWS Region code, for example, us-east-2.

To set up global pipeline variables

1. Open Azure DevOps, open the build definition, and then choose variables.
2. Choose Add new Variable.
3. Choose a variable name from one of the four supported names listed previously, and then choose the appropriate value based on your use case.
4. Once you save your changes, this variable will be used by all of your AWS tasks.
To set up dynamic pipeline variables

1. Create a job to get the variables.
2. Create a second job that uses your AWS credentials.
3. Give the first job an output variable that contains the credentials.
4. Make the second job rely on the first job.

For more information about Azure DevOps pipeline variables, see Define variables in the Microsoft Azure DevOps online documentation.

Supply standard AWS environment variables in the build agent process

You can specify credentials with standard named AWS environment variables. These variables can be used to get credentials from a custom credentials store.

The following are all the supported standard named AWS environment variables:

- AWS_ACCESS_KEY_ID – IAM access key ID.
- AWS_SECRET_ACCESS_KEY – IAM secret access key.
- AWS_SESSION_TOKEN – IAM session token.
- AWS_ROLE_ARN – Amazon Resource Name (ARN) of the role you want to assume.
- AWS_REGION – AWS Region code, for example, us-east-2.

For more information about Azure DevOps pipeline variables, see Define variables in the Microsoft Azure DevOps online documentation.

Supply credentials with Amazon EC2 build agents

For build agents running on Amazon Elastic Compute Cloud (Amazon EC2) instances, the tasks can automatically obtain credential and Region information from instance metadata associated with the Amazon EC2 instance.

To use Amazon EC2 instance metadata credentials, the instance must have started with an instance profile that references a role that grants permissions to the task. This allows the role to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

Set up an Amazon EC2 instance as a self-hosted Azure pipelines agent. For more information, see Azure Pipelines agent in the Microsoft Azure DevOps online documentation. After that's completed, AWS tasks can be added without setting any credentials explicitly. When running on a build machine, your IAM credentials are picked up automatically.
Using the AWS Toolkit for Azure DevOps

The following tutorials demonstrate how to use tasks from the AWS Toolkit for Microsoft Azure DevOps in your Azure DevOps projects.

Prerequisites

- Either an Azure DevOps account or on-premises Azure DevOps server.
- An AWS account and preferably an associated IAM user account.
- Task-specific permissions.
- An Azure DevOps project with the build definition template specified in each tutorial.

See Getting started (p. 5) for instructions on how to install the AWS Toolkit for Azure DevOps and set up your credentials.

Topics

- Archiving build artifacts to AWS (p. 8)
- Deploying an ASP.NET web app to AWS (p. 14)

Archiving build artifacts to AWS

The following tutorial demonstrates how to use the AWS S3 Upload task to upload archival data to an Amazon Simple Storage Service (Amazon S3) bucket from an Azure DevOps build definition.

Prerequisites

- The AWS Toolkit for Azure DevOps installed in Azure DevOps or an on-premises Azure DevOps server.
- An AWS account and preferably an associated IAM user account.
- An existing S3 bucket or a unique S3 bucket name to use during this procedure.
- A code project for an ASP.NET Core Web Application, which you will push to your Azure DevOps project.

Archiving build artifacts with the AWS S3 Upload task

Create a new Azure DevOps project and add a new pipeline to the project based on the ASP.NET Core template. To follow along with the screenshots shown below, use the classic editor (that is, without YAML).
The build process page for this pipeline contains the following default tasks.
Add the S3 Upload task to the build definition

To capture the build output produced by the Publish task and upload it to Amazon S3 you need to add the Amazon S3 Upload task between the existing Publish and Publish Artifact tasks.

Select the "+" icon at the top of the task list. In the right hand panel, optionally enter something in the search box, for example "Amazon", and scroll through the available tasks until you see the Amazon S3 Upload task. Select the Add button to add it to the build definition.
If the new task was not added immediately after the *Publish* task, drag it into that position.

Click on the new task to see its properties in the right pane.
Configure the task properties

- **AWS Credentials**
  
  If you have already configured AWS credentials for your project, you can select them from the dropdown list. If not, you can add credentials for this task by choosing the **New** button next to the **AWS Credentials** field. For information about filling out the resulting **Add AWS service connection** form, see the topic on the section called “Supply task credentials using a service connection” (p. 6).

This task requires credentials for a user with a policy enabling the user to put objects to S3. If the **Create S3 bucket** option is enabled (see the following) the user also needs permission to create a bucket.

  **Note**
  
  We recommend that you do not use your account’s root credentials. Instead, create one or more IAM users, and then use those credentials. For more information, see Best practices for managing AWS access keys in the Amazon Web Services General Reference.

- **AWS Region**
  
  Set the AWS Region in which the bucket exists or will be created; for example, us-east-1, us-west-2, and so on.

- **Bucket Name**
  
  Enter the name of the bucket. Bucket names must be globally unique.

- **Source Folder**
  
  This field points to a folder in your build area that contains the content to be uploaded. For this tutorial, use the variable "$(Build.ArtifactStagingDirectory)" (without the quotation marks). This is the same variable that is specified by default in the **Publish** task (the --output argument), as well as in other tasks.

  **Note**
  
  Azure DevOps provides a number of variables that you can use to avoid hard-coded paths.

- **Filename Patterns**
  
  This field can contain one or more globbing patterns used to select files under the **Source Folder** for upload. The default value "***" selects all files recursively. Multiple patterns can be specified, one per line. For this tutorial, the **Publish** task, which precedes the **S3 Upload** task, emits a .zip file that contains the build. This is the file that will be uploaded.

- **Target Folder**
  
  This is the **key prefix** in the bucket that will be applied to all of the uploaded files. You can think of this as a folder path. If no value is given, the files are uploaded to the root of the bucket. Note that by default the relative folder hierarchy is preserved.

- **Create S3 bucket if it does not exist**
  
  Select this check box if the target bucket doesn’t exist. The task will fail if the bucket cannot be created for some reason (for example, not a unique name, lack of permissions).

- **Overwrite** (in the **Advanced** section)
  
  Changing this check box has no effect. If a file with the same name already exists in the Amazon S3 bucket, it will always be overwritten.

- **Flatten folders** (in the **Advanced** section)
  
  Select this check box if you want to flatten the folder structure. All files will be placed into the specified target folder in the bucket, removing their relative paths to the source folder.
Run the build

With the new task configured, you are ready to run the build. Choose **Save & queue**.

During the build you can view the log by clicking the build number in the queue message.

When the build has completed, you will be able to see S3 upload logs similar to the following.
Deploying an ASP.NET web app to AWS

The following tutorial demonstrates how to use the *AWS Elastic Beanstalk Deploy Application* task to deploy a web application to the AWS Cloud from an Azure DevOps build definition.

**Prerequisites**

- The AWS Toolkit for Azure DevOps installed in Azure DevOps or on an on-premises Azure DevOps server.
- An AWS account and preferably an associated IAM user account.
- An AWS Elastic Beanstalk application and environment.
- A code project for an *ASP.NET Web Application (.NET Framework)* or an *ASP.NET Core Web Application*, which you will push to your Azure DevOps project.

**Deploying an ASP.NET application using the AWS Elastic Beanstalk Deploy Application task**

Create a new Azure DevOps project and upload your *ASP.NET Web Application (.NET Framework)* or *ASP.NET Core Web Application* files to it.

Then, add a new pipeline to the project based on the *ASP.NET Core (.NET Framework)* template, which will produce a Web Deploy archive for deployment. To follow the screenshots shown, use the classic editor (that is, without YAML).
The build process page for this pipeline contains the following default tasks.
Add the AWS Elastic Beanstalk Deploy Application task to the build definition

Select the "+" icon at the top of the task list. Optionally, in the panel on the right, enter something in the search box, for example "AWS", and scroll through the available tasks until you see the AWS Elastic Beanstalk Deploy Application task. Select Add to add it to the bottom of the build definition.
Click the new task to see its properties in the right pane.
Configure the task properties

- **AWS Credentials**

  If you have already configured AWS credentials for your project, you can select them from the drop-down list. If not, you can add credentials for this task by choosing New next to the AWS Credentials field. For information about filling out the resulting Add AWS service connection form, see the section called “Supply task credentials using a service connection” (p. 6).

  This task requires credentials for a user with a policy enabling the user to update an Elastic Beanstalk environment and describe an environment’s status and events.

  **Note**
  
  We recommend that you do not use your account’s root credentials. Instead, create one or more IAM users, and then use those credentials. For more information, see Best practices for managing AWS access keys in the Amazon Web Services General Reference.

- **AWS Region**

  Set the AWS Region in which the Elastic Beanstalk environment is running (for example, us-east-1, us-west-2).

- **Application Name**

  The name you used to create the Elastic Beanstalk application. An Elastic Beanstalk application is the container for the environment for the .NET web application.

- **Environment Name**
The name of the Elastic Beanstalk environment that is associated with the Application Name. An Elastic Beanstalk environment contains the actual provisioned resources that are running the .NET web application.

- **Deployment Bundle Type**

  Leave this field set to the default: **ASP.NET (Source: Web Deploy Archive).**

- **Web Deploy Archive**

  The full path to the Web Deploy archive, including the file name of the archive file, which has a .zip extension. The value for this field can be found in the Build Solution task, the DesktopBuildPackageLocation argument in the MSBuild Arguments field. The directory part of the full path is also used in other tasks.

  If, for example, the project was set up according to the instructions here, the value for this field will be 
  
  "$(build.artifactstagingdirectory)\WebApp.zip" (without the quotation marks).

  **Note**

  Azure DevOps provides a number of variables that you can use to avoid hard-coded paths.

- **Event poll delay (seconds)**

  The time, in seconds, to wait between calls to retrieve the latest events from the deployment to the environment. The default is 5 seconds. For large deployments or slow connections, you might need to increase this value.

**Run the build**

With the new task configured, you are ready to run the build. Choose **Save & queue**.
When the build has completed running, you should see a log similar to the following.

```
#(section)Starting: Deploy to Elastic Beanstalk: eb-deploy-walkthrough

Task : AWS Elastic Beanstalk Deploy Application
Description : Deploys an application to Amazon EC2 Instance(s) using AWS Elastic Beanstalk
Version : 1.1.4

(etc.)
```

```
48 Started updating environment to version: v5ebp3s5s340407
49 Waiting for deployment to complete
50 Events from Elastic Beanstalk:
51 Set Jun 15 2019 02:21:53 GMT=0000 (Coordinated Universal Time) INFO Environment update is starting.
52 Set Jun 15 2019 02:21:57 GMT=0000 (Coordinated Universal Time) INFO Deploying new version to instance(s).
53 Set Jun 15 2019 02:22:16 GMT=0000 (Coordinated Universal Time) INFO Environment health has transitioned from OK to LAM.
54 Set Jun 15 2019 02:22:17 GMT=0000 (Coordinated Universal Time) INFO Started Application Update
55 Set Jun 15 2019 02:22:18 GMT=0000 (Coordinated Universal Time) INFO UpdatedAppVersion Completed
56 Set Jun 15 2019 02:22:32 GMT=0000 (Coordinated Universal Time) INFO New application version was deployed to running instance(s).
58 Deployment to application eb-deploy-walkthrough completed
```

```
#(section)Finishing: Deploy to Elastic Beanstalk: eb-deploy-walkthrough
```
Task reference

This reference describes the tasks that are included in the AWS Toolkit for Microsoft Azure DevOps.

Prerequisites

You must have an AWS account. For information on setting up an account, see Setting up the AWS Toolkit for Azure DevOps (p. 3).

Each task requires that AWS credentials for your account be available to the build agent running your task. Each task also requires the Region in which the API calls to AWS services will be made.

You can do one of the following:

- Specify credentials explicitly for each task. Do this by configuring a named service endpoint (of endpoint type AWS) and then referring to the endpoint name in the AWS Credentials field for each task. For information about this method, see the topic on the section called “Supply task credentials using a service connection” (p. 6).

In this case, the AWS Region can be set in the AWS Region property of a task.

- Supply credentials and Region to tasks using environment variables in the process hosting the build agent.

- If your build agent is running on an Amazon EC2 instance you can also elect to have credentials (and Region) be obtained automatically from the instance metadata associated with the instance. For credentials to be available from EC2 instance metadata the instance must have been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

Note: If you choose to use an AWS service endpoint to supply credentials to tasks, we strongly recommend using an AWS Identity and Access Management user account, with appropriate permissions to scope the privileges of the user account to only those needed to execute the tasks you need.

Topics

- AWS CLI task (p. 22)
- AWS Tools for Windows PowerShell Script task (p. 23)
- AWS Shell Script task (p. 25)
- AWS CloudFormation Create/Update Stack task (p. 27)
- AWS CloudFormation Delete Stack task (p. 33)
- AWS CloudFormation Execute Change Set task (p. 34)
- AWS CodeDeploy Application Deployment task (p. 36)
- Amazon ECR Push task (p. 39)
- AWS Elastic Beanstalk Create Version task (p. 41)
- AWS Elastic Beanstalk Deploy Application task (p. 43)
- AWS Lambda Deploy Function task (p. 46)
- AWS Lambda Invoke Function task (p. 50)
- AWS Lambda .NET Core task (p. 51)
- Amazon S3 Download task (p. 54)
- Amazon S3 Upload task (p. 57)
AWS CLI task

Synopsis

Runs a command using the AWS CLI. Note that you must have the AWS CLI installed to use this task. For more information, see Installing the AWS Command Line Interface.

Description

The AWS CLI uses a multipart structure on the command line. It starts with the base call to AWS. The next part specifies a top-level command, which often represents an AWS service that the AWS CLI supports. Each AWS service has additional subcommands that specify the operation to perform. You can specify the general AWS CLI options, or the specific parameters for an operation, in any order on the command line. If you specify an exclusive parameter multiple times, only the last value applies.

<command> <subcommand> [options and parameters]

Parameters can take various types of input values such as numbers, strings, lists, maps, and JSON structures.

Parameters

You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

Display name*

The default name of the task instance, which can be modified: AWS CLI

AWS Credentials

Specifies the AWS credentials to be used by the task in the build agent environment.

You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:

- From task variables named `AWS.AccessKeyId`, `AWS.SecretAccessKey` and optionally `AWS.SessionToken`.
- From credentials set in environment variables in the build agent process. When using environment variables in the build agent process you may use the standard AWS environment variables: `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` and optionally `AWS_SESSION_TOKEN`.
- If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have...
been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

**AWS Region**

The AWS Region code (for example, us-east-1, us-west-2) of the Region containing the AWS resources the task will use or create. For more information, see Regions and endpoints in the Amazon Web Services General Reference.

If a Region is not specified in the task configuration the task will attempt to obtain the Region to be used using the standard AWS environment variable AWS_REGION in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

**Note**
The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed may still be used by entering the region code of the Region (for example, us_west_2).

**Command***
The AWS CLI command to run. Run aws help in the AWS Command Line Interface to get a complete list of commands, or see CommandStructure in the AWS Command Line Interface.

**Subcommand**
The AWS CLI subcommand to run. Run aws help in the AWS Command Line Interface to get a complete list of commands, or see CommandStructure in the AWS Command Line Interface.

**Options and Parameters**
The arguments to pass to the AWS CLI command. Run aws <command> --help in the AWS Command Line Interface to get the complete list of arguments supported by the command.

**Advanced**

**Fail on Standard Error**
If true, this task fails if any errors are written to the StandardError stream.

**Task Permissions**
Permissions for this task to call AWS service APIs depend on the configured command.

**AWS Tools for Windows PowerShell Script task**

**Synopsis**
Runs a PowerShell script that uses cmdlets from the AWS Tools for Windows PowerShell module. The module is automatically installed if it isn't already available in the environment.
Description

This task accepts a PowerShell command or script that uses cmdlets from the Tools for Windows PowerShell module to interact with AWS services. You can specify the script to run via its file name, or you can enter it into the task configuration. Before running the supplied script, the task tests to see if the required Tools for Windows PowerShell module is already installed. If it isn't installed, the latest available version from the PowerShell Gallery is downloaded and installed.

Note

If an installation is performed, the module is installed in the current user scope. The location is compatible with automatic module load. As a result, you don't need to import the module in your script.

Parameters

You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

Display name*

The default name of the task instance, which can be modified: AWS Tools for Windows PowerShell Script

AWS Credentials

 Specifies the AWS credentials to be used by the task in the build agent environment.

You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:

- From task variables named AWS.AccessKeyId, AWS.SecretAccessKey and optionally AWS.SessionToken.
- From credentials set in environment variables in the build agent process. When using environment variables in the build agent process you may use the standard AWS environment variables: AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY and optionally AWS_SESSION_TOKEN.
- If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

AWS Region

The AWS Region code (for example, us-east-1, us-west-2) of the Region containing the AWS resources the task will use or create. For more information, see Regions and endpoints in the Amazon Web Services General Reference.

If a Region is not specified in the task configuration the task will attempt to obtain the Region to be used using the standard AWS environment variable AWS_REGION in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

Note: The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed may still be used by entering the region code of the Region (for example, us_west_2).
Arguments

Optional arguments to pass to the script. You can use ordinal or named parameters.

Script Source*

The type of script to run. Choose Script File to run a script that is contained in a file. Choose Inline Script to enter the script to run in the task configuration.

Script Path*

Required if the Script Source parameter is set to Script File. Specify the full path to the script you want to run.

Inline Script*

Required if the Script Source parameter is set to Inline Script. Enter the text of the script to run.

ErrorActionPreference

Prepends the line $ErrorActionPreference = 'VALUE' at the top of your script.

Advanced

Fail on Standard Error

If this option is selected, the task will fail if any errors are written to the error pipeline, or if any data is written to the Standard Error stream. Otherwise, the task relies on the exit code to determine failure.

Ignore $LASTEXITCODE

If this option is not selected, the line if ((Test-Path -LiteralPath variable:LASTEXITCODE)) { exit $LASTEXITCODE } is appended to the end of your script. This causes the last exit code from an external command to propagate as the exit code of PowerShell. Otherwise, the line is not appended to the end of your script.

Working Directory

The working directory where the script runs.

Task Permissions

Permissions for this task to call AWS service APIs depend on the activities in the supplied script.

AWS Shell Script task

Synopsis

Run a shell script using Bash with AWS credentials.
Description

Runs a shell script in Bash, setting AWS credentials and Region information into the shell environment using the standard environment keys `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY`, `AWS_SESSION_TOKEN` and `AWS_REGION`.

Parameters

You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

Display name*

The default name of the task instance, which can be modified: AWS Shell Script

AWS Credentials

Specifies the AWS credentials to be used by the task in the build agent environment.

You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:

- From task variables named `AWS.AccessKeyID`, `AWS.SecretAccessKey` and optionally `AWS.SessionToken`.
- From credentials set in environment variables in the build agent process. When using environment variables in the build agent process you may use the standard AWS environment variables: `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` and optionally `AWS_SESSION_TOKEN`.
- If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

AWS Region

The AWS Region code (us-east-1, us-west-2 etc.) of the Region containing the AWS resources the task will use or create. For more information, see Regions and endpoints in the Amazon Web Services General Reference.

If a Region is not specified in the task configuration, the task will attempt to obtain the Region to be used using the standard AWS environment variable `AWS_REGION` in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

Note: The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed can still be used by entering the Region code of the Region (for example, `us_west_2`).

Arguments

The arguments to be passed to the shell script.
Task Permissions

Script Source

The source of the script to run in the shell. Choose Script file to enter the file path to the script to be run or Inline script to specify the source code for the script in the task configuration.

Script Path

When Script Source is set to Script file, specifies the file path to the script to execute. This must be a fully qualified path or a path relative to the $(System.DefaultWorkingDirectory) location. The script file must exist.

Inline Script

The source code of the script to run when Script Source is set to Inline script. A maximum of 5000 characters is allowed.

Specify Working Directory

If selected a custom working directory, which must exist, can be specified for the script. The default behavior when unchecked is to set the working directory for the shell to be the script file location.

Working Directory

If Specify Working Directory is checked, contains the custom working directory for the script.

Fail on Standard Error

If this option is selected, the task will fail if any errors are written to the standard error stream.

Task Permissions

Permissions for this task to call AWS service APIs depend on the activities in the supplied script.

AWS CloudFormation Create/Update Stack task

Synopsis

Creates a new AWS CloudFormation stack or updates the stack if it exists.

Description

Creates or updates a stack based on the specified parameters. When you need to change a stack’s settings or its resources, update the stack instead of deleting it and creating a new stack.

Parameters

You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

Display name*

The default name of the task instance, which can be modified: Create/Update Stack
AWS Toolkit for Microsoft Azure DevOps User Guide
Parameters

AWS Credentials

Specifies the AWS credentials to be used by the task in the build agent environment.

You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:

- From task variables named `AWS.AccessKeyId`, `AWS.SecretAccessKey` and optionally `AWS.SessionToken`.
- From credentials set in environment variables in the build agent process. When using environment variables in the build agent process you may use the standard AWS environment variables: `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` and optionally `AWS_SESSION_TOKEN`.
- If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

AWS Region

The AWS Region code (for example, us-east-1, us-west-2) of the Region containing the AWS resources the task will use or create. For more information, see Regions and endpoints in the Amazon Web Services General Reference.

If a Region is not specified in the task configuration the task will attempt to obtain the Region to be used using the standard AWS environment variable `AWS_REGION` in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

Note: The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed may still be used by entering the region code of the Region (for example, `us_west_2`).

Stack Name*

The name associated with the stack. The name must be unique in the region in which you are creating the stack.

A stack name can contain only alphanumeric characters (case-sensitive) and hyphens. It must start with an alphabetic character and cannot be longer than 128 characters.

Template Source*

Specifies the location of the template to use to create or update the stack. You can specify the template using the path to a file in the local file system, a URL to the file, or an object in Amazon S3. If you select an object in Amazon S3, you can specify the bucket and object name (key).

Note that CloudFormation limits the size of template files uploaded to the service to 51,200 bytes. If your template is larger than the allowed size you should choose either the URL or Amazon S3 location options. You can also specify a bucket name for the local file option. If a bucket name is specified, the template is uploaded to the bucket by the task. The object key will be the template filename, less any path.

When the task uploads the template to a bucket or you specify an Amazon S3 bucket name and object key, the task generates a URL to the object and supplies the URL to CloudFormation.
**Template File***

The path to the template file for the stack. For more information, see Template Anatomy in the AWS CloudFormation User Guide.

**S3 Bucket**

The name of the bucket to which a local template file can be uploaded, or which contains the template to be used. If Template Source is set to Amazon S3 bucket and object key this parameter is required.

For more information, see Template Anatomy in the AWS CloudFormation User Guide.

**S3 Object Key**

The name of the template file in the S3 bucket. The task will generate a URL to the file when specifying the location of the template file to CloudFormation. If Template Source is set to Amazon S3 bucket and object key this parameter is required.

For more information, see Template Anatomy in the AWS CloudFormation User Guide.

**Template URL**

URL reference to the template file in Amazon S3. This field is required if Template Source is set to URL to the template file. When stored in Amazon S3 template files are subject to a maximum size of 460,800 bytes.

For more information, see Template Anatomy in the AWS CloudFormation User Guide.

**Template Parameters Source**

Specifies the source of parameter values to supply with the template. If your template uses parameters you can supply their values in JSON or YAML content, from a file in the build area or inline in the task.

If your template does not need parameter value to be supplied leave the 'Local file' option field empty. Note that a value for parameters must be specified if if the field is set to Inline.

**Template Parameters File**

Optional path to an existing file containing the template parameters in JSON or YAML format. If your template does not require parameters leave the field empty.

CloudFormation expects the file to contain an array of one or more parameter objects. Each object specifies the name of the parameter as ParameterKey and the corresponding value in ParameterValue, for example (in JSON format):

```
[  
    {  
        "ParameterKey": "parameter1", "ParameterValue": "parameter1value"  
    },  
    {  
        "ParameterKey": "parameter2", "ParameterValue": "parameter2value"  
    }  
]
```

For more information, see Template Anatomy in the AWS CloudFormation User Guide.
Template Parameters

Parameter values for the template in JSON or YAML format when Template Parameters. A value must be provided if *Template Parameters Source is set to Inline.

CloudFormation expects the file to contain an array of one or more parameter objects. Each object specifies the name of the parameter as ParameterKey and the corresponding value in ParameterValue, for example (in JSON format):

```
[
  {
    "ParameterKey": "parameter1", "ParameterValue": "parameter1value"
  }, {
    "ParameterKey": "parameter2", "ParameterValue": "parameter2value"
  }
]
```

For more information, see Template Anatomy in the AWS CloudFormation User Guide.

Create or Update the Stack Using a Change Set

If selected a change set will be created that contains a list of changes that will be applied to a stack and then validated. If the changes validate successfully the change set can then be executed to effect the changes. You can elect to use a change set to both create a new stack or update an existing stack.

**Note:** when using this task to deploy a serverless application template you must select to use a change set.

**Change Set Name**

The name of the change set. The name must be unique among all change sets that are associated with the specified stack.

A change set name can contain only alphanumeric, case sensitive characters and hyphens. It must start with an alphabetic character and cannot exceed 128 characters. This parameter is required if the option to use a change set is selected.

**Description**

A description to help you identify this change set. Max length 1024 characters.

**Automatically Execute the Change Set**

If checked, the change set is automatically executed when validation succeeds. If it isn't checked the change set is validated but not executed. You can execute the change set later by using the AWS CloudFormation Execute Change Set task.

**Capabilities**

Capabilities that must be specified before AWS CloudFormation can update certain stacks. Some stack templates might include resources that can affect permissions in your AWS account, for example, by creating new AWS Identity and Access Management (IAM) users. For those stacks, you must explicitly acknowledge their capabilities by specifying this parameter.
If your stack manipulates IAM resources, you can specify either capability otherwise an InsufficientCapabilities error will be returned.

**Create or Update IAM Resources (‘CAPABILITY_IAM’)**

If your stack manipulates IAM resources, you can specify either capability. Otherwise, an InsufficientCapabilities error is returned.

**Create or Update Named IAM Resources (‘CAPABILITY_NAMED_IAM’)**

If your stack manipulates IAM resources with custom names, you must add this capability otherwise an InsufficientCapabilities error is returned.

**Advanced**

**Role ARN**

The Amazon Resource Name (ARN) of an IAM role that AWS CloudFormation assumes when executing the change set. AWS CloudFormation uses the role’s credentials to make calls on your behalf. AWS CloudFormation uses this role for all future operations on the stack. As long as users have permission to operate on the stack, AWS CloudFormation uses this role even if the users don't have permission to pass it. Ensure that the role grants least privilege. If you don’t specify a value, AWS CloudFormation uses the role that was previously associated with the stack. If no role is available, AWS CloudFormation uses a temporary session that is generated from your user credentials.

Advanced

**Role ARN**

The Amazon Resource Name (ARN) of an IAM role that AWS CloudFormation assumes when executing the change set. AWS CloudFormation uses the role’s credentials to make calls on your behalf. AWS CloudFormation uses this role for all future operations on the stack. As long as users have permission to operate on the stack, AWS CloudFormation uses this role even if the users don't have permission to pass it. Ensure that the role grants least privilege. If you don’t specify a value, AWS CloudFormation uses the role that was previously associated with the stack. If no role is available, AWS CloudFormation uses a temporary session that is generated from your user credentials.

If you don’t specify a value, AWS CloudFormation uses the role that was previously associated with the stack. If no role is available, AWS CloudFormation uses a temporary session that is generated from your user credentials.

**Resource Types**

The template resource types that you have permissions to work with if you execute this change set. For example, `AWS::EC2::Instance`, `AWS::EC2::*`, or `Custom::MyCustomInstance`.

If the list of resource types doesn’t include a resource type that you’re updating, the stack update fails. By default, AWS CloudFormation grants permissions to all resource types. IAM uses this parameter for condition keys in IAM policies for AWS CloudFormation.

For more information, see Controlling Access with AWS Identity and Access Management in the AWS CloudFormation User Guide.

**Notification ARNs**

One or more Amazon Resource Name (ARN) of Amazon SNS topics that AWS CloudFormation associates with the stack. To remove all associated notification topics, specify an empty list.

**Tags**

Collection of tags to apply to the resources created by your template. Tags can be specified as `tagkey=tagvalue`, one per line.

**Rollback Triggers**

Rollback triggers enable you to have AWS CloudFormation monitor the state of your application during stack creation and updating, and to rollback that operation if the application breaches the threshold of any of the alarms you’ve specified. Learn more.
**Trigger Monitoring Time**

The amount of time, in minutes, during which AWS CloudFormation should monitor all the rollback triggers after the stack creation or update operation deploys all necessary resources.

If you specify a monitoring period but do not specify any rollback triggers, AWS CloudFormation still waits the specified period of time before cleaning up old resources after update operations. You can use this monitoring period to perform any manual stack validation desired, and manually cancel the stack creation or update (using `CancelUpdateStack`, for example) as necessary.

If you specify 0 for this parameter, AWS CloudFormation still monitors the specified rollback triggers during stack creation and update operations. Then, for update operations, it begins disposing of old resources immediately once the operation completes.

**Rollback Trigger ARNs**

The Amazon Resource Names (ARNs) of the triggers to monitor during stack creation or update actions.

By default AWS CloudFormation saves the rollback triggers specified for a stack and applies them to any subsequent update operations for the stack, unless you specify otherwise. If you do specify rollback triggers for this parameter, those triggers replace any list of triggers previously specified for the stack. This means:

- To use the rollback triggers previously specified for this stack, if any, don't specify this parameter.
- To specify new or updated rollback triggers, you must specify all the triggers that you want used for this stack, even triggers you've specified before (for example, when creating the stack or during a previous stack update). Any triggers that you don't include in the updated list of triggers are no longer applied to the stack.

If a specified trigger is missing, the entire stack operation fails and is rolled back.

A maximum of five triggers can be supplied.

**Options**

**On Failure**

Determines what action to take if stack creation fails. The default is to roll back.

**Disable Rollback**

If checked, disables rollback of the stack if stack creation failed. You can specify `DisableRollback` or `OnFailure`, but not both.

**Log warning during stack update if AWS CloudFormation reports no work to be done**

If selected and an update-stack operation, with or without a change set, results in no changes being reported by the service, then a warning message is emitted into the task logs. If not selected, the message from the service is ignored and no warning is emitted.

**Output Variable**

The name of the variable that will contain the ID of the stack on task completion. You can use `$(variableName)` to refer to the stack ID in subsequent tasks.
Max Timeout

Maximum time, specified in minutes, that the task should wait for the stack creation or update to complete. By default a maximum of 60 minutes is used.

Task Permissions

This task requires permissions to call the following AWS service APIs (depending on selected task options, not all APIs may be used):

- cloudformation:CreateChangeSet
- cloudformation:CreateStack
- cloudformation:DeleteChangeSet
- cloudformation:DescribeChangeSet
- cloudformation:DescribeStacks
- cloudformation:DescribeStackResources
- cloudformation:ExecuteChangeSet
- cloudformation:UpdateStack

The task may also require permissions to upload your application template to the specified Amazon S3 bucket.

AWS CloudFormation Delete Stack task

Synopsis

Deletes an AWS CloudFormation stack.

Description

Deletes the specified AWS CloudFormation stack.

Parameters

You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

Display name*

The default name of the task instance, which can be modified: Delete Stack

AWS Credentials

Specifies the AWS credentials to be used by the task in the build agent environment.

You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:
• From task variables named `AWS.AccessKeyID`, `AWS.SecretAccessKey` and optionally `AWS.SessionToken`.
• From credentials set in environment variables in the build agent process. When using environment variables in the build agent process you may use the standard AWS environment variables: `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` and optionally `AWS_SESSION_TOKEN`.
• If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

AWS Region

The AWS Region code (for example, us-east-1, us-west-2) of the Region containing the AWS resources the task will use or create. For more information, see Regions and endpoints in the Amazon Web Services General Reference.

If a Region is not specified in the task configuration the task will attempt to obtain the Region to be used using the standard AWS environment variable `AWS_REGION` in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

Note: The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed may still be used by entering the region code of the Region (for example, `us_west_2`).

Stack Name*

The name or unique ID of the stack to be deleted.

Task Permissions

This task requires permissions to call the following AWS service APIs (depending on selected task options, not all APIs may be used):

• cloudformation:DeleteStack
• cloudformation:DescribeStacks

AWS CloudFormation Execute Change Set task

Synopsis

Executes an AWS CloudFormation change set to create or update a stack.

Description

When you execute a change set, AWS CloudFormation deletes all other change sets associated with the stack because they aren't valid for the updated stack.

AWS CloudFormation updates a stack using the input information that was provided when the specified change set was created.
If a stack policy is associated with the stack, AWS CloudFormation enforces the policy during the update. You can't specify a temporary stack policy that overrides the current policy.

Parameters

You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

**Display name***

The default name of the task instance, which can be modified: Execute Change Set

**AWS Credentials**

Specifies the AWS credentials to be used by the task in the build agent environment.

You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:

- From task variables named `AWS.AccessKeyId`, `AWS.SecretAccessKey` and optionally `AWS.SessionToken`.
- From credentials set in environment variables in the build agent process. When using environment variables in the build agent process you may use the standard AWS environment variables: `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` and optionally `AWS_SESSION_TOKEN`.
- If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

**AWS Region**

The AWS Region code (for example, us-east-1, us-west-2) of the Region containing the AWS resources the task will use or create. For more information, see Regions and endpoints in the Amazon Web Services General Reference.

If a Region is not specified in the task configuration the task will attempt to obtain the Region to be used using the standard AWS environment variable `AWS_REGION` in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

**Note:** The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed may still be used by entering the region code of the Region (for example, `us_west_2`).

**Change Set Name***

The name or Amazon Resource Name (ARN) of the change set that you want to execute.

**Stack Name**

The stack name or ARN of the stack associated with the change set. This value is required if you specify the name of a change set to execute. If the ARN of the change set ARN is specified this field is optional.
Task Permissions

The name must be unique in the region in which you are creating the stack. A stack name can contain only alphanumeric characters (case-sensitive) and hyphens. It must start with an alphabetic character and cannot be longer than 128 characters.

Output Variable

The name of the variable that will contain the ID of the stack on task completion. The variable can be used as $(variableName) to refer to the stack ID in subsequent tasks.

Task Permissions

This task requires permissions to call the following AWS service APIs (depending on selected task options, not all APIs may be used):

- cloudformation:DescribeStacks
- cloudformation:DescribeChangeSet
- cloudformation:DescribeStackResources
- cloudformation:ExecuteChangeSet

AWS CodeDeploy Application Deployment task

Synopsis

Deploys an application to Amazon EC2 instances by using AWS CodeDeploy.

Description

This can be a variety of application content, such as code, web and configuration files, executable files, packages, scripts, and multimedia files.

Parameters

You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

Display name*

The default name of the task instance, which can be modified: Deploy with CodeDeploy

AWS Credentials

Specifies the AWS credentials to be used by the task in the build agent environment.

You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:

- From task variables named AWS.AccessKeyId, AWS.SecretAccessKey and optionally AWS.SessionToken.
- From credentials set in environment variables in the build agent process. When using environment variables in the build agent process you may use the standard AWS environment variables: AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY and optionally AWS_SESSION_TOKEN.
• If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

AWS Region

The AWS Region code (for example, us-east-1, us-west-2) of the Region containing the AWS resources the task will use or create. For more information, see Regions and endpoints in the Amazon Web Services General Reference.

If a Region is not specified in the task configuration the task will attempt to obtain the Region to be used using the standard AWS environment variable AWS_REGION in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

Note: The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed may still be used by entering the region code of the Region (for example, us_west_2).

Application Name*

The name of the AWS CodeDeploy application.

Deployment Group Name*

The name of the deployment group the revision is to be deployed to.

Deployment Revision Source*

Specifies the source of the revision to be deployed. You can select from:

• Folder or archive file in the workspace: the task will create or use an existing zip archive in the location specified to Revision Bundle, upload the archive to Amazon S3 and supply the key of the S3 object to CodeDeploy as the revision source.

• Archive file in Amazon S3: select to specify the key of an archive previously uploaded to Amazon S3 as the deployment revision source.

Revision Bundle*

The location of the application revision artifacts to deploy. You can supply a filename or folder. If a folder is supplied the task will recursively zip the folder contents into an archive file before uploading the archive to Amazon S3. If a filename is supplied the task uploads it unmodified to Amazon S3. CodeDeploy requires the appspec.yml file describing the application to exist at the root of the specified folder or archive file.

Required if Deployment Revision Source is set to Folder or archive file in the workspace.

S3 Bucket Name*

The name of the Amazon S3 bucket to which the revision bundle is uploaded or can be found, if Archive file in Amazon S3 was selected for Deployment Revision Source.
**Target Folder**

Optional folder (key prefix) for the uploaded revision bundle in the bucket. If not specified the, bundle is uploaded to the root of the bucket.

*Available when* Folder or archive file in the workspace *is selected for Deployment Revision Source.*

**Revision Bundle Key**

The Amazon S3 object key of the previously uploaded archive file containing the deployment revision artifacts.

*Required if* Deployment Revision Source *is set to Archive file in Amazon S3.*

**Description**

Optional description for the deployment.

**Existing File Behavior**

How AWS CodeDeploy should handle files that already exist in a deployment target location but weren't part of the previous successful deployment.

**Advanced**

**Update Outdated Instances Only**

If checked, deploys to only those instances that are not running the latest application revision.

**Ignore Application Stop Failures**

When checked, if the deployment causes the ApplicationStop deployment lifecycle event to an instance to fail, the deployment to that instance is not considered failed at that point. It continues on to the BeforeInstall deployment lifecycle event.

**Max Timeout**

Maximum time, specified in minutes, that the task should wait for the stack creation or update to complete. By default a maximum of 60 minutes is used.

**Output**

**Output Variable**

The name of the variable that will contain the deployment ID on task completion. You can use the variable $(variableName) to refer to the function result in subsequent tasks.

**Task Permissions**

This task requires permissions to call the following AWS service APIs (depending on selected task options, not all APIs may be used):

- codedeploy:GetApplication
- codedeploy:GetDeploymentGroup
- codedeploy:CreateDeployment
- codedeploy:GetDeployment
Depending on selected parameters the task may also require permissions to verify your deployment bundle exists in S3 or upload your application bundle to the specified Amazon S3 bucket. Depending on the size of the application bundle, either PutObject or the S3 multi-part upload APIs may be used.

Amazon ECR Push task

(Amazon Elastic Container Registry Push Image Task)

Synopsis

Pushes a Docker image identified by name, with optional tag, or image ID to the Amazon Elastic Container Registry (ECR).

Description

This task pushes a Docker image to the Elastic Container Registry. The image to push can be identified using its image ID or by name, with optional tag suffix. The task handles the work of appropriately tagging the image as required by ECR and also the login process to your registry prior to executing the Docker Push command.

Parameters

You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

Display name*

The default name of the task instance, which can be modified: Push Image

AWS Credentials

Specifies the AWS credentials to be used by the task in the build agent environment.

You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:

- From task variables named `AWS.AccessKeyId`, `AWS.SecretAccessKey` and optionally `AWS.SessionToken`.
- From credentials set in environment variables in the build agent process. When using environment variables in the build agent process you may use the standard AWS environment variables: `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` and optionally `AWS_SESSION_TOKEN`.
- If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

AWS Region

The AWS Region code (for example, us-east-1, us-west-2) of the Region containing the AWS resources the task will use or create. For more information, see Regions and endpoints in the Amazon Web Services General Reference.
If a Region is not specified in the task configuration the task will attempt to obtain the Region to be used using the standard AWS environment variable \texttt{AWS\_REGION} in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

\textbf{Note:} The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed may still be used by entering the \textit{region code} of the Region (for example, \texttt{us\_west\_2}).

\textbf{Image Identity*}

How the image to be pushed is identified. You can select from either the image ID or the image name. If image name is selected a tag can also be specified.

\textbf{Source Image Name}

The name of the image to push. Required if \textit{Image Identity} is set to \textit{Image name with optional tag}.

\textbf{Source Image Tag}

Optional tag that can be suffixed to the image name. If a tag is not specified, 'latest' is assumed.

\textbf{Source Image ID}

The ID of the image to push. Required if \textit{Image Identity} is set to \textit{Image ID}.

\textbf{Target Repository Name*}

The name of the repository to which the image will be pushed.

\textbf{Target Repository Tag}

Optional tag for the new image in the repository. If not specified, ECR will assume 'latest'.

\textbf{Create repository if it does not exist}

If checked, the task will check to see if the repository exists and if it does not, will attempt to create it.

\textbf{Image Tag Output Variable}

The name of a build variable that will be created or updated with the pushed image reference. The image tag will be of the form \texttt{aws\_account\_id.dkr.ecr.region.amazonaws.com/imagename}, where \texttt{imagename} is in the format \texttt{repositoryname[:tag]}.

\textbf{Task Permissions}

This task requires permissions to call the following AWS service APIs (depending on selected task options, not all APIs may be used):

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```
AWS Elastic Beanstalk Create Version task

Synopsis

Creates a new version of an application that can be deployed subsequently to an Elastic Beanstalk environment associated with the application.

Description

With Elastic Beanstalk, you can quickly deploy and manage applications in the AWS Cloud without worrying about the infrastructure that runs those applications. Elastic Beanstalk reduces management complexity without restricting choice or control. You simply upload your application, and Elastic Beanstalk automatically handles the details of capacity provisioning, load balancing, scaling, and application health monitoring.

This task can upload and register new versions of ASP.NET applications (as Web Deploy archives), ASP.NET Core applications or an existing application bundle previously uploaded to Amazon S3. The application version can then be deployed separately to an Elastic Beanstalk environment associated with the application using the Elastic Beanstalk Deployment task.

Parameters

You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

Display name*

The default name of the task instance, which can be modified: Create Elastic Beanstalk Version

AWS Credentials

Specifies the AWS credentials to be used by the task in the build agent environment.
You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:

- From task variables named `AWS.AccessKeyId`, `AWS.SecretAccessKey` and optionally `AWS.SessionToken`.
- From credentials set in environment variables in the build agent process. When using environment variables in the build agent process you may use the standard AWS environment variables: `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` and optionally `AWS_SESSION_TOKEN`.
- If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

**AWS Region**

The AWS Region code (for example, us-east-1, us-west-2) of the Region containing the AWS resources the task will use or create. For more information, see Regions and endpoints in the Amazon Web Services General Reference.

If a Region is not specified in the task configuration the task will attempt to obtain the Region to be used using the standard AWS environment variable `AWS_REGION` in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

**Note:** The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed may still be used by entering the region code of the Region (for example, `us_west_2`).

**Application Name***

The name of the Elastic Beanstalk application.

**Deployment Bundle Type***

The type of application bundle for which a new revision will be created in (EB). You can select from

- ASP.NET: the deployment bundle is expected to be a Web Deploy archive, built previously, which the task will upload.
- ASP.NET Core: the deployment bundle will be created by the task (using the `dotnet publish` command line tool) and uploaded.
- Existing deployment bundle: choose to deploy a bundle that has been built and uploaded previously to Amazon S3.

**Web Deploy Archive**

Required if Deployment Bundle Type is set to ASP.NET. The path to the web deploy archive containing the application to deploy to Elastic Beanstalk.

**Published Application Path**

Required if Deployment Bundle Type is set to ASP.NET Core. The output location where the `dotnet publish` command in your previous build steps placed the deployment artifacts to be published. Configure using either:
The path to the output folder containing the artifacts. Use this if the dotnet publish command in your build was configured to not create a zip file of the published application.

The path and filename of the zip file containing the artifacts. Use this if the dotnet publish command in your build was configured to create a zip file of the application artifacts.

### Deployment Bundle Bucket

Required if Deployment Bundle Type is set to Existing deployment bundle. The name of the Amazon S3 bucket containing the revision bundle to deploy.

### Deployment Bundle Object Key

Required if Deployment Bundle Type is set to Existing deployment bundle. The Amazon S3 object key of the revision bundle file to be deployed.

### Description

Optional description for the new revision.

### Version Label

Version label for the new application revision. If not specified the task will construct a version label based on the current date and time, expressed in milliseconds (for example v20171120222623).

### Version Label Output Variable

Optional variable name to which the version label for the revision will be stored on conclusion of the task. This is useful when Version Label is not specified and the task generates a version label for the revision. You can refer to this variable in subsequent build steps to obtain the deployed version label.

### Task Permissions

This task requires permissions to call the following AWS service APIs (depending on selected task options, not all APIs may be used):

- elasticbeanstalk:CreateApplicationVersion
- elasticbeanstalk:CreateStorageLocation
- elasticbeanstalk:DescribeApplications
- elasticbeanstalk:DescribeEnvironments

The task also requires permissions to upload your application content to the specified Amazon S3 bucket. Depending on the size of the application bundle, either PutObject or the S3 multi-part upload APIs may be used.

### AWS Elastic Beanstalk Deploy Application task

#### Synopsis

Deploys a new version of an application to an Elastic Beanstalk environment associated with the application.
Description

With Elastic Beanstalk, you can quickly deploy and manage applications in the AWS Cloud without worrying about the infrastructure that runs those applications. Elastic Beanstalk reduces management complexity without restricting choice or control. You simply upload your application, and Elastic Beanstalk automatically handles the details of capacity provisioning, load balancing, scaling, and application health monitoring.

This task can deploy ASP.NET applications (as Web Deploy archives), ASP.NET Core applications, an existing built application or a previously registered application version using the Elastic Beanstalk Create Version task.

Parameters

You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

Display name*

The default name of the task instance, which can be modified: Deploy to Elastic Beanstalk

AWS Credentials

Specifies the AWS credentials to be used by the task in the build agent environment.

You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:

- From task variables named AWS.AccessKeyId, AWS.SecretAccessKey and optionally AWS.SessionToken.
- From credentials set in environment variables in the build agent process. When using environment variables in the build agent process you may use the standard AWS environment variables: AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY and optionally AWS_SESSION_TOKEN.
- If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

AWS Region

The AWS Region code (for example, us-east-1, us-west-2) of the Region containing the AWS resources the task will use or create. For more information, see Regions and Endpoints in the Amazon Web Services General Reference.

If a Region is not specified in the task configuration, the task will attempt to obtain the Region to be used by using the standard AWS environment variable AWS_REGION in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region by using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

Note: The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed can still be used by entering the region code of the Region (for example, us_west_2).
**Application Name***

The name of the Elastic Beanstalk application.

**Environment Name***

The name of the Elastic Beanstalk environment that will run the application.

An environment represents the AWS resources (e.g., load balancer, Auto Scaling group, and Amazon EC2 instances) created specifically to run your application.

**Deployment Bundle Type***

The type of application bundle to deploy. You can select from

- **ASP.NET**: the deployment bundle is expected to be a Web Deploy archive, built previously, which the task will upload.
- **ASP.NET Core**: the deployment bundle will be created by the task (using the `dotnet publish` command line tool) and uploaded.
- **Existing deployment bundle**: choose to deploy a bundle that has been built and uploaded previously to Amazon S3.
- **Existing application version**: choose to deploy a revision previously registered with Elastic Beanstalk.

**Web Deploy Archive**

Required if **Deployment Bundle Type** is set to **ASP.NET**. The path to the web deploy archive containing the application to deploy to Elastic Beanstalk.

**Published Application Path**

Required if **Deployment Bundle Type** is set to **ASP.NET Core**. The output location where the `dotnet publish` command in your previous build steps placed the deployment artifacts to be published. Configure using either:

- The path to the output folder containing the artifacts. Use this if the `dotnet publish` command in your build was configured to not create a .zip file of the published application.
- The path and filename of the .zip file containing the artifacts. Use this if the `dotnet publish` command in your build was configured to create a .zip file of the application artifacts.

**Deployment Bundle Bucket**

Required if **Deployment Bundle Type** is set to **Existing deployment bundle**. The name of the Amazon S3 bucket containing the revision bundle to deploy.

**Deployment Bundle Object Key**

Required if **Deployment Bundle Type** is set to **Existing deployment bundle**. The Amazon S3 object key of the revision bundle file to be deployed.

**Version Label**

Version label for the new application revision. If not specified the task will construct a version label based on the current date and time, expressed in milliseconds (for example `v20171120222623`).
Version Label Output Variable

Optional variable name to which the version label for the revision will be stored on conclusion of the task. This is useful when Version Label is not specified and the task generates a version label for the revision. You can refer to this variable in subsequent build steps to obtain the deployed version label.

Task Permissions

This task requires permissions to call the following AWS service APIs (depending on selected task options, not all APIs may be used):

- elasticbeanstalk:CreateApplicationVersion
- elasticbeanstalk:CreateStorageLocation
- elasticbeanstalk:DescribeApplications
- elasticbeanstalk:DescribeEnvironments
- elasticbeanstalk:DescribeEvents
- elasticbeanstalk:UpdateEnvironment

The task also requires permissions to upload your application content to the specified Amazon S3 bucket. Depending on the size of the application bundle, either PutObject or the S3 multi-part upload APIs may be used.

AWS Lambda Deploy Function task

Synopsis

Supports deployment of AWS Lambda functions for all supported Lambda language runtimes. Note that this task can be used to deploy .NET Core-based functions but it does not build the deployment package first. To perform a build and deployment for .NET Core-based functions, or to deploy .NET Core-based serverless applications, please refer to the AWS Lambda .NET Core Deployment task.

Description

Applications that are based on Lambda (also referred to as serverless applications) are composed of functions triggered by events. A typical serverless application consists of one or more functions triggered by events such as object uploads to Amazon S3, Amazon SNS notifications, and API actions. Those functions can stand alone or use other resources such as Amazon DynamoDB tables or Amazon S3 buckets. The most basic serverless application is simply a function.

Parameters

You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

Display name*

The default name of the task instance, which can be modified: Deploy Lambda Function

AWS Credentials

Specifies the AWS credentials to be used by the task in the build agent environment.
You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:

- From task variables named AWS.AccessKeyId, AWS.SecretAccessKey and optionally AWS.SessionToken.
- From credentials set in environment variables in the build agent process. When using environment variables in the build agent process you may use the standard AWS environment variables: AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY and optionally AWS_SESSION_TOKEN.
- If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

**AWS Region**

The AWS Region code (for example, us-east-1, us-west-2) of the Region containing the AWS resources the task will use or create. For more information, see Regions and endpoints in the Amazon Web Services General Reference.

If a Region is not specified in the task configuration the task will attempt to obtain the Region to be used using the standard AWS environment variable AWS_REGION in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

**Note:** The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed may still be used by entering the region code of the Region (for example, us_west_2).

**Deployment Mode***

Selects the type of deployment. You can deploy new function code to an existing function or you can specify settings for both code and configuration. For the 'code and configuration' mode if the function does not exist it will be created.

**Function Name***

The name of the Lambda function to create or update. You can also specify the Amazon Resource Name (ARN) for an existing function.

**Description**

A short, user-defined function description. Lambda does not use this value.

**Function Handler***

*The function within your code that Lambda calls to begin execution. For Node.js, it is the module-name.export value in your function. For Java, it can be package.class-name::handler or package.class-name. For more information and other examples see Programming Model.

**Runtime***

The runtime environment for the Lambda function you are uploading. The list of runtimes available in the pick list are those known at the time this version of the tools was released. To use a runtime not shown in the list simply enter the runtime identifier in the field.
Parameters

**Code Location**
Specifies the source location of the deployment package to be uploaded. You can choose from a file in the local file system or a file previously uploaded to Amazon S3. If the source location is Amazon S3 you can also optionally supply a specific version of the file.

**Zip File Path**
Path to the zip file containing the function code to deploy. Required if Code Location is set to Zip file in the work area.

**S3 Bucket**
The name of the Amazon S3 bucket containing the previously uploaded zip file of the function's code. Required if Code Location is set to Zip file in Amazon S3.

**S3 Object Key**
The key (name) of the object in the bucket containing the function's code. Required if Code Location is set to Zip file in Amazon S3.

**S3 Object Version**
Version of the S3 object containing the function code. If not specified the latest version of the object is used.

**Role ARN or Name**
The Amazon Resource Name (ARN), or name, of the IAM role that Lambda assumes when it executes your function to access any other Amazon Web Services (AWS) resources. If a role name is supplied the task will attempt to retrieve the ARN automatically.

**Memory Size**
The amount of memory, in MB, your Lambda function is given. Lambda uses this memory size to infer the amount of CPU and memory allocated to your function. Your function use-case determines your CPU and memory requirements. For example, a database operation might need less memory compared to an image processing function. The default value is 128 MB. The value must be a multiple of 64 MB.

**Timeout**
The function execution time at which Lambda should terminate the function. Because the execution time has cost implications, we recommend you set this value based on your expected execution time. The default is 3 seconds.

**Publish**
If set requests AWS Lambda to create or update the Lambda function and publish a version as an atomic operation.

**Advanced**
Advanced settings are only displayed when creating a new function, or updating code and configuration for an existing function.
Dead Letter ARN

The Amazon Resource Name (ARN) of an Amazon SQS queue or Amazon SNS topic to be used as your Dead Letter Queue (DLQ).

KMS Key ARN

The Amazon Resource Name (ARN) of the KMS key used to encrypt your function's environment variables. If not provided, AWS Lambda will use a default service key.

Environment Variables

Key-value pairs that represent your environment's configuration settings. Enter as Name=Value, one per line.

Tags

List of tags (key-value pairs) assigned to the new function. Enter as key=value, one per line. Tags can only be specified when creating a new function and are ignored when updating functions.

Security Group IDs

List of security group IDs, one per line. If your Lambda function accesses resources in a VPC at least one security group and one subnet ID belonging to the same VPC must be specified.

Subnet IDs

List of subnet IDs, one per line. If your Lambda function accesses resources in a VPC at least one security group and one subnet ID belonging to the same VPC must be specified.

Tracing configuration

Your function's trace settings. Can be either X-Ray, PassThrough or Active. If PassThrough, Lambda will only trace the request from an upstream service if it contains a tracing header with "sampled=1". If Active, Lambda will respect any tracing header it receives from an upstream service. The default setting of X-Ray means that if no tracing header is received, Lambda will call X-Ray for a tracing decision.

Output Variable

The name of the variable that will contain the Amazon Resource Name (ARN) of the created or updated function on task completion. The variable can be used as $(variableName) to refer to the function result in subsequent tasks.

Task Permissions

This task requires permissions to call the following AWS service APIs (depending on selected task options, not all APIs may be used):

- lambda:CreateFunction
- lambda:GetFunction
- lambda:GetFunctionConfiguration
- lambda:UpdateFunctionCode
- lambda:UpdateFunctionConfiguration
AWS Lambda Invoke Function task

Synopsis

Invokes an AWS Lambda function with a JSON payload.

Description

This task invokes a previously deployed Lambda function.

Parameters

You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

Display name*

The default name of the task instance, which can be modified: Invoke Lambda Function

AWS Credentials

Specifies the AWS credentials to be used by the task in the build agent environment.

You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:

- From task variables named $AWS.AccessKeyID$, $AWS.SecretAccessKey$ and optionally $AWS.SessionToken$.
- From credentials set in environment variables in the build agent process. When using environment variables in the build agent process you may use the standard AWS environment variables: $AWS_ACCESS_KEY_ID$, $AWS_SECRET_ACCESS_KEY$ and optionally $AWS_SESSION_TOKEN$.
- If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

AWS Region

The AWS Region code (for example, us-east-1, us-west-2) of the Region containing the AWS resources the task will use or create. For more information, see Regions and endpoints in the Amazon Web Services General Reference.

If a Region is not specified in the task configuration the task will attempt to obtain the Region to be used using the standard AWS environment variable $AWS_REGION$ in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

Note: The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed may still be used by entering the region code of the Region (for example, $us_west_2$).
**Function Name**

The name of the Lambda function to invoke. You can also specify the Amazon Resource Name (ARN) of the function.

**Payload**

The JSON formatted payload to pass to the function.

**Invocation Type**

Either *Asynchronous execution* or *Synchronous execution returning the output from the function*.

**Synchronous Execution Output**

**Output Variable**

The name of the variable that will contain the function output on task completion. You can use the variable as `${variableName}` to refer to the function result in subsequent tasks.

**Log Type**

For synchronous execution, returns the base64-encoded last 4 KB of log data produced by your Lambda function in the `x-amz-log-result` header.

**Task Permissions**

This task requires permissions to call the following AWS service APIs (depending on selected task options, not all APIs may be used):

- lambda:GetFunctionConfiguration
- lambda:InvokeFunction

---

**AWS Lambda .NET Core task**

**Synopsis**

Builds, packages, and deploys a .NET Core AWS Lambda function or serverless application. Optionally the task can create the deployment package for subsequent deployment in another build or release pipeline.

*Note:* this task is specific to Lambda functions written in C# or F#. For other languages supported by Lambda please refer to the AWS Lambda Deploy Function task.

**Description**

Applications based on Lambda (also referred to as serverless applications) are composed of functions triggered by events. A typical serverless application consists of one or more functions triggered by events such as object uploads to Amazon S3, Amazon SNS notifications, and API actions. Those functions can stand alone or use other resources such as Amazon DynamoDB tables or Amazon S3 buckets. The most basic serverless application is simply a function.
Parameters

You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

Display name*

The default name of the task instance, which can be modified: Deploy .NET Core to Lambda

AWS Credentials

Specifies the AWS credentials to be used by the task in the build agent environment.

You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:

- From task variables named `AWS.AccessKeyId`, `AWS.SecretAccessKey` and optionally `AWS.SessionToken`.
- From credentials set in environment variables in the build agent process. When using environment variables in the build agent process you may use the standard AWS environment variables: `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` and optionally `AWS_SESSION_TOKEN`.
- If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

AWS Region

The AWS Region code (for example, us-east-1, us-west-2) of the Region containing the AWS resources the task will use or create. For more information, see Regions and endpoints in the Amazon Web Services General Reference.

If a Region is not specified in the task configuration the task will attempt to obtain the Region to be used using the standard AWS environment variable `AWS_REGION` in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

Note: The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed may still be used by entering the region code of the Region (for example, us_west_2).

Deployment Type*

The type of deployment to perform, or package to build or deploy.

- Function deploys a single function to Lambda, or creates a package zip file for subsequent deployment.
- Serverless Application performs a deployment using AWS CloudFormation (allowing multiple functions to be deployed at the same time) or builds the application and uploads it to Amazon S3, outputting the serverless template file for subsequent deployment of the updated code using AWS CloudFormation.
**Parameters**

**Note:** both options will perform the relevant NuGet package restore and build operations to create the resulting deployment package.

**Create deployment package only**

If selected the task creates the outputs for the selected deployment type but does not perform the deployment to AWS Lambda or AWS CloudFormation.

**Package-only output file**

Available when **Create deployment package only** is selected.

When **Deployment Type** is set to **Function** specifies the output folder and filename of the packaged .zip file. This .zip file can then be used with the **AWS Lambda Deploy Function** task to perform the deployment at a later stage.

When **Deployment Type** is set to **Serverless Application** specifies the output folder and file name where the serverless template file, updated to contain the Amazon S3 location of the built project code and artifacts, will be placed. This updated template can then be used with the **AWS CloudFormation Create/Update Stack** task, or AWS CloudFormation change set tasks, to perform the deployment at a later stage.

**Path to Lambda Project**

The relative path to the location of the Lambda function or serverless application project to package and/or deploy.

**Function Deployment: Lambda Function Properties**

**Function Name**

The name of the Lambda function to invoke. You can also specify the Amazon Resource Name (ARN) of the function when deploying to an existing function.

**Function Role**

The name of the IAM role providing access to AWS services for the deployed Lambda function.

**Function Handler**

The function within your code that Lambda calls to begin execution. The format is `<assembly-name>::<namespace.type-name>::<function-name>`.

**Function Memory (MB)**

The memory allocated to the Lambda function. The value must be in multiples of 64.

**Function Timeout (Seconds)**

The function execution time at which Lambda should terminate the function.

**Serverless Application Deployment: Serverless Application Properties**

**Stack Name**

The name of the AWS CloudFormation stack to deploy to.
**Note:** This field is required when performing a deployment of a serverless application using this task. When performing a package-only build this field is ignored as the stack name is only relevant during deployment.

**S3 Bucket**

The name of the Amazon S3 bucket used to store the built project code. This field is required when performing a either a deployment or package-only build of a serverless application.

**S3 Prefix**

The object key prefix to be used for the packaged objects that will be uploaded to Amazon S3 for subsequent deployment.

**Advanced**

**Additional Command Line Arguments for Lambda Tools**

Additional arguments that can be passed to the `dotnet lambda` CLI extension command that is used to build, package and deploy your function or serverless application using this task.

**Task Permissions**

This task requires permissions to call the following AWS service APIs (depending on selected task options, not all APIs may be used):

- `lambda:CreateFunction`
- `lambda:UpdateFunctionCode`
- `lambda:GetFunctionConfiguration`
- `cloudformation:CreateChangeSet`
- `cloudformation:ExecuteChangeSet`
- `cloudformation:DescribeStackEvents`
- `cloudformation:DeleteStack`
- `cloudformation:DescribeChangeSet`
- `cloudformation:DescribeStacks`
- `s3:CreateBucket`
- `s3:GetBucketLocation`

The task also requires permissions to upload your Lambda function or serverless application content to the specified Amazon S3 bucket. Depending on the size of the application bundle, either `PutObject` or the S3 multi-part upload APIs may be used.

**Amazon S3 Download task**

**Synopsis**

Downloads file and folder content from an Amazon Simple Storage Service (S3) bucket.
Description

Downloads file and folder content from an Amazon Simple Storage Service (S3) bucket to a folder location. The source location in the bucket, or key prefix, can also be specified. If a source location is not supplied, the bucket root is used. You specify the files to download using a set of one or more globbing patterns. The default pattern is **, causing all files in all folders at and beneath the source location to be downloaded, preserving the relative folder paths.

Parameters

You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

Display name*

The default name of the task instance, which can be modified: S3 Download

AWS Credentials

Specifies the AWS credentials to be used by the task in the build agent environment.

You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:

- From task variables named AWS.AccessKeyId, AWS.SecretAccessKey and optionally AWS.SessionToken.
- From credentials set in environment variables in the build agent process. When using environment variables in the build agent process you may use the standard AWS environment variables: AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY and optionally AWS_SESSION_TOKEN.
- If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

AWS Region

The AWS Region code (for example, us-east-1, us-west-2) of the Region containing the AWS resources the task will use or create. For more information, see Regions and endpoints in the Amazon Web Services General Reference.

If a Region is not specified in the task configuration the task will attempt to obtain the Region to be used using the standard AWS environment variable AWS_REGION in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

Note: The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed may still be used by entering the region code of the Region (for example, us_west_2).

Bucket Name*

The name of the Amazon S3 bucket containing the content to download.
Source Folder

The source folder (or S3 key prefix) in the bucket that the filename selection patterns will be run against to select objects to download. If not set the root of the bucket is assumed.

Filename Patterns

Glob patterns to select the file and folder content to download. Supports multiple lines of minimatch patterns. The default is **.

Target Folder*

The target folder on the build host to contain the downloaded content. You can browse for it or you can use variables.

Server-Side Encryption

Encryption Key Management

When you retrieve an object from Amazon S3 that was encrypted by using server-side encryption with customer-provided encryption keys (SSE-C), set Use customer-provided encryption key and provide the customer key data to enable the objects to be decrypted. If the objects were encrypted using an Amazon S3-provided key leave this option set to the default value, Not using server-side encryption, or encrypted using an Amazon S3 managed key.

Customer Key

Available, and required, when Encryption Key Management is set to Use customer-provided encryption key. Hex-encoded string representing the encryption key for Amazon S3 to use in decrypting data. This value is used to decrypt the object and then is discarded; Amazon does not store the encryption key. This value must be appropriate for use with the AES256 encryption algorithm used for encryption when customer managed keys are selected.

Advanced

Overwrite

Changing this checkbox has no effect. If a file (an Amazon S3 object) with the same name already exists in the Amazon S3 bucket, it will always be overwritten.

Force path style addressing

If selected path style URLs will be used when working with the bucket. The default is off meaning the task will automatically switch between virtual host style addressing and path style addressing depending on whether the bucket name is DNS compatible.

For more information see Virtual Hosting of Buckets.

Flatten folders

If selected, the task will remove the key prefix from the downloaded objects causing them to be written to the selected download folder without subpaths.

If this option is unchecked, the key prefix of each object is preserved and objects are downloaded to a subfolder hierarchy matching the key prefix of the object.
Note
If folder flattening is selected and multiple objects with the same name but different key
prefixes exist in the download set, earlier objects will be overwritten with later objects.

Task Permissions

This task requires permissions to call the following AWS service APIs (depending on selected task
options, not all APIs may be used):

• s3:GetObject
• s3:HeadBucket
• s3:ListObjects

Amazon S3 Upload task

Synopsis

Uploads file and folder content to an Amazon Simple Storage Service (S3) bucket.

Description

This task accepts a source location from which to upload files to an Amazon S3 bucket. The target
location in the bucket, or key prefix, can also be specified. If you don’t supply a target location, the files
are uploaded to the bucket root. You specify the files to upload by using a set of one or more globbing
patterns. The default pattern is **, which causes all files in all folders at and beneath the source location
to be uploaded, preserving the relative folder paths.

The task can optionally create the bucket to which the content is to be uploaded.

Parameters

You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

Display name*

The default name of the task instance, which can be modified: S3 Upload

AWS Credentials

Specifies the AWS credentials to be used by the task in the build agent environment.

You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can
leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:

• From task variables named AWS.AccessKeyId, AWS.SecretAccessKey and optionally AWS.SessionToken.
• From credentials set in environment variables in the build agent process. When using environment
variables in the build agent process you may use the standard AWS environment variables:
AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY and optionally AWS_SESSION_TOKEN.
• If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with
the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have
been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

AWS Region

The AWS Region code (for example, us-east-1, us-west-2) of the Region containing the AWS resources the task will use or create. For more information, see Regions and endpoints in the Amazon Web Services General Reference.

If a Region is not specified in the task configuration the task will attempt to obtain the Region to be used using the standard AWS environment variable AWS_REGION in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

Note: The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed may still be used by entering the region code of the Region (for example, us_west_2).

Bucket Name*

The name of the Amazon S3 bucket to which the content will be uploaded. If the bucket does not exist it can be created if the Create S3 bucket if it does not exist option is selected.

Note: bucket names must be globally unique.

Source Folder

The source folder that the filename selection patterns will be run against. If not set the root of the work area is assumed. You can also use variables to specify the folder.

Example: code:$(Build.ArtifactStagingDirectory)

Filename Patterns

Glob patterns to select the file and folder content to be uploaded. Supports multiple lines of minimatch patterns.

Target Folder*

The target folder (referred to as a key prefix in Amazon S3) in the bucket to contain the uploaded content. If not set the root of the bucket is assumed. You can also use variables to specify the folder/key prefix value.

Access Control (ACL)

The canned access control list (ACL) to apply to the uploaded content. See Canned ACL for an explanation of the possible values. By default all uploaded content is marked Private.

Create S3 Bucket if it does not exist

If checked and the specified bucket does not exist, the task attempts to automatically create it.

Note: bucket names must be globally unique.
Server-Side Encryption

Encryption Key Management

You can optionally request Amazon S3 to encrypt data at rest using server-side encryption. Server-side encryption is about data encryption at rest, that is, Amazon S3 encrypts your data as it writes it to disks in its data centers and decrypts it for you when you access it.

Select Use AWS-managed encryption keys if you want Amazon S3 to manage keys used to encrypt data. To manage and provide your own keys select Use customer-provided encryption keys. Selecting Not using server-side encryption disables server-side encryption for the uploaded objects.

Encryption Algorithm

Specifies a server-side encryption algorithm to use when Amazon S3 creates an object.

KMS Master Encryption Key ID

The ID of the AWS Key Management Service (KMS) master encryption key to be used when encrypting the object.

This field is required if Encryption Algorithm is set to aws:kms.

Customer Key

Hex-encoded string representing the encryption key for Amazon S3 to use in encrypting data. This value is used to store the object and then is discarded; Amazon S3 does not store the encryption key. This value must be appropriate for use with the AES256 encryption algorithm used for encryption when customer managed keys are selected.

This field is required when Encryption Key Management is set to Use customer-provided encryption key.

Advanced

Overwrite

Changing this checkbox has no effect. If a file (an Amazon S3 object) with the same name already exists in the Amazon S3 bucket, it will always be overwritten.

Flatten Folders

If selected the relative subfolders of the files being uploaded are removed and all files are placed directly into the target location. The default behavior is to preserve the relative folder hierarchy.

Content Type

Sets a custom content type for the uploaded files. If a custom content type is not specified the task will apply built-in defaults for common file types (html, css, js, image files etc.). This parameter can be used to override the built-in defaults.

Note: any value specified is applied to all files processed by the task.

Storage Class

Choose a storage class depending on your use case scenario and performance access requirements.

- STANDARD – This storage class (the default) is ideal for performance-sensitive use cases and frequently accessed data.
• **STANDARD_IA** – This storage class (IA, for infrequent access) is optimized for long-lived and less frequently accessed data, for example backups and older data where frequency of access has diminished, but the use case still demands high performance. **Note** There is a retrieval fee associated with STANDARD_IA objects which makes it most suitable for infrequently accessed data.

• **REDUCED_REDUndANCY** – The Reduced Redundancy Storage (RRS) storage class is designed for noncritical, reproducible data stored at lower levels of redundancy than the STANDARD storage class, which reduces storage costs.

For more information see [Storage Classes](#) in the Amazon S3 documentation for more information.

**Force path style addressing**

If selected path style URLs will be used for S3 objects. The default is off meaning the task will automatically switch between virtual host style addressing and path style addressing depending on whether the bucket name is DNS compatible.

For more information see [Virtual Hosting of Buckets](#).

**Task Permissions**

This task requires permissions to call the following AWS service APIs (depending on selected task options, not all APIs may be used):

- `s3:CreateBucket`
- `s3:HeadBucket`

Content uploads are performed using S3’s PutObject API and/or the multi-part upload APIs. The specific APIs used depend on the size of the individual files being uploaded.

---

**AWS Secrets Manager Create/Update Secret task**

**Synopsis**

Updates a secret, optionally creating a secret if it does not exist.

**Description**

Use this task to create a new secret in Secrets Manager or to update the value for an existing secret.

**Parameters**

You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

**Display name** *

The default name of the task instance, which can be modified: Secrets Manager Create/Update Secret

**AWS Credentials**

Specifies the AWS credentials to be used by the task in the build agent environment.
You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:

- From task variables named `AWS.AccessKeyId`, `AWS.SecretAccessKey` and optionally `AWS.SessionToken`.
- From credentials set in environment variables in the build agent process. When using environment variables in the build agent process you may use the standard AWS environment variables: `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` and optionally `AWS_SESSION_TOKEN`.
- If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

**AWS Region**

The AWS Region code (for example, `us-east-1`, `us-west-2`) of the Region containing the AWS resources the task will use or create. For more information, see Regions and endpoints in the Amazon Web Services General Reference.

If a Region is not specified in the task configuration the task will attempt to obtain the Region to be used using the standard AWS environment variable `AWS_REGION` in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

**Note:** The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed may still be used by entering the region code of the Region (for example, `us_west_2`).

**Secret Name**

Specifies the friendly name of the new secret. The secret name must be ASCII letters, digits, or the following characters: `/+._=` (spaces are not permitted).


If updating an existing secret you can specify either the Amazon Resource Name (ARN) or the friendly name of the secret.

**Description**

Optional description of the secret.

**Secret Value Location**

Specifies the source of the value to be stored in the secret. You can enter text values for secrets inline in the task configuration or in a file loaded when the task runs. Binary secret values must be loaded from a file.

**Secret Value**

Specifies the text value that you want to store in this secret. For storing multiple values we recommend that you use a JSON text string argument and specify key/value pairs.

Required if Secret Value Location is set to Inline.
Secret Value Type

Specifies whether the file contents being stored in the secret text or binary data.

**Note:** to satisfy the service's API requirements the task will automatically base-64 encode secrets specified as binary type; you do not need to perform the base-64 encoding prior to specifying the secret value in the task.

Path to File Containing Secret Value

Specifies the file containing the value (text or binary) that you want to store in this secret.

Required if **Secret Value Location** is set to **From File**.

KMS Key ID

Specifies the ARN or alias of the AWS KMS customer master key (CMK) to be used to encrypt the secret.

If you don't specify this value, then Secrets Manager defaults to using the AWS account's default CMK (the one named aws/secretsmanager). If a KMS CMK with that name doesn't yet exist, then Secrets Manager creates it for you automatically the first time it needs to encrypt a secret.

**Important:** You can use the account's default CMK to encrypt and decrypt only if you call this operation using credentials from the same account that owns the secret. If the secret is in a different account, then you must create a custom CMK and specify the ARN in this field.

Create secret if it does not exist

If the specified secret does not exist, attempt to create a new secret. Secrets Manager automatically attaches the staging label _AWSCURRENT_ to the new version. If this option is not selected, the task will return an error if the secret cannot be found.

Tags for New Secret

Optional list of tags (key-value pairs) that can be assigned to the new secret. Enter as Key=Value, one per line. Up to 50 tags can be applied to a secret.

Output variable name to contain the secret's ARN

Optional name of a variable to store the ARN of the new or updated secret on task completion.

Output variable name to contain the secret's version ID

Optional name of a variable to store the version ID of the new or updated secret on task completion.

Task Permissions

This task requires permissions to call the following AWS service APIs (depending on selected task options, not all APIs may be used):

- secretsmanager:CreateSecret
- secretsmanager:PutSecretValue
- secretsmanager:UpdateSecret
AWS Secrets Manager Get Secret task

Synopsis
Stores the value of a secret in AWS Secrets Manager into a secret build variable.

Description
Use this task to retrieve the value of a secret stored in AWS Secrets Manager and store it locally in an Azure DevOps build variable. The build variable will be automatically set to 'secret' mode to automatically mask the value when logged or otherwise displayed.

Parameters
You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

Display name*
The default name of the task instance, which can be modified: Secrets Manager Get Secret

AWS Credentials
Specifies the AWS credentials to be used by the task in the build agent environment.

You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:

- From task variables named AWS.AccessKeyId, AWS.SecretAccessKey and optionally AWS.SessionToken.
- From credentials set in environment variables in the build agent process. When using environment variables in the build agent process you may use the standard AWS environment variables: AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY and optionally AWS_SESSION_TOKEN.
- If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

AWS Region
The AWS Region code (for example, us-east-1, us-west-2) of the Region containing the AWS resources the task will use or create. For more information, see Regions and endpoints in the Amazon Web Services General Reference.

If a Region is not specified in the task configuration the task will attempt to obtain the Region to be used using the standard AWS environment variable AWS_REGION in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

Note: The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed may still be used by entering the region code of the Region (for example, us_west_2).
Secret ID/Name
Specifies the secret containing the version that you want to retrieve. You can specify either the Amazon Resource Name (ARN) or the friendly name of the secret.

Version ID
Specifies the unique identifier of the version of the secret that you want to retrieve. If you specify this parameter then don't specify Version Stage. If you don't specify either a Version Stage or Version ID then the default is to perform the operation on the version with the version stage value of AWSCURRENT.

Version Stage
Specifies the version of the secret that you want to retrieve using the staging label attached to the version.

Staging labels are used to keep track of different versions during the rotation process. If you use this parameter then don't specify Version ID. If you don't specify either a Version Stage or Version ID, then the default is to perform the operation on the version with the version stage value of AWSCURRENT.

Task Permissions
This task requires permissions to call the following AWS service APIs (depending on selected task options, not all APIs may be used):

- secretsmanager:GetSecretValue

AWS Send SNS or SQS Message task

Synopsis
Sends a message to an Amazon Simple Notification Service (SNS) topic or to an Amazon Simple Queue Service (SQS) queue.

Description
This task accepts a message to be sent to an Amazon SNS topic or to an Amazon SQS queue. If the message is to be sent to a queue, you can configure an optional delay (in seconds). If you don't specify a delay, the task assumes the default delay that is associated with the queue.

Parameters
You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

Display name*
The default name of the task instance, which can be modified: Send Message

AWS Credentials
Specifies the AWS credentials to be used by the task in the build agent environment.
Parameters

You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:

- From task variables named `AWS.AccessKeyId`, `AWS.SecretAccessKey` and optionally `AWS.SessionToken`.
- From credentials set in environment variables in the build agent process. When using environment variables in the build agent process you may use the standard AWS environment variables: `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` and optionally `AWS_SESSION_TOKEN`.
- If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

**AWS Region**

The AWS Region code (for example, us-east-1, us-west-2) of the Region containing the AWS resources the task will use or create. For more information, see Regions and endpoints in the Amazon Web Services General Reference.

If a Region is not specified in the task configuration the task will attempt to obtain the Region to be used using the standard AWS environment variable `AWS_REGION` in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

**Note:** The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed may still be used by entering the region code of the Region (for example, `us_west_2`).

**Message Target***

The destination for the message. A message can be sent to a Amazon SNS (SNS) topic or a Amazon SQS (SQS) queue.

**Message**

The message content to send. The maximum size for both queue and topic targets is 256KB (262144 bytes, not 262144 characters).

For more information on the allowed values and content see the respective service help pages for Publish and SendMessage.

**Topic ARN***

The Amazon Resource Name (ARN) of the Amazon SNS topic to which the message will be sent. Required when Message Target is set to SNS Topic.

**Queue Url***

The URL of the Amazon SQS queue to which the message will be sent. Required when Message Target is set to SQS Queue.

**Delay (seconds)**

Available for Amazon SQS queues only.
Task Permissions

This task requires permissions to call the following AWS service APIs (depending on selected task options, not all APIs may be used):

- `sns:GetTopicAttributes`
- `sns:Publish`
- `sqs:GetQueueAttributes`
- `sqs:SendMessage`

AWS SSM Get Parameter task

(AWS Systems Manager Get Parameter Task)

Synopsis

Reads one or more values from Systems Manager Parameter Store into build variables.

Description

This task reads a parameter value, or hierarchy of values identified by common path, into build variables in the build or release definition. These variables are then accessible from downstream tasks in the definition. The names used for the build variables are customizable.

Parameters

You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

Display name*

The default name of the task instance, which can be modified: Systems Manager Get Parameter

AWS Credentials

Specifies the AWS credentials to be used by the task in the build agent environment.

You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:

- From task variables named `AWS.AccessKeyId`, `AWS.SecretAccessKey` and optionally `AWS.SessionToken`.
- From credentials set in environment variables in the build agent process. When using environment variables in the build agent process you may use the standard AWS environment variables: `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` and optionally `AWS_SESSION_TOKEN`.
- If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have
Parameters

been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

AWS Region

The AWS Region code (for example, us-east-1, us-west-2) of the Region containing the AWS resources the task will use or create. For more information, see Regions and endpoints in the Amazon Web Services General Reference.

If a Region is not specified in the task configuration the task will attempt to obtain the Region to be used using the standard AWS environment variable AWS_REGION in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

Note: The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed may still be used by entering the region code of the Region (for example, us_west_2).

Read Mode*

Whether the task gets the value of a single named parameter or values from a parameter hierarchy identified by common parameter path.

Parameter Name

The name identifying a single parameter to be read from the store. Required if Read Mode is set to Get value for single parameter.

Parameter Version

If unspecified the value associated with the latest version of the parameter is read. If specified the task requests the value associated with the supplied version. Parameter versions start at 1 and increment each time a new value is stored for the parameter.

This field is only available when Read Mode is set to get a single parameter value.

Parameter Path

The path hierarchy for the parameters to be read. Hierarchies start with, and are separated by, a forward slash (/) and may contain up to five levels. The path hierarchy can identify a specific parameter in the hierarchy by appending the parameter name, or can identify a group of parameters sharing the hierarchy path. If the supplied hierarchy contains multiple parameters, all parameter values in the hierarchy are downloaded.

Note: SecureString parameters found in a hierarchy will be automatically set as secret variables.

Required if Read Mode is set to Get values for parameter hierarchy.

Recursive

Available when reading a parameter hierarchy. If selected then parameter values for the specified Parameter Path and all sub-paths are read. If not selected only the values for parameters matching the supplied path are read, values in sub-paths are ignored.
Variable Name Transform

Specifies how the build variable names to hold the parameter values are created. You can choose from:

- Use parameter names (including any paths) as variable names. The full parameter name is used to set the build variable name.
- Use leaf of parameter names as variable names. The path is removed and the resulting leaf text is used as the build variable name.
- Replace text in the parameter name using a regular expression to form the build variable name.
- Use custom name. Available for single parameter read mode only, enables entry of a custom name for the build variable.

Custom Variable Name

The name of the build variable to hold the parameter value. This value is required if Variable Name Transform is set to Use custom name.

Search Pattern

A regular expression defining the text in the parameter name that is to be replaced to form the variable name. This field is required if Variable Name Transform is set to Replace text in the parameter name using a regular expression.

Replacement Text

The text to use to replace the matched pattern defined in the Search Pattern option. If an empty string is supplied the text identified by the pattern is simply removed from the parameter name.

Global Match

If selected then a global match is performed with the specified pattern. If not selected the replacement stops after the first match.

Case-insensitive Match

If selected a case-insensitive match is performed with the specified pattern.

Task Permissions

This task requires permissions to call the following AWS service APIs (depending on selected task options, not all APIs may be used):

- ssm:GetParameter
- ssm:GetParametersByPath

AWS SSM Set Parameter Task

(AWS Systems Manager Set Parameter Task)

Synopsis

Creates or updates a parameter in Systems Manager Parameter Store.
Description
Use this task to create or update a parameter in Systems Manager Parameter Store.

Parameters
You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

Display name*
The default name of the task instance, which can be modified: Systems Manager Set Parameter

AWS Credentials
Specifies the AWS credentials to be used by the task in the build agent environment.

You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:

- From task variables named `AWS.AccessKeyId`, `AWS.SecretAccessKey` and optionally `AWS.SessionToken`.
- From credentials set in environment variables in the build agent process. When using environment variables in the build agent process you may use the standard AWS environment variables: `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` and optionally `AWS_SESSION_TOKEN`.
- If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

AWS Region
The AWS region code (for example, us-east-1, us-west-2) of the Region containing the AWS resources the task will use or create. For more information, see Regions and endpoints in the Amazon Web Services General Reference.

If a Region is not specified in the task configuration the task will attempt to obtain the Region to be used using the standard AWS environment variable `AWS_REGION` in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

Note: The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed may still be used by entering the region code of the Region (for example, `us_west_2`).

Parameter Name
The name identifying a single parameter to be created or updated in the store.

Parameter Type
The type of parameter to be written Choose from -
• String: the parameter is assigned a single string value
• String list: the parameter value is a comma-separated list of strings
• Secure string: the parameter value is encrypted at rest using either a service- or customer-provided KMS key

**Note:** If the parameter exists and is a secure string, this field is ignored and the secure string status of the parameter is retained.

### Parameter Value

The value for the parameter.

### KMS Key ID

If the parameter type is set to *Secure string*, identifies the customer-provided KMS key used to encrypt the parameter value at rest. If a secure string type is specified but no key provided a service-provided KMS key is used to encrypt the parameter value.

### Task Permissions

This task requires permissions to call the following AWS service APIs (depending on selected task options, not all APIs may be used):

- `ssm:GetParameter`
- `ssm:PutParameter`

## AWS SSM Run Command Task

(AWS Systems Manager Run Command Task)

### Synopsis

Runs a Systems Manager or user-provided Command on a fleet of EC2 instances. Commands can also target on-premise machines if the required Systems Manager agent is installed.

### Description

This task runs a Systems Manager Command, or a user-provided Command, on a fleet of EC2 instances. On-premise machines can also be targets if the required Systems Manager agent is installed. The command to run is identified by name. The targets on which the command will be run are identified using either instance IDs or tags. Parameters specific to the selected Command can also be specified.

### Parameters

You can set the following parameters for the task. Required parameters are noted by an asterisk (*). Other parameters are optional.

**Display name***

The default name of the task instance, which can be modified: Systems Manager Get Parameter
**AWS Credentials**

Specifies the AWS credentials to be used by the task in the build agent environment.

You can specify credentials using a service endpoint (of type AWS) in the task configuration or you can leave unspecified. If unspecified the task will attempt to obtain credentials from the following sources in order:

- From task variables named `AWS.AccessKeyId`, `AWS.SecretAccessKey` and optionally `AWS.SessionToken`.
- From credentials set in environment variables in the build agent process. When using environment variables in the build agent process you may use the standard AWS environment variables: `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` and optionally `AWS_SESSION_TOKEN`.
- If the build agent is running on an Amazon EC2 instance, from the instance metadata associated with the EC2 instance. For credentials to be available from EC2 instance metadata the instance must have been started with an instance profile referencing a role granting permissions to the task to make calls to AWS on your behalf. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances.

**AWS Region**

The AWS Region code (for example, us-east-1, us-west-2) of the Region containing the AWS resources the task will use or create. For more information, see Regions and endpoints in the Amazon Web Services General Reference.

If a Region is not specified in the task configuration the task will attempt to obtain the Region to be used using the standard AWS environment variable `AWS_REGION` in the build agent process's environment. Tasks running in build agents hosted on Amazon EC2 instances (Windows or Linux) will also attempt to obtain the Region using the instance metadata associated with the EC2 instance if no Region is configured on the task or set in the environment variable.

**Note:** The Regions listed in the picker are those known at the time this software was released. New Regions that are not listed may still be used by entering the region code of the Region (for example, `us_west_2`).

**Document Name**

The name of the Systems Manager document to execute. This can be a public document or a custom document private to your account and to which the credentials supplied to the task have access.

**Parameters**

The required and optional parameters for the document to be executed, specified as JSON. Refer to the specific command to be run for details.

Example format:

```
{ "parameter1" : [ "value" ], "parameter2" :
[ "value","value2" ] }
```

**Comment**

User-specified information about the command, such as a brief description of what the command should do. Maximum length 100 characters.

**Service Role ARN**

The Amazon Resource Name (ARN) or name of the IAM role Systems Manager uses to send notifications. If the name of a role is supplied the task will automatically determine the ARN.
Select Targets by*

Sets how the list of instances to be targeted are specified. You can supply a list of instance IDs, or tags (as key=value pairs) for search criteria or you can supply the instance IDs using the name of a build variable. The value of the build variable should be a comma delimited list of IDs.

Instance IDs

The instance IDs where the command should execute.

You can specify a maximum of 50 IDs, one per line. For more information about how to use Targets, see Sending Commands to a Fleet.

This parameter is required if Select Targets by is set to Manually select instances.

Tags

A list of tags that targets instances using a Key=Value combination that you specify, one per line. For more information about how to use Targets, see Sending Commands to a Fleet.

This parameter is required if Select Targets by is set to From tags.

Variable Name

The name of the build variable containing the list of instance IDs to target, as a comma delimited list.

Note: you should specify just the variable name, do not enclose it in $() syntax.

This parameter is required if Select Targets by is set to Build variable name.

Execution Concurrency

The maximum number of instances that are allowed to execute the command at the same time. You can specify a number such as 10 or a percentage such as 10%. The default value is 50.

For more information about how to use MaxConcurrency, see Using Concurrency Controls.

Max Errors Before Stop

The maximum number of errors allowed without the command failing. When the command fails one more time beyond the value of MaxErrors, the systems stops sending the command to additional targets. You can specify a number like 10 or a percentage like 10%. The default value is 50.

For more information about how to use MaxErrors, see Using Error Controls.

Timeout (seconds)

If this time is reached and the command has not already started executing, it will not execute.

Minimum value of 30, maximum value of 2592000. Default value: 600.

Notification ARN

An Amazon Resource Name (ARN) for a Amazon SNS (SNS) topic. Run Command pushes notifications about command status changes to this topic.
Notification Events

The different events for which you can receive notifications. For more information see Setting Up Events and Notifications.

Notification Type

- **Command**: Receive notification when the status of a command changes.
- **Invocation**: For commands sent to multiple instances, receive notification on a per-instance basis when the status of a command changes.

S3 Bucket Name

The name of the Amazon S3 bucket where command execution responses should be stored.

S3 Key Prefix

The key prefix (folder structure) within the S3 bucket where the S3 objects containing the responses should be stored.

Command ID Output Variable

The name of a variable that will contain the unique ID assigned to the command. The command ID can be used future references to the request.

Task Permissions

This task requires permissions to call the following AWS service APIs (depending on selected task options, not all APIs may be used):

- ssm:SendCommand
Security for AWS Toolkit for Microsoft Azure DevOps

Cloud security at Amazon Web Services (AWS) is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations. Security is a shared responsibility between AWS and you. The Shared Responsibility Model describes this as Security of the Cloud and Security in the Cloud.

**Security of the Cloud** – AWS is responsible for protecting the infrastructure that runs all of the services offered in the AWS Cloud and providing you with services that you can use securely. Our security responsibility is the highest priority at AWS, and the effectiveness of our security is regularly tested and verified by third-party auditors as part of the AWS Compliance Programs.

**Security in the Cloud** – Your responsibility is determined by the AWS service you are using, and other factors including the sensitivity of your data, your organization's requirements, and applicable laws and regulations.

This AWS product or service follows the shared responsibility model through the specific Amazon Web Services (AWS) services it supports. For AWS service security information, see the AWS service security documentation page and AWS services that are in scope of AWS compliance efforts by compliance program.

**Topics**

- Data Protection in AWS Toolkit for Microsoft Azure DevOps (p. 74)
- Identity and Access Management for this AWS Product or Service (p. 75)
- Compliance Validation for this AWS Product or Service (p. 75)
- Resilience for this AWS Product or Service (p. 76)
- Infrastructure Security for this AWS Product or Service (p. 76)

Data Protection in AWS Toolkit for Microsoft Azure DevOps

The AWS shared responsibility model applies to data protection in AWS Toolkit for Microsoft Azure DevOps. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. This content includes the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the AWS Security Blog.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual user accounts with AWS Identity and Access Management (IAM). That way each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We recommend TLS 1.2 or later.
- Set up API and user activity logging with AWS CloudTrail.
• Use AWS encryption solutions, along with all default security controls within AWS services.
• Use advanced managed security services such as Amazon Macie, which assists in discovering and securing personal data that is stored in Amazon S3.
• If you require FIPS 140-2 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-2.

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form fields such as a Name field. This includes when you work with Toolkit for Azure DevOps or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

Identity and Access Management for this AWS Product or Service

AWS Identity and Access Management (IAM) is an Amazon Web Services (AWS) service that helps an administrator securely control access to AWS resources. IAM administrators control who can be authenticated (signed in) and authorized (have permissions) to use resources in AWS services. IAM is an AWS service that you can use with no additional charge.

To use this AWS product or service to access AWS, you need an AWS account and AWS credentials. To increase the security of your AWS account, we recommend that you use an IAM user to provide access credentials instead of using your AWS account credentials.

For details about working with IAM, see AWS Identity and Access Management.

For an overview of IAM users and why they are important for the security of your account, see AWS Security Credentials in the Amazon Web Services General Reference.

This AWS product or service follows the shared responsibility model through the specific Amazon Web Services (AWS) services it supports. For AWS service security information, see the AWS service security documentation page and AWS services that are in scope of AWS compliance efforts by compliance program.

Compliance Validation for this AWS Product or Service

This AWS product or service follows the shared responsibility model through the specific Amazon Web Services (AWS) services it supports. For AWS service security information, see the AWS service security documentation page and AWS services that are in scope of AWS compliance efforts by compliance program.

The security and compliance of AWS services is assessed by third-party auditors as part of multiple AWS compliance programs. These include SOC, PCI, FedRAMP, HIPAA, and others. AWS provides a frequently updated list of AWS services in scope of specific compliance programs at AWS Services in Scope by Compliance Program.

Third-party audit reports are available for you to download using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.
For more information about AWS compliance programs, see AWS Compliance Programs.

Your compliance responsibility when using this AWS product or service to access an AWS service is determined by the sensitivity of your data, your organization’s compliance objectives, and applicable laws and regulations. If your use of an AWS service is subject to compliance with standards such as HIPAA, PCI, or FedRAMP, AWS provides resources to help:

- **Security and Compliance Quick Start Guides** – Deployment guides that discuss architectural considerations and provide steps for deploying security-focused and compliance-focused baseline environments on AWS.
- **Architecting for HIPAA Security and Compliance Whitepaper** – A whitepaper that describes how companies can use AWS to create HIPAA-compliant applications.
- **AWS Compliance Resources** – A collection of workbooks and guides that might apply to your industry and location.
- **AWS Config** – A service that assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- **AWS Security Hub** – A comprehensive view of your security state within AWS that helps you check your compliance with security industry standards and best practices.

### Resilience for this AWS Product or Service

The Amazon Web Services (AWS) global infrastructure is built around AWS Regions and Availability Zones.

AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking.

With Availability Zones, you can design and operate applications and databases that automatically fail over between Availability Zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.

This AWS product or service follows the shared responsibility model through the specific Amazon Web Services (AWS) services it supports. For AWS service security information, see the AWS service security documentation page and AWS services that are in scope of AWS compliance efforts by compliance program.

### Infrastructure Security for this AWS Product or Service

This AWS product or service follows the shared responsibility model through the specific Amazon Web Services (AWS) services it supports. For AWS service security information, see the AWS service security documentation page and AWS services that are in scope of AWS compliance efforts by compliance program.
Document history for AWS Toolkit for Microsoft Azure DevOps

_Last documentation update:_ June 18, 2019

The following table describes important changes to the AWS Toolkit for Microsoft Azure DevOps over the course of its history. For notification about updates to this documentation, you can subscribe to an RSS feed.

<table>
<thead>
<tr>
<th>update-history-change</th>
<th>update-history-description</th>
<th>update-history-date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Added task to Lambda Deploy Function: Task Permissions (p. 46)</td>
<td>As of Azure DevOps Toolkit 1.12.0, it is now necessary to include permissions for the task: lambda.GetFunctionConfiguration.</td>
<td>December 1, 2021</td>
</tr>
<tr>
<td>Security Content (p. 74)</td>
<td>Added security content.</td>
<td>February 4, 2020</td>
</tr>
<tr>
<td>Refresh (p. 77)</td>
<td>Screenshots with relevant content and task names were refreshed for user-interface updates.</td>
<td>June 18, 2019</td>
</tr>
<tr>
<td>AWS CloudFormation Create/Update Stack Task (p. 27)</td>
<td>Added information about the option to log warning messages when no changes are reported.</td>
<td>March 29, 2019</td>
</tr>
<tr>
<td>Amazon Elastic Container Registry Push Image (p. 39)</td>
<td>Added permission requirements to the task.</td>
<td>July 25, 2018</td>
</tr>
<tr>
<td>AWS Systems Manager Set Parameter (p. 68)</td>
<td>Added the AWS Systems Manager Set Parameter task.</td>
<td>July 23, 2018</td>
</tr>
<tr>
<td>AWS Shell Script (p. 25)</td>
<td>Added the AWS Shell Script task.</td>
<td>July 23, 2018</td>
</tr>
<tr>
<td>AWS Secrets Manager Get Secret (p. 63)</td>
<td>Added the AWS Secrets Manager Get Secret task.</td>
<td>July 23, 2018</td>
</tr>
<tr>
<td>AWS Secrets Manager Create/Update Secret (p. 60)</td>
<td>Added the AWS Secrets Manager Create/Update Secret task.</td>
<td>July 23, 2018</td>
</tr>
<tr>
<td>AWS Elastic Beanstalk Create Version (p. 41)</td>
<td>Added the AWS Elastic Beanstalk Create Version task.</td>
<td>July 23, 2018</td>
</tr>
<tr>
<td>Amazon Elastic Container Registry Push Image (p. 39)</td>
<td>Added the Amazon Elastic Container Registry Push Image task.</td>
<td>November 28, 2017</td>
</tr>
<tr>
<td>AWS Systems Manager Run Command (p. 70)</td>
<td>Added the AWS Systems Manager Run Command task.</td>
<td>November 28, 2017</td>
</tr>
<tr>
<td>AWS Systems Manager Get Parameter (p. 66)</td>
<td>Added the AWS Systems Manager Get Parameter task.</td>
<td>November 28, 2017</td>
</tr>
<tr>
<td>AWS Lambda .NET Core Deployment task (p. 51)</td>
<td>Added the AWS Lambda .NET Core Deployment task.</td>
<td>November 28, 2017</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>Initial Release (p. 77)</td>
<td>Initial release of SDK developer guide for AWS Toolkit for Microsoft Azure DevOps.</td>
<td>August 14, 2017</td>
</tr>
</tbody>
</table>