AWS CloudFormation

User Guide

API Version 2010-05-15

AWS CloudFormation: User Guide

Table of Contents

What is AWS CloudFormation?

AWS CloudFormation is a service that helps you model and set up your Amazon Web Services resources so that you can spend less time managing those resources and more time focusing on your applications that run in AWS. You create a template that describes all the AWS resources that you want (like Amazon EC2 instances or Amazon RDS DB instances), and AWS CloudFormation takes care of provisioning and configuring those resources for you. You don't need to individually create and configure AWS resources and figure out what's dependent on what; AWS CloudFormation handles all of that. The following scenarios demonstrate how AWS CloudFormation can help.

Simplify Infrastructure Management

For a scalable web application that also includes a back-end database, you might use an Auto Scaling group, an Elastic Load Balancing load balancer, and an Amazon Relational Database Service database instance. Normally, you might use each individual service to provision these resources. And after you create the resources, you would have to configure them to work together. All these tasks can add complexity and time before you even get your application up and running.

Instead, you can create or modify an existing AWS CloudFormation template. A template describes all of your resources and their properties. When you use that template to create an AWS CloudFormation stack, AWS CloudFormation provisions the Auto Scaling group, load balancer, and database for you. After the stack has been successfully created, your AWS resources are up and running. You can delete the stack just as easily, which deletes all the resources in the stack. By using AWS CloudFormation, you easily manage a collection of resources as a single unit.

Quickly Replicate Your Infrastructure

If your application requires additional availability, you might replicate it in multiple regions so that if one region becomes unavailable, your users can still use your application in other regions. The challenge in replicating your application is that it also requires you to replicate your resources. Not only do you need to record all the resources that your application requires, but you must also provision and configure those resources in each region.

When you use AWS CloudFormation, you can reuse your template to set up your resources consistently and repeatedly. Just describe your resources once and then provision the same resources over and over in multiple regions.

Easily Control and Track Changes to Your Infrastructure

In some cases, you might have underlying resources that you want to upgrade incrementally. For example, you might change to a higher performing instance type in your Auto Scaling launch configuration so that you can reduce the maximum number of instances in your Auto Scaling group. If problems occur after you complete the update, you might need to roll back your infrastructure to the original settings. To do this manually, you not only have to remember which resources were changed, you also have to know what the original settings were.

When you provision your infrastructure with AWS CloudFormation, the AWS CloudFormation template describes exactly what resources are provisioned and their settings. Because these templates are text files, you simply track differences in your templates to track changes to your infrastructure, similar to the way developers control revisions to source code. For example, you can use a version control system with your templates so that you know exactly what changes were made, who made them, and when. If at any point you need to reverse changes to your infrastructure, you can use a previous version of your template.

Related Information

- For more information about AWS CloudFormation stacks and templates, see AWS CloudFormation Concepts (p. 2).
- For an overview about how to use AWS CloudFormation, see How Does AWS CloudFormation Work? (p. 3).
- For pricing information, see AWS CloudFormation Pricing.

AWS CloudFormation Concepts

When you use AWS CloudFormation, you work with *templates* and *stacks*. You create templates to describe your AWS resources and their properties. Whenever you create a stack, AWS CloudFormation provisions the resources that are described in your template.

Topics

Templates

An AWS CloudFormation template is a JSON or YAML formatted text file. You can save these files with any extension, such as `.json`, `.yaml`, `.template`, or `.txt`. AWS CloudFormation uses these templates as blueprints for building your AWS resources. For example, in a template, you can describe an Amazon EC2 instance, such as the instance type, the AMI ID, block device mappings, and its Amazon EC2 key pair name. Whenever you create a stack, you also specify a template that AWS CloudFormation uses to create whatever you described in the template.

For example, if you created a stack with the following template, AWS CloudFormation provisions an instance with an `ami-2f726546` AMI ID, `t1.micro` instance type, `testkey` key pair name, and an Amazon EBS volume.

You can also specify multiple resources in a single template and configure these resources to work together. For example, you can modify the previous template to include an Elastic IP (EIP) and associate it with the Amazon EC2 instance, as shown in the following example:

The previous templates are centered around a single Amazon EC2 instance; however, AWS CloudFormation templates have additional capabilities that you can use to build complex sets of resources and reuse those templates in multiple contexts. For example, you can add input parameters whose values are specified when you create an AWS CloudFormation stack. In other words, you can specify a value like the instance type when you create a stack instead of when you create the template, making the template easier to reuse in different situations.

For more information about template creation and capabilities, see Template Anatomy (p. 128).

For more information about declaring specific resources, see AWS Resource Types Reference (p. 390).

To start designing your own templates with AWS CloudFormation Designer, go to https://console.aws.amazon.com/cloudformation/designer.

Stacks

When you use AWS CloudFormation, you manage related resources as a single unit called a stack. You create, update, and delete a collection of resources by creating, updating, and deleting stacks. All the resources in a stack are defined by the stack's AWS CloudFormation template. Suppose you created a template that includes an Auto Scaling group, Elastic Load Balancing load balancer, and an Amazon Relational Database Service (Amazon RDS) database instance. To create those resources, you create a stack by submitting the template that you created, and AWS CloudFormation provisions all those resources for you. You can work with stacks by using the AWS CloudFormation console, API, or AWS CLI.

For more information about creating, updating, or deleting stacks, see Working with Stacks (p. 71).

Change Sets

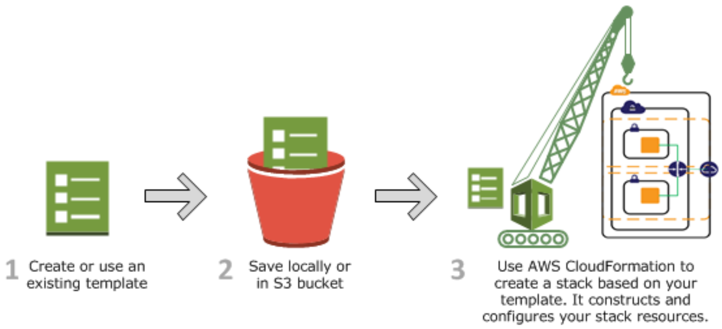
If you need to make changes to the running resources in a stack, you update the stack. Before making changes to your resources, you can generate a change set, which is summary of your proposed changes. Change sets allow you to see how your changes might impact your running resources, especially for critical resources, before implementing them.

For example, if you change the name of an Amazon RDS database instance, AWS CloudFormation will create a new database and delete the old one. You will lose the data in the old database unless you've already backed it up. If you generate a change set, you will see that your change will cause your database to be replaced, and you will be able to plan accordingly before you update your stack. For more information, see Updating Stacks Using Change Sets (p. 93).

How Does AWS CloudFormation Work?

When you create a stack, AWS CloudFormation makes underlying service calls to AWS to provision and configure your resources. Note that AWS CloudFormation can perform only actions that you have permission to do. For example, to create EC2 instances by using AWS CloudFormation, you need permissions to create instances. You'll need similar permissions to terminate instances when you delete stacks with instances. You use AWS Identity and Access Management (IAM) to manage permissions.

The calls that AWS CloudFormation makes are all declared by your template. For example, suppose you have a template that describes an EC2 instance with a `t1.micro` instance type. When you use that template to create a stack, AWS CloudFormation calls the Amazon EC2 create instance API and specifies the instance type as `t1.micro`. The following diagram summarizes the AWS CloudFormation workflow for creating stacks.



1. You can design an AWS CloudFormation template (a JSON or YAML-formatted document) in AWS CloudFormation Designer or write one in a text editor. You can also choose to use a provided template. The template describes the resources you want and their settings. For example, suppose you want to create an EC2 instance. Your template can declare an EC2 instance and describe its properties, as shown in the following example:
2. Save the template locally or in an S3 bucket. If you created a template, save it with any file extension like `.json`, `.yaml`, or `.txt`.
3. Create an AWS CloudFormation stack by specifying the location of your template file , such as a path on your local computer or an Amazon S3 URL. If the template contains parameters, you can specify input values when you create the stack. Parameters enable you to pass in values to your template so that you can customize your resources each time you create a stack.

   You can create stacks by using the AWS CloudFormation console (p. 72), API, or AWS CLI.

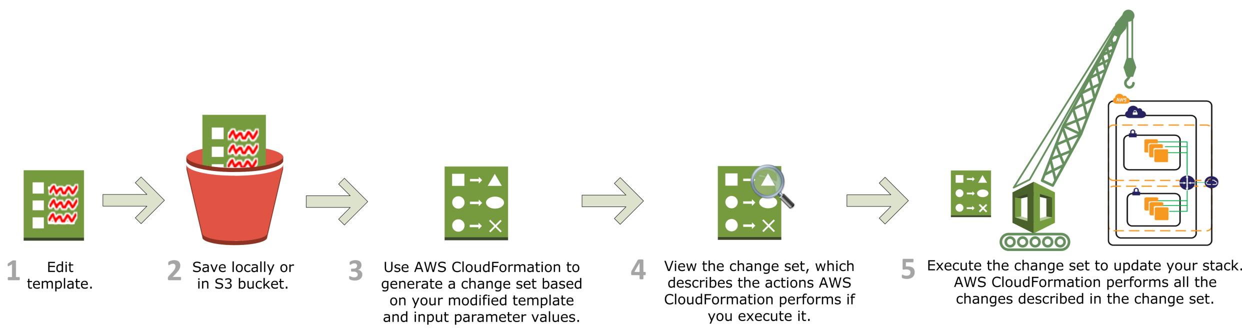
   Note
   If you specify a template file stored locally, AWS CloudFormation uploads it to an S3 bucket in your AWS account. AWS CloudFormation creates a bucket for each region in which you upload a template file. The buckets are accessible to anyone with Amazon Simple Storage Service (Amazon S3) permissions in your AWS account. If a bucket created by AWS CloudFormation is already present, the template is added to that bucket.
   You can use your own bucket and manage its permissions by manually uploading templates to Amazon S3. Then whenever you create or update a stack, specify the Amazon S3 URL of a template file.

AWS CloudFormation provisions and configures resources by making calls to the AWS services that are described in your template.

After all the resources have been created, AWS CloudFormation reports that your stack has been created. You can then start using the resources in your stack. If stack creation fails, AWS CloudFormation rolls back your changes by deleting the resources that it created.

Updating a Stack with Change Sets

When you need to update your stack's resources, you can modify the stack's template. You don't need to create a new stack and delete the old one. To update a stack, create a change set by submitting a modified version of the original stack template, different input parameter values, or both. AWS CloudFormation compares the modified template with the original template and generates a change set. The change set lists the proposed changes. After reviewing the changes, you can execute the change set to update your stack or you can create a new change set. The following diagram summarizes the workflow for updating a stack.



Important
Updates can cause interruptions. Depending on the resource and properties that you are updating, an update might interrupt or even replace an existing resource. For more information, see AWS CloudFormation Stacks Updates (p. 89).

1. You can modify an AWS CloudFormation stack template by using AWS CloudFormation Designer or a text editor. For example, if you want to change the instance type for an EC2 instance, you would change the value of the `InstanceType` property in the original stack's template.

   For more information, see Modifying a Stack Template (p. 90).
2. Save the AWS CloudFormation template locally or in an S3 bucket.
3. Create a change set by specifying the stack that you want to update and the location of the modified template, such as a path on your local computer or an Amazon S3 URL. If the template contains parameters, you can specify values when you create the change set.

   For more information about creating change sets, see Updating Stacks Using Change Sets (p. 93).

   > **Note**
   > If you specify a template that is stored on your local computer, AWS CloudFormation automatically uploads your template to an S3 bucket in your AWS account.
4. View the change set to check that AWS CloudFormation will perform the changes that you expect. For example, check whether AWS CloudFormation will replace any critical stack resources. You can create as many change sets as you need until you have included the changes that you want.

   > **Important**
   > Change sets don't indicate whether your stack update will be successful. For example, a change set doesn't check if you will surpass an account limit (p. 16), if you're updating a resource (p. 390) that doesn't support updates, or if you have insufficient permissions (p. 8) to modify a resource, all of which can cause a stack update to fail.
5. Execute the change set that you want to apply to your stack. AWS CloudFormation updates your stack by updating only the resources that you modified and signals that your stack has been successfully updated. If the stack updates fails, AWS CloudFormation rolls back changes to restore the stack to the last known working state.

Deleting a Stack

When you delete a stack, you specify the stack to delete, and AWS CloudFormation deletes the stack and all the resources in that stack. You can delete stacks by using the AWS CloudFormation console (p. 79), API, or AWS CLI.

If you want to delete a stack but want to retain some resources in that stack, you can use a deletion policy (p. 1297) to retain those resources.

After all the resources have been deleted, AWS CloudFormation signals that your stack has been successfully deleted. If AWS CloudFormation cannot delete a resource, the stack will not be deleted. Any resources that haven't been deleted will remain until you can successfully delete the stack.

Additional Resources

- For more information about creating AWS CloudFormation templates, see Template Anatomy (p. 128).
- For more information about creating, updating, or deleting stacks, see Working with Stacks (p. 71).

Setting Up

Before you start using AWS CloudFormation, you might need to know what IAM permissions you need, how to start logging AWS CloudFormation API calls, or what endpoints to use. The following topics provide this information so that you can start using AWS CloudFormation.

Topics

Signing Up for an AWS Account and Pricing

Before you can use AWS CloudFormation or any Amazon Web Services, you must first sign up for an AWS account.

To sign up for an AWS account

1. Open https://aws.amazon.com/, and then choose **Create an AWS Account**.
2. Follow the online instructions.

   Part of the sign-up procedure involves receiving a phone call and entering a PIN using the phone keypad.

After signing up for an AWS account, you can use AWS CloudFormation through the AWS Management Console, AWS CloudFormation API, or AWS CLI.

Pricing

AWS CloudFormation is a free service; however, you are charged for the AWS resources you include in your stacks at the current rates for each. For more information about AWS pricing, go to the detail page for each product on http://aws.amazon.com.

Controlling Access with AWS Identity and Access Management

With AWS Identity and Access Management (IAM), you can create IAM users to control who has access to which resources in your AWS account. You can use IAM with AWS CloudFormation to control what users can do with AWS CloudFormation, such as whether they can view stack templates, create stacks, or delete stacks.

In addition to AWS CloudFormation actions, you can manage what AWS services and resources are available to each user. That way, you can control which resources users can access when they use AWS CloudFormation. For example, you can specify which users can create Amazon EC2 instances, terminate database instances, or update VPCs. Those same permissions are applied anytime they use AWS CloudFormation to do those actions.

For more information about all the services that you can control access to, see AWS Services that Support IAM in *IAM User Guide*.

Topics

AWS CloudFormation Actions

When you create a group or an IAM user in your AWS account, you can associate an IAM policy with that group or user, which specifies the permissions that you want to grant. For example, imagine you have a group of entry-level developers. You can create a `Junior application developers` group that includes all entry-level developers. Then, you associate a policy with that group that allows users to only view AWS CloudFormation stacks. In this scenario, you might have a policy such as the following sample:

The policy grants permissions to all `DescribeStack` API actions listed in the `Action` element.

> **Note**
> If you don't specify a stack name or ID in your statement, you must also grant the permission to use all resources for the action using the `*` wildcard for the `Resource` element.

In addition to AWS CloudFormation actions, IAM users who create or delete stacks require additional permissions that depends on the stack templates. For example, if you have a template that describes an Amazon SQS Queue, the user must have the corresponding permissions for Amazon SQS actions to successfully create the stack, as shown in the following sample policy:

For a list of all AWS CloudFormation actions that you can allow or deny, see the *AWS CloudFormation API Reference*.

AWS CloudFormation Console-Specific Actions

IAM users who use the AWS CloudFormation console require additional permissions that are not required for using the AWS Command Line Interface or AWS CloudFormation APIs. Compared to the CLI and API, the console provides additional features that require additional permissions, such as template uploads to Amazon S3 buckets and drop-down lists for AWS-specific parameter types (p. 136).

For all the following actions, grant permissions to all resources; don't limit actions to specific stacks or buckets.

The following required action is used only by the AWS CloudFormation console and is not documented in the API reference. The action allows users to upload templates to Amazon S3 buckets.

```
cloudformation:CreateUploadBucket
```

When users upload templates,they require the following Amazon S3 permissions:

```
s3:PutObject
s3:ListBucket
s3:GetObject
s3:CreateBucket
```

For templates with AWS-specific parameter types (p. 136), users need permissions to make the corresponding describe API calls. For example, if a template includes the `AWS::EC2::KeyPair::KeyName` parameter type, users need permission to call the EC2 `DescribeKeyPairs` action (this is how the console gets values for the parameter drop-down list). The following examples are actions that users need for other parameter types:

```
ec2:DescribeSecurityGroups (for the AWS::EC2::SecurityGroup::Id parameter type)
ec2:DescribeSubnets (for the Subnet::Id parameter type)
ec2:DescribeVpcs (for the AWS::EC2::VPC::Id parameter type)
```

AWS CloudFormation Resources

AWS CloudFormation supports resource-level permissions, so you can specify actions for a specific stack, as shown in the following policy:

The sample policy uses a wild card at the end of the stack name so that delete stack and update stack are denied on the full stack ID (such as `arn:aws:cloudformation:us-east-1:123456789012:stack/MyProductionStack/abc9dbf0-43c2-11e3-a6e8-50fa526be49c`) and on the stack name (such as `MyProductionStack`).

Currently, only AWS CloudFormation stack IDs are supported in the `Resource` element.

AWS CloudFormation Conditions

In an IAM policy, you can optionally specify conditions that control when a policy is in effect. For example, you can define a policy that allows IAM users to create a stack only when they specify a certain template URL. You can define AWS CloudFormation-specific conditions and AWS-wide conditions, such as `DateLessThan`, which specifies when a policy stops taking effect. For more information and a list of AWS-wide conditions, see Condition in IAM Policy Elements Reference in *IAM User Guide*.

> **Note**
> Do not use the `aws:SourceIp` AWS-wide condition. AWS CloudFormation provisions resources by using its own IP address, not the IP address of the originating request. For example, when you create a stack, AWS CloudFormation makes requests from its IP address to launch an EC2 instance or to create an S3 bucket, not from the IP address from the `CreateStack` call or the `aws cloudformation create-stack` command.

The following list describes the AWS CloudFormation-specific conditions. These conditions are applied only when users create or update stacks:

`cloudformation:ChangeSetName`

An AWS CloudFormation change set name that you want to associate with a policy. Use this condition to control which change sets IAM users can execute or delete.

`cloudformation:ResourceTypes`

The template resource types, such as `AWS::EC2::Instance`, that you want to associate with a policy. Use this condition to control which resource types IAM users can work with when they create or update a stack. This condition is checked against the resource types that users declare in the `ResourceTypes` parameter, which is currently supported only for CLI and API requests. When using this parameter, users must specify all the resource types that are in their template. For more information about the `ResourceTypes` parameter, see the CreateStack action in the *AWS CloudFormation API Reference*.

The following list describes how to define resource types. For a list of resource types, see AWS Resource Types Reference (p. 390).

`AWS::*`

Specify all AWS resources.

`AWS::`*`service_name`*`::*`

Specify all resources for a specific AWS service.

`AWS::`*`service_name`*`::`*`resource_type`*

Specify a specific AWS resource type, such as `AWS::EC2::Instance` (all EC2 instances).

`Custom::*`

Specify all custom resources.

`Custom::`*`resource_type`*

Specify a specific custom resource type, which is defined in the template.

`cloudformation:RoleARN`

The Amazon Resource Name (ARN) of an IAM service role that you want to associate with a policy. Use this condition to control which service role IAM users can use when they work with stacks or change sets.

`cloudformation:StackPolicyUrl`

An Amazon S3 stack policy URL that you want to associate with a policy. Use this condition to control which stack policies IAM users can associate with a stack during a create or update stack action. For more information about stack policies, see Prevent Updates to Stack Resources (p. 110).

Note
To ensure that IAM users can only create or update stacks with the stack policies that you uploaded, set the S3 bucket to `read only` for those users.

`cloudformation:TemplateUrl`

An Amazon S3 template URL that you want to associate with a policy. Use this condition to control which templates IAM users can use when they create or update stacks.

Note
To ensure that IAM users can only create or update stacks with the templates that you uploaded, set the S3 bucket to `read only` for those users.

Examples

The following example policy allows users to use only the `https://s3.amazonaws.com/testbucket/test.template` template URL to create or update a stack.

The following example policy allows users to create stacks but denies requests if the stack's template include any resource from the IAM service. The policy also requires users to specify the `ResourceTypes` parameter, which is available only for CLI and API requests. This policy uses explicit deny statements so that if any other policy grants additional permissions, this policy always remain in effect (an explicit deny statement always overrides an explicit allow statement).

The following example policy is similar to the preceding example. The policy allows users to create a stack unless the stack's template includes any resource from the IAM service. It also requires users to specify the `ResourceTypes` parameter, which is available only for CLI and API requests. This policy is simpler, but it doesn't use explicit deny statements. Other policies, granting additional permissions, could override this policy.

Acknowledging IAM Resources in AWS CloudFormation Templates

Before you can create a stack, AWS CloudFormation validates your template. During validation, AWS CloudFormation checks your template for IAM resources that it might create. IAM resources, such as an IAM user with full access, can access and modify any resource in your AWS account. Therefore, we recommend that you review the permissions associated with each IAM resource before proceeding so that you don't unintentionally create resources with escalated permissions. To ensure that you've done so, you must acknowledge that the template contains those resources, giving AWS CloudFormation the specified capabilities before it creates the stack.

You can acknowledge the capabilities of AWS CloudFormation templates by using the AWS CloudFormation console, AWS Command Line Interface (CLI), or API:

- In the AWS CloudFormation console, on the **Review** page of the Create Stack or Update Stack wizards, choose **I acknowledge that this template may create IAM resources**.
- In the CLI, when you use the `aws cloudformation create-stack` and `aws cloudformation update-stack` commands, specify the `CAPABILITY_IAM` or `CAPABILITY_NAMED_IAM` value for the `--capabilities` parameter. If your template includes IAM resources, you can specify either capability. If your template includes custom names for IAM resources, you must specify `CAPABILITY_NAMED_IAM`.
- In the API, when you use the CreateStack and UpdateStack actions, specify `Capabilities.member.1=CAPABILITY_IAM` or `Capabilities.member.1=CAPABILITY_NAMED_IAM`. If your template includes IAM resources, you can specify either capability. If your template includes custom names for IAM resources, you must specify `CAPABILITY_NAMED_IAM`.

Important
If your template contains custom named IAM resources, don't create multiple stacks reusing the same template. IAM resources must be globally unique within your account. If you use the same template to create multiple stacks in different regions, your stacks might share the same IAM resources, instead of each having a unique one. Share resources among stacks can have unintended consequences from which you can't recover. For example, if you delete or update shared IAM resources in one stack, you will unintentionally modify the resources of other stacks.

Manage Credentials for Applications Running on Amazon EC2 Instances

If you have an application that runs on an Amazon EC2 instance and needs to make requests to AWS resources such as Amazon S3 buckets or an DynamoDB table, the application requires AWS security credentials. However, distributing and embedding long-term security credentials in every instance that you launch is a challenge and a potential security risk. Instead of using long-term credentials, like IAM user credentials, we recommend that you create an IAM role that is associated with an Amazon EC2 instance when the instance is launched. An application can then get temporary security credentials from the Amazon EC2 instance. You don't have to embed long-term credentials on the instance. Also, to make managing credentials easier, you can specify just a single role for multiple Amazon EC2 instances; you don't have to create unique credentials for each instance.

For a template snippet that shows how to launch an instance with a role, see IAM Role Template Examples (p. 326).

Note

Applications on instances that use temporary security credentials can call any AWS CloudFormation actions. However, because AWS CloudFormation interacts with many other AWS services, you must verify that all the services that you want to use support temporary security credentials. For more information, see AWS Services that Support AWS STS.

Grant Temporary Access (Federated Access)

In some cases, you might want to grant users with no AWS credentials temporary access to your AWS account. Instead of creating and deleting long-term credentials whenever you want to grant temporary access, use AWS Security Token Service (AWS STS). For example, you can use IAM roles. From one IAM role, you can programmatically create and then distribute many temporary security credentials (which include an access key, secret access key, and security token). These credentials have a limited life, so they cannot be used to access your AWS account after they expire. You can also create multiple IAM roles in order to grant individual users different levels of permissions. IAM roles are useful for scenarios like federated identities and single sign-on.

A federated identity is a distinct identity that you can use across multiple systems. For enterprise users with an established on-premises identity system (such as LDAP or Active Directory), you can handle all authentication with your on-premises identity system. After a user has been authenticated, you provide temporary security credentials from the appropriate IAM user or role. For example, you can create an `administrators` role and a `developers` role, where administrators have full access to the AWS account and developers have permissions to work only with AWS CloudFormation stacks. After an administrator is authenticated, the administrator is authorized to obtain temporary security credentials from the `administrators` role. However, for developers, they can obtain temporary security credentials from only the `developers` role.

You can also grant federated users access to the AWS Management Console. After users authenticate with your on-premises identity system, you can programmatically construct a temporary URL that gives direct access to the AWS Management Console. When users use the temporary URL, they won't need to sign in to AWS because they have already been authenticated (single sign-on). Also, because the URL is constructed from the users' temporary security credentials, the permissions that are available with those credentials determine what permissions users have in the AWS Management Console.

You can use several different AWS STS APIs to generate temporary security credentials. For more information about which API to use, see Ways to Get Temporary Security Credentials in *Using Temporary Security Credentials*.

Important

You cannot work with IAM when you use temporary security credentials that were generated from the `GetFederationToken` API. Instead, if you need to work with IAM, use temporary security credentials from a role.

AWS CloudFormation interacts with many other AWS services. When you use temporary security credentials with AWS CloudFormation, verify that all the services that you want to use support temporary security credentials. For more information, see AWS Services that Support AWS STS.

For more information, see the following related resources in *Using Temporary Security Credentials*:

- Scenarios for Granting Temporary Access
- Giving Federated Users Direct Access to the AWS Management Console

AWS CloudFormation Service Role

A *service role* is an AWS Identity and Access Management (IAM) role that allows AWS CloudFormation to make calls to resources in a stack on your behalf. You can specify an IAM role that allows AWS CloudFormation to create, update, or delete your stack resources. By default, AWS CloudFormation uses a temporary session that it generates from your user credentials for stack operations. If you specify a service role, AWS CloudFormation uses the role's credentials.

Use a service role to explicitly specify the actions that AWS CloudFormation can perform which might not always be the same actions that you or other users can do. For example, you might have administrative privileges, but you can limit AWS CloudFormation access to only Amazon EC2 actions.

You create the service role and its permission policy with the IAM service. For more information about creating a service role, see Creating a Role to Delegate Permissions to an AWS Service in the *IAM User Guide*. Specify AWS CloudFormation (`cloudformation.amazonaws.com`) as the service that can assume the role.

To associate a service role with a stack, specify the role when you create the stack. For details, see Setting Stack Options (p. 76). You can also change the service role when you update (p. 89) or delete the stack. Before you specify a service role, ensure that you have permission to pass it (`iam:PassRole`). The `iam:PassRole` permission specifies which roles you can use.

> **Important**
> When you specify a service role, AWS CloudFormation always uses that role for all operations that are performed on that stack. Other users that have permissions to perform operations on this stack will be able to use this role, even if they don't have permission to pass it. If the role includes permissions that the user shouldn't have, you can unintentionally escalate a user's permissions. Ensure that the role grants least privilege.

Logging AWS CloudFormation API Calls in AWS CloudTrail

AWS CloudFormation is integrated with AWS CloudTrail, a service that captures API calls made by or on behalf of your AWS account. This information is collected and written to log files that are stored in an Amazon S3 bucket that you specify. API calls are logged when you use the AWS CloudFormation API, the AWS CloudFormation console, a back-end console, or the AWS CLI. Using the information collected by CloudTrail, you can determine what request was made to AWS CloudFormation, the source IP address the request was made from, who made the request, when it was made, and so on.

To learn more about CloudTrail, including how to configure and enable it, see the *AWS CloudTrail User Guide*.

Topics

AWS CloudFormation Information in CloudTrail

If CloudTrail logging is turned on, calls made to all AWS CloudFormation actions are captured in log files. All the AWS CloudFormation actions are documented in the *AWS CloudFormation API Reference*. For example, calls to the **CreateStack**, **DeleteStack**, and **ListStacks** actions generate entries in CloudTrail log files.

Every log entry contains information about who generated the request. For example, if a request is made to list AWS CloudFormation stacks (**ListStacks**), CloudTrail logs the user identity of the person or service that made the request. The user identity information helps you determine whether the request was made with root or IAM user credentials, with temporary security credentials for a role or federated user, or by another AWS service. For more information about CloudTrail fields, see CloudTrail Event Reference in the *AWS CloudTrail User Guide*.

You can store your log files in your bucket for as long as you want, but you can also define Amazon S3 lifecycle rules to archive or delete log files automatically. By default, your log files are encrypted by using Amazon S3 server-side encryption (SSE).

Understanding AWS CloudFormation Log File Entries

CloudTrail log files can contain one or more log entries composed of multiple JSON-formatted events. A log entry represents a single request from any source and includes information about the requested action, any input parameters, the date and time of the action, and so on. The log entries do not appear in any particular order. That is, they do not represent an ordered stack trace of the public API calls.

The following example record shows a CloudTrail log entry for the **CreateStack** action. The action was made by an IAM user named Alice.

Note

Only the input parameter key names are logged; no parameter values are logged.


```
{
  "eventVersion": "1.01",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAABCDEFGHIJKLNMOPQ",
    "arn": "arn:aws:iam::012345678910:user/Alice",
    "accountId": "012345678910",
    "accessKeyId": "AKIDEXAMPLE",
    "userName": "Alice"
  },
  "eventTime": "2014-03-24T21:02:43Z",
  "eventSource": "cloudformation.amazonaws.com",
  "eventName": "CreateStack",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "127.0.0.1",
  "userAgent": "aws-cli/1.2.11 Python/2.7.4 Linux/2.6.18-164.el5",
  "requestParameters": {
    "templateURL": "https://s3.amazonaws.com/Alice-dev/create_stack",
    "tags": [
      {
        "key": "test",
        "value": "tag"
      }
    ],
    "stackName": "my-test-stack",
    "disableRollback": true,
    "parameters": [
      {
        "parameterKey": "password"
      },
      {
        "parameterKey": "securitygroup"
      }
    ]
  },
  "responseElements": {
    "stackId": "arn:aws:cloudformation:us-east-1:012345678910:stack/my-test-stack/a38e6a60-
b397-11e3-b0fc-08002755629e"
  },
```

```
  "requestID": "9f960720-b397-11e3-bb75-a5b75389b02d",
  "eventID": "9bf6cfb8-83e1-4589-9a70-b971e727099b"
}
```


The following sample record shows that Alice called the **UpdateStack** action on the `my-test-stack` stack:


```
{
  "eventVersion": "1.01",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAABCDEFGHIJKLNMOPQ",
    "arn": "arn:aws:iam::012345678910:user/Alice",
    "accountId": "012345678910",
    "accessKeyId": "AKIDEXAMPLE",
    "userName": "Alice"
  },
  "eventTime": "2014-03-24T21:04:29Z",
  "eventSource": "cloudformation.amazonaws.com",
  "eventName": "UpdateStack",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "127.0.0.1",
  "userAgent": "aws-cli/1.2.11 Python/2.7.4 Linux/2.6.18-164.el5",
  "requestParameters": {
    "templateURL": "https://s3.amazonaws.com/Alice-dev/create_stack",
    "parameters": [
      {
        "parameterKey": "password"
      },
      {
        "parameterKey": "securitygroup"
      }
    ],
    "stackName": "my-test-stack"
  },
  "responseElements": {
    "stackId": "arn:aws:cloudformation:us-east-1:012345678910:stack/my-test-stack/a38e6a60-
b397-11e3-b0fc-08002755629e"
  },
  "requestID": "def0bf5a-b397-11e3-bb75-a5b75389b02d",
  "eventID": "637707ce-e4a3-4af1-8edc-16e37e851b17"
}
```


The following sample record shows that Alice called the **ListStacks** action.


```
{
  "eventVersion": "1.01",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAABCDEFGHIJKLNMOPQ",
    "arn": "arn:aws:iam::012345678910:user/Alice",
    "accountId": "012345678910",
    "accessKeyId": "AKIDEXAMPLE",
    "userName": "Alice"
  },
  "eventTime": "2014-03-24T21:03:16Z",
  "eventSource": "cloudformation.amazonaws.com",
  "eventName": "ListStacks",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "127.0.0.1",
  "userAgent": "aws-cli/1.2.11 Python/2.7.4 Linux/2.6.18-164.el5",
  "requestParameters": null,
  "responseElements": null,
  "requestID": "b7d351d7-b397-11e3-bb75-a5b75389b02d",
  "eventID": "918206d0-7281-4629-b778-b91eb0d83ce5"
```

```
}
```

The following sample record shows that Alice called the **DescribeStacks** action on the `my-test-stack` stack.


```
{
  "eventVersion": "1.01",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAABCDEFGHIJKLNMOPQ",
    "arn": "arn:aws:iam::012345678910:user/Alice",
    "accountId": "012345678910",
    "accessKeyId": "AKIDEXAMPLE",
    "userName": "Alice"
  },
  "eventTime": "2014-03-24T21:06:15Z",
  "eventSource": "cloudformation.amazonaws.com",
  "eventName": "DescribeStacks",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "127.0.0.1",
  "userAgent": "aws-cli/1.2.11 Python/2.7.4 Linux/2.6.18-164.el5",
  "requestParameters": {
    "stackName": "my-test-stack"
  },
  "responseElements": null,
  "requestID": "224f2586-b398-11e3-bb75-a5b75389b02d",
  "eventID": "9e5b2fc9-1ba8-409b-9c13-587c2ea940e2"
}
```


The following sample record shows that Alice called the **DeleteStack** action on the `my-test-stack` stack.


```
{
  "eventVersion": "1.01",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAABCDEFGHIJKLNMOPQ",
    "arn": "arn:aws:iam::012345678910:user/Alice",
    "accountId": "012345678910",
    "accessKeyId": "AKIDEXAMPLE",
    "userName": "Alice"
  },
  "eventTime": "2014-03-24T21:07:15Z",
  "eventSource": "cloudformation.amazonaws.com",
  "eventName": "DeleteStack",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "127.0.0.1",
  "userAgent": "aws-cli/1.2.11 Python/2.7.4 Linux/2.6.18-164.el5",
  "requestParameters": {
    "stackName": "my-test-stack"
  },
  "responseElements": null,
  "requestID": "42dae739-b398-11e3-bb75-a5b75389b02d",
  "eventID": "4965eb38-5705-4942-bb7f-20ebe79aa9aa"
}
```


AWS CloudFormation Limits

Your AWS account has AWS CloudFormation limits that you might need to know when authoring templates and creating stacks. By understanding these limits, you can avoid limitation errors that would require you to redesign your templates or stacks.

AWS CloudFormation limits

Limit	Description	Value	Tuning Strategy
cfn-signal wait condition data (p. 1358)	Maximum amount of data that cfn-signal can pass.	4,096 bytes	To pass a larger amount, send the data to an Amazon S3 bucket, and then use cfn-signal to pass the Amazon S3 URL to that bucket.
Custom resource response (p. 467)	Maximum amount of data that a custom resource provider can pass.	4,096 bytes	
Mappings (p. 128)	Maximum number of mappings that you can declare in your AWS CloudFormation template.	100 mappings	To specify more mappings, separate your template into multiple templates by using, for example, nested stacks (p. 485).
Mapping attributes (p. 128)	Maximum number of mapping attributes for each mapping that you can declare in your AWS CloudFormation template.	64 attributes	To specify more mapping attributes, separate the attributes into multiple mappings.
Mapping name and mapping attribute name (p. 128)	Maximum size of each mapping name.	255 characters	
Outputs (p. 128)	Maximum number of outputs that you can declare in your AWS CloudFormation template.	60 outputs	
Output name (p. 128)	Maximum size of an output name.	255 characters	
Parameters (p. 128)	Maximum number of parameters that you can declare in your AWS CloudFormation template.	60 parameters	To specify more parameters, you can use mappings or lists in order to assign multiple values to a single parameter.
Parameter name (p. 128)	Maximum size of a parameter name.	255 characters	
Parameter value (p. 128)	Maximum size of a parameter value.	4,096 bytes	To use a larger parameter value, create multiple parameters and then use `Fn::Join` to append the multiple

Limit	Description	Value	Tuning Strategy
			values into a single value.
Resources (p. 128)	Maximum number of resources that you can declare in your AWS CloudFormation template.	200 resources	To specify more resources, separate your template into multiple templates by using, for example, nested stacks (p. 485).
Resource name (p. 128)	Maximum size of a resource name.	255 characters	
Stacks (p. 71)	Maximum number of AWS CloudFormation stacks that you can create.	200 stacks	To create more stacks, delete stacks that you don't need or request an increase in the maximum number of stacks in your AWS account. For more information, see AWS Service Limits in the *AWS General Reference*.
Template body size in a request (p. 128)	Maximum size of a template body that you can pass in a `CreateStack`, `UpdateStack`, or `ValidateTemplate` request.	51,200 bytes	To use a larger template body, separate your template into multiple templates by using, for example, nested stacks (p. 485). Or upload the template to an Amazon S3 bucket.
Template body size in an Amazon S3 object (p. 128)	Maximum size of a template body that you can pass in an Amazon S3 object for a `CreateStack`, `UpdateStack`, `ValidateTemplate` request with an Amazon S3 template URL.	460,800 bytes	To use a larger template body, separate your template into multiple templates by using, for example, nested stacks (p. 485).
Template description (p. 128)	Maximum size of a template description.	1,024 bytes	

AWS CloudFormation Endpoints

To reduce data latency in your applications, most Amazon Web Services products allow you to select a regional endpoint to make your requests. An endpoint is a URL that is the entry point for a web service.

When you work with stacks by using the command line interface or API actions, you can specify a regional endpoint. For more information about the regions and endpoints for AWS CloudFormation, see Regions and Endpoints in the *Amazon Web Services General Reference*.

AWS CloudFormation and VPC Endpoints

You can use a VPC endpoint to create a private connection between your VPC and another AWS service without requiring access over the Internet, through a NAT instance, a VPN connection, or AWS Direct Connect. If you use AWS CloudFormation to create resources in a VPC with a VPC endpoint, you might need to modify your IAM endpoint policy so that it permits access to certain S3 buckets.

AWS CloudFormation has S3 buckets in each region to monitor responses to a custom resource (p. 361) request or a wait condition (p. 228). If a template includes custom resources or wait conditions in a VPC, the VPC endpoint policy must allow users to send responses to the following buckets:

- For custom resources, permit traffic to the `cloudformation-custom-resource-response-region` bucket.
- For wait conditions, permit traffic to the `cloudformation-waitcondition-region` bucket.

If the endpoint policy blocks traffic to these buckets, AWS CloudFormation won't receive responses and the stack operation fails. For example, if you have a resource in a VPC in the `us-west-2` region that must respond to a wait condition, the resource must be able to send a response to the `cloudformation-waitcondition-us-west-2` bucket.

For a list of regions that AWS CloudFormation supports, see the Regions and Endpoints page in the *Amazon Web Services General Reference*.

Getting Started with AWS CloudFormation

Because you can use AWS CloudFormation to launch many different types of resources, the getting started walkthrough will touch on just a few simple concepts to help you get an idea of how to use AWS CloudFormation.

In this section, you will use the AWS Management Console to create a stack from an example template from the AWS CloudFormation Sample Template Library and learn the basics of creating a template.

In the following walkthrough, we'll use a sample template to launch, update, and delete a stack. After you learn the fundamentals, you can learn more about creating more complex templates and stacks.

AWS CloudFormation makes deploying a set of Amazon Web Services (AWS) resources as simple as submitting a template. A *template* is a simple text file that describes a *stack*, a collection of AWS resources you want to deploy together as a group. You use the template to define all the AWS resources you want in your stack. This can include Amazon Elastic Compute Cloud instances, Amazon Relational Database Service DB Instances, and other resources. For a list of resource types, see AWS Resource Types Reference (p. 390).

The following video walks you through the stack creation example presented in the Get Started (p. 20) section: Getting Started with AWS CloudFormation

Topics

Get Started

With the right template, you can deploy at once all the AWS resources you need for an application. In this section, you'll examine a template that declares the resources for a WordPress blog, creates a WordPress blog as a stack, monitors the stack creation process, examines the resources on the stack, and then deletes the stack. You use the AWS Management Console to complete these tasks.

Step 1: Pick a template

Next, you'll need a template that specifies the resources that you want in your stack. For this step, you use a sample template that is already prepared. The sample template creates a basic WordPress blog that uses a single Amazon EC2 instance with a local MySQL database for storage. The template also creates an Amazon EC2 security group to control firewall settings for the Amazon EC2 instance.

Important
AWS CloudFormation is free, but the AWS resources that AWS CloudFormation creates are live (and not running in a sandbox). You will incur the standard usage fees for these resources until you terminate them in the last task in this tutorial. The total charges will be minimal. For information about how you might minimize any charges, go to http://aws.amazon.com/free/.

To view the template

- You can view the JSON or YAML WordPress sample template. You don't need to download it because you will use the template URL later in this guide. For more information about the template formats, see AWS CloudFormation Template Formats (p. 127).

A template is a JSON or YAML text file that contains the configuration information about the AWS resources you want to create in the stack. For this walkthrough, the sample template includes six top-level sections: `AWSTemplateFormatVersion`, `Description`, `Parameters`, `Mappings`, `Resources`, and `Outputs`; however, only the `Resources` section is required.

The Resources section contains the definitions of the AWS resources you want to create with the template. Each resource is listed separately and specifies the properties that are necessary for creating that particular resource. The following resource declaration is the configuration for the EC2 instance, which in this example has the logical name `WebServer`:

If you have created EC2 instances before, you can recognize properties, such as `ImageId`, `InstanceType`, and `KeyName`, that determine the configuration of the instance. Resource declarations are an efficient way to specify all these configuration settings at once. When you put resource declarations in a template, you can create and configure all the declared resources easily by using the template to create a stack. To launch the same configuration of resources, all you have to do is create a new stack that uses the same template.

The resource declaration begins with a string that specifies the logical name for the resource. As you'll see, the logical name can be used to refer to resources within the template.

You use the *Parameters* section to declare values that can be passed to the template when you create the stack. A parameter is an effective way to specify sensitive information, such as user names and passwords, that you don't want to store in the template itself. It is also a way to specify information that might be unique to the specific application or configuration you are deploying, for example, a domain name or instance type. When you create the WordPress stack later in this section, you'll see the set of parameters declared in the template appear on the **Specify Details** page of the **Create Stack** wizard, where you can specify the parameters before you create the stack.

The following parameters are used in the template to specify values that are used in properties of the EC2 instance:

In the `WebServer` resource declaration, you see the `KeyName` property specified with the `KeyName` parameter:

The braces contain a call to the Ref (p. 1343) function with `KeyName` as its input. The Ref function returns the value of the object it refers to. In this case, the Ref function sets the `KeyName` property to the value that was specified for `KeyName` when the stack was created.

The Ref function can also set a resource's property to the value of another resource. For example, the resource declaration `WebServer` contains the following property declaration:

The `SecurityGroups` property takes a list of EC2 security groups. The Ref function has an input of `WebServerSecurityGroup`, which is the logical name of a security group in the template, and adds the name of `WebServerSecurityGroup` to the `SecurityGroups` property.

In the template, you'll also find a *Mappings* section. You use mappings to declare conditional values that are evaluated in a similar manner as a lookup table statement. The template uses mappings to select the correct Amazon machine image (AMI) for the region and the architecture type for the instance type. *Outputs* define custom values that are returned by the `aws cloudformation describe-stacks` command and in the AWS CloudFormation console **Outputs** tab after the stack is created. You can use output values to return information from the resources in the stack, such as the URL for a website that was created in the template. We cover mappings, outputs, and other things about templates in more detail in Learn Template Basics (p. 24).

That's enough about templates for now. Let's start creating a stack.

Step 2: Make sure you have prepared any required items for the stack

Before you create a stack from a template, you must ensure that all dependent resources that the template requires are available. A template can use or refer to both existing AWS resources and resources declared in the template itself. AWS CloudFormation takes care of checking references to resources in the template and also checks references to existing resources to ensure that they exist in the region where you are creating the stack. If your template refers to a dependent resource that does not exist, stack creation fails.

The example WordPress template contains an input parameter, `KeyName`, that specifies the key pair used for the Amazon EC2 instance that is declared in the template. The template depends on the user who creates a stack from the template to supply a valid Amazon EC2 key pair for the `KeyName` parameter. If you supply a valid key pair name, the stack creates successfully. If you don't supply a valid key pair name, the stack is rolled back.

Make sure you have a valid Amazon EC2 key pair and record the key pair name before you create the stack.

To see your key pairs, open the Amazon EC2 console, then click **Key Pairs** in the navigation pane.

> **Note**
> If you don't have an Amazon EC2 key pair, you must create the key pair in the same region where you are creating the stack. For information about creating a key pair, see Getting an SSH Key Pair in the *Amazon EC2 User Guide for Linux Instances*.

Now that you have a valid key pair, let's use the WordPress template to create a stack.

Step 3: Create the stack

You will create your stack based on the *WordPress-1.0.0* file discussed earlier. The template contains several AWS resources, such as an EC2 instance.

To create the WordPress stack

1. Sign in to the AWS Management Console and open the AWS CloudFormation console at https://console.aws.amazon.com/cloudformation/.
2. If this is a new AWS CloudFormation account, click **Create New Stack**. Otherwise, click **Create Stack**.
3. In the **Template** section, select **Specify an Amazon S3 Template URL** to type or paste the URL for the sample WordPress template, and then click **Next**:

   ```
   https://s3-us-west-2.amazonaws.com/cloudformation-templates-us-west-2/
   WordPress_Single_Instance.template
   ```

   > **Note**
   > AWS CloudFormation templates that are stored in an S3 bucket must be accessible to the user who is creating the stack, and must be located in the *same region* as the stack that is being created. Therefore, if the S3 bucket is located in the us-east-1 Region, the stack must also be created in us-east-1.

4. In the **Specify Details** section, enter a stack name in the **Name** field. For this example, use **`MyWPTestStack`**. The stack name cannot contain spaces.
5. In the **KeyName** field, enter the name of a valid Amazon EC2 key pair in the same region you are creating the stack.

   Note
   On the **Specify Parameters** page, you'll recognize the parameters from the Parameters section of the template.
6. Click **Next**.
7. In this scenario, we won't add any tags. Click **Next**. Tags, which are key-value pairs, can help you identify your stacks. For more information, see Adding Tags to Your AWS CloudFormation Stack.
8. Review the information for the stack. When you're satisfied with the settings, click **Create**.

Your stack might take several minutes to create—but you probably don't want to just sit around waiting. If you're like us, you'll want to know how the stack creation is going.

Step 4: Monitor the progress of stack creation

After you complete the **Create Stack** wizard, AWS CloudFormation begins creating the resources that are specified in the template. Your new stack, `MyWPTestStack`, appears in the list at the top portion of the **CloudFormation** console. Its status should be CREATE_IN_PROGRESS. You can see detailed status for a stack by viewing its events.

To view the events for the stack

1. On the AWS CloudFormation console, select the stack `MyWPTestStack` in the list.
2. In the stack details pane, click the **Events** tab.

   The console automatically refreshes the event list with the most recent events every 60 seconds.

The **Events** tab displays each major step in the creation of the stack sorted by the time of each event, with latest events on top.

The first event (at the bottom of the event list) is the start of the stack creation process:

```
2013-04-24 18:54 UTC-7 CREATE_IN_PROGRESS AWS::CloudFormation::Stack MyWPTestStack User
initiated
```

Next are events that mark the beginning and completion of the creation of each resource. For example, creation of the EC2 instance results in the following entries:

```
2013-04-24 18:59 UTC-7 CREATE_COMPLETE AWS::EC2::Instance...
```

```
2013-04-24 18:54 UTC-7 CREATE_IN_PROGRESS AWS::EC2::Instance...
```

The `CREATE_IN_PROGRESS` event is logged when AWS CloudFormation reports that it has begun to create the resource. The `CREATE_COMPLETE` event is logged when the resource is successfully created.

When AWS CloudFormation has successfully created the stack, you will see the following event at the top of the **Events** tab:

```
2013-04-24 19:17 UTC-7 CREATE_COMPLETE AWS::CloudFormation::Stack MyWPTestStack
```

If AWS CloudFormation cannot create a resource, it reports a `CREATE_FAILED` event and, by default, rolls back the stack and deletes any resources that have been created. The **Status Reason** column displays the issue that caused the failure.

Step 5: Use your stack resources

When the stack `MyWPTestStack` has a status of `CREATE_COMPLETE`, AWS CloudFormation has finished creating the stack, and you can start using its resources.

The sample WordPress stack creates a WordPress website. You can continue with the WordPress setup by running the WordPress installation script.

To complete the WordPress installation

1. On the **Outputs** tab, in the **WebsiteURL** row, click the link in the **Value** column.

   The `WebsiteURL` output value is the URL of the installation script for the WordPress website that you created with the stack.
2. On the web page for the WordPress installation, follow the on-screen instructions to complete the WordPress installation. For more information about installing WordPress, see http://codex.wordpress.org/Installing_WordPress.

   After you complete the installation and log in, you are directed to the dashboard where you can set additional options for your WordPress blog. Then, you can start writing posts for your blog that you successfully created by using a AWS CloudFormation template.

Step 6: Clean Up

You have completed the AWS CloudFormation getting started tasks. To make sure you are not charged for any unwanted services, you can clean up by deleting the stack and its resources.

To delete the stack and its resources

1. From the AWS CloudFormation console, select the `MyWPTestStack` stack.
2. Click **Delete Stack**.
3. In the confirmation message that appears, click **Yes, Delete**.

The status for `MyWPTestStack` changes to `DELETE_IN_PROGRESS`. In the same way you monitored the creation of the stack, you can monitor its deletion by using the **Event** tab. When AWS CloudFormation completes the deletion of the stack, it removes the stack from the list.

Congratulations! You successfully picked a template, created a stack, viewed and used its resources, and deleted the stack and its resources. Not only that, you were able to set up a WordPress blog using a AWS CloudFormation template. You can find other templates in the AWS CloudFormation Sample Template Library.

Now it's time to learn more about templates so that you can easily modify existing templates or create your own: Learn Template Basics (p. 24).

Learn Template Basics

Topics

In Get Started (p. 20), you learned how to use a template to create a stack. You saw resources declared in a template and how they map to resources in the stack. We also touched on input parameters and how they enable you to pass in specific values when you create a stack from a template. In this section, we'll go deeper into resources and parameters. We'll also cover the other components of templates so that you'll know how to use these components together to create templates that produce the AWS resources you want.

What is an AWS CloudFormation Template?

A template is a declaration of the AWS resources that make up a stack. The template is stored as a text file whose format complies with the JavaScript Object Notation (JSON) or YAML standard. Because they are just text files, you can create and edit them in any text editor and manage them in your source control system with the rest of your source code. For more information about the template formats, see AWS CloudFormation Template Formats (p. 127).

In the template, you declare the AWS resources you want to create and configure. You declare an object as a name-value pair or a pairing of a name with a set of child objects enclosed. The syntax depends on the format you use. For more information, see the Template Anatomy (p. 128). The only required top-level object is the Resources object, which must declare at least one resource. Let's start with the most basic template containing only a Resources object, which contains a single resource declaration.

Resources: Hello Bucket!

The Resources object contains a list of resource objects. A resource declaration contains the resource's attributes, which are themselves declared as child objects. A resource must have a `Type` attribute, which defines the kind of AWS resource you want to create. The `Type` attribute has a special format:

```
AWS::ProductIdentifier::ResourceType
```

For example, the resource type for an Amazon S3 bucket is AWS::S3::Bucket (p. 937). For a full list of resource types, see Template Reference (p. 390).

Let's take a look at a very basic template. The following template declares a single resource of type AWS::S3::Bucket: with the name HelloBucket.

If you use this template to create a stack, AWS CloudFormation will create an Amazon S3 bucket. Creating a bucket is simple, because AWS CloudFormation can create a bucket with default settings. For other resources, such as an Auto Scaling group or EC2 instance, AWS CloudFormation requires more information. Resource declarations use a `Properties` attribute to specify the information used to create a resource.

Depending on the resource type, some properties are required, such as the ImageId property for an AWS::EC2::Instance (p. 574) resource, and others are optional. Some properties have default values, such as the AccessControl property of the AWS::S3::Bucket resource, so specifying a value for those properties is optional. Other properties are not required but may add functionality that you want, such as the WebsiteConfiguration property of the AWS::S3::Bucket resource. Specifying a value for such properties is entirely optional and based on your needs. In the example above, because the AWS::S3::Bucket resource has only optional properties and we didn't need any of the optional features, we could accept the defaults and omit the Properties attribute.

To view the properties for each resource type, see the topics in Resource Property Types Reference (p. 1004).

Resource Properties and Using Resources Together

Usually, a property for a resource is simply a string value. For example, the following template specifies a canned ACL (PublicRead) for the AccessControl property of the bucket.

Some resources can have multiple properties, and some properties can have one or more subproperties. For example, the AWS::S3::Bucket (p. 937) resource has two properties, AccessControl and WebsiteConfiguration. The WebsiteConfiguration property has two subproperties, IndexDocument and ErrorDocument. The following template shows our original bucket resource with the additional properties.

One of the greatest benefits of templates and AWS CloudFormation is the ability to create a set of resources that work together to create an application or solution. The name used for a resource within the template is a logical name. When AWS CloudFormation creates the resource, it generates a physical name that is based on the combination of the logical name, the stack name, and a unique ID.

You're probably wondering how you set properties on one resource based on the name or property of another resource. For example, you can create a CloudFront distribution backed by an S3 bucket or an EC2 instance that uses EC2 security groups, and all of these resources can be created in the same template. AWS CloudFormation has a number of intrinsic functions that you can use to refer to other resources and their properties. You can use the Ref function (p. 1343) to refer to an identifying property of a resource. Frequently, this is the physical name of the resource; however, sometimes it can be an identifier, such as the IP address for an AWS::EC2::EIP (p. 564) resource or an Amazon Resource Name (ARN) for an Amazon SNS topic. For a list of values returned by the Ref function, see Ref function (p. 1343). The following template contains an AWS::EC2::Instance (p. 574) resource. The resource's SecurityGroups property calls the Ref function to refer to the AWS::EC2::SecurityGroup resource InstanceSecurityGroup.

The SecurityGroups property is a list of security groups, and in this example we have only one item in the list. The following template has an additional item in the property list of the SecurityGroup.

MyExistingSecurityGroup is a string that refers to an existing EC2 security group instead of a security group declared in a template. You use literal strings to refer to existing AWS resources.

In the example above, the KeyName property of the AWS::EC2::Instance (p. 574) is the literal string mykey. This means that a key pair with the name mykey must exist in the region where the stack is being created; otherwise, stack creation will fail because the key pair does not exist. The key pair you use can vary with the region where you are creating the stack, or you may want to share the template with someone else so that they can use it with their AWS account. If so, you can use an input parameter so that the key pair name can be specified when the stack is created. The Ref function can refer to input parameters that are specified at stack creation time. The following template adds a Parameters object containing the KeyName parameter, which is used to specify the KeyName property for the AWS::EC2::Instance resource. The parameter type is `AWS::EC2::KeyPair::KeyName`, which ensures a user specifies a valid key pair name in his or her account and in the region where the stack is being created.

The Ref function is handy if the parameter or the value returned for a resource is exactly what you want; however, you may need other attributes of a resource. For example, if you want to create a CloudFront distribution with an S3 origin, you need to specify the bucket location by using a DNS-style address. A number of resources have additional attributes whose values you can use in your template. To get these attributes, you use the Fn::GetAtt (p. 1324) function. The following template creates a CloudFront distribution resource that specifies the DNS name of an S3 bucket resource using Fn::GetAtt function to get the bucket's DomainName attribute.

The Fn::GetAtt function takes two parameters, the logical name of the resource and the name of the attribute to be retrieved. For a full list of available attributes for resources, see Fn::GetAtt (p. 1324). You'll notice that the Fn::GetAtt function lists its two parameters in an array. For functions that take multiple parameters, you use an array to specify their parameters.

Receiving User Input Using Input Parameters

So far, you've learned about resources and a little bit about how to use them together within a template. You've learned how to refer to input parameters, but we haven't gone deeply into how to define the input parameters themselves. Let's take a look at parameter declarations and how you can restrict and validate user input.

You declare parameters in a template's Parameters object. A parameter contains a list of attributes that define its value and constraints against its value. The only required attribute is Type, which can be String, Number, or an AWS-specific type. You can also add a Description attribute that tells a user more about what kind of value they should specify. The parameter's name and description appear in the Specify Parameters page when a user uses the template in the Create Stack wizard.

The following template fragment is a Parameters object that declares the parameters used in the Specify Parameters page above.

For parameters with default values, AWS CloudFormation uses the default values unless users specify another value. If you omit the default attribute, users are required to specify a value for that parameter; however, requiring the user to input a value does not ensure that the value is valid. To validate the value of a parameter, you can declare constraints or specify an AWS-specific parameter type.

You'll notice that the `KeyName` parameter has no `Default` attribute and the other parameters do. For example, the `WordPress` parameter has the attribute `Default: admin`, but the `KeyName` parameter has none. Users must specify a key name value at stack creation. If they don’t, AWS CloudFormation fails to create the stack and throws an exception: `Parameters: [KeyName] must have values`.

For AWS-specific parameter types, AWS CloudFormation validates input values against existing values in the user's AWS account and in the region where he or she is creating the stack *before* creating any stack resources. In the sample template, the `KeyName` parameter is an AWS-specific parameter type of `AWS::EC2::KeyPair::KeyName`. AWS CloudFormation checks that users specify a valid EC2 key pair name before creating the stack. Another example of an AWS-specific parameter type is `AWS::EC2::VPC::Id`, which requires users to specify a valid VPC ID. In addition to upfront validation, the AWS console shows a drop-down list of valid values for AWS-specific parameter types, such as valid EC2 key pair names or VPC IDs, when users use the Create Stack wizard.

For the `String` type, you can use the following attributes to declare constraints: `MinLength`, `MaxLength`, `Default`, `AllowedValues`, and `AllowedPattern`. In the example above, the `WordPressUser` parameter has three constraints: the parameter value must be 1 to 16 character long (`MinLength`, `MaxLength`) and must begin with a letter followed by any combination of letters and numbers (`AllowedPattern`).

For the `Number` type, you can declare the following constraints: `MinValue`, `MaxValue`, `Default`, and `AllowedValues`. A number can be an integer or a float value. In the example above, the `WebServerPort` parameter must be a number between 1 and 65535 inclusive (`MinValue`, `MaxValue`).

Earlier in this section, we mentioned that parameters are a good way to specify sensitive or implementation-specific data, such as passwords or user names, that you need to use but do not want to embed in the template itself. For sensitive information, you can use the `NoEcho` attribute to prevent a parameter value from being displayed in the console, command line tools, or API. If you set the `NoEcho` attribute to `true`, the parameter value is returned as asterisks (`*****`). In the example above, the `WordPressUser` parameter value is not visible to anyone viewing the stack's settings, and its value is returned as asterisks.

Specifying Conditional Values Using Mappings

Parameters are a great way to enable users to specify unique or sensitive values for use in the properties of stack resources; however, there may be settings that are region dependent or are somewhat complex for users to figure out because of other conditions or dependencies. In these cases, you would want to put some logic in the template itself so that users can specify simpler values (or none at all) to get the results that they want. In an earlier example, we hardcoded the AMI ID for the ImageId property of our EC2 instance. This works fine in the US-East region, where it represents the AMI that we want. However, if the user tries to build the stack in a different region he or she will get the wrong AMI or no AMI at all. (AMI IDs are unique to a region, so the same AMI ID in a different region may not represent any AMI or a completely different one.)

To avoid this problem, you need a way to specify the right AMI ID based on a conditional input (in this example, the region where the stack is created). There are two template features that can help, the Mappings object and the AWS::Region pseudo parameter.

The AWS::Region pseudo parameter is a value that AWS CloudFormation resolves as the region where the stack is created. Pseudo parameters are resolved by AWS CloudFormation when you create the stack. Mappings enable you to use an input value as a condition that determines another value. Similar to a switch statement, a mapping associates one set of values with another. Using the AWS::Region parameter together with a mapping, you can ensure that an AMI ID appropriate to the region is specified. The following template contains a Mappings object with a mapping named RegionMap that is used to map an AMI ID to the appropriate region.

In the RegionMap, each region is mapped to a name-value pair. The name-value pair is a label, and the value to map. In the RegionMap, AMI is the label and the AMI ID is the value. To use a map to return a value, you use the Fn::FindInMap (p. 1322) function, passing the name of the map, the value used to find the mapped value, and the label of the mapped value you want to return. In the example above, the ImageId property of the resource Ec2Instance uses the Fn::FindInMap function to determine its value by specifying RegionMap as the map to use, AWS::Region as the input value to map from, and AMI as the label to identify the value to map to. For example, if this template were used to create a stack in the us-west-1 region, ImageId would be set to ami-655a0a20.

> **Tip**
> The AWS::Region pseudo parameter enables you to get the region where the stack is created. Some resources, such as AWS::EC2::Instance (p. 574), AWS::AutoScaling::AutoScalingGroup (p. 433), and AWS::ElasticLoadBalancing::LoadBalancer (p. 719), have a property that specifies availability zones. You can use the Fn::GetAZs function (p. 1332) to get the list of all availability zones in a region.

Constructed Values and Output Values

Parameters and mappings are an excellent way to pass or determine specific values at stack creation time, but there can be situations where a value from a parameter or other resource attribute is only part of the value you need. For example, in the following fragment from the WordPress template, the Fn::Join function constructs the Target subproperty of the HealthCheck property for the ElasticLoadBalancer resource by concatenating the WebServerPort parameter with other literal strings to form the value needed.

The Fn::Join function takes two parameters, a delimiter that separates the values you want to concatenate and an array of values in the order that you want them to appear. In the example above, the Fn::Join function specifies an empty string as the delimiter and HTTP:, the value of the WebServerPort parameter, and a / character as the values to concatenate. If WebServerPort had a value of 8888, the Target property would be set to the following value:

```
HTTP:8888/
```

The Fn::Join function is also useful for declaring output values for the stack. The Outputs object in the template contains declarations for the values that you want to have available after the stack is created. An output is a convenient way to capture important information about your resources or input parameters. For example, in the WordPress template, we declare the following Outputs object.

Each output value has a name, a Value attribute that contains declaration of the value returned as the output value, and optionally a description of the value. In the previous example, InstallURL is the string returned by a Fn::Join function call that concatenates http://, the DNS name of the resource ElasticLoadBalancer, and /wp-admin/install.php. The output value would be similar to the following:

```
http://mywptests-elasticl-1gb51l6sl8y5v-206169572.us-east-1.elb.amazonaws.com/wp-admin/
install.php
```

In the Get Started tutorial, we used this link to conveniently go to the installation page for the WordPress blog that we created. AWS CloudFormation generates the output values after it finishes creating the stack. You can view output values in the Outputs tab of the AWS CloudFormation console or by using the aws cloudformation describe-stacks command.

Next Steps

We just walked through the basic parts of a template and how to use them. You learned about the following about templates:

- Declaring resources and their properties
- Referencing other resources with the Ref function and resource attributes using the Fn::GetAtt function
- Using parameters to enable users to specify values at stack creation time and using constraints to validate parameter input
- Using mappings to determine conditional values
- Using the Fn::Join function to construct values based on parameters, resource attributes, and other strings
- Using output values based to capture information about the stack's resources.

We didn't cover two top level objects in a template: AWSTemplateFormatVersion and Description. AWSTemplateFormatVersion is simply the version of the template format—if you don't specify it, AWS CloudFormation will use the latest version. The Description is any valid JSON or YAML string. This description appears in the Specify Parameters page of the Create Stack wizard. For more information, see Format Version (p. 131) and Description (p. 131).

Of course, there are more advanced template and stack features. Here is a list of a few important ones that you'll want to learn more about:

Optional attributes that can be used with any resource:

- DependsOn attribute (p. 1298) enables you to specify that one resource must be created after another.
- DeletionPolicy attribute (p. 1297) enables you to specify how AWS CloudFormation should handle the deletion of a resource.
- Metadata (p. 1303) attribute enables you to specify structured data with a resource.

AWS::CloudFormation::Stack (p. 485) enables you to nest another stack as a resource within your template.

Walkthrough: Updating a Stack

With AWS CloudFormation, you can update the properties for resources in your existing stacks. These changes can range from simple configuration changes, such as updating the alarm threshold on a CloudWatch alarm, to more complex changes, such as updating the Amazon Machine Image (AMI) running on an Amazon EC2 instance. Many of the AWS resources in a template can be updated, and we continue to add support for more.

This section walks through a simple progression of updates of a running stack. It shows how the use of templates makes it possible to use a version control system for the configuration of your AWS infrastructure, just as you use version control for the software you are running. We will walk through the following steps:

1. Create the Initial Stack (p. 35)—create a stack using a base Amazon Linux AMI, installing the Apache Web Server and a simple PHP application using the AWS CloudFormation helper scripts.
2. Update the Application (p. 36)—update one of the files in the application and deploy the software using AWS CloudFormation.
3. Update the Instance Type (p. 38)—change the instance type of the underlying Amazon EC2 instance.
4. Update the AMI on an Amazon EC2 instance (p. 40)—change the Amazon Machine Image (AMI) for the Amazon EC2 instance in your stack.
5. Add a Key Pair to an Instance (p. 41)—add an Amazon EC2 key pair to the instance, and then update the security group to allow SSH access to the instance.
6. Change the Stack's Resources (p. 42)—add and remove resources from the stack, converting it to an auto-scaled, load-balanced application by updating the template.

A Simple Application

We'll begin by creating a stack that we can use throughout the rest of this section. We have provided a simple template that launches a single instance PHP web application hosted on the Apache Web Server and running on an Amazon Linux AMI.

The Apache Web Server, PHP, and the simple PHP application are all installed by the AWS CloudFormation helper scripts that are installed by default on the Amazon Linux AMI. The following template snippet shows the metadata that describes the packages and files to install, in this case the Apache Web Server and the PHP infrastructure from the Yum repository for the Amazon Linux AMI. The snippet also shows the Services section, which ensures that the Apache Web Server is running. In the Properties section of the Amazon EC2 instance definition, the UserData property contains the CloudInit script that calls cfn-init to install the packages and files.

```
"WebServerInstance": {
  "Type" : "AWS::EC2::Instance",
  "Metadata" : {
    "AWS::CloudFormation::Init" : {
      "config" : {
        "packages" : {
          "yum" : {
            "httpd"             : [],
            "php"               : []
          }
        },

        "files" : {

          "/var/www/html/index.php" : {
            "content" : { "Fn::Join" : ["", [
              "<?php\n",
              "echo '<h1>AWS CloudFormation sample PHP application</h1>';\n",
              "echo '<p>", { "Ref" : "WelcomeMessage" }, "</p>';\n",
              "?>\n"
            ]]},
            "mode"    : "000644",
            "owner"   : "apache",
            "group"   : "apache"
          },
        },

        :

        "services" : {
          "sysvinit" : {
```

```
            "httpd"    : { "enabled" : "true", "ensureRunning" : "true" }
          }
        }
      }
    }
  },

  "Properties": {
    :
    "UserData"       : { "Fn::Base64" : { "Fn::Join" : ["", [
      "#!/bin/bash\n",
      "yum install -y aws-cfn-bootstrap\n",

      :

      "# Install the files and packages from the metadata\n",
      "/opt/aws/bin/cfn-init -v ",
      "         --stack ", { "Ref" : "AWS::StackName" },
      "         --resource WebServerInstance ",
      "         --region ", { "Ref" : "AWS::Region" }, "\n",
      :
    ]]}}
  }
},
```

The application itself is a very simple two-line "Hello, World" example that is entirely defined within the template. For a real-world application, the files may be stored on Amazon S3, GitHub, or another repository and referenced from the template. AWS CloudFormation can download packages (such as RPMs or RubyGems), as well as reference individual files and expand `.zip` and `.tar` files to create the application artifacts on the Amazon EC2 instance.

The template enables and configures the cfn-hup daemon to listen for changes to the configuration defined in the metadata for the Amazon EC2 instance. By using the cfn-hup daemon, you can update application software, such as the version of Apache or PHP, or you can update the PHP application file itself from AWS CloudFormation. The following snippet from the same Amazon EC2 resource in the template shows the pieces necessary to configure cfn-hup to call cfn-init to update the software if any changes to the metadata are detected:

```
"WebServerInstance": {
  "Type" : "AWS::EC2::Instance",
  "Metadata" : {
    "AWS::CloudFormation::Init" : {
      "config" : {

          :

        "files" : {

          :

          "/etc/cfn/cfn-hup.conf" : {
            "content" : { "Fn::Join" : ["", [
              "[main]\n",
              "stack=", { "Ref" : "AWS::StackName" }, "\n",
              "region=", { "Ref" : "AWS::Region" }, "\n"
            ]]},
            "mode"    : "000400",
            "owner"   : "root",
            "group"   : "root"
          },

          "/etc/cfn/hooks.d/cfn-auto-reloader.conf" : {
```

```
              "content": { "Fn::Join" : ["", [
                "[cfn-auto-reloader-hook]\n",
                "triggers=post.update\n",
                "path=Resources.WebServerInstance.Metadata.AWS::CloudFormation::Init\n",
                "action=/opt/aws/bin/cfn-init -s ", { "Ref" : "AWS::StackId" }, " -r
 WebServerInstance ",
                " --region     ", { "Ref" : "AWS::Region" }, "\n",
                "runas=root\n"
              ]]}
            }
          },
          :
    },

    "Properties": {

         :

      "UserData"       : { "Fn::Base64" : { "Fn::Join" : ["", [

        :

        "# Start up the cfn-hup daemon to listen for changes to the Web Server metadata\n",
        "/opt/aws/bin/cfn-hup || error_exit 'Failed to start cfn-hup'\n",

        :
      ]]}}
    }
  },
```

To complete the stack, the template creates an Amazon EC2 security group.

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",

  "Description" : "AWS CloudFormation Sample Template: Sample template that can be used to
 test EC2 updates. **WARNING** This template creates an Amazon Ec2 Instance. You will be
 billed for the AWS resources used if you create a stack from this template.",

  "Parameters" : {

    "InstanceType" : {
      "Description" : "WebServer EC2 instance type",
      "Type" : "String",
      "Default" : "m1.small",
      "AllowedValues" : [ "t1.micro", "t2.micro", "t2.small", "t2.medium", "m1.small",
 "m1.medium", "m1.large", "m1.xlarge", "m2.xlarge",
"m2.2xlarge", "m2.4xlarge", "m3.medium", "m3.large", "m3.xlarge", "m3.2xlarge",
 "c1.medium", "c1.xlarge", "c3.large", "c3.xlarge", "c3.2xlarge",
"c3.4xlarge", "c3.8xlarge", "g2.2xlarge", "r3.large", "r3.xlarge", "r3.2xlarge",
 "r3.4xlarge", "r3.8xlarge", "i2.xlarge", "i2.2xlarge", "i2.4xlarge",
"i2.8xlarge", "hi1.4xlarge", "hs1.8xlarge", "cr1.8xlarge", "cc2.8xlarge", "cg1.4xlarge"],
      "ConstraintDescription" : "must be a valid EC2 instance type."
    }
  },

  "Mappings" : {
    "AWSInstanceType2Arch" : {
      "t1.micro"    : { "Arch" : "PV64"   },
      "t2.micro"    : { "Arch" : "HVM64"  },
      "t2.small"    : { "Arch" : "HVM64"  },
      "t2.medium"   : { "Arch" : "HVM64"  },
      "m1.small"    : { "Arch" : "PV64"   },
      "m1.medium"   : { "Arch" : "PV64"   },
      "m1.large"    : { "Arch" : "PV64"   },
```

```
      "m1.xlarge"   : { "Arch" : "PV64"   },
      "m2.xlarge"   : { "Arch" : "PV64"   },
      "m2.2xlarge"  : { "Arch" : "PV64"   },
      "m2.4xlarge"  : { "Arch" : "PV64"   },
      "m3.medium"   : { "Arch" : "HVM64"  },
      "m3.large"    : { "Arch" : "HVM64"  },
      "m3.xlarge"   : { "Arch" : "HVM64"  },
      "m3.2xlarge"  : { "Arch" : "HVM64"  },
      "c1.medium"   : { "Arch" : "PV64"   },
      "c1.xlarge"   : { "Arch" : "PV64"   },
      "c3.large"    : { "Arch" : "HVM64"  },
      "c3.xlarge"   : { "Arch" : "HVM64"  },
      "c3.2xlarge"  : { "Arch" : "HVM64"  },
      "c3.4xlarge"  : { "Arch" : "HVM64"  },
      "c3.8xlarge"  : { "Arch" : "HVM64"  },
      "g2.2xlarge"  : { "Arch" : "HVMG2"  },
      "r3.large"    : { "Arch" : "HVM64"  },
      "r3.xlarge"   : { "Arch" : "HVM64"  },
      "r3.2xlarge"  : { "Arch" : "HVM64"  },
      "r3.4xlarge"  : { "Arch" : "HVM64"  },
      "r3.8xlarge"  : { "Arch" : "HVM64"  },
      "i2.xlarge"   : { "Arch" : "HVM64"  },
      "i2.2xlarge"  : { "Arch" : "HVM64"  },
      "i2.4xlarge"  : { "Arch" : "HVM64"  },
      "i2.8xlarge"  : { "Arch" : "HVM64"  },
      "hi1.4xlarge" : { "Arch" : "HVM64"  },
      "hs1.8xlarge" : { "Arch" : "HVM64"  },
      "cr1.8xlarge" : { "Arch" : "HVM64"  },
      "cc2.8xlarge" : { "Arch" : "HVM64"  }
    },

    "AWSRegionArch2AMI" : {
      "us-east-1"      : { "PV64" : "ami-50842d38", "HVM64" : "ami-08842d60", "HVMG2" :
"ami-3a329952"  },
      "us-west-2"      : { "PV64" : "ami-af86c69f", "HVM64" : "ami-8786c6b7", "HVMG2" :
"ami-47296a77"  },
      "us-west-1"      : { "PV64" : "ami-c7a8a182", "HVM64" : "ami-cfa8a18a", "HVMG2" :
"ami-331b1376"  },
      "eu-west-1"      : { "PV64" : "ami-aa8f28dd", "HVM64" : "ami-748e2903", "HVMG2" :
"ami-00913777"  },
      "ap-southeast-1" : { "PV64" : "ami-20e1c572", "HVM64" : "ami-d6e1c584", "HVMG2" :
"ami-fabe9aa8"  },
      "ap-northeast-1" : { "PV64" : "ami-21072820", "HVM64" : "ami-35072834", "HVMG2" :
"ami-5dd1ff5c"  },
      "ap-southeast-2" : { "PV64" : "ami-8b4724b1", "HVM64" : "ami-fd4724c7", "HVMG2" :
"ami-e98ae9d3"  },
      "sa-east-1"      : { "PV64" : "ami-9d6cc680", "HVM64" : "ami-956cc688", "HVMG2" :
"NOT_SUPPORTED" },
      "cn-north-1"     : { "PV64" : "ami-a857c591", "HVM64" : "ami-ac57c595", "HVMG2" :
"NOT_SUPPORTED" },
      "eu-central-1"   : { "PV64" : "ami-a03503bd", "HVM64" : "ami-b43503a9", "HVMG2" :
"ami-b03503ad"  }
    }
  },

  "Resources" : {

    "WebServerInstance": {
      "Type" : "AWS::EC2::Instance",
      "Metadata" : {
        "Comment" : "Install a simple PHP application",
        "AWS::CloudFormation::Init" : {
          "config" : {
            "packages" : {
              "yum" : {
                "httpd"             : [],
```

```
                "php"               : []
              }
            },

            "files" : {

              "/var/www/html/index.php" : {
                "content" : { "Fn::Join" : ["", [
                  "<?php\n",
                  "echo '<h1>AWS CloudFormation sample PHP application</h1>';\n",
                  "?>\n"
                ]]},
                "mode"    : "000644",
                "owner"   : "apache",
                "group"   : "apache"
              },


              "/etc/cfn/cfn-hup.conf" : {
                "content" : { "Fn::Join" : ["", [
                  "[main]\n",
                  "stack=", { "Ref" : "AWS::StackId" }, "\n",
                  "region=", { "Ref" : "AWS::Region" }, "\n"
                ]]},
                "mode"    : "000400",
                "owner"   : "root",
                "group"   : "root"
              },

              "/etc/cfn/hooks.d/cfn-auto-reloader.conf" : {
                "content": { "Fn::Join" : ["", [
                  "[cfn-auto-reloader-hook]\n",
                  "triggers=post.update\n",
                  "path=Resources.WebServerInstance.Metadata.AWS::CloudFormation::Init\n",
                  "action=/opt/aws/bin/cfn-init -s ", { "Ref" : "AWS::StackId" }, " -r
 WebServerInstance ",
                                                     " --region     ", { "Ref" :
 "AWS::Region" }, "\n",
                  "runas=root\n"
                ]]}
              }
            },

            "services" : {
              "sysvinit" : {
                "httpd"    : { "enabled" : "true", "ensureRunning" : "true" },
                "cfn-hup" : { "enabled" : "true", "ensureRunning" : "true",
                    "files" : ["/etc/cfn/cfn-hup.conf", "/etc/cfn/hooks.d/cfn-auto-
reloader.conf"]}
              }
            }
          }
        }
      },

      "Properties": {
        "ImageId" : { "Fn::FindInMap" : [ "AWSRegionArch2AMI", { "Ref" : "AWS::Region" },
                          { "Fn::FindInMap" : [ "AWSInstanceType2Arch", { "Ref" :
 "InstanceType" }, "Arch" ] } ] },
        "InstanceType"   : { "Ref" : "InstanceType" },
        "SecurityGroups" : [ {"Ref" : "WebServerSecurityGroup"} ],
        "UserData"       : { "Fn::Base64" : { "Fn::Join" : ["", [
             "#!/bin/bash -xe\n",
             "yum install -y aws-cfn-bootstrap\n",

             "# Install the files and packages from the metadata\n",
```

```
              "/opt/aws/bin/cfn-init -v ",
              "         --stack ", { "Ref" : "AWS::StackName" },
              "         --resource WebServerInstance ",
              "         --region ", { "Ref" : "AWS::Region" }, "\n",

              "# Start up the cfn-hup daemon to listen for changes to the Web Server
 metadata\n",
              "/opt/aws/bin/cfn-hup || error_exit 'Failed to start cfn-hup'\n",

              "# Signal the status from cfn-init\n",
              "/opt/aws/bin/cfn-signal -e $? ",
              "         --stack ", { "Ref" : "AWS::StackName" },
              "         --resource WebServerInstance ",
              "         --region ", { "Ref" : "AWS::Region" }, "\n"
        ]]}}
      },
      "CreationPolicy" : {
        "ResourceSignal" : {
          "Timeout" : "PT5M"
        }
      }
    },

    "WebServerSecurityGroup" : {
      "Type" : "AWS::EC2::SecurityGroup",
      "Properties" : {
        "GroupDescription" : "Enable HTTP access via port 80",
        "SecurityGroupIngress" : [
          {"IpProtocol" : "tcp", "FromPort" : "80", "ToPort" : "80", "CidrIp" :
 "0.0.0.0/0"}
        ]
      }
    }
  },

  "Outputs" : {
    "WebsiteURL" : {
      "Description" : "Application URL",
      "Value" : { "Fn::Join" : ["", ["http://", { "Fn::GetAtt" : [ "WebServerInstance",
 "PublicDnsName" ]}]] }
    }
  }
}
```

This example uses a single Amazon EC2 instance, but you can use the same mechanisms on more complex solutions that make use of Elastic Load Balancers and Auto Scaling groups to manage a collection of application servers. There are, however, some special considerations for Auto Scaling groups. For more information, see Updating Auto Scaling Groups (p. 38).

Create the Initial Stack

For the purposes of this example, we’ll use the AWS Management Console to create an initial stack from the sample template.

Caution
Completing this procedure will deploy live AWS services. You will be charged the standard usage rates as long as these services are running.

To create the stack from the AWS Management Console

1. Copy the previous template and save it locally on your system as a text file. Note the location because you'll need to use the file in a subsequent step.

2. Log in to the AWS CloudFormation console at https://console.aws.amazon.com/cloudformation .
3. Click **Create New Stack**.
4. In the **Create New Stack** wizard, on the **Select Template** screen, type `UpdateTutorial` in the **Name** field. On the same page, select **Upload a template to Amazon S3** and browse to the file that you downloaded in the first step, and then click **Next**.
5. On the **Specify Parameters** screen, in the **Instance Type** box, type `t1.micro`. Then click **Next**.
6. On the **Options** screen, click **Next**.
7. On the **Review** screen, verify that all the settings are as you want them, and then click **Create**.

After the status of your stack is CREATE_COMPLETE, the output tab will display the URL of your website. If you click the value of the WebsiteURL output, you will see your new PHP application working.

Update the Application

Now that we have deployed the stack, let's update the application. We'll make a simple change to the text that is printed out by the application. To do so, we’ll add an echo command to the index.php file as shown in this template snippet:

```
"WebServerInstance": {
      "Type" : "AWS::EC2::Instance",
      "Metadata" : {
        "AWS::CloudFormation::Init" : {
          "config" : {
              :

            "files" : {

              "/var/www/html/index.php" : {
                "content" : { "Fn::Join" : ["", [
                  "<?php\n",
                  "echo '<h1>AWS CloudFormation sample PHP application</h1>';\n",
                  "echo 'Updated version via UpdateStack';\n ",
                  "?>\n"
                ]]},
                "mode"    : "000644",
                "owner"   : "apache",
                "group"   : "apache"
              },

              :

      }
    },
```

Use a text editor to manually edit the template file that you saved locally.

Now, we'll update the stack.

To update the stack from the AWS Management Console

1. Log in to the AWS CloudFormation console, at: https://console.aws.amazon.com/cloudformation.
2. On the AWS CloudFormation dashboard, click the stack you created previously, and then click **Update Stack**.
3. In the **Update Stack** wizard, on the **Select Template** screen, select **Upload a template to Amazon S3**, select the modified template, and then click **Next**.
4. On the **Options** screen, click **Next**.

5. Click **Next** because the stack doesn't have a stack policy. All resources can be updated without an overriding policy.
6. On the **Review** screen, verify that all the settings are as you want them, and then click **Update**.

If you update the stack from the AWS Management Console, you will notice that the parameters that were used to create the initial stack are prepopulated on the **Parameters** page of the **Update Stack** wizard. If you use the `aws cloudformation update-stack` command, be sure to type in the same values for the parameters that you used originally to create the stack.

When your stack is in the UPDATE_COMPLETE state, you can click the WebsiteURL output value again to verify that the changes to your application have taken effect. By default, the cfn-hup daemon runs every 15 minutes, so it may take up to 15 minutes for the application to change once the stack has been updated.

To see the set of resources that were updated, go to the AWS CloudFormation console. On the **Events** tab, look at the stack events. In this particular case, the metadata for the Amazon EC2 instance WebServerInstance was updated, which caused AWS CloudFormation to also reevaluate the other resources (`WebServerSecurityGroup`) to ensure that there were no other changes. None of the other stack resources were modified. AWS CloudFormation will update only those resources in the stack that are affected by any changes to the stack. Such changes can be direct, such as property or metadata changes, or they can be due to dependencies or data flows through Ref, GetAtt, or other intrinsic template functions.

This simple update illustrates the process; however, you can make much more complex changes to the files and packages that are deployed to your Amazon EC2 instances. For example, you might decide that you need to add MySQL to the instance, along with PHP support for MySQL. To do so, simply add the additional packages and files along with any additional services to the configuration and then update the stack to deploy the changes. In the following template snippet, the changes are highlighted in red:

```
    "WebServerInstance": {
      "Type" : "AWS::EC2::Instance",
      "Metadata" : {
        "Comment" : "Install a simple PHP application",
        "AWS::CloudFormation::Init" : {
          "config" : {
            "packages" : {
              "yum" : {
                "httpd"             : [],
                "php"               : [],
                "php-mysql"         : [],
                "mysql-server"      : [],
                "mysql-libs"        : [],
                "mysql"             : []
              }
            },

            :

            "services" : {
              "sysvinit" : {
                "httpd"    : { "enabled" : "true", "ensureRunning" : "true" },
                "cfn-hup" : { "enabled" : "true", "ensureRunning" : "true",
                    "files" : ["/etc/cfn/cfn-hup.conf", "/etc/cfn/hooks.d/cfn-auto-
reloader.conf"]},
                "mysqld"   : { "enabled" : "true", "ensureRunning" : "true" }
              }
            }
          }
        }
      },

      "Properties": {
           :
```

```
      }
    }
```

You can update the CloudFormation metadata to update to new versions of the packages used by the application. In the previous examples, the version property for each package is empty, indicating that cfn-init should install the latest version of the package.

```
    "packages" : {
      "yum" : {
        "httpd"             : [],
        "php"               : []
     }
```

You can optionally specify a version string for a package. If you change the version string in subsequent update stack calls, the new version of the package will be deployed. Here's an example of using version numbers for RubyGems packages. Any package that supports versioning can have specific versions.

```
  "packages" : {
    "rubygems" : {
      "mysql"           : [],
      "rubygems-update" : ["1.6.2"],
      "rake"            : ["0.8.7"],
      "rails"           : ["2.3.11"]
    }
    }
```

Updating Auto Scaling Groups

If you are using Auto Scaling groups in your template, as opposed to Amazon EC2 instance resources, updating the application will work in exactly the same way; however, AWS CloudFormation does not provide any synchronization or serialization across the Amazon EC2 instances in an Auto Scaling group. The cfn-hup daemon on each host will run independently and update the application on its own schedule. When you use cfn-hup to update the on-instance configuration, each instance will run the cfn-hup hooks on its own schedule; there is no coordination between the instances in the stack. You should consider the following:

- If the cfn-hup changes run on all Amazon EC2 instances in the Auto Scaling group at the same time, your service might be unavailable during the update.
- If the cfn-hup changes run at different times, old and new versions of the software may be running at the same.

To avoid these issues, consider forcing a rolling update on your instances in the Auto Scaling group. For more information, see UpdatePolicy (p. 1303).

Changing Resource Properties

With AWS CloudFormation, you can change the properties of an existing resource in the stack. The following sections describe various updates that solve specific problems; however, any property of any resource that supports updating in the stack can be modified as necessary.

Update the Instance Type

The stack we have built so far uses a t1.micro Amazon EC2 instance. Let's suppose that your newly created website is getting more traffic than a t1.micro instance can handle, and now you want to move to an m1.small Amazon EC2 instance type. If the architecture of the instance type changes, the instance will be created with a different AMI. If you check out the mappings in the template, you will see that both the t1.micro and m1.small are the same architectures and use the same Amazon Linux AMIs.

```
"Mappings" : {
    "AWSInstanceType2Arch" : {
      "t1.micro"    : { "Arch" : "PV64"   },
      "t2.micro"    : { "Arch" : "HVM64"  },
      "t2.small"    : { "Arch" : "HVM64"  },
      "t2.medium"   : { "Arch" : "HVM64"  },
      "m1.small"    : { "Arch" : "PV64"   },
      "m1.medium"   : { "Arch" : "PV64"   },
      "m1.large"    : { "Arch" : "PV64"   },
      "m1.xlarge"   : { "Arch" : "PV64"   },
      "m2.xlarge"   : { "Arch" : "PV64"   },
      "m2.2xlarge"  : { "Arch" : "PV64"   },
      "m2.4xlarge"  : { "Arch" : "PV64"   },
      "m3.medium"   : { "Arch" : "HVM64"  },
      "m3.large"    : { "Arch" : "HVM64"  },
      "m3.xlarge"   : { "Arch" : "HVM64"  },
      "m3.2xlarge"  : { "Arch" : "HVM64"  },
      "c1.medium"   : { "Arch" : "PV64"   },
      "c1.xlarge"   : { "Arch" : "PV64"   },
      "c3.large"    : { "Arch" : "HVM64"  },
      "c3.xlarge"   : { "Arch" : "HVM64"  },
      "c3.2xlarge"  : { "Arch" : "HVM64"  },
      "c3.4xlarge"  : { "Arch" : "HVM64"  },
      "c3.8xlarge"  : { "Arch" : "HVM64"  },
      "g2.2xlarge"  : { "Arch" : "HVMG2"  },
      "r3.large"    : { "Arch" : "HVM64"  },
      "r3.xlarge"   : { "Arch" : "HVM64"  },
      "r3.2xlarge"  : { "Arch" : "HVM64"  },
      "r3.4xlarge"  : { "Arch" : "HVM64"  },
      "r3.8xlarge"  : { "Arch" : "HVM64"  },
      "i2.xlarge"   : { "Arch" : "HVM64"  },
      "i2.2xlarge"  : { "Arch" : "HVM64"  },
      "i2.4xlarge"  : { "Arch" : "HVM64"  },
      "i2.8xlarge"  : { "Arch" : "HVM64"  },
      "hi1.4xlarge" : { "Arch" : "HVM64"  },
      "hs1.8xlarge" : { "Arch" : "HVM64"  },
      "cr1.8xlarge" : { "Arch" : "HVM64"  },
      "cc2.8xlarge" : { "Arch" : "HVM64"  }
    },

    "AWSRegionArch2AMI" : {
      "us-east-1"      : { "PV64" : "ami-50842d38", "HVM64" : "ami-08842d60", "HVMG2" :
 "ami-3a329952"  },
      "us-west-2"      : { "PV64" : "ami-af86c69f", "HVM64" : "ami-8786c6b7", "HVMG2" :
 "ami-47296a77"  },
      "us-west-1"      : { "PV64" : "ami-c7a8a182", "HVM64" : "ami-cfa8a18a", "HVMG2" :
 "ami-331b1376"  },
      "eu-west-1"      : { "PV64" : "ami-aa8f28dd", "HVM64" : "ami-748e2903", "HVMG2" :
 "ami-00913777"  },
      "ap-southeast-1" : { "PV64" : "ami-20e1c572", "HVM64" : "ami-d6e1c584", "HVMG2" :
 "ami-fabe9aa8"  },
      "ap-northeast-1" : { "PV64" : "ami-21072820", "HVM64" : "ami-35072834", "HVMG2" :
 "ami-5dd1ff5c"  },
      "ap-southeast-2" : { "PV64" : "ami-8b4724b1", "HVM64" : "ami-fd4724c7", "HVMG2" :
 "ami-e98ae9d3"  },
      "sa-east-1"      : { "PV64" : "ami-9d6cc680", "HVM64" : "ami-956cc688", "HVMG2" :
 "NOT_SUPPORTED" },
      "cn-north-1"     : { "PV64" : "ami-a857c591", "HVM64" : "ami-ac57c595", "HVMG2" :
 "NOT_SUPPORTED" },
      "eu-central-1"   : { "PV64" : "ami-a03503bd", "HVM64" : "ami-b43503a9", "HVMG2" :
 "ami-b03503ad"  }
```

```
    }
  }
```

Let's use the template that we modified in the previous section to change the instance type. Because InstanceType was an input parameter to the template, we don't need to modify the template; we can simply change the value of the parameter in the Stack Update wizard, on the Specify Parameters page.

To update the stack from the AWS Management Console

1. Log in to the AWS CloudFormation console at https://console.aws.amazon.com/cloudformation.
2. On the AWS CloudFormation dashboard, click the stack you created previously, and then click **Update Stack**.
3. In the **Update Stack** wizard, on the **Select Template** screen, select **Use current template**, and then click **Next**.

   The Specify Details page appears with the parameters that were used to create the initial stack are pre-populated in the **Specify Parameters** section.
4. Change the value of the **InstanceType** text box from `t1.micro` to `t2.small`. Then, click **Next**.
5. On the **Options** screen, click **Next**.
6. Click **Next** because the stack doesn't have a stack policy. All resources can be updated without an overriding policy.
7. On the **Review** screen, verify that all the settings are as you want them, and then click **Update**.

You can dynamically change the instance type of an EBS-backed Amazon EC2 instance by starting and stopping the instance. AWS CloudFormation tries to optimize the change by updating the instance type and restarting the instance, so the instance ID does not change. When the instance is restarted, however, the public IP address of the instance does change. To ensure that the Elastic IP address is bound correctly after the change, AWS CloudFormation will also update the Elastic IP address. You can see the changes in the AWS CloudFormation console on the Events tab.

To check the instance type from the AWS Management Console, open the Amazon EC2 console, and locate your instance there.

Update the AMI on an Amazon EC2 instance

Now let's look at how we might change the Amazon Machine Image (AMI) running on the instance. We will trigger the AMI change by updating the stack to use a new Amazon EC2 instance type, such as t2.medium, which is an HVM64 instance type.

As in the previous section, we'll use our existing template to change the instance type used by our example stack. In the Stack Update wizard, on the Specify Parameters page, change the value of the Instance Type.

In this case, we cannot simply start and stop the instance to modify the AMI; AWS CloudFormation considers this a change to an immutable property of the resource. In order to make a change to an immutable property, AWS CloudFormation must launch a replacement resource, in this case a new Amazon EC2 instance running the new AMI.

After the new instance is running, AWS CloudFormation updates the other resources in the stack to point to the new resource. When all new resources are created, the old resource is deleted, a process known as UPDATE_CLEANUP. This time, you will notice that the instance ID and application URL of the instance in the stack has changed as a result of the update. The events in the Event table contain a description "Requested update has a change to an immutable property and hence creating a new physical resource" to indicate that a resource was replaced.

If you have application code written into the AMI that you want to update, you can use the same stack update mechanism to update the AMI to load your new application.

To update the AMI for an instance on your stack

1. Create your new AMIs containing your application or operating system changes. For more information, go to Creating Your Own AMIs in the *Amazon EC2 User Guide for Linux Instances*.
2. Update your template to incorporate the new AMI IDs.
3. Update the stack, either from the AWS Management Console as explained in Update the Application (p. 36) or by using the AWS command `aws cloudformation update-stack`.

When you update the stack, AWS CloudFormation detects that the AMI ID has changed, and then it triggers a stack update in the same way as we triggered the one above.

Update the Amazon EC2 Launch Configuration for an Auto Scaling Group

If you are using Auto Scaling groups rather than Amazon EC2 instances, the process of updating the running instances is a little different. With Auto Scaling resources, the configuration of the Amazon EC2 instances, such as the instance type or the AMI ID is encapsulated in the Auto Scaling launch configuration. You can make changes to the launch configuration in the same way as we made changes to the Amazon EC2 instance resources in the previous sections. However, changing the launch configuration does not impact any of the running Amazon EC2 instances in the Auto Scaling group. An updated launch configuration applies only to new instances that are created after the update.

If you want to propagate the change to your launch configuration across all the instances in your Auto Scaling group, you can use an update attribute. For more information, see UpdatePolicy (p. 1303).

Adding Resource Properties

So far, we've looked at changing existing properties of a resource in a template. You can also add properties that were not originally specified in the template. To illustrate that, we'll add an Amazon EC2 key pair to an existing EC2 instance and then open up port 22 in the Amazon EC2 Security Group so that you can use Secure Shell (SSH) to access the instance.

Add a Key Pair to an Instance

To add SSH access to an existing Amazon EC2 instance

1. Add two additional parameters to the template to pass in the name of an existing Amazon EC2 key pair and SSH location.

```
 "Parameters" : {

   "KeyName" : {
     "Description" : "Name of an existing Amazon EC2 key pair for SSH access",
     "Type": "AWS::EC2::KeyPair::KeyName"
   },
   "SSHLocation" : {
     "Description" : " The IP address range that can be used to SSH to the EC2 
instances",
     "Type": "String",
     "MinLength": "9",
     "MaxLength": "18",
     "Default": "0.0.0.0/0",
     "AllowedPattern": "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\\d{1,2})",
     "ConstraintDescription": "must be a valid IP CIDR range of the form x.x.x.x/x."
   }
   :
 },
```

2. Add the KeyName property to the Amazon EC2 instance.

```
          "WebServerInstance": {
          "Type" : "AWS::EC2::Instance",
          :
          "Properties": {
          :
          "KeyName" : { "Ref" : "KeyName" },
          :
          }
          },
```

3. Add port 22 and the SSH location to the ingress rules for the Amazon EC2 security group.

```
    "WebServerSecurityGroup" : {
      "Type" : "AWS::EC2::SecurityGroup",
      "Properties" : {
        "GroupDescription" : "Enable HTTP and SSH",
        "SecurityGroupIngress" : [
          {"IpProtocol" : "tcp", "FromPort" : "22", "ToPort" : "22", "CidrIp" :
{ "Ref" : "SSHLocation"}},
          {"IpProtocol" : "tcp", "FromPort" : "80", "ToPort" : "80", "CidrIp" :
"0.0.0.0/0"}
        ]
      }
    },
```

4. Update the stack, either from the AWS Management Console as explained in Update the Application (p. 36) or by using the AWS command `aws cloudformation update-stack`.

Change the Stack's Resources

Since application needs can change over time, AWS CloudFormation allows you to change the set of resources that make up the stack. To demonstrate, we'll take the single instance application from Adding Resource Properties (p. 41) and convert it to an auto-scaled, load-balanced application by updating the stack.

This will create a simple, single instance PHP application using an Elastic IP address. We'll now turn the application into a highly available, auto-scaled, load balanced application by changing its resources during an update.

1. Add an Elastic Load Balancer resource.

```
    "ElasticLoadBalancer" : {
      "Type" : "AWS::ElasticLoadBalancing::LoadBalancer",
      "Properties" : {
        "CrossZone" : "true",
        "AvailabilityZones" : { "Fn::GetAZs" : "" },
        "LBCookieStickinessPolicy" : [ {
          "PolicyName" : "CookieBasedPolicy",
          "CookieExpirationPeriod" : "30"
        } ],
        "Listeners" : [ {
          "LoadBalancerPort" : "80",
          "InstancePort" : "80",
          "Protocol" : "HTTP",
          "PolicyNames" : [ "CookieBasedPolicy" ]
        } ],
        "HealthCheck" : {
          "Target" : "HTTP:80/",
```

```
          "HealthyThreshold" : "2",
          "UnhealthyThreshold" : "5",
          "Interval" : "10",
          "Timeout" : "5"
        }
      }
    }
```

2. Convert the EC2 instance in the template into an Auto Scaling Launch Configuration. The properties are identical, so we only need to change the type name from:

```
"WebServerInstance": {
  "Type" : "AWS::EC2::Instance",
```

   to:

```
"LaunchConfig": {
  "Type" : "AWS::AutoScaling::LaunchConfiguration",
```

   For clarity in the template, we changed the name of the resource from *WebServerInstance* to *LaunchConfig*, so you'll need to update the resource name referenced by cfn-init and cfn-hup (just search for WebServerInstance and replace it with LaunchConfig, except for cfn-signal). For cfn-signal, you'll need to signal the Auto Scaling group (WebServerGroup) not the instance, as shown in the following snippet:

```
             "# Signal the status from cfn-init\n",
             "/opt/aws/bin/cfn-signal -e $? ",
             "         --stack ", { "Ref" : "AWS::StackName" },
             "         --resource WebServerGroup ",
             "         --region ", { "Ref" : "AWS::Region" }, "\n"
```

3. Add an Auto Scaling Group resource.

```
    "WebServerGroup" : {
      "Type" : "AWS::AutoScaling::AutoScalingGroup",
      "Properties" : {
        "AvailabilityZones" : { "Fn::GetAZs" : "" },
        "LaunchConfigurationName" : { "Ref" : "LaunchConfig" },
        "MinSize" : "1",
        "DesiredCapacity" : "1",
        "MaxSize" : "5",
        "LoadBalancerNames" : [ { "Ref" : "ElasticLoadBalancer" } ]
      },
      "CreationPolicy" : {
        "ResourceSignal" : {
          "Timeout" : "PT15M"
        }
      },
      "UpdatePolicy": {
        "AutoScalingRollingUpdate": {
          "MinInstancesInService": "1",
          "MaxBatchSize": "1",
          "PauseTime" : "PT15M",
          "WaitOnResourceSignals": "true"
        }
      }
    }
```

4. Update the Security Group definition to lock down the traffic to the instances from the load balancer.

```
    "WebServerSecurityGroup" : {
      "Type" : "AWS::EC2::SecurityGroup",
      "Properties" : {
        "GroupDescription" : "Enable HTTP access via port 80 locked down to the ELB and
 SSH access",
        "SecurityGroupIngress" : [
          {"IpProtocol" : "tcp", "FromPort" : "80", "ToPort" : "80",
 "SourceSecurityGroupOwnerId" : {"Fn::GetAtt" : ["ElasticLoadBalancer",
 "SourceSecurityGroup.OwnerAlias"]},
"SourceSecurityGroupName" : {"Fn::GetAtt" : ["ElasticLoadBalancer",
 "SourceSecurityGroup.GroupName"]}},
          {"IpProtocol" : "tcp", "FromPort" : "22", "ToPort" : "22", "CidrIp" :
 { "Ref" : "SSHLocation"}}
        ]
      }
    }
```

5. Update the Outputs to return the DNS Name of the Elastic Load Balancer as the location of the application from:

```
"WebsiteURL" : {
  "Value" : { "Fn::Join" : ["", ["http://",
      { "Fn::GetAtt" : [ "WebServerInstance", "PublicDnsName" ]}]]},
  "Description" : "Application URL"
}
```

   to:

```
"WebsiteURL" : {
  "Value" : { "Fn::Join" : ["", ["http://",
      { "Fn::GetAtt" : [ "ElasticLoadBalancer", "DNSName" ]}]]},
  "Description" : "Application URL"
}
```

For reference, the follow sample shows the complete template. If you use this template to update the stack, you will convert your simple, single instance application into a highly available, multi-AZ, auto-scaled and load balanced application. Only the resources that need to be updated will be altered, so had there been any data stores for this application, the data would have remained intact. Now, you can use AWS CloudFormation to grow or enhance your stacks as your requirements change.

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",

  "Description" : "AWS CloudFormation Sample Template: Sample template that can be used to
 test EC2 updates. **WARNING** This template creates an Amazon Ec2 Instance. You will be
 billed for the AWS resources used if you create a stack from this template.",

  "Parameters" : {

    "KeyName": {
      "Description" : "Name of an existing EC2 KeyPair to enable SSH access to the
 instance",
      "Type": "AWS::EC2::KeyPair::KeyName",
      "ConstraintDescription" : "must be the name of an existing EC2 KeyPair."
    },

    "SSHLocation" : {
```

```
      "Description" : " The IP address range that can be used to SSH to the EC2 instances",
      "Type": "String",
      "MinLength": "9",
      "MaxLength": "18",
      "Default": "0.0.0.0/0",
      "AllowedPattern": "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\\d{1,2})",
      "ConstraintDescription": "must be a valid IP CIDR range of the form x.x.x.x/x."
    },

    "InstanceType" : {
      "Description" : "WebServer EC2 instance type",
      "Type" : "String",
      "Default" : "m1.small",
      "AllowedValues" : [ "t1.micro", "t2.micro", "t2.small", "t2.medium", "m1.small",
 "m1.medium", "m1.large", "m1.xlarge", "m2.xlarge",
"m2.2xlarge", "m2.4xlarge", "m3.medium", "m3.large", "m3.xlarge", "m3.2xlarge",
 "c1.medium", "c1.xlarge", "c3.large", "c3.xlarge", "c3.2xlarge",
"c3.4xlarge", "c3.8xlarge", "g2.2xlarge", "r3.large", "r3.xlarge", "r3.2xlarge",
 "r3.4xlarge", "r3.8xlarge", "i2.xlarge", "i2.2xlarge", "i2.4xlarge",
"i2.8xlarge", "hi1.4xlarge", "hs1.8xlarge", "cr1.8xlarge", "cc2.8xlarge", "cg1.4xlarge"],
      "ConstraintDescription" : "must be a valid EC2 instance type."
    }
  },

  "Mappings" : {
    "AWSInstanceType2Arch" : {
      "t1.micro"    : { "Arch" : "PV64"   },
      "t2.micro"    : { "Arch" : "HVM64"  },
      "t2.small"    : { "Arch" : "HVM64"  },
      "t2.medium"   : { "Arch" : "HVM64"  },
      "m1.small"    : { "Arch" : "PV64"   },
      "m1.medium"   : { "Arch" : "PV64"   },
      "m1.large"    : { "Arch" : "PV64"   },
      "m1.xlarge"   : { "Arch" : "PV64"   },
      "m2.xlarge"   : { "Arch" : "PV64"   },
      "m2.2xlarge"  : { "Arch" : "PV64"   },
      "m2.4xlarge"  : { "Arch" : "PV64"   },
      "m3.medium"   : { "Arch" : "HVM64"  },
      "m3.large"    : { "Arch" : "HVM64"  },
      "m3.xlarge"   : { "Arch" : "HVM64"  },
      "m3.2xlarge"  : { "Arch" : "HVM64"  },
      "c1.medium"   : { "Arch" : "PV64"   },
      "c1.xlarge"   : { "Arch" : "PV64"   },
      "c3.large"    : { "Arch" : "HVM64"  },
      "c3.xlarge"   : { "Arch" : "HVM64"  },
      "c3.2xlarge"  : { "Arch" : "HVM64"  },
      "c3.4xlarge"  : { "Arch" : "HVM64"  },
      "c3.8xlarge"  : { "Arch" : "HVM64"  },
      "g2.2xlarge"  : { "Arch" : "HVMG2"  },
      "r3.large"    : { "Arch" : "HVM64"  },
      "r3.xlarge"   : { "Arch" : "HVM64"  },
      "r3.2xlarge"  : { "Arch" : "HVM64"  },
      "r3.4xlarge"  : { "Arch" : "HVM64"  },
      "r3.8xlarge"  : { "Arch" : "HVM64"  },
      "i2.xlarge"   : { "Arch" : "HVM64"  },
      "i2.2xlarge"  : { "Arch" : "HVM64"  },
      "i2.4xlarge"  : { "Arch" : "HVM64"  },
      "i2.8xlarge"  : { "Arch" : "HVM64"  },
      "hi1.4xlarge" : { "Arch" : "HVM64"  },
      "hs1.8xlarge" : { "Arch" : "HVM64"  },
      "cr1.8xlarge" : { "Arch" : "HVM64"  },
      "cc2.8xlarge" : { "Arch" : "HVM64"  }
    },

    "AWSRegionArch2AMI" : {
```

```
     "us-east-1"      : { "PV64" : "ami-50842d38", "HVM64" : "ami-08842d60", "HVMG2" :
"ami-3a329952"  },
     "us-west-2"      : { "PV64" : "ami-af86c69f", "HVM64" : "ami-8786c6b7", "HVMG2" :
"ami-47296a77"  },
     "us-west-1"      : { "PV64" : "ami-c7a8a182", "HVM64" : "ami-cfa8a18a", "HVMG2" :
"ami-331b1376"  },
     "eu-west-1"      : { "PV64" : "ami-aa8f28dd", "HVM64" : "ami-748e2903", "HVMG2" :
"ami-00913777"  },
     "ap-southeast-1" : { "PV64" : "ami-20e1c572", "HVM64" : "ami-d6e1c584", "HVMG2" :
"ami-fabe9aa8"  },
     "ap-northeast-1" : { "PV64" : "ami-21072820", "HVM64" : "ami-35072834", "HVMG2" :
"ami-5dd1ff5c"  },
     "ap-southeast-2" : { "PV64" : "ami-8b4724b1", "HVM64" : "ami-fd4724c7", "HVMG2" :
"ami-e98ae9d3"  },
     "sa-east-1"      : { "PV64" : "ami-9d6cc680", "HVM64" : "ami-956cc688", "HVMG2" :
"NOT_SUPPORTED" },
     "cn-north-1"     : { "PV64" : "ami-a857c591", "HVM64" : "ami-ac57c595", "HVMG2" :
"NOT_SUPPORTED" },
     "eu-central-1"   : { "PV64" : "ami-a03503bd", "HVM64" : "ami-b43503a9", "HVMG2" :
"ami-b03503ad"  }
   }
 },

 "Resources" : {

   "ElasticLoadBalancer" : {
     "Type" : "AWS::ElasticLoadBalancing::LoadBalancer",
     "Properties" : {
       "CrossZone" : "true",
       "AvailabilityZones" : { "Fn::GetAZs" : "" },
       "LBCookieStickinessPolicy" : [ {
         "PolicyName" : "CookieBasedPolicy",
         "CookieExpirationPeriod" : "30"
       } ],
       "Listeners" : [ {
         "LoadBalancerPort" : "80",
         "InstancePort" : "80",
         "Protocol" : "HTTP",
         "PolicyNames" : [ "CookieBasedPolicy" ]
       } ],
       "HealthCheck" : {
         "Target" : "HTTP:80/",
         "HealthyThreshold" : "2",
         "UnhealthyThreshold" : "5",
         "Interval" : "10",
         "Timeout" : "5"
       }
     }
   },

   "WebServerGroup" : {
     "Type" : "AWS::AutoScaling::AutoScalingGroup",
     "Properties" : {
       "AvailabilityZones" : { "Fn::GetAZs" : "" },
       "LaunchConfigurationName" : { "Ref" : "LaunchConfig" },
       "MinSize" : "1",
       "DesiredCapacity" : "1",
       "MaxSize" : "5",
       "LoadBalancerNames" : [ { "Ref" : "ElasticLoadBalancer" } ]
     },
     "CreationPolicy" : {
       "ResourceSignal" : {
         "Timeout" : "PT15M"
       }
     },
     "UpdatePolicy": {
```

```
        "AutoScalingRollingUpdate": {
          "MinInstancesInService": "1",
          "MaxBatchSize": "1",
          "PauseTime" : "PT15M",
          "WaitOnResourceSignals": "true"
        }
      }
    },

    "LaunchConfig": {
      "Type" : "AWS::AutoScaling::LaunchConfiguration",
      "Metadata" : {
        "Comment" : "Install a simple PHP application",
        "AWS::CloudFormation::Init" : {
          "config" : {
            "packages" : {
              "yum" : {
                "httpd"             : [],
                "php"               : []
              }
            },

            "files" : {

              "/var/www/html/index.php" : {
                "content" : { "Fn::Join" : ["", [
                  "<?php\n",
                  "echo '<h1>AWS CloudFormation sample PHP application</h1>';\n",
                  "echo 'Updated version via UpdateStack';\n ",
                  "?>\n"
                ]]},
                "mode"    : "000644",
                "owner"   : "apache",
                "group"   : "apache"
              },


              "/etc/cfn/cfn-hup.conf" : {
                "content" : { "Fn::Join" : ["", [
                  "[main]\n",
                  "stack=", { "Ref" : "AWS::StackId" }, "\n",
                  "region=", { "Ref" : "AWS::Region" }, "\n"
                ]]},
                "mode"    : "000400",
                "owner"   : "root",
                "group"   : "root"
              },

              "/etc/cfn/hooks.d/cfn-auto-reloader.conf" : {
                "content": { "Fn::Join" : ["", [
                  "[cfn-auto-reloader-hook]\n",
                  "triggers=post.update\n",
                  "path=Resources.LaunchConfig.Metadata.AWS::CloudFormation::Init\n",
                  "action=/opt/aws/bin/cfn-init -s ", { "Ref" : "AWS::StackId" }, " -r
LaunchConfig ",
                                                   " --region     ", { "Ref" :
"AWS::Region" }, "\n",
                  "runas=root\n"
                ]]}
              }
            },

            "services" : {
              "sysvinit" : {
                "httpd"    : { "enabled" : "true", "ensureRunning" : "true" },
                "cfn-hup" : { "enabled" : "true", "ensureRunning" : "true",
```

```
                    "files" : ["/etc/cfn/cfn-hup.conf", "/etc/cfn/hooks.d/cfn-auto-
reloader.conf"]}
              }
            }
          }
        }
      },

      "Properties": {
        "ImageId" : { "Fn::FindInMap" : [ "AWSRegionArch2AMI", { "Ref" : "AWS::Region" },
                          { "Fn::FindInMap" : [ "AWSInstanceType2Arch", { "Ref" :
 "InstanceType" }, "Arch" ] } ] },
        "InstanceType"   : { "Ref" : "InstanceType" },
        "KeyName"        : { "Ref" : "KeyName" },
        "SecurityGroups" : [ {"Ref" : "WebServerSecurityGroup"} ],
        "UserData"       : { "Fn::Base64" : { "Fn::Join" : ["", [
             "#!/bin/bash -xe\n",
             "yum install -y aws-cfn-bootstrap\n",

             "# Install the files and packages from the metadata\n",
             "/opt/aws/bin/cfn-init -v ",
             "         --stack ", { "Ref" : "AWS::StackName" },
             "         --resource LaunchConfig ",
             "         --region ", { "Ref" : "AWS::Region" }, "\n",

             "# Start up the cfn-hup daemon to listen for changes to the Web Server
 metadata\n",
             "/opt/aws/bin/cfn-hup || error_exit 'Failed to start cfn-hup'\n",

             "# Signal the status from cfn-init\n",
             "/opt/aws/bin/cfn-signal -e $? ",
             "         --stack ", { "Ref" : "AWS::StackName" },
             "         --resource WebServerGroup ",
             "         --region ", { "Ref" : "AWS::Region" }, "\n"
        ]]}}
      }
    },

    "WebServerSecurityGroup" : {
      "Type" : "AWS::EC2::SecurityGroup",
      "Properties" : {
        "GroupDescription" : "Enable HTTP access via port 80 locked down to the ELB and SSH
 access",
        "SecurityGroupIngress" : [
          {"IpProtocol" : "tcp", "FromPort" : "80", "ToPort" : "80",
 "SourceSecurityGroupOwnerId" : {"Fn::GetAtt" : ["ElasticLoadBalancer",
 "SourceSecurityGroup.OwnerAlias"]},"SourceSecurityGroupName" : {"Fn::GetAtt" :
 ["ElasticLoadBalancer", "SourceSecurityGroup.GroupName"]}},
          {"IpProtocol" : "tcp", "FromPort" : "22", "ToPort" : "22", "CidrIp" : { "Ref" :
 "SSHLocation"}}
        ]
      }
    }
  },

  "Outputs" : {
    "WebsiteURL" : {
      "Description" : "Application URL",
      "Value" : { "Fn::Join" : ["", ["http://", { "Fn::GetAtt" : [ "ElasticLoadBalancer",
 "DNSName" ]}]] }
    }
  }
}
```

Availability and Impact Considerations

Different properties have different impacts on the resources in the stack. You can use AWS CloudFormation to update any property; however, before you make any changes, you should consider these questions:

1. How does the update affect the resource itself? For example, updating an alarm threshold will render the alarm inactive during the update. As we have seen, changing the instance type requires that the instance be stopped and restarted. AWS CloudFormation uses the Update or Modify actions for the underlying resources to make changes to resources. To understand the impact of updates, you should check the documentation for the specific resources.
2. Is the change mutable or immutable? Some changes to resource properties, such as changing the AMI on an Amazon EC2 instance, are not supported by the underlying services. In the case of mutable changes, AWS CloudFormation will use the Update or Modify type APIs for the underlying resources. For immutable property changes, AWS CloudFormation will create new resources with the updated properties and then link them to the stack before deleting the old resources. Although AWS CloudFormation tries to reduce the down time of the stack resources, replacing a resource is a multistep process, and it will take time. During stack reconfiguration, your application will not be fully operational. For example, it may not be able to serve requests or access a database.

Related Resources

For more information about using AWS CloudFormation to start applications and on integrating with other configuration and deployment services such as Puppet and Opscode Chef, see the following whitepapers:

- Bootstrapping Applications via AWS CloudFormation
- Integrating AWS CloudFormation with Opscode Chef
- Integrating AWS CloudFormation with Puppet

The template used throughout this section is a "Hello, World" PHP application. The template library also has an Amazon ElastiCache sample template that shows how to integrate a PHP application with ElasticCache using cfn-hup and cfn-init to respond to changes in the Amazon ElastiCache Cache Cluster configuration, all of which can be performed by Update Stack.

AWS CloudFormation Best Practices

Best practices are recommendations that can help you use AWS CloudFormation more effectively and securely throughout its entire workflow. Learn how to plan and organize your stacks, create templates that describe your resources and the software applications that run on them, and manage your stacks and their resources. The following best practices are based on real-world experience from current AWS CloudFormation customers.

Planning and organizing

Creating templates

Managing stacks

Organize Your Stacks By Lifecycle and Ownership

Use the lifecycle and ownership of your AWS resources to help you decide what resources should go in each stack. Normally, you might put all your resources in one stack, but as your stack grows in scale and broadens in scope, managing a single stack can be cumbersome and time consuming. By grouping resources with common lifecycles and ownership, owners can make changes to their set of resources by using their own process and schedule without affecting other resources.

For example, imagine a team of developers and engineers who own a website that is hosted on autoscaling instances behind a load balancer. Because the website has its own lifecycle and is maintained by the website team, you can create a stack for the website and its resources. Now imagine that the website also uses back-end databases, where the databases are in a separate stack that are owned and maintained by database administrators. Whenever the website team or database team needs to update their resources, they can do so without affecting each other's stack. If all resources were in a single stack, coordinating and communicating updates can be difficult.

For additional guidance about organizing your stacks, you can use two common frameworks: a multi-layered architecture and service-oriented architecture (SOA).

A layered architecture organizes stacks into multiple horizontal layers that build on top of one another, where each layer has a dependency on the layer directly below it. You can have one or more stacks in each layer, but within each layer, your stacks should have AWS resources with similar lifecycles and ownership.

With a service-oriented architecture, you can organize big business problems into manageable parts. Each of these parts is a service that has a clearly defined purpose and represents a self-contained unit of functionality. You can map these services to a stack, where each stack has its own lifecycle and owners. All of these services (stacks) can be wired together so that they can interact with one another.

Use Cross-Stack References to Export Shared Resources

When you organize your AWS resources based on lifecycle and ownership, you might want to build a stack that uses resources that are in another stack. You can hard-code values or use input parameters to pass resource names and IDs. However, these methods can make templates difficult to reuse or can increase the overhead to get a stack running. Instead, use cross-stack references to export resources from a stack so that other stacks can use them. Stacks can use the exported resources by calling them using the `Fn::ImportValue` function.

For example, you might have a network stack that includes a VPC, a security group, and a subnet. You want all public web applications to use these resources. By exporting the resources, you allow all stacks with public web applications to use them. For more information, see Walkthrough: Refer to Resource Outputs in Another AWS CloudFormation Stack (p. 200).

Use IAM to Control Access

IAM is an AWS service that you can use to manage users and their permissions in AWS. You can use IAM with AWS CloudFormation to specify what AWS CloudFormation actions users can perform, such as viewing stack templates, creating stacks, or deleting stacks. Furthermore, anyone managing AWS CloudFormation stacks will require permissions to resources within those stacks. For example, if users want to use AWS CloudFormation to launch, update, or terminate Amazon EC2 instances, they must have permission to call the relevant Amazon EC2 actions.

In most cases, users require full access to manage all of the resources in a template. AWS CloudFormation makes calls to create, modify, and delete those resources on their behalf. To separate permissions between a user and the AWS CloudFormation service, use a service role. AWS CloudFormation uses the service role's policy to make calls instead of the user's policy. For more information, see AWS CloudFormation Service Role (p. 13).

Verify Quotas for All Resource Types

Before launching a stack, ensure that you can create all the resources that you want without hitting your AWS account limits. If you hit a limit, AWS CloudFormation won't create your stack successfully until you increase your quota or delete extra resources. Each service can have various limits that you should be aware of before launching a stack. For example, by default, you can only launch 200 AWS CloudFormation stacks per region in your AWS account. For more information about limits and how to increase the default limits, see AWS Service Limits in the *AWS General Reference*.

Reuse Templates to Replicate Stacks in Multiple Environments

After you have your stacks and resources set up, you can reuse your templates to replicate your infrastructure in multiple environments. For example, you can create environments for development, testing, and production so that you can test changes before implementing them into production. To make templates reusable, use the parameters, mappings, and conditions sections so that you can customize your stacks when you create them. For example, for your development environments, you can specify a lower-cost instance type compared to your production environment, but all other configurations and settings remain the same. For more information about parameters, mappings, and conditions, see Template Anatomy (p. 128).

Use Nested Stacks to Reuse Common Template Patterns

As your infrastructure grows, common patterns can emerge in which you declare the same components in each of your templates. You can separate out these common components and create dedicated templates for them. That way, you can mix and match different templates but use nested stacks to create a single, unified stack. Nested stacks are stacks that create other stacks. To create nested stacks, use the AWS::CloudFormation::Stack (p. 485) resource in your template to reference other templates.

For example, assume that you have a load balancer configuration that you use for most of your stacks. Instead of copying and pasting the same configurations into your templates, you can create a dedicated template for the load balancer. Then, you just use the AWS::CloudFormation::Stack (p. 485) resource to reference that template from within other templates. If the load balancer template is updated, any stack that is referencing it will use the updated load balancer (only after you update the stack). In addition to simplifying updates, this approach lets you use experts to create and maintain components that you might not be necessarily familiar with. All you need to do is reference their templates.

Do Not Embed Credentials in Your Templates

Rather than embedding sensitive information in your AWS CloudFormation templates, use input parameters to pass in information whenever you create or update a stack. If you do, make sure to use the `NoEcho` property to obfuscate the parameter value.

For example, suppose your stack creates a new database instance. When the database is created, AWS CloudFormation needs to pass a database administrator password. You can pass in a password by using an input parameter instead of embedding it in your template. For more information, see Parameters (p. 132).

Use AWS-Specific Parameter Types

If your template requires inputs for existing AWS-specific values, such as existing Amazon Virtual Private Cloud IDs or an Amazon EC2 key pair name, use AWS-specific parameter types. For example, you can specify a parameter as type `AWS::EC2::KeyPair::KeyName`, which takes an existing key pair name that is in your AWS account and in the region where you are creating the stack. AWS CloudFormation can quickly validate values for AWS-specific parameter types before creating your stack. Also, if you use the AWS CloudFormation console, AWS CloudFormation shows a drop-down list of valid values, so you don't have to look up or memorize the correct VPC IDs or key pair names. For more information, see Parameters (p. 132).

Use Parameter Constraints

With constraints, you can describe allowed input values so that AWS CloudFormation catches any invalid values before creating a stack. You can set constraints such as a minimum length, maximum length, and allowed patterns. For example, you can set constraints on a database user name value so that it must be a minimum length of eight character and contain only alpha-numeric characters. For more information, see Parameters (p. 132).

Use AWS::CloudFormation::Init to Deploy Software Applications on Amazon EC2 Instances

When you launch stacks, you can install and configure software applications on Amazon EC2 instances by using the cfn-init helper script and the `AWS::CloudFormation::Init` resource. By using `AWS::CloudFormation::Init`, you can describe the configurations that you want rather than scripting procedural steps. You can also update configurations without recreating instances. And if anything goes wrong with your configuration, AWS CloudFormation generates logs that you can use to investigate issues.

In your template, specify installation and configuration states in the `AWS::CloudFormation::Init` (p. 470) resource. For a walkthrough that shows how to use cfn-init and `AWS::CloudFormation::Init`, see Deploying Applications on Amazon EC2 with AWS CloudFormation (p. 212).

Use the Latest Helper Scripts

The helper scripts (p. 1354) are updated periodically. Be sure you include the following command in the `UserData` property of your template before you call the helper scripts to ensure that your launched instances get the latest helper scripts:

```
yum install -y aws-cfn-bootstrap
```

For more information about getting the latest helper scripts, see the CloudFormation Helper Scripts Reference (p. 1354).

Validate Templates Before Using Them

Before you use a template to create or update a stack, you can use AWS CloudFormation to validate it. Validating a template can help you catch syntax and some semantic errors, such as circular dependencies, before AWS CloudFormation creates any resources. If you use the AWS CloudFormation console, the console automatically validates the template after you specify input parameters. For the AWS CLI or AWS CloudFormation API, use the aws cloudformation validate-template command or ValidateTemplate action.

During validation, AWS CloudFormation first checks if the template is valid JSON. If it isn't, AWS CloudFormation checks if the template is valid YAML. If both checks fail, AWS CloudFormation returns a template validation error.

Manage All Stack Resources Through AWS CloudFormation

After you launch a stack, use the AWS CloudFormation console, API, or AWS CLI to update resources in your stack. Do not make changes to stack resources outside of AWS CloudFormation. Doing so can create a mismatch between your stack's template and the current state of your stack resources, which can cause errors if you update or delete the stack. For more information, see Walkthrough: Updating a Stack (p. 29).

Create Change Sets Before Updating Your Stacks

Change sets allow you to see how proposed changes to a stack might impact your running resources before you implement them. AWS CloudFormation doesn't make any changes to your stack until you execute the change set, allowing you to decide whether to proceed with your proposed changes or create another change set.

Use change sets to check how your changes might impact your running resources, especially for critical resources. For example, if you change the name of an Amazon RDS database instance, AWS CloudFormation will create a new database and delete the old one; you will lose the data in the old database unless you've already backed it up. If you generate a change set, you will see that your change will replace your database. This can help you plan before you update your stack. For more information, see Updating Stacks Using Change Sets (p. 93).

Use Stack Policies

Stack policies help protect critical stack resources from unintentional updates that could cause resources to be interrupted or even replaced. A stack policy is a JSON document that describes what update actions can be performed on designated resources. Specify a stack policy whenever you create a stack that has critical resources.

During a stack update, you must explicitly specify the protected resources that you want to update; otherwise, no changes are made to protected resources. For more information, see Prevent Updates to Stack Resources (p. 110).

Use AWS CloudTrail to Log AWS CloudFormation Calls

AWS CloudTrail tracks anyone making AWS CloudFormation API calls in your AWS account. API calls are logged whenever anyone uses the AWS CloudFormation API, the AWS CloudFormation console, a back-end console, or AWS CloudFormation AWS CLI commands. Enable logging and specify an

Amazon S3 bucket to store the logs. That way, if you ever need to, you can audit who made what AWS CloudFormation call in your account. For more information, see Logging AWS CloudFormation API Calls in AWS CloudTrail (p. 13).

Use Code Reviews and Revision Controls to Manage Your Templates

Your stack templates describe the configuration of your AWS resources, such as their property values. To review changes and to keep an accurate history of your resources, use code reviews and revision controls. These methods can help you track changes between different versions of your templates, which can help you track changes to your stack resources. Also, by maintaining a history, you can always revert your stack to a certain version of your template.

Update Your Amazon EC2 Linux Instances Regularly

On all your Amazon EC2 Linux instances and Amazon EC2 Linux instances created with AWS CloudFormation, regularly run the `yum update` command to update the RPM package. This ensures that you get the latest fixes and security updates.

Continuous Delivery with AWS CodePipeline

Continuous delivery is a release practice in which code changes are automatically built, tested, and prepared for release to production. With AWS CloudFormation and AWS CodePipeline, you can use continuous delivery to automatically build and test changes to your AWS CloudFormation templates before promoting them to production stacks. This release process lets you rapidly and reliably make changes to your AWS infrastructure.

For example, you can create a workflow that automatically builds a test stack when you submit an updated template to a code repository. After AWS CloudFormation builds the test stack, you can test it and then decide whether to push the changes to a production stack. For more information about the benefits of continuous delivery, see What is Continuous Delivery?.

Use AWS CodePipeline to build a continuous delivery workflow by building a pipeline for AWS CloudFormation stacks. AWS CodePipeline has built-in integration with AWS CloudFormation, so you can specify AWS CloudFormation-specific actions, such as creating, updating, or deleting a stack, within a pipeline. For more information about AWS CodePipeline, see the AWS CodePipeline User Guide.

Topics

Walkthrough: Building a Pipeline for Test and Production Stacks

Imagine a release process where you submit an AWS CloudFormation template, which AWS CloudFormation then uses to automatically build a test stack. After you review the test stack, you can preview how your changes will modify your production stack, and then choose whether to implement them. To accomplish this workflow, you could use AWS CloudFormation to build your test stack, delete the test stack, create a change set, and then execute the change set. However, with each action, you need to manually interact with AWS CloudFormation. In this walkthrough, we'll build an AWS CodePipeline pipeline that automates many of these actions, helping you achieve a continuous delivery workflow with your AWS CloudFormation stacks.

Prerequisites

This walkthrough assumes that you have used AWS CodePipeline and AWS CloudFormation, and know how pipelines and AWS CloudFormation templates and stacks work. For more information about AWS CodePipeline, see the AWS CodePipeline User Guide. You also need to have an Amazon S3 bucket in the same AWS region in which you will create your pipeline.

> **Important**
> The sample Word Press template creates an EC2 instance that requires a connection to the Internet. Check that you have a default VPC and subnet that allow traffic to the Internet.

Walkthrough Overview

This walkthrough builds a pipeline for a sample WordPress site in a stack. The pipeline is separated into three stages. Each stage must contain at least one action, which is a task the pipeline performs on your artifacts (your input). A stage organizes actions in a pipeline. AWS CodePipeline must complete all actions in a stage before the stage processes new artifacts, for example, if you submitted new input to rerun the pipeline.

By the end of this walkthrough, you'll have a pipeline that performs the following workflow:

1. The first stage of the pipeline retrieves a source artifact (an AWS CloudFormation template and its configuration files) from a repository.

   You'll prepare an artifact that includes a sample WordPress template and upload it to an S3 bucket.
2. In the second stage, the pipeline creates a test stack and then waits for your approval.

   After you review the test stack, you can choose to continue with the original pipeline or create and submit another artifact to make changes. If you approve, this stage deletes the test stack, and then the pipeline continues to the next stage.
3. In the third stage, the pipeline creates a change set against a production stack, and then waits for your approval.

   In your initial run, you won't have a production stack. The change set shows you all of the resources that AWS CloudFormation will create. If you approve, this stage executes the change set and builds your production stack.

> **Note**
> AWS CloudFormation is a free service. However, you are charged for the AWS resources, such as the EC2 instance, that you include in your stack at the current rate for each. For more information about AWS pricing, see the detail page for each product at http://aws.amazon.com.

Step 1: Edit the Artifact and Upload It to an S3 Bucket

Before you build your pipeline, you must set up your source repository and files. AWS CodePipeline copies these source files into your pipeline's artifact store, and then uses them to perform actions in your pipeline, such as creating an AWS CloudFormation stack.

When you use Amazon Simple Storage Service (Amazon S3) as the source repository, AWS CodePipeline requires you to zip your source files before uploading them to an S3 bucket. The zipped file is an AWS CodePipeline artifact that can contain an AWS CloudFormation template, a template configuration file, or both. We provide an artifact that contains a sample WordPress template and two template configuration files. The two configuration files specify parameter values for the WordPress template. AWS CodePipeline uses them when it creates the WordPress stacks. One file contains parameter values for a test stack, and the other for a production stack. You'll need to edit the configuration files, for example, to specify an existing EC2 key-pair name that you own. For more information about artifacts, see AWS CloudFormation Artifacts (p. 67).

After you build your artifact, you'll upload it to an S3 bucket.

To edit and upload the artifact

1. Download and open the sample artifact: https://s3.amazonaws.com/cloudformation-examples/user-guide/continuous-deployment/wordpress-single-instance.zip.

   The artifact contains three files:

   - The sample WordPress template: `wordpress-single-instance.yaml`
   - The template configuration file for the test stack.: `test-stack-configuration.json`
   - The template configuration file for the production stack: `prod-stack-configuration.json`
2. Extract all of the files, and then use any text editor to modify the template configuration files.

   Open the configuration files to see that they contain key-value pairs that map to the WordPress template's parameters. The configuration files specify the parameter values that your pipeline uses when it creates the test and production stacks.

   Edit the `test-stack-configuration.json` file to specify parameter values for the test stack and the `prod-stack-configuration.json` file for the production stack.

   - Change the values of the `DBPassword` and `DBRootPassword` keys to passwords that you can use to log in to your WordPress database. As defined in the WordPress template, the parameter values must contain only alphanumeric characters.
   - Change the value of the `KeyName` key to an existing EC2 key-pair name in the region in which you will create your pipeline.
3. Add the modified configuration files to the original artifact (`.zip`) file, replacing duplicate files.

   You now have a customized artifact that you can upload to an S3 bucket.
4. Upload the artifact to an S3 bucket that you own.

   Note the file's location. You'll specify the location of this file when you build your pipeline.

   Notes about the artifact and S3 bucket:

   - Use a bucket that is in the same AWS region in which you will create your pipeline.
   - AWS CodePipeline requires that the bucket is versioning enabled.
   - You can also use services that don't require you to zip your files before uploading them, like GitHub or AWS CodeCommit, for your source repository.
   - Artifacts can contain sensitive information such as passwords. Limit access so that only permitted users can view the file. When you do, ensure that AWS CodePipeline can still access the file.

You now have an artifact that AWS CodePipeline can pull in to your pipeline. In the next step, you'll specify the artifact's location and build the WordPress pipeline.

Step 2: Create the Pipeline Stack

To create the WordPress pipeline, you'll use a sample AWS CloudFormation template. In addition to building the pipeline, the template sets up AWS Identity and Access Management (IAM) service roles for AWS CodePipeline and AWS CloudFormation, an S3 bucket for the AWS CodePipeline artifact store, and an Amazon Simple Notification Service (Amazon SNS) topic to which the pipeline sends notifications, such as notifications about reviews. The sample template makes it easy to provision and configure these resources in a single AWS CloudFormation stack.

For more details about the configuration of the pipeline, see What the Pipeline Does (p. 59).

Important
The sample WordPress template creates an EC2 instance that requires a connection to the Internet. Check that your default VPC and subnet allow traffic to the Internet.

To create the pipeline stack

1. Download the sample template at https://s3.amazonaws.com/cloudformation-examples/user-guide/continuous-deployment/basic-pipeline.yml. Save it on your computer.
2. Open the AWS CloudFormation console at https://console.aws.amazon.com/cloudformation/.
3. Choose an AWS region that supports AWS CodePipeline and AWS CloudFormation.

   For more information, see AWS Regions and Endpoints in the *AWS General Reference*.
4. Choose **Create Stack**.
5. In the **Template** section, choose **Upload a template to Amazon S3**, and then choose the template that you just downloaded, `basic-pipeline.yml`.
6. Choose **Next**.
7. For **Stack name**, type `sample-WordPress-pipeline`.
8. In the **Parameters** section, specify the following parameter values, and then choose **Next**. When setting stack parameters, if you kept the same names for the WordPress template and its configuration files, you can use the default values. If not, specify the filenames that you used.

   PipelineName

      The name of your pipeline, such as `WordPress-test-pipeline`.

   S3Bucket

      The name of the S3 bucket where you saved your artifact (`.zip` file).

   SourceS3Key

      The filename of your artifact. If you saved the artifact in a folder, include it as part of the filename, such as `folder/subfolder/wordpress-single-instance.zip`.

   Email

      The email address to which AWS CodePipeline sends pipeline notification, such as `myemail@example.com`.
9. For this walkthrough, you don't need to add tags or specify advanced settings, so choose **Next**.
10. Ensure that the stack name and template URL are correct, and then choose **Create**.

It might take several minutes for AWS CloudFormation to create your stack. To monitor progress, view the stack events. For more information, see Viewing Stack Data and Resources (p. 78).

After your stack has been created, AWS CodePipeline starts your new pipeline. To view its status, see the AWS CodePipeline console. From the list of pipelines, choose **sample-WordPress-pipeline**.

What the Pipeline Does

This section explains the pipeline's three stages, using snippets from the sample WordPress pipeline template.

Stage 1: Source

The first stage of the pipeline is a source stage in which you specify the location of your source code. Every time you push a revision to this location, AWS CodePipeline reruns your pipeline.

The source code is located in an S3 bucket and is identified by its filename. You specified these values as input parameter values when you created the pipeline stack. To allow using the source artifact in subsequent stages, the snippet specifies the `OutputArtifacts` property, with the name `TemplateSource`. To use this artifact in later stages, you specify `TemplateSource` as an input artifact.

```
- Name: S3Source
  Actions:
    - Name: TemplateSource
      ActionTypeId:
        Category: Source
        Owner: AWS
        Provider: S3
        Version: '1'
      Configuration:
        S3Bucket: !Ref 'S3Bucket'
        S3ObjectKey: !Ref 'SourceS3Key'
      OutputArtifacts:
        - Name: TemplateSource
```

Stage 2: TestStage

In the `TestStage` stage, the pipeline creates the test stack, waits for approval, and then deletes the test stack.

For the `CreateStack` action, the pipeline uses the test configuration file and WordPress template to create the test stack. Both files are contained in the `TemplateSource` input artifact, which is brought in from the source stage. The snippet uses the `REPLACE_ON_FAILURE` action mode. If stack creation fails, the pipeline replaces it so that you don't need to clean up or troubleshoot the stack before you can rerun the pipeline. The action mode is useful for quickly iterating on test stacks. For the `RoleArn` property, the value is an AWS CloudFormation service role that is declared elsewhere in the template.

The `ApproveTestStack` action pauses the pipeline and sends a notification to the email address that you specified when you created the pipeline stack. While the pipeline is paused, you can check the WordPress test stack and its resources. Use AWS CodePipeline to approve or reject this action. The `CustomData` property includes a description of the action you're approving, which the pipeline adds to the notification email.

After you approve this action, AWS CodePipeline moves to the `DeleteTestStack` action and deletes the test WordPress stack and its resources.

```
- Name: TestStage
  Actions:
    - Name: CreateStack
      ActionTypeId:
        Category: Deploy
        Owner: AWS
        Provider: CloudFormation
        Version: '1'
      InputArtifacts:
        - Name: TemplateSource
      Configuration:
        ActionMode: REPLACE_ON_FAILURE
        RoleArn: !GetAtt [CFNRole, Arn]
        StackName: !Ref TestStackName
        TemplateConfiguration: !Sub "TemplateSource::${TestStackConfig}"
        TemplatePath: !Sub "TemplateSource::${TemplateFileName}"
      RunOrder: '1'
    - Name: ApproveTestStack
      ActionTypeId:
        Category: Approval
        Owner: AWS
```

```
        Provider: Manual
        Version: '1'
      Configuration:
        NotificationArn: !Ref CodePipelineSNSTopic
        CustomData: !Sub 'Do you want to create a change set against the production stack
 and delete the ${TestStackName} stack?'
      RunOrder: '2'
    - Name: DeleteTestStack
      ActionTypeId:
        Category: Deploy
        Owner: AWS
        Provider: CloudFormation
        Version: '1'
      Configuration:
        ActionMode: DELETE_ONLY
        RoleArn: !GetAtt [CFNRole, Arn]
        StackName: !Ref TestStackName
      RunOrder: '3'
```

Stage 3: ProdStage

The `ProdStage` stage of the pipeline creates a change set against the existing production stack, waits for approval, and then executes the change set.

A change set provides a preview of all modifications AWS CloudFormation will make to your production stack before implementing them. On your first pipeline run, you won't have a running production stack. The change set shows the actions that AWS CloudFormation performed when creating the test stack. To create the change set, the `CreateChangeSet` action uses the WordPress sample template and the production template configuration from the `TemplateSource` input artifact.

Similar to the previous stage, the `ApproveChangeSet` action pauses the pipeline and sends an email notification. While the pipeline is paused, you can view the change set to check all of the proposed modifications to the production WordPress stack. Use AWS CodePipeline to approve or reject this action to continue or stop the pipeline, respectively.

After you approve this action, the `ExecuteChangeSet` action executes the changes set, so that AWS CloudFormation performs all of the actions described in the change set. For the initial run, AWS CloudFormation creates the WordPress production stack. On subsequent runs, AWS CloudFormation updates the stack.

```
- Name: ProdStage
  Actions:
    - Name: CreateChangeSet
      ActionTypeId:
        Category: Deploy
        Owner: AWS
        Provider: CloudFormation
        Version: '1'
      InputArtifacts:
        - Name: TemplateSource
      Configuration:
        ActionMode: CHANGE_SET_REPLACE
        RoleArn: !GetAtt [CFNRole, Arn]
        StackName: !Ref ProdStackName
        ChangeSetName: !Ref ChangeSetName
        TemplateConfiguration: !Sub "TemplateSource::${ProdStackConfig}"
        TemplatePath: !Sub "TemplateSource::${TemplateFileName}"
      RunOrder: '1'
    - Name: ApproveChangeSet
      ActionTypeId:
        Category: Approval
        Owner: AWS
        Provider: Manual
```

```
        Version: '1'
      Configuration:
        NotificationArn: !Ref CodePipelineSNSTopic
        CustomData: !Sub 'A new change set was created for the ${ProdStackName} stack. Do
 you want to implement the changes?'
      RunOrder: '2'
    - Name: ExecuteChangeSet
      ActionTypeId:
        Category: Deploy
        Owner: AWS
        Provider: CloudFormation
        Version: '1'
      Configuration:
        ActionMode: CHANGE_SET_EXECUTE
        ChangeSetName: !Ref ChangeSetName
        RoleArn: !GetAtt [CFNRole, Arn]
        StackName: !Ref ProdStackName
      RunOrder: '3'
```

Step 3: View the WordPress Stack

As AWS CodePipeline runs through the pipeline, it uses AWS CloudFormation to create test and production stacks. To see the status of these stacks and their output, use the AWS CloudFormation console.

To view a stack

1. Open the AWS CloudFormation console at https://console.aws.amazon.com/cloudformation/.
2. Depending on whether your pipeline is in the test or production stage, choose the `Test-MyWordPressSite` or the `Prod-MyWordPressSite` stack.
3. To check the status of your stack, view the stack events (p. 78).

If the stack is in a failed state, view the status reason to find the stack error. Fix the error, and then rerun the pipeline. If the stack is in the `CREATE_COMPLETE` state, view its outputs to get the URL of your WordPress site.

You've successfully used AWS CodePipeline to build a continuous delivery workflow for a sample WordPress site. If you submit changes to the S3 bucket, AWS CodePipeline automatically detects a new version, and then reruns your pipeline. This workflow makes it easier to submit and test changes before making changes to your production site.

Step 4: Clean Up Resources

To make sure that you are not charged for unwanted services, delete your resources.

> **Important**
> Delete the test and production WordPress stacks before deleting the pipeline stack. The pipeline stack contains a service role that's required to delete the WordPress stacks. If you deleted the pipeline stack first, you can associate another service role Amazon Resource Name (ARN) with the WordPress stacks, and then delete them.

To delete objects in the artifact store

1. Open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. Choose the S3 bucket that AWS CodePipeline used as your pipeline's artifact store.

   The bucket's name follows the format: *`stackname-artifactstorebucket-id`*. If you followed this walkthrough, the bucket's name might look similar to the following example: `sample-WordPress-pipeline-artifactstorebucket-12345abcd12345`.

3. Delete all of the objects in the artifact store S3 bucket.

   When you delete the pipeline stack in the next step, this bucket must be empty. Otherwise, AWS CloudFormation won't be able to delete the bucket.

To delete stacks

1. From the AWS CloudFormation console, choose the stack that you want to delete.

   If the WordPress stacks that were created by the pipeline are still running, choose them first. By default, the stack names are `Test-MyWordPressSite` and `Prod-MyWordPressSite`.

   If you already deleted the WordPress stacks, choose the `sample-WordPress-pipeline` stack.
2. Choose **Actions**, and then choose **Delete Stack**.
3. In the confirmation message, choose **Yes, Delete**.

AWS CloudFormation deletes the stack all of the stack's resources, such as the EC2 instance, notification topic, service role, and the pipeline.

Now that you understand how to build a basic AWS CloudFormation workflow with AWS CodePipeline, you can use the sample template and artifacts as a starting point for building your own.

AWS CloudFormation Configuration Properties Reference

When you build an AWS CodePipeline pipeline, you add a `Deploy` action to the pipeline with AWS CloudFormation as a provider. You then must specify which AWS CloudFormation action the pipeline invokes and the action's settings. This topic describes the AWS CloudFormation configuration properties. To specify properties, you can use the AWS CodePipeline console, or you can create a JSON object to use for the AWS CLI, AWS CodePipeline API, or AWS CloudFormation templates.

Topics

Configuration Properties (Console)

The AWS CodePipeline console shows the configuration properties and indicates the properties that are required based on the `Action mode` that you choose.

> **Note**
> When you create a new pipeline, you can specify only the **Create or update a stack** or **Create or replace a change set** action modes. Also, properties in the **Advanced** section are available only when you edit an existing pipeline.

Action mode

The AWS CloudFormation action that AWS CodePipeline invokes when processing the associated stage. Choose one of the following action modes:

- **Create or replace a change set** creates the change set if it doesn't exist based on the stack name and template that you submit. If the change set exists, AWS CloudFormation deletes it, and then creates a new one.

- **Create or update a stack** creates the stack if the specified stack doesn't exist. If the stack exists, AWS CloudFormation updates the stack. Use this action to update existing stacks. AWS CodePipeline won't replace the stack.
- **Delete a stack** deletes a stack. If you specify a stack that doesn't exist, the action completes successfully without deleting a stack.
- **Execute a change set** executes a change set.
- **Replace a failed stack** creates the stack if the specified stack doesn't exist. If the stack exists and is in a failed state (reported as `ROLLBACK_COMPLETE`, `ROLLBACK_FAILED`, `CREATE_FAILED`, `DELETE_FAILED`, or `UPDATE_ROLLBACK_FAILED`), AWS CloudFormation deletes the stack and then creates a new stack. If the stack isn't in a failed state, AWS CloudFormation updates it. Use this action to automatically replace failed stacks without recovering or troubleshooting them. You would typically choose this mode for testing.

Stack name

The name of an existing stack or a stack that you want to create.

Change set name

The name of an existing change set or a new change set that you want to create for the specified stack.

Template

The location of an AWS CloudFormation template file, which follows the format `ArtifactName::TemplateFileName`.

Template configuration

The location of a template configuration file, which follows the format `ArtifactName::TemplateConfigurationFileName`. The template configuration file can contain template parameter values and a stack policy. If you include sensitive information, such as passwords, restrict access to this file. For more information, see AWS CloudFormation Artifacts (p. 67).

Capabilities

For stacks that contain certain resources, explicit acknowledgement that AWS CloudFormation might create or update those resources. For example, you must specify `CAPABILITY_IAM` if your stack template contains AWS Identity and Access Management (IAM) resources. For more information, see Acknowledging IAM Resources in AWS CloudFormation Templates (p. 11).

If you have IAM resources in your stack template, you must specify this property.

Role name

The name of the IAM service role that AWS CloudFormation assumes when it operates on resources in the specified stack.

Output file name

In the **Advanced** section, you can specify an output file name, such as `CreateStackOutput.json`, that AWS CodePipeline adds to the output artifact after performing the specified action.

If you don't specify a name, AWS CodePipeline doesn't generate an output artifact.

Parameter overrides

In the **Advanced** section, you can specify a JSON object that overrides template parameter values in the template configuration file. All parameter names must be present in the stack template.

We recommend that you use the template configuration file to specify most of your parameter values. Use parameter overrides to specify only dynamic parameter values (values that are unknown until you run the pipeline).

The following example defines a value for the `ParameterName` parameter by using a parameter override function. The function retrieves a value from an AWS CodePipeline input artifact. For more information about parameter override functions, see Using Parameter Override Functions with AWS CodePipeline Pipelines (p. 68).

```
{
  "ParameterName" : { "Fn::GetParam" : ["ArtifactName", "config-file-name.json",
 "ParamName"]}
}
```

Configuration Properties (JSON Object)

When you specify `CloudFormation` as a provider for a stage action, define the following properties within the `Configuration` property. Use the JSON object for the AWS CLI, AWS CodePipeline API, or AWS CloudFormation templates. For examples, see Walkthrough: Building a Pipeline for Test and Production Stacks (p. 56)

`ActionMode`

The AWS CloudFormation action that AWS CodePipeline invokes when processing the associated stage. Specify only one of the following action modes:

- `CHANGE_SET_EXECUTE` executes a change set.
- `CHANGE_SET_REPLACE` creates the change set if it doesn't exist based on the stack name and template that you submit. If the change set exists, AWS CloudFormation deletes it, and then creates a new one.
- `CREATE_UPDATE` creates the stack if the specified stack doesn't exist. If the stack exists, AWS CloudFormation updates the stack. Use this action to update existing stacks. AWS CodePipeline won't replace the stack.
- `DELETE_ONLY` deletes a stack. If you specify a stack that doesn't exist, the action completes successfully without deleting a stack.
- `REPLACE_ON_FAILURE` creates a stack if the specified stack doesn't exist. If the stack exists and is in a failed state (reported as `ROLLBACK_COMPLETE`, `ROLLBACK_FAILED`, `CREATE_FAILED`, `DELETE_FAILED`, or `UPDATE_ROLLBACK_FAILED`), AWS CloudFormation deletes the stack and then creates a new stack. If the stack isn't in a failed state, AWS CloudFormation updates it. Use this action to automatically replace failed stacks without recovering or troubleshooting them. You would typically choose this mode for testing.

This property is required.

`Capabilities`

For stacks that contain certain resources, explicit acknowledgement that AWS CloudFormation might create or update those resources. For example, you must specify `CAPABILITY_IAM` if your stack template contains AWS Identity and Access Management (IAM) resources. For more information, see Acknowledging IAM Resources in AWS CloudFormation Templates (p. 11).

This property is conditional. If you have IAM resources in your stack template, you must specify this property.

`ChangeSetName`

The name of an existing change set or a new change set that you want to create for the specified stack.

This property is required for the following action modes: `CHANGE_SET_REPLACE` and `CHANGE_SET_EXECUTE`. For all other action modes, this property is ignored.

`OutputFileName`

A name for the output file, such as `CreateStackOutput.json`. AWS CodePipeline adds the file to the output artifact after performing the specified action.

This property is optional. If you don't specify a name, AWS CodePipeline doesn't generate an output artifact.

`ParameterOverrides`

A JSON object that specifies values for template parameters. If you specify parameters that are also specified in the template configuration file, these values override them. All parameter names must be present in the stack template.

We recommend that you use the template configuration file to specify most of your parameter values. Use parameter overrides to specify only dynamic parameter values (values that are unknown until you run the pipeline).

The following example defines a value for the `ParameterName` parameter by using a parameter override function. The function retrieves a value from an AWS CodePipeline input artifact. For more information about parameter override functions, see Using Parameter Override Functions with AWS CodePipeline Pipelines (p. 68).

```
{
  "ParameterName" : { "Fn::GetParam" : ["ArtifactName", "config-file-name.json",
 "ParamName"]}
}
```

This property is optional.

`RoleArn`

The Amazon Resource Name (ARN) of the IAM service role that AWS CloudFormation assumes when it operates on resources in a stack.

This property is required for the following action modes: `CREATE_UPDATE`, `REPLACE_ON_FAILURE`, `DELETE_ONLY`, and `CHANGE_SET_REPLACE`.

`StackName`

The name of an existing stack or a stack that you want to create.

This property is required for all action modes.

`TemplateConfiguration`

The location of a template configuration file, which follows the format *`ArtifactName`*`::`*`TemplateConfigurationFileName`*. The template configuration file can contain template parameter values and a stack policy. Note that if you include sensitive information, such as passwords, restrict access to this file. For more information, see AWS CloudFormation Artifacts (p. 67).

This property is optional.

`TemplatePath`

The location of an AWS CloudFormation template file, which follows the format *`ArtifactName`*`::`*`TemplateFileName`*.

This property is required for the following action modes: `CREATE_UPDATE`, `REPLACE_ON_FAILURE`, and `CHANGE_SET_REPLACE`. For all other action modes, this property is ignored.

AWS CloudFormation Artifacts

AWS CodePipeline performs tasks on artifacts as AWS CodePipeline runs a pipeline. For AWS CloudFormation, artifacts can include a stack template file, a template configuration file, or both. AWS CodePipeline uses these artifacts to work with AWS CloudFormation stacks and change sets.

If you use Amazon Simple Storage Service (Amazon S3) as a source repository, you must zip the template and template configuration files into a single file before you upload them to an S3 bucket. For other repositories, such as GitHub and AWS CodeCommit, upload artifacts without zipping them. For more information, see Create a Pipeline in AWS CodePipeline in the *AWS CodePipeline User Guide*.

You can add as many files as you need to your repository. For example, you might want to include two different configurations for the same template: one for a test configuration and another for a production configuration.

This topic describes each artifact type.

Topics

Stack Template File

A stack template file defines the resources that AWS CloudFormation provisions and configures. These files are the same templates files that you use when you create or update stacks using AWS CloudFormation. You can use YAML or JSON-formatted templates. For more information about templates, see Template Anatomy (p. 128).

Template Configuration File

A template configuration file is a JSON-formatted text file that specifies template parameter values, a stack policy (p. 110), or both. Use these configuration files to specify parameter values or a stack policy for a stack. All of the parameter values that you specify must be declared in the associated template.

If you include sensitive information—such as passwords—in this file, restrict access to it. For example, if you upload your artifact to an S3 bucket, use S3 bucket policies or user policies to restrict access.

To create a configuration file, use the following format :

```
{
  "Parameters" : {
    "NameOfTemplateParameter" : "ValueOfParameter",
    ...
  },
  "StackPolicy" : {
    "Statement" : [
      StackPolicyStatement
    ]
  }
}
```

The following example specifies `TestEC2Key` for the `KeyName` parameter, and adds a stack policy that allows all update actions except for an update that deletes a resource.

```
{
  "Parameters" : {
```

```
    "KeyName" : "TestEC2Key"
  },
  "StackPolicy" : {
    "Statement" : [
      {
        "Effect" : "Allow",
        "NotAction" : "Update:Delete",
        "Principal": "*",
        "Resource" : "*"
      }
    ]
  }
}
```

Using Parameter Override Functions with AWS CodePipeline Pipelines

In an AWS CodePipeline stage, you can specify parameter overrides (p. 63) for AWS CloudFormation actions. Parameter overrides let you specify template parameter values that override values in a template configuration file. AWS CloudFormation provides functions to help you to specify dynamic values (values that are unknown until the pipeline runs).

Topics

Fn::GetArtifactAtt

The `Fn::GetArtifactAtt` function retrieves the value of an attribute from an input artifact, such as the S3 bucket name where the artifact is stored. Use this function to specify attributes of an artifact, such as its filename or S3 bucket name.

When you run a pipeline, AWS CodePipeline copies and writes files to the pipeline's artifact store (an S3 bucket). AWS CodePipeline generates the filenames in the artifact store. These filenames are unknown before you run the pipeline.

For example, in your pipeline, you might have a source stage where AWS CodePipeline copies your AWS Lambda function source code to the artifact store. In the next stage, you have an AWS CloudFormation template that creates the Lambda function, but AWS CloudFormation requires the filename to create the function. You must use the `Fn::GetArtifactAtt` function to pass the exact S3 bucket and file names.

Syntax

Use the following syntax to retrieve an attribute value of an artifact.

```
{ "Fn::GetArtifactAtt" : [ "artifactName", "attributeName" ] }
```

`artifactName`

The name of the input artifact. You must declare this artifact as input for the associated action.

`attributeName`

The name of the artifact attribute whose value you want to retrieve. For details about each artifact attribute, see the following Attributes section.

Example

The following parameter overrides specify the `BucketName` and `ObjectKey` parameters by retrieving the S3 bucket name and filename of the `LambdaFunctionSource` artifact. This example assumes that AWS CodePipeline copied Lambda function source code and saved it as an artifact, for example, as part of a source stage.

```
{
  "BucketName" : { "Fn::GetArtifactAtt" : ["LambdaFunctionSource", "BucketName"]},
  "ObjectKey" : { "Fn::GetArtifactAtt" : ["LambdaFunctionSource", "ObjectKey"]}
}
```

Attributes

You can retrieve the following attributes for an artifact.

`BucketName`

The name of the S3 bucket where the artifact is stored.

`ObjectKey`

The name of the `.zip` file that contains the artifact that is generated by AWS CodePipeline, such as `1ABCyZZ.zip`.

`URL`

The Amazon Simple Storage Service (Amazon S3) URL of the artifact, such as `https://s3-us-west-2.amazonaws.com/artifactstorebucket-yivczw8jma0c/test/TemplateSo/1ABCyZZ.zip`.

Fn::GetParam

The `Fn::GetParam` function returns a value from a key-value pair in a JSON-formatted file. The JSON file must be included in an artifact.

Use this function to retrieve output values from an AWS CloudFormation stack and use them as input for another action. For example, if you specify an output filename for an AWS CloudFormation action, AWS CodePipeline saves the output in a JSON file and then adds it to the output artifact's `.zip` file. Use the `Fn::GetParam` function to retrieve the output value, and use it as input for another action.

Syntax

Use the following syntax to retrieve a value from a key-value pair.

```
{ "Fn::GetParam" : [ "artifactName", "JSONFileName", "keyName" ] }
```

`artifactName`

The name of the artifact, which must be included as an input artifact for the associated action.

`JSONFileName`

The name of a JSON file that is contained in the artifact.

`keyName`

The name of the key whose value you want to retrieve.

Examples

The following examples demonstrate how to use the `Fn::GetParam` function in a parameter override.

Syntax

The following parameter override specifies the `WebSiteURL` parameter by retrieving the value of the `URL` key from the `stack-output.json` file that is in the `WebStackOutput` artifact.

```
{
  "WebSiteURL" : { "Fn::GetParam" : ["WebStackOutput", "stack-output.json", "URL"]}
}
```

AWS CloudFormation Template Snippets

The following AWS CloudFormation template snippets, from an AWS CodePipeline pipeline, demonstrate how to pass stack outputs. These snippets show two stages of pipeline definition. The first stage creates a stack and save its outputs in the `TestOutput.json` file in the `StackAOutput` artifact. These values are specified by the `OutputFileName` and `OutputArtifacts` properties.

In a subsequent stage, stack B uses the outputs from stack A. In the `ParameterOverrides` property, the example uses the `Fn::GetParam` function to specify the `StackBInputParam` parameter. The resulting value is the value associated with the `StackAOutputName` key.

Working with Stacks

A stack is a collection of AWS resources that you can manage as a single unit. In other words, you can create, update, or delete a collection of resources by creating, updating, or deleting stacks. All the resources in a stack are defined by the stack's AWS CloudFormation template. A stack, for instance, can include all the resources required to run a web application, such as a web server, a database, and networking rules. If you no longer require that web application, you can simply delete the stack, and all of its related resources are deleted.

AWS CloudFormation ensures all stack resources are created or deleted as appropriate. Because AWS CloudFormation treats the stack resources as a single unit, they must all be created or deleted successfully for the stack to be created or deleted. If a resource cannot be created, AWS CloudFormation rolls the stack back and automatically deletes any resources that were created. If a resource cannot be deleted, any remaining resources are retained until the stack can be successfully deleted.

You can work with stacks by using the AWS CloudFormation console, API, or AWS CLI.

> **Note**
> You are charged for the stack resources for the time they were operating (even if you deleted the stack right away).

Topics

Using the AWS CloudFormation Console

The AWS CloudFormation console allows you to create, monitor, update and delete stacks directly from your web browser. This section contains guidance on using the AWS CloudFormation console to perform common actions.

In This Section

Logging In to the AWS CloudFormation Console

The AWS CloudFormation console allows you to create, monitor, update, and delete your AWS CloudFormation stacks with a web-based interface. It is part of the AWS Management Console.

You can access the AWS CloudFormation console in a number of ways:

- Open the AWS CloudFormation console directly with the URL `https://console.aws.amazon.com/cloudformation/` . If you are not logged in to the AWS Management Console yet, you need to log in before using the AWS CloudFormation console.
- If you are logged into and using the AWS Management Console, you can access the AWS CloudFormation console by opening the **Services** menu and selecting **CloudFormation** in one of the following sub-menus:
  - **Deployment and Management**
  - **All Services**

If you don't have any AWS CloudFormation stacks running, you are presented with the option to **Create a stack**. Otherwise, you see a list of your currently-running stacks.

See Also

- Creating a Stack (p. 72)

Creating a Stack on the AWS CloudFormation Console

Before you create a stack, you must have a template that describes what resources AWS CloudFormation will include in your stack. For more information, see Working with AWS CloudFormation Templates (p. 127).

Creating Stacks Using Change Sets

To preview how a AWS CloudFormation stack will be configured before creating the stack, create a change set. This functionality allows you to examine various configurations and make corrections and changes to your stack before executing the change set.

Creating a Change Set for a New Stack

To create a change set for a new stack, submit the configuration that you want to use by providing a template, input parameter values, or both.

To create a change set (console)

1. In the AWS CloudFormation console, choose **Create Stack**, and then choose **Create Change Set for New Stack**.
2. On the **Select Template** page, specify the location of your template.

   - For a template stored locally, choose **Upload a template to Amazon S3**. **Choose File** to navigate to the file, choose the file, and then choose **Next**.
   - For a template stored in an Amazon S3 bucket, choose **Specify an Amazon S3 URL**. Type or paste the URL for the template, and then choose **Next**.

     If your template is stored in a versioning-enabled bucket, you can specify a specific version, for example: `https://s3.amazonaws.com/templates/myTemplate.template?versionId=123ab1cdeKdOW5IH4GAcYbEngcpTJTDW`

     For more information, see Managing Objects in a Versioning-Enabled Bucket in the *Amazon Simple Storage Service Console User Guide*.
3. On the **Specify Details** page, configure the following items:

   - Type the **Stack name**.
   - (Optional) To identify your change set, type its **Name** and **Description**.
   - If your template contains parameters, type the parameter values in the **Parameters** section.

   When you finish, choose **Next**.
4. (Optional) On the **Options** page, update the stack's service role, the stack tags, and the stack's Amazon SNS notification topic, and then choose **Next**.
5. On the **Review** page, review the proposed configuration.

   If the template includes AWS Identity and Access Management (IAM) resources, select **I acknowledge that this template may create IAM resources** to acknowledge that AWS CloudFormation might create IAM resources if you execute this change set. IAM resources can modify permissions in your AWS account. Review these resources to ensure that you allow the correction actions. For more information, see Controlling Access with AWS Identity and Access Management (p. 8).

   When you finish, choose **Create change set**.

   While AWS CloudFormation begins to create the change set, the status of the change set is **CREATE_IN_PROGRESS**. When AWS CloudFormation completes the creation progress, it sets its status to **CREATE_COMPLETE**. In the **Changes** section, AWS CloudFormation lists the proposed configuration of your stack.

   If AWS CloudFormation fails to create the change set and reports the **CREATE_FAILED** status, fix the error displayed in the **Status** field, and then create a new change set. At this stage, you can try various configurations and make corrections and changes to your stack before executing the next change set.
6. To create a new stack using the change set, choose **Execute**, and then choose **Execute** again.

   When you create a change set, AWS CloudFormation launches a stack and reports the **REVIEW_IN_PROGRESS** status until you execute the change set.

Creating a stack on the AWS CloudFormation console is an easy, wizard-driven process that consists of the following steps:

1. Starting the Create Stack wizard (p. 74)
2. Selecting a stack template (p. 74)
3. Specifying stack parameters (p. 75)

4. Setting Stack Options (p. 76)
5. Reviewing your stack (p. 77)

After creating a stack, you can monitor the stack's progress, view the stack's resources and outputs, update the stack, and delete it. Information about these actions are provided in their associated topics.

Starting the Create Stack Wizard

To create a stack on the AWS CloudFormation console

1. Log in to the AWS Management Console and select **CloudFormation** in the **Services** menu.
2. Create a new stack by using one of the following options:
   - Click **Create Stack**. This is the *only* option if you have a currently running stack.
   - Click **Create New Stack** in the **CloudFormation Stacks** main window. This option is visible only if you have no running stacks.
   - Click **Launch CloudFormer** in the **CloudFormation Stacks** main window to create a stack from currently running resources. This option is visible only if you have no running stacks.

     For more information about using CloudFormer to create AWS CloudFormation stacks, see Using CloudFormer to Create Templates (p. 384).

Next, you choose a stack template (p. 74).

Selecting a Stack Template

After starting the Create Stack wizard (p. 74), you specify the template that you want AWS CloudFormation to use to create your stack.

AWS CloudFormation templates are JSON- or YAML-formatted files that specify the AWS resources that make up your stack. For more information about AWS CloudFormation templates, see Working with AWS CloudFormation Templates (p. 127).

To choose a stack template:

1. On the **Select Template** page, choose a stack template by using one of the following options:

   Design a template

   To create or modify a template, use AWS CloudFormation Designer, a drag-and-drop interface. For more information, see What Is AWS CloudFormation Designer? (p. 157).

   Choose a template
   - **Select a sample template.**

     Select an AWS CloudFormation template from a list of samples. For descriptions of the templates, see Sample Templates (p. 1367).

     To create a stack from existing AWS resources by using the CloudFormer tool, select **CloudFormer** from the list. For more information, see Using CloudFormer to Create Templates (p. 384).
   - **Upload a template to Amazon S3**.

     Select an AWS CloudFormation template on your local computer. Choose **Choose File** to select the template file that you want to upload. The template can be a maximum size of 460,800 bytes.

If you use the CLI or API to create a stack, you can upload a template with a maximum size of 512,000 bytes.

> **Note**
> If you upload a local template file, AWS CloudFormation uploads it to an Amazon Simple Storage Service (Amazon S3) bucket in your AWS account. The buckets are accessible to anyone with Amazon S3 permissions in your AWS account. If you don't already have an S3 bucket that was created by AWS CloudFormation, it creates a unique bucket for each Region in which you upload a template file. If you already have an S3 bucket that was created by AWS CloudFormation in your AWS account, AWS CloudFormation adds the template to that bucket.
> You can use your own bucket and manage its permissions by manually uploading templates to Amazon S3. When you create or update a stack, specify the Amazon S3 URL of a template file.

- **Specify an Amazon S3 template URL**.

  Specify a URL to a template in an S3 bucket.

  > **Important**
  > If your template includes nested stacks (for example, stacks described in other template documents located in subdirectories), ensure that your S3 bucket contains the necessary files and directories.

  If you have a template in a versioning-enabled bucket, you can specify a specific version of the template, such as `https://s3.amazonaws.com/templates/myTemplate.template?versionId=123ab1cdeKdOW5IH4GAcYbEngcpTJTDW`. For more information, see Managing Objects in a Versioning-Enabled Bucket in the *Amazon Simple Storage Service Console User Guide*.

  The URL must point to a template with a maximum size of 460,800 bytes that is stored in an S3 bucket that you have read permissions to and that is located in the same region as the stack. The URL can be a maximum of 1024 characters long.

2. To accept your settings, choose **Next**, and proceed with specifying the stack name and parameters (p. 75).

   Before creating resources, AWS CloudFormation validates your template to catch syntactic and some semantic errors, such as circular dependencies. During validation, AWS CloudFormation first checks if the template is valid JSON. If it isn't, AWS CloudFormation checks if the template is valid YAML. If both checks fail, AWS CloudFormation returns a template validation error.

Specifying Stack Name and Parameters

After selecting a stack template, specify the stack name and values for the parameters that were defined in the template.

With parameters, you can customize your stack at creation time. Your parameter values can be used in the stack template to modify how resources are configured. That way you don't have to hard code values in multiple templates to specify different settings. For more information about parameters in an AWS CloudFormation template, see Parameters (p. 132).

To specify the stack name parameter values

1. On the **Specify Details** page, type a stack name in the **Stack name** box.

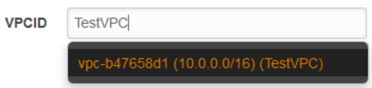
   The stack name is an identifier that helps you find a particular stack from a list of stacks. A stack name can contain only alphanumeric characters (case sensitive) and hyphens. It must start with an alphabetic character and cannot be longer than 128 characters.
2. In the **Parameters** section, specify parameters that are defined in the stack template.

You can use or change any parameters with default values.

3. When you are satisfied with the parameter values, click **Next** to proceed with setting options for your stack (p. 76).

AWS-specific Parameter Types

When you create stacks that contain AWS-specific parameter types, the AWS CloudFormation console provides drop-down lists of valid values for those parameters. Depending on the parameter type, you can search for values by ID, name, or the value of the `Name` tag. For example, with the `AWS::EC2::VPC::Id` parameter type, you can search for a specific VPC ID, such as `vpc-b47658d1`. If the VPC was tagged with a name, such as `Name:TestVPC`, you can also search for `TestVPC`. Currently, you can search only for tag values with the `Name` key.



Note

The console doesn't provide a drop-down list or enable you to search for values with the `AWS::EC2::Image::Id` parameter type; AWS CloudFormation only verifies if the input values are valid Amazon Elastic Compute Cloud image IDs.

Group and Sort Parameters

The console alphabetically lists input parameters by their logical ID. When you create a template, you can use the `AWS::CloudFormation::Interface` metadata key to override the default ordering. For more information and an example of the `AWS::CloudFormation::Interface` metadata key, see AWS::CloudFormation::Interface (p. 483).

Setting AWS CloudFormation Stack Options

After specifying parameters (p. 132) that are defined in the template, you can set additional options for your stack.

You can set the following stack options:

Tags

Tags are arbitrary key-value pairs that can be used to identify your stack for purposes such as cost allocation. For more information about what tags are and how they can be used, see Tagging Your Resources in the *Amazon EC2 User Guide*.

A **Key** consists of any alphanumeric characters or spaces. Tag keys can be up to 127 characters long. A **Value** consists of any alphanumeric characters or spaces. Tag values can be up to 255 characters long.

Permissions

An existing AWS Identity and Access Management (IAM) service role that AWS CloudFormation can assume.

Instead of using your account credentials, AWS CloudFormation uses the role's credentials to create your stack. For more information, see AWS CloudFormation Service Role (p. 13).

Notification Options

A new or existing Amazon Simple Notification Service topic where notifications about stack events are sent.

If you create an Amazon SNS topic, you must specify a name and an email address, where stack event notifications are sent.

Timeout

The number of minutes before stack creation times out. If the stack could not be created before the time expires, creation fails due to timeout and the stack is rolled back. By default, the stack creation never times out.

Rollback on failure

Specifies whether the stack should be rolled back if stack creation fails. Typically, you want to accept the default value of **Yes**. Select **No** if you want the stack's state retained even if creation fails, such as when you are debugging a stack template.

Stack policy

Defines the resources that you want to protect from unintentional updates during a stack update. By default, all resources can be updated during a stack update. For more information, see Prevent Updates to Stack Resources (p. 110).

To set stack options

1. On the **Options** screen of the **Create Stack** wizard, you can specify tags or set additional options by expanding the **Advanced** section.
2. When you have entered all of your stack options, click **Next Step** to proceed with reviewing your stack (p. 77).

Reviewing Your Stack and Estimating Stack Cost on the AWS CloudFormation Console

The final step before your stack is launched is to review the values entered while creating the stack. You can also estimate the cost of your stack.

1. On the **Review** page, review the details of your stack.

   If you need to change any of the values prior to launching the stack, click **Back** to go back to the page that has the setting that you want to change.
2. (Optional) You can click the **Cost** link to estimate the cost of your stack. The AWS Simple Monthly Calculator displays values from your stack template and launch settings.
3. After you review the stack launch settings and the estimated cost of your stack, click **Create** to launch your stack.

   Your stack appears in the list of AWS CloudFormation stacks, with a status of **CREATE_IN_PROGRESS**.

   While your stack is being created (or afterward), you can use the stack detail pane to view your stack's events, data, or resources (p. 78). AWS CloudFormation automatically refreshes stack events every minute. By viewing stack creation events, you can understand the sequence of events that lead to your stack's creation (or failure, if you are debugging your stack).

   After your stack has been successfully created, its status changes to **CREATE_COMPLETE**. You can then select it (if necessary) and click the **Outputs** tab to view your stack's outputs if you have defined any in the template.

Creating an EC2 Key Pair

The use of some AWS CloudFormation resources and templates will require you to specify an Amazon EC2 key pair for authentication, such as when you are configuring SSH access to your instances.

Amazon EC2 key pairs can be created with the AWS Management Console. For more information, see Amazon EC2 Key Pairs in the *Amazon EC2 User Guide for Linux Instances*.

Estimating the Cost of Your AWS CloudFormation Stack

There is no additional charge for AWS CloudFormation. You pay for AWS resources (e.g. Amazon EC2 instances, Elastic Load Balancing load balancers and so on) created using AWS CloudFormation as if you created them by hand.

To estimate the cost of your stack

1. On the **Review** page of the **Create Stack** dialog, click the **Cost** link.

   This link opens the **AWS Simple Monthly Calculator** in a new browser page (or tab, depending on how your browser is set up).

   > **Note**
   > Because you launched the calculator from the AWS CloudFormation console, it is pre-populated with your template configuration and parameter values. There are many additional configurable values that can provide you with a better estimate if you have an idea of how much data transfer you expect to your Amazon EC2 instance.
2. Click the **Estimate of your Monthly Bill** tab for a monthly estimate of running your stack, along with a categorized display of what factors contributed to the estimate.

Viewing AWS CloudFormation Stack Data and Resources on the AWS Management Console

After you've created an AWS CloudFormation stack, you can use the AWS Management Console to view its data and resources. You can view the following stack information:

Outputs

Displays outputs that were declared in the stack's template.

Resources

Displays the resources that are part of the stack.

Events

Displays the operations that are tracked when you create, update, or delete the stack.

Template

Displays the stack's template.

For stacks that contain transform, choose **View original template** to view the user-submitted template or **View processed template** to view the template after AWS CloudFormation has processed the transforms. AWS CloudFormation uses the processed template to create or update your stack.

Parameters

Displays the stack's parameters and their values.

Tags

Displays any tags that were associated with the stack.

Stack Policy

Describes the stack resources that are protected against stack updates. To update these resources, they must be explicitly allowed during a stack update.

To view information about your AWS CloudFormation stack

1. Select your stack in the AWS CloudFormation console. This displays information in the stack detail pane.
2. In the detail pane, click a tab to view the related information about your stack.

   For example, click **Outputs** to view the outputs that are associated with your stack.

Deleting a Stack on the AWS CloudFormation Console

To delete a stack

1. From the list of stacks in the AWS CloudFormation console, select the stack that you want to delete (it must be currently running).
2. Choose **Actions** and then **Delete Stack**.
3. Click **Yes, Delete** when prompted.

   Note
   After stack deletion has begun, you cannot abort it. The stack proceeds to the **DELETE_IN_PROGRESS** state.

   After the stack deletion is complete, the stack will be in the **DELETE_COMPLETE** state. Stacks in the **DELETE_COMPLETE** state are not displayed in the AWS CloudFormation console by default. To display deleted stacks, you must change the stack view setting as described in Viewing Deleted Stacks (p. 79).

   If the delete failed, the stack will be in the **DELETE_FAILED** state. For solutions, see the Delete Stack Fails (p. 1369) troubleshooting topic.

Viewing Deleted Stacks on the AWS CloudFormation Console

By default, the AWS CloudFormation console does not display stacks in the **DELETE_COMPLETE** state. To display information about deleted stacks, you must change the stack view.

To view deleted stacks

- In the AWS CloudFormation console, select **Deleted** from the **Filter** list.

  AWS CloudFormation lists all of your deleted stacks (stacks with **DELETE_COMPLETE** status).

See Also

- Deleting a Stack (p. 79)
- Viewing Stack Data and Resources (p. 78)

Related Topics

• Using the AWS CLI (p. 80)

Using the AWS Command Line Interface

With the AWS Command Line Interface (CLI), you can create, monitor, update and delete stacks from your system's terminal. You can also use the AWS CLI to automate actions through scripts. For more information about the AWS CLI, see the AWS Command Line Interface User Guide.

If you use Windows PowerShell, AWS also offers the AWS Tools for Windows PowerShell.

> **Note**
> The prior AWS CloudFormation CLI tools are still available, but not recommended. If you need information about the prior AWS CloudFormation CLI tools, see the AWS CloudFormation CLI Reference in the documentation archive.

Topics

Creating a Stack

To create a stack you run the `aws cloudformation create-stack` command. You must provide the stack name, the location of a valid template, and any input parameters.

Parameters are separated with a space and the key names are case sensitive. If you mistype a parameter key name when you run `aws cloudformation create-stack`, AWS CloudFormation doesn't create the stack and reports that the template doesn't contain that parameter.

> **Note**
> If you specify a local template file, AWS CloudFormation uploads it to an Amazon S3 bucket in your AWS account. AWS CloudFormation creates a unique bucket for each region in which you upload a template file. The buckets are accessible to anyone with Amazon S3 permissions in your AWS account. If an AWS CloudFormation-created bucket already exists, the template is added to that bucket.
> You can use your own bucket and manage its permissions by manually uploading templates to Amazon S3. Then whenever you create or update a stack, specify the Amazon S3 URL of a template file.

By default, `aws cloudformation describe-stacks` returns parameter values. To prevent sensitive parameter values such as passwords from being returned, include a `NoEcho` property set to `TRUE` in your AWS CloudFormation template.

The following example creates the `myteststack` stack:

```
PROMPT> aws cloudformation create-stack --stack-name myteststack --template-body file:///
home/testuser/mytemplate.json --parameters ParameterKey=Parm1,ParameterValue=test1
 ParameterKey=Parm2,ParameterValue=test2
{
  "StackId" : "arn:aws:cloudformation:us-west-2:123456789012:stack/
myteststack/330b0120-1771-11e4-af37-50ba1b98bea6"
}
```

Describing and Listing Your Stacks

You can use two AWS CLI commands to get information about your AWS CloudFormation stacks: `aws cloudformation list-stacks` and `aws cloudformation describe-stacks`.

> **Note**
> See the section called “AWS CloudFormation Resources” (p. 9) for a discussion of how IAM policies may limit what a user can do with these two AWS CLI commands.

aws cloudformation list-stacks

The `aws cloudformation list-stacks` command enables you to get a list of any of the stacks you have created (even those which have been deleted up to 90 days). You can use an option to filter results by stack status, such as `CREATE_COMPLETE` and `DELETE_COMPLETE`. The `aws cloudformation list-stacks` command returns summary information about any of your running or deleted stacks, including the name, stack identifier, template, and status.

> **Note**
> The aws cloudformation list-stacks command returns information on deleted stacks for 90 days after they have been deleted.

The following example shows a summary of all stacks that have a status of CREATE_COMPLETE:

```
PROMPT> aws cloudformation list-stacks --stack-status-filter CREATE_COMPLETE
[
    {
        "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/myteststack/
644df8e0-0dff-11e3-8e2f-5088487c4896",
        "TemplateDescription": "AWS CloudFormation Sample Template S3_Bucket: Sample
 template showing how to create a publicly accessible S3 bucket. **WARNING** This template
 creates an
S3 bucket. You will be billed for the AWS resources used if you create a stack from this
 template.",
        "StackStatusReason": null,
        "CreationTime": "2013-08-26T03:27:10.190Z",
        "StackName": "myteststack",
        "StackStatus": "CREATE_COMPLETE"
    }
]
```

aws cloudformation describe-stacks

The `aws cloudformation describe-stacks` command provides information on your running stacks. You can use an option to filter results on a stack name. This command returns information about the stack, including the name, stack identifier, and status.

The following example shows summary information for the `myteststack` stack:

```
PROMPT> aws cloudformation describe-stacks --stack-name myteststack
{
    "Stacks":  [
        {
```

```
            "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/myteststack/
a69442d0-0b8f-11e3-8b8a-500150b352e0",
            "Description": "AWS CloudFormation Sample Template S3_Bucket: Sample template
 showing how to create a publicly accessible S3 bucket. **WARNING** This template creates
 an S3 bucket.
You will be billed for the AWS resources used if you create a stack from this template.",
            "Tags": [],
            "Outputs": [
                {
                    "Description": "Name of S3 bucket to hold website content",
                    "OutputKey": "BucketName",
                    "OutputValue": "myteststack-s3bucket-jssofi1zie2w"
                }
            ],
            "StackStatusReason": null,
            "CreationTime": "2013-08-23T01:02:15.422Z",
            "Capabilities": [],
            "StackName": "myteststack",
            "StackStatus": "CREATE_COMPLETE",
            "DisableRollback": false
        }
    ]
}
```

If you don't use the `--stack-name` option to limit the output to one stack, information on all your running stacks is returned.

Stack Status Codes

You can specify one or more stack status codes to list only stacks with the specified status codes. The following table describes each stack status code:

Stack Status	Description
`CREATE_COMPLETE`	Successful creation of one or more stacks.
`CREATE_IN_PROGRESS`	Ongoing creation of one or more stacks.
`CREATE_FAILED`	Unsuccessful creation of one or more stacks. View the stack events to see any associated error messages. Possible reasons for a failed creation include insufficient permissions to work with all resources in the stack, parameter values rejected by an AWS service, or a timeout during resource creation.
`DELETE_COMPLETE`	Successful deletion of one or more stacks. Deleted stacks are retained and viewable for 90 days.
`DELETE_FAILED`	Unsuccessful deletion of one or more stacks. Because the delete failed, you might have some resources that are still running; however, you cannot work with or update the stack. Delete the stack again or view the stack events to see any associated error messages.
`DELETE_IN_PROGRESS`	Ongoing removal of one or more stacks.
`REVIEW_IN_PROGRESS`	Ongoing creation of one or more stacks with an expected `StackId` but without any templates or resources. **Important** A stack with this status code counts against the maximum possible number of stacks (p. 16).

Stack Status	Description
ROLLBACK_COMPLETE	Successful removal of one or more stacks after a failed stack creation or after an explicitly canceled stack creation. Any resources that were created during the create stack action are deleted.
ROLLBACK_FAILED	Unsuccessful removal of one or more stacks after a failed stack creation or after an explicitly canceled stack creation. Delete the stack or view the stack events to see any associated error messages.
ROLLBACK_IN_PROGRESS	Ongoing removal of one or more stacks after a failed stack creation or after an explicitly cancelled stack creation.
UPDATE_COMPLETE	Successful update of one or more stacks.
UPDATE_COMPLETE_CLEANUP_IN_PROGRESS	Ongoing removal of old resources for one or more stacks after a successful stack update. For stack updates that require resources to be replaced, AWS CloudFormation creates the new resources first and then deletes the old resources to help reduce any interruptions with your stack. In this state, the stack has been updated and is usable, but AWS CloudFormation is still deleting the old resources.
UPDATE_IN_PROGRESS	Ongoing update of one or more stacks.
UPDATE_ROLLBACK_COMPLETE	Successful return of one or more stacks to a previous working state after a failed stack update.
UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS	Ongoing removal of new resources for one or more stacks after a failed stack update. In this state, the stack has been rolled back to its previous working state and is usable, but AWS CloudFormation is still deleting any new resources it created during the stack update.
UPDATE_ROLLBACK_FAILED	Unsuccessful return of one or more stacks to a previous working state after a failed stack update. You can delete the stack or contact customer support to restore the stack to a usable state.
UPDATE_ROLLBACK_IN_PROGRESS	Ongoing return of one or more stacks to the previous working state after failed stack update.

Viewing Stack Event History

You can track the status of the resources AWS CloudFormation is creating and deleting with the `aws cloudformation describe-stack-events` command. The amount of time to create or delete a stack depends on the complexity of your stack.

In the following example, a sample stack is created from a template file by using the `aws cloudformation create-stack` command. After the stack is created, the events that were reported during stack creation are shown by using the aws cloudformation describe-stack-events command.

The following example creates a stack with the name `myteststack` using the `sampletemplate.json` template file:

```
PROMPT> aws cloudformation create-stack --stack-name myteststack --template-body file:///
home/local/test/sampletemplate.json
```

```
[
    {
        "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896",
        "Description": "AWS CloudFormation Sample Template S3_Bucket: Sample template
 showing how to create a publicly accessible S3 bucket. **WARNING** This template creates
 an S3 bucket.
You will be billed for the AWS resources used if you create a stack from this template.",
        "Tags": [],
        "Outputs": [
            {
                "Description": "Name of S3 bucket to hold website content",
                "OutputKey": "BucketName",
                "OutputValue": "myteststack-s3bucket-jssofi1zie2w"
            }
        ],
        "StackStatusReason": null,
        "CreationTime": "2013-08-23T01:02:15.422Z",
        "Capabilities": [],
        "StackName": "myteststack",
        "StackStatus": "CREATE_COMPLETE",
        "DisableRollback": false
    }
]
```

The following example describes the `myteststack` stack:

```
PROMPT> aws cloudformation describe-stack-events --stack-name myteststack
{
    "StackEvents": [
        {
            "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896",
            "EventId": "af67ef60-0b8f-11e3-8b8a-500150b352e0",
            "ResourceStatus": "CREATE_COMPLETE",
            "ResourceType": "AWS::CloudFormation::Stack",
            "Timestamp": "2013-08-23T01:02:30.070Z",
            "StackName": "myteststack",
            "PhysicalResourceId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
myteststack/a69442d0-0b8f-11e3-8b8a-500150b352e0",
            "LogicalResourceId": "myteststack"
        },
        {
            "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896",
            "EventId": "S3Bucket-CREATE_COMPLETE-1377219748025",
            "ResourceStatus": "CREATE_COMPLETE",
            "ResourceType": "AWS::S3::Bucket",
            "Timestamp": "2013-08-23T01:02:28.025Z",
            "StackName": "myteststack",
            "ResourceProperties": "{\"AccessControl\":\"PublicRead\"}",
            "PhysicalResourceId": "myteststack-s3bucket-jssofi1zie2w",
            "LogicalResourceId": "S3Bucket"
        },
        {
            "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896",
            "EventId": "S3Bucket-CREATE_IN_PROGRESS-1377219746688",
            "ResourceStatus": "CREATE_IN_PROGRESS",
            "ResourceType": "AWS::S3::Bucket",
            "Timestamp": "2013-08-23T01:02:26.688Z",
            "ResourceStatusReason": "Resource creation Initiated",
            "StackName": "myteststack",
            "ResourceProperties": "{\"AccessControl\":\"PublicRead\"}",
            "PhysicalResourceId": "myteststack-s3bucket-jssofi1zie2w",
```

```
            "LogicalResourceId": "S3Bucket"
        },
        {
            "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896",
            "EventId": "S3Bucket-CREATE_IN_PROGRESS-1377219743862",
            "ResourceStatus": "CREATE_IN_PROGRESS",
            "ResourceType": "AWS::S3::Bucket",
            "Timestamp": "2013-08-23T01:02:23.862Z",
            "StackName": "myteststack",
            "ResourceProperties": "{\"AccessControl\":\"PublicRead\"}",
            "PhysicalResourceId": null,
            "LogicalResourceId": "S3Bucket"
        },
        {
            "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896",
            "EventId": "a69469e0-0b8f-11e3-8b8a-500150b352e0",
            "ResourceStatus": "CREATE_IN_PROGRESS",
            "ResourceType": "AWS::CloudFormation::Stack",
            "Timestamp": "2013-08-23T01:02:15.422Z",
            "ResourceStatusReason": "User Initiated",
            "StackName": "myteststack",
            "PhysicalResourceId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
myteststack/a69442d0-0b8f-11e3-8b8a-500150b352e0",
            "LogicalResourceId": "myteststack"
        }
    ]
}
```

Note

You can run the aws cloudformation describe-stack-events command while the stack is being created to view events as they are reported.

The most recent events are reported first. The following table describe the fields returned by the aws cloudformation describe-stack-events command:

Field	Description
`EventId`	Event identifier
`StackName`	Name of the stack that the event corresponds to
`StackId`	Identifier of the stack that the event corresponds to
`LogicalResourceId`	Logical identifier of the resource
`PhysicalResourceId`	Physical identifier of the resource
`ResourceProperties`	Properties of the resource
`ResourceType`	Type of the resource
`Timestamp`	Time when the event occurred
`ResourceStatus`	The status of the resource, which can be one of the following status codes: `CREATE_COMPLETE` | `CREATE_FAILED` | `CREATE_IN_PROGRESS` | `DELETE_COMPLETE` | `DELETE_FAILED` | `DELETE_IN_PROGRESS` | `DELETE_SKIPPED` | `UPDATE_COMPLETE` | `UPDATE_FAILED` | `UPDATE_IN_PROGRESS`. The `DELETE_SKIPPED` status applies to resources with a deletion policy attribute of retain.

Field	Description
`ResourceStatusReason`	More information on the status

Listing Resources

Immediately after you run the `aws cloudformation create-stack` command, you can list its resources using the `aws cloudformation list-stack-resources` command. This command lists a summary of each resource in the stack that you specify with the `--stack-name` parameter. The report includes a summary of the stack, including the creation or deletion status.

The following example shows the resources for the `myteststack` stack:

```
PROMPT> aws cloudformation list-stack-resources --stack-name myteststack
{
    "StackResourceSummaries": [
        {
            "ResourceStatus": "CREATE_COMPLETE",
            "ResourceType": "AWS::S3::Bucket",
            "ResourceStatusReason": null,
            "LastUpdatedTimestamp": "2013-08-23T01:02:28.025Z",
            "PhysicalResourceId": "myteststack-s3bucket-sample",
            "LogicalResourceId": "S3Bucket"
        }
    ]
}
```

AWS CloudFormation reports resource details on any running or deleted stack. If you specify the name of a stack whose status is `CREATE_IN_PROCESS`, AWS CloudFormation reports only those resources whose status is `CREATE_COMPLETE`.

> **Note**
> The aws cloudformation describe-stack-resources command returns information on deleted stacks for 90 days after they have been deleted.

Retrieving a Template

AWS CloudFormation stores the template you use to create your stack as part of the stack. You can retrieve the template from AWS CloudFormation using the `aws cloudformation get-template` command.

> **Note**
> The `aws cloudformation get-template` command returns the deleted stacks templates for up to 90 days after the stack has been deleted.

The following example shows the template for the `myteststack` stack:

```
PROMPT> aws cloudformation get-template --stack-name myteststack
{
    "TemplateBody": {
        "AWSTemplateFormatVersion": "2010-09-09",
        "Outputs": {
            "BucketName": {
                "Description": "Name of S3 bucket to hold website content",
                "Value": {
                    "Ref": "S3Bucket"
                }
            }
        },
```

```
        "Description": "AWS CloudFormation Sample Template S3_Bucket: Sample template
 showing how to create a publicly accessible S3 bucket. **WARNING** This template creates
 an S3 bucket.
You will be billed for the AWS resources used if you create a stack from this template.",
        "Resources": {
            "S3Bucket": {
                "Type": "AWS::S3::Bucket",
                "Properties": {
                    "AccessControl": "PublicRead"
                }
            }
        }
    }
}
```

The output contains the entire template body, enclosed in quotation marks.

Validating a Template

To check your template file for syntax errors, you can use the `aws cloudformation validate-template` command.

> **Note**
> The `aws cloudformation validate-template` command is designed to check only the syntax of your template. It does not ensure that the property values that you have specified for a resource are valid for that resource. Nor does it determine the number of resources that will exist when the stack is created.

To check the operational validity, you need to attempt to create the stack. There is no sandbox or test area for AWS CloudFormation stacks, so you are charged for the resources you create during testing.

During validation, AWS CloudFormation first checks if the template is valid JSON. If it isn't, AWS CloudFormation checks if the template is valid YAML. If both checks fail, AWS CloudFormation returns a template validation error. You can validate templates locally by using the `--template-body` parameter, or remotely with the `--template-url` parameter. The following example validates a template in a remote location:

```
PROMPT> aws cloudformation validate-template --template-url https://s3.amazonaws.com/
cloudformation-templates-us-east-1/S3_Bucket.template
{
    "Description": "AWS CloudFormation Sample Template S3_Bucket: Sample template showing
 how to create a publicly accessible S3 bucket. **WARNING** This template creates an S3
 bucket.
You will be billed for the AWS resources used if you create a stack from this template.",
    "Parameters": [],
    "Capabilities": []
}
```

The expected result is no error message, with information about all parameters listed.

The following example shows an error with a local template file:

```
PROMPT> aws cloudformation validate-template --template-body file:///home/local/test/
sampletemplate.json
{
    "ResponseMetadata": {
        "RequestId": "4ae33ec0-1988-11e3-818b-e15a6df955cd"
    },
    "Errors": [
        {
            "Message": "Template format error: JSON not well-formed. (line 11, column 8)",
```

```
            "Code": "ValidationError",
            "Type": "Sender"
        }
    ],
    "Capabilities": [],
    "Parameters": []
}
A client error (ValidationError) occurred: Template format error: JSON not well-formed.
 (line 11, column 8)
```

Uploading Local Artifacts to an S3 Bucket

For some resource properties that require an Amazon S3 location (a bucket name and filename), you can specify local references instead. For example, you might specify the S3 location of your AWS Lambda function's source code or an Amazon API Gateway REST API's OpenAPI (formerly Swagger) file. Instead of manually uploading the files to an S3 bucket and then adding the location to your template, you can specify local references, called local artifacts, in your template and then use the `package` command to quickly upload them. A local artifact is a path to a file or folder that the `package` command uploads to Amazon S3. For example, an artifact can be a local path to your AWS Lambda function's source code or an Amazon API Gateway REST API's OpenAPI file.

If you specify a file, the command directly uploads it to the S3 bucket. After uploading the artifacts, the command returns a copy of your template, replacing references to local artifacts with the S3 location where the command uploaded the artifacts. Then, you can use the returned template to create or update a stack.

If you specify a folder, the command creates a .zip file for the folder, and then uploads the .zip file. If you don’t specify a path, the command creates a .zip file for the working directory, and uploads it. You can specify an absolute or relative path, where the relative path is relative to your template’s location.

You can use local artifacts only for resource properties that the `package` command supports. For more information about this command and a list of the supported resource properties, see the `aws cloudformation package` command in the AWS Command Line Interface Reference.

The following template specifies the local artifact for a Lambda function's source code. The source code is stored in the user's `/home/user/code/lambdafunction` folder.

The following command creates a .zip file containing the function's source code folder, and then uploads the .zip file to the root folder of the `my-bucket` bucket.

The command saves the template that it generates to the path specified by the `--output` option. The command replaces the artifact with the S3 location, as shown in the following example:

Quickly Deploying Templates with Transforms

AWS CloudFormation requires you to use a change set to create a template that includes transforms. Instead of independently creating and then executing a change set, use the `aws cloudformation deploy` command. When you run this command, it creates a change set, executes the change set, and then terminates. This command reduces the numbers of required steps when you create or update a stack that includes transforms.

The following command creates a new stack by using the `my-template.json` template.

```
aws cloudformation deploy --template /path_to_template/my-template.json --stack-name my-
new-stack --parameter-overrides Key1=Value1 Key2=Value2
```

For more information, see the `aws cloudformation deploy` command in the AWS Command Line Interface Reference

Deleting a Stack

To delete a stack, you run the `aws cloudformation delete-stack` command. You must specify the name of the stack that you want to delete. When you delete a stack, you delete the stack and all of its resources.

The following example deletes the `myteststack` stack:

```
PROMPT> aws cloudformation delete-stack --stack-name myteststack
```

AWS CloudFormation Stacks Updates

When you need to make changes to a stack's settings or change its resources, you update the stack instead of deleting it and creating a new stack. For example, if you have a stack with an EC2 instance, you can update the stack to change the instance's AMI ID.

When you update a stack, you submit changes, such as new input parameter values or an updated template. AWS CloudFormation compares the changes you submit with the current state of your stack and updates only the changed resources. For a summary of the update workflow, see How Does AWS CloudFormation Work? (p. 3).

> **Note**
> When updating a stack, AWS CloudFormation might interrupt resources or replace updated resources, depending on which properties you update. For more information about resource update behaviors, see Update Behaviors of Stack Resources (p. 89).

Update Methods

AWS CloudFormation provides two methods for updating stacks: *direct update* or creating and executing *change sets*. When you directly update a stack, you submit changes and AWS CloudFormation immediately deploys them. Use direct updates when you want to quickly deploy your updates.

With change sets, you can preview the changes AWS CloudFormation will make to your stack, and then decide whether to apply those changes. Change sets are JSON-formatted documents that summarize the changes AWS CloudFormation will make to a stack. Use change sets when you want to ensure that AWS CloudFormation doesn't make unintentional changes or when you want to consider several options. For example, you can use a change set to verify that AWS CloudFormation won't replace your stack's database instances during an update.

Topics

Update Behaviors of Stack Resources

When you submit an update, AWS CloudFormation updates resources based on differences between what you submit and the stack's current template. Resources that have not changed run without disruption during the update process. For updated resources, AWS CloudFormation uses one of the following update behaviors:

Update with No Interruption

AWS CloudFormation updates the resource without disrupting operation of that resource and without changing the resource's physical ID. For example, if you update any property on an AWS::CloudTrail::Trail (p. 493) resource, AWS CloudFormation updates the trail without disruption.

Updates with Some Interruption

AWS CloudFormation updates the resource with some interruption and retains the physical ID. For example, if you update certain properties on an AWS::EC2::Instance (p. 574) resource, the instance might have some interruption while AWS CloudFormation and Amazon EC2 reconfigure the instance.

Replacement

AWS CloudFormation recreates the resource during an update, which also generates a new physical ID. AWS CloudFormation creates the replacement resource first, changes references from other dependent resources to point to the replacement resource, and then deletes the old resource. For example, if you update the `Engine` property of an AWS::RDS::DBInstance (p. 881) resource type, AWS CloudFormation creates a new resource and replaces the current DB instance resource with the new one.

The method AWS CloudFormation uses depends on which property you update for a given resource type. The update behavior for each property is described in the AWS Resource Types Reference (p. 390).

Depending on the update behavior, you can decide when to modify resources to reduce the impact of these changes on your application. In particular, you can plan when resources must be *replaced* during an update. For example, if you update the `Port` property of an AWS::RDS::DBInstance (p. 881) resource type, AWS CloudFormation replaces the DB instance by creating a new DB instance with the updated port setting and deletes the old DB instance. Before the update, you might plan to do the following to prepare for the database replacement:

- Take a snapshot of the current databases.
- Prepare a strategy for how applications that use that DB instance will handle an interruption while the DB instance is being replaced.
- Ensure that the applications that use that DB instance take into account the updated port setting and any other updates you have made.
- Use the DB snapshot to restore the databases on the new DB instance.

This example is not exhaustive; it's meant to give you an idea of the things to plan for when a resource is replaced during an update.

Note
If the template includes one or more nested stacks (p. 485), AWS CloudFormation also initiates an update for every nested stack. This is necessary to determine whether the nested stacks have been modified. AWS CloudFormation updates only those resources in the nested stacks that have changes specified in corresponding templates.

Modifying a Stack Template

If you want to modify resources and properties that are declared in a stack template, you must modify the stack's template. To ensure that you update only the resources that you intend to update, use the template for the existing stack as a starting point and make your updates to that template. If you are managing your template in a source control system, use a copy of that template as a starting point. Otherwise, you can get a copy of a stack template from AWS CloudFormation.

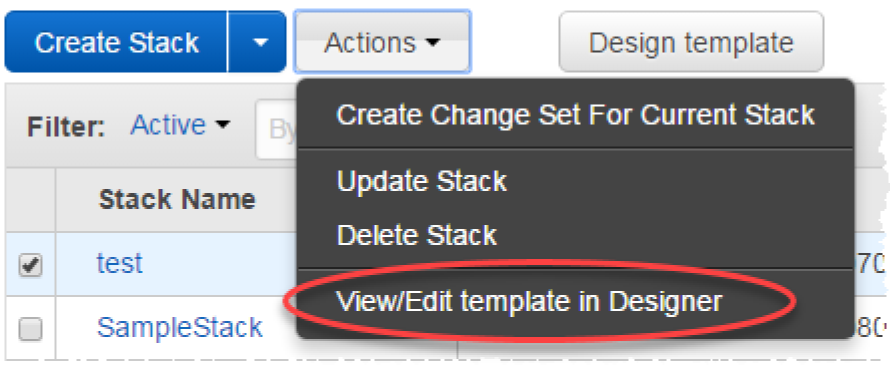
If you want to modify just the parameters or settings of a stack (like a stack's Amazon SNS topic), you can reuse the existing stack template. You don't need to get a copy of the stack template or make modifications to the stack template.

Note
If your template includes an unsupported change, AWS CloudFormation returns a message saying that the change is not permitted. This message might occur asynchronously, however, because resources are created and updated by AWS CloudFormation in a non-deterministic order by default.

Topics

Update a Stack's Template (Console)

1. In the AWS CloudFormation console, select the stack that you want to update and then choose the **Actions** and then **View in Designer**.



   AWS CloudFormation opens a copy of the stack's template in AWS CloudFormation Designer.
2. Modify the template.

   You can use the AWS CloudFormation Designer drag-and-drop interface or the integrated JSON and YAML editor to modify the template. For more information about using AWS CloudFormation Designer, see What Is AWS CloudFormation Designer? (p. 157).

   Modify only the resources that you want to update. Use the *same values* as the current stack configuration for resources and properties that you aren't updating. You can modify the template by completing any of the following actions:

   - Add new resources, or remove existing resources.

     For most resources, changing the logical name of a resource is equivalent to deleting that resource and replacing it with a new one. Any other resources that depend on the renamed resource also need to be updated and might cause them to be replaced. Other resources require you to update a property (not just the logical name) in order to trigger an update.
   - Add, modify, or delete properties of existing resources.

     Consult the AWS Resource Types Reference (p. 390) for information about the effects of updating particular resource properties. For each property, the effects of an update will be one of the following:
     - *Update requires*: No interruption (p. 90)
     - *Update requires*: Some interruptions (p. 90)

  - *Update requires*: Replacement (p. 90)
- Add, modify, or delete attributes for resources (`Metadata`, `DependsOn`, `CreationPolicy`, `UpdatePolicy`, and `DeletionPolicy`).

  Important
  You cannot update the `CreationPolicy`, `DeletionPolicy`. or `UpdatePolicy` attribute by itself. You can update them only when you include changes that add, modify, or delete resources. For example, you can add or modify a metadata attribute of a resource.
- Add, modify, or delete parameter declarations. However, you cannot add, modify, or delete a parameter that is used by a resource that does not support updates.
- Add, modify, or delete mapping declarations.

  Important
  If the values in a mapping are not being used by your stack, you can't update the mapping by itself. You need to include changes that add, modify, or delete resources. For example, you can add or modify a metadata attribute of a resource. If you update a mapping value that your stack is using, you don't need to make any other changes to trigger an update.
- Add, modify, or delete condition declarations.

  Important
  You cannot update conditions by themselves. You can update conditions only when you include changes that add, modify, or delete resources. For example, you can add or modify a metadata attribute of a resource.
- Add, modify, or delete output value declarations.

Some resources or properties may have constraints on property values or changes to those values. For example, changes to the `AllocatedStorage` property of an AWS::RDS::DBInstance (p. 881) resource must be greater than the current setting. If the value specified for the update does not meet those constraints, the update for that resource fails. For the specific constraints on `AllocatedStorage` changes, see ModifyDBInstance.

Updates to a resource can affect the properties of other resources. If you used the Ref function (p. 1343) or the Fn::GetAtt function (p. 1324) to specify an attribute from an updated resource as part of a property value in another resource in the template, AWS CloudFormation also updates the resource that contains the reference to the property that has changed. For example, if you updated the `MasterUsername` property of an `AWS::RDS::DBInstance` resource and you had an `AWS::AutoScaling::LaunchConfiguration` resource that had a `UserData` property that contained a reference to the DB instance name using the `Ref` function, AWS CloudFormation would recreate the DB instance with a new name and also update the `LaunchConfiguration` resource.

3. To check for syntax errors in your template, from the AWS CloudFormation Designer toolbar, choose **Validate template** ().

   View and fix any errors in the **Messages** pane, and then validate the template again. If you don't see any errors, your template is syntactically valid.
4. From the AWS CloudFormation Designer toolbar, choose the **File** menu () and then **Save** to save the template in an S3 bucket or locally.

Get and Update a Template for a Stack (CLI)

1. To get the template for the stack you want to update, use the command `aws cloudformation get-template`.
2. Copy the template, paste it into a text file, modify it, and save it. Copy *only* the template. The command encloses the template in quotation marks, but do not copy the quotation marks surrounding the

template. The template itself starts with an open brace and ends with the final close brace. Specify changes to the stack's resources in this file.

Updating Stacks Using Change Sets

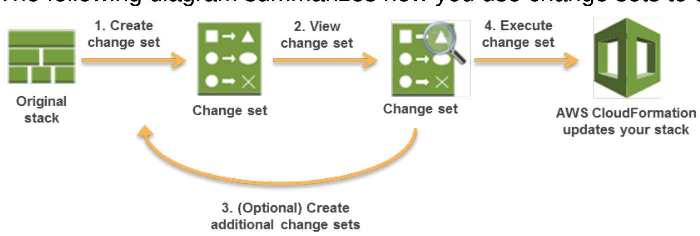
When you need to update a stack, understanding how your changes will affect running resources before you implement them can help you update stacks with confidence. Change sets allow you to preview how proposed changes to a stack might impact your running resources, for example, whether your changes will delete or replace any critical resources, AWS CloudFormation makes the changes to your stack only when you decide to execute the change set, allowing you to decide whether to proceed with your proposed changes or explore other changes by creating another change set. You can create and manage change sets using the AWS CloudFormation console, AWS CLI, or AWS CloudFormation API.

Topics

Important
Change sets don't indicate whether AWS CloudFormation will successfully update a stack. For example, a change set doesn't check if you will surpass an account limit (p. 16), if you're updating a resource (p. 390) that doesn't support updates, or if you have insufficient permissions (p. 8) to modify a resource, all of which can cause a stack update to fail. If an update fails, AWS CloudFormation attempts to roll back your resources to their original state.

Change Set Overview

The following diagram summarizes how you use change sets to update a stack:



1. Create a change set by submitting changes for the stack that you want to update. You can submit a modified stack template or modified input parameter values. AWS CloudFormation compares your stack with the changes that you submitted to generate the change set; it doesn't make changes to your stack at this point.
2. View the change set to see which stack settings and resources will change. For example, you can see which resources AWS CloudFormation will add, modify, or delete.
3. Optional: If you want to consider other changes before you decide which changes to make, create additional change sets. Creating multiple change sets helps you understand and evaluate how different changes will affect your resources. You can create as many change sets as you need.
4. Execute the change set that contains the changes that you want to apply to your stack. AWS CloudFormation updates your stack with those changes.

Note
After you execute a change, AWS CloudFormation removes all change sets that are associated with the stack because they aren't applicable to the updated stack.

You can also delete change sets to prevent executing a change set that shouldn't be applied.

Creating a Change Set

To create a change set for a running stack, submit the changes that you want to make by providing a modified template, new input parameter values, or both. AWS CloudFormation generates a change set by comparing your stack with the changes you submitted.

To modify a template, for example to add a new resource to your stack, modify a copy of the current template before creating the change set. For more information, see Modifying a Stack Template (p. 90).

To create a change set (console)

1. In the AWS CloudFormation console, from the list of stacks, select the running stack for which you want to create a change set.
2. Choose **Actions**, and then choose **Create Change Set**.
3. If you modified the stack template, specify the location of the updated template. If not, select **Use current template**.

   - For a template stored locally on your computer, select **Upload a template to Amazon S3**. Choose **Choose File** to navigate to the file and select it, and then click **Next**.
   - For a template stored in an Amazon S3 bucket, select **Specify an Amazon S3 URL**. Enter or paste the URL for the template, and then click **Next**.

     If you have a template in a versioning-enabled bucket, you can specify a specific version of the template, such as `https://s3.amazonaws.com/templates/myTemplate.template?versionId=123ab1cdeKdOW5IH4GAcYbEngcpTJTDW`. For more information, see Managing Objects in a Versioning-Enabled Bucket in the *Amazon Simple Storage Service Console User Guide*.
4. On the **Specify Details** page, type information about the change set and, if necessary, modify the parameter values that you want to change, and then choose **Next**.

   In the **Specify Details** section, specify a name for the change set. You can also specify a description of the change set to identify its purpose.

   If your template contains parameters, in the **Parameters** section, change applicable parameter values. If you're reusing the stack's template, AWS CloudFormation populates each parameter with the current value in the stack,with the exception of parameters declared with the `NoEcho` attribute. To use existing values for those parameters, select **Use existing value**.
5. On the **Options** page, you can update the stack's service role, the stack tags, or the stack's Amazon SNS notification topic, as applicable, and then choose **Next**.
6. Review the changes for this change set.

   If the template includes AWS Identity and Access Management (IAM) resources, select **I acknowledge that this template may create IAM resources** to acknowledge that AWS CloudFormation might create IAM resources if you execute this change set. IAM resources can modify permissions in your AWS account; review these resources to ensure that you're permitting only the actions that you intend. For more information, see Controlling Access with AWS Identity and Access Management (p. 8).
7. Choose **Create change set**.

   You're redirected to the change set's detail page. While AWS CloudFormation generates the change set, the status of the change set is **CREATE_IN_PROGRESS**. After it has created the change set, AWS CloudFormation sets the status to **CREATE_COMPLETE**. In the **Changes** section, AWS CloudFormation lists all of the changes that it will make to your stack. For more information, see Viewing a Change Set (p. 95).

If AWS CloudFormation fails to create the change set (reports `FAILED` status), fix the error displayed in the **Status** field, and recreate the change set.

To create a change set (AWS CLI)

- Run the aws cloudformation create-change-set command.

  You submit your changes as command options. You can specify new parameter values, a modified template, or both. For example, the following command creates a change set named `SampleChangeSet` for the `SampleStack` stack. The change set uses the current stack's template, but with a different value for the `Purpose` parameter:

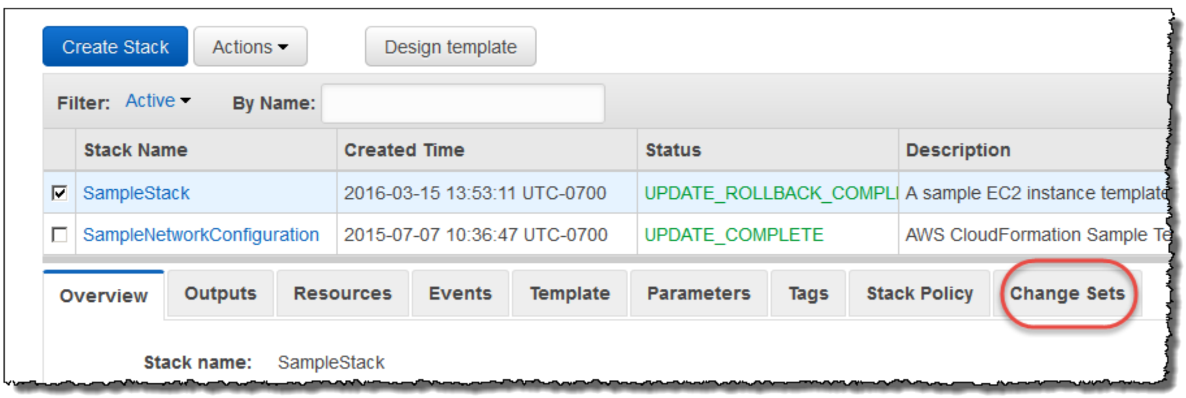
```
aws cloudformation create-change-set --stack-name arn:aws:cloudformation:us-
east-1:123456789012:stack/SampleStack/1a2345b6-0000-00a0-a123-00abc0abc000
--change-set-name SampleChangeSet --use-previous-template --
parameters ParameterKey="InstanceType",UsePreviousValue=true
 ParameterKey="KeyPairName",UsePreviousValue=true
 ParameterKey="Purpose",ParameterValue="production"
```

Viewing a Change Set

After you create a change set, you can view the proposed changes before executing them. You can use the AWS CloudFormation console, AWS CLI, or AWS CloudFormation API to view change sets. The AWS CloudFormation console provides a summary of the changes and a detailed list of changes in JSON format. The AWS CLI and AWS CloudFormation API return a detailed list of changes in JSON format.

To view a change (console)

1. In the AWS CloudFormation console, choose the stack that has the change set that you want to view.
2. In the stack detail pane, choose **Change Sets** to view a list of the stack's change sets.



3. Choose the change set that you want view.

   The AWS CloudFormation console directs you to the change set's detail page, where you can see the time the change set was created, its status, the input used to generate the change set, and a summary of changes.

   In the **Changes** section, each line represents a resource that AWS CloudFormation will add, delete, or modify. AWS CloudFormation adds a resource when you add a resource to the stack's template. AWS

CloudFormation deletes a resource when you delete an existing resource from the stack's template. AWS CloudFormation modifies a resource when you change the properties of a resource. Note that a modification can cause the resource to be interrupted or replaced (recreated). For more information about resource update behaviors, see Update Behaviors of Stack Resources (p. 89).

To focus on specific changes, use the filter view. For example, filter for a specific resource type, such as **`AWS::EC2::Instance`**. To filter for a specific resource, specify its logical or physical ID, such as **`myWebServer`** or **`i-123abcd4`**.

If you want to consider other changes before you decide which changes to make, create additional change sets.

To view a change set (AWS CLI)

1. To get the ID of the change set, run the aws cloudformation list-change-sets command.

   Specify the stack ID of the stack that has the change set that you want to view, as shown in the following example:

```
aws cloudformation list-change-sets --stack-name arn:aws:cloudformation:us-
east-1:123456789012:stack/SampleStack/1a2345b6-0000-00a0-a123-00abc0abc000
```

   AWS CloudFormation returns a list of change sets, similar to the following:

```
{
    "Summaries": [
        {
            "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
SampleStack/1a2345b6-0000-00a0-a123-00abc0abc000",
            "Status": "CREATE_COMPLETE",
            "ChangeSetName": "SampleChangeSet",
            "CreationTime": "2016-03-16T20:44:05.889Z",
            "StackName": "SampleStack",
            "ChangeSetId": "arn:aws:cloudformation:us-east-1:123456789012:changeSet/
SampleChangeSet/1a2345b6-0000-00a0-a123-00abc0abc000"
        },
        {
            "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
SampleStack/1a2345b6-0000-00a0-a123-00abc0abc000",
            "Status": "CREATE_COMPLETE",
            "ChangeSetName": "SampleChangeSet-conditional",
            "CreationTime": "2016-03-16T21:15:56.398Z",
            "StackName": "SampleStack",
            "ChangeSetId": "arn:aws:cloudformation:us-east-1:123456789012:changeSet/
SampleChangeSet-conditional/1a2345b6-0000-00a0-a123-00abc0abc000"
        },
        {
            "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
SampleStack/1a2345b6-0000-00a0-a123-00abc0abc000",
            "Status": "CREATE_COMPLETE",
            "ChangeSetName": "SampleChangeSet-replacement",
            "CreationTime": "2016-03-16T21:03:37.706Z",
            "StackName": "SampleStack",
            "ChangeSetId": "arn:aws:cloudformation:us-east-1:123456789012:changeSet/
SampleChangeSet-replacement/1a2345b6-0000-00a0-a123-00abc0abc000"
        }
    ]
}
```

2. Run the aws cloudformation describe-change-set command, specifying the ID of the change set that you want to view. For example:

```
aws cloudformation describe-change-set --change-set-name arn:aws:cloudformation:us-
east-1:123456789012:changeSet/SampleChangeSet/1a2345b6-0000-00a0-a123-00abc0abc000
```

AWS CloudFormation returns information about the specified change set:


```
{
    "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
SampleStack/1a2345b6-0000-00a0-a123-00abc0abc000",
    "Status": "CREATE_COMPLETE",
    "ChangeSetName": "SampleChangeSet-direct",
    "Parameters": [
        {
            "ParameterValue": "testing",
            "ParameterKey": "Purpose"
        },
        {
            "ParameterValue": "ellioty-useast1",
            "ParameterKey": "KeyPairName"
        },
        {
            "ParameterValue": "t2.micro",
            "ParameterKey": "InstanceType"
        }
    ],
    "Changes": [
        {
            "ResourceChange": {
                "ResourceType": "AWS::EC2::Instance",
                "PhysicalResourceId": "i-1abc23d4",
                "Details": [
                    {
                        "ChangeSource": "DirectModification",
                        "Evaluation": "Static",
                        "Target": {
                            "Attribute": "Tags",
                            "RequiresRecreation": "Never"
                        }
                    }
                ],
                "Action": "Modify",
                "Scope": [
                    "Tags"
                ],
                "LogicalResourceId": "MyEC2Instance",
                "Replacement": "False"
            },
            "Type": "Resource"
        }
    ],
    "CreationTime": "2016-03-17T23:35:25.813Z",
    "Capabilities": [],
    "StackName": "SampleStack",
    "NotificationARNs": [],
    "ChangeSetId": "arn:aws:cloudformation:us-east-1:123456789012:changeSet/
SampleChangeSet-direct/9edde307-960d-4e6e-ad66-b09ea2f20255"
}
```


The `Changes` key lists changes to resources. If you were to execute this change set, AWS CloudFormation would update the tags of the `i-1abc23d4` EC2 instance. For a description of each field, see the Change data type in the *AWS CloudFormation API Reference*.

For additional examples of change sets, see Example Change Sets (p. 99).

Executing a Change Set

To make the changes described in a change set to your stack, execute the change set.

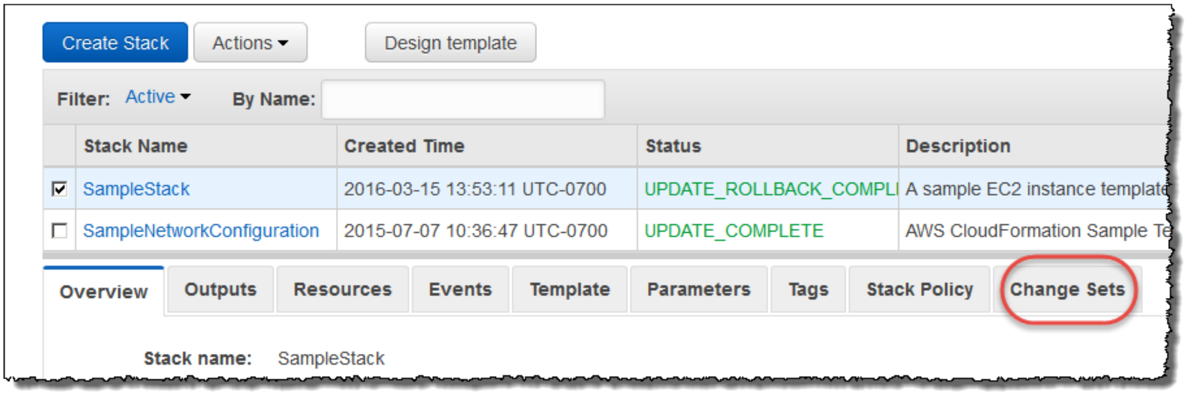
> **Important**
> After you execute a change set, AWS CloudFormation deletes all change sets that are associated with the stack because they aren't valid for the updated stack. If an update fails, you need to create a new change set.

Stack Policies and Executing a Change Set

If you execute a change set on a stack that has a stack policy associated with it, AWS CloudFormation enforces the policy when it updates the stack. You can't specify a temporary stack policy that overrides the existing policy when you execute a change set. To update a protected resource, you must update the stack policy or use the direct update (p. 106) method.

To execute a change set (console)

1. In the AWS CloudFormation console, choose the stack that you want to update.
2. In the stack detail pane, choose **Change Sets** to view a list of the stack's change sets.



3. Choose the change set that you want execute.

   The AWS CloudFormation console directs you to the detail page of the change set.
4. Choose **Execute**.

5. Confirm that this is the change set you want to execute, and then choose **Execute**.

   AWS CloudFormation immediately starts updating the stack. You can monitor the progress of the update by viewing the **Events** (p. 78) tab.

To execute a change set (AWS CLI)

- Run the aws cloudformation execute-change-set command.

  Specify the change set ID of the change set that you want to execute, as shown in the following example:

```
aws cloudformation execute-change-set --change-set-name arn:aws:cloudformation:us-
east-1:123456789012:changeSet/SampleChangeSet/1a2345b6-0000-00a0-a123-00abc0abc000
```

The command in the example executes a change set with the ID `arn:aws:cloudformation:us-east-1:123456789012:changeSet/SampleChangeSet/1a2345b6-0000-00a0-a123-00abc0abc000`.

After you run the command, AWS CloudFormation starts updating the stack. To view the stack's progress, use the aws cloudformation describe-stacks (p. 81) command.

Deleting a Change Set

Deleting a change set removes it from the list of change sets for the stack. Deleting a change set prevents you or another user from accidentally executing a change set that shouldn't be applied. AWS CloudFormation retains all change sets until you update the stack unless you delete them.

To delete a change set (console)

1. In the AWS CloudFormation console, choose the stack that contains the change set that you want to delete.
2. In the stack detail pane, choose **Change Sets** to view a list of the stack's change sets.
3. Choose the change set that you want delete.

   The AWS CloudFormation console directs you to the detail page for the change set.
4. Choose **Other Actions**, and then choose **Delete**.

5. Confirm that this is the change set you want to delete, and then choose **Delete**.

   AWS CloudFormation deletes the change set from the stack's list of change sets.

To delete a change set (AWS CLI)

- Run the aws cloudformation delete-change-set command, specifying the ID of the change set that you want to delete, as shown in the following example:

```
aws cloudformation delete-change-set --change-set-name arn:aws:cloudformation:us-
east-1:123456789012:changeSet/SampleChangeSet/1a2345b6-0000-00a0-a123-00abc0abc000
```

Example Change Sets

This section provides examples of the change sets that AWS CloudFormation would create for common stack changes. They show how to edit a template directly; modify a single input parameter; plan for resource recreation (replacements), which prevents you from losing data that wasn't backed up or interrupting applications that are running in your stack; and add and remove resources. To illustrate how change sets work, we'll walk through the changes that were submitted and discuss the resulting change set. Because each example builds on and assumes that you understand the previous example, we recommend that you read them in order. For a description of each field in a change set, see the Change data type in the *AWS CloudFormation API Reference*.

You can use the console (p. 95), AWS CLI, or AWS CloudFormation API to view change set details.

We generated each of the following change sets from a stack with the following sample template:

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Description" : "A sample EC2 instance template for testing change sets.",
  "Parameters" : {
    "Purpose" : {
      "Type" : "String",
      "Default" : "testing",
      "AllowedValues" : ["testing", "production"],
      "Description" : "The purpose of this instance."
    },
    "KeyPairName" : {
      "Type": "AWS::EC2::KeyPair::KeyName",
      "Description" : "Name of an existing EC2 KeyPair to enable SSH access to the
 instance"
    },
    "InstanceType" : {
      "Type" : "String",
      "Default" : "t2.micro",
      "AllowedValues" : ["t2.micro", "t2.small", "t2.medium"],
      "Description" : "The EC2 instance type."
    }
  },
  "Resources" : {
    "MyEC2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "KeyName" : { "Ref" : "KeyPairName" },
        "InstanceType" : { "Ref" : "InstanceType" },
        "ImageId" : "ami-8fcee4e5",
        "Tags" : [
          {
            "Key" : "Purpose",
            "Value" : { "Ref" : "Purpose" }
          }
        ]
      }
    }
  }
}
```

Directly Editing a Template

When you directly modify resources in the stack's template to generate a change set, AWS CloudFormation classifies the change as a direct modification, as opposed to changes trigged by an updated parameter value. The following change set, which added a new tag to the `i-1abc23d4` instance, is an example of a direct modification. All other input values, such as the parameter values and capabilities, are unchanged, so we'll focus on the `Changes` structure.

```
{
    "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
SampleStack/1a2345b6-0000-00a0-a123-00abc0abc000",
    "Status": "CREATE_COMPLETE",
    "ChangeSetName": "SampleChangeSet-direct",
    "Parameters": [
```

```
        {
            "ParameterValue": "testing",
            "ParameterKey": "Purpose"
        },
        {
            "ParameterValue": "MyKeyName",
            "ParameterKey": "KeyPairName"
        },
        {
            "ParameterValue": "t2.micro",
            "ParameterKey": "InstanceType"
        }
    ],
    "Changes": [
        {
            "ResourceChange": {
                "ResourceType": "AWS::EC2::Instance",
                "PhysicalResourceId": "i-1abc23d4",
                "Details": [
                    {
                        "ChangeSource": "DirectModification",
                        "Evaluation": "Static",
                        "Target": {
                            "Attribute": "Tags",
                            "RequiresRecreation": "Never"
                        }
                    }
                ],
                "Action": "Modify",
                "Scope": [
                    "Tags"
                ],
                "LogicalResourceId": "MyEC2Instance",
                "Replacement": "False"
            },
            "Type": "Resource"
        }
    ],
    "CreationTime": "2016-03-17T23:35:25.813Z",
    "Capabilities": [],
    "StackName": "SampleStack",
    "NotificationARNs": [],
    "ChangeSetId": "arn:aws:cloudformation:us-east-1:123456789012:changeSet/
SampleChangeSet-direct/1a2345b6-0000-00a0-a123-00abc0abc000"
}
```

In the `Changes` structure, there's only one `ResourceChange` structure. This structure describes information such as the type of resource AWS CloudFormation will change, the action AWS CloudFormation will take, the ID of the resource, the scope of the change, and whether the change requires a replacement (where AWS CloudFormation creates a new resource and then deletes the old one). In the example, the change set indicates that AWS CloudFormation will modify the `Tags` attribute of the `i-1abc23d4` EC2 instance, and doesn't require the instance to be replaced.

In the `Details` structure, AWS CloudFormation labels this change as a direct modification that will never require the instance to be recreated (replaced). You can confidently execute this change, knowing that AWS CloudFormation won't replace the instance.

AWS CloudFormation shows this change as a `Static` evaluation. A static evaluation means that AWS CloudFormation can determine the tag's value before executing the change set. In some cases, AWS CloudFormation can determine a value only after you execute a change set. AWS CloudFormation labels those changes as `Dynamic` evaluations. For example, if you reference an updated resource that is conditionally replaced, AWS CloudFormation can't determine whether the reference to the updated resource will change.

Modifying an Input Parameter Value

When you modify an input parameter value, AWS CloudFormation generates two changes for each resource that uses the updated parameter value. In this example, we want to highlight what those changes look like and which information you should focus on. The following example was generated by changing the value of the `Purpose` input parameter only.

The `Purpose` parameter specifies a tag key value for the EC2 instance. In the example, the parameter value was changed from `testing` to `production`. The new value is shown in the `Parameters` structure.


```
{
    "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
SampleStack/1a2345b6-0000-00a0-a123-00abc0abc000",
    "Status": "CREATE_COMPLETE",
    "ChangeSetName": "SampleChangeSet",
    "Parameters": [
        {
            "ParameterValue": "production",
            "ParameterKey": "Purpose"
        },
        {
            "ParameterValue": "MyKeyName",
            "ParameterKey": "KeyPairName"
        },
        {
            "ParameterValue": "t2.micro",
            "ParameterKey": "InstanceType"
        }
    ],
    "Changes": [
        {
            "ResourceChange": {
                "ResourceType": "AWS::EC2::Instance",
                "PhysicalResourceId": "i-1abc23d4",
                "Details": [
                    {
                        "ChangeSource": "DirectModification",
                        "Evaluation": "Dynamic",
                        "Target": {
                            "Attribute": "Tags",
                            "RequiresRecreation": "Never"
                        }
                    },
                    {
                        "CausingEntity": "Purpose",
                        "ChangeSource": "ParameterReference",
                        "Evaluation": "Static",
                        "Target": {
                            "Attribute": "Tags",
                            "RequiresRecreation": "Never"
                        }
                    }
                ],
                "Action": "Modify",
                "Scope": [
                    "Tags"
                ],
                "LogicalResourceId": "MyEC2Instance",
                "Replacement": "False"
            },
            "Type": "Resource"
        }
    ],
    "CreationTime": "2016-03-16T23:59:18.447Z",
```

```
    "Capabilities": [],
    "StackName": "SampleStack",
    "NotificationARNs": [],
    "ChangeSetId": "arn:aws:cloudformation:us-east-1:123456789012:changeSet/
SampleChangeSet/1a2345b6-0000-00a0-a123-00abc0abc000"
}
```

The `Changes` structure functions similar to way it does in the Directly Editing a Template (p. 100) example. There's only one `ResourceChange` structure; it describes a change to the `Tags` attribute of the `i-1abc23d4` EC2 instance.

However, in the `Details` structure, the change set shows two changes for the `Tags` attribute, even though only a single parameter value was changed. Resources that reference a changed parameter value (using the `Ref` intrinsic function) always result in two changes: one with a `Dynamic` evaluation and another with a `Static` evaluation. You can see these types of changes by viewing the following fields:

- For the `Static` evaluation change, view the `ChangeSource` field. In this example, the `ChangeSource` field equals `ParameterReference`, meaning that this change is a result of an updated parameter reference value. The change set must contain a similar `Dynamic` evaluation change.
- You can find the matching `Dynamic` evaluation change by comparing the `Target` structure for both changes, which will contain the same information. In this example, the `Target` structures for both changes contain the same values for the `Attribute` and `RequireRecreation` fields.

For these types of changes, focus on the static evaluation, which gives you the most detailed information about the change. In this example, the static evaluation shows that the change is the result of a change in a parameter reference value (`ParameterReference`). The exact parameter that was changed is indicated by the `CauseEntity` field (the `Purpose` parameter).

Determining the Value of the Replacement Field

The `Replacement` field in a `ResourceChange` structure indicates whether AWS CloudFormation will recreate the resource. Planning for resource recreation (replacements) prevents you from losing data that wasn't backed up or interrupting applications that are running in your stack.

The value in the `Replacement` field depends on whether a change requires a replacement, indicated by the `RequiresRecreation` field in a change's `Target` structure. For example, if the `RequiresRecreation` field is `Never`, the `Replacement` field is `False`. However, if there are multiple changes on a single resource and each change has a different value for the `RequiresRecreation` field, AWS CloudFormation updates the resource using the most intrusive behavior. In other words, if only one of the many changes requires a replacement, AWS CloudFormation must replace the resource and, therefore, sets the `Replacement` field to `True`.

The following change set was generated by changing the values for every parameter (`Purpose`, `InstanceType`, and `KeyPairName`), which are all used by the EC2 instance. With these changes, AWS CloudFormation will be required to be replace the instance because the `Replacement` field is equal to `True`.

```
{
    "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
SampleStack/1a2345b6-0000-00a0-a123-00abc0abc000",
    "Status": "CREATE_COMPLETE",
    "ChangeSetName": "SampleChangeSet-multiple",
    "Parameters": [
        {
            "ParameterValue": "production",
            "ParameterKey": "Purpose"
        },
        {
            "ParameterValue": "MyNewKeyName",
            "ParameterKey": "KeyPairName"
        },
```

```
        {
            "ParameterValue": "t2.small",
            "ParameterKey": "InstanceType"
        }
    ],
    "Changes": [
        {
            "ResourceChange": {
                "ResourceType": "AWS::EC2::Instance",
                "PhysicalResourceId": "i-7bef86f8",
                "Details": [
                    {
                        "ChangeSource": "DirectModification",
                        "Evaluation": "Dynamic",
                        "Target": {
                            "Attribute": "Properties",
                            "Name": "KeyName",
                            "RequiresRecreation": "Always"
                        }
                    },
                    {
                        "ChangeSource": "DirectModification",
                        "Evaluation": "Dynamic",
                        "Target": {
                            "Attribute": "Properties",
                            "Name": "InstanceType",
                            "RequiresRecreation": "Conditionally"
                        }
                    },
                    {
                        "ChangeSource": "DirectModification",
                        "Evaluation": "Dynamic",
                        "Target": {
                            "Attribute": "Tags",
                            "RequiresRecreation": "Never"
                        }
                    },
                    {
                        "CausingEntity": "KeyPairName",
                        "ChangeSource": "ParameterReference",
                        "Evaluation": "Static",
                        "Target": {
                            "Attribute": "Properties",
                            "Name": "KeyName",
                            "RequiresRecreation": "Always"
                        }
                    },
                    {
                        "CausingEntity": "InstanceType",
                        "ChangeSource": "ParameterReference",
                        "Evaluation": "Static",
                        "Target": {
                            "Attribute": "Properties",
                            "Name": "InstanceType",
                            "RequiresRecreation": "Conditionally"
                        }
                    },
                    {
                        "CausingEntity": "Purpose",
                        "ChangeSource": "ParameterReference",
                        "Evaluation": "Static",
                        "Target": {
                            "Attribute": "Tags",
                            "RequiresRecreation": "Never"
                        }
                    }
```

```
                ],
                "Action": "Modify",
                "Scope": [
                    "Tags",
                    "Properties"
                ],
                "LogicalResourceId": "MyEC2Instance",
                "Replacement": "True"
            },
            "Type": "Resource"
        }
    ],
    "CreationTime": "2016-03-17T00:39:35.974Z",
    "Capabilities": [],
    "StackName": "SampleStack",
    "NotificationARNs": [],
    "ChangeSetId": "arn:aws:cloudformation:us-east-1:123456789012:changeSet/
SampleChangeSet-multiple/1a2345b6-0000-00a0-a123-00abc0abc000"
}
```

Identify the change that requires the resource to be replaced by viewing each change (the static evaluations in the `Details` structure). In this example, each change has a different value for the `RequireRecreation` field, but the change to the `KeyName` property has the most intrusive update behavior, always requiring a recreation. AWS CloudFormation will replace the instance because the key name was changed.

If the key name were unchanged, the change to the `InstanceType` property would have the most intrusive update behavior (`Conditionally`), so the `Replacement` field would be `Conditionally`. To find the conditions in which AWS CloudFormation replaces the instance, view the update behavior for the `InstanceType` property.

Adding and Removing Resources

The following example was generated by submitting a modified template that removes the EC2 instance and adds an Auto Scaling group and launch configuration.

```
{
    "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
SampleStack/1a2345b6-0000-00a0-a123-00abc0abc000",
    "Status": "CREATE_COMPLETE",
    "ChangeSetName": "SampleChangeSet-addremove",
    "Parameters": [
        {
            "ParameterValue": "testing",
            "ParameterKey": "Purpose"
        },
        {
            "ParameterValue": "MyKeyName",
            "ParameterKey": "KeyPairName"
        },
        {
            "ParameterValue": "t2.micro",
            "ParameterKey": "InstanceType"
        }
    ],
    "Changes": [
        {
            "ResourceChange": {
                "Action": "Add",
                "ResourceType": "AWS::AutoScaling::AutoScalingGroup",
                "Scope": [],
                "Details": [],
                "LogicalResourceId": "AutoScalingGroup"
```

```
            },
            "Type": "Resource"
        },
        {
            "ResourceChange": {
                "Action": "Add",
                "ResourceType": "AWS::AutoScaling::LaunchConfiguration",
                "Scope": [],
                "Details": [],
                "LogicalResourceId": "LaunchConfig"
            },
            "Type": "Resource"
        },
        {
            "ResourceChange": {
                "ResourceType": "AWS::EC2::Instance",
                "PhysicalResourceId": "i-1abc23d4",
                "Details": [],
                "Action": "Remove",
                "Scope": [],
                "LogicalResourceId": "MyEC2Instance"
            },
            "Type": "Resource"
        }
    ],
    "CreationTime": "2016-03-18T01:44:08.444Z",
    "Capabilities": [],
    "StackName": "SampleStack",
    "NotificationARNs": [],
    "ChangeSetId": "arn:aws:cloudformation:us-east-1:123456789012:changeSet/
SampleChangeSet-addremove/1a2345b6-0000-00a0-a123-00abc0abc000"
}
```

In the `Changes` structure, there are three `ResourceChange` structures, one for each resource. For each resource, the `Action` field indicates whether AWS CloudFormation adds or removes the resource. The `Scope` and `Details` fields are empty because they apply only to modified resources.

For new resources, AWS CloudFormation can't determine the value of some fields until you execute the change set. For example, AWS CloudFormation doesn't provide the physical IDs of the Auto Scaling group and launch configuration because they don't exist yet. AWS CloudFormation creates the new resources when you execute the change set.

Updating Stacks Directly

When you want to quickly deploy updates to your stack, perform a direct update. With a direct update, you submit a template or input parameters that specify updates to the resources in the stack, and AWS CloudFormation immediately deploys them. If you want to use a template to make your updates, you can modify the current template and store it locally or in an S3 bucket.

For resource properties that don't support updates, you must keep the current values. To preview the changes that AWS CloudFormation will make to your stack before you update it, use change sets. For more information, see Updating Stacks Using Change Sets (p. 93).

> **Note**
> When updating a stack, AWS CloudFormation might interrupt resources or replace updated resources, depending on which properties you update. For more information about resource update behaviors, see Update Behaviors of Stack Resources (p. 89).

To update a AWS CloudFormation stack (console)

1. In the AWS CloudFormation console, from the list of stacks, select the running stack that you want to update.

2. Choose **Actions** and then **Update Stack**.
3. If you modified the stack template, specify the location of the updated template. If not, select **Use current template**.
   - For a template stored locally on your computer, select **Upload a template to Amazon S3**. Choose **Choose File** to navigate to the file and select it, and then click **Next**.
   - For a template stored in an Amazon S3 bucket, select **Specify an Amazon S3 URL**. Enter or paste the URL for the template, and then click **Next**.

     If you have a template in a versioning-enabled bucket, you can specify a specific version of the template, such as `https://s3.amazonaws.com/templates/myTemplate.template?versionId=123ab1cdeKdOW5IH4GAcYbEngcpTJTDW`. For more information, see Managing Objects in a Versioning-Enabled Bucket in the *Amazon Simple Storage Service Console User Guide*.
4. If your template contains parameters, on the **Specify Parameters** page, enter or modify the parameter values, and then click **Next**.

   AWS CloudFormation populates each parameter with the value that is currently set in the stack with the exception of parameters declared with the `NoEcho` attribute; however, you can still use current values by choosing **Use existing value**.
5. On the **Options** page, you can update the stack's service role, enter an overriding stack policy, or update the Amazon SNS notification topic. An overriding stack policy lets you update protected resources. For more information, see Prevent Updates to Stack Resources (p. 110).

   Click **Next**.
6. Review the stack information and the changes that you submitted.

   In the **Review** section, check that you submitted the correct information, such as the correct parameter values or template URL. If your template contains IAM resources, select **I acknowledge that this template may create IAM resources** to specify that you want to use IAM resources in the template. For more information about using IAM resources in templates, see Controlling Access with AWS Identity and Access Management (p. 8).

   In the **Preview your changes** section, check that AWS CloudFormation will make all the changes that you expect. For example, you can check that AWS CloudFormation adds, removes, and modifies the resources that you intended to add, remove, or modify. AWS CloudFormation generates this preview by creating a change set for the stack. For more information, see Updating Stacks Using Change Sets (p. 93).
7. Click **Update**.

   Your stack enters the **UPDATE_IN_PROGRESS** state. After it has finished updating, the state is set to **UPDATE_COMPLETE**.

   If the stack update fails, AWS CloudFormation automatically rolls back changes, and sets the state to **UPDATE_ROLLBACK_COMPLETE**.

   Note
   You can cancel an update while it's in the **UPDATE_IN_PROGRESS** state. For more information, see Canceling a Stack Update (p. 109).

To update a AWS CloudFormation stack (AWS CLI)

- Use the `aws cloudformation update-stack` command to directly update a stack. You specify the stack, and parameter values and capabilities that you want to update, and, if you want use an updated template, the name of the template.

  The following example updates the template and input parameters for the `mystack` stack:

```
PROMPT> aws cloudformation update-stack --stack-name mystack --template-url https://
s3.amazonaws.com/sample/updated.template
--parameters ParameterKey=VPCID,ParameterValue=SampleVPCID
 ParameterKey=SubnetIDs,ParameterValue=SampleSubnetID1\\,SampleSubnetID2
```

The following example updates just the `SubnetIDs` parameter values for the `mystack` stack:

```
PROMPT> aws cloudformation update-stack --stack-name mystack --use-previous-template
--parameters ParameterKey=VPCID,UsePreviousValue=true
 ParameterKey=SubnetIDs,ParameterValue=SampleSubnetID1\\,UpdatedSampleSubnetID2
```

The following example adds two stack notification topics to the `mystack` stack:

```
PROMPT> aws cloudformation update-stack --stack-name mystack --use-previous-template
--notification-arns "arn:aws:sns:us-east-1:12345678912:mytopic" "arn:aws:sns:us-
east-1:12345678912:mytopic2"
```

The following example removes all stack notification topics from the `mystack` stack:

```
PROMPT> aws cloudformation update-stack --stack-name mystack --use-previous-template
--notification-arns []
```

Monitoring the Progress of a Stack Update

You can monitor the progress of a stack update by viewing the stack's events. The console's **Events** tab displays each major step in the creation and update of the stack sorted by the time of each event with latest events on top. The start of the stack update process is marked with an UPDATE_IN_PROGRESS event for the stack:

```
2011-09-30 09:35 PDT AWS::CloudFormation::Stack MyStack UPDATE_IN_PROGRESS
```

Next are events that mark the beginning and completion of the update of each resource that was changed in the update template. For example, updating an AWS::RDS::DBInstance (p. 881) resource named MyDB would result in the following entries:

```
2011-09-30 09:35 PDT AWS::RDS::DBInstance MyDB UPDATE_COMPLETE
2011-09-30 09:35 PDT AWS::RDS::DBInstance MyDB UPDATE_IN_PROGRESS
```

The UPDATE_IN_PROGRESS event is logged when AWS CloudFormation reports that it has begun to update the resource. The UPDATE_COMPLETE event is logged when the resource is successfully created.

When AWS CloudFormation has successfully updated the stack, you will see the following event:

```
2011-09-30 09:35 PDT AWS::CloudFormation::Stack MyStack UPDATE_COMPLETE
```

If an update of a resource fails, AWS CloudFormation reports an UPDATE_FAILED event that includes a reason for the failure. For example, if your update template specified a property change that is not supported by the resource such as reducing the size of AllocatedStorage for an AWS::RDS::DBInstance (p. 881) resource, you would see events like these:

```
2011-09-30 09:36 PDT AWS::RDS::DBInstance MyDB UPDATE_FAILED Size cannot be less than
 current size; requested: 5; current: 10
2011-09-30 09:35 PDT AWS::RDS::DBInstance MyDB UPDATE_IN_PROGRESS
```

If a resource update fails, AWS CloudFormation rolls back any resources that it has updated during the upgrade to their configurations before the update. Here is an example of the events you would see during an update rollback:

```
2011-09-30 09:38 PDT AWS::CloudFormation::Stack MyStack UPDATE_ROLLBACK_COMPLETE
2011-09-30 09:38 PDT AWS::RDS::DBInstance MyDB UPDATE_COMPLETE
2011-09-30 09:37 PDT AWS::RDS::DBInstance MyDB UPDATE_IN_PROGRESS
2011-09-30 09:37 PDT AWS::CloudFormation::Stack MyStack UPDATE_ROLLBACK_IN_PROGRESS The
 following resource(s) failed to update: [MyDB]
```

Topics

To view stack events by using the console

1. In the AWS CloudFormation console, select the stack that you updated and then click the **Events** tab to view the stacks events.
2. To update the event list with the most recent events, click the refresh button in the AWS CloudFormation console.

To view stack events by using the command line

- Use the command `aws cloudformation describe-stack-events` to view the events for a stack.

Canceling a Stack Update

After a stack update has begun, you can cancel the stack update if the stack is still in the `UPDATE_IN_PROGRESS` state. After an update has finished, you cannot cancel it. You can, however, update a stack again with any previous settings.

If you cancel a stack update, the stack is rolled back to the stack configuration that existed prior to initiating the stack update.

Topics

To cancel a stack update by using the console

1. From the list of stacks in the AWS CloudFormation console, select the stack that is currently being updated (its state must be `UPDATE_IN_PROGRESS`) .

2. Choose **Actions** and then **Cancel Update**.
3. To continue canceling the update, click **Yes, Cancel Update** when prompted. Otherwise, click **Cancel** to resume the update.

The stack proceeds to the **UPDATE_ROLLBACK_IN_PROGRESS** state. After the update cancellation is complete, the stack is set to **UPDATE_ROLLBACK_COMPLETE**.

To cancel a stack update by using the command line

- Use the command `aws cloudformation cancel-update-stack` to cancel an update.

Prevent Updates to Stack Resources

When you create a stack, all update actions are allowed on all resources. By default, anyone with stack update permissions can update all of the resources in the stack. During an update, some resources might require an interruption or be completely replaced, resulting in new physical IDs or completely new storage. You can prevent stack resources (p. 390) from being unintentionally updated or deleted during a stack update by using a stack policy. A stack policy is a JSON document that defines the update actions that can be performed on designated resources.

After you set a stack policy, all of the resources in the stack are protected by default. To allow updates on specific resources, you specify an explicit `Allow` statement for those resources in your stack policy. You can define only one stack policy per stack, but, you can protect multiple resources within a single policy. A stack policy applies to all AWS CloudFormation users who attempt to update the stack. You can't associate different stack policies with different users.

A stack policy applies only during stack updates. It doesn't provide access controls like an AWS Identity and Access Management (IAM) policy. Use a stack policy only as a fail-safe mechanism to prevent accidental updates to specific stack resources. To control access to AWS resources or actions, use IAM.

Topics

Example Stack Policy

The following example stack policy prevents updates to the `ProductionDatabase` resource:

```
{
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    },
    {
      "Effect" : "Deny",
      "Action" : "Update:*",
      "Principal": "*",
```

```
      "Resource" : "LogicalResourceId/ProductionDatabase"
    }
  ]
}
```

When you set a stack policy, all resources are protected by default. To allow updates on all resources, we add an `Allow` statement that allows all actions on all resources. Although the `Allow` statement specifies all resources, the explicit `Deny` statement overrides it for the resource with the `ProductionDatabase` logical ID. This `Deny` statement prevents all update actions, such as replacement or deletion, on the `ProductionDatabase` resource.

The `Principal` element is required, but supports only the wild card (`*`), which means that the statement applies to all principals.

> **Note**
> During a stack update, AWS CloudFormation automatically updates resources that depend on other updated resources. For example, AWS CloudFormation updates a resource that references an updated resource. AWS CloudFormation makes no physical changes, such as the resources' ID, to automatically updated resources, but if a stack policy is associated with those resources, you must have permission to update them.

Defining a Stack Policy

When you create a stack, no stack policy is set, so all update actions are allowed on all resources. To protect stack resources from update actions, define a stack policy and then set it on your stack. A stack policy is a JSON document that defines the AWS CloudFormation stack update actions that AWS CloudFormation users can perform and the resources that the actions apply to. You set the stack policy when you create a stack, by specifying a text file that contains your stack policy or typing it out. When you set a stack policy on your stack, any update not explicitly allowed is denied by default.

You define a stack policy with five elements: `Effect`, `Action`, `Principal`, `Resource`, and `Condition`. The following pseudo code shows stack policy syntax.

```
{
  "Statement" : [
    {
      "Effect" : "Deny_or_Allow",
      "Action" : "update_actions",
      "Principal" : "*",
      "Resource" : "LogicalResourceId/resource_logical_ID",
      "Condition" : {
        "StringEquals_or_StringLike" : {
          "ResourceType" : [resource_type, ...]
        }
      }
    }
  ]
}
```

Effect

Determines whether the actions that you specify are denied or allowed on the resource(s) that you specify. You can specify only `Deny` or `Allow`, such as:

```
"Effect" : "Deny"
```

> **Important**
> If a stack policy includes overlapping statements (both allowing and denying updates on a resource), a `Deny` statement always overrides an `Allow` statement. To ensure that a resource is protected, use a `Deny` statement for that resource.

Action

Specifies the update actions that are denied or allowed:

Update:Modify

Specifies update actions during which resources might experience no interruptions or some interruptions while changes are being applied. All resources maintain their physical IDs.

Update:Replace

Specifies update actions during which resources are recreated. AWS CloudFormation creates a new resource with the specified updates and then deletes the old resource. Because the resource is recreated, the physical ID of the new resource might be different.

Update:Delete

Specifies update actions during which resources are removed. Updates that completely remove resources from a stack template require this action.

Update:*

Specifies all update actions. The asterisk is a wild card that represents all update actions.

The following example shows how to specify just the replace and delete actions:

```
"Action" : ["Update:Replace", "Update:Delete"]
```

To allow all update actions except for one, use `NotAction`. For example, to allow all update actions except for `Update:Delete`, use `NotAction`, as shown in this example:

```
{
  "Statement" : [
    {
      "Effect" : "Allow",
      "NotAction" : "Update:Delete",
      "Principal": "*",
      "Resource" : "*"
    }
  ]
}
```

For more information about stack updates, see AWS CloudFormation Stacks Updates (p. 89).

Principal

The `Principal` element specifies the entity that the policy applies to. This element is required but supports only the wild card (`*`), which means that the policy applies to all principals.

Resource

Specifies the logical IDs of the resources that the policy applies to. To specify types of resources (p. 390), use the `Condition` element.

To specify a single resource, use its logical ID. For example:

```
"Resource" : ["LogicalResourceId/myEC2instance"]
```

You can use a wild card with logical IDs. For example, if you use a common logical ID prefix for all related resources, you can specify all of them with a wild card:

```
"Resource" : ["LogicalResourceId/CriticalResource*"]
```

You can also use a `Not` element with resources. For example, to allow updates to all resources except for one, use a `NotResource` element to protect that resource:

```
{
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "NotResource" : "LogicalResourceId/ProductionDatabase"
    }
  ]
}
```

When you set a stack policy, any update not explicitly allowed is denied. By allowing updates to all resources except for the `ProductionDatabase` resource, you deny updates to the `ProductionDatabase` resource.

Conditions

Specifies the resource type (p. 390) that the policy applies to. To specify the logical IDs of specific resources, use the `Resource` element.

You can specify a resource type, such as all EC2 and RDS DB instances, as shown in the following example:

```
{
  "Statement" : [
  {
    "Effect" : "Deny",
    "Principal" : "*",
    "Action" : "Update:*",
    "Resource" : "*",
    "Condition" : {
      "StringEquals" : {
        "ResourceType" : ["AWS::EC2::Instance", "AWS::RDS::DBInstance"]
      }
    }
  },
  {
    "Effect" : "Allow",
    "Principal" : "*",
    "Action" : "Update:*",
    "Resource" : "*"
  }
  ]
}
```

The `Allow` statement grants update permissions to all resources and the `Deny` statement denies updates to EC2 and RDS DB instances. The `Deny` statement always overrides allow actions.

You can use a wild card with resource types. For example, you can deny update permissions to all Amazon EC2 resources—such as instances, security groups, and subnets—by using a wild card, as shown in the following example:

```
"Condition" : {
  "StringLike" : {
    "ResourceType" : ["AWS::EC2::*"]
```

```
  }
}
```

You must use the `StringLike` condition when you use wild cards.

Setting a Stack Policy

You can use the console or AWS CLI to apply a stack policy when you create a stack. You can also use the AWS CLI to apply a stack policy to an existing stack. After you apply a stack policy, you can't remove it from the stack, but you can use the AWS CLI to modify it.

Stack policies apply to all AWS CloudFormation users who attempt to update the stack. You can't associate different stack policies with different users.

For information about writing stack policies, see Defining a Stack Policy (p. 111).

To set a stack policy when you create a stack (console)

1. Open the AWS CloudFormation console at https://console.aws.amazon.com/cloudformation/.
2. On the **CloudFormation Stacks** page, choose **Create Stack**.
3. In the Create Stack wizard, on the **Options** page, expand the **Advanced** section.
4. Choose `Browse`, and then choose the file that contains the stack policy, or type the policy in the `Stack policy` text box.

To set a stack policy when you create a stack (CLI)

- Use the `aws cloudformation create-stack` command with the `--stack-policy-body` option to type in a modified policy or the `--stack-policy-url` option to specify a file containing the policy.

To set a stack policy on an existing stack (CLI only)

- Use the `aws cloudformation set-stack-policy` command with the `--stack-policy-body` option to type in a modified policy or the `--stack-policy-url` option to specify a file containing the policy.

  Note
  To add a policy to an existing stack, you must have permission to the AWS CloudFormation `SetStackPolicy` action.

Updating Protected Resources

To update protected resources, create a temporary policy that overrides the stack policy and allows updates on those resources. Specify the override policy when you update the stack. The override policy doesn't permanently change the stack policy.

To update protected resources, you must have permission to use the AWS CloudFormation `SetStackPolicy` action. For information about setting AWS CloudFormation permissions, see Controlling Access with AWS Identity and Access Management (p. 8).

Note
During a stack update, AWS CloudFormation automatically updates resources that depend on other updated resources. For example, AWS CloudFormation updates a resource that references an updated resource. AWS CloudFormation makes no physical changes, such as the resources' ID, to automatically updated resources, but if a stack policy is associated with those resources, you must have permission to update them.

To update a protected resource (console)

1. Open the AWS CloudFormation console at https://console.aws.amazon.com/cloudformation/.
2. Select the stack that you want to update, choose **Actions**, and then choose **Update Stack**.
3. If you modified the stack template, specify the location of the updated template. If not, choose **Use current template**.

   - For a template stored locally on your computer, choose **Upload a template to Amazon S3**. Choose **Choose File** to navigate to the file, select it, and then choose **Next**.
   - For a template stored in an Amazon S3 bucket, choose **Specify an Amazon S3 URL**. Type or paste the URL for the template, and then choose **Next**.

     If you have a template in a versioning-enabled bucket, you can specify a specific version of the template, such as `https://s3.amazonaws.com/templates/myTemplate.template?versionId=123ab1cdeKdOW5IH4GAcYbEngcpTJTDW`. For more information, see Managing Objects in a Versioning-Enabled Bucket in the *Amazon Simple Storage Service Console User Guide*.
4. If your template contains parameters, on the **Specify Parameters** page, enter or modify the parameter values, and then choose **Next**.

   AWS CloudFormation populates each parameter with the value that is currently set in the stack except for parameters declared with the `NoEcho` attribute. You can use current values for those parameters by choosing **Use existing value**.
5. On the **Options** page, choose the file that contains the overriding stack policy or type a policy, and then choose **Next**. The override policy must specify an `Allow` statement for the protected resources that you want to update.

   For example, to update all protected resources, specify a temporary override policy that allows all updates:

```
{
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    }
  ]
}
```

   Note
   AWS CloudFormation applies the override policy only during this update. The override policy doesn't permanently change the stack policy. To modify a stack policy, see Modifying a Stack Policy (p. 116).
6. Review the stack information and the changes that you submitted.

   In the **Review** section, check that you submitted the correct information, such as the correct parameter values or template URL. If your template contains IAM resources, choose **I acknowledge that this template may create IAM resources** to specify that you want to use IAM resources in the template. For more information about using IAM resources in templates, see Controlling Access with AWS Identity and Access Management (p. 8).

   In the **Preview your changes** section, check that AWS CloudFormation will make all the changes that you expect. For example, check that AWS CloudFormation adds, removes, and modifies the resources that you intended to add, remove, or modify. AWS CloudFormation generates this preview by creating a change set for the stack. For more information, see Updating Stacks Using Change Sets (p. 93).
7. Choose **Update**.

Your stack enters the **UPDATE_IN_PROGRESS** state. After it has finished updating, the state is set to **UPDATE_COMPLETE**.

If the stack update fails, AWS CloudFormation automatically rolls back changes, and sets the state to **UPDATE_ROLLBACK_COMPLETE**.

To update a protected resource (CLI)

- Use the `aws cloudformation update-stack` command with the `--stack-policy-during-update-body` option to type in a modified policy or the `--stack-policy-during-update-url` option to specify a file containing the policy.

  Note
  AWS CloudFormation applies the override policy only during this update. The override policy doesn't permanently change the stack policy. To modify a stack policy, see Modifying a Stack Policy (p. 116).

Modifying a Stack Policy

To protect additional resources or to remove protection from resources, modify the stack policy. For example, when you add a database that you want to protect to your stack, add a `Deny` statement for that database to the stack policy. To modify the policy, you must have permission to use the `SetStackPolicy` action.

Use the AWS CLI to modify stack policies.

To modify a stack policy (CLI)

- Use the `aws cloudformation set-stack-policy` command with the `--stack-policy-body` option to type in a modified policy or the `--stack-policy-url` option to specify a file containing the policy.

You can't delete a stack policy. To remove all protection from all resources, you modify the policy to explicitly allow all actions on all resources. The following policy allows all updates on all resources:

```
{
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    }
  ]
}
```

More Example Stack Policies

The following example policies show how to prevent updates to all stack resources and to specific resources, and prevent specific types of updates.

Prevent Updates to All Stack Resources

To prevent updates to all stack resources, the following policy specifies a `Deny` statement for all update actions on all resources.

```
{
```

```
  "Statement" : [
    {
      "Effect" : "Deny",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    }
  ]
}
```

Prevent Updates to a Single Resource

The following policy denies all update actions on the database with the `MyDatabase` logical ID. It allows all update actions on all other stack resources with an `Allow` statement. The `Allow` statement doesn't apply to the `MyDatabase` resource because the `Deny` statement always overrides allow actions.

```
{
  "Statement" : [
    {
      "Effect" : "Deny",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "LogicalResourceId/MyDatabase"
    },
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    }
  ]
}
```

You can achieve the same result as the previous example by using a default denial. When you set a stack policy, AWS CloudFormation denies any update that is not explicitly allowed. The following policy allows updates to all resources except for the `ProductionDatabase` resource, which is denied by default.

```
{
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "NotResource" : "LogicalResourceId/ProductionDatabase"
    }
  ]
}
```

Important
There is risk in using a default denial. If you have an `Allow` statement elsewhere in the policy (such as an `Allow` statement that uses a wildcard), you might unknowingly grant update permission to resources that you don't intend to. Because an explicit denial overrides any allow actions, you can ensure that a resource is protected by using a `Deny` statement.

Prevent Updates to All Instances of a Resource Type

The following policy denies all update actions on the RDS DB instance resource type. It allows all update actions on all other stack resources with an `Allow` statement. The `Allow` statement doesn't apply to the RDS DB instance resources because a `Deny` statement always overrides allow actions.

```
{
  "Statement" : [
    {
      "Effect" : "Deny",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*",
      "Condition" : {
        "StringEquals" : {
          "ResourceType" : ["AWS::RDS::DBInstance"]
        }
      }
    },
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    }
  ]
}
```

Prevent Replacement Updates for an Instance

The following policy denies updates that would cause a replacement of the instance with the `MyInstance` logical ID. It allows all update actions on all other stack resources with an `Allow` statement. The `Allow` statement doesn't apply to the `MyInstance` resource because the `Deny` statement always overrides allow actions.

```
{
  "Statement" : [
    {
      "Effect" : "Deny",
      "Action" : "Update:Replace",
      "Principal": "*",
      "Resource" : "LogicalResourceId/MyInstance"
    },
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    }
  ]
}
```

Prevent Updates to Nested Stacks

The following policy denies all update actions on the AWS CloudFormation stack resource type (nested stacks). It allows all update actions on all other stack resources with an `Allow` statement. The `Allow` statement doesn't apply to the AWS CloudFormationstack resources because the `Deny` statement always overrides allow actions.

```
{
  "Statement" : [
    {
      "Effect" : "Deny",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*",
```

```
      "Condition" : {
        "StringEquals" : {
          "ResourceType" : ["AWS::CloudFormation::Stack"]
        }
      }
    },
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    }
  ]
}
```

Continue Rolling Back an Update

A stack goes into the `UPDATE_ROLLBACK_FAILED` state when AWS CloudFormation cannot roll back all changes during an update. For example, you might have a stack that begins to roll back to an old database instance that was deleted outside of AWS CloudFormation. Because AWS CloudFormation doesn't know that the database was deleted, it assumes that the database instance still exists and attempts to roll back to it, causing the update rollback to fail.

When a stack is in the `UPDATE_ROLLBACK_FAILED` state, you can continue to roll it back to a working state (`UPDATE_ROLLBACK_COMPLETE`). You can't update a stack that is in the `UPDATE_ROLLBACK_FAILED` state. However, if you can continue to roll it back, you can return the stack to its original settings and then try to update it again.

In most cases, you must fix the error that causes the update rollback to fail before you can continue to roll back your stack. In other cases, you can continue to roll back the update without any changes, for example when a stack operation times out.

> **Note**
> If you use nested stacks, rolling back the parent stack will attempt to roll back all the child stacks as well.

To continue rolling back an update (console)

1. Open the AWS CloudFormation console at https://console.aws.amazon.com/cloudformation/.
2. Select the stack that you want to update, choose **Actions**, and then choose **Continue update rollback**.

   If none of the solutions in the troubleshooting guide worked, you can use the advanced option to skip the resources that AWS CloudFormation can't successfully roll back. You must look up (p. 78) and type the logical ID's of the resources that you want to skip. Specify only resources that are in the `UPDATE_FAILED` state because the rollback failed, not for other reasons, for example, because an update was canceled.

   > **Warning**
   > AWS CloudFormation sets the status of the specified resources to `UPDATE_COMPLETE` and continues to roll back the stack. After the rollback is complete, the state of the skipped resources will be inconsistent with the state of the resources in the stack template. Before performing another stack update, you must update the stack or resources to be consistent with each other. If you don't, subsequent stack updates might fail, and the stack will become unrecoverable.

   Specify the minimum number of resources required to successfully roll back your stack. For example, a failed resource update might cause dependent resources to fail. In this case, it might not be necessary to skip the dependent resources.

To specify resources in a nested stack, use the following format: *NestedStackName.ResourceLogicalID*. You can specify a nested stack resource (the logical ID of an `AWS::CloudFormation::Stack` resource) only if it's in one of the following states: `DELETE_IN_PROGRESS`, `DELETE_COMPLETE`, or `DELETE_FAILED`.

To continue rolling back an update (AWS CLI)

- Use the `aws cloudformation continue-update-rollback` command with the `stack-name` option to specify the ID of the stack that you want to continue to roll back.

Exporting Stack Output Values

To share information between stacks, export a stack's output values. Other stacks that are in the same AWS account and region can import the exported values. For example, you might have a single networking stack that exports the IDs of a subnet and security group for public web servers. Stacks with a public web server can easily import those networking resources. You don't need to hard code resource IDs in the stack's template or pass IDs as input parameters.

To export a stack's output value, use the `Export` field in the Output (p. 155) section of the stack's template. To import those values, use the `Fn::ImportValue` (p. 1334) function in the template for the other stacks. For a walkthrough and sample templates, see Walkthrough: Refer to Resource Outputs in Another AWS CloudFormation Stack (p. 200).

Note
After another stack imports an output value, you can't delete the stack that is exporting the output value or modify the exported output value. All of the imports must be removed before you can delete the exporting stack or modify the output value.

Exporting Stack Output Values vs. Using Nested Stacks

A nested stack is a stack that you create within another stack by using the `AWS::CloudFormation::Stack` (p. 485) resource. With nested stacks, you deploy and manage all resources from a single stack. You can use outputs from one stack in the nested stack group as inputs to another stack in the group. This differs from exporting values.

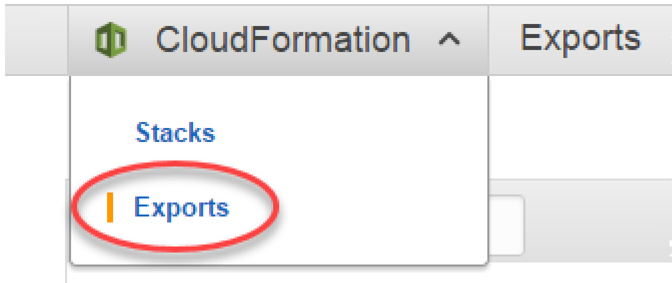
If you want to isolate information sharing to within a nested stack group, we suggest that you use nested stacks. To share information with other stacks (not just within the group of nested stacks), export values. For example, you can create a single stack with a subnet and then export its ID. Other stacks can use that subnet by importing its ID; each stack doesn't need to create its own subnet. Note that as long as stacks are importing the subnet ID, you can't change or delete it.

Listing Exported Output Values

To see the values that you can import, list all of the exported output values by using the AWS CloudFormation console, AWS CLI, or AWS CloudFormation API. AWS CloudFormation shows the names and values of the exported outputs for the current region and the stack from which the outputs are exported. To reference an exported output value in a stack's template, use the export name and the `Fn::ImportValue` (p. 1334) function.

To list exported output values (console)

- In the AWS CloudFormation console, from the **CloudFormation** drop-down menu, choose **Exports**.



To list exported output values (AWS CLI)

- Run the aws cloudformation list-exports command.

To list exported output values (API)

- Run the `ListExports` API.

Listing Stacks That Import an Exported Output Value

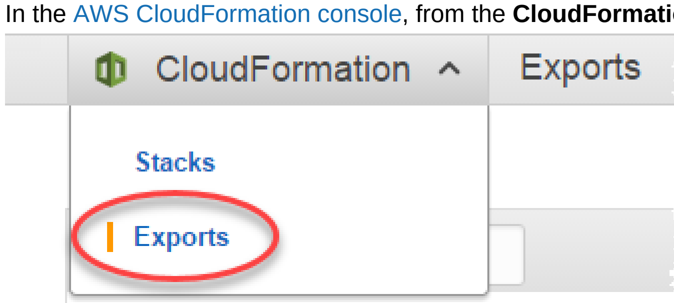
When you export an output value, stacks that are in the same AWS account and region can import that value. To see which stacks are importing a particular output value, use the list import action.

To delete or modify exported output values, use the `ListImports` action to track which stacks are importing them., and then modify those stacks to remove the `Fn::ImportValue` (p. 1334) functions that reference the output values. You must remove all of the imports that reference exported output values before you can delete or modify the exported output values.

For more information about exporting and importing output values, see Exporting Stack Output Values (p. 120).

To list stacks that import an exported output value (console)

1. In the AWS CloudFormation console, from the **CloudFormation** drop-down menu, choose **Exports**.



2. From the list of exported output values, choose the value. The `Imports` section of the detail page lists all of the stacks that are importing the value.

To list stacks that import an exported output value (CLI)

- Run the aws cloudformation list-imports command, providing the name of the exported output value.

AWS CloudFormation returns a list of stacks that are importing the value.

To list stacks that import an exported output value (API)

- Run the `ListImports` API, providing the name of the exported output value.

  AWS CloudFormation returns a list of stacks that are importing the value.

Working with Microsoft Windows Stacks on AWS CloudFormation

AWS CloudFormation allows you to create Microsoft Windows stacks based on Amazon EC2 Windows Amazon Machine Images (AMIs) and provides you with the ability to install software, to use remote desktop to access your stack, and to update and configure your stack.

The topics in this section are designed to demonstrate how common tasks related to creation and management of Windows instances are accomplished with AWS CloudFormation.

In This Section

Microsoft Windows Amazon Machine Images (AMIs) and AWS CloudFormation Templates

With AWS CloudFormation, you can create Microsoft Windows stacks for running Windows server instances. A number of pre-configured templates are available to launch directly from the AWS CloudFormation Sample Templates page, such as the following templates:

- Windows_Single_Server_SharePoint_Foundation.template - SharePoint® Foundation 2010 running on Microsoft Windows Server® 2008 R2
- Windows_Single_Server_Active_Directory.template - Create a single server installation of Active Directory running on Microsoft Windows Server® 2008 R2.
- Windows_Roles_And_Features.template - Create a single server specifying server roles running on Microsoft Windows Server® 2008 R2.
- ElasticBeanstalk_Windows_Sample.template - Launch an AWS Elastic Beanstalk sample application on Windows Server 2008 R2 running IIS 7.5.

> **Note**
> Microsoft, Windows Server, and SharePoint are trademarks of the Microsoft group of companies.

Although these stacks are already configured, you can use any EC2 Windows AMI as the basis of an AWS CloudFormation Windows stack.

Bootstrapping AWS CloudFormation Windows Stacks

This topic describes how to bootstrap a Windows stack and troubleshoot stack creation issues. If you will be creating your own Windows image for use with CloudFormation, see the information at Configuring a

Windows Instance Using EC2ConfigService in the *Amazon EC2 Microsoft Windows Guide* for instructions. You must set up a Windows instance with EC2ConfigService for it to work with the AWS CloudFormation bootstrapping tools.

Topics

Example of Bootstrapping a Windows Stack

For the purposes of illustration, we'll examine the AWS CloudFormation single-instance Sharepoint server template, which can be viewed, in its entirety, at the following URL:

- https://s3.amazonaws.com/cloudformation-templates-us-east-1/Windows_Single_Server_SharePoint_Foundation.template

This example demonstrates how to:

- Create an IAM User and Security Group for access to the instance
- Configure initialization files: `cfn-credentials`, `cfn-hup.conf`, and `cfn-auto-reloader.conf`
- Download and install a package such as Sharepoint Foundation 2010 on the server instance.
- Use a WaitCondition to ensure resources are ready
- Retrieve an IP for the instance with Amazon Elastic IP (EIP).

The AWS CloudFormation helper script `cfn-init` is used to perform each of these actions, based on information in the AWS::CloudFormation::Init (p. 470) resource in the Windows Single Server Sharepoint Foundation template.

The AWS::CloudFormation::Init section is named "SharePointFoundation", and begins with a standard declaration:

```
"SharePointFoundation": {
   "Type" : "AWS::EC2::Instance",
   "Metadata" : {
     "AWS::CloudFormation::Init" : {
       "config" : {
```

After this, the **files** section of AWS::CloudFormation::Init is declared:

```
"files" : {
  "c:\\cfn\\cfn-hup.conf" : {
    "content" : { "Fn::Join" : ["", [
      "[main]\n",
      "stack=", { "Ref" : "AWS::StackName" }, "\n",
      "region=", { "Ref" : "AWS::Region" }, "\n"
      ]]}
  },
  "c:\\cfn\\hooks.d\\cfn-auto-reloader.conf" : {
    "content": { "Fn::Join" : ["", [
      "[cfn-auto-reloader-hook]\n",
      "triggers=post.update\n",
      "path=Resources.SharePointFoundation.Metadata.AWS::CloudFormation::Init\n",
```

```
      "action=cfn-init.exe -v -s ", { "Ref" : "AWS::StackName" },
                                     " -r SharePointFoundation",
                                     " --region ", { "Ref" : "AWS::Region" }, "\n"
    ]]}
  },
  "C:\\SharePoint\\SharePointFoundation2010.exe" : {
    "source" : "http://d3adzpja92utk0.cloudfront.net/SharePointFoundation.exe"
  }
},
```

Three files are created here and placed in the `C:\cfn` directory on the server instance. They are:

- `cfn-hup.conf`, the configuration file for cfn-hup.
- `cfn-auto-reloader.conf`, the configuration file for the hook used by cfn-hup to initiate an update (calling cfn-init) when the metadata in AWS::CloudFormation::Init changes.

There is also a file that is downloaded to the server: `SharePointFoundation.exe`. This file is used to install SharePoint on the server instance.

Important

Since paths on Windows use a backslash ('\') character, you must always remember to properly escape all backslashes by prepending another backslash whenever you refer to a Windows path in the AWS CloudFormation template.

Next is the **commands** section, which are `cmd.exe` commands.

```
"commands" : {
  "1-extract" : {
    "command" : "C:\\SharePoint\\SharePointFoundation2010.exe /extract:C:\\SharePoint\
\SPF2010 /quiet /log:C:\\SharePoint\\SharePointFoundation2010-extract.log"
  },
  "2-prereq" : {
    "command" : "C:\\SharePoint\\SPF2010\\PrerequisiteInstaller.exe /unattended"
  },
  "3-install" : {
    "command" : "C:\\SharePoint\\SPF2010\\setup.exe /config C:\\SharePoint\\SPF2010\\Files\
\SetupSilent\\config.xml"
  }
```

Because commands in the instance are processed in *alphabetical order by name*, each command has been prepended with a number indicating its desired execution order. Thus, we can make sure that the installation package is first extracted, all prerequisites are then installed, and finally, installation of SharePoint is started.

Next is the **Properties** section:

```
"Properties": {
  "InstanceType" : { "Ref" : "InstanceType" },
  "ImageId" : { "Fn::FindInMap" : [ "AWSRegionArch2AMI", { "Ref" : "AWS::Region" },
              { "Fn::FindInMap" : [ "AWSInstanceType2Arch", { "Ref" : "InstanceType" },
 "Arch" ] } ] },
  "SecurityGroups" : [ {"Ref" : "SharePointFoundationSecurityGroup"} ],
  "KeyName" : { "Ref" : "KeyPairName" },
  "UserData" : { "Fn::Base64" : { "Fn::Join" : ["", [
    "<script>\n",

    "cfn-init.exe -v -s ", { "Ref" : "AWS::StackName" },
    " -r SharePointFoundation",
    " --region ", { "Ref" : "AWS::Region" }, "\n",
```

```
    "cfn-signal.exe -e %ERRORLEVEL% ", { "Fn::Base64" : { "Ref" :
 "SharePointFoundationWaitHandle" }}, "\n",

    "</script>"
    ]]}}
  }
```

In this section, the `UserData` property contains a `cmd.exe` script that will be executed by `cfn-init`, surrounded by <script> tags. You can use a Windows Powershell script here instead by surrounding your script with <powershell> tags. For Windows stacks, you must base64 encode the wait condition handle URL again.

SharePointFoundationWaitHandle is referenced here and run with `cfn-signal`. The **WaitConditionHandle** and associated **WaitCondition** are declared next in the template:

```
"SharePointFoundationWaitHandle" : {
   "Type" : "AWS::CloudFormation::WaitConditionHandle"
},

"SharePointFoundationWaitCondition" : {
   "Type" : "AWS::CloudFormation::WaitCondition",
   "DependsOn" : "SharePointFoundation",
   "Properties" : {
     "Handle" : {"Ref" : "SharePointFoundationWaitHandle"},
     "Timeout" : "3600"
   }
}
```

Since executing all of the steps and installing SharePoint might take a while, but not an entire hour, the WaitCondition waits an hour (3600 seconds) before timing out.

If all goes well, an Elastic IP is used to provide access to the SharePoint instance:

```
"Outputs" : {
 "SharePointFoundationURL" : {
   "Value" : { "Fn::Join" : ["", ["http://", { "Ref" : "SharePointFoundationEIP" } ]] },
   "Description" : "SharePoint Team Site URL. Please retrieve Administrator password of the
 instance and use it to access the URL"
 }
```

Once stack creation is complete, the IP address supplied by EIP will be displayed in the **Outputs** tab of the AWS CloudFormation console. However, before you can access the instance you will need to retrieve the auto-generated temporary Administrator password for the instance. For more information, see Connecting to Your Windows Instance Using RDP in the *Amazon EC2 User Guide for Windows Instances*.

How to Manage Windows Services

You manage Windows services in the same way as Linux services, except that you use a `windows` key instead of `sysvinit`. The following example starts the `cfn-hup` service, sets it to Automatic, and restarts the service if cfn-init modifies the `c:\cfn\cfn-hup.conf` or `c:\cfn\hooks.d\cfn-auto-reloader.conf` configuration files.

```
"services" : {
  "windows" : {
    "cfn-hup" : {
```

```
      "enabled" : "true",
      "ensureRunning" : "true",
      "files" : ["c:\\cfn\\cfn-hup.conf", "c:\\cfn\\hooks.d\\cfn-auto-reloader.conf"]
    }
  }
}
```

You can manage other Windows services in the same way by using the name—not the display name—to reference the service.

How to Troubleshoot Stack Creation Issues

If your stack fails during creation, the default behavior is to Rollback on failure. While this is normally a good default because it avoids unnecessary charges, it makes it difficult to debug why your stack creation is failing.

To turn this behavior off, click **Show Advanced Options** when creating your stack with the AWS CloudFormation console, and click the **No** selector next to **Rollback on failure**. This will allow you to log into your instance and view the logfiles to pinpoint issues encountered when running your startup scripts.

Important logs to look at are:

- The EC2 configuration log at `C:\Program Files\Amazon\Ec2ConfigService\Logs\Ec2ConfigLog.txt`
- The **cfn-init** log at `C:\cfn\log\cfn-init.log`

Working with AWS CloudFormation Templates

To provision and configure your stack resources, you must understand AWS CloudFormation templates, which are formatted text files in JSON or YAML. These templates describe the resources that you want to provision in your AWS CloudFormation stacks. You can use AWS CloudFormation Designer or any text editor to create and save templates. For information about the structure and syntax of a template, see Template Anatomy (p. 128).

If you're unfamiliar with JSON or YAML, you can use AWS CloudFormation Designer to help you get started with AWS CloudFormation templates. AWS CloudFormation Designer is a tool for visually creating and modifying templates. For more information, see What Is AWS CloudFormation Designer? (p. 157).

Template Snippets (p. 231) provides examples that demonstrate how to write templates for a particular resource. For example, you can view snippets for Amazon EC2 instances, Amazon S3 domains, AWS CloudFormation mappings, and more. Snippets are grouped by resource, with general-purpose AWS CloudFormation snippets in General Template Snippets (p. 232).

For details about the supported resources, type names, intrinsic functions, and pseudo parameters you can use in your templates, see Template Reference (p. 390).

Topics

AWS CloudFormation Template Formats

You can author AWS CloudFormation templates in JSON or YAML formats. We support all AWS CloudFormation features and functions for both formats, including in AWS CloudFormation Designer.

When deciding which format to use, pick the format that you're most comfortable working in. Also consider that YAML inherently provides some features, such as commenting, that aren't available in JSON.

> **Important**
> We recommend that you not add `#` YAML comments to your templates in Designer. If your YAML template has `#` comments, Designer does not preserve those comments when converting the template to JSON. In addition, if you modify your template in Designer (for example, if you move a resource on the canvas), your comments are lost.

You can add comments to the AWS CloudFormation templates you create outside of Designer. The following example shows a YAML template with inline comments.

```
AWSTemplateFormatVersion: "2010-09-09"
Description: A sample template
Resources:
  MyEC2Instance: #An inline comment
    Type: "AWS::EC2::Instance"
    Properties:
      ImageId: "ami-2f726546" #Another comment -- This is a Linux AMI
      InstanceType: t1.micro
      KeyName: testkey
      BlockDeviceMappings:
        -
          DeviceName: /dev/sdm
          Ebs:
            VolumeType: io1
            Iops: 200
            DeleteOnTermination: false
            VolumeSize: 20
```

For more information about the template syntax for each format, see Template Anatomy (p. 128).

AWS CloudFormation supports the following JSON and YAML specifications:

JSON

AWS CloudFormation follows the ECMA-404 JSON standard. For more information about the JSON format, see http://www.json.org.

YAML

AWS CloudFormation supports the YAML Version 1.1 specification with a few exceptions. AWS CloudFormation doesn't support the following features:

- The `binary`, `omap`, `pairs`, `set`, and `timestamp` tags
- Aliases
- Hash merges

For more information about YAML, see http://www.yaml.org.

Template Anatomy

A template is a JSON- or YAML-formatted text file that describes your AWS infrastructure. The following examples show an AWS CloudFormation template structure and its sections.

JSON

The following example shows a JSON-formatted template fragment.

```
{
  "AWSTemplateFormatVersion" : "version date",

  "Description" : "JSON string",

  "Metadata" : {
    template metadata
  },

  "Parameters" : {
    set of parameters
  },

  "Mappings" : {
    set of mappings
  },

  "Conditions" : {
    set of conditions
  },

  "Transform" : {
    set of transforms
  },

  "Resources" : {
    set of resources
  },

  "Outputs" : {
    set of outputs
  }
}
```

YAML

The following example shows a YAML-formatted template fragment.

```
---
AWSTemplateFormatVersion: "version date"

Description:
  String

Metadata:
  template metadata

Parameters:
  set of parameters

Mappings:
  set of mappings

Conditions:
  set of conditions

Transform:
  set of transforms

Resources:
  set of resources

Outputs:
```

```
set of outputs
```

Template Sections

Templates include several major sections. The `Resources` section is the only required section. Some sections in a template can be in any order. However, as you build your template, it might be helpful to use the logical ordering of the following list, as values in one section might refer to values from a previous section. The list gives a brief overview of each section.

Format Version (optional) (p. 131)

Specifies the AWS CloudFormation template version that the template conforms to. The template format version is not the same as the API or WSDL version. The template format version can change independently of the API and WSDL versions.

Description (optional) (p. 131)

A text string that describes the template. This section must always follow the template format version section.

Metadata (optional) (p. 131)

Objects that provide additional information about the template.

Parameters (optional) (p. 132)

Specifies values that you can pass in to your template at runtime (when you create or update a stack). You can refer to parameters in the `Resources` and `Outputs` sections of the template.

Mappings (optional) (p. 140)

A mapping of keys and associated values that you can use to specify conditional parameter values, similar to a lookup table. You can match a key to a corresponding value by using the Fn::FindInMap (p. 1322) intrinsic function in the `Resources` and `Outputs` section.

Conditions (optional) (p. 144)

Defines conditions that control whether certain resources are created or whether certain resource properties are assigned a value during stack creation or update. For example, you could conditionally create a resource that depends on whether the stack is for a production or test environment.

Transform (optional) (p. 148)

For serverless applications (also referred to as Lambda-based applications), specifies the version of the AWS Serverless Application Model (AWS SAM) to use. When you specify a transform, you can use AWS SAM syntax to declare resources in your template. The model defines the syntax that you can use and how it is processed.

You can also use the `AWS::Include` transform to work with template snippets that are stored separately from the main AWS CloudFormation template. You store your snippet files in an Amazon S3 bucket and then reuse the functions across multiple templates.

Resources (required) (p. 153)

Specifies the stack resources and their properties, such as an Amazon Elastic Compute Cloud instance or an Amazon Simple Storage Service bucket. You can refer to resources in the `Resources` and `Outputs` sections of the template.

Outputs (optional) (p. 155)

Describes the values that are returned whenever you view your stack's properties. For example, you can declare an output for an S3 bucket name and then call the `aws cloudformation describe-stacks` AWS CLI command to view the name.

Format Version

The `AWSTemplateFormatVersion` section (optional) identifies the capabilities of the template. The latest template format version is `2010-09-09` and is currently the only valid value.

> **Note**
> The template format version is not the same as the API or WSDL version. The template format version can change independently of the API and WSDL versions.

The value for the template format version declaration must be a literal string. You cannot use a parameter or function to specify the template format version. If you don't specify a value, AWS CloudFormation assumes the latest template format version. The following snippet is an example of a valid template format version declaration:

JSON

```
"AWSTemplateFormatVersion" : "2010-09-09"
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
```

Description

The `Description` section (optional) enables you to include arbitrary comments about your template. The `Description` must follow the `AWSTemplateFormatVersion` section.

The value for the description declaration must be a literal string that is between 0 and 1024 bytes in length. You cannot use a parameter or function to specify the description. The following snippet is an example of a description declaration:

JSON

```
"Description" : "Here are some details about the template."
```

YAML

```
Description: >
  Here are some
  details about
  the template.
```

Metadata

You can use the optional `Metadata` section to include arbitrary JSON or YAML objects that provide details about the template. For example, you can include template implementation details about specific resources, as shown in the following snippet:

> **Important**
> During a stack update, you cannot update the `Metadata` section by itself. You can update it only when you include changes that add, modify, or delete resources.

JSON

```
"Metadata" : {
  "Instances" : {"Description" : "Information about the instances"},
  "Databases" : {"Description" : "Information about the databases"}
}
```

YAML

```
Metadata:
  Instances:
    Description: "Information about the instances"
  Databases:
    Description: "Information about the databases"
```

Metadata Keys

Some AWS CloudFormation features retrieve settings or configuration information that you define from the `Metadata` section. You define this information in the following AWS CloudFormation-specific metadata keys:

`AWS::CloudFormation::Init`

Defines configuration tasks for the cfn-init helper script. This script is useful for configuring and installing applications on EC2 instances. For more information, see AWS::CloudFormation::Init (p. 470).

`AWS::CloudFormation::Interface`

Defines the grouping and ordering of input parameters when they are displayed in the AWS CloudFormation console. By default, the AWS CloudFormation console alphabetically sorts parameters by their logical ID. For more information, see AWS::CloudFormation::Interface (p. 483).

`AWS::CloudFormation::Designer`

Describes how your resources are laid out in AWS CloudFormation Designer (Designer). Designer automatically adds this information when you use it create and update templates. For more information, see What Is AWS CloudFormation Designer? (p. 157).

Parameters

You can use the optional `Parameters` section to pass values into your template when you create a stack. With parameters, you can create templates that are customized each time you create a stack. Each parameter must contain a value when you create a stack. You can specify a default value to make the parameter optional so that you don't need to pass in a value when creating a stack. AWS CloudFormation will use the default value. For more information about creating stacks, see Working with Stacks (p. 71).

EC2 Instance Type Parameter

The following example creates a parameter for Amazon EC2 instance types. When you create a stack, you can specify the value for the `InstanceTypeParameter` parameter. That way, you can choose what instance type you want when you create a stack. By default, the template uses `t2.micro`.

JSON

```
"Parameters" : {
  "InstanceTypeParameter" : {
```

```
    "Type" : "String",
    "Default" : "t2.micro",
    "AllowedValues" : ["t2.micro", "m1.small", "m1.large"],
    "Description" : "Enter t2.micro, m1.small, or m1.large. Default is t2.micro."
  }
}
```

YAML

```
Parameters:
  InstanceTypeParameter:
    Type: String
    Default: t2.micro
    AllowedValues:
      - t2.micro
      - m1.small
      - m1.large
    Description: Enter t2.micro, m1.small, or m1.large. Default is t2.micro.
```

Using the EC2 Instance Type Parameter

Within the same template, you can use the `Ref` intrinsic function to specify the parameter value in other parts of the template. The following snippet uses the `InstanceTypeParameter` parameter to specify the instance type for an EC2 instance resource:

JSON

```
"Ec2Instance" : {
  "Type" : "AWS::EC2::Instance",
  "Properties" : {
    "InstanceType" : { "Ref" : "InstanceTypeParameter" },
    "ImageId" : "ami-2f726546"
  }
}
```

YAML

```
Ec2Instance:
  Type: AWS::EC2::Instance
  Properties:
    InstanceType:
      Ref: InstanceTypeParameter
    ImageId: ami-2f726546
```

Syntax

The `Parameters` section consists of the key name `Parameters`. You can have a maximum of 60 parameters in an AWS CloudFormation template.

For each parameter, you must declare a logical name, which must be alphanumeric and unique among all logical names within the template. After you declare the parameter's logical name, you can specify the parameter's properties. You must declare parameters as one of following types: `String`, `Number`, `CommaDelimitedList`, or an AWS-specific type. For `String`, `Number`, and AWS-specific parameter types, you can define constraints that AWS CloudFormation uses to validate the value of the parameter.

AWS-specific parameter types are AWS values such as Amazon EC2 key pair names and VPC IDs. AWS CloudFormation validates these parameter values against existing values in users' AWS accounts. AWS-specific parameter types are helpful in catching invalid values at the start of creating or updating a stack.

Important
For sensitive parameter values (such as passwords), set the `NoEcho` property to `true`. That way, whenever anyone describes your stack, the parameter value is shown as asterisks (`*****`).

JSON

```
"Parameters" : {
  "ParameterLogicalID" : {
    "Type" : "DataType",
    "ParameterProperty" : "value"
  }
}
```

YAML

```
Parameters:
  ParameterLogicalID:
    Type: DataType
    ParameterProperty: value
```

Properties

`AllowedPattern`

A regular expression that represents the patterns you want to allow for `String` types.

Required: No

`AllowedValues`

An array containing the list of values allowed for the parameter.

Required: No

`ConstraintDescription`

A string that explains the constraint when the constraint is violated. For example, without a constraint description, a parameter that has an allowed pattern of `[A-Za-z0-9]+` displays the following error message when the user specifies an invalid value:

```
Malformed input-Parameter MyParameter must match pattern [A-Za-z0-9]+
```

By adding a constraint description, such as `must only contain upper- and lowercase letters, and numbers`, you can display a customized error message:

```
Malformed input-Parameter MyParameter must only contain upper and lower case letters
 and numbers
```

Required: No

`Default`

A value of the appropriate type for the template to use if no value is specified when a stack is created. If you define constraints for the parameter, you must specify a value that adheres to those constraints.

Required: No

`Description`

A string of up to 4000 characters that describes the parameter.

Required: No

`MaxLength`

An integer value that determines the largest number of characters you want to allow for `String` types.

Required: No

`MaxValue`

A numeric value that determines the largest numeric value you want to allow for `Number` types.

Required: No

`MinLength`

An integer value that determines the smallest number of characters you want to allow for `String` types.

Required: No

`MinValue`

A numeric value that determines the smallest numeric value you want to allow for `Number` types.

Required: No

`NoEcho`

Whether to mask the parameter value whenever anyone makes a call that describes the stack. If you set the value to `true`, the parameter value is masked with asterisks (`*****`).

Required: No

`Type`

The data type for the parameter (`DataType`).

Required: Yes

You can specify the following values for the `Type` property:

`String`

A literal string.

For example, users could specify `"MyUserName"`.

`Number`

An integer or float. AWS CloudFormation validates the parameter value as a number; however, when you use the parameter elsewhere in your template (for example, by using the `Ref` intrinsic function), the parameter value becomes a string.

For example, users could specify `"8888"`.

`List<Number>`

An array of integers or floats that are separated by commas. AWS CloudFormation validates the parameter value as numbers; however, when you use the parameter elsewhere in your template (for example, by using the `Ref` intrinsic function), the parameter value becomes a list of strings.

For example, users could specify `"80,20"`, and a `Ref` will result in `["80","20"]`.

`CommaDelimitedList`

An array of literal strings that are separated by commas. The total number of strings should be one more than the total number of commas. Also, each member string is space trimmed.

For example, users could specify `"test,dev,prod"`, and a `Ref` will result in `["test","dev","prod"]`.

AWS-Specific Parameter Types

For AWS-specific parameter types, template users must specify existing AWS values that are in their account. AWS CloudFormation supports the following AWS-specific types:

AWS::EC2::AvailabilityZone::Name

An Availability Zone, such as us-west-2a.

AWS::EC2::Image::Id

An Amazon EC2 image ID, such as ami-ff527ecf. Note that the AWS CloudFormation console won't show a drop-down list of values for this parameter type.

AWS::EC2::Instance::Id

An Amazon EC2 instance ID, such as i-1e731a32.

AWS::EC2::KeyPair::KeyName

An Amazon EC2 key pair name.

AWS::EC2::SecurityGroup::GroupName

An EC2-Classic or default VPC security group name, such as my-sg-abc.

AWS::EC2::SecurityGroup::Id

A security group ID, such as sg-a123fd85.

AWS::EC2::Subnet::Id

A subnet ID, such as subnet-123a351e.

AWS::EC2::Volume::Id

An Amazon EBS volume ID, such as vol-3cdd3f56.

AWS::EC2::VPC::Id

A VPC ID, such as vpc-a123baa3.

AWS::Route53::HostedZone::Id

An Amazon Route 53 hosted zone ID, such as Z23YXV4OVPL04A.

List<AWS::EC2::AvailabilityZone::Name>

An array of Availability Zones for a region, such as us-west-2a, us-west-2b.

List<AWS::EC2::Image::Id>

An array of Amazon EC2 image IDs, such as ami-ff527ecf, ami-e7527ed7. Note that the AWS CloudFormation console won't show a drop-down list of values for this parameter type.

List<AWS::EC2::Instance::Id>

An array of Amazon EC2 instance IDs, such as i-1e731a32, i-1e731a34.

List<AWS::EC2::SecurityGroup::GroupName>

An array of EC2-Classic or default VPC security group names, such as my-sg-abc, my-sg-def.

List<AWS::EC2::SecurityGroup::Id>

An array of security group IDs, such as sg-a123fd85, sg-b456fd85.

List<AWS::EC2::Subnet::Id>

An array of subnet IDs, such as subnet-123a351e, subnet-456b351e.

`List<AWS::EC2::Volume::Id>`

An array of Amazon EBS volume IDs, such as `vol-3cdd3f56, vol-4cdd3f56`.

`List<AWS::EC2::VPC::Id>`

An array of VPC IDs, such as `vpc-a123baa3, vpc-b456baa3`.

`List<AWS::Route53::HostedZone::Id>`

An array of Amazon Route 53 hosted zone IDs, such as `Z23YXV4OVPL04A, Z23YXV4OVPL04B`.

AWS CloudFormation validates input values for these types against existing values in a user's account. For example, with the `AWS::EC2::VPC::Id` type, a user must enter an existing VPC ID (p. 75) that is in her account and in the region in which she is creating the stack.

Group and Sort Parameters in the AWS CloudFormation Console

When you use the AWS CloudFormation console to create or update a stack, the console alphabetically lists input parameters by their logical ID. To override the default ordering, you can use the `AWS::CloudFormation::Interface` metadata key. By grouping and ordering parameters, you make it easier for users to specify parameter values. For example, you could group all VPC-related parameters so that they aren't scattered throughout an alphabetical list.

In the metadata key, you can specify the groups to create, the parameters to include in each group, and the order in which the console shows each parameter within its group. You can also define friendly parameter names so that the console shows descriptive names instead of logical IDs. All parameters that you reference in the metadata key must be declared in the `Parameters` section of the template.

For more information and an example of the `AWS::CloudFormation::Interface` metadata key, see AWS::CloudFormation::Interface (p. 483).

Examples

Basic Input Parameters

The following example `Parameters` section declares two parameters. The `DBPort` parameter is of type `Number` with a default of `3306`. The minimum value that can be specified is `1150`, and the maximum value that can be specified is `65535`. The `DBPwd` parameter is of type `String` with no default value. The `NoEcho` property is set to `true` to prevent describe stack calls, such as the `aws cloudformation describe-stacks` AWS CLI command, from returning the parameter value. The minimum length that can be specified is `1`, and the maximum length that can be specified is `41`. The pattern allows lowercase and uppercase alphabetic characters and numerals.

JSON

```
"Parameters" : {
  "DBPort" : {
    "Default" : "3306",
    "Description" : "TCP/IP port for the database",
    "Type" : "Number",
    "MinValue" : "1150",
    "MaxValue" : "65535"
  },
  "DBPwd" : {
    "NoEcho" : "true",
    "Description" : "The database admin account password",
    "Type" : "String",
    "MinLength" : "1",
    "MaxLength" : "41",
    "AllowedPattern" : "[a-zA-Z0-9]*"
```

```
  }
}
```

YAML

```
Parameters:
  DBPort:
    Default: 3306
    Description: TCP/IP port for the database
    Type: Number
    MinValue: 1150
    MaxValue: 65535
  DBPwd:
    NoEcho: true
    Description: The database admin account password
    Type: String
    MinLength: 1
    MaxLength: 41
    AllowedPattern: "[a-zA-Z0-9]*"
```

AWS-Specific Parameter Types

When you use AWS-specific parameter types, anyone who uses your template to create or update a stack must specify existing AWS values that are in his account and in the region for the current stack. AWS-specific parameter types help ensure that input values for these types exist and are correct before AWS CloudFormation creates or updates any resources. For example, if you use the `AWS::EC2::KeyPair::KeyName` parameter type, AWS CloudFormation validates the input value against users' existing key pair names before it creates any resources, such as Amazon EC2 instances.

If a user uses the AWS Management Console, AWS CloudFormation prepopulates AWS-specific parameter types with valid values (p. 75). That way the user doesn't have to remember and correctly enter a specific name or ID. She just selects one or more values from a drop-down list. Also, depending on the parameter type, users can search for values by ID, name, or Name tag value. For more information, see Specifying Stack Name and Parameters (p. 75).

The following example declares two parameters with the types `AWS::EC2::KeyPair::KeyName` and `AWS::EC2::Subnet::Id`. These types limit valid values to existing key pair names and subnet IDs. Because the `mySubnetIDs` parameter is specified as a list, a user can specify one or more subnet IDs.

JSON

```
"Parameters" : {
  "myKeyPair" : {
    "Description" : "Amazon EC2 Key Pair",
    "Type" : "AWS::EC2::KeyPair::KeyName"
  },
  "mySubnetIDs" : {
    "Description" : "Subnet IDs",
    "Type" : "List<AWS::EC2::Subnet::Id>"
  }
}
```

YAML

```
Parameters:
  myKeyPair:
    Description: Amazon EC2 Key Pair
    Type: "AWS::EC2::KeyPair::KeyName"
  mySubnetIDs:
    Description: Subnet IDs
```

```
    Type: "List<AWS::EC2::Subnet::Id>"
```

AWS CLI and API Support

Currently, users can't use the AWS CLI or AWS CloudFormation API to view a list of valid values for AWS-specific parameters. However, they can view information about each parameter, such as the parameter type, by using the aws cloudformation get-template-summary command or `GetTemplateSummary` API.

Comma-delimited List Parameter Type

You can use the `CommaDelimitedList` parameter type to specify multiple string values in a single parameter. That way, you can use a single parameter instead of many different parameters to specify multiple values. For example, if you create three different subnets with their own CIDR blocks, you could use three different parameters to specify three different CIDR blocks. But it's simpler just to use a single parameter that takes a list of three CIDR blocks, as shown in the following snippet:

JSON

```
"Parameters" : {
  "DbSubnetIpBlocks": {
    "Description": "Comma-delimited list of three CIDR blocks",
    "Type": "CommaDelimitedList",
    "Default": "10.0.48.0/24, 10.0.112.0/24, 10.0.176.0/24"
  }
}
```

YAML

```
Parameters:
  DbSubnetIpBlocks:
    Description: "Comma-delimited list of three CIDR blocks"
    Type: CommaDelimitedList
    Default: "10.0.48.0/24, 10.0.112.0/24, 10.0.176.0/24"
```

Return a Value from a Comma-delimited List Parameter

To refer to a specific value in a list, use the `Fn::Select` intrinsic function in the `Resources` section of your template. You pass the index value of the object that you want and a list of objects, as shown in the following snippet:

JSON

```
"DbSubnet1" : {
  "Type" : "AWS::EC2::Subnet",
  "Properties" : {
    "AvailabilityZone" : {"Fn::Join" : ["",[ { "Ref" : "AWS::Region" }, { "Fn::Select" :
 [ "0", {"Ref" : "VpcAzs"} ] } ] ]} ,
    "VpcId" :  { "Ref" : "VPC" },
    "CidrBlock" : { "Fn::Select" : [ "0", {"Ref" : "DbSubnetIpBlocks"} ] }
  }
},
"DbSubnet2" : {
  "Type" : "AWS::EC2::Subnet",
  "Properties" : {
    "AvailabilityZone" : {"Fn::Join" : ["",[ { "Ref" : "AWS::Region" }, { "Fn::Select" :
 [ "1", {"Ref" : "VpcAzs"} ] } ] ]} ,
    "VpcId" : { "Ref" : "VPC" },
    "CidrBlock" : { "Fn::Select" : [ "1", {"Ref" : "DbSubnetIpBlocks"} ] }
  }
},
```

```
"DbSubnet3" : {
  "Type" : "AWS::EC2::Subnet",
  "Properties" : {
    "AvailabilityZone" : {"Fn::Join" : ["",[ { "Ref" : "AWS::Region" }, { "Fn::Select" :
 [ "2", {"Ref" : "VpcAzs"} ] } ] ]} ,
    "VpcId" : { "Ref" : "VPC" },
    "CidrBlock" : { "Fn::Select" : [ "2", {"Ref" : "DbSubnetIpBlocks"} ] }
  }
}
```

YAML

```
DbSubnet1:
  Type: AWS::EC2::Subnet
  Properties:
    AvailabilityZone: !Sub
      - "${AWS::Region}${AZ}"
      - AZ: !Select [0, !Ref VpcAzs]
    VpcId: !Ref VPC
    CidrBlock: !Select [0, !Ref DbSubnetIpBlocks]
DbSubnet2:
  Type: AWS::EC2::Subnet
  Properties:
    AvailabilityZone: !Sub
      - "${AWS::Region}${AZ}"
      - AZ: !Select [1, !Ref VpcAzs]
    VpcId: !Ref VPC
    CidrBlock: !Select [1, !Ref DbSubnetIpBlocks]
DbSubnet3:
  Type: AWS::EC2::Subnet
  Properties:
    AvailabilityZone: !Sub
      - "${AWS::Region}${AZ}"
      - AZ: !Select [2, !Ref VpcAzs]
    VpcId: !Ref VPC
    CidrBlock: !Select [2, !Ref DbSubnetIpBlocks]
```

Mappings

The optional `Mappings` section matches a key to a corresponding set of named values. For example, if you want to set values based on a region, you can create a mapping that uses the region name as a key and contains the values you want to specify for each specific region. You use the `Fn::FindInMap` intrinsic function to retrieve values in a map.

You cannot include parameters, pseudo parameters, or intrinsic functions in the `Mappings` section.

Syntax

The `Mappings` section consists of the key name `Mappings`. The keys and values in mappings must be literal strings. The following example shows a `Mappings` section containing a single mapping named `Mapping01` (the logical name).

Within a mapping, each map is a key followed by another mapping. The key identifies a map of name-value pairs and must be unique within the mapping. The name can contain only alphanumeric characters (A-Za-z0-9).

JSON

```
"Mappings" : {
  "Mapping01" : {
```

```
    "Key01" : {
      "Name" : "Value01"
    },
    "Key02" : {
      "Name" : "Value02"
    },
    "Key03" : {
      "Name" : "Value03"
    }
  }
}
```

YAML

```
Mappings:
  Mapping01:
    Key01:
      Name: Value01
    Key02:
      Name: Value02
    Key03:
      Name: Value03
```

Examples

Basic Mapping

The following example shows a `Mappings` section with a map `RegionMap`, which contains five keys that map to name-value pairs containing single string values. The keys are region names. Each name-value pair is the AMI ID for the 32-bit AMI in the region represented by the key.

The name-value pairs have a name (32 in the example) and a value. By naming the values, you can map more than one set of values to a key.

JSON

```
"Mappings" : {
  "RegionMap" : {
    "us-east-1"      : { "32" : "ami-6411e20d"},
    "us-west-1"      : { "32" : "ami-c9c7978c"},
    "eu-west-1"      : { "32" : "ami-37c2f643"},
    "ap-southeast-1" : { "32" : "ami-66f28c34"},
    "ap-northeast-1" : { "32" : "ami-9c03a89d"}
  }
}
```

YAML

```
Mappings:
  RegionMap:
    us-east-1:
      "32": "ami-6411e20d"
    us-west-1:
      "32": "ami-c9c7978c"
    eu-west-1:
      "32": "ami-37c2f643"
    ap-southeast-1:
      "32": "ami-66f28c34"
    ap-northeast-1:
      "32": "ami-9c03a89d"
```

Mapping with Multiple Values

The following example has region keys that are mapped to two sets of values: one named 32 and the other 64.

JSON

```
"RegionMap" : {
  "us-east-1"      : { "32" : "ami-6411e20d", "64" : "ami-7a11e213" },
  "us-west-1"      : { "32" : "ami-c9c7978c", "64" : "ami-cfc7978a" },
  "eu-west-1"      : { "32" : "ami-37c2f643", "64" : "ami-31c2f645" },
  "ap-southeast-1" : { "32" : "ami-66f28c34", "64" : "ami-60f28c32" },
  "ap-northeast-1" : { "32" : "ami-9c03a89d", "64" : "ami-a003a8a1" }
}
```

YAML

```
RegionMap:
  us-east-1:
    "32": "ami-6411e20d"
    "64": "ami-7a11e213"
  us-west-1:
    "32": "ami-c9c7978c"
    "64": "ami-cfc7978a"
  eu-west-1:
    "32": "ami-37c2f643"
    "64": "ami-31c2f645"
  ap-southeast-1:
    "32": "ami-66f28c34"
    "64": "ami-60f28c32"
  ap-northeast-1:
    "32": "ami-9c03a89d"
    "64": "ami-a003a8a1"
```

Return a Value from a Mapping

You can use the `Fn::FindInMap` (p. 1322) function to return a named value based on a specified key. The following example template contains an Amazon EC2 resource whose `ImageId` property is assigned by the `FindInMap` function. The `FindInMap` function specifies key as the region where the stack is created (using the AWS::Region pseudo parameter (p. 1351)) and `32` as the name of the value to map to.

JSON

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",

  "Mappings" : {
    "RegionMap" : {
      "us-east-1" : { "32" : "ami-6411e20d", "64" : "ami-7a11e213" },
      "us-west-1" : { "32" : "ami-c9c7978c", "64" : "ami-cfc7978a" },
      "eu-west-1" : { "32" : "ami-37c2f643", "64" : "ami-31c2f645" },
      "ap-southeast-1" : { "32" : "ami-66f28c34", "64" : "ami-60f28c32" },
      "ap-northeast-1" : { "32" : "ami-9c03a89d", "64" : "ami-a003a8a1" }
    }
  },

  "Resources" : {
    "myEC2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "ImageId" : { "Fn::FindInMap" : [ "RegionMap", { "Ref" : "AWS::Region" }, "32"]},
```

```
        "InstanceType" : "m1.small"
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Mappings:
  RegionMap:
    us-east-1:
      "32": "ami-6411e20d"
      "64": "ami-7a11e213"
    us-west-1:
      "32": "ami-c9c7978c"
      "64": "ami-cfc7978a"
    eu-west-1:
      "32": "ami-37c2f643"
      "64": "ami-31c2f645"
    ap-southeast-1:
      "32": "ami-66f28c34"
      "64": "ami-60f28c32"
    ap-northeast-1:
      "32": "ami-9c03a89d"
      "64": "ami-a003a8a1"
Resources:
  myEC2Instance:
    Type: "AWS::EC2::Instance"
    Properties:
      ImageId: !FindInMap [RegionMap, !Ref "AWS::Region", 32]
      InstanceType: m1.small
```

Input Parameter and FindInMap

You can use an input parameter with the `Fn::FindInMap` function to refer to a specific value in a map. For example, suppose you have a list of regions and environment types that map to a specific AMI ID. You can select the AMI ID that your stack uses by using an input parameter (`EnvironmentType`). To determine the region, use the `AWS::Region` pseudo parameter, which gets the AWS region in which you create the stack.

JSON

```
{
  "Parameters" : {
    "EnvironmentType": {
      "Description": "The environment type",
      "Type": "String",
      "Default": "test",
      "AllowedValues": ["prod", "test"],
      "ConstraintDescription": "must be a prod or test"
    }
  },

  "Mappings" : {
    "RegionAndInstanceTypeToAMIID" : {
      "us-east-1": {
        "test": "ami-8ff710e2",
        "prod": "ami-f5f41398"
      },
      "us-west-2" : {
        "test" : "ami-eff1028f",
        "prod" : "ami-d0f506b0"
      },
```

```
      ...other regions and AMI IDs...

    }
  },

  "Resources" : {

   ...other resources...

    },

  "Outputs" : {
    "TestOutput" : {
      "Description" : "Return the name of the AMI ID that matches the region and
 environment type keys",
      "Value" : { "Fn::FindInMap" : [ "RegionAndInstanceTypeToAMIID", { "Ref" :
 "AWS::Region" }, { "Ref" : "EnvironmentType" } ]}
    }
  }

}
```

YAML

```
  Parameters:
    EnvironmentType:
      Description: The environment type
      Type: String
      Default: test
      AllowedValues:
        - prod
        - test
      ConstraintDescription: must be a prod or test
  Mappings:
    RegionAndInstanceTypeToAMIID:
      us-east-1:
        test: "ami-8ff710e2"
        prod: "ami-f5f41398"
      us-west-2:
        test: "ami-eff1028f"
        prod: "ami-d0f506b0"

      ...other regions and AMI IDs...

  Resources:

   ...other resources...

  Outputs:
    TestOutput:
      Description: Return the name of the AMI ID that matches the region and environment
 type keys
      Value: !FindInMap [RegionAndInstanceTypeToAMIID, !Ref "AWS::Region", !Ref
 EnvironmentType]
```

Conditions

The optional `Conditions` section includes statements that define when a resource is created or when a property is defined. For example, you can compare whether a value is equal to another value. Based on the result of that condition, you can conditionally create resources. If you have multiple conditions, separate them with commas.

You might use conditions when you want to reuse a template that can create resources in different contexts, such as a test environment versus a production environment. In your template, you can add an `EnvironmentType` input parameter, which accepts either **`prod`** or **`test`** as inputs. For the production environment, you might include Amazon EC2 instances with certain capabilities; however, for the test environment, you want to use reduced capabilities to save money. With conditions, you can define which resources are created and how they're configured for each environment type.

Conditions are evaluated based on input parameter values that you specify when you create or update a stack. Within each condition, you can reference another condition, a parameter value, or a mapping. After you define all your conditions, you can associate them with resources and resource properties in the `Resources` and `Outputs` sections of a template.

At stack creation or stack update, AWS CloudFormation evaluates all the conditions in your template before creating any resources. Any resources that are associated with a true condition are created. Any resources that are associated with a false condition are ignored.

> **Important**
> During a stack update, you cannot update conditions by themselves. You can update conditions only when you include changes that add, modify, or delete resources.

How to Use Conditions Overview

To conditionally create resources, you must include statements in at least three different sections of a template:

`Parameters` section

Define the input values that you want to evaluate in your conditions. Conditions will result in true or false based on values from these input parameter.

`Conditions` section

Define conditions by using the intrinsic condition functions. These conditions determine when AWS CloudFormation creates the associated resources.

`Resources` and `Outputs` sections

Associate conditions with the resources or outputs that you want to conditionally create. AWS CloudFormation creates entities that are associated with a true condition and ignores entities that are associated with a false condition. Use the `Condition` key and a condition's logical ID to associate it with a resource or output. To conditionally specify a property, use the `Fn::If` function. For more information, see Condition Functions (p. 1312).

Syntax

The `Conditions` section consists of the key name `Conditions`. Each condition declaration includes a logical ID and intrinsic functions that are evaluated when you create or update a stack. The following pseudo template outlines the `Conditions` section:

JSON

```
"Conditions" : {
  "Logical ID" : {Intrinsic function}
}
```

YAML

```
Conditions:
```

```
Logical ID:
  Intrinsic function
```

Condition Intrinsic Functions

You can use the following intrinsic functions to define conditions:

- `Fn::And`
- `Fn::Equals`
- `Fn::If`
- `Fn::Not`
- `Fn::Or`

For the syntax and information about each function, see Condition Functions (p. 1312).

Examples

The following sample template includes an `EnvType` input parameter, where you can specify `prod` to create a stack for production or `test` to create a stack for testing. For a production environment, AWS CloudFormation creates an Amazon EC2 instance and attaches a volume to the instance. For a test environment, AWS CloudFormation creates only the Amazon EC2 instance.

The `CreateProdResources` condition evaluates to `true` if the `EnvType` parameter is equal to `prod`. In the sample template, the `NewVolume` and `MountPoint` resources are associated with the `CreateProdResources` condition. Therefore, the resources are created only if the `EnvType` parameter is equal to `prod`.

JSON

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",

  "Mappings" : {
    "RegionMap" : {
      "us-east-1"      : { "AMI" : "ami-7f418316", "TestAz" : "us-east-1a" },
      "us-west-1"      : { "AMI" : "ami-951945d0", "TestAz" : "us-west-1a" },
      "us-west-2"      : { "AMI" : "ami-16fd7026", "TestAz" : "us-west-2a" },
      "eu-west-1"      : { "AMI" : "ami-24506250", "TestAz" : "eu-west-1a" },
      "sa-east-1"      : { "AMI" : "ami-3e3be423", "TestAz" : "sa-east-1a" },
      "ap-southeast-1" : { "AMI" : "ami-74dda626", "TestAz" : "ap-southeast-1a" },
      "ap-southeast-2" : { "AMI" : "ami-b3990e89", "TestAz" : "ap-southeast-2a" },
      "ap-northeast-1" : { "AMI" : "ami-dcfa4edd", "TestAz" : "ap-northeast-1a" }
    }
  },

  "Parameters" : {
    "EnvType" : {
      "Description" : "Environment type.",
      "Default" : "test",
      "Type" : "String",
      "AllowedValues" : ["prod", "test"],
      "ConstraintDescription" : "must specify prod or test."
    }
  },

  "Conditions" : {
    "CreateProdResources" : {"Fn::Equals" : [{"Ref" : "EnvType"}, "prod"]}
  },
```

```
  "Resources" : {
    "EC2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "ImageId" : { "Fn::FindInMap" : [ "RegionMap", { "Ref" : "AWS::Region" }, "AMI" ]}
      }
    },

    "MountPoint" : {
      "Type" : "AWS::EC2::VolumeAttachment",
      "Condition" : "CreateProdResources",
      "Properties" : {
        "InstanceId" : { "Ref" : "EC2Instance" },
        "VolumeId"  : { "Ref" : "NewVolume" },
        "Device" : "/dev/sdh"
      }
    },

    "NewVolume" : {
      "Type" : "AWS::EC2::Volume",
      "Condition" : "CreateProdResources",
      "Properties" : {
        "Size" : "100",
        "AvailabilityZone" : { "Fn::GetAtt" : [ "EC2Instance", "AvailabilityZone" ]}
      }
    }
  },

  "Outputs" : {
    "VolumeId" : {
      "Value" : { "Ref" : "NewVolume" },
      "Condition" : "CreateProdResources"
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Mappings:
  RegionMap:
    us-east-1:
      AMI: "ami-7f418316"
      TestAz: "us-east-1a"
    us-west-1:
      AMI: "ami-951945d0"
      TestAz: "us-west-1a"
    us-west-2:
      AMI: "ami-16fd7026"
      TestAz: "us-west-2a"
    eu-west-1:
      AMI: "ami-24506250"
      TestAz: "eu-west-1a"
    sa-east-1:
      AMI: "ami-3e3be423"
      TestAz: "sa-east-1a"
    ap-southeast-1:
      AMI: "ami-74dda626"
      TestAz: "ap-southeast-1a"
    ap-southeast-2:
      AMI: "ami-b3990e89"
      TestAz: "ap-southeast-2a"
    ap-northeast-1:
      AMI: "ami-dcfa4edd"
```

```
      TestAz: "ap-northeast-1a"
Parameters:
  EnvType:
    Description: Environment type.
    Default: test
    Type: String
    AllowedValues:
      - prod
      - test
    ConstraintDescription: must specify prod or test.
Conditions:
  CreateProdResources: !Equals [ !Ref EnvType, prod ]
Resources:
  EC2Instance:
    Type: "AWS::EC2::Instance"
    Properties:
      ImageId: !FindInMap [RegionMap, !Ref "AWS::Region", AMI]
  MountPoint:
    Type: "AWS::EC2::VolumeAttachment"
    Condition: CreateProdResources
    Properties:
      InstanceId:
        !Ref EC2Instance
      VolumeId:
        !Ref NewVolume
      Device: /dev/sdh
  NewVolume:
    Type: "AWS::EC2::Volume"
    Condition: CreateProdResources
    Properties:
      Size: 100
      AvailabilityZone:
        !GetAtt EC2Instance.AvailabilityZone
Outputs:
  VolumeId:
    Condition: CreateProdResources
    Value:
      !Ref NewVolume
```

Transform

The `Transform` section is optional and specifies a transform that AWS CloudFormation uses to process your template. AWS CloudFormation transforms are statements that condense and simplify template authoring. For example, you can simplify a resource declaration that might have taken multiple lines into a single line.

Transforms are declarative statements within AWS CloudFormation templates that tell AWS CloudFormation how to process your template. `Transform` builds on the simple, declarative language of AWS CloudFormation with a powerful macro system. You can create AWS CloudFormation transformations to simplify the expression of AWS infrastructure as code. In doing so, transforms help simplify how you write templates.

<title>AWS::Serverless</title>

An `AWS::Serverless` transform is a specific version of the AWS Serverless Application Model (AWS SAM). This model defines the AWS SAM syntax that you can use and how AWS CloudFormation processes it. When you create a change set, AWS CloudFormation resolves all `Transform` functions.

For more information about serverless applications and the model, see Deploying Lambda-based Applications in the *AWS Lambda Developer Guide*.

<title>AWS::Include</title>

You can also use the `AWS::Include` transform to work with template snippets that are stored separately from the main AWS CloudFormation template. You can insert those snippets into your main template when Creating a Change Set (p. 94) or Updating Stacks Using Change Sets (p. 93).

Topics

AWS::Include Transform

You can use the `AWS::Include` transform to work with template snippets that are stored separately from the main AWS CloudFormation template. When you specify `Name: 'AWS::Include'` and the `Location` parameter, the `Transform` key is a placeholder where snippets are injected. AWS CloudFormation inserts those snippets into your main template when Creating a Change Set (p. 94) or Updating Stacks Using Change Sets (p. 93).

You might have a Lambda function that you want to reuse in one or more AWS CloudFormation templates. The `AWS::Include` transform lets you create a reference to a transform snippet in an Amazon S3 bucket. You can add `AWS::Include` to the `Transform` function in your AWS CloudFormation template. The `AWS::Include` function behaves similarly to an `include`, `copy`, or `import` directive in programming languages.

Usage

You can use the `AWS::Include` transform anywhere within the AWS CloudFormation template except in the template parameters section or the template version field. For example, you can use `AWS::Include` in the mappings section.

Syntax at the Top Level of a Template

To include a transform at the top level of a template, use the following syntax.

JSON

```
{
   "Transform" : {
       "Name" : "AWS::Include",
       "Parameters" : {
           "Location" : "s3://MyAmazonS3BucketName/MyFileName.json"
        }
    }
}
```

YAML

```
Transform:
  Name: 'AWS::Include'
  Parameters:
    Location: 's3://MyAmazonS3BucketName/MyFileName.yaml'
```

Syntax When the Transform Is Embedded Within a Section of a Template

To include a transform that is embedded within a section, use the following syntax.

JSON

```
{
```

```
    "Fn::Transform" : {
        "Name" : "AWS::Include",
        "Parameters" : {
            "Location" : "s3://MyAmazonS3BucketName/MyFileName.json"
         }
    }
}
```

YAML

```
'Fn::Transform':
  Name: 'AWS::Include'
  Parameters:
    Location: s3://MyAmazonS3BucketName/MyFileName.yaml
```

Parameters

Location

The location is an Amazon S3 URI, with a specific file name in an S3 bucket. For example, `s3://MyBucketName/MyFile.yaml`.

Remarks

When using `AWS::Include`, keep the following in mind:

- `AWS::Include` is supported only in regions where AWS Lambda is available. For a list of regions where Lambda is available, see http://docs.aws.amazon.com/general/latest/gr/rande.html#lambda_region.
- We currently support Amazon S3 URI, but no other Amazon S3 format (such as Amazon S3 ARN). It must be an Amazon S3 bucket, as opposed to something like a GitHub repository.
- Anyone with access to the Amazon S3 URL can include the snippet in their template.
- Your template snippet must be valid YAML or valid JSON.
- A template snippet must pass validation checks for a create stack or update stack operation.
- AWS CloudFormation resolves transforms first, and then processes the template. The resulting template must be valid JSON or YAML and must not exceed the template size limit.
- If your snippets change, your stack doesn't automatically pick up those changes. To get those changes, you must update the stack with the updated snippets. If you update your stack, make sure your included snippets haven't changed without your knowledge. To verify before updating the stack, check the change set.
- When using the update rollback feature, AWS CloudFormation uses a copy of the original template. It will roll back to the original template even if the included snippet was changed.
- Nested transforms do not work because we do not process transforms iteratively.
- When creating templates and snippets, you can mix YAML and JSON template languages.
- We do not currently support using shorthand notations for YAML snippets.
- You can use multiple transforms within a single template. Nevertheless, you cannot simultaneously have `AWS::Include` transforms at both the top level of a template and embedded within a section of a template.
- You can provide a cross-region replication Amazon S3 URI with `AWS::Include`. Be sure to check S3 bucket names when accessing cross-region replication objects. For more information, see Cross-Region Replication.

Example

The following example shows how to use the `AWS::Include` transform to execute a wait condition handle.

Both the JSON and the YAML versions use the following wait condition snippet. Save the file as `single_wait_condition.yaml`, and store it in an S3 bucket with the same name as *MyAmazonS3BucketName*.

```
WebServerWaitHandle:
  Type: 'AWS::CloudFormation::WaitConditionHandle'
```

JSON

```
{
   "Resources": {
      "MyWaitHandle": {
         "Type": "AWS::CloudFormation::WaitConditionHandle"
      },
      "Fn::Transform": {
         "Name": "AWS::Include",
         "Parameters": {
            "Location": "s3://MyAmazonS3BucketName/single_wait_condition.yaml"
         }
      }
   }
}
```

YAML

```
Resources:
  MyWaitHandle:
    Type: 'AWS::CloudFormation::WaitConditionHandle'
  'Fn::Transform':
    Name: 'AWS::Include'
    Parameters:
      Location : "s3://MyAmazonS3BucketName/single_wait_condition.yaml"
```

AWS::Serverless Transform

Use a transform to simplify template authoring for serverless applications. For example, the following template uses AWS SAM syntax to simplify the declaration of a Lambda function and its execution role.

```
Transform: AWS::Serverless-2016-10-31
Resources:
  MyServerlessFunctionLogicalID:
    Type: AWS::Serverless::Function
    Properties:
      Handler: index.handler
      Runtime: nodejs4.3
      CodeUri: 's3://testBucket/mySourceCode.zip'
```

When the template is submitted, AWS CloudFormation expands the AWS SAM syntax, as defined by the transform. The processed template expands the `AWS::Serverless::Function` resource, declaring an Lambda function and an execution role.

```
{
  "Resources": {
    "MyServerlessFunctionLogicalID": {
      "Type": "AWS::Lambda::Function",
      "Properties": {
        "Handler": "index.handler",
        "Code": {
          "S3Bucket": "testBucket",
```

```
          "S3Key": "mySourceCode.zip"
        },
        "Role": {
          "Fn::GetAtt": ["FunctionNameRole", "Arn"]
        },
        "Runtime": "nodejs4.3"
      }
    },
    "FunctionNameRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "ManagedPolicyArns": ["arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole"],
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [{
            "Action": ["sts:AssumeRole"],
            "Effect": "Allow",
            "Principal": {
              "Service": ["lambda.amazonaws.com"]
            }
          }]
        }
      }
    }
  }
}
```

AWS CloudFormation uses the processed template to create or update a stack. If you don't specify a transform value, AWS CloudFormation doesn't process your template, and the AWS SAM syntax fails template validation.

Syntax

The value for the transform declaration must be a literal string. You cannot use a parameter or function to specify a transform value. The following snippet is an example of a transform declaration:

JSON

```
"Transform" : "AWS::Serverless-2016-10-31"
```

YAML

```
Transform: "AWS::Serverless-2016-10-31"
```

Template Stage

The stage of a template indicates whether the template is the original user-submitted template or a one where AWS CloudFormation has processed the transforms. The original template is the one that users submitted to create or update the stack. The processed template is the template AWS CloudFormation used to create or update the stack after processing the transform(s). Use the processed template for troubleshooting stack issues. If a stack doesn't include transforms, the original and processed templates are identical.

You can use the AWS CloudFormation console (p. 78) or AWS CLI (p. 86) to see the stage of a stack's template.

Working with Stacks That Contain Transforms

To create or update a stack with transforms, you must create a change set (p. 72), and then execute it. A change set describes the actions AWS CloudFormation will take based on the processed template. During processing, AWS CloudFormation translates AWS SAM syntax into syntax that is defined by the transform. Processing can add multiples resources that you might not be aware of. For example, the specialized `AWS::Serverless::Function` resource adds an AWS Identity and Access Management (IAM) execution role and a Lambda function.

To ensure that you're aware of all of the changes introduced by transforms, AWS CloudFormation requires you to use change sets. After you review the change set, execute it to apply the changes or create another one.

> **Note**
> A transform can add IAM resources to your template. For these resources, AWS CloudFormation requires you to acknowledge their capabilities (p. 11). Because AWS CloudFormation can't know which resources are added before processing your template, you might need to acknowledge IAM capabilities when you create the change set, depending on whether your transforms contain IAM resources. That way, when you execute the change set, AWS CloudFormation has the necessary capabilities when creating IAM resources.

If you use the AWS CLI, you can use the `package` and `deploy` commands to reduce the number of steps for launching stacks with transforms. For more information, see Deploying Lambda-based Applications in the *AWS Lambda Developer Guide*.

Resources

The required `Resources` section declares the AWS resources that you want to include in the stack, such as an Amazon EC2 instance or an Amazon S3 bucket. You must declare each resource separately; however, if you have multiple resources of the same type, you can declare them together by separating them with commas.

Syntax

The `Resources` section consists of the key name `Resources`. The following pseudo template outlines the `Resources` section:

JSON

```
"Resources" : {
    "Logical ID" : {
        "Type" : "Resource type",
        "Properties" : {
            Set of properties
        }
    }
}
```

YAML

```
Resources:
  Logical ID:
    Type: Resource type
    Properties:
      Set of properties
```

Resource Fields

Logical ID

The logical ID must be alphanumeric (A-Za-z0-9) and unique within the template. Use the logical name to reference the resource in other parts of the template. For example, if you want to map an Amazon

Elastic Block Store volume to an Amazon EC2 instance, you reference the logical IDs to associate the block stores with the instance.

In addition to the logical ID, certain resources also have a physical ID, which is the actual assigned name for that resource, such as an EC2 instance ID or an S3 bucket name. Use the physical IDs to identify resources outside of AWS CloudFormation templates, but only after the resources have been created. For example, you might give an EC2 instance resource a logical ID of `MyEC2Instance`; but when AWS CloudFormation creates the instance, AWS CloudFormation automatically generates and assigns a physical ID (such as `i-28f9ba55`) to the instance. You can use this physical ID to identify the instance and view its properties (such as the DNS name) by using the Amazon EC2 console. For resources that support custom names, you can assign your own names (physical IDs) to help you quickly identify resources. For example, you can name an S3 bucket that stores logs as `MyPerformanceLogs`. For more information, see Name Type (p. 1217).

Resource type

The resource type identifies the type of resource that you are declaring. For example, `AWS::EC2::Instance` declares an EC2 instance. For a list of all of the resource types, see AWS Resource Types Reference (p. 390).

Resource properties

Resource properties are additional options that you can specify for a resource. For example, for each EC2 instance, you must specify an Amazon Machine Image (AMI) ID for that instance. You declare the AMI ID as a property of the instance, as shown in the following example:

If a resource doesn't require that properties be declared, omit the properties section of that resource.

Property values can be literal strings, lists of strings, Booleans, parameter references, pseudo references, or the value returned by a function. The following example shows you how to declare different property value types:

You can conditionally create a resource by associating a condition with it. You must define the condition in the `Conditions (p. 144)` section of the template.

Examples

The following example shows a resource declaration. It defines two resources. The `MyInstance` resource includes the `MyQueue` resource as part of its `UserData` property:

JSON

```
"Resources" : {
  "MyInstance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
          "UserData" : {
              "Fn::Base64" : {
                  "Fn::Join" : [ "", [ "Queue=", { "Ref" : "MyQueue" } ] ]
               } },
          "AvailabilityZone" : "us-east-1a",
          "ImageId" : "ami-20b65349"
      }
  },

  "MyQueue" : {
      "Type" : "AWS::SQS::Queue",
      "Properties" : {
      }
  }
```

```
}
```

YAML

```
Resources:
  MyInstance:
    Type: "AWS::EC2::Instance"
    Properties:
      UserData:
        "Fn::Base64":
          !Sub |
            Queue=${MyQueue}
      AvailabilityZone: "us-east-1a"
      ImageId: "ami-20b65349"
  MyQueue:
    Type: "AWS::SQS::Queue"
    Properties: {}
```

Outputs

The optional `Outputs` section declares output values that you can import into other stacks (p. 1334) (to create cross-stack references (p. 200)), return in response (to describe stack calls), or view on the AWS CloudFormation console (p. 78). For example, you can output the S3 bucket name for a stack to make the bucket easier to find.

Syntax

The `Outputs` section consists of the key name `Outputs`, followed by a space and a single colon. You can declare a maximum of 60 outputs in a template.

The following example demonstrates the structure of the `Outputs` section.

JSON

Use braces to enclose all output declarations. Delimit multiple outputs with commas.

```
"Outputs" : {
  "Logical ID" : {
    "Description" : "Information about the value",
    "Value" : "Value to return",
    "Export" : {
      "Name" : "Value to export"
    }
  }
}
```

YAML

```
Outputs:
  Logical ID:
    Description: Information about the value
    Value: Value to return
    Export:
      Name: Value to export
```

Output Fields

The `Outputs` section can include the following fields.

Logical ID

An identifier for the current output. The logical ID must be alphanumeric (`a-z`, `A-Z`, `0-9`) and unique within the template.

Description (optional)

A `String` type that describes the output value. The description can be a maximum of 4 K in length.

Value (required)

The value of the property returned by the `aws cloudformation describe-stacks` command. The value of an output can include literals, parameter references, pseudo-parameters, a mapping value, or intrinsic functions.

Export (optional)

The name of the resource output to be exported for a cross-stack reference (p. 200).

> **Note**
>
> The following restrictions apply to cross-stack references:
>
> - For each AWS account, `Export` names must be unique within a region.
> - You can't create cross-stack references across regions. You can use the intrinsic function `Fn::ImportValue` to import only values that have been exported within the same region.
> - For outputs, the value of the `Name` property of an `Export` can't use `Ref` or `GetAtt` functions that depend on a resource.
>
>   Similarly, the `ImportValue` function can't include `Ref` or `GetAtt` functions that depend on a resource.
> - You can't delete a stack if another stack references one of its outputs.
> - You can't modify or remove an output value that is referenced by another stack.

You can use intrinsic functions to customize the `Name` value of an export. The following examples use the `Fn::Join` function.

JSON

```
"Export" : {
  "Name" : {
    "Fn::Join" : [ ":", [ { "Ref" : "AWS::StackName" }, "AccountVPC" ] ]
  }
}
```

YAML

```
Export:
  Name: !Join [ ":", [ !Ref "AWS::StackName", AccountVPC ] ]
```

To associate a condition with an output, define the condition in the `Conditions` `(p. 144)` section of the template.

Examples

The following examples illustrate how stack output works.

Stack Output

In the following example, the output named `BackupLoadBalancerDNSName` returns the DNS name for the resource with the logical ID `BackupLoadBalancer` only when the `CreateProdResources` condition is true. (The second output shows how to specify multiple outputs.)

JSON

```
"Outputs" : {
  "BackupLoadBalancerDNSName" : {
    "Description": "The DNSName of the backup load balancer",
    "Value" : { "Fn::GetAtt" : [ "BackupLoadBalancer", "DNSName" ]},
    "Condition" : "CreateProdResources"
  },
  "InstanceID" : {
    "Description": "The Instance ID",
    "Value" : { "Ref" : "EC2Instance" }
  }
}
```

YAML

```
Outputs:
  BackupLoadBalancerDNSName:
    Description: The DNSName of the backup load balancer
    Value: !GetAtt BackupLoadBalancer.DNSName
    Condition: CreateProdResources
  InstanceID:
    Description: The Instance ID
    Value: !Ref EC2Instance
```

Cross-Stack Output

In the following examples, the output named `StackVPC` returns the ID of a VPC, and then exports the value for cross-stack referencing with the name `VPCID` appended to the stack's name.

JSON

```
"Outputs" : {
  "StackVPC" : {
    "Description" : "The ID of the VPC",
    "Value" : { "Ref" : "MyVPC" },
    "Export" : {
      "Name" : {"Fn::Sub": "${AWS::StackName}-VPCID" }
    }
  }
}
```

YAML

```
Outputs:
  StackVPC:
    Description: The ID of the VPC
    Value: !Ref MyVPC
    Export:
      Name: !Sub "${AWS::StackName}-VPCID"
```

What Is AWS CloudFormation Designer?

AWS CloudFormation Designer (Designer) is a graphic tool for creating, viewing, and modifying AWS CloudFormation templates. With Designer, you can diagram your template resources using a drag-and-drop interface, and then edit their details using the integrated JSON and YAML editor. Whether you are a new or an experienced AWS CloudFormation user, AWS CloudFormation Designer can help you quickly see the interrelationship between a template's resources and easily modify templates.

Designer is part of the AWS CloudFormation console. To use it, open Designer at https://console.aws.amazon.com/cloudformation/designer and sign in with your AWS credentials.

Topics

Why Use AWS CloudFormation Designer?

AWS CloudFormation Designer (Designer) provides the following benefits: it allows you to see graphic representations of the resources in your template, it simplifies template authoring, and it simplifies template editing.

Visualize Template Resources

Parsing JSON- or YAML-formatted text files to see the resources that are in your template and their relationships can be difficult. In Designer, you can see a graphic representation of the resources that are included in a template and how they relate to each other.

Designer defines the information about your resources, such as their size and relative position, in template metadata. When you open a template, Designer automatically adds this metadata so that the current layout is preserved when you save your template. When you reopen a template in Designer, it displays the diagram exactly as it appeared when you last saved the template.

All layout information is defined in the `AWS::CloudFormation::Designer` metadata key, which is used only by Designer and won't interfere with creating AWS CloudFormation stacks. The following example of template metadata shows the layout information that Designer adds to a template as metadata:

JSON

```
"Metadata": {
  "AWS::CloudFormation::Designer": {
    "6b56eaae-0bb6-4215-aad6-12345EXAMPLE": {
      "size": {
        "width": 60,
        "height": 60
      },
      "position": {
        "x": 340,
        "y": 430
      },
      "z": 2,
      "parent": "21ccc9b0-29e9-4a86-9cf2-12345EXAMPLE",
      "embeds": [],
      "ismemberof": [
        "c3eead73-6a76-4532-9268-12345EXAMPLE"
      ]
    },
...
```

YAML

```
Metadata:
```

```
  'AWS::CloudFormation::Designer':
    6b56eaae-0bb6-4215-aad6-12345EXAMPLE:
      size:
        width: 60
        height: 60
      position:
        x: 340
        'y': 430
      z: 2
      parent: 21ccc9b0-29e9-4a86-9cf2-12345EXAMPLE
      embeds: []
      ismemberof:
        - c3eead73-6a76-4532-9268-12345EXAMPLE
...
```

Simplify Template Authoring

When you author template resources in a text editor, you must manually edit JSON or YAML, which can be tedious and error-prone. By using Designer, you spend less time manually coding your templates and more time designing your AWS infrastructure. In Designer, you drag and drop new resources to add them to your template, and you drag connections between resources to establish relationships. Designer automatically modifies the JSON or YAML.

When you create templates, Designer enforces some basic relationships between resources to help you create valid templates. For example, you cannot add an EC2 instance directly inside a VPC; you must add the instance inside a subnet in the VPC.

You can also validate a template directly in Designer. It provides the same level of validation as the `ValidateTemplate` API call, which checks that the JSON or YAML syntax is valid, that all referenced parameters are declared, and that there are no circular dependencies.

Simplify Editing with the integrated JSON and YAML editor

With the integrated editor, you can make all of your template modifications in the AWS CloudFormation console. You don't need to use a separate text editor to modify and save your templates. The integrated editor also provides an auto-complete feature that lists all property names for a resource, so you don't need to look them up or memorize them. In addition, you can use the integrated editor to convert JSON templates to YAML and vice versa.

AWS CloudFormation Designer Interface Overview

Designer has four panes. The **canvas** pane shows a diagram of your template resources so that you can see them and their relationships at a glance. To add resources to your template, you drag them from the **Resources types** pane onto the **canvas** pane. Use the **Integrated JSON and YAML editor** pane to specify template details, such as resource properties or template parameters. After you've modified the template, you can save it to a local file or to an Amazon S3 bucket. When you convert a valid template from JSON to YAML or vice-versa, the **Messages** pane displays a success or failure message. When you open or validate an invalid template, the **Messages** pane displays validation errors.

> **Note**
> Designer cannot show or modify running resources in your stacks; use it only for creating, modifying, and saving templates.

The following figure illustrates the Designer panes and its main components.

1. Toolbar

   The toolbar provides quick access to commands for common actions, such as opening and saving templates, undoing or redoing changes, creating a stack, and validating your template. You can also download the diagram as an image, get help, or refresh the diagram in the canvas pane.

2. Resource types pane

   The **Resource types** pane lists all of the template resources that you can add to your template, categorized by their AWS service name. You add resources by dragging them from the **Resource types** pane to the canvas. Most of the supported resources are listed in the AWS Resource Types Reference (p. 390). The **Resource types** pane doesn't list connecting resources, such as the `AWS::EC2::SubnetRouteTableAssociation` resource. You create these resources when you connect the relevant resources, such as when you connect a route table to a subnet. For more information, see Canvas Pane (p. 160).

   Note
   Designer can display only AWS CloudFormation-supported resource types. It cannot display other entities, such as Availability Zones (AZs) or the resources of a nested stack.
3. Canvas pane

   The **canvas** pane displays your template resources as a diagram. You use it to add or remove resources, create relationships between resources, and arrange their layout. The changes that you make in the **canvas** automatically modify the template's JSON or YAML. For more information, see Canvas Pane (p. 160).
4. Fit to window button

   A button that resizes the **canvas** pane to fit your template's diagram.
5. Full screen and Split screen buttons

   Buttons to select different views of Designer. You can select a full-screen view of the canvas, a full-screen view of the**Integrated JSON and YAML editor**, or a split-screen view of the canvas and editor.
6. Integrated JSON and YAML editor pane

   In the integrated editor, you specify the details of your template, such as resource properties or template parameters. When you select an item in the **canvas**, Designer highlights the related JSON or YAML in the editor. After editing the JSON or YAML, you must refresh the **canvas** (choose ) to update the diagram. You can convert a valid template between JSON and YAML by selecting the appropriate radio button in **Choose template language**. Designer can only convert valid YAML or valid JSON templates. If the conversion succeeds, the **Messages** pane displays a message like: *Successfully converted the template to YAML*. AWS CloudFormation Designer does not preserve formatting when converting a template.

   Important
   We recommend that you do not add `#` YAML comments to your templates in Designer. If your YAML template has `#` comments, Designer doesn't preserve those comments when editing the YAML or converting to JSON. If you edit or modify your template in Designer (for example, if you drag a resource on the canvas), your comments are lost.

   Once you choose a template language, any new resources you drag onto the canvas will be created in the language you have selected. To change back to another language, make sure your template is valid and then select **YAML** or **JSON** where it says **Choose template language**.

   For more information, see Integrated JSON and YAML Editor (p. 165).
7. Messages pane

   When you convert a template from JSON to YAML or vice-versa, the **Messages** pane displays a success or failure message. When you open, validate, or attempt to create a stack with an invalid template, the **Messages** pane displays validation errors.

Canvas Pane

Designer displays your template resources as a diagram in the **canvas** pane. You can modify the diagram's layout, add or remove resources, and add or remove connections between resources in this pane. For example, you can add an Auto Scaling group and a launch configuration from the **Resource types** pane to the **canvas** pane. To connect these related resources, you simply drag a connection between them.

How Does Designer Model Resources?

When you drag a resource from the **Resource types** pane to the **canvas** pane, Designer models it as a container or as a square object.

Containers

> Container resources are resizable rectangles that can contain other resources. For example, Designer models the `AWS::EC2::VPC` resource type as a container. You can drag resources, such as a subnet, into the VPC.

Square objects

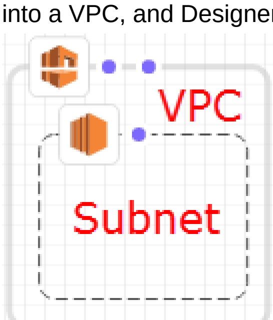
> Square objects resources can't be resized or contain other resources. For example, Designer models the `AWS::EC2::Instance` resource type as a square object.

Connecting Resources

You connect resources to create associations between related resources. For example, when you add an Internet gateway and a VPC to the **canvas** pane, they have no relationship. To attach the gateway to the VPC, you must connect them. The method for connecting resources depends on the resource type and how Designer models the resource. The following descriptions and figures explain each method.

Adding resources to containers

When you drag valid resource into containers, Designer automatically creates associations between the resource and the container. For example, VPCs are container resources; you can drag a subnet into a VPC, and Designer automatically associates the two resources.



These associations are represented in your template as a `Ref` intrinsic function, as shown in the following example:

JSON

```
"PublicSubnet": {
  "Type": "AWS::EC2::Subnet",
  "Properties": {
      "VpcId": {
          "Ref": "VPC"
      },
      "CidrBlock": "10.0.0.0/24"
  }
```

YAML

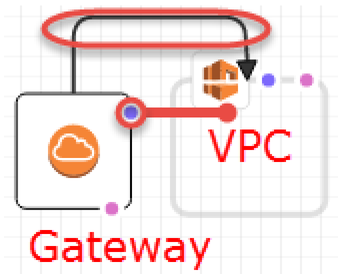
```
PublicSubnet:
```

```
  Type: 'AWS::EC2::Subnet'
  Properties:
    VpcId: !Ref VPC
    CidrBlock: 10.0.0.0/24
```

In some cases, dropping a resource into a container doesn't create an association; you must drag a connection between the resources (see the next method for information about dragging connections between resources). To see if Designer associates resources, use the integrated JSON and YAML editor to look for a `Ref` from one resource to the other. For example, when you add an Auto Scaling group in a subnet container, Designer doesn't specify the group's `VPCZoneIdentifier` (subnet) property. To associate the two resources, you must drag a connection from the Auto Scaling group to the subnet.

Dragging connections between resources

The edge of each square and container resource has one or more dots, which represent the resources that you can create connections with. To create a connection, drag a connector line from the dot to the corresponding resource type. For example, to attach an Internet gateway to a VPC, drag a line from the VPC gateway attachment dot to anywhere on the VPC.



These associations are represented in your template as a `Ref` intrinsic function or as a separate resource type. For example, when you connect an Internet gateway with a VPC, Designer creates an `AWS::EC2::VPCGatewayAttachment` resource type in your template to associate them. Resources like these are not listed in the **Resource types** pane.

JSON

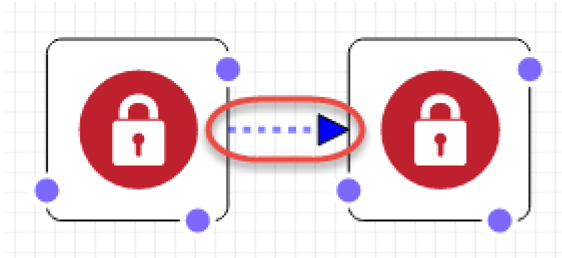
```
"VPCGatewayAttachment": {
  "Type": "AWS::EC2::VPCGatewayAttachment",
  "Properties": {
      "InternetGatewayId": {
          "Ref": "InternetGateway"
      },
      "VpcId": {
          "Ref": "VPC"
      }
  }
```

YAML

```
VPCGatewayAttachment:
  Type: 'AWS::EC2::VPCGatewayAttachment'
  Properties:
    InternetGatewayId: !Ref InternetGateway
    VpcId: !Ref VPC
```

Coding connections between resources

In some cases, you must edit the template's JSON or YAML to create connections, such as when you connect two security groups. When you must edit the JSON or YAML to create connections, you create hard-coded connections (dashed-line connections). You cannot create or edit these connections in the **canvas** pane.



Typically, when you embed references (`Ref`) within a resource's property, you create hard-coded connections. For example, you can define a connection between two security groups where one security group has an embedded ingress rule that permits traffic from the other. The following `WebServerSecurityGroup` resource has an ingress rule with a reference to the `PublicLoadBalancerSecurityGroup` resource.

JSON

```
"WebServerSecurityGroup": {
  "Type": "AWS::EC2::SecurityGroup",
  "Properties": {
      "VpcId": {
          "Ref": "VPC"
      },
      "GroupDescription": "Allow access from HTTP and SSH traffic",
      "SecurityGroupIngress": [
          {
              "IpProtocol": "tcp",
              "FromPort": "80",
              "ToPort": "80",
              "CidrIp": "0.0.0.0/0"
          },
          {
              "IpProtocol": "tcp",
              "FromPort": "22",
              "ToPort": "22",
              "CidrIp": {
                  "Ref": "SSHLocation"
              }
          }
      ]
  }
...
```

YAML

```
WebServerSecurityGroup:
  Type: 'AWS::EC2::SecurityGroup'
  Properties:
    VpcId: !Ref VPC
    GroupDescription: Allow access from HTTP and SSH traffic
    SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: '80'
        ToPort: '80'
        CidrIp: 0.0.0.0/0
      - IpProtocol: tcp
        FromPort: '22'
        ToPort: '22'
```

```
        CidrIp: !Ref SSHLocation
```

Accessing Common Resource Actions with the Resource Menu

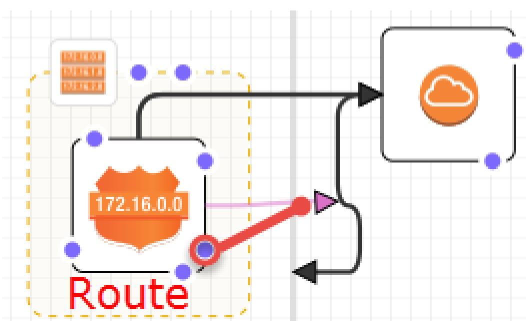
The **Resource** menu provides easy access to common resource actions: editing resource properties, duplicating a resource, deleting a resource, or viewing the documentation for the resource. To view the **Resource** menu, right-click on a resource in the **canvas** pane. The documentation link goes to the template reference (p. 390), which describes the properties and syntax for that resource.

Defining Explicit Dependencies

To specify the order in which AWS CloudFormation creates and deletes resources, you can create explicit dependencies. Explicit dependencies are useful for overriding parallel resource creation and deletion. AWS CloudFormation automatically determines which resources in a template can be processed in parallel and which cannot. When you specify a property that references an attribute from another source (using the `Ref` intrinsic function) or gets an attribute from another resource (with the `Fn::GetAtt` intrinsic function) in the same template, this implies a dependency and AWS CloudFormation builds them in the correct order.

However, in some cases, you must explicitly define dependencies. For example, a routing rule cannot use an Internet gateway until the gateway has been attached to the VPC. Normally, AWS CloudFormation creates the routing rule immediately after it creates the Internet gateway due to an implicit dependency. But, AWS CloudFormation might create the rule before the Internet gateway has attached to the VPC, which causes an error. Therefore, you must explicitly define a dependency on the gateway-VPC attachment.

To create an explicit dependency, drag a line from the `DependsOn` (*****) dot on the route to the gateway-VPC attachment.



For more information about when you might need to create an explicit dependency, see DependsOn Attribute (p. 1298).

JSON

In JSON, these explicit dependencies are represented as a `DependsOn` attribute on a resource, as shown in the following example:

```
"PublicRoute": {
  "Type": "AWS::EC2::Route",
  "DependsOn": "VPCGatewayAttachment",
  "Properties": {
      "DestinationCidrBlock": "0.0.0.0/0",
      "RouteTableId": {
          "Ref": "PublicRouteTable"
      },
      "GatewayId": {
          "Ref": "InternetGateway"
      }
```

```
}
```

YAML

In YAML, these explicit dependencies are represented as a `DependsOn` attribute on a resource, as shown in the following example:

```
PublicRoute:
  Type: 'AWS::EC2::Route'
  DependsOn:
    - VPCGatewayAttachment
  Properties:
    DestinationCidrBlock: 0.0.0.0/0
    RouteTableId: !Ref PublicRouteTable
    GatewayId: !Ref InternetGateway
```

Integrated JSON and YAML Editor

Use Designer's integrated JSON and YAML editor to view and edit template details. For example, you can use the integrated editor to define the properties of a resource or to change a template parameter. The integrated editor has two views: a **Components** view and a **Template** view.

To make minor changes to a specific section of a template, use the **Components** view. In the **Components** view, the components that you can edit are divided into tabs. These tabs change depending on whether you have a resource selected.

For example, if you select a resource, Designer provides tabs to edit the resource's properties and attributes, such as an update policy or creation policy. If you haven't selected anything, Designer provides tabs for editing the template parameters, mappings, conditions, metadata, and outputs. Any changes that you make in the **Components** view must be valid JSON or YAML markup. If you introduce invalid JSON or YAML, Designer reverts the invalid markup to the valid markup when you leave the **Components** view.

To make broad changes to your template, use the **Template** view. In the **Template** view, the integrated JSON and YAML editor shows you the raw JSON or YAML of your entire template. When you want to make changes to a resource, select it in the canvas pane. Designer automatically highlights that resource in the integrated JSON and YAML editor.

Converting templates into YAML or JSON

You can convert a valid template back and forth between JSON and YAML by selecting the appropriate radio button in **Choose template language**. Designer can only convert valid YAML or valid JSON templates. If the conversion succeeds, the **Messages** pane displays a message like: *Successfully converted the template to YAML*.

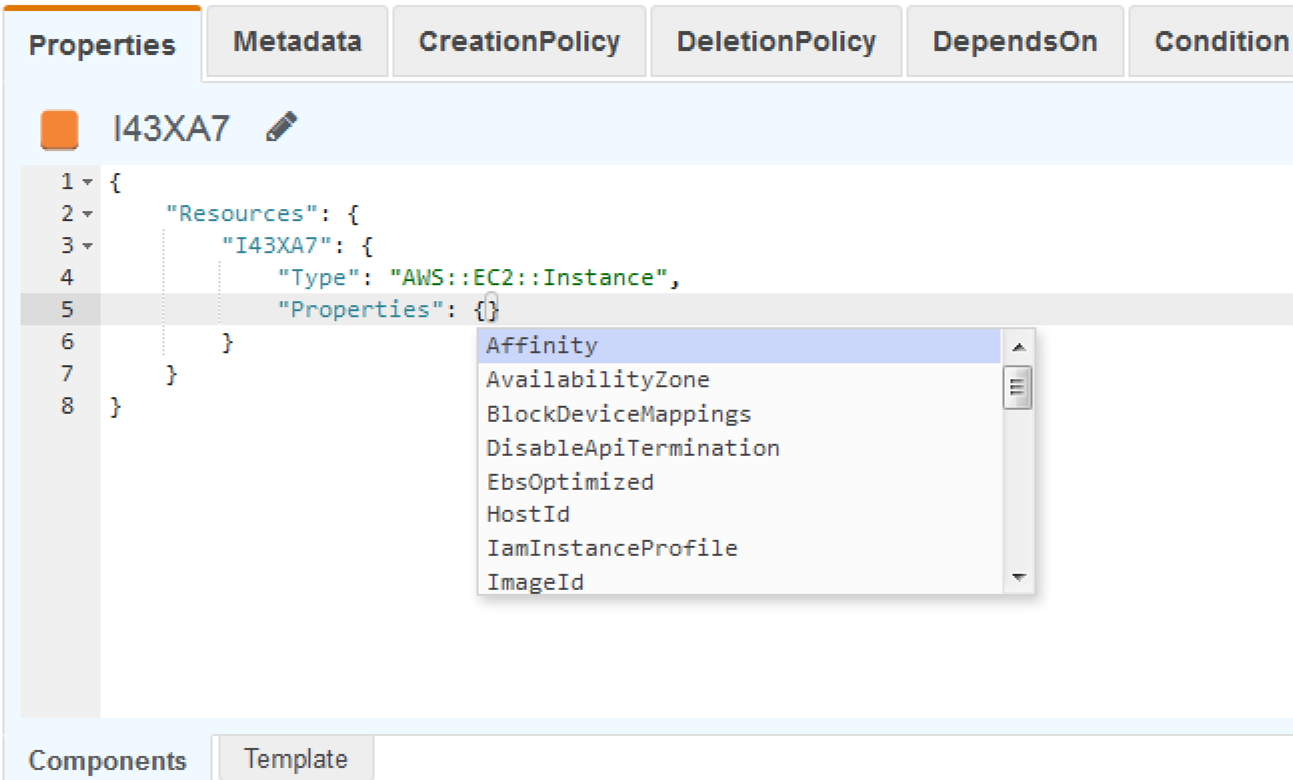
> **Important**
> We recommend that you do not add `#` YAML comments to your templates in Designer. If your YAML template has `#` comments, Designer doesn't preserve those comments when editing the YAML or converting to JSON. If you edit or modify your template in Designer (for example, if you drag a resource on the canvas), your comments are lost.

Once you choose a template language, any new resources you drag onto the canvas will be created in the language you have selected. To change back to another language, make sure your template is valid and then select **YAML** or **JSON** where it says **Choose template language**.

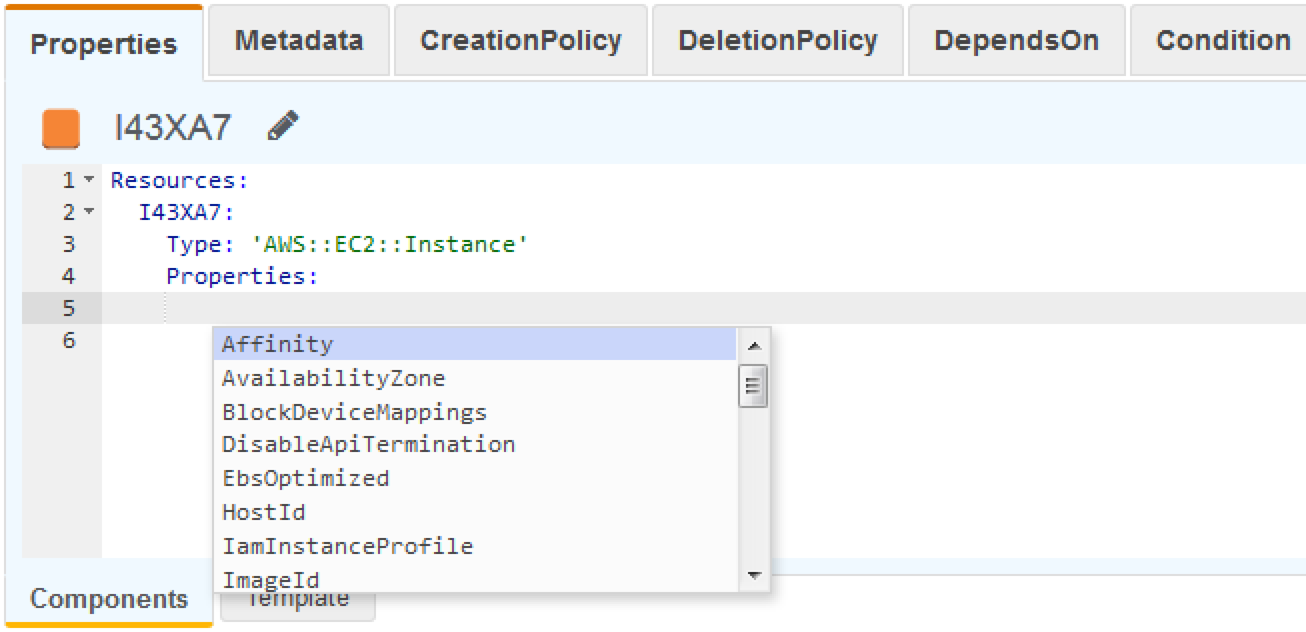
> **Note**
> When you convert a template to YAML, Designer uses short form notation for functions. For example, `- !GetAtt`. In addition, any visual links that you draw will use short form notation in YAML mode. For more information about intrinsic functions, see `Ref` (p. 1343).

Autocomplete

The integrated JSON and YAML editor includes an auto-complete feature that helps you specify resource properties, so you don't have to remember property names. To see a list of valid properties in a JSON template, press **`Ctrl+Space`** within the `Properties` curly braces (`{}`), as shown in the following example:



For a YAML template, you can first delete the opening and closing curly braces and press **`Enter`** to go to a new line. To see a list of valid properties, press **`Ctrl+Space`** on the new line after `Properties`, as shown in the following example:



Keyboard Shortcuts

Designer's integrated JSON and YAML editor provides the following keyboard shortcuts:

Ctrl+Space

Within the `Properties` key of a resource, lists all of the available properties for the resource.

Ctrl+F

Searches for a specified value.

To highlight everything that matches the specified value, press **`Alt+Enter`**.

How to Get Started With Designer

For examples of how to use AWS CloudFormation Designer to create and update templates, see the following walkthroughs:

- Walkthrough: Use AWS CloudFormation Designer to Create a Basic Web Server (p. 168)
- Walkthrough: Use AWS CloudFormation Designer to Modify a Stack's Template (p. 185)

Walkthroughs

Templates are JSON- or YAML-formatted text files that describe the AWS resources that you want to provision or update in your AWS CloudFormation stacks. To create templates, you can use AWS CloudFormation Designer or a text editor.

The following walkthroughs show how to create sample AWS CloudFormation templates using AWS CloudFormation Designer and plain text.

Topics

Walkthrough: Use AWS CloudFormation Designer to Create a Basic Web Server

AWS CloudFormation Designer graphically represents your templates to help you see the resources in the template and how they're connected. The integrated JSON and YAML editor makes it easy to modify templates directly in the AWS CloudFormation console. To demonstrate how to use both of these components, we'll use AWS CloudFormation Designer to build a basic web server in a VPC. Then, we'll save the template and use it to create an AWS CloudFormation stack. By the end of the walkthrough, you'll have a template similar to the following sample: https://console.aws.amazon.com/cloudformation/designer/home?templateUrl=https://s3.amazonaws.com/cloudformation-examples/sample-ec2-vpc.template&region=us-east-1.

In the walkthrough, you will complete the following steps:

1. Add and connect resources. (p. 169)

   When you first open AWS CloudFormation Designer, you start with a blank template. We'll use AWS CloudFormation Designer to start populating the template by dragging resources, such as a VPC and an EC2 instance into your template. We'll also create links between them. For example, we'll use AWS CloudFormation Designer to create a connection between the Internet gateway and the VPC.
2. Add template parameters, mappings, and outputs. (p. 171)

   We'll use the AWS CloudFormation Designer integrated editor to add other template components to make the template more useful. For example, we'll add parameters to the template so that you can specify input values when you create a stack. That way you don't have to constantly edit the template for property values that you might commonly change.
3. Specify resource properties. (p. 178)

   We'll use the integrated editor again to specify configuration settings for our resources.
4. Provision resources (p. 184)

   None of your template resources are up and running until you create a stack. We'll use the template that you just created to launch an AWS CloudFormation stack, which will provision all the resources that are defined in your template.

   Note
   AWS CloudFormation is a free service; however, you are charged for the AWS resources you include in your stacks at the current rate for each. For more information about AWS pricing, see the detail page for each product on http://aws.amazon.com.

Prerequisites

This walkthrough assumes that you have a working knowledge of Amazon Virtual Private Cloud (Amazon VPC), Amazon Elastic Compute Cloud (Amazon EC2), and AWS CloudFormation. For context, each procedure provides some basic information about each resource.

Also, before you begin, make sure you have an Amazon EC2 key pair in the region in which you're creating your stack. For more information, see Amazon EC2 Key Pairs in the *Amazon EC2 User Guide for Linux Instances*.

Step 1: Add and Connect Resources

We'll use the AWS CloudFormation Designer drag-and-drop interface to add an EC2 instance and network resources, such as a VPC, subnet, route table, and Internet gateway. After adding all the resources, we'll create connections between them. For example, we'll associate the Internet gateway with a VPC.

To add resources to a template

1. Open AWS CloudFormation Designer at https://console.aws.amazon.com/cloudformation/designer.
2. In the integrated editor, choose **Edit** ( ).
3. Change the template name to `BasicWebServerInVPC` and then press **Enter**.

   Currently, we have a blank template that isn't valid. In the next steps, we'll add resources to make it valid.
4. Drag a **VPC** resource type from the **Resources** pane onto the **Canvas** pane.

   The resources are organized by resource categories. All of the resources we're adding are in the **EC2** category.

   AWS CloudFormation Designer immediately modifies your template to include a VPC resource, with the results looking similar to the following JSON snippet.

```
"Resources": {
  "VPC431KO": {
    "Type": "AWS::EC2::VPC",
    "Properties": {},
    "Metadata": {
      "AWS::CloudFormation::Designer": {
        "id": "445730ea-0d11-45ba-b6ac-12345EXAMPLE"
      }
    }
  }
}
```

   The YAML snippet looks similar to the following.

```
Resources:
  VPC431KO:
    Type: 'AWS::EC2::VPC'
    Metadata:
      'AWS::CloudFormation::Designer':
        id: 9430b008-7a03-41ed-b63e-12345EXAMPLE
```

   Note that we still need to specify the VPC properties, such as the VPC's CIDR block. We'll do that later. This is true for all resources that we'll add.
5. Rename the VPC.

   Note
   When you rename a resource, you rename its logical ID, which is the name that is referenced in the template (not the name assigned when AWS CloudFormation creates the resource). For more information, see Resources (p. 153).

   a. Choose the VPC resource.

   b. In the integrated editor, choose the **Edit** icon ( ).
   c. Change the name to `VPC`, and then choose **Enter**.

   Next, we'll add resources to the VPC.

6. Drag a corner of the VPC resource to expand it so that it's large enough to fit several more resources.

   We need to add a subnet because you can't add an EC2 instance, which hosts the website, directly into the VPC; instances must be located in a subnet.
7. Add a **Subnet** resource type inside the VPC and rename it `PublicSubnet`.

   We will use the subnet to allocate a range of IP addresses in the VPC that you can associate with other AWS resources, such as an Amazon EC2 instance.

   When you add the subnet inside the VPC, AWS CloudFormation Designer automatically associates the subnet with the VPC. This association is a container model, where resources inside the container are automatically associated with the container resource.
8. Add an **Instance** resource type inside the `PublicSubnet` resource and rename it `WebServerInstance`.

   The instance is a virtual computing environment where you'll host a basic website. Similar to the way this worked with the subnet and VPC, adding the instance in the subnet automatically associates the instance with the subnet.
9. Add a **SecurityGroup** resource type inside the VPC and rename it `WebServerSecurityGroup`.

   The security group is a virtual firewall that controls the inbound and outbound traffic of the web server instance. It's also required for instances in a VPC. We'll need to associate the web server instance with this security group, which we'll do later when we specify the instance's properties.
10. Add an **InternetGateway** resource type anywhere outside of the VPC and rename it `InternetGateway`.

    The Internet gateway enables communication between the instance that is inside the VPC and the Internet. Without the Internet gateway, no one can access your website.

    Although, you can drag the Internet gateway inside the VPC, this doesn't create an association with the VPC. The Internet gateway doesn't follow the container model; instead, you must drag a connection from the Internet gateway to the VPC, as described in the next step.
11. Create a connection between the `InternetGateway` resource and the `VPC` resource.

    a. On the `InternetGateway` resource, hover over the Internet gateway attachment (`AWS::EC2::VPCGatewayAttachment`).
    b. Drag a connection to the VPC.

       The border of valid target resources changes color. In this case, the VPC is the only valid target resource. This connection creates an attachment resource that associates the Internet gateway with the VPC.
12. Next, we need to add a route table and route to specify how to direct network traffic from within a subnet. Add a **RouteTable** inside the VPC and rename it `PublicRouteTable`.

    This associates a new route table with the VPC.
13. To add a routing rule to the route table, add a **Route** resource type inside the `PublicRouteTable` resource and rename it `PublicRoute`.

    We'll use the route to specify where to direct traffic.
14. For the public route, we want the Internet gateway to be the destination target. Use `GatewayId` to create a connection from the `PublicRoute` resource to the Internet gateway, similar to the way you created a connection between the Internet gateway and the VPC.

AWS CloudFormation can't associate a route with an Internet gateway until you associate the Internet gateway with the VPC. This means we need to create an explicit dependency on the Internet gateway-VPC attachment, as described in the next step. For more information, see DependsOn Attribute (p. 1298).

15. Create an explicit dependency between the `PublicRoute` resource and the Internet gateway-VPC attachment.

    a. On the `PublicRoute` resource, hover over the **DependsOn** dot.
    b. Drag a connection to the Internet gateway-VPC attachment (`AWS::EC2::VPCGatewayAttachment`).

       With `DependsOn` connections, AWS CloudFormation Designer creates a dependency (a `DependsOn` attribute), where the originating resource depends on the target resource. In this case, AWS CloudFormation Designer adds a `DependsOn` attribute to the `PublicRoute` resource and specifies the gateway-VPC attachment as a dependency.

16. Create another dependency from the `WebServerInstance` resource to the `PublicRoute` resource.

    The `WebServerInstance` resource depends on the public route to route traffic to the Internet. Without the public route, the instance cannot send a signal (using the cfn-signal helper script) to notify AWS CloudFormation when the instance configuration and application deployments are complete.

17. Drag a connection from the `PublicRouteTable` resource to the `PublicSubnet` resource to associate the route table and subnet.

    Now the public subnet will use the public route table to direct traffic.

18. From the AWS CloudFormation Designer toolbar, save the template locally by using the **File** menu (the file icon).

    AWS CloudFormation Designer saves your template on your hard drive. You can use the template later to create a stack. We recommend that you save the template regularly to avoid losing changes.

In this step, we added seven resources to your template and renamed their logical IDs with friendly names. We also established visual connections with most of the resources to create associations and a dependency. However, before we can create a stack with this template, we need to create a few more connections (such as associating the instance with the security group) and to specify properties for each resource. In the next step, we'll walk through modifying other components of your template, such as input parameters, by using the AWS CloudFormation Designer integrated editor.

Step 2: Add Parameters, Mappings, and Outputs

Before we specify resource properties, we need to add other template components to make reusing the template in multiple environments easier. In this step, we'll use the AWS CloudFormation Designer integrated editor to add parameters, mappings, and outputs. Then, we can refer to these parameters and mappings when we specify resource properties. The walkthrough provides sample JSON and YAML that you can use to copy and paste in to the integrated editor.

To add parameters

Parameters are input values that you specify when you create a stack. They're useful for passing in values so that you don't have hard coded values in templates. For example, you don't need to hard code your web server's instance type in your template; instead, you can use a parameter to specify the instance type when you create a stack. That way you can use the same template to create multiple web servers with different instance types. For more information, see Parameters (p. 132).

1. Click on an open area in the AWS CloudFormation Designer canvas.

   Depending on what you have selected, the integrated editor shows either template-level or resource-level components that you can edit. At the template-level, you can edit all other sections of a template, such as template parameters, mappings, and outputs, except for the Resources section. At the resource-level, you can edit resource properties and attributes.

Clicking on an open area in the canvas allows you to edit template-level components. To edit resource-level components, select a resource.

2. In the integrated editor pane, choose the **Parameters** tab.
3. Copy the parameters in the following snippet and paste them into the integrated editor.

The following JSON snippet adds parameters for specifying your web server's instance type, an Amazon EC2 key-pair name for SSH access to the web server, and the IP address range that can be used to access the web server using SSH.

```
{
  "Parameters": {
    "InstanceType": {
      "Description": "WebServer EC2 instance type",
      "Type": "String",
      "Default": "t2.micro",
      "AllowedValues": [
        "t1.micro",
        "t2.micro",
        "t2.small",
        "t2.medium",
        "m1.small",
        "m1.medium",
        "m1.large",
        "m1.xlarge",
        "m2.xlarge",
        "m2.2xlarge",
        "m2.4xlarge",
        "m3.medium",
        "m3.large",
        "m3.xlarge",
        "m3.2xlarge",
        "c1.medium",
        "c1.xlarge",
        "c3.large",
        "c3.xlarge",
        "c3.2xlarge",
        "c3.4xlarge",
        "c3.8xlarge",
        "c4.large",
        "c4.xlarge",
        "c4.2xlarge",
        "c4.4xlarge",
        "c4.8xlarge",
        "g2.2xlarge",
        "r3.large",
        "r3.xlarge",
        "r3.2xlarge",
        "r3.4xlarge",
        "r3.8xlarge",
        "i2.xlarge",
        "i2.2xlarge",
        "i2.4xlarge",
        "i2.8xlarge",
        "d2.xlarge",
        "d2.2xlarge",
        "d2.4xlarge",
        "d2.8xlarge",
        "hi1.4xlarge",
        "hs1.8xlarge",
        "cr1.8xlarge",
```

```
        "cc2.8xlarge",
        "cg1.4xlarge"
      ],
      "ConstraintDescription": "must be a valid EC2 instance type."
    },
    "KeyName": {
      "Description": "Name of an EC2 KeyPair to enable SSH access to the instance.",
      "Type": "AWS::EC2::KeyPair::KeyName",
      "ConstraintDescription": "must be the name of an existing EC2 KeyPair."
    },
    "SSHLocation": {
      "Description": " The IP address range that can be usedto access the web server
 using SSH.",
      "Type": "String",
      "MinLength": "9",
      "MaxLength": "18",
      "Default": "0.0.0.0/0",
      "AllowedPattern": "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\\d{1,2})",
      "ConstraintDescription": "must be a valid IP CIDR range of the form x.x.x.x/x."
    }
  }
}
```

Here is the same snippet in YAML.

```
Parameters:
  InstanceType:
    Description: WebServer EC2 instance type
    Type: String
    Default: t2.micro
    AllowedValues:
      - t1.micro
      - t2.micro
      - t2.small
      - t2.medium
      - m1.small
      - m1.medium
      - m1.large
      - m1.xlarge
      - m2.xlarge
      - m2.2xlarge
      - m2.4xlarge
      - m3.medium
      - m3.large
      - m3.xlarge
      - m3.2xlarge
      - c1.medium
      - c1.xlarge
      - c3.large
      - c3.xlarge
      - c3.2xlarge
      - c3.4xlarge
      - c3.8xlarge
      - c4.large
      - c4.xlarge
      - c4.2xlarge
      - c4.4xlarge
      - c4.8xlarge
      - g2.2xlarge
      - r3.large
      - r3.xlarge
      - r3.2xlarge
      - r3.4xlarge
      - r3.8xlarge
```

```
      - i2.xlarge
      - i2.2xlarge
      - i2.4xlarge
      - i2.8xlarge
      - d2.xlarge
      - d2.2xlarge
      - d2.4xlarge
      - d2.8xlarge
      - hi1.4xlarge
      - hs1.8xlarge
      - cr1.8xlarge
      - cc2.8xlarge
      - cg1.4xlarge
    ConstraintDescription: must be a valid EC2 instance type.
  KeyName:
    Description: Name of an EC2 KeyPair to enable SSH access to the instance.
    Type: 'AWS::EC2::KeyPair::KeyName'
    ConstraintDescription: must be the name of an existing EC2 KeyPair.
  SSHLocation:
    Description: ' The IP address range that can be usedto access the web server using
 SSH.'
    Type: String
    MinLength: '9'
    MaxLength: '18'
    Default: 0.0.0.0/0
    AllowedPattern: '(\d{1,3})\.(\d{1,3})\.(\d{1,3})\.(\d{1,3})/(\d{1,2})'
    ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.
```

To add mappings

Mappings are a set of keys that are associated with a set of name-value pairs. They're useful for specifying values based on an input parameter value. For this walkthrough, we'll use a mapping to specify an AMI ID for an EC2 instance based on the instance type and region in which you create the stack. For more information, see Mappings (p. 140).

1. In the integrated editor pane, choose the **Mappings** tab.
2. Copy the following JSON mappings and paste them into the integrated editor.

```
{
  "Mappings": {
    "AWSInstanceType2Arch" : {
      "t1.micro"    : { "Arch" : "PV64"   },
      "t2.micro"    : { "Arch" : "HVM64"  },
      "t2.small"    : { "Arch" : "HVM64"  },
      "t2.medium"   : { "Arch" : "HVM64"  },
      "m1.small"    : { "Arch" : "PV64"   },
      "m1.medium"   : { "Arch" : "PV64"   },
      "m1.large"    : { "Arch" : "PV64"   },
      "m1.xlarge"   : { "Arch" : "PV64"   },
      "m2.xlarge"   : { "Arch" : "PV64"   },
      "m2.2xlarge"  : { "Arch" : "PV64"   },
      "m2.4xlarge"  : { "Arch" : "PV64"   },
      "m3.medium"   : { "Arch" : "HVM64"  },
      "m3.large"    : { "Arch" : "HVM64"  },
      "m3.xlarge"   : { "Arch" : "HVM64"  },
      "m3.2xlarge"  : { "Arch" : "HVM64"  },
      "c1.medium"   : { "Arch" : "PV64"   },
      "c1.xlarge"   : { "Arch" : "PV64"   },
      "c3.large"    : { "Arch" : "HVM64"  },
      "c3.xlarge"   : { "Arch" : "HVM64"  },
      "c3.2xlarge"  : { "Arch" : "HVM64"  },
      "c3.4xlarge"  : { "Arch" : "HVM64"  },
```

```
      "c3.8xlarge"  : { "Arch" : "HVM64"  },
      "c4.large"    : { "Arch" : "HVM64"  },
      "c4.xlarge"   : { "Arch" : "HVM64"  },
      "c4.2xlarge"  : { "Arch" : "HVM64"  },
      "c4.4xlarge"  : { "Arch" : "HVM64"  },
      "c4.8xlarge"  : { "Arch" : "HVM64"  },
      "g2.2xlarge"  : { "Arch" : "HVMG2"  },
      "r3.large"    : { "Arch" : "HVM64"  },
      "r3.xlarge"   : { "Arch" : "HVM64"  },
      "r3.2xlarge"  : { "Arch" : "HVM64"  },
      "r3.4xlarge"  : { "Arch" : "HVM64"  },
      "r3.8xlarge"  : { "Arch" : "HVM64"  },
      "i2.xlarge"   : { "Arch" : "HVM64"  },
      "i2.2xlarge"  : { "Arch" : "HVM64"  },
      "i2.4xlarge"  : { "Arch" : "HVM64"  },
      "i2.8xlarge"  : { "Arch" : "HVM64"  },
      "d2.xlarge"   : { "Arch" : "HVM64"  },
      "d2.2xlarge"  : { "Arch" : "HVM64"  },
      "d2.4xlarge"  : { "Arch" : "HVM64"  },
      "d2.8xlarge"  : { "Arch" : "HVM64"  },
      "hi1.4xlarge" : { "Arch" : "HVM64"  },
      "hs1.8xlarge" : { "Arch" : "HVM64"  },
      "cr1.8xlarge" : { "Arch" : "HVM64"  },
      "cc2.8xlarge" : { "Arch" : "HVM64"  }
    },
    "AWSRegionArch2AMI" : {
      "us-east-1"        : {"PV64" : "ami-1ccae774", "HVM64" : "ami-1ecae776",
 "HVMG2" : "ami-8c6b40e4"},
      "us-west-2"        : {"PV64" : "ami-ff527ecf", "HVM64" : "ami-e7527ed7",
 "HVMG2" : "ami-abbe919b"},
      "us-west-1"        : {"PV64" : "ami-d514f291", "HVM64" : "ami-d114f295",
 "HVMG2" : "ami-f31ffeb7"},
      "eu-west-1"        : {"PV64" : "ami-bf0897c8", "HVM64" : "ami-a10897d6",
 "HVMG2" : "ami-d5bc24a2"},
      "eu-central-1"     : {"PV64" : "ami-ac221fb1", "HVM64" : "ami-a8221fb5",
 "HVMG2" : "ami-7cd2ef61"},
      "ap-northeast-1"   : {"PV64" : "ami-27f90e27", "HVM64" : "ami-cbf90ecb",
 "HVMG2" : "ami-6318e863"},
      "ap-southeast-1"   : {"PV64" : "ami-acd9e8fe", "HVM64" : "ami-68d8e93a",
 "HVMG2" : "ami-3807376a"},
      "ap-southeast-2"   : {"PV64" : "ami-ff9cecc5", "HVM64" : "ami-fd9cecc7",
 "HVMG2" : "ami-89790ab3"},
      "sa-east-1"        : {"PV64" : "ami-bb2890a6", "HVM64" : "ami-b52890a8",
 "HVMG2" : "NOT_SUPPORTED"},
      "cn-north-1"       : {"PV64" : "ami-fa39abc3", "HVM64" : "ami-f239abcb",
 "HVMG2" : "NOT_SUPPORTED"}
    }
  }
}
```

Here are the same mappings in YAML.

```
Mappings:
  AWSInstanceType2Arch:
    t1.micro:
      Arch: PV64
    t2.micro:
      Arch: HVM64
    t2.small:
      Arch: HVM64
    t2.medium:
      Arch: HVM64
    m1.small:
      Arch: PV64
```

```
m1.medium:
  Arch: PV64
m1.large:
  Arch: PV64
m1.xlarge:
  Arch: PV64
m2.xlarge:
  Arch: PV64
m2.2xlarge:
  Arch: PV64
m2.4xlarge:
  Arch: PV64
m3.medium:
  Arch: HVM64
m3.large:
  Arch: HVM64
m3.xlarge:
  Arch: HVM64
m3.2xlarge:
  Arch: HVM64
c1.medium:
  Arch: PV64
c1.xlarge:
  Arch: PV64
c3.large:
  Arch: HVM64
c3.xlarge:
  Arch: HVM64
c3.2xlarge:
  Arch: HVM64
c3.4xlarge:
  Arch: HVM64
c3.8xlarge:
  Arch: HVM64
c4.large:
  Arch: HVM64
c4.xlarge:
  Arch: HVM64
c4.2xlarge:
  Arch: HVM64
c4.4xlarge:
  Arch: HVM64
c4.8xlarge:
  Arch: HVM64
g2.2xlarge:
  Arch: HVMG2
r3.large:
  Arch: HVM64
r3.xlarge:
  Arch: HVM64
r3.2xlarge:
  Arch: HVM64
r3.4xlarge:
  Arch: HVM64
r3.8xlarge:
  Arch: HVM64
i2.xlarge:
  Arch: HVM64
i2.2xlarge:
  Arch: HVM64
i2.4xlarge:
  Arch: HVM64
i2.8xlarge:
  Arch: HVM64
d2.xlarge:
  Arch: HVM64
```

```
    d2.2xlarge:
      Arch: HVM64
    d2.4xlarge:
      Arch: HVM64
    d2.8xlarge:
      Arch: HVM64
    hi1.4xlarge:
      Arch: HVM64
    hs1.8xlarge:
      Arch: HVM64
    cr1.8xlarge:
      Arch: HVM64
    cc2.8xlarge:
      Arch: HVM64
  AWSRegionArch2AMI:
    us-east-1:
      PV64: ami-1ccae774
      HVM64: ami-1ecae776
      HVMG2: ami-8c6b40e4
    us-west-2:
      PV64: ami-ff527ecf
      HVM64: ami-e7527ed7
      HVMG2: ami-abbe919b
    us-west-1:
      PV64: ami-d514f291
      HVM64: ami-d114f295
      HVMG2: ami-f31ffeb7
    eu-west-1:
      PV64: ami-bf0897c8
      HVM64: ami-a10897d6
      HVMG2: ami-d5bc24a2
    eu-central-1:
      PV64: ami-ac221fb1
      HVM64: ami-a8221fb5
      HVMG2: ami-7cd2ef61
    ap-northeast-1:
      PV64: ami-27f90e27
      HVM64: ami-cbf90ecb
      HVMG2: ami-6318e863
    ap-southeast-1:
      PV64: ami-acd9e8fe
      HVM64: ami-68d8e93a
      HVMG2: ami-3807376a
    ap-southeast-2:
      PV64: ami-ff9cecc5
      HVM64: ami-fd9cecc7
      HVMG2: ami-89790ab3
    sa-east-1:
      PV64: ami-bb2890a6
      HVM64: ami-b52890a8
      HVMG2: NOT_SUPPORTED
    cn-north-1:
      PV64: ami-fa39abc3
      HVM64: ami-f239abcb
      HVMG2: NOT_SUPPORTED
```

To add outputs

Outputs declare values that you want available to a `describe stacks` API call or through the AWS CloudFormation console stack **Outputs** tab. For this walkthrough, we'll output the website URL so that you can easily view the website after we create it. For more information, see Outputs (p. 155).

1. In the integrated editor pane, select the **Outputs** tab.

2. Copy the following JSON output and paste it into the integrated editor.

   The output uses an `Fn::GetAtt` intrinsic function to get the public IP of the web server instance.

```
{
  "Outputs": {
    "URL": {
      "Value": {
        "Fn::Join": [
          "",
          [
            "http://",
            {
              "Fn::GetAtt": [
                "WebServerInstance",
                "PublicIp"
              ]
            }
          ]
        ]
      },
      "Description": "Newly created application URL"
    }
  }
}
```

   Here is the same output in YAML.

```
Outputs:
  URL:
    Value: !Join
      - ''
      - - 'http://'
        - !GetAtt
          - WebServerInstance
          - PublicIp
    Description: Newly created application URL
```

3. Save your template again so that you don't lose your changes. You can safely save your changes to the same file that you created in the previous section.

Now that the template parameters, mappings, and outputs are in place, we can specify resource properties.

Step 3: Specify Resource Properties

Many resources have required properties that define their configurations or settings, such as which instance type to use for the web server. Similar to what we did in the previous step, we'll use the AWS CloudFormation Designer integrated editor to specify resource properties. We provide sample JSON and YAML that you can copy and paste into the integrated editor.

To specify resource properties

1. On the AWS CloudFormation Designer canvas, choose the `VPC` resource.

   The integrated editor shows the resource-level components that you can edit, such as the resource properties and attributes.
2. In the integrated editor pane, choose the **Properties** tab.
3. Copy the following JSON snippet and paste it into the integrated editor between the **Properties** braces (`{}`).

This snippet specifies DNS settings and the CIDR block of the VPC.

```
"EnableDnsSupport": "true",
"EnableDnsHostnames": "true",
"CidrBlock": "10.0.0.0/16"
```

For YAML, type a new line after `Properties:` and paste the following snippet.

```
EnableDnsSupport: 'true'
EnableDnsHostnames: 'true'
CidrBlock: 10.0.0.0/16
```

Note
For efficiency, we provide JSON and YAML snippets that you can copy and paste. Note, however, that the editor has an auto-complete feature that you can use to manually specify each property. For more information, see Integrated JSON and YAML Editor (p. 165).

4. Repeat this process for the following resources:

`PublicSubnet`

Add the following CIDR block property after the VPC ID property. AWS CloudFormation Designer automatically added the VPC ID property when you dragged the subnet inside the VPC.

Note
You'll see a few other associations that AWS CloudFormation Designer automatically created for you. Add just the new properties, which are in bold.

JSON

```
"VpcId": {
  "Ref": "VPC"
},
"CidrBlock": "10.0.0.0/24"
```

YAML

```
VpcId: !Ref VPC
CidrBlock: 10.0.0.0/24
```

`PublicRoute`

Add the following destination CIDR block property, which directs all traffic to the Internet gateway:

JSON

```
"DestinationCidrBlock": "0.0.0.0/0",
"RouteTableId": {
  "Ref": "PublicRouteTable"
},
"GatewayId": {
  "Ref": "InternetGateway"
}
```

YAML

```
DestinationCidrBlock: 0.0.0.0/0
```

```
      RouteTableId: !Ref PublicRouteTable
      GatewayId: !Ref InternetGateway
```

`WebServerSecurityGroup`

Add the following inbound rules that determine what traffic can reach the web server instance. The rules allow all HTTP and certain SSH traffic, which you specify as a parameter value when you create a stack.

JSON

```
        "VpcId": {
          "Ref": "VPC"
        },
        "GroupDescription" : "Allow access from HTTP and SSH traffic",
        "SecurityGroupIngress": [
          {
            "IpProtocol": "tcp",
            "FromPort": "80",
            "ToPort": "80",
            "CidrIp": "0.0.0.0/0"
          },
          {
            "IpProtocol": "tcp",
            "FromPort": "22",
            "ToPort": "22",
            "CidrIp": {
              "Ref": "SSHLocation"
            }
          }
        ]
```

YAML

```
      VpcId: !Ref VPC
      GroupDescription: Allow access from HTTP and SSH traffic
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: '80'
          ToPort: '80'
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: '22'
          ToPort: '22'
          CidrIp: !Ref SSHLocation
```

`WebServerInstance`

You need to specify a number of properties for the web server instance, so we'll highlight just a few for demonstration purposes. The `InstanceType` and `ImageId` properties use the parameter and mapping values that we specified in the previous section. When you create a stack, you specify the instance type as a parameter value. The `ImageId` value is a mapping that is based on your stack's region and the instance type that you specified.

The `NetworkInterfaces` property specifies network settings for the web server instance. This property allows us to associate the security group and subnet with the instance. Although AWS CloudFormation Designer used the `SubnetId` property to associate the instance with the subnet, we need to use the `NetworkInterfaces` property because that's the only way to give the web server a public IP. And when you specify the `NetworkInterfaces` property, you are required to specify the subnet and security group within that property.

In the `UserData` property, we specify configuration scripts that run after the instance is up and running. All of the configuration information is defined in the instance's metadata, which we'll add in the next step.

Replace all properties with the following snippet:

Important
Do not append this snippet to existing properties.

JSON

```
      "InstanceType": {
         "Ref": "InstanceType"
       },
       "ImageId": {
         "Fn::FindInMap": [
           "AWSRegionArch2AMI",
           {
             "Ref": "AWS::Region"
           },
           {
             "Fn::FindInMap": [
               "AWSInstanceType2Arch",
               {
                 "Ref": "InstanceType"
               },
               "Arch"
             ]
           }
         ]
       },
       "KeyName": {
         "Ref": "KeyName"
       },
       "NetworkInterfaces": [
         {
           "GroupSet": [
             {
               "Ref": "WebServerSecurityGroup"
             }
           ],
           "AssociatePublicIpAddress": "true",
           "DeviceIndex": "0",
           "DeleteOnTermination": "true",
           "SubnetId": {
             "Ref": "PublicSubnet"
           }
         }
       ],
       "UserData": {
         "Fn::Base64": {
           "Fn::Join": [
             "",
             [
               "#!/bin/bash -xe\n",
               "yum install -y aws-cfn-bootstrap\n",
               "# Install the files and packages from the metadata\n",
               "/opt/aws/bin/cfn-init -v ",
               "         --stack ",
               {
                 "Ref": "AWS::StackName"
               },
               "         --resource WebServerInstance ",
```

```
              "         --configsets All ",
              "         --region ",
              {
                "Ref": "AWS::Region"
              },
              "\n",
              "# Signal the status from cfn-init\n",
              "/opt/aws/bin/cfn-signal -e $? ",
              "         --stack ",
              {
                "Ref": "AWS::StackName"
              },
              "         --resource WebServerInstance ",
              "         --region ",
              {
                "Ref": "AWS::Region"
              },
              "\n"
            ]
          ]
        }
      }
```

YAML

```
      InstanceType: !Ref InstanceType
      ImageId: !FindInMap
        - AWSRegionArch2AMI
        - !Ref 'AWS::Region'
        - !FindInMap
          - AWSInstanceType2Arch
          - !Ref InstanceType
          - Arch
      KeyName: !Ref KeyName
      NetworkInterfaces:
        - GroupSet:
            - !Ref WebServerSecurityGroup
          AssociatePublicIpAddress: 'true'
          DeviceIndex: '0'
          DeleteOnTermination: 'true'
          SubnetId: !Ref PublicSubnet
      UserData: !Base64
        'Fn::Join':
          - ''
          - - |
              #!/bin/bash -xe
            - |
              yum install -y aws-cfn-bootstrap
            - |
              # Install the files and packages from the metadata
            - '/opt/aws/bin/cfn-init -v '
            - '         --stack '
            - !Ref 'AWS::StackName'
            - '         --resource WebServerInstance '
            - '         --configsets All '
            - '         --region '
            - !Ref 'AWS::Region'
            - |+

            - |
              # Signal the status from cfn-init
            - '/opt/aws/bin/cfn-signal -e $? '
            - '         --stack '
            - !Ref 'AWS::StackName'
```

```
            - '         --resource WebServerInstance '
            - '         --region '
            - !Ref 'AWS::Region'
            - |+
```

5. Add the web server configuration metadata to the `WebServerInstance` resource.

   a. Choose the `WebServerInstance` resource, and then choose the **Metadata** tab in the integrated editor pane.

   b. *If you are authoring your template in JSON*: Within the `Metadata` braces (`{}`) and after the `AWS::CloudFormation::Designer` closing brace, add a comma (`,`).

   c. After the `AWS::CloudFormation::Designer` property, add the following snippet, which instructs the cfn-init helper script to start the web server and create a basic web page.

   JSON

```
        "AWS::CloudFormation::Init" : {
          "configSets" : {
            "All" : [ "ConfigureSampleApp" ]
          },
          "ConfigureSampleApp" : {
            "packages" : {
              "yum" : {
                "httpd" : []
              }
            },
            "files" : {
              "/var/www/html/index.html" : {
                "content" : { "Fn::Join" : ["\n", [
                  "<h1>Congratulations, you have successfully launched the AWS
 CloudFormation sample.</h1>"
                ]]},
                "mode"    : "000644",
                "owner"   : "root",
                "group"   : "root"
              }
            },
            "services" : {
              "sysvinit" : {
                "httpd"    : { "enabled" : "true", "ensureRunning" : "true" }
              }
            }
          }
        }
```

   YAML

```
      'AWS::CloudFormation::Init':
        configSets:
          All:
            - ConfigureSampleApp
        ConfigureSampleApp:
          packages:
            yum:
              httpd: []
          files:
            /var/www/html/index.html:
              content: !Join
                - |+

                - - >-
                    <h1>Congratulations, you have successfully launched the AWS
```

```
                    CloudFormation sample.</h1>
              mode: '000644'
              owner: root
              group: root
          services:
            sysvinit:
              httpd:
                enabled: 'true'
                ensureRunning: 'true'
```

6. On the AWS CloudFormation Designer toolbar, choose **Validate template** ( ) to check for syntax errors in your template.

   View and fix errors in the **Messages** pane, and then validate the template again. If you don't see errors, your template is syntactically valid.
7. Save your completed template to keep all the changes you made.

You now have a complete AWS CloudFormation template that you can use to create a basic web server in a VPC. To create the template, we first added and connected template resources by using the AWS CloudFormation Designer canvas pane. Then, we used the integrated editor to add other template components and to specify resource properties. In the next step, we'll use this template to create a stack.

Step 4: Provision Resources

To create a stack, you can launch the AWS CloudFormation Create Stack Wizard from AWS CloudFormation Designer. We'll use the template that we created in the previous steps to create an AWS CloudFormation stack. After AWS CloudFormation provisions all of your resources, you'll have a basic website up and running.

To create the stack

1. On the AWS CloudFormation Designer toolbar, choose **Create Stack** (the cloud icon).

   AWS CloudFormation Designer saves the open template in an S3 bucket, and then launches the AWS CloudFormation Create Stack Wizard. AWS CloudFormation uses the same S3 bucket that it creates whenever you upload templates.
2. AWS CloudFormation automatically populates the template URL; choose **Next**.
3. In the **Stack** section, verify that the **Name** field specifies the stack that you want to update: **`BasicWebServerStack`**.
4. Choose **Next**.

   You can use the currently defined values for the parameters.
5. For this walkthrough, you don't need to add tags or specify advanced settings, so choose **Next**.
6. Ensure that the stack name and Amazon EC2 key-pair name are correct, and then choose **Create**.

It can take several minutes for AWS CloudFormation to create your stack. To monitor progress, view the stack events. For more information about viewing stack events, see Viewing Stack Data and Resources (p. 78). After the stack is created, view the stack outputs and go to the sample website URL to verify that the website running. For more information, see Viewing Stack Data and Resources (p. 78).

Now that you've successfully created a template and launched a stack using AWS CloudFormation Designer, you can use the stack in the following walkthrough: Walkthrough: Use AWS CloudFormation Designer to Modify a Stack's Template (p. 185), which modifies the template to create a scalable web server.

Walkthrough: Use AWS CloudFormation Designer to Modify a Stack's Template

You can use AWS CloudFormation Designer to easily modify a stack's template, and then submit it to AWS CloudFormation to update the stack. Typically, when you modify a stack, you need to get a copy of its template, modify the template in a text editor, and then use AWS CloudFormation to update the stack. With AWS CloudFormation Designer, you can quickly get a copy of any running stack's template, modify it, and then update the stack without ever leaving the console.

In this walkthrough, we'll start with a basic web server (p. 168) stack, and then modify it so that the web server is scalable and durable. By the end of the walkthrough, you'll have a template similar to the following sample: https://console.aws.amazon.com/cloudformation/designer/home?templateUrl=https://s3.amazonaws.com/cloudformation-examples/sample-as-vpc.template&region=us-east-1.

In this walkthrough, we will complete the following steps:

1. Get a stack's template. (p. 185)

   We'll get a copy of a running stack's template; the same basic web server stack in the following walkthrough: Walkthrough: Use AWS CloudFormation Designer to Create a Basic Web Server (p. 168).
2. Modify the template. (p. 186)

   We'll use AWS CloudFormation Designer to modify the stack's template so that your website is scalable and durable by replacing the EC2 instance with an Auto Scaling group and an Elastic Load Balancing load balancer.
3. Update the stack. (p. 194)

   After saving the modifications, we'll update the basic web server stack with the modified template.

   > **Note**
   > AWS CloudFormation is a free service; however, you are charged for the AWS resources you include in your stacks at the current rate for each. For more information about AWS pricing, see the detail page for each product on http://aws.amazon.com.
4. Delete the stack. (p. 194)

   We'll delete the stack to clean up all of the resources.

Prerequisites

This walkthrough assumes that you have a working knowledge of Amazon Virtual Private Cloud (Amazon VPC), Auto Scaling, Elastic Load Balancing, and AWS CloudFormation. For context, each procedure provides some basic information about each resource.

Additionally, the walkthrough assumes that you completed the following walkthrough: Walkthrough: Use AWS CloudFormation Designer to Create a Basic Web Server (p. 168). From that walkthrough, you should have a running stack named `BasicWebServerStack`.

Step 1: Get a Stack Template

In this step, we'll use AWS CloudFormation Designer to get and open a copy of a running stack's template.

To get a copy of a running stack's template

1. Open the AWS CloudFormation console at https://console.aws.amazon.com/cloudformation/.

2. From the list of stacks, select the `BasicWebServerStack`.
3. Choose **Action**, and then **View in Designer**.

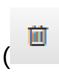
AWS CloudFormation gets a copy of the `BasicWebServerStack` stack's template and displays it in AWS CloudFormation Designer, where you can view the template resources and their relationships. In the following step, we'll use AWS CloudFormation Designer to modify the template.

Step 2: Modify a Template

We'll modify the basic web server template by using AWS CloudFormation Designer's drag-and-drop interface and integrated JSON and YAML editor to replace the single Amazon EC2 instance with an Auto Scaling group and load balancer to make the web site scalable. If traffic to the web site suddenly increases, use Auto Scaling to quickly increase the number of web servers. The load balancer will equally distributes the traffic among the instances.

To modify a stack template

1. Remove the `WebServerInstance` resource.

   a. Right-click the `WebServerInstance` resource.
   b. From the resource menu, choose **Delete** ().
   c. Choose **OK** to confirm.
2. From the **Resources** pane, add the following resources into the `PublicSubnet` resource: **AutoScalingGroup**, **LaunchConfiguration**, and **LoadBalancer**.

   Before adding resources, you might need to expand the subnet to include all resources. The resources are organized by resource categories. The Auto Scaling group and launch configuration are in the **AutoScaling** category, and the load balancer is in the **ElasticLoadBalancing** category.

   Note
   These resources do not follow the container model, so AWS CloudFormation Designer doesn't automatically associate them with the subnet. We'll create connections later on in this step.
3. From the **Resources** pane in the **EC2** category, add the **SecurityGroup** resource anywhere in the VPC except in the subnet.

   This security group will control the inbound and outbound traffic of the load balancer.
4. Rename the resources to make them easier to identify:

   - Rename **AutoScalingGroup** to `WebServerFleet`
   - Rename **LaunchConfiguration** to `WebServerLaunchConfig`
   - Rename **LoadBalancer** to `PublicElasticLoadBalancer`
   - Rename **SecurityGroup** to `PublicLoadBalancerSecurityGroup`
5. Create associations for the resources that you added.

   a. Drag two separate connections from the `PublicElasticLoadBalancer` and `WebServerFleet` resources to the `PublicSubnet` resource.

      These connections associate the load balancer and Auto Scaling group with the public subnet.
   b. Drag a connection from the `PublicElasticLoadBalancer` resource to the `PublicLoadBalancerSecurityGroup` resource to associate the load balancer with its security group.
   c. Drag a connection from the `WebServerFleet` resource to the `PublicElasticLoadBalancer` resource and another connection from the `WebServerFleet` to the `WebServerLaunchConfig`

resource. These connections associate the Auto Scaling group with the load balancer and launch configuration.

d. Drag a connection from the `WebServerLaunchConfig` resource to the `WebServerSecurityGroup` resource to associate the launch configuration with the security group.

e. Drag a depends on connection (`*`) from the `WebServerFleet` resource to the `PublicRoute` resource.

   AWS CloudFormation won't create the `WebServerFleet` resource until the public route is complete. Otherwise, if the public route isn't available when the web server instances are starting up, they won't be able to send signals (using the cfn-signal helper script) to notify AWS CloudFormation when their configurations and application deployments are complete.

6. Specify the properties for the resources that you added.

   a. On the AWS CloudFormation Designer canvas, choose the `PublicElasticLoadBalancer` resource.

   b. In the integrated editor pane, choose the **Properties** tab, and then copy the following snippet and paste it between the **Properties** braces (`{}`).

   AWS CloudFormation Designer automatically added the security group and subnet association, so you need to add only the `Listeners` and `HealthCheck` properties. The `Listeners` property specifies where and what type of traffic to listen for, and the `HealthCheck` property describes the settings for determining the health status of the load balancer.

   JSON

```
          "Listeners": [
            {
              "LoadBalancerPort": "80",
              "InstancePort": "80",
              "Protocol": "HTTP"
            }
          ],
          "HealthCheck": {
            "Target": "HTTP:80/",
            "HealthyThreshold": "3",
            "UnhealthyThreshold": "5",
            "Interval": "90",
            "Timeout": "60"
          },
          "SecurityGroups": [
            {
              "Ref": "PublicLoadBalancerSecurityGroup"
            }
          ],
          "Subnets": [
            {
              "Ref": "PublicSubnet"
            }
          ]
```

   YAML

```
      Listeners:
        - LoadBalancerPort: '80'
          InstancePort: '80'
          Protocol: HTTP
      HealthCheck:
        Target: 'HTTP:80/'
        HealthyThreshold: '3'
        UnhealthyThreshold: '5'
```

```
    Interval: '90'
    Timeout: '60'
  SecurityGroups:
    - !Ref PublicLoadBalancerSecurityGroup
  Subnets:
    - !Ref PublicSubnet
```

c. Repeat this process for the following resources:

`WebServerFleet`

Add the `MaxSize`, `MinSize`, and `DesiredCapacity` properties. These properties specify the maximum and minimum number of instances that you can launch in the Auto Scaling group and the initial number of instances to start with. The desired capacity value refers to a new parameter, which we'll add later in this procedure.

JSON

```
        "MinSize": "1",
        "MaxSize": "10",
        "DesiredCapacity": {
          "Ref": "WebServerCount"
        },
        "VPCZoneIdentifier": [
          {
            "Ref": "PublicSubnet"
          }
        ],
        "LaunchConfigurationName": {
          "Ref": "WebServerLaunchConfig"
        },
        "LoadBalancerNames": [
          {
            "Ref": "PublicElasticLoadBalancer"
          }
        ]
```

YAML

```
      MinSize: '1'
      MaxSize: '10'
      DesiredCapacity: !Ref WebServerCount
      VPCZoneIdentifier:
        - !Ref PublicSubnet
      LaunchConfigurationName: !Ref WebServerLaunchConfig
      LoadBalancerNames:
        - !Ref PublicElasticLoadBalancer
```

`PublicLoadBalancerSecurityGroup`

Add the following inbound and outbound rules that determine the traffic that can reach and leave the load balancer. The rules allows all HTTP traffic to reach and leave the load balancer.

JSON

```
        "GroupDescription": "Public Elastic Load Balancing security group with
HTTP access on port 80 from the Internet",
        "SecurityGroupIngress": [
          {
            "IpProtocol": "tcp",
```

```
            "FromPort": "80",
            "ToPort": "80",
            "CidrIp": "0.0.0.0/0"
          }
        ],
        "SecurityGroupEgress": [
          {
            "IpProtocol": "tcp",
            "FromPort": "80",
            "ToPort": "80",
            "CidrIp": "0.0.0.0/0"
          }
        ],
        "VpcId": {
          "Ref": "VPC"
        }
```

YAML

```
      GroupDescription: >-
        Public Elastic Load Balancing security group with HTTP access on port
80
        from the Internet
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: '80'
          ToPort: '80'
          CidrIp: 0.0.0.0/0
      SecurityGroupEgress:
        - IpProtocol: tcp
          FromPort: '80'
          ToPort: '80'
          CidrIp: 0.0.0.0/0
      VpcId: !Ref VPC
```

`WebServerSecurityGroup`

Modify the HTTP inbound rule to allow only traffic from the load balancer.

JSON

```
        "GroupDescription": "Allow access from load balancer and SSH traffic",
        "SecurityGroupIngress": [
          {
            "IpProtocol": "tcp",
            "FromPort": "80",
            "ToPort": "80",
            "SourceSecurityGroupId": {
              "Ref": "PublicLoadBalancerSecurityGroup"
            }
          },
          {
            "IpProtocol": "tcp",
            "FromPort": "22",
            "ToPort": "22",
            "CidrIp": {
              "Ref": "SSHLocation"
            }
          }
        ],
        "VpcId": {
          "Ref": "VPC"
```

```
        }
```

YAML

```
      VpcId: !Ref VPC
      GroupDescription: Allow access from load balancer and SSH traffic
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: '80'
          ToPort: '80'
          SourceSecurityGroupId: !Ref PublicLoadBalancerSecurityGroup
        - IpProtocol: tcp
          FromPort: '22'
          ToPort: '22'
          CidrIp: !Ref SSHLocation
```

`WebServerLaunchConfig`

The launch configuration has a number of different properties that you need to specify, so we'll highlight just a few of them. The `InstanceType` and `ImageId` properties use the parameter and mapping values that were already specified in the template. You specify the instance type as a parameter value when you create a stack. The `ImageId` value is a mapping that is based on your stack's region and the instance type that you specified.

In the `UserData` property, we specify configurations scripts that run after the instance is up and running. All of the configuration information is defined in the instance's metadata, which we'll add in the next step.

JSON

```
        "InstanceType": {
          "Ref": "InstanceType"
        },
        "ImageId": {
          "Fn::FindInMap": [
            "AWSRegionArch2AMI",
            {
              "Ref": "AWS::Region"
            },
            {
              "Fn::FindInMap": [
                "AWSInstanceType2Arch",
                {
                  "Ref": "InstanceType"
                },
                "Arch"
              ]
            }
          ]
        },
        "KeyName": {
          "Ref": "KeyName"
        },
        "AssociatePublicIpAddress": "true",
        "UserData": {
          "Fn::Base64": {
            "Fn::Join": [
              "",
              [
                "#!/bin/bash -xe\n",
                "yum install -y aws-cfn-bootstrap\n",
                "# Install the files and packages from the metadata\n",
```

```
                "/opt/aws/bin/cfn-init -v ",
                "         --stack ",
                {
                  "Ref": "AWS::StackName"
                },
                "         --resource WebServerLaunchConfig ",
                "         --configsets All ",
                "         --region ",
                {
                  "Ref": "AWS::Region"
                },
                "\n",
                "# Signal the status from cfn-init\n",
                "/opt/aws/bin/cfn-signal -e $? ",
                "         --stack ",
                {
                  "Ref": "AWS::StackName"
                },
                "         --resource WebServerFleet ",
                "         --region ",
                {
                  "Ref": "AWS::Region"
                },
                "\n"
              ]
            ]
          }
        },
        "SecurityGroups": {
          "Ref": "WebServerSecurityGroup"
        }
```

YAML

```
      InstanceType: !Ref InstanceType
      ImageId: !FindInMap
        - AWSRegionArch2AMI
        - !Ref 'AWS::Region'
        - !FindInMap
          - AWSInstanceType2Arch
          - !Ref InstanceType
          - Arch
      KeyName: !Ref KeyName
      AssociatePublicIpAddress: 'true'
      UserData: !Base64
        'Fn::Join':
          - ''
          - - |
              #!/bin/bash -xe
            - |
              yum install -y aws-cfn-bootstrap
            - |
              # Install the files and packages from the metadata
            - '/opt/aws/bin/cfn-init -v '
            - '         --stack '
            - !Ref 'AWS::StackName'
            - '         --resource WebServerLaunchConfig '
            - '         --configsets All '
            - '         --region '
            - !Ref 'AWS::Region'
            - |+

            - |
              # Signal the status from cfn-init
```

```
            - '/opt/aws/bin/cfn-signal -e $? '
            - '         --stack '
            - !Ref 'AWS::StackName'
            - '         --resource WebServerFleet '
            - '         --region '
            - !Ref 'AWS::Region'
            - |+

      SecurityGroups: !Ref WebServerSecurityGroup
```

7. Add the launch configuration metadata to the `WebServerLaunchConfig` resource, which instructs the cfn-init helper script to start the web server and create a basic web page.

   a. Choose the `WebServerLaunchConfig` resource, and then choose the **Metadata** tab in the integrated editor.

   b. *If you are authoring your template in JSON*: Within the `Metadata` braces (`{}`), after the `AWS::CloudFormation::Designer` closing brace, add a comma (`,`).

   c. Add the following snippet, which instructs the cfn-init helper script to start the web server and create a basic web page, after the `AWS::CloudFormation::Designer` property.

   JSON

```
        "AWS::CloudFormation::Init" : {
          "configSets" : {
            "All" : [ "ConfigureSampleApp" ]
          },
          "ConfigureSampleApp" : {
            "packages" : {
              "yum" : {
                "httpd" : []
              }
            },
            "files" : {
              "/var/www/html/index.html" : {
                "content" : { "Fn::Join" : ["\n", [
                  "<h1>Congratulations, you have successfully launched the AWS
CloudFormation sample.</h1>"
                ]]},
                "mode"    : "000644",
                "owner"   : "root",
                "group"   : "root"
              }
            },
            "services" : {
              "sysvinit" : {
                "httpd"    : { "enabled" : "true", "ensureRunning" : "true" }
              }
            }
          }
        }
```

   YAML

```
      'AWS::CloudFormation::Init':
        configSets:
          All:
            - ConfigureSampleApp
        ConfigureSampleApp:
          packages:
            yum:
              httpd: []
          files:
```

```
            /var/www/html/index.html:
              content: !Join
                - |+

                - - >-
                    <h1>Congratulations, you have successfully launched the AWS
                    CloudFormation sample.</h1>
              mode: '000644'
              owner: root
              group: root
          services:
            sysvinit:
              httpd:
                enabled: 'true'
                ensureRunning: 'true'
```

8. Add the `WebServerCount` parameter. This parameter specifies how many instances to create when AWS CloudFormation creates the Auto Scaling group.

   a. Click on an open area on the AWS CloudFormation Designer canvas.

   b. In the integrated editor pane, choose the **Parameters** tab.

   c. Add the following parameter in the integrated editor:

      JSON

```
    "WebServerCount": {
      "Description": "Number of Amazon EC2 instances to launch for the WebServer
 server",
      "Type": "Number",
      "Default": "1"
    }
```

      YAML

```
  WebServerCount:
    Description: Number of Amazon EC2 instances to launch for the WebServer server
    Type: Number
    Default: '1'
```

9. Modify the template output to show the DNS name of the load balancer.

   a. In the integrated editor pane, choose the **Outputs** tab.

   b. Modify the JSON to use the load balancer DNS name, as shown in the following snippet.

      JSON

```
{
  "Outputs": {
    "URL": {
      "Value": {
        "Fn::GetAtt": [
          "PublicElasticLoadBalancer",
          "DNSName"
        ]
      },
      "Description": "Newly created application URL"
    }
  }
}
```

If you're authoring your template in YAML, use the following snippet.

```
Outputs:
  URL:
    Value: !GetAtt
      - PublicElasticLoadBalancer
      - DNSName
    Description: Newly created application URL
```

10. On the AWS CloudFormation Designer toolbar, choose **Validate template** ( ) to check for syntax errors in your template.

    View and fix errors in the **Messages** pane, and then validate the template again. If you don't see errors, your template is syntactically valid.
11. From the AWS CloudFormation Designer toolbar, save the template locally by choosing **File** ( ) and then **Save**.

You now have a modified AWS CloudFormation template that you can use to update the basic web server stack. In the next step, we'll use this template to update the basic web server stack.

Step 3: Update the Stack

To implement your template changes, we need to update the basic web server stack. You can launch the AWS CloudFormation Update Stack Wizard directly from AWS CloudFormation Designer.

To update the stack

1. On the AWS CloudFormation Designer toolbar, choose **Create Stack** ( ).

   AWS CloudFormation Designer saves the opened template in an S3 bucket and then launches the AWS CloudFormation Update Stack Wizard. Because we modified the `BasicWebServerStack` stack's template, AWS CloudFormation launches the Update Stack Wizard for that stack.
2. AWS CloudFormation automatically populates the template URL; choose **Next**.
3. In the **Stack** section, in the **Name** field, verify that the stack name is `BasicWebServerStack`.
4. In the **Parameters** section, use the existing values; choose **Next**.
5. For this walkthrough, you don't need to add tags or specify advanced settings, so choose **Next**.
6. Ensure that the stack name is correct, and then choose **Update**.

It can take several minutes for AWS CloudFormation to update your stack. To monitor progress, view the stack events. For more information, see Viewing Stack Data and Resources (p. 78). After the stack is updated, view the stack outputs and go to the website URL to verify that the website is running. For more information, see Viewing Stack Data and Resources (p. 78). You successfully updated a template and a stack using AWS CloudFormation Designer.

To ensure that you are not charged for unwanted services, you can delete this stack.

Step 4: Clean Up Resources

To make sure you are not charged for unwanted services, delete your stack and it's resources.

To delete the stack

1. From the AWS CloudFormation console, choose the **BasicWebServerStack** stack.

2. Choose **Delete Stack**.
3. In the confirmation message, choose **Yes, Delete**.

It can take several minutes for AWS CloudFormation to delete your stack. To monitor progress, view the stack events. After the stack is deleted, all the resources that you created are deleted. Now that you understand how to use AWS CloudFormation Designer, you can use it to build and modify your own templates.

Walkthrough: Peer with an Amazon VPC in Another AWS Account

You can peer with a virtual private cloud (VPC) in another AWS account by using AWS::EC2::VPCPeeringConnection (p. 649). You need to authorize two separate AWS accounts within a single AWS CloudFormation stack.

For more information about VPC peering and its limitations, see VPC Peering Overview in the *Amazon VPC Peering Guide*.

Prerequisites

1. You need a peer VPC ID, a peer AWS account ID, and a cross-account access role for the peering connection.

   Note
   This walkthrough refers to two accounts: First is an account that allows cross-account peering (the *accepter account*). Second is an account that requests the peering connection (the *requester account*).
2. To accept the VPC peering connection, the cross-account access role must be assumable by you. The resource behaves the same way as a VPC peering connection resource in the same account.

Step 1: Create a VPC and a Cross-Account Role

<title>Create a VPC and a cross-account access role (example)</title>

In this step, you'll create the VPC and role in the *accepter account*.

1. In the AWS Management Console, choose **AWS CloudFormation**.
2. Choose **Create Stack**.
3. You have several options. To use AWS CloudFormation Designer to create a new, blank template, choose **Design template**.

   If you are creating the template in another text editor, choose **Upload a template to Amazon S3** or **Specify an Amazon S3 template URL**, as appropriate.
4. Use the following example template to create the VPC and the cross-account role allowing another account to achieve peering.

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Create a VPC and an assumable role for cross account VPC peering.",
  "Parameters": {
    "PeerRequesterAccountId": {
      "Type": "String"
    }
  },
```

```
  "Resources": {
    "vpc": {
      "Type": "AWS::EC2::VPC",
      "Properties": {
        "CidrBlock": "10.1.0.0/16",
        "EnableDnsSupport": false,
        "EnableDnsHostnames": false,
        "InstanceTenancy": "default"
      }
    },
    "peerRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Statement": [
            {
              "Principal": {
                "AWS": {
                  "Ref": "PeerRequesterAccountId"
                }
              },
              "Action": [
                "sts:AssumeRole"
              ],
              "Effect": "Allow"
            }
          ]
        },
        "Path": "/",
        "Policies": [
          {
            "PolicyName": "root",
            "PolicyDocument": {
              "Version": "2012-10-17",
              "Statement": [
                {
                  "Effect": "Allow",
                  "Action": "ec2:AcceptVpcPeeringConnection",
                  "Resource": "*"
                }
              ]
            }
          }
        ]
      }
    }
  },
  "Outputs": {
    "VPCId": {
      "Value": {
        "Ref": "vpc"
      }
    },
    "RoleARN": {
      "Value": {
        "Fn::GetAtt": [
          "peerRole",
          "Arn"
        ]
      }
    }
  }
}
```

5. Choose **Next**.

6. Give the stack a name (for example, **`VPC-owner`**), and then type the AWS account ID of the *requester account* in the **PeerRequesterAccountId** field.
7. Accept the defaults, and then choose **Next**.
8. Choose **I acknowledge that AWS CloudFormation might create IAM resources**, and then choose **Create**.

Step 2: Create a Template That Includes AWS::EC2::VPCPeeringConnection

Now that you've created the VPC and cross-account role, you can peer with the VPC using another AWS account (the *requester account*).

<title>To create a template that includes the AWS::EC2::VPCPeeringConnection (p. 649) resource (example)</title>

1. Go back to the AWS CloudFormation console home page.
2. Choose **Create Stack**.
3. Choose **Design template** to use AWS CloudFormation Designer to create a new, blank template.

   If you are creating the template in another text editor, choose **Upload a template to Amazon S3** or **Specify an Amazon S3 template URL**, as appropriate.
4. Use the following example template to create a VPC and a VPC peering connection using the peer role you created in Step 1.

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Create a VPC and a VPC Peering connection using the PeerRole to
 accept.",
  "Parameters": {
    "PeerVPCAccountId": {
      "Type": "String"
    },
    "PeerVPCId": {
      "Type": "String"
    },
    "PeerRoleArn": {
      "Type": "String"
    }
  },
  "Resources": {
    "vpc": {
      "Type": "AWS::EC2::VPC",
      "Properties": {
        "CidrBlock": "10.2.0.0/16",
        "EnableDnsSupport": false,
        "EnableDnsHostnames": false,
        "InstanceTenancy": "default"
      }
    },
    "vpcPeeringConnection": {
      "Type": "AWS::EC2::VPCPeeringConnection",
      "Properties": {
        "VpcId": {
          "Ref": "vpc"
        },
        "PeerVpcId": {
          "Ref": "PeerVPCId"
        },
        "PeerOwnerId": {
```

```
          "Ref": "PeerVPCAccountId"
        },
        "PeerRoleArn": {
          "Ref": "PeerRoleArn"
        }
      }
    }
  },
  "Outputs": {
    "VPCId": {
      "Value": {
        "Ref": "vpc"
      }
    },
    "VPCPeeringConnectionId": {
      "Value": {
        "Ref": "vpcPeeringConnection"
      }
    }
  }
}
```

5. Choose **Next**.
6. Give the stack a name (for example, **`VPC-peering-connection`**).
7. Accept the defaults, and then choose **Next**.
8. Choose **I acknowledge that AWS CloudFormation might create IAM resources**, and then choose **Create**.

Creating a Template with a Highly Restrictive Policy

You might want to create a highly restrictive policy for peering your VPC with another AWS account.

The following example template shows how to change the VPC peer owner template (the *accepter account* created in Step 1 above) so that it is more restrictive.

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Description": "Create a VPC and an assumable role for cross account VPC peering.",

    "Parameters": {
        "PeerRequesterAccountId": {
            "Type": "String"
        }
    },
    "Resources": {
        "peerRole": {
            "Properties": {
                "AssumeRolePolicyDocument": {
                    "Statement": [
                        {
                            "Action": [
                                "sts:AssumeRole"
                            ],
                            "Effect": "Allow",
                            "Principal": {
                                "AWS": {
                                    "Ref": "PeerRequesterAccountId"
                                }
                            }
                        }
                    ]
                },
```

```
                    "Path": "/",
                    "Policies": [
                        {
                            "PolicyDocument": {
                                "Statement": [
                                    {
                                        "Action": "ec2:acceptVpcPeeringConnection",
                                        "Effect": "Allow",
                                        "Resource": {
                                            "Fn::Sub": "arn:aws:ec2:${AWS::Region}:
${AWS::AccountId}:vpc/${vpc}"
                                        }
                                    },
                                    {
                                        "Action": "ec2:acceptVpcPeeringConnection",
                                        "Condition": {
                                            "StringEquals": {
                                                "ec2:AccepterVpc": {
                                                    "Fn::Sub": "arn:aws:ec2:${AWS::Region}:
${AWS::AccountId}:vpc/${vpc}"
                                                }
                                            }
                                        },
                                        "Effect": "Allow",
                                        "Resource": {
                                            "Fn::Sub": "arn:aws:ec2:${AWS::Region}:
${AWS::AccountId}:vpc-peering-connection/*"
                                        }
                                    }
                                ],
                                "Version": "2012-10-17"
                            },
                            "PolicyName": "root"
                        }
                    ]
                },
                "Type": "AWS::IAM::Role"
            },
            "vpc": {
                "Properties": {
                    "CidrBlock": "10.1.0.0/16",
                    "EnableDnsHostnames": false,
                    "EnableDnsSupport": false,
                    "InstanceTenancy": "default"
                },
                "Type": "AWS::EC2::VPC"
            }
        },
        "Outputs": {
            "RoleARN": {
                "Value": {
                    "Fn::GetAtt": [
                        "peerRole",
                        "Arn"
                    ]
                }
            },
            "VPCId": {
                "Value": {
                    "Ref": "vpc"
                }
            }
        }
    }
```

To access the VPC, you can use the same requester template as in Step 2 above.

Walkthrough: Refer to Resource Outputs in Another AWS CloudFormation Stack

To export resources from one AWS CloudFormation stack to another, create a cross-stack reference. Cross-stack references let you use a layered or service-oriented architecture. Instead of including all resources in a single stack, you create related AWS resources in separate stacks; then you can refer to required resource outputs from other stacks. By restricting cross-stack references to outputs, you control the parts of a stack that are referenced by other stacks.

For example, you might have a network stack with a VPC, a security group, and a subnet for public web applications, and a separate public web application stack. To ensure that the web applications use the security group and subnet from the network stack, you create a cross-stack reference that allows the web application stack to reference resource outputs from the network stack. With a cross-stack reference, owners of the web application stacks don't need to create or maintain networking rules or assets.

To create a cross-stack reference, use the `Export` output field to flag the value of a resource output for export. Then, use the `Fn::ImportValue` intrinsic function to import the value. For more information, see Outputs (p. 155) and `Fn::ImportValue` (p. 1334).

Prerequisites

Before you begin this walkthrough, check that you have AWS Identity and Access Management (IAM) permissions to use all of the following services: Amazon VPC, Amazon EC2, and AWS CloudFormation.

> **Note**
> AWS CloudFormation is a free service. However, you are charged for the AWS resources that you include in your stacks at the current rate for each one. For more information about AWS pricing, see the detail page for each product.
> The following restrictions apply to cross-stack references:
>
> - For each AWS account, `Export` names must be unique within a region.
> - You can't create cross-stack references across regions. You can use the intrinsic function `Fn::ImportValue` to import only values that have been exported within the same region.
> - For outputs, the value of the `Name` property of an `Export` can't use `Ref` or `GetAtt` functions that depend on a resource.
>
>   Similarly, the `ImportValue` function can't include `Ref` or `GetAtt` functions that depend on a resource.
> - You can't delete a stack if another stack references one of its outputs.
> - You can't modify or remove an output value that is referenced by another stack.

Step 1: Use a Sample Template to Create a Network Stack

The network stack contains the VPC, security group, and subnet that you will use in the web application stack. In addition to these resources, the network stack creates an Internet gateway and routing tables to enable public access.

> **Note**
> You must create this stack before you create the web application stack. If you create the web application stack first, it won't have a security group or subnet.

To create the network stack

1. Open the AWS CloudFormation console and choose **Create Stack**.

2. In the **Select Template** section, choose **Upload a template to Amazon S3**, choose **Specify an Amazon S3 template URL**, and then copy and paste the following URL into the text box: `https://s3.amazonaws.com/cloudformation-examples/user-guide/cross-stack/SampleNetworkCrossStack.template`

   The link provides the location of the network stack template. To see the resources that the stack will create, choose the link, which opens the template. In the outputs section, you can see the networking resources that the sample template exports. The names of the exported resources are prefixed with the stack's name in case you export networking resources from other stacks. When users import networking resources, they can specify from which stack the resources are imported.
3. After reviewing the template, choose **Next**.
4. For **Stack name**, type **`SampleNetworkCrossStack`**, and then choose **Next**.

   Note
   Record the name of this stack. You'll need the stack name when you launch the web application stack.
5. Choose **Next**. For this walkthrough, you don't need to add tags or specify advanced settings.
6. Ensure that the stack name and template URL are correct, and then choose **Create**.

   It might take several minutes for AWS CloudFormation to create your stack. Wait until all resources have been successfully created before proceeding to create the web application stack.
7. To monitor progress, view the stack events. For more information, see Viewing Stack Data and Resources (p. 78).

Step 2: Use a Sample Template to Create a Web Application Stack

The web application stack creates an EC2 instance that uses the security group and subnet from the network stack.

Note
You must create this stack in the same region as the network stack.

To create the web application stack

1. Open the AWS CloudFormation console, and choose **Create Stack**.
2. In the **Select Template** section, choose **Specify an Amazon S3 template URL**, and then copy and paste the following URL into the text box: https://s3.amazonaws.com/cloudformation-examples/user-guide/cross-stack/SampleWebAppCrossStack.template

   The link provides the location of the web application template. To see the resources that the stack will create, choose the link, which will open the template. In the resources section, view the EC2 instance's properties. You can see how the networking resources are imported from another stack by using the `Fn::ImportValue` function.
3. After reviewing the template, choose **Next**.
4. For **Stack name**, type **`SampleWebAppCrossStack`**. In the **Parameters** section, use the default value for the **NetworkStackName** parameter, and then choose **Next**.

   The sample template uses the parameter value to specify from which stack to import values.
5. Choose **Next**. For this walkthrough, you don't need to add tags or specify advanced settings.
6. Ensure that the stack name and template URL are correct, and then choose **Create**.

   It might take several minutes for AWS CloudFormation to create your stack.

7. After the stack has been created, view its resources and note the instance ID. For more information on viewing stack resources, see Viewing Stack Data and Resources (p. 78).

   To verify the instance's security group and subnet, view the instance's properties in the Amazon EC2 console. If the instance uses the security group and subnet from the `SampleNetworkCrossStack` stack, you have successfully created a cross-stack reference.

   Use the console to view the stack outputs and the example website URL to verify that the web application is running. For more information, see Viewing Stack Data and Resources (p. 78).

Step 3: Clean Up Your Resources

To ensure that you are not charged for unwanted services, delete the stacks.

To delete the stacks

1. In the AWS CloudFormation console, choose the SampleWebAppCrossStack stack.
2. Choose **Actions**, and then choose **Delete Stack**.
3. In the confirmation message, choose **Yes, Delete**.
4. After the stack has been deleted, repeat the same steps for the SampleNetworkCrossStack stack.

   > **Note**
   > Wait until AWS CloudFormation completely deletes the SampleWebAppCrossStack stack. If the EC2 instance is still running in the VPC, AWS CloudFormation won't delete the VPC in the SampleNetworkCrossStack stack.

   All of the resources that you have previously created are deleted.

Use the sample templates from this walkthrough to build your own cross-referenced stacks.

Walkthrough: Create a Scalable, Load-balancing Web Server

This template creates a sample web site that uses Auto Scaling and Elastic Load Balancing and is configured to use multiple availability zones. The template also contains CloudWatch alarms that execute Auto Scaling policies to add or remove instances from the Auto Scaling group when the defined thresholds are exceeded.

This template creates one or more Amazon EC2 instances. You will be billed for the AWS resources used if you create a stack from this template.

> **Note**
> The template assumes that your account supports the EC2-VPC platform. In other words, you have a default VPC that allows instances to access the Internet. If you don't have a default VPC, you can create one. For more information, see Amazon EC2 and Amazon Virtual Private Cloud in the *Amazon EC2 User Guide for Linux Instances*.

You can get the latest version of this sample template at https://s3.amazonaws.com/cloudformation-templates-us-east-1/AutoScalingMultiAZWithNotifications.template.

Auto Scaling Multi-AZ Template

```
{
```

```
  "AWSTemplateFormatVersion" : "2010-09-09",

  "Description" : "AWS CloudFormation Sample Template AutoScalingMultiAZWithNotifications:
 Create a multi-az, load balanced and Auto Scaled sample web site running on an Apache Web
 Serever. The application is configured to span all Availability Zones in the region and is
 Auto-Scaled based on the CPU utilization of the web servers. Notifications will be sent
 to the operator email address on scaling events. The instances are load balanced with a
 simple health check against the default web page. **WARNING** This template creates one
 or more Amazon EC2 instances and an Elastic Load Balancer. You will be billed for the AWS
 resources used if you create a stack from this template.",

  "Parameters" : {
    "InstanceType" : {
      "Description" : "WebServer EC2 instance type",
      "Type" : "String",
      "Default" : "t2.small",
      "AllowedValues" : [ "t1.micro", "t2.nano", "t2.micro", "t2.small", "t2.medium",
 "t2.large", "m1.small", "m1.medium", "m1.large", "m1.xlarge", "m2.xlarge", "m2.2xlarge",
 "m2.4xlarge", "m3.medium", "m3.large", "m3.xlarge", "m3.2xlarge", "m4.large", "m4.xlarge",
 "m4.2xlarge", "m4.4xlarge", "m4.10xlarge", "c1.medium", "c1.xlarge", "c3.large",
 "c3.xlarge", "c3.2xlarge", "c3.4xlarge", "c3.8xlarge", "c4.large", "c4.xlarge",
 "c4.2xlarge", "c4.4xlarge", "c4.8xlarge", "g2.2xlarge", "g2.8xlarge", "r3.large",
 "r3.xlarge", "r3.2xlarge", "r3.4xlarge", "r3.8xlarge", "i2.xlarge", "i2.2xlarge",
 "i2.4xlarge", "i2.8xlarge", "d2.xlarge", "d2.2xlarge", "d2.4xlarge", "d2.8xlarge",
 "hi1.4xlarge", "hs1.8xlarge", "cr1.8xlarge", "cc2.8xlarge", "cg1.4xlarge"]
,
      "ConstraintDescription" : "must be a valid EC2 instance type."
    },

    "OperatorEMail": {
      "Description": "EMail address to notify if there are any scaling operations",
      "Type": "String",
      "AllowedPattern": "([a-zA-Z0-9_\\-\\.]+)@((\\[[0-9]{1,3}\\.[0-9]{1,3}\\.[0-9]{1,3}\
\.)|(([a-zA-Z0-9\\-]+\\.)+))([a-zA-Z]{2,4}|[0-9]{1,3})(\\]?)",
      "ConstraintDescription": "must be a valid email address."
    },

    "KeyName" : {
      "Description" : "The EC2 Key Pair to allow SSH access to the instances",
      "Type" : "AWS::EC2::KeyPair::KeyName",
      "ConstraintDescription" : "must be the name of an existing EC2 KeyPair."
    },

    "SSHLocation" : {
      "Description" : "The IP address range that can be used to SSH to the EC2 instances",
      "Type": "String",
      "MinLength": "9",
      "MaxLength": "18",
      "Default": "0.0.0.0/0",
      "AllowedPattern": "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\\d{1,2})",
      "ConstraintDescription": "must be a valid IP CIDR range of the form x.x.x.x/x."
    }
  },

  "Mappings" : {
    "Region2Examples" : {
      "us-east-1"      : { "Examples" : "https://s3.amazonaws.com/cloudformation-examples-
us-east-1" },
      "us-west-2"      : { "Examples" : "https://s3-us-west-2.amazonaws.com/cloudformation-
examples-us-west-2" },
      "us-west-1"      : { "Examples" : "https://s3-us-west-1.amazonaws.com/cloudformation-
examples-us-west-1" },
      "eu-west-1"      : { "Examples" : "https://s3-eu-west-1.amazonaws.com/cloudformation-
examples-eu-west-1" },
      "eu-west-2"      : { "Examples" : "https://s3-eu-west-2.amazonaws.com/cloudformation-
examples-eu-west-2" },
```

```
      "eu-central-1"   : { "Examples" : "https://s3-eu-central-1.amazonaws.com/
cloudformation-examples-eu-central-1" },
      "ap-southeast-1" : { "Examples" : "https://s3-ap-southeast-1.amazonaws.com/
cloudformation-examples-ap-southeast-1" },
      "ap-northeast-1" : { "Examples" : "https://s3-ap-northeast-1.amazonaws.com/
cloudformation-examples-ap-northeast-1" },
      "ap-northeast-2" : { "Examples" : "https://s3-ap-northeast-2.amazonaws.com/
cloudformation-examples-ap-northeast-2" },
      "ap-southeast-2" : { "Examples" : "https://s3-ap-southeast-2.amazonaws.com/
cloudformation-examples-ap-southeast-2" },
      "ap-south-1"     : { "Examples" : "https://s3-ap-south-1.amazonaws.com/
cloudformation-examples-ap-south-1" },
      "us-east-2"      : { "Examples" : "https://s3-us-east-2.amazonaws.com/cloudformation-
examples-us-east-2" },
      "ca-central-1"   : { "Examples" : "https://s3-ca-central-1.amazonaws.com/
cloudformation-examples-ca-central-1" },
      "sa-east-1"      : { "Examples" : "https://s3-sa-east-1.amazonaws.com/cloudformation-
examples-sa-east-1" },
      "cn-north-1"     : { "Examples" : "https://s3.cn-north-1.amazonaws.com.cn/
cloudformation-examples-cn-north-1" }
    }
,
    "AWSInstanceType2Arch" : {
      "t1.micro"    : { "Arch" : "PV64"   },
      "t2.nano"     : { "Arch" : "HVM64"  },
      "t2.micro"    : { "Arch" : "HVM64"  },
      "t2.small"    : { "Arch" : "HVM64"  },
      "t2.medium"   : { "Arch" : "HVM64"  },
      "t2.large"    : { "Arch" : "HVM64"  },
      "m1.small"    : { "Arch" : "PV64"   },
      "m1.medium"   : { "Arch" : "PV64"   },
      "m1.large"    : { "Arch" : "PV64"   },
      "m1.xlarge"   : { "Arch" : "PV64"   },
      "m2.xlarge"   : { "Arch" : "PV64"   },
      "m2.2xlarge"  : { "Arch" : "PV64"   },
      "m2.4xlarge"  : { "Arch" : "PV64"   },
      "m3.medium"   : { "Arch" : "HVM64"  },
      "m3.large"    : { "Arch" : "HVM64"  },
      "m3.xlarge"   : { "Arch" : "HVM64"  },
      "m3.2xlarge"  : { "Arch" : "HVM64"  },
      "m4.large"    : { "Arch" : "HVM64"  },
      "m4.xlarge"   : { "Arch" : "HVM64"  },
      "m4.2xlarge"  : { "Arch" : "HVM64"  },
      "m4.4xlarge"  : { "Arch" : "HVM64"  },
      "m4.10xlarge" : { "Arch" : "HVM64"  },
      "c1.medium"   : { "Arch" : "PV64"   },
      "c1.xlarge"   : { "Arch" : "PV64"   },
      "c3.large"    : { "Arch" : "HVM64"  },
      "c3.xlarge"   : { "Arch" : "HVM64"  },
      "c3.2xlarge"  : { "Arch" : "HVM64"  },
      "c3.4xlarge"  : { "Arch" : "HVM64"  },
      "c3.8xlarge"  : { "Arch" : "HVM64"  },
      "c4.large"    : { "Arch" : "HVM64"  },
      "c4.xlarge"   : { "Arch" : "HVM64"  },
      "c4.2xlarge"  : { "Arch" : "HVM64"  },
      "c4.4xlarge"  : { "Arch" : "HVM64"  },
      "c4.8xlarge"  : { "Arch" : "HVM64"  },
      "g2.2xlarge"  : { "Arch" : "HVMG2"  },
      "g2.8xlarge"  : { "Arch" : "HVMG2"  },
      "r3.large"    : { "Arch" : "HVM64"  },
      "r3.xlarge"   : { "Arch" : "HVM64"  },
      "r3.2xlarge"  : { "Arch" : "HVM64"  },
      "r3.4xlarge"  : { "Arch" : "HVM64"  },
      "r3.8xlarge"  : { "Arch" : "HVM64"  },
      "i2.xlarge"   : { "Arch" : "HVM64"  },
      "i2.2xlarge"  : { "Arch" : "HVM64"  },
```

```
      "i2.4xlarge"  : { "Arch" : "HVM64"  },
      "i2.8xlarge"  : { "Arch" : "HVM64"  },
      "d2.xlarge"   : { "Arch" : "HVM64"  },
      "d2.2xlarge"  : { "Arch" : "HVM64"  },
      "d2.4xlarge"  : { "Arch" : "HVM64"  },
      "d2.8xlarge"  : { "Arch" : "HVM64"  },
      "hi1.4xlarge" : { "Arch" : "HVM64"  },
      "hs1.8xlarge" : { "Arch" : "HVM64"  },
      "cr1.8xlarge" : { "Arch" : "HVM64"  },
      "cc2.8xlarge" : { "Arch" : "HVM64"  }
    },

    "AWSInstanceType2NATArch" : {
      "t1.micro"    : { "Arch" : "NATPV64"   },
      "t2.nano"     : { "Arch" : "NATHVM64"  },
      "t2.micro"    : { "Arch" : "NATHVM64"  },
      "t2.small"    : { "Arch" : "NATHVM64"  },
      "t2.medium"   : { "Arch" : "NATHVM64"  },
      "t2.large"    : { "Arch" : "NATHVM64"  },
      "m1.small"    : { "Arch" : "NATPV64"   },
      "m1.medium"   : { "Arch" : "NATPV64"   },
      "m1.large"    : { "Arch" : "NATPV64"   },
      "m1.xlarge"   : { "Arch" : "NATPV64"   },
      "m2.xlarge"   : { "Arch" : "NATPV64"   },
      "m2.2xlarge"  : { "Arch" : "NATPV64"   },
      "m2.4xlarge"  : { "Arch" : "NATPV64"   },
      "m3.medium"   : { "Arch" : "NATHVM64"  },
      "m3.large"    : { "Arch" : "NATHVM64"  },
      "m3.xlarge"   : { "Arch" : "NATHVM64"  },
      "m3.2xlarge"  : { "Arch" : "NATHVM64"  },
      "m4.large"    : { "Arch" : "NATHVM64"  },
      "m4.xlarge"   : { "Arch" : "NATHVM64"  },
      "m4.2xlarge"  : { "Arch" : "NATHVM64"  },
      "m4.4xlarge"  : { "Arch" : "NATHVM64"  },
      "m4.10xlarge" : { "Arch" : "NATHVM64"  },
      "c1.medium"   : { "Arch" : "NATPV64"   },
      "c1.xlarge"   : { "Arch" : "NATPV64"   },
      "c3.large"    : { "Arch" : "NATHVM64"  },
      "c3.xlarge"   : { "Arch" : "NATHVM64"  },
      "c3.2xlarge"  : { "Arch" : "NATHVM64"  },
      "c3.4xlarge"  : { "Arch" : "NATHVM64"  },
      "c3.8xlarge"  : { "Arch" : "NATHVM64"  },
      "c4.large"    : { "Arch" : "NATHVM64"  },
      "c4.xlarge"   : { "Arch" : "NATHVM64"  },
      "c4.2xlarge"  : { "Arch" : "NATHVM64"  },
      "c4.4xlarge"  : { "Arch" : "NATHVM64"  },
      "c4.8xlarge"  : { "Arch" : "NATHVM64"  },
      "g2.2xlarge"  : { "Arch" : "NATHVMG2"  },
      "g2.8xlarge"  : { "Arch" : "NATHVMG2"  },
      "r3.large"    : { "Arch" : "NATHVM64"  },
      "r3.xlarge"   : { "Arch" : "NATHVM64"  },
      "r3.2xlarge"  : { "Arch" : "NATHVM64"  },
      "r3.4xlarge"  : { "Arch" : "NATHVM64"  },
      "r3.8xlarge"  : { "Arch" : "NATHVM64"  },
      "i2.xlarge"   : { "Arch" : "NATHVM64"  },
      "i2.2xlarge"  : { "Arch" : "NATHVM64"  },
      "i2.4xlarge"  : { "Arch" : "NATHVM64"  },
      "i2.8xlarge"  : { "Arch" : "NATHVM64"  },
      "d2.xlarge"   : { "Arch" : "NATHVM64"  },
      "d2.2xlarge"  : { "Arch" : "NATHVM64"  },
      "d2.4xlarge"  : { "Arch" : "NATHVM64"  },
      "d2.8xlarge"  : { "Arch" : "NATHVM64"  },
      "hi1.4xlarge" : { "Arch" : "NATHVM64"  },
      "hs1.8xlarge" : { "Arch" : "NATHVM64"  },
      "cr1.8xlarge" : { "Arch" : "NATHVM64"  },
      "cc2.8xlarge" : { "Arch" : "NATHVM64"  }
```

```
    }
,
    "AWSRegionArch2AMI" : {
      "us-east-1"        : {"PV64" : "ami-2a69aa47", "HVM64" : "ami-6869aa05", "HVMG2" :
 "ami-a41a3fb3"},
      "us-west-2"        : {"PV64" : "ami-7f77b31f", "HVM64" : "ami-7172b611", "HVMG2" :
 "ami-caf253aa"},
      "us-west-1"        : {"PV64" : "ami-a2490dc2", "HVM64" : "ami-31490d51", "HVMG2" :
 "ami-00347e60"},
      "eu-west-1"        : {"PV64" : "ami-4cdd453f", "HVM64" : "ami-f9dd458a", "HVMG2" :
 "ami-e2f7bd91"},
      "eu-west-2"        : {"PV64" : "NOT_SUPPORTED", "HVM64" : "ami-886369ec", "HVMG2" :
 "NOT_SUPPORTED"},
      "eu-central-1"     : {"PV64" : "ami-6527cf0a", "HVM64" : "ami-ea26ce85", "HVMG2" :
 "ami-d2ff04bd"},
      "ap-northeast-1"   : {"PV64" : "ami-3e42b65f", "HVM64" : "ami-374db956", "HVMG2" :
 "ami-4c78d52d"},
      "ap-northeast-2"   : {"PV64" : "NOT_SUPPORTED", "HVM64" : "ami-2b408b45", "HVMG2" :
 "NOT_SUPPORTED"},
      "ap-southeast-1"   : {"PV64" : "ami-df9e4cbc", "HVM64" : "ami-a59b49c6", "HVMG2" :
 "ami-f3f95990"},
      "ap-southeast-2"   : {"PV64" : "ami-63351d00", "HVM64" : "ami-dc361ebf", "HVMG2" :
 "ami-3a122e59"},
      "ap-south-1"       : {"PV64" : "NOT_SUPPORTED", "HVM64" : "ami-ffbdd790", "HVMG2" :
 "ami-21a7d34e"},
      "us-east-2"        : {"PV64" : "NOT_SUPPORTED", "HVM64" : "ami-f6035893", "HVMG2" :
 "NOT_SUPPORTED"},
      "ca-central-1"     : {"PV64" : "NOT_SUPPORTED", "HVM64" : "ami-730ebd17", "HVMG2" :
 "NOT_SUPPORTED"},
      "sa-east-1"        : {"PV64" : "ami-1ad34676", "HVM64" : "ami-6dd04501", "HVMG2" :
 "NOT_SUPPORTED"},
      "cn-north-1"       : {"PV64" : "ami-77559f1a", "HVM64" : "ami-8e6aa0e3", "HVMG2" :
 "NOT_SUPPORTED"}
    }

  },

  "Resources" : {
    "NotificationTopic": {
      "Type": "AWS::SNS::Topic",
      "Properties": {
        "Subscription": [ { "Endpoint": { "Ref": "OperatorEMail" }, "Protocol": "email" } ]
      }
    },

    "WebServerGroup" : {
      "Type" : "AWS::AutoScaling::AutoScalingGroup",
      "Properties" : {
        "AvailabilityZones" : { "Fn::GetAZs" : ""},
        "LaunchConfigurationName" : { "Ref" : "LaunchConfig" },
        "MinSize" : "1",
        "MaxSize" : "3",
        "LoadBalancerNames" : [ { "Ref" : "ElasticLoadBalancer" } ],
        "NotificationConfiguration" : {
          "TopicARN" : { "Ref" : "NotificationTopic" },
          "NotificationTypes" : [ "autoscaling:EC2_INSTANCE_LAUNCH",
                                  "autoscaling:EC2_INSTANCE_LAUNCH_ERROR",
                                  "autoscaling:EC2_INSTANCE_TERMINATE",
                                  "autoscaling:EC2_INSTANCE_TERMINATE_ERROR"]
        }
      },
      "CreationPolicy" : {
        "ResourceSignal" : {
          "Timeout" : "PT15M",
          "Count"   : "1"
        }
```

```
      },
      "UpdatePolicy": {
        "AutoScalingRollingUpdate": {
          "MinInstancesInService": "1",
          "MaxBatchSize": "1",
          "PauseTime" : "PT15M",
          "WaitOnResourceSignals": "true"
        }
      }
    },

    "LaunchConfig" : {
      "Type" : "AWS::AutoScaling::LaunchConfiguration",
      "Metadata" : {
        "Comment" : "Install a simple application",
        "AWS::CloudFormation::Init" : {
          "config" : {
            "packages" : {
              "yum" : {
                "httpd" : []
              }
            },

            "files" : {
              "/var/www/html/index.html" : {
                "content" : { "Fn::Join" : ["\n", [
                  "<img src=\"", {"Fn::FindInMap" : ["Region2Examples", {"Ref" :
 "AWS::Region"}, "Examples"]}, "/cloudformation_graphic.png\" alt=\"AWS CloudFormation Logo
\"/>",
                  "<h1>Congratulations, you have successfully launched the AWS
 CloudFormation sample.</h1>"
                ]]},
                "mode"    : "000644",
                "owner"   : "root",
                "group"   : "root"
              },

              "/etc/cfn/cfn-hup.conf" : {
                "content" : { "Fn::Join" : ["", [
                  "[main]\n",
                  "stack=", { "Ref" : "AWS::StackId" }, "\n",
                  "region=", { "Ref" : "AWS::Region" }, "\n"
                ]]},
                "mode"    : "000400",
                "owner"   : "root",
                "group"   : "root"
              },

              "/etc/cfn/hooks.d/cfn-auto-reloader.conf" : {
                "content": { "Fn::Join" : ["", [
                  "[cfn-auto-reloader-hook]\n",
                  "triggers=post.update\n",
                  "path=Resources.LaunchConfig.Metadata.AWS::CloudFormation::Init\n",
                  "action=/opt/aws/bin/cfn-init -v ",
                  "         --stack ", { "Ref" : "AWS::StackName" },
                  "         --resource LaunchConfig ",
                  "         --region ", { "Ref" : "AWS::Region" }, "\n",
                  "runas=root\n"
                ]]}
              }
            },

            "services" : {
              "sysvinit" : {
                "httpd"    : { "enabled" : "true", "ensureRunning" : "true" },
                "cfn-hup" : { "enabled" : "true", "ensureRunning" : "true",
```

```
                              "files" : ["/etc/cfn/cfn-hup.conf", "/etc/cfn/hooks.d/cfn-
auto-reloader.conf"]}
              }
            }
          }
        }
      },
      "Properties" : {
        "KeyName" : { "Ref" : "KeyName" },
        "ImageId" : { "Fn::FindInMap" : [ "AWSRegionArch2AMI", { "Ref" : "AWS::Region" },
                                          { "Fn::FindInMap" : [ "AWSInstanceType2Arch",
 { "Ref" : "InstanceType" }, "Arch" ] } ] },
        "SecurityGroups" : [ { "Ref" : "InstanceSecurityGroup" } ],
        "InstanceType" : { "Ref" : "InstanceType" },
        "UserData"       : { "Fn::Base64" : { "Fn::Join" : ["", [
             "#!/bin/bash -xe\n",
             "yum update -y aws-cfn-bootstrap\n",

             "/opt/aws/bin/cfn-init -v ",
             "         --stack ", { "Ref" : "AWS::StackName" },
             "         --resource LaunchConfig ",
             "         --region ", { "Ref" : "AWS::Region" }, "\n",

             "/opt/aws/bin/cfn-signal -e $? ",
             "         --stack ", { "Ref" : "AWS::StackName" },
             "         --resource WebServerGroup ",
             "         --region ", { "Ref" : "AWS::Region" }, "\n"
        ]]}}
      }
    },

    "WebServerScaleUpPolicy" : {
      "Type" : "AWS::AutoScaling::ScalingPolicy",
      "Properties" : {
        "AdjustmentType" : "ChangeInCapacity",
        "AutoScalingGroupName" : { "Ref" : "WebServerGroup" },
        "Cooldown" : "60",
        "ScalingAdjustment" : "1"
      }
    },
    "WebServerScaleDownPolicy" : {
      "Type" : "AWS::AutoScaling::ScalingPolicy",
      "Properties" : {
        "AdjustmentType" : "ChangeInCapacity",
        "AutoScalingGroupName" : { "Ref" : "WebServerGroup" },
        "Cooldown" : "60",
        "ScalingAdjustment" : "-1"
      }
    },

    "CPUAlarmHigh": {
     "Type": "AWS::CloudWatch::Alarm",
     "Properties": {
        "AlarmDescription": "Scale-up if CPU > 90% for 10 minutes",
        "MetricName": "CPUUtilization",
        "Namespace": "AWS/EC2",
        "Statistic": "Average",
        "Period": "300",
        "EvaluationPeriods": "2",
        "Threshold": "90",
        "AlarmActions": [ { "Ref": "WebServerScaleUpPolicy" } ],
        "Dimensions": [
          {
            "Name": "AutoScalingGroupName",
            "Value": { "Ref": "WebServerGroup" }
          }
```

```
        ],
        "ComparisonOperator": "GreaterThanThreshold"
      }
    },
    "CPUAlarmLow": {
     "Type": "AWS::CloudWatch::Alarm",
     "Properties": {
        "AlarmDescription": "Scale-down if CPU < 70% for 10 minutes",
        "MetricName": "CPUUtilization",
        "Namespace": "AWS/EC2",
        "Statistic": "Average",
        "Period": "300",
        "EvaluationPeriods": "2",
        "Threshold": "70",
        "AlarmActions": [ { "Ref": "WebServerScaleDownPolicy" } ],
        "Dimensions": [
          {
            "Name": "AutoScalingGroupName",
            "Value": { "Ref": "WebServerGroup" }
          }
        ],
        "ComparisonOperator": "LessThanThreshold"
      }
    },

    "ElasticLoadBalancer" : {
      "Type" : "AWS::ElasticLoadBalancing::LoadBalancer",
      "Properties" : {
        "AvailabilityZones" : { "Fn::GetAZs" : "" },
        "CrossZone" : "true",
        "Listeners" : [ {
          "LoadBalancerPort" : "80",
          "InstancePort" : "80",
          "Protocol" : "HTTP"
        } ],
        "HealthCheck" : {
          "Target" : "HTTP:80/",
          "HealthyThreshold" : "3",
          "UnhealthyThreshold" : "5",
          "Interval" : "30",
          "Timeout" : "5"
        }
      }
    },

    "InstanceSecurityGroup" : {
      "Type" : "AWS::EC2::SecurityGroup",
      "Properties" : {
        "GroupDescription" : "Enable SSH access and HTTP from the load balancer only",
        "SecurityGroupIngress" : [ {
          "IpProtocol" : "tcp",
          "FromPort" : "22",
          "ToPort" : "22",
          "CidrIp" : { "Ref" : "SSHLocation"}
        },
        {
          "IpProtocol" : "tcp",
          "FromPort" : "80",
          "ToPort" : "80",
          "SourceSecurityGroupOwnerId" : {"Fn::GetAtt" : ["ElasticLoadBalancer",
"SourceSecurityGroup.OwnerAlias"]},
          "SourceSecurityGroupName" : {"Fn::GetAtt" : ["ElasticLoadBalancer",
"SourceSecurityGroup.GroupName"]}
        } ]
      }
    }
```

```
  },

  "Outputs" : {
    "URL" : {
      "Description" : "The URL of the website",
      "Value" :  { "Fn::Join" : [ "", [ "http://", { "Fn::GetAtt" :
 [ "ElasticLoadBalancer", "DNSName" ]}]]}
    }
  }
}
```

Template Walkthrough

The example template contains an Auto Scaling group with a LoadBalancer, a security group that defines ingress rules, CloudWatch alarms, and Auto Scaling policies.

The template has three input parameters: InstanceType is the type of EC2 instance to use for the Auto Scaling group and has a default of m1.small; WebServerPort is the TCP port for the web server and has a default of 8888; KeyName is the name of an EC2 key pair to be used for the Auto Scaling group. KeyName must be specified at stack creation (parameters with no default value must be specified at stack creation).

The AWS::AutoScaling::AutoScalingGroup (p. 433) resource WebServerGroup declares the following Auto Scaling group configuration:

- *AvailabilityZones* specifies the availability zones where the auto scaling group's EC2 instances will be created. The Fn::GetAZs (p. 1332) function call `{ "Fn::GetAZs" : "" }` specifies all availability zones for the region in which the stack is created.
- *MinSize* and *MaxSize* set the minimum and maximum number of EC2 instances in the Auto Scaling group.
- *LoadBalancerNames* lists the LoadBalancers used to route traffic to the Auto Scaling group. The LoadBalancer for this group is the ElasticLoadBalancer resource.

The AWS::AutoScaling::LaunchConfiguration (p. 440) resource LaunchConfig declares the following configurations to use for the EC2 instances in the WebServerGroup Auto Scaling group:

- *KeyName* takes the value of the KeyName input parameter as the EC2 key pair to use.
- *UserData* is the Base64 encoded value of the WebServerPort parameter, which is passed to an application .
- *SecurityGroups* is a list of EC2 security groups that contain the firewall ingress rules for EC2 instances in the Auto Scaling group. In this example, there is only one security group and it is declared as a AWS::EC2::SecurityGroup (p. 608) resource: InstanceSecurityGroup. This security group contains two ingress rules: 1) a TCP ingress rule that allows access from all IP addresses ("CidrIp" : "0.0.0.0/0") for port 22 (for SSH access) and 2) a TCP ingress rule that allows access from the ElasticLoadBalancer resource for the WebServerPort port by specifying the LoadBalancer's source security group. The GetAtt (p. 1324) function is used to get the SourceSecurityGroup.OwnerAlias and SourceSecurityGroup.GroupName properties from the ElasticLoadBalancer resource. For more information about the Elastic Load Balancing security groups, see Manage Security Groups in Amazon EC2-Classic or Manage Security Groups in Amazon VPC.
- *ImageId* is the evaluated value of a set of nested maps. We added the maps so that the template contained the logic for choosing the right image ID. That logic is based on the instance type that was specified with the InstanceType parameter (AWSInstanceType2Arch maps the instance type to an architecture 32 or 64) and the region where the stack is created (AWSRegionArch2AMI maps the region and architecture to a image ID):

```
{ "Fn::FindInMap" : [ "AWSRegionArch2AMI",
   { "Ref" : "AWS::Region" },
```

```
    { "Fn::FindInMap" : [ "AWSInstanceType2Arch",
       { "Ref" : "InstanceType" },
       "Arch" ]
    }
    ]}
```

For example, if you use this template to create a stack in the us-east-1 region and specify m1.small as InstanceType, AWS CloudFormation would evaluate the inner map for AWSInstanceType2Arch as the following:

```
{ "Fn::FindInMap" : [ "AWSInstanceType2Arch", "m1.small", "Arch" ] }
```

In the AWSInstanceType2Arch mapping, the Arch value for the m1.small key maps to 32, which is used as the value for the outer map. The key is the evaluated result of the AWS::Region pseudo parameter which is the region where the stack is being created. For this example, AWS::Region is us-east-1; therefore, the outer map is evaluated as follows:

```
Fn::FindInMap" : [ "AWSRegionArch2AMI", "us-east-1", "32"]
```

In the AWSRegionArch2AMI mapping, the value 32 for the key us-east-1 maps to ami-6411e20d. This means that ImageId would be ami-6411e20d.

The AWS::ElasticLoadBalancing::LoadBalancer (p. 719) resource ElasticLoadBalancer declares the following LoadBalancer configuration:

- *AvailabilityZones* is a list of availability zones where the LoadBalancer will distribute traffic. In this example, the Fn::GetAZs function call `{ "Fn::GetAZs" : "" }` specifies all availability zones for the region in which the stack is created.
- *Listeners* is a list of load balancing routing configurations that specify the port that the LoadBalancer accepts requests, the port on the registered EC2 instances where the LoadBalancer forwards requests, and the protocol used to route requests.
- *HealthCheck* is the configuration that Elastic Load Balancing uses to check the health of the EC2 instances that the LoadBalancer routes traffic to. In this example, the HealthCheck targets the root address of the EC2 instances using the port specified by WebServerPort over the HTTP protocol. If the WebServerPort is 8888, the `{ "Fn::Join" : [ "", ["HTTP:", { "Ref" : "WebServerPort" }, "/"]]}` function call is evaluated as the string `HTTP:8888/`. It also specifies that the EC2 instances have an interval of 30 seconds between health checks (Interval). The Timeout is defined as the length of time Elastic Load Balancing waits for a response from the health check target (5 seconds in this example). After the Timeout period lapses, Elastic Load Balancing marks that EC2 instance's health check as unhealthy. When an EC2 instance fails 5 consecutive health checks (UnhealthyThreshold), Elastic Load Balancing stops routing traffic to that EC2 instance until that instance has 3 consecutive healthy health checks at which point Elastic Load Balancing considers the EC2 instance healthy and begins routing traffic to that instance again.

The AWS::AutoScaling::ScalingPolicy (p. 452) resource WebServerScaleUpPolicy is an Auto Scaling policy that scales up the Auto Scaling group WebServerGroup. The `AdjustmentType` property is set to ChangeInCapacity. This means that the `ScalingAdjustment` represents the number of instances to add (if `ScalingAdjustment` is positive, instances are added; if negative, instances are deleted). In this example, `ScalingAdjustment` is 1; therefore, the policy increments the number of EC2 instances in the group by 1 when the policy is executed. The Cooldown property specifies that Auto Scaling waits 60 seconds before starting any other policy or trigger related actions.

The AWS::CloudWatch::Alarm (p. 498) resource CPUAlarmHigh specifies the scaling policy WebServerScaleUpPolicy as the action to execute when the alarm is in an `ALARM` state (AlarmActions). The alarm monitors the EC2 instances in the WebServerGroup Auto Scaling group (Dimensions). The alarm measures the average (Statistic) EC2 instance CPU utilization (Namespace and MetricName) of the instances in the WebServerGroup (Dimensions) over a 300 second interval (Period). When this value (average CPU utilization over 300 seconds) remains greater than 90 percent (ComparisonOperator and Threshold) for 2 consecutive periods (EvaluationPeriod), the alarm will go into an `ALARM` state and CloudWatch will execute the WebServerScaleUpPolicy policy (AlarmActions) described above scale up the WebServerGroup.

The CPUAlarmLow alarm measures the same metrics but has an alarm that triggers when CPU utilization is less than 75 percent (ComparisonOperator and Threshold) and executes the WebServerScaleDownPolicy policy to remove 1 EC2 instance from the Auto Scaling group WebServerGroup.

Deploying Applications on Amazon EC2 with AWS CloudFormation

You can use AWS CloudFormation to automatically install, configure, and start applications on Amazon EC2 instances. Doing so enables you to easily duplicate deployments and update existing installations without connecting directly to the instance, which can save you a lot of time and effort.

AWS CloudFormation includes a set of helper scripts (cfn-init, cfn-signal, cfn-get-metadata, and cfn-hup) that are based on cloud-init. You call these helper scripts from your AWS CloudFormation templates to install, configure, and update applications on Amazon EC2 instances that are in the same template.

The following walkthrough describes how to create a template that launches a LAMP stack by using cfn helper scripts to install, configure and start Apache, MySQL, and PHP. You'll start with a simple template that sets up a basic Amazon EC2 instance running Amazon Linux, and then continue adding to the template until it describes a full LAMP stack.

For additional strategies and examples about deploying applications with AWS CloudFormation, see the Bootstrapping Applications via AWS CloudFormation article.

Topics

Basic Amazon EC2 Instance

You start with a basic template that defines a single Amazon EC2 instance with a security group that allows SSH traffic on port 22 and HTTP traffic on port 80, as shown in the following example:

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",

  "Description" : "AWS CloudFormation sample template LAMP_Single_Instance: Create a LAMP
 stack using a single EC2
instance and a local MySQL database for storage. This template demonstrates using the AWS
 CloudFormation bootstrap
scripts to install the packages and files necessary to deploy the Apache web server, PHP,
 and MySQL at instance launch time.
**WARNING** This template creates an Amazon EC2 instance. You will be billed for the AWS
 resources used if you create a stack from this template.",

  "Parameters" : {
    "KeyName": {
```

```
      "Description" : "Name of an existing EC2 KeyPair to enable SSH access to the
instance",
      "Type": "AWS::EC2::KeyPair::KeyName",
      "ConstraintDescription" : "Can contain only ASCII characters."
    },
    "InstanceType" : {
      "Description" : "WebServer EC2 instance type",
      "Type" : "String",
      "Default" : "m1.small",
      "AllowedValues" : [ "t1.micro", "t2.micro", "t2.small", "t2.medium", "m1.small",
"m1.medium", "m1.large", "m1.xlarge", "m2.xlarge", "m2.2xlarge", "m2.4xlarge",
"m3.medium", "m3.large", "m3.xlarge", "m3.2xlarge", "c1.medium", "c1.xlarge", "c3.large",
"c3.xlarge", "c3.2xlarge", "c3.4xlarge", "c3.8xlarge", "g2.2xlarge", "r3.large",
"r3.xlarge", "r3.2xlarge", "r3.4xlarge", "r3.8xlarge", "i2.xlarge", "i2.2xlarge",
"i2.4xlarge", "i2.8xlarge", "hi1.4xlarge", "hs1.8xlarge", "cr1.8xlarge", "cc2.8xlarge",
"cg1.4xlarge"],
      "ConstraintDescription" : "Must be a valid EC2 instance type"
    },
    "SSHLocation" : {
      "Description" : "The IP address range that can be used to SSH to the EC2 instances",
      "Type": "String",
      "MinLength": "9",
      "MaxLength": "18",
      "Default": "0.0.0.0/0",
      "AllowedPattern": "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\\d{1,2})",
      "ConstraintDescription": "Must be a valid IP CIDR range of the form x.x.x.x/x"
    }
  },

  "Mappings" : {
    "AWSInstanceType2Arch" : {
      "t1.micro"    : { "Arch" : "PV64"   },
      "t2.micro"    : { "Arch" : "HVM64"  },
      "t2.small"    : { "Arch" : "HVM64"  },
      "t2.medium"   : { "Arch" : "HVM64"  },
      "m1.small"    : { "Arch" : "PV64"   },
      "m1.medium"   : { "Arch" : "PV64"   },
      "m1.large"    : { "Arch" : "PV64"   },
      "m1.xlarge"   : { "Arch" : "PV64"   },
      "m2.xlarge"   : { "Arch" : "PV64"   },
      "m2.2xlarge"  : { "Arch" : "PV64"   },
      "m2.4xlarge"  : { "Arch" : "PV64"   },
      "m3.medium"   : { "Arch" : "HVM64"  },
      "m3.large"    : { "Arch" : "HVM64"  },
      "m3.xlarge"   : { "Arch" : "HVM64"  },
      "m3.2xlarge"  : { "Arch" : "HVM64"  },
      "c1.medium"   : { "Arch" : "PV64"   },
      "c1.xlarge"   : { "Arch" : "PV64"   },
      "c3.large"    : { "Arch" : "HVM64"  },
      "c3.xlarge"   : { "Arch" : "HVM64"  },
      "c3.2xlarge"  : { "Arch" : "HVM64"  },
      "c3.4xlarge"  : { "Arch" : "HVM64"  },
      "c3.8xlarge"  : { "Arch" : "HVM64"  },
      "g2.2xlarge"  : { "Arch" : "HVMG2"  },
      "r3.large"    : { "Arch" : "HVM64"  },
      "r3.xlarge"   : { "Arch" : "HVM64"  },
      "r3.2xlarge"  : { "Arch" : "HVM64"  },
      "r3.4xlarge"  : { "Arch" : "HVM64"  },
      "r3.8xlarge"  : { "Arch" : "HVM64"  },
      "i2.xlarge"   : { "Arch" : "HVM64"  },
      "i2.2xlarge"  : { "Arch" : "HVM64"  },
      "i2.4xlarge"  : { "Arch" : "HVM64"  },
      "i2.8xlarge"  : { "Arch" : "HVM64"  },
      "hi1.4xlarge" : { "Arch" : "HVM64"  },
      "hs1.8xlarge" : { "Arch" : "HVM64"  },
      "cr1.8xlarge" : { "Arch" : "HVM64"  },
```

```
      "cc2.8xlarge" : { "Arch" : "HVM64"  }
    },

    "AWSRegionArch2AMI" : {
      "us-east-1"      : { "PV64" : "ami-50842d38", "HVM64" : "ami-08842d60", "HVMG2" :
 "ami-3a329952"  },
      "us-west-2"      : { "PV64" : "ami-af86c69f", "HVM64" : "ami-8786c6b7", "HVMG2" :
 "ami-47296a77"  },
      "us-west-1"      : { "PV64" : "ami-c7a8a182", "HVM64" : "ami-cfa8a18a", "HVMG2" :
 "ami-331b1376"  },
      "eu-west-1"      : { "PV64" : "ami-aa8f28dd", "HVM64" : "ami-748e2903", "HVMG2" :
 "ami-00913777"  },
      "ap-southeast-1" : { "PV64" : "ami-20e1c572", "HVM64" : "ami-d6e1c584", "HVMG2" :
 "ami-fabe9aa8"  },
      "ap-northeast-1" : { "PV64" : "ami-21072820", "HVM64" : "ami-35072834", "HVMG2" :
 "ami-5dd1ff5c"  },
      "ap-southeast-2" : { "PV64" : "ami-8b4724b1", "HVM64" : "ami-fd4724c7", "HVMG2" :
 "ami-e98ae9d3"  },
      "sa-east-1"      : { "PV64" : "ami-9d6cc680", "HVM64" : "ami-956cc688", "HVMG2" :
 "NOT_SUPPORTED" },
      "cn-north-1"     : { "PV64" : "ami-a857c591", "HVM64" : "ami-ac57c595", "HVMG2" :
 "NOT_SUPPORTED" },
      "eu-central-1"   : { "PV64" : "ami-a03503bd", "HVM64" : "ami-b43503a9", "HVMG2" :
 "ami-b03503ad"  }
    }
  },

  "Resources" : {

    "WebServerInstance": {
      "Type": "AWS::EC2::Instance",
      "Properties": {
        "ImageId" : { "Fn::FindInMap" : [ "AWSRegionArch2AMI", { "Ref" : "AWS::Region" },
                          { "Fn::FindInMap" : [ "AWSInstanceType2Arch", { "Ref" :
 "InstanceType" }, "Arch" ] } ] },
        "InstanceType"   : { "Ref" : "InstanceType" },
        "SecurityGroups" : [ {"Ref" : "WebServerSecurityGroup"} ],
        "KeyName"        : { "Ref" : "KeyName" }
      }
    },
    "WebServerSecurityGroup" : {
      "Type" : "AWS::EC2::SecurityGroup",
      "Properties" : {
        "GroupDescription" : "Enable HTTP access via port 80",
        "SecurityGroupIngress" : [
          {"IpProtocol" : "tcp", "FromPort" : "80", "ToPort" : "80", "CidrIp" :
 "0.0.0.0/0"},
          {"IpProtocol" : "tcp", "FromPort" : "22", "ToPort" : "22", "CidrIp" : { "Ref" :
 "SSHLocation"}}
        ]
      }
    }
  },

  "Outputs" : {
    "WebsiteURL" : {
      "Description" : "URL for newly created LAMP stack",
      "Value" : { "Fn::Join" : ["", ["http://", { "Fn::GetAtt" : [ "WebServerInstance",
 "PublicDnsName" ]}]] }
    }
  }
}
```

In addition to the Amazon EC2 instance and security group, we create three input parameters that specify the instance type, an Amazon EC2 key pair to use for SSH access, and an IP address range that can be used to SSH to the instance. The mapping section ensures that AWS CloudFormation uses the correct AMI ID for the stack's region and the Amazon EC2 instance type. Finally, the output section outputs the public URL of the web server.

LAMP Installation

You'll build on the previous basic Amazon EC2 template to automatically install Apache, MySQL, and PHP. To install the applications, you'll add a `UserData` property and `Metadata` property. However, the template won't configure and start the applications until the next section.

In the following example, sections marked with an ellipsis (...) are omitted for brevity. Additions to the template are shown in red italic text.

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",

  "Description" : "AWS CloudFormation Sample Template LAMP_Install_Only: ...",

  "Parameters" : {

    "KeyName" : { ... },

    "InstanceType" : { ...  },

  "Mappings" : { ... },

  "Resources" : {
    "WebServerInstance": {
      "Type": "AWS::EC2::Instance",
      "Metadata" : {
        "Comment1" : "Configure the bootstrap helpers to install the Apache Web Server and
 PHP",
        "Comment2" : "Save website content to /var/www/html/index.php",

        "AWS::CloudFormation::Init" : {
          "configSets" : {
            "Install" : [ "Install" ]
          },

          "Install" : {
            "packages" : {
              "yum" : {
                "mysql"        : [],
                "mysql-server" : [],
                "mysql-libs"   : [],
                "httpd"        : [],
                "php"          : [],
                "php-mysql"    : []
              }
            },

            "files" : {
              "/var/www/html/index.php" : {
                "content" : { "Fn::Join" : [ "", [
                  "<html>\n",
                  "  <head>\n",
                  "    <title>AWS CloudFormation PHP Sample</title>\n",
                  "    <meta http-equiv=\"Content-Type\" content=\"text/html;
 charset=ISO-8859-1\">\n",
                  "  </head>\n",
                  "  <body>\n",
                  "    <h1>Welcome to the AWS CloudFormation PHP Sample</h1>\n",
                  "    <p/>\n",
                  "    <?php\n",
```

```
                  "       // Print out the current data and time\n",
                  "       print \"The Current Date and Time is: <br/>\";\n",
                  "       print date(\"g:i A l, F j Y.\");\n",
                  "     ?>\n",
                  "     <p/>\n",
                  "     <?php\n",
                  "       // Setup a handle for CURL\n",
                  "       $curl_handle=curl_init();\n",
                  "       curl_setopt($curl_handle,CURLOPT_CONNECTTIMEOUT,2);\n",
                  "       curl_setopt($curl_handle,CURLOPT_RETURNTRANSFER,1);\n",
                  "       // Get the hostname of the intance from the instance metadata\n",
                  "       curl_setopt($curl_handle,CURLOPT_URL,'http://169.254.169.254/
latest/meta-data/public-hostname');\n",
                  "       $hostname = curl_exec($curl_handle);\n",
                  "       if (empty($hostname))\n",
                  "       {\n",
                  "         print \"Sorry, for some reason, we got no hostname back <br />
\";\n",
                  "       }\n",
                  "       else\n",
                  "       {\n",
                  "         print \"Server = \" . $hostname . \"<br />\";\n",
                  "       }\n",
                  "       // Get the instance-id of the intance from the instance metadata
\n",
                  "       curl_setopt($curl_handle,CURLOPT_URL,'http://169.254.169.254/
latest/meta-data/instance-id');\n",
                  "       $instanceid = curl_exec($curl_handle);\n",
                  "       if (empty($instanceid))\n",
                  "       {\n",
                  "         print \"Sorry, for some reason, we got no instance id back <br /
>\";\n",
                  "       }\n",
                  "       else\n",
                  "       {\n",
                  "         print \"EC2 instance-id = \" . $instanceid . \"<br />\";\n",
                  "       }\n",
                  "       $Database   = \"", {"Ref" : "DBName"}, "\";\n",
                  "       $DBUser     = \"", {"Ref" : "DBUsername"}, "\";\n",
                  "       $DBPassword = \"", {"Ref" : "DBPassword"}, "\";\n",
                  "       print \"Database = \" . $Database . \"<br />\";\n",
                  "       $dbconnection = mysql_connect($Database, $DBUser, $DBPassword)\n",
                  "                       or die(\"Could not connect: \" . ysql_error());
\n",
                  "       print (\"Connected to $Database successfully\");\n",
                  "       mysql_close($dbconnection);\n",
                  "     ?>\n",
                  "     <h2>PHP Information</h2>\n",
                  "     <p/>\n",
                  "     <?php\n",
                  "       phpinfo();\n",
                  "     ?>\n",
                  "   </body>\n",
                  "</html>\n"
                ]]},
                "mode"  : "000600",
                "owner" : "apache",
                "group" : "apache"
              },
            "services" : {
              "sysvinit" : {
                "httpd"   : { "enabled" : "true", "ensureRunning" : "true" }
              }
            }
          }
      },
```

```
      "Properties": {
        "ImageId" : { "Fn::FindInMap" : [ "AWSRegionArch2AMI", { "Ref" : "AWS::Region" },
                          { "Fn::FindInMap" : [ "AWSInstanceType2Arch", { "Ref" :
 "InstanceType" }, "Arch" ] } ] },
        "InstanceType"   : { "Ref" : "InstanceType" },
        "SecurityGroups" : [ {"Ref" : "WebServerSecurityGroup"} ],
        "KeyName"        : { "Ref" : "KeyName" },
        "UserData"       : { "Fn::Base64" : { "Fn::Join" : ["", [
             "#!/bin/bash -xe\n",
             "yum install -y aws-cfn-bootstrap\n",

             "# Install the files and packages from the metadata\n",
             "/opt/aws/bin/cfn-init -v ",
             "         --stack ", { "Ref" : "AWS::StackName" },
             "         --resource WebServerInstance ",
             "         --configsets Install ",
             "         --region ", { "Ref" : "AWS::Region" }, "\n"
  ]]}}
      }
    },


    "WebServerSecurityGroup" : { ...  }
  },

  "Outputs" : { ... }
}
```

The `UserData` property runs two shell commands: install the AWS CloudFormation helper scripts and then run the cfn-init (p. 1355) helper script. Because the helper scripts are updated periodically, running the `yum install -y aws-cfn-bootstrap` command ensures that you get the latest helper scripts. When you run cfn-init, it reads metadata from the AWS::CloudFormation::Init (p. 470) resource, which describes the actions to be carried out by cfn-init. For example, you can use cfn-init and AWS::CloudFormation::Init to install packages, write files to disk, or start a service. In our case, cfn-init installs the listed packages (httpd, mysql, and php) and creates the `/var/www/html/index.php` file (a sample PHP application).

LAMP Configuration

Now that we have a template that installs Linux, Apache, MySQL, and PHP, we'll need to expand the template so that it automatically configures and runs Apache, MySQL, and PHP. In the following example, we expand on the `Parameters` section, `AWS::CloudFormation::Init` resource, and `UserData` property to complete the configuration. As with the previous template, sections marked with an ellipsis (...) are omitted for brevity. Additions to the template are shown in red italic text.

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",

  "Description" : "AWS CloudFormation Sample Template LAMP_Single_Instance: Create a LAMP
 stack using a single EC2 instance and a local MySQL database for storage. This template
 demonstrates using the AWS CloudFormation bootstrap scripts to install the packages and
 files necessary to deploy the Apache web server, PHP and MySQL at instance launch time.
 **WARNING** This template creates an Amazon EC2 instance. You will be billed for the AWS
 resources used if you create a stack from this template.",

  "Parameters" : {

    "KeyName" : { ...  },

    "DBName": {
      "Default": "MyDatabase",
      "Description" : "MySQL database name",
      "Type": "String",
      "MinLength": "1",
```

```
      "MaxLength": "64",
      "AllowedPattern" : "[a-zA-Z][a-zA-Z0-9]*",
      "ConstraintDescription" : "Must begin with a letter and contain only alphanumeric 
characters"
    },

    "DBUsername": {
      "NoEcho": "true",
      "Description" : "Username for MySQL database access",
      "Type": "String",
      "MinLength": "1",
      "MaxLength": "16",
      "AllowedPattern" : "[a-zA-Z][a-zA-Z0-9]*",
      "ConstraintDescription" : "Must begin with a letter and contain only alphanumeric 
characters"
    },

    "DBPassword": {
      "NoEcho": "true",
      "Description" : "Password for MySQL database access",
      "Type": "String",
      "MinLength": "1",
      "MaxLength": "41",
      "AllowedPattern" : "[a-zA-Z0-9]*",
      "ConstraintDescription" : "Must contain only alphanumeric characters"
    },

    "DBRootPassword": {
      "NoEcho": "true",
      "Description" : "Root password for MySQL",
      "Type": "String",
      "MinLength": "1",
      "MaxLength": "41",
      "AllowedPattern" : "[a-zA-Z0-9]*",
      "ConstraintDescription" : "Must contain only alphanumeric characters"
    },

    "InstanceType" : { ...  }

  },

  "Mappings" : {
  ...
  },

  "Resources" : {
    "WebServer": {
      "Type": "AWS::EC2::Instance",
      "Metadata" : {
        "Comment1" : "Configure the bootstrap helpers to install the Apache Web Server and 
PHP",
        "Comment2" : "Save website content to /var/www/html/index.php",

        "AWS::CloudFormation::Init" : {
          "configSets" : {
            "InstallAndRun" : [ "Install", "Configure" ]
          },

          "Install" : {
            "packages" : {
              "yum" : {
                "mysql"        : [],
                "mysql-server" : [],
                "mysql-libs"   : [],
                "httpd"        : [],
                "php"          : [],
```

```
                "php-mysql"    : []
              }
            },

            "files" : {
              "/var/www/html/index.php" : {
                "content" : { ... },
                "mode"  : "000600",
                "owner" : "apache",
                "group" : "apache"
              },
            "/tmp/setup.mysql" : {
                "content" : { "Fn::Join" : ["", [
                  "CREATE DATABASE ", { "Ref" : "DBName" }, ";\n",
                  "GRANT ALL ON ", { "Ref" : "DBName" }, ".* TO '", { "Ref" :
 "DBUsername" }, "'@localhost IDENTIFIED BY '", { "Ref" : "DBPassword" }, "';\n"
                  ]]},
                "mode"  : "000400",
                "owner" : "root",
                "group" : "root"
              },
              "/etc/cfn/cfn-hup.conf" : {
                "content" : { "Fn::Join" : ["", [
                  "[main]\n",
                  "stack=", { "Ref" : "AWS::StackId" }, "\n",
                  "region=", { "Ref" : "AWS::Region" }, "\n"
                ]]},
                "mode"    : "000400",
                "owner"   : "root",
                "group"   : "root"
              },

              "/etc/cfn/hooks.d/cfn-auto-reloader.conf" : {
                "content": { "Fn::Join" : ["", [
                  "[cfn-auto-reloader-hook]\n",
                  "triggers=post.update\n",
                  "path=Resources.WebServerInstance.Metadata.AWS::CloudFormation::Init\n",
                  "action=/opt/aws/bin/cfn-init -v ",
                  "         --stack ", { "Ref" : "AWS::StackName" },
                  "         --resource WebServerInstance ",
                  "         --configsets InstallAndRun ",
                  "         --region ", { "Ref" : "AWS::Region" }, "\n",
                  "runas=root\n"
                ]]}
              }
            },
            },

            "services" : {
              "sysvinit" : {
                "mysqld"  : { "enabled" : "true", "ensureRunning" : "true" },
                "httpd"   : { "enabled" : "true", "ensureRunning" : "true" },
                "cfn-hup" : { "enabled" : "true", "ensureRunning" : "true",
                              "files" : ["/etc/cfn/cfn-hup.conf", "/etc/cfn/hooks.d/cfn-
auto-reloader.conf"]}
              }
            }
          },
          "Configure" : {
            "commands" : {
              "01_set_mysql_root_password" : {
                "command" : { "Fn::Join" : ["", ["mysqladmin -u root password '", { "Ref" :
 "DBRootPassword" }, "'"]]},
                "test" : { "Fn::Join" : ["", ["$(mysql ", { "Ref" : "DBUsername" }, " -u
 root --password='", { "Ref" : "DBRootPassword" }, "' >/dev/null 2>&1 </dev/null); (( $? !=
 0 ))"]]}
```

```
              },
              "02_create_database" : {
                "command" : { "Fn::Join" : ["", ["mysql -u root --password='", { "Ref" :
"DBRootPassword" }, "' < /tmp/setup.mysql"]]},
                "test" : { "Fn::Join" : ["", ["$(mysql ", { "Ref" : "DBUsername" }, " -u
root --password='", { "Ref" : "DBRootPassword" }, "' >/dev/null 2>&1 </dev/null); (( $? !=
0 ))"]]}
              }
            }
          }
        }
      },
      "Properties": {
        "ImageId" : { "Fn::FindInMap" : [ "AWSRegionArch2AMI", { "Ref" : "AWS::Region" },
                          { "Fn::FindInMap" : [ "AWSInstanceType2Arch", { "Ref" :
"InstanceType" }, "Arch" ] } ] },
        "InstanceType"   : { "Ref" : "InstanceType" },
        "SecurityGroups" : [ {"Ref" : "WebServerSecurityGroup"} ],
        "KeyName"        : { "Ref" : "KeyName" },
        "UserData"       : { "Fn::Base64" : { "Fn::Join" : ["", [
          "#!/bin/bash -xe\n",
          "yum install -y aws-cfn-bootstrap\n",

          "# Install the files and packages from the metadata\n",
          "/opt/aws/bin/cfn-init ",
          "         --stack ", { "Ref" : "AWS::StackName" },
          "         --resource WebServerInstance ",^M
          "         --configsets InstallAndRun ",
          "         --region ", { "Ref" : "AWS::Region" }, "\n"
  ]]}}
      }
    },

    "WebServerSecurityGroup" : { ... }
  },

  "Outputs" : { ... }
}
```

The example adds more parameters to obtain information for configuring the MySQL database, such as the database name, user name, password, and root password. The parameters also contain constraints that catch incorrectly formatted values before AWS CloudFormation creates the stack.

In the `AWS::CloudFormation::Init` resource, we added a MySQL setup file, containing the database name, user name, and password. The example also adds a `services` property to ensure that the httpd and mysqld services are running (`ensureRunning` set to `true`) and to ensure that the services are restarted if the instance is rebooted (`enabled` set to `true`). A good practice is to also include the cfn-hup (p. 1363) helper script, with which you can make configuration updates to running instances by updating the stack template. For example, you could change the sample PHP application and then run a stack update to deploy the change.

In order to run the MySQL commands after the installation is complete, the example adds another configuration set to run the commands. Configuration sets are useful when you have a series of tasks that must be completed in a specific order. The example first runs the `Installation` configuration set and then the `Configure` configuration set. The `Configure` configuration set specifies the database root password and then creates a database. In the commands section, the commands are processed in alphabetical order by name, so the example adds a number before each command name to indicate its desired run order.

CreationPolicy Attribute

Finally, you need a way to instruct AWS CloudFormation to complete stack creation only after all the services (such as Apache and MySQL) are running and not after all the stack resources are created. In other words, if you use the template from the previous section to launch a stack, AWS CloudFormation sets the status of the stack as `CREATE_COMPLETE` after it successfully creates all the resources. However, if one or more services failed to start, AWS CloudFormation still sets the stack status as `CREATE_COMPLETE`. To prevent the status from changing to `CREATE_COMPLETE` until all the services have successfully started, you can add a CreationPolicy (p. 1293) attribute to the instance. This attribute puts the instance's status in `CREATE_IN_PROGRESS` until AWS CloudFormation receives the required number of success signals or the timeout period is exceeded, so you can control when the instance has been successfully created.

The following example adds a creation policy to the Amazon EC2 instance to ensure that cfn-init completes the LAMP installation and configuration before the stack creation is completed. In conjunction with the creation policy, the example needs to run the cfn-signal (p. 1358) helper script to signal AWS CloudFormation when all the applications are installed and configured.

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",

  "Description" : "AWS CloudFormation Sample Template LAMP_Single_Instance: ...",

  "Parameters" : { ... },

  "Mappings" : { ... },

  "Resources" : {
    "WebServerInstance": {
      "Type": "AWS::EC2::Instance",
      "Metadata" : { ... },
      "Properties": {
        "ImageId" : { "Fn::FindInMap" : [ "AWSRegionArch2AMI", { "Ref" : "AWS::Region" },
                          { "Fn::FindInMap" : [ "AWSInstanceType2Arch", { "Ref" :
 "InstanceType" }, "Arch" ] } ] },
        "InstanceType"   : { "Ref" : "InstanceType" },
        "SecurityGroups" : [ {"Ref" : "WebServerSecurityGroup"} ],
        "KeyName"        : { "Ref" : "KeyName" },
        "UserData"       : { "Fn::Base64" : { "Fn::Join" : ["", [
             "#!/bin/bash -xe\n",
             "yum update aws-cfn-bootstrap\n",

             "# Install the files and packages from the metadata\n",
             "/opt/aws/bin/cfn-init ",
             "         --stack ", { "Ref" : "AWS::StackName" },
             "         --resource WebServerInstance ",^M
             "         --configsets InstallAndRun ",
             "         --region ", { "Ref" : "AWS::Region" }, "\n",

             "# Signal the status from cfn-init\n",
             "/opt/aws/bin/cfn-signal -e $? ",
             "         --stack ", { "Ref" : "AWS::StackName" },
             "         --resource WebServerInstance ",
             "         --region ", { "Ref" : "AWS::Region" }, "\n"
        ]]}}
      },
      "CreationPolicy" : {
        "ResourceSignal" : {
          "Timeout" : "PT5M"
        }
      }
    },

    "WebServerSecurityGroup" : { ...
    }
  },

  "Outputs" : {
    "WebsiteURL" : { ...
```

```
    }
  }
}
```

The creation policy attribute uses the ISO 8601 format to define a timeout period of 5 minutes. And because you're waiting for just 1 instance to be configured, you only need to wait for one success signal, which is the default count.

In the `UserData` property, the template runs the cfn-signal script to send a success signal with an exit code if all the services are configured and started successfully. When you use the cfn-signal script, you must include the stack ID or name and the logical ID of the resource that you want to signal. If the configuration fails or if the timeout period is exceeded, cfn-signal sends a failure signal that causes the resource creation to fail.

The following example shows final complete template. You can also view the template at the following location:

https://s3.amazonaws.com/cloudformation-templates-us-east-1/LAMP_Single_Instance.template

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",

  "Description" : "AWS CloudFormation Sample Template LAMP_Single_Instance: Create a LAMP
 stack using a single EC2 instance and a local MySQL database for storage. This template
 demonstrates using the AWS CloudFormation bootstrap scripts to install the packages and
 files necessary to deploy the Apache web server, PHP and MySQL at instance launch time.
 **WARNING** This template creates an Amazon EC2 instance. You will be billed for the AWS
 resources used if you create a stack from this template.",

  "Parameters" : {

    "KeyName": {
      "Description" : "Name of an existing EC2 KeyPair to enable SSH access to the
 instance",
      "Type": "AWS::EC2::KeyPair::KeyName",
      "ConstraintDescription" : "Can contain only ASCII characters."
    },

    "DBName": {
      "Default": "MyDatabase",
      "Description" : "MySQL database name",
      "Type": "String",
      "MinLength": "1",
      "MaxLength": "64",
      "AllowedPattern" : "[a-zA-Z][a-zA-Z0-9]*",
      "ConstraintDescription" : "Must begin with a letter and contain only alphanumeric
 characters"
    },

    "DBUsername": {
      "NoEcho": "true",
      "Description" : "User name for MySQL database access",
      "Type": "String",
      "MinLength": "1",
      "MaxLength": "16",
      "AllowedPattern" : "[a-zA-Z][a-zA-Z0-9]*",
      "ConstraintDescription" : "Must begin with a letter and contain only alphanumeric
 characters"
    },

    "DBPassword": {
      "NoEcho": "true",
      "Description" : "Password for MySQL database access",
```

```
      "Type": "String",
      "MinLength": "1",
      "MaxLength": "41",
      "AllowedPattern" : "[a-zA-Z0-9]*",
      "ConstraintDescription" : "Must contain only alphanumeric characters"
    },

    "DBRootPassword": {
      "NoEcho": "true",
      "Description" : "Root password for MySQL",
      "Type": "String",
      "MinLength": "1",
      "MaxLength": "41",
      "AllowedPattern" : "[a-zA-Z0-9]*",
      "ConstraintDescription" : "Must contain only alphanumeric characters"
    },

    "InstanceType" : {
      "Description" : "WebServer EC2 instance type",
      "Type" : "String",
      "Default" : "m1.small",
      "AllowedValues" : [ "t1.micro", "t2.micro", "t2.small", "t2.medium", "m1.small",
 "m1.medium", "m1.large", "m1.xlarge", "m2.xlarge", "m2.2xlarge", "m2.4xlarge",
 "m3.medium", "m3.large", "m3.xlarge", "m3.2xlarge", "c1.medium", "c1.xlarge", "c3.large",
 "c3.xlarge", "c3.2xlarge", "c3.4xlarge", "c3.8xlarge", "g2.2xlarge", "r3.large",
 "r3.xlarge", "r3.2xlarge", "r3.4xlarge", "r3.8xlarge", "i2.xlarge", "i2.2xlarge",
 "i2.4xlarge", "i2.8xlarge", "hi1.4xlarge", "hs1.8xlarge", "cr1.8xlarge", "cc2.8xlarge",
 "cg1.4xlarge"],
      "ConstraintDescription" : "Must be a valid EC2 instance type"
    },
    "SSHLocation" : {
      "Description" : "The IP address range that can be used to SSH to the EC2 instances",
      "Type": "String",
      "MinLength": "9",
      "MaxLength": "18",
      "Default": "0.0.0.0/0",
      "AllowedPattern": "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\\d{1,2})",
      "ConstraintDescription": "Must be a valid IP CIDR range of the form x.x.x.x/x"
    }
  },

  "Mappings" : {
    "AWSInstanceType2Arch" : {
      "t1.micro"    : { "Arch" : "PV64"   },
      "t2.micro"    : { "Arch" : "HVM64"  },
      "t2.small"    : { "Arch" : "HVM64"  },
      "t2.medium"   : { "Arch" : "HVM64"  },
      "m1.small"    : { "Arch" : "PV64"   },
      "m1.medium"   : { "Arch" : "PV64"   },
      "m1.large"    : { "Arch" : "PV64"   },
      "m1.xlarge"   : { "Arch" : "PV64"   },
      "m2.xlarge"   : { "Arch" : "PV64"   },
      "m2.2xlarge"  : { "Arch" : "PV64"   },
      "m2.4xlarge"  : { "Arch" : "PV64"   },
      "m3.medium"   : { "Arch" : "HVM64"  },
      "m3.large"    : { "Arch" : "HVM64"  },
      "m3.xlarge"   : { "Arch" : "HVM64"  },
      "m3.2xlarge"  : { "Arch" : "HVM64"  },
      "c1.medium"   : { "Arch" : "PV64"   },
      "c1.xlarge"   : { "Arch" : "PV64"   },
      "c3.large"    : { "Arch" : "HVM64"  },
      "c3.xlarge"   : { "Arch" : "HVM64"  },
      "c3.2xlarge"  : { "Arch" : "HVM64"  },
      "c3.4xlarge"  : { "Arch" : "HVM64"  },
      "c3.8xlarge"  : { "Arch" : "HVM64"  },
      "g2.2xlarge"  : { "Arch" : "HVMG2"  },
```

```
      "r3.large"    : { "Arch" : "HVM64"  },
      "r3.xlarge"   : { "Arch" : "HVM64"  },
      "r3.2xlarge"  : { "Arch" : "HVM64"  },
      "r3.4xlarge"  : { "Arch" : "HVM64"  },
      "r3.8xlarge"  : { "Arch" : "HVM64"  },
      "i2.xlarge"   : { "Arch" : "HVM64"  },
      "i2.2xlarge"  : { "Arch" : "HVM64"  },
      "i2.4xlarge"  : { "Arch" : "HVM64"  },
      "i2.8xlarge"  : { "Arch" : "HVM64"  },
      "hi1.4xlarge" : { "Arch" : "HVM64"  },
      "hs1.8xlarge" : { "Arch" : "HVM64"  },
      "cr1.8xlarge" : { "Arch" : "HVM64"  },
      "cc2.8xlarge" : { "Arch" : "HVM64"  }
    },

    "AWSRegionArch2AMI" : {
      "us-east-1"      : { "PV64" : "ami-50842d38", "HVM64" : "ami-08842d60", "HVMG2" :
"ami-3a329952"  },
      "us-west-2"      : { "PV64" : "ami-af86c69f", "HVM64" : "ami-8786c6b7", "HVMG2" :
"ami-47296a77"  },
      "us-west-1"      : { "PV64" : "ami-c7a8a182", "HVM64" : "ami-cfa8a18a", "HVMG2" :
"ami-331b1376"  },
      "eu-west-1"      : { "PV64" : "ami-aa8f28dd", "HVM64" : "ami-748e2903", "HVMG2" :
"ami-00913777"  },
      "ap-southeast-1" : { "PV64" : "ami-20e1c572", "HVM64" : "ami-d6e1c584", "HVMG2" :
"ami-fabe9aa8"  },
      "ap-northeast-1" : { "PV64" : "ami-21072820", "HVM64" : "ami-35072834", "HVMG2" :
"ami-5dd1ff5c"  },
      "ap-southeast-2" : { "PV64" : "ami-8b4724b1", "HVM64" : "ami-fd4724c7", "HVMG2" :
"ami-e98ae9d3"  },
      "sa-east-1"      : { "PV64" : "ami-9d6cc680", "HVM64" : "ami-956cc688", "HVMG2" :
"NOT_SUPPORTED" },
      "cn-north-1"     : { "PV64" : "ami-a857c591", "HVM64" : "ami-ac57c595", "HVMG2" :
"NOT_SUPPORTED" },
      "eu-central-1"   : { "PV64" : "ami-a03503bd", "HVM64" : "ami-b43503a9", "HVMG2" :
"ami-b03503ad"  }
    }

  },

  "Resources" : {

    "WebServerInstance": {
      "Type": "AWS::EC2::Instance",
      "Metadata" : {
        "AWS::CloudFormation::Init" : {
          "configSets" : {
            "InstallAndRun" : [ "Install", "Configure" ]
          },

          "Install" : {
            "packages" : {
              "yum" : {
                "mysql"        : [],
                "mysql-server" : [],
                "mysql-libs"   : [],
                "httpd"        : [],
                "php"          : [],
                "php-mysql"    : []
              }
            },

            "files" : {
              "/var/www/html/index.php" : {
                "content" : { "Fn::Join" : [ "", [
                  "<html>\n",
```

```
                  "  <head>\n",
                  "    <title>AWS CloudFormation PHP Sample</title>\n",
                  "    <meta http-equiv=\"Content-Type\" content=\"text/html;
 charset=ISO-8859-1\">\n",
                  "  </head>\n",
                  "  <body>\n",
                  "    <h1>Welcome to the AWS CloudFormation PHP Sample</h1>\n",
                  "    <p/>\n",
                  "    <?php\n",
                  "      // Print out the current data and time\n",
                  "      print \"The Current Date and Time is: <br/>\";\n",
                  "      print date(\"g:i A l, F j Y.\");\n",
                  "    ?>\n",
                  "    <p/>\n",
                  "    <?php\n",
                  "      // Setup a handle for CURL\n",
                  "      $curl_handle=curl_init();\n",
                  "      curl_setopt($curl_handle,CURLOPT_CONNECTTIMEOUT,2);\n",
                  "      curl_setopt($curl_handle,CURLOPT_RETURNTRANSFER,1);\n",
                  "      // Get the hostname of the intance from the instance metadata\n",
                  "      curl_setopt($curl_handle,CURLOPT_URL,'http://169.254.169.254/
latest/meta-data/public-hostname');\n",
                  "      $hostname = curl_exec($curl_handle);\n",
                  "      if (empty($hostname))\n",
                  "      {\n",
                  "        print \"Sorry, for some reason, we got no hostname back <br />
\";\n",
                  "      }\n",
                  "      else\n",
                  "      {\n",
                  "        print \"Server = \" . $hostname . \"<br />\";\n",
                  "      }\n",
                  "      // Get the instance-id of the intance from the instance metadata
\n",
                  "      curl_setopt($curl_handle,CURLOPT_URL,'http://169.254.169.254/
latest/meta-data/instance-id');\n",
                  "      $instanceid = curl_exec($curl_handle);\n",
                  "      if (empty($instanceid))\n",
                  "      {\n",
                  "        print \"Sorry, for some reason, we got no instance id back <br /
>\";\n",
                  "      }\n",
                  "      else\n",
                  "      {\n",
                  "        print \"EC2 instance-id = \" . $instanceid . \"<br />\";\n",
                  "      }\n",
                  "      $Database   = \"", {"Ref" : "DBName"}, "\";\n",
                  "      $DBUser     = \"", {"Ref" : "DBUsername"}, "\";\n",
                  "      $DBPassword = \"", {"Ref" : "DBPassword"}, "\";\n",
                  "      print \"Database = \" . $Database . \"<br />\";\n",
                  "      $dbconnection = mysql_connect($Database, $DBUser, $DBPassword)\n",
                  "                      or die(\"Could not connect: \" . ysql_error());
\n",
                  "      print (\"Connected to $Database successfully\");\n",
                  "      mysql_close($dbconnection);\n",
                  "    ?>\n",
                  "    <h2>PHP Information</h2>\n",
                  "    <p/>\n",
                  "    <?php\n",
                  "      phpinfo();\n",
                  "    ?>\n",
                  "  </body>\n",
                  "</html>\n"
                ]]},
                "mode"  : "000600",
                "owner" : "apache",
```

```
                "group" : "apache"
              },

              "/tmp/setup.mysql" : {
                "content" : { "Fn::Join" : ["", [
                  "CREATE DATABASE ", { "Ref" : "DBName" }, ";\n",
                  "GRANT ALL ON ", { "Ref" : "DBName" }, ".* TO '", { "Ref" :
 "DBUsername" }, "'@localhost IDENTIFIED BY '", { "Ref" : "DBPassword" }, "';\n"
                  ]]},
                "mode"  : "000400",
                "owner" : "root",
                "group" : "root"
              },
              "/etc/cfn/cfn-hup.conf" : {
                "content" : { "Fn::Join" : ["", [
                  "[main]\n",
                  "stack=", { "Ref" : "AWS::StackId" }, "\n",
                  "region=", { "Ref" : "AWS::Region" }, "\n"
                ]]},
                "mode"    : "000400",
                "owner"   : "root",
                "group"   : "root"
              },

              "/etc/cfn/hooks.d/cfn-auto-reloader.conf" : {
                "content": { "Fn::Join" : ["", [
                  "[cfn-auto-reloader-hook]\n",
                  "triggers=post.update\n",
                  "path=Resources.WebServerInstance.Metadata.AWS::CloudFormation::Init\n",
                  "action=/opt/aws/bin/cfn-init -v ",
                  "         --stack ", { "Ref" : "AWS::StackName" },
                  "         --resource WebServerInstance ",
                  "         --configsets InstallAndRun ",
                  "         --region ", { "Ref" : "AWS::Region" }, "\n",
                  "runas=root\n"
                ]]}
              }
            },

            "services" : {
              "sysvinit" : {
                "mysqld"  : { "enabled" : "true", "ensureRunning" : "true" },
                "httpd"   : { "enabled" : "true", "ensureRunning" : "true" },
                "cfn-hup" : { "enabled" : "true", "ensureRunning" : "true",
                              "files" : ["/etc/cfn/cfn-hup.conf", "/etc/cfn/hooks.d/cfn-
auto-reloader.conf"]}
              }
            }
          },

          "Configure" : {
            "commands" : {
              "01_set_mysql_root_password" : {
                "command" : { "Fn::Join" : ["", ["mysqladmin -u root password '", { "Ref" :
 "DBRootPassword" }, "'"]]},
                "test" : { "Fn::Join" : ["", ["$(mysql ", { "Ref" : "DBUsername" }, " -u
 root --password='", { "Ref" : "DBRootPassword" }, "' >/dev/null 2>&1 </dev/null); (( $? !=
 0 ))"]]}
              },
              "02_create_database" : {
                "command" : { "Fn::Join" : ["", ["mysql -u root --password='", { "Ref" :
 "DBRootPassword" }, "' < /tmp/setup.mysql"]]},
                "test" : { "Fn::Join" : ["", ["$(mysql ", { "Ref" : "DBUsername" }, " -u
 root --password='", { "Ref" : "DBRootPassword" }, "' >/dev/null 2>&1 </dev/null); (( $? !=
 0 ))"]]}
              }
```

```
          }
        }
      }
    },
    "Properties": {
      "ImageId" : { "Fn::FindInMap" : [ "AWSRegionArch2AMI", { "Ref" : "AWS::Region" },
                        { "Fn::FindInMap" : [ "AWSInstanceType2Arch", { "Ref" :
 "InstanceType" }, "Arch" ] } ] },
      "InstanceType"   : { "Ref" : "InstanceType" },
      "SecurityGroups" : [ {"Ref" : "WebServerSecurityGroup"} ],
      "KeyName"        : { "Ref" : "KeyName" },
      "UserData"       : { "Fn::Base64" : { "Fn::Join" : ["", [
           "#!/bin/bash -xe\n",
           "yum install -y aws-cfn-bootstrap\n",

           "# Install the files and packages from the metadata\n",
           "/opt/aws/bin/cfn-init -v ",
           "         --stack ", { "Ref" : "AWS::StackName" },
           "         --resource WebServerInstance ",
           "         --configsets InstallAndRun ",
           "         --region ", { "Ref" : "AWS::Region" }, "\n",

           "# Signal the status from cfn-init\n",
           "/opt/aws/bin/cfn-signal -e $? ",
           "         --stack ", { "Ref" : "AWS::StackName" },
           "         --resource WebServerInstance ",
           "         --region ", { "Ref" : "AWS::Region" }, "\n"
      ]]}}
    },
    "CreationPolicy" : {
      "ResourceSignal" : {
        "Timeout" : "PT5M"
      }
    }
  },

  "WebServerSecurityGroup" : {
    "Type" : "AWS::EC2::SecurityGroup",
    "Properties" : {
      "GroupDescription" : "Enable HTTP access via port 80",
      "SecurityGroupIngress" : [
        {"IpProtocol" : "tcp", "FromPort" : "80", "ToPort" : "80", "CidrIp" :
 "0.0.0.0/0"},
        {"IpProtocol" : "tcp", "FromPort" : "22", "ToPort" : "22", "CidrIp" : { "Ref" :
 "SSHLocation"}}
      ]
    }
  }
 },

 "Outputs" : {
   "WebsiteURL" : {
     "Description" : "URL for newly created LAMP stack",
     "Value" : { "Fn::Join" : ["", ["http://", { "Fn::GetAtt" : [ "WebServerInstance",
 "PublicDnsName" ]}]] }
   }
 }
}
```

Creating Wait Conditions in a Template

Important

For Amazon EC2 and Auto Scaling resources, we recommend that you use a CreationPolicy attribute instead of wait conditions. Add a CreationPolicy attribute to those resources, and use the cfn-signal helper script to signal when an instance creation process has completed successfully. For more information, see CreationPolicy (p. 1293) or Deploying Applications on Amazon EC2 with AWS CloudFormation (p. 212).

Using the AWS::CloudFormation::WaitCondition (p. 487) resource and CreationPolicy (p. 1293) attribute, you can do the following:

- Coordinate stack resource creation with other configuration actions that are external to the stack creation
- Track the status of a configuration process

For example, you can start the creation of another resource after an application configuration is partially complete, or you can send signals during an installation and configuration process to track its progress.

Using a Wait Condition Handle

Note

If you use the VPC endpoint feature, resources in the VPC that respond to wait conditions must have access to AWS CloudFormation-specific Amazon Simple Storage Service (Amazon S3) buckets. Resources must send wait condition responses to a pre-signed Amazon S3 URL. If they can't send responses to Amazon S3, AWS CloudFormation won't receive a response and the stack operation fails. For more information, see AWS CloudFormation and VPC Endpoints (p. 19).

You can use the wait condition and wait condition handle to make AWS CloudFormation pause the creation of a stack and wait for a signal before it continues to create the stack. For example, you might want to download and configure applications on an Amazon EC2 instance before considering the creation of that Amazon EC2 instance complete.

The following list provides a summary of how a wait condition with a wait condition handle works:

- AWS CloudFormation creates a wait condition just like any other resource. When AWS CloudFormation creates a wait condition, it reports the wait condition's status as CREATE_IN_PROGRESS and waits until it receives the requisite number of success signals or the wait condition's timeout period has expired. If AWS CloudFormation receives the requisite number of success signals before the time out period expires, it continues creating the stack; otherwise, it sets the wait condition's status to CREATE_FAILED and rolls the stack back.
- The `Timeout` property determines how long AWS CloudFormation waits for the requisite number of success signals. `Timeout` is a minimum-bound property, meaning the timeout occurs no sooner than the time you specify, but can occur shortly thereafter. The maximum time that you can specify is 43200 seconds (12 hours).
- Typically, you want a wait condition to begin immediately after the creation of a specific resource, such as an Amazon EC2 instance, RDS DB instance, or Auto Scaling group. You do this by adding the DependsOn attribute (p. 1298) to a wait condition. When you add a DependsOn attribute to a wait condition, you specify that the wait condition is created only after the creation of a particular resource has completed. When the wait condition is created, AWS CloudFormation begins the timeout period and waits for success signals.
- You can also use the DependsOn attribute on other resources. For example, you may want an RDS DB instance to be created and a database configured on that DB instance first before creating the EC2 instances that use that database. In this case, you create a wait condition that has a DependsOn attribute that specifies the DB instance, and you create EC2 instance resources that have DependsOn attributes that specify the wait condition. This would ensure that the EC2 instances would only be created directly after the DB instance and the wait condition were completed.

- AWS CloudFormation must receive a specified number of success signals for a wait condition before setting that wait condition's status to CREATE_COMPLETE continuing the creation of the stack. The wait condition's Count property specifies the number of success signals. If none is set, the default is 1.
- A wait condition requires a wait condition handle to set up a presigned URL that is used as the signaling mechanism. The presigned URL enables you to send a signal without having to supply your AWS credentials. You use that presigned URL to signal success or failure, which is encapsulated in a JSON statement. For the format of that JSON statement, see the Wait Condition Signal JSON Format (p. 231).
- If a wait condition receives the requisite number of success signals (as defined in the Count property) before the timeout period expires, AWS CloudFormation marks the wait condition as CREATE_COMPLETE and continues creating the stack. Otherwise, AWS CloudFormation fails the wait condition and rolls the stack back (for example, if the timeout period expires without requisite success signals or if a failure signal is received).

To use a wait condition in a stack:

1. Declare an AWS::CloudFormation::WaitConditionHandle resource in the stack's template. A wait condition handle has no properties; however, a reference to a WaitConditionHandle resource resolves to a pre-signed URL that you can use to signal success or failure to the WaitCondition. For example:

```
"myWaitHandle" : {
     "Type" : "AWS::CloudFormation::WaitConditionHandle",
     "Properties" : {
     }
}
```

2. Declare an AWS::CloudFormation::WaitCondition resource in the stack's template. A WaitCondition resource has two required properties: Handle is a reference to a WaitConditionHandle declared in the template and Timeout is the number seconds for AWS CloudFormation to wait. You can optionally set the Count property, which determines the number of success signals that the wait condition must receive before AWS CloudFormation can resume creating the stack.

   To control when the wait condition is triggered, you set a DependsOn attribute on the wait condition. A DependsOn clause associates a resource with the wait condition. After AWS CloudFormation creates the DependsOn resource, it blocks further stack resource creation until one of the following events occur: a) the timeout period expires b) The requisite number of success signals are received c) A failure signal is received.

   Here is an example of a wait condition that begins after the successful creation of the Ec2Instance resource, uses the myWaitHandle resource as the WaitConditionHandle, has a timeout of 4500 seconds, and has the default Count of 1 (since no Count property is specified):

```
"myWaitCondition" : {
    "Type" : "AWS::CloudFormation::WaitCondition",
    "DependsOn" : "Ec2Instance",
    "Properties" : {
        "Handle" : { "Ref" : "myWaitHandle" },
         "Timeout" : "4500"
    }
}
```

3. Get the presigned URL to use for signaling.

   In the template, the presigned URL can be retrieved by passing the logical name of the AWS::CloudFormation::WaitConditionHandle resource to the Ref intrinsic function. For example, you can use the UserData property on AWS::EC2::Instance resources to pass the presigned URL to the Amazon EC2 instances so that scripts or applications running on those instances can signal success or failure to AWS CloudFormation:

```
"UserData" : {
   "Fn::Base64" : {
       "Fn::Join" : [ "", ["SignalURL=", { "Ref" : "myWaitHandle" } ] ]
   }
}
```

Note: In the AWS Management Console or the AWS CloudFormation command line tools, the presigned URL is displayed as the physical ID of the wait condition handle resource.

4. Select a method for detecting when the stack enters the wait condition.

   If you create the stack with notifications enabled, AWS CloudFormation publishes a notification for every stack event to the specified topic. If you or your application subscribe to that topic, you can monitor the notifications for the wait condition handle creation event and retrieve the presigned URL from the notification message.

   You can also monitor the stack's events using the AWS Management Console, the AWS CloudFormation command line tools, or the AWS CloudFormation API.

5. Use the presigned URL to signal success or failure.

   To send a signal, you send an HTTP request message using the presigned URL. The request method must be PUT and the Content-Type header must be an empty string or omitted. The request message must be a JSON structure of the form specified in Wait Condition Signal JSON Format (p. 231).

   You need to send the number of success signals specified by the Count property in order for AWS CloudFormation to continue stack creation. If you have a Count that is greater than 1, the UniqueId value for each signal must be unique across all signals sent to a particular wait condition. The UniqueId is an arbitrary alphanumerical string.

   A Curl command is one way to send a signal. The following example shows a Curl command line that signals success to a wait condition.

```
curl -T /tmp/a "https://cloudformation-waitcondition-test.s3.amazonaws.com/
arn%3Aaws%3Acloudformation%3Aus-east-1%3A034017226601%3Astack
%2Fstack-gosar-20110427004224-test-stack-with-WaitCondition--VEYW
%2Fe498ce60-70a1-11e0-81a7-5081d0136786%2FmyWaitConditionHandle?
Expires=1303976584&AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE&Signature=ik1twT6hpS4cgNAw7wyOoRejVoo
%3D"
```

   where the file /tmp/a contains the following JSON structure:

```
{
   "Status" : "SUCCESS",
   "Reason" : "Configuration Complete",
   "UniqueId" : "ID1234",
   "Data" : "Application has completed configuration."
}
```

   This example shows a Curl command line that sends the same success signal except it sends the JSON structure as a parameter on the command line.

```
curl -X PUT -H 'Content-Type:' --data-binary '{"Status" : "SUCCESS","Reason" :
 "Configuration Complete","UniqueId" : "ID1234","Data" : "Application
 has completed configuration."}' "https://cloudformation-waitcondition-
test.s3.amazonaws.com/arn%3Aaws%3Acloudformation%3Aus-east-1%3A034017226601%3Astack
%2Fstack-gosar-20110427004224-test-stack-with-WaitCondition--VEYW
%2Fe498ce60-70a1-11e0-81a7-5081d0136786%2FmyWaitConditionHandle?
Expires=1303976584&AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE&Signature=ik1twT6hpS4cgNAw7wyOoRejVoo
%3D"
```

Wait Condition Signal JSON Format

When you signal a wait condition, you must use the following JSON format:

```
{
  "Status" : "StatusValue",
  "UniqueId" : "Some UniqueId",
  "Data" : "Some Data",
  "Reason" : "Some Reason"
  }
```

Where:

StatusValue must be one of the following values:

- *SUCCESS* indicates a success signal.
- *FAILURE* indicates a failure signal and triggers a failed wait condition and a stack rollback.

UniqueId identifies the signal to AWS CloudFormation. If the Count property of the wait condition is greater than 1, the UniqueId value must be unique across all signals sent for a particular wait condition; otherwise, AWS CloudFormation will consider the signal a retransmission of the previously sent signal with the same UniqueId, and it will ignore the signal.

Data is any information that you want to send back with the signal. The Data value can be accessed by calling the Fn::GetAtt function (p. 1324) within the template. For example, if you create the following output value for the wait condition mywaitcondition, you can use the `aws cloudformation describe-stacks` command, DescribeStacks action, or Outputs tab of the CloudFormation console to view the Data sent by valid signals sent to AWS CloudFormation:

```
        "WaitConditionData" : {
            "Value" : { "Fn::GetAtt" : [ "mywaitcondition", "Data" ]},
            "Description" : "The data passed back as part of signalling the WaitCondition"
        },
```

The Fn::GetAtt function returns the UniqueId and Data as a name/value pair within a JSON structure. The following is an example of the Data attribute returned by the WaitConditionData output value defined above:

```
{"Signal1":"Application has completed configuration."}
```

Reason is a string with no other restrictions on its content besides JSON compliance.

Template Snippets

This section provides a number of example scenarios that you can use to understand how to declare various AWS CloudFormation template parts. You can also use the snippets as a starting point for sections of your custom templates.

Note
Because AWS CloudFormation templates must be JSON compliant, there is no provision for a line continuation character. The wrapping of the snippets in this document may be random if the line is longer that 80 characters.

Topics

General Template Snippets

The following examples show different AWS CloudFormation template features that aren't specific to an AWS service.

Topics

Base64 Encoded UserData Property

This example shows the assembly of a UserData property using the Fn::Base64 and Fn::Join functions. The references `MyValue` and `MyName` are parameters that must be defined in the Parameters section of the template. The literal string `Hello World` is just another value this example passes in as part of the `UserData`.

JSON

```
"UserData" : {
    "Fn::Base64" : {
        "Fn::Join" : [ ",", [
            { "Ref" : "MyValue" },
            { "Ref" : "MyName" },
            "Hello World" ] ]
    }
}
```

YAML

```
UserData:
  Fn::Base64: !Sub |
     Ref: MyValue
     Ref: MyName
     Hello World
```

Base64 Encoded UserData Property with AccessKey and SecretKey

This example shows the assembly of a UserData property using the Fn::Base64 and Fn::Join functions. It includes the `AccessKey` and `SecretKey` information. The references `AccessKey` and `SecretKey` are parameters that must be defined in the Parameters section of the template.

JSON

```
"UserData" : {
    "Fn::Base64" : {
        "Fn::Join" : [ "", [
            "ACCESS_KEY=", { "Ref" : "AccessKey" },
            "SECRET_KEY=", { "Ref" : "SecretKey" } ]
        ]
    }
}
```

YAML

```
UserData:
  Fn::Base64: !Sub |
     ACCESS_KEY=${AccessKey}
     SECRET_KEY=${SecretKey}
```

Parameters Section with One Literal String Parameter

The following example depicts a valid Parameters section declaration in which a single `String` type parameter is declared.

JSON

```
"Parameters" : {
    "UserName" : {
        "Type" : "String",
        "Default" : "nonadmin",
        "Description" : "Assume a vanilla user if no command-line spec provided"
    }
}
```

YAML

```
Parameters:
  UserName:
    Type: String
    Default: nonadmin
    Description: Assume a vanilla user if no command-line spec provided
```

Parameters Section with String Parameter with Regular Expression Constraint

The following example depicts a valid Parameters section declaration in which a single `String` type parameter is declared. The AdminUserAccount parameter has a default of admin. The parameter value must have a minimum length of 1, a maximum length of 16, and contains alphabetic characters and numbers but must begin with an alphabetic character.

JSON

```
"Parameters" : {
    "AdminUserAccount": {
      "Default": "admin",
      "NoEcho": "true",
      "Description" : "The admin account user name",
      "Type": "String",
      "MinLength": "1",
      "MaxLength": "16",
      "AllowedPattern" : "[a-zA-Z][a-zA-Z0-9]*"
    }
}
```

YAML

```
Parameters:
  AdminUserAccount:
    Default: admin
    NoEcho: true
    Description: The admin account user name
    Type: String
    MinLength: 1
    MaxLength: 16
    AllowedPattern: '[a-zA-Z][a-zA-Z0-9]*'
```

Parameters Section with Number Parameter with MinValue and MaxValue Constraints

The following example depicts a valid Parameters section declaration in which a single `Number` type parameter is declared. The WebServerPort parameter has a default of 80 and a minimum value 1 and maximum value 65535.

JSON

```
"Parameters" : {
    "WebServerPort": {
      "Default": "80",
      "Description" : "TCP/IP port for the web server",
      "Type": "Number",
      "MinValue": "1",
      "MaxValue": "65535"
    }
}
```

YAML

```
Parameters:
  WebServerPort:
    Default: 80
    Description: TCP/IP port for the web server
    Type: Number
    MinValue: 1
    MaxValue: 65535
```

Parameters Section with Number Parameter with AllowedValues Constraint

The following example depicts a valid Parameters section declaration in which a single `Number` type parameter is declared. The WebServerPort parameter has a default of 80 and allows only values of 80 and 8888.

JSON

```
"Parameters" : {
    "WebServerPortLimited": {
      "Default": "80",
      "Description" : "TCP/IP port for the web server",
      "Type": "Number",
      "AllowedValues" : ["80", "8888"]
    }
}
```

YAML

```
Parameters:
  WebServerPortLimited:
    Default: 80
    Description: TCP/IP port for the web server
    Type: Number
    AllowedValues:
    - 80
```

```
    - 8888
```

Parameters Section with One Literal CommaDelimitedList Parameter

The following example depicts a valid Parameters section declaration in which a single `CommaDelimitedList` type parameter is declared. The NoEcho property is set to `TRUE`, which will mask its value with asterisks (*****) in the `aws cloudformation describe-stacks` output.

JSON

```
"Parameters" : {
    "UserRoles" : {
        "Type" : "CommaDelimitedList",
        "Default" : "guest,newhire",
        "NoEcho" : "TRUE"
    }
}
```

YAML

```
Parameters:
  UserRoles:
    Type: CommaDelimitedList
    Default: "guest,newhire"
    NoEcho: true
```

Parameters Section with Parameter Value Based on Pseudo Parameter

The following example shows commands in the EC2 user data that use the pseudo parameters `AWS::StackName` and `AWS::Region`. For more information about pseudo parameters, see Pseudo Parameters Reference (p. 1351).

JSON

```
          "UserData"       : { "Fn::Base64" : { "Fn::Join" : ["", [
             "#!/bin/bash -xe\n",
             "yum install -y aws-cfn-bootstrap\n",

             "/opt/aws/bin/cfn-init -v ",
             "         --stack ", { "Ref" : "AWS::StackName" },
             "         --resource LaunchConfig ",
             "         --region ", { "Ref" : "AWS::Region" }, "\n",

             "/opt/aws/bin/cfn-signal -e $? ",
             "         --stack ", { "Ref" : "AWS::StackName" },
             "         --resource WebServerGroup ",
             "         --region ", { "Ref" : "AWS::Region" }, "\n"
        ]]}}
      }
```

YAML

```
UserData:
  Fn::Base64: !Sub |
```

```
     #!/bin/bash -xe
     yum update -y aws-cfn-bootstrap
     /opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --resource LaunchConfig --region
 ${AWS::Region}
     /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --resource WebServerGroup --
region ${AWS::Region}
```

Mapping Section with Three Mappings

The following example depicts a valid Mapping section declaration that contains three mappings. The map, when matched with a mapping key of `Stop`, `SlowDown`, or `Go`, provides the RGB values assigned to the corresponding `RGBColor` attribute.

JSON

```
"Mappings" : {
    "LightColor" : {
        "Stop" : {
            "Description" : "red",
            "RGBColor" : "RED 255 GREEN 0 BLUE 0"
        },
        "SlowDown" : {
            "Description" : "yellow",
            "RGBColor" : "RED 255 GREEN 255 BLUE 0"
        },
        "Go" : {
            "Description" : "green",
            "RGBColor" : "RED 0 GREEN 128 BLUE 0"
        }
    }
}
```

YAML

```
Mappings:
  LightColor:
    Stop:
      Description: red
      RGBColor: "RED 255 GREEN 0 BLUE 0"
    SlowDown:
      Description: yellow
      RGBColor: "RED 255 GREEN 255 BLUE 0"
    Go:
      Description: green
      RGBColor: "RED 0 GREEN 128 BLUE 0"
```

Description Based on Literal String

The following example depicts a valid Description section declaration where the value is based on a literal string. This snippet can be for templates, parameters, resources, properties, or outputs.

JSON

```
"Description" : "Replace this value"
```

YAML

```
Description: "Replace this value"
```

Outputs Section with One Literal String Output

This example shows a output assignment based on a literal string.

JSON

```
"Outputs" : {
    "MyPhone" : {
        "Value" : "Please call 555-5555",
        "Description" : "A random message for aws cloudformation describe-stacks"
    }
}
```

YAML

```
Outputs:
  MyPhone:
    Value: Please call 555-5555
    Description: A random message for aws cloudformation describe-stacks
```

Outputs Section with One Resource Reference and One Pseudo Reference Output

This example shows an Outputs section with two output assignments. One is based on a resource, and the other is based on a pseudo reference.

JSON

```

"Outputs" : {
   "SNSTopic" : { "Value" : { "Ref" : "MyNotificationTopic" } },
   "StackName" : { "Value" : { "Ref" : "AWS::StackName" } }
}
```

YAML

```
Outputs:
  SNSTopic:
    Value: Ref: MyNotificationTopic
  StackName:
    Value: Ref: AWS::StackName
```

Outputs Section with an Output Based on a Function, a Literal String, a Reference, and a Pseudo Parameter

This example shows an Outputs section with one output assignment. The Join function is used to concatenate the value, using a percent sign as the delimiter.

JSON

```
"Outputs" : {
    "MyOutput" : {
        "Value" : { "Fn::Join" :
            [ "%", [ "A-string", {"Ref" : "AWS::StackName" } ] ]
```

```
        }
    }
}
```

YAML

```
Outputs:
  MyOutput:
    Value: !Join [ %, [ 'A-string', !Ref 'AWS::StackName' ]]
```

Template Format Version

The following snippet depicts a valid Template Format Version section declaration.

JSON

```
"AWSTemplateFormatVersion" : "2010-09-09"
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
```

AWS Tag Property

This example shows an AWS Tag property. You would specify this property within the Properties section of a resource. When the resource is created, it will be tagged with the tags you declare.

JSON

```
"Tags" : [
      {
        "Key" : "keyname1",
        "Value" : "value1"
      },
      {
        "Key" : "keyname2",
        "Value" : "value2"
      }
    ]
}
```

YAML

```
Tags:
  -
    Key: "keyname1"
    Value: "value1"
  -
    Key: "keyname2"
    Value: "value2"
```

Auto Scaling Template Snippets

Topics

Auto Scaling Launch Configuration Resource

This example shows an Auto Scaling AWS::AutoScaling::LaunchConfiguration resource. The SecurityGroups property specifies both an AWS::EC2::SecurityGroup resource named myEC2SecurityGroup and an existing EC2 security group named myExistingEC2SecurityGroup. The BlockDeviceMappings property lists two devices: a 50 gigabyte EBS volume mapped to /dev/sdk and a virtual device ephemeral0 mapped to /dev/sdc.

JSON

```
"SimpleConfig" : {
   "Type" : "AWS::AutoScaling::LaunchConfiguration",
   "Properties" : {
      "ImageId" : "ami-6411e20d",
      "SecurityGroups" : [ { "Ref" : "myEC2SecurityGroup" },
 "myExistingEC2SecurityGroup" ],
      "InstanceType" : "m1.small",
      "BlockDeviceMappings" : [ {
            "DeviceName" : "/dev/sdk",
            "Ebs" : {"VolumeSize" : "50"}
         }, {
            "DeviceName" : "/dev/sdc",
            "VirtualName" : "ephemeral0"
      } ]
   }
}
```

YAML

```
SimpleConfig:
  Type: AWS::AutoScaling::LaunchConfiguration
  Properties:
    ImageId: ami-6411e20d
    SecurityGroups:
    - Ref: myEC2SecurityGroup
    - myExistingEC2SecurityGroup
    InstanceType: m1.small
    BlockDeviceMappings:
    - DeviceName: "/dev/sdk"
      Ebs:
        VolumeSize: '50'
    - DeviceName: "/dev/sdc"
      VirtualName: ephemeral0
```

Auto Scaling Group Resource

This example shows an Auto Scaling AWS::AutoScaling::AutoScalingGroup (p. 433) resource. The AvailabilityZones property specifies the availability zones where the auto-scaling group's EC2 instances will be created. In this example, the Fn::GetAZs (p. 1332) function call `{ "Fn::GetAZs" : "" }` specifies all availability zones for the region in which the stack is created. The LoadBalancerNames property lists the LoadBalancers used to route traffic to the Auto Scaling group. In this example, one LoadBalancer is specified, the AWS::ElasticLoadBalancing::LoadBalancer (p. 719) resource LB.

JSON

```
"MyServerGroup" : {
   "Type" : "AWS::AutoScaling::AutoScalingGroup",
   "Properties" : {
      "AvailabilityZones" : { "Fn::GetAZs" : ""},
      "LaunchConfigurationName" : { "Ref" : "SimpleConfig" },
      "MinSize" : "1",
      "MaxSize" : "3",
      "LoadBalancerNames" : [ { "Ref" : "LB" } ]
   }
}
```

YAML

```
MyServerGroup:
  Type: AWS::AutoScaling::AutoScalingGroup
  Properties:
    AvailabilityZones:
      Fn::GetAZs: ''
    LaunchConfigurationName:
      Ref: SimpleConfig
    MinSize: '1'
    MaxSize: '3'
    LoadBalancerNames:
    - Ref: LB
```

Auto Scaling Policy Triggered by CloudWatch Alarm

This example shows an AWS::AutoScaling::ScalingPolicy (p. 452) resource that scales up the Auto Scaling group asGroup. The `AdjustmentType` property specifies ChangeInCapacity, which means that the `ScalingAdjustment` represents the number of instances to add (if `ScalingAdjustment` is positive) or delete (if it is negative). In this example, `ScalingAdjustment` is 1; therefore, the policy increments the number of EC2 instances in the group by 1 when the policy is executed.

The AWS::CloudWatch::Alarm (p. 498) resource CPUAlarmHigh specifies the scaling policy ScaleUpPolicy as the action to execute when the alarm is in an `ALARM` state (`AlarmActions`).

JSON

```
"ScaleUpPolicy" : {
   "Type" : "AWS::AutoScaling::ScalingPolicy",
   "Properties" : {
      "AdjustmentType" : "ChangeInCapacity",
      "AutoScalingGroupName" : { "Ref" : "asGroup" },
      "Cooldown" : "1",
      "ScalingAdjustment" : "1"
   }
},
"CPUAlarmHigh": {
   "Type": "AWS::CloudWatch::Alarm",
   "Properties": {
      "EvaluationPeriods": "1",
```

```
      "Statistic": "Average",
      "Threshold": "10",
      "AlarmDescription": "Alarm if CPU too high or metric disappears indicating instance
 is down",
      "Period": "60",
      "AlarmActions": [ { "Ref": "ScaleUpPolicy" } ],
      "Namespace": "AWS/EC2",
      "Dimensions": [ {
         "Name": "AutoScalingGroupName",
         "Value": { "Ref": "asGroup" }
      } ],
      "ComparisonOperator": "GreaterThanThreshold",
      "MetricName": "CPUUtilization"
   }
}
```

YAML

```
ScaleUpPolicy:
  Type: AWS::AutoScaling::ScalingPolicy
  Properties:
    AdjustmentType: ChangeInCapacity
    AutoScalingGroupName:
      Ref: asGroup
    Cooldown: '1'
    ScalingAdjustment: '1'
CPUAlarmHigh:
  Type: AWS::CloudWatch::Alarm
  Properties:
    EvaluationPeriods: '1'
    Statistic: Average
    Threshold: '10'
    AlarmDescription: Alarm if CPU too high or metric disappears indicating instance
      is down
    Period: '60'
    AlarmActions:
    - Ref: ScaleUpPolicy
    Namespace: AWS/EC2
    Dimensions:
    - Name: AutoScalingGroupName
      Value:
        Ref: asGroup
    ComparisonOperator: GreaterThanThreshold
    MetricName: CPUUtilization
```

Auto Scaling Group with Notifications

This example shows an AWS::AutoScaling::AutoScalingGroup (p. 433) resource that sends Amazon SNS notifications when the specified events take place. The `NotificationConfigurations` property specifies the SNS topic where AWS CloudFormation sends a notification and the events that will cause AWS CloudFormation to send notifications. When the events specified by `NotificationTypes` occur, AWS CloudFormation will send a notification to the SNS topic specified by `TopicARN`. In this example, AWS CloudFormation sends a notification to the SNS topic topic1 when the `autoscaling:EC2_INSTANCE_LAUNCH` and `autoscaling:EC2_INSTANCE_LAUNCH_ERROR` events occur.

JSON

```
"MyAsGroupWithNotification" : {
  "Type" : "AWS::AutoScaling::AutoScalingGroup",
  "Properties" : {
    "AvailabilityZones" : { "Ref" : "azList" },
```

```
    "LaunchConfigurationName" : { "Ref" : "myLCOne" },
    "MinSize" : "0",
    "MaxSize" : "2",
    "DesiredCapacity" : "1",
    "NotificationConfigurations" : [
      {
        "TopicARN" : { "Ref" : "topic1" },
        "NotificationTypes" : [
          "autoscaling:EC2_INSTANCE_LAUNCH",
          "autoscaling:EC2_INSTANCE_LAUNCH_ERROR",
          "autoscaling:EC2_INSTANCE_TERMINATE",
          "autoscaling:EC2_INSTANCE_TERMINATE_ERROR"
        ]
      }
    ]
  }
}
```

YAML

```
MyAsGroupWithNotification:
  Type: AWS::AutoScaling::AutoScalingGroup
  Properties:
    AvailabilityZones:
      Ref: azList
    LaunchConfigurationName:
      Ref: myLCOne
    MinSize: '0'
    MaxSize: '2'
    DesiredCapacity: '1'
    NotificationConfigurations:
    - TopicARN:
        Ref: topic1
      NotificationTypes:
      - autoscaling:EC2_INSTANCE_LAUNCH
      - autoscaling:EC2_INSTANCE_LAUNCH_ERROR
      - autoscaling:EC2_INSTANCE_TERMINATE
      - autoscaling:EC2_INSTANCE_TERMINATE_ERROR
```

Auto Scaling with an UpdatePolicy

This example shows how to use an UpdatePolicy (p. 1303) with an auto-scaling group.

JSON

```
"ASG1" : {
   "UpdatePolicy" : {
      "AutoScalingRollingUpdate" : {
         "MinInstancesInService" : "1",
         "MaxBatchSize" : "1",
         "PauseTime" : "PT12M5S"
      }
   },
   "Type" : "AWS::AutoScaling::AutoScalingGroup",
   "Properties" : {
      "AvailabilityZones" : { "Fn::GetAZs" : { "Ref" : "AWS::Region" } },
      "LaunchConfigurationName" : { "Ref" : "ASLC" },
      "MaxSize" : "3",
      "MinSize" : "1"
   }
```

```
}
```

YAML

```
ASG1:
  UpdatePolicy:
    AutoScalingRollingUpdate:
      MinInstancesInService: '1'
      MaxBatchSize: '1'
      PauseTime: PT12M5S
  Type: AWS::AutoScaling::AutoScalingGroup
  Properties:
    AvailabilityZones:
      Fn::GetAZs:
        Ref: AWS::Region
    LaunchConfigurationName:
      Ref: ASLC
    MaxSize: '3'
    MinSize: '1'
```

AWS CloudFormation Template Snippets

Topics

Nested Stacks

Nesting a Stack in a Template

This example template contains a nested stack resource called `myStack`. When AWS CloudFormation creates a stack from the template, it creates the `myStack`, whose template is specified in the `TemplateURL` property. The output value `StackRef` returns the stack ID for `myStack` and the value `OutputFromNestedStack` returns the output value `BucketName` from within the `myStack` resource. The Outputs.*`nestedstackoutputname`* format is reserved for specifying output values from nested stacks and can be used anywhere within the containing template.

For more information, see AWS::CloudFormation::Stack (p. 485).

JSON

```
{
    "AWSTemplateFormatVersion" : "2010-09-09",
    "Resources" : {
        "myStack" : {
        "Type" : "AWS::CloudFormation::Stack",
        "Properties" : {
           "TemplateURL" : "https://s3.amazonaws.com/cloudformation-templates-us-east-1/
S3_Bucket.template",
              "TimeoutInMinutes" : "60"
        }
        }
    },
    "Outputs": {
       "StackRef": {"Value": { "Ref" : "myStack"}},
       "OutputFromNestedStack" : {
             "Value" : { "Fn::GetAtt" : [ "myStack", "Outputs.BucketName" ] }
```

```
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  myStack:
    Type: AWS::CloudFormation::Stack
    Properties:
      TemplateURL: https://s3.amazonaws.com/cloudformation-templates-us-east-1/
S3_Bucket.template
      TimeoutInMinutes: '60'
Outputs:
  StackRef:
    Value: !Ref myStack
  OutputFromNestedStack:
    Value: !GetAtt myStack.Outputs.BucketName
```

Nesting a Stack with Input Parameters in a Template

This example template contains a stack resource that specifies input parameters. When AWS CloudFormation creates a stack from this template, it uses the value pairs declared within the Parameters property as the input parameters for the template used to create the `myStackWithParams` stack. In this example, the `InstanceType` and `KeyName` parameters are specified.

For more information, see AWS::CloudFormation::Stack (p. 485).

JSON

```
{
    "AWSTemplateFormatVersion" : "2010-09-09",
    "Resources" : {
        "myStackWithParams" : {
          "Type" : "AWS::CloudFormation::Stack",
        "Properties" : {
            "TemplateURL" : "https://s3.amazonaws.com/cloudformation-templates-us-east-1/
EC2ChooseAMI.template",
            "Parameters" : {
                "InstanceType" : "t1.micro",
                "KeyName" : "mykey"
            }
           }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  myStackWithParams:
    Type: AWS::CloudFormation::Stack
    Properties:
      TemplateURL: https://s3.amazonaws.com/cloudformation-templates-us-east-1/
EC2ChooseAMI.template
      Parameters:
        InstanceType: t1.micro
        KeyName: mykey
```

Wait Condition

Using a Wait Condition with an Amazon EC2 Instance

Important
For Amazon EC2 and Auto Scaling resources, we recommend that you use a CreationPolicy attribute instead of wait conditions. Add a CreationPolicy attribute to those resources, and use the cfn-signal helper script to signal when an instance creation process has completed successfully.

If you can't use a creation policy, you view the following example template, which declares an Amazon EC2 instance with a wait condition. The wait condition myWaitCondition uses myWaitConditionHandle for signaling, uses the DependsOn attribute (p. 1298) to specify that the wait condition will trigger after the Amazon EC2 instance resource has been created, and uses the Timeout property to specify a duration of 4500 seconds for the wait condition. In addition, the presigned URL that signals the wait condition is passed to the Amazon EC2 instance with the UserData property of the Ec2Instance resource, thus enabling an application or script running on that Amazon EC2 instance to retrieve the pre-signed URL and employ it to signal a success or failure to the wait condition. Note that you need to create the application or script that signals the wait condition. The output value ApplicationData contains the data passed back from the wait condition signal.

For more information, see Creating Wait Conditions in a Template (p. 228), AWS::CloudFormation::WaitCondition (p. 487), and AWS::CloudFormation::WaitConditionHandle (p. 491).

JSON

```
{
    "AWSTemplateFormatVersion" : "2010-09-09",
    "Mappings" : {
        "RegionMap" : {
            "us-east-1" : {
                "AMI" : "ami-76f0061f"
            },
            "us-west-1" : {
                "AMI" : "ami-655a0a20"
            },
            "eu-west-1" : {
                "AMI" : "ami-7fd4e10b"
            },
            "ap-northeast-1" : {
                "AMI" : "ami-8e08a38f"
            },
            "ap-southeast-1" : {
                "AMI" : "ami-72621c20"
            }
        }
    },
    "Resources" : {
        "Ec2Instance" : {
            "Type" : "AWS::EC2::Instance",
            "Properties" : {
                "UserData" : { "Fn::Base64" : {"Ref" : "myWaitHandle"}},
                "ImageId" : { "Fn::FindInMap" : [ "RegionMap", { "Ref" : "AWS::Region" },
 "AMI" ]}
            }
        },
        "myWaitHandle" : {
            "Type" : "AWS::CloudFormation::WaitConditionHandle",
            "Properties" : {
            }
        },
        "myWaitCondition" : {
```

```
            "Type" : "AWS::CloudFormation::WaitCondition",
            "DependsOn" : "Ec2Instance",
            "Properties" : {
                "Handle" : { "Ref" : "myWaitHandle" },
                "Timeout" : "4500"
            }
        }
    },
    "Outputs" : {
        "ApplicationData" : {
            "Value" : { "Fn::GetAtt" : [ "myWaitCondition", "Data" ]},
            "Description" : "The data passed back as part of signalling the WaitCondition."
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Mappings:
  RegionMap:
    us-east-1:
      AMI: ami-76f0061f
    us-west-1:
      AMI: ami-655a0a20
    eu-west-1:
      AMI: ami-7fd4e10b
    ap-northeast-1:
      AMI: ami-8e08a38f
    ap-southeast-1:
      AMI: ami-72621c20
Resources:
  Ec2Instance:
    Type: AWS::EC2::Instance
    Properties:
      UserData:
        Fn::Base64: !Ref myWaitHandle
      ImageId:
        Fn::FindInMap:
        - RegionMap
        - Ref: AWS::Region
        - AMI
  myWaitHandle:
    Type: AWS::CloudFormation::WaitConditionHandle
    Properties: {}
  myWaitCondition:
    Type: AWS::CloudFormation::WaitCondition
    DependsOn: Ec2Instance
    Properties:
      Handle: !Ref myWaitHandle
      Timeout: '4500'
Outputs:
  ApplicationData:
    Value: !GetAtt myWaitCondition.Data
    Description: The data passed back as part of signalling the WaitCondition.
```

Using Curl to signal a Wait Condition

This example shows a Curl command line that signals success to a wait condition.

```
curl -T /tmp/a "https://cloudformation-waitcondition-test.s3.amazonaws.com/
arn%3Aaws%3Acloudformation%3Aus-east-1%3A034017226601%3Astack
%2Fstack-gosar-20110427004224-test-stack-with-WaitCondition--VEYW
```

```
%2Fe498ce60-70a1-11e0-81a7-5081d0136786%2FmyWaitConditionHandle?
Expires=1303976584&AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE&Signature=ik1twT6hpS4cgNAw7wyOoRejVoo
%3D"
```

where the file /tmp/a contains the following JSON structure:

```
{
  "Status" : "SUCCESS",
  "Reason" : "Configuration Complete",
  "UniqueId" : "ID1234",
  "Data" : "Application has completed configuration."
}
```

This example shows a Curl command line that sends the same success signal except it sends the JSON as a parameter on the command line.

```
curl -X PUT -H 'Content-Type:' --data-binary '{"Status" : "SUCCESS","Reason" :
 "Configuration Complete","UniqueId" : "ID1234","Data" : "Application
 has completed configuration."}' "https://cloudformation-waitcondition-
test.s3.amazonaws.com/arn%3Aaws%3Acloudformation%3Aus-east-1%3A034017226601%3Astack
%2Fstack-gosar-20110427004224-test-stack-with-WaitCondition--VEYW
%2Fe498ce60-70a1-11e0-81a7-5081d0136786%2FmyWaitConditionHandle?
Expires=1303976584&AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE&Signature=ik1twT6hpS4cgNAw7wyOoRejVoo
%3D"
```

Amazon CloudFront Template Snippets

Topics

Amazon CloudFront Distribution Resource with an Amazon S3 Origin

The following example template shows an Amazon CloudFront Distribution (p. 492) using an S3Origin (p. 1053).

JSON

```
{
    "AWSTemplateFormatVersion" : "2010-09-09",
    "Resources" : {
        "myDistribution" : {
            "Type" : "AWS::CloudFront::Distribution",
            "Properties" : {
                "DistributionConfig" : {
                    "Origins" : [ {
                        "DomainName" : "mybucket.s3.amazonaws.com",
                        "Id" : "myS3Origin",
                        "S3OriginConfig" : {
                            "OriginAccessIdentity" : "origin-access-identity/cloudfront/
E127EXAMPLE51Z"
                        }
                    }],
                    "Enabled" : "true",
```

```
                    "Comment" : "Some comment",
                    "DefaultRootObject" : "index.html",
                    "Logging" : {
                        "IncludeCookies" : "false",
                        "Bucket" : "mylogs.s3.amazonaws.com",
                        "Prefix" : "myprefix"
                    },
                    "Aliases" : [ "mysite.example.com", "yoursite.example.com" ],
                    "DefaultCacheBehavior" : {
                        "AllowedMethods" : [ "DELETE", "GET", "HEAD", "OPTIONS", "PATCH",
 "POST", "PUT" ],
                        "TargetOriginId" : "myS3Origin",
                        "ForwardedValues" : {
                            "QueryString" : "false",
                            "Cookies" : { "Forward" : "none" }
                        },
                        "TrustedSigners" : [ "1234567890EX", "1234567891EX" ],
                        "ViewerProtocolPolicy" : "allow-all"
                    },
                   "PriceClass" : "PriceClass_200",
                   "Restrictions" : {
                       "GeoRestriction" : {
                           "RestrictionType" : "whitelist",
                           "Locations" : [ "AQ", "CV" ]
                       }
                   },
                   "ViewerCertificate" : { "CloudFrontDefaultCertificate" : "true" }
                }
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  myDistribution:
    Type: AWS::CloudFront::Distribution
    Properties:
      DistributionConfig:
        Origins:
        - DomainName: mybucket.s3.amazonaws.com
          Id: myS3Origin
          S3OriginConfig:
            OriginAccessIdentity: origin-access-identity/cloudfront/E127EXAMPLE51Z
        Enabled: 'true'
        Comment: Some comment
        DefaultRootObject: index.html
        Logging:
          IncludeCookies: 'false'
          Bucket: mylogs.s3.amazonaws.com
          Prefix: myprefix
        Aliases:
        - mysite.example.com
        - yoursite.example.com
        DefaultCacheBehavior:
          AllowedMethods:
          - DELETE
          - GET
          - HEAD
          - OPTIONS
          - PATCH
          - POST
```

```
          - PUT
          TargetOriginId: myS3Origin
          ForwardedValues:
            QueryString: 'false'
            Cookies:
              Forward: none
          TrustedSigners:
          - 1234567890EX
          - 1234567891EX
          ViewerProtocolPolicy: allow-all
        PriceClass: PriceClass_200
        Restrictions:
          GeoRestriction:
            RestrictionType: whitelist
            Locations:
            - AQ
            - CV
        ViewerCertificate:
          CloudFrontDefaultCertificate: 'true'
```

Amazon CloudFront Distribution Resource with Custom Origin

The following example template shows an Amazon CloudFront Distribution (p. 492) using a CustomOrigin (p. 1051).

JSON

```
{
    "AWSTemplateFormatVersion" : "2010-09-09",
    "Resources" : {
        "myDistribution" : {
            "Type" : "AWS::CloudFront::Distribution",
            "Properties" : {
                "DistributionConfig" : {
                    "Origins" : [ {
                            "DomainName" : "www.example.com",
                            "Id" : "myCustomOrigin",
                            "CustomOriginConfig" : {
                                "HTTPPort" : "80",
                                "HTTPSPort" : "443",
                                "OriginProtocolPolicy" : "http-only"
                            }
                    } ],
                    "Enabled" : "true",
                    "Comment" : "Somecomment",
                    "DefaultRootObject" : "index.html",
                    "Logging" : {
                        "IncludeCookies" : "true",
                        "Bucket" : "mylogs.s3.amazonaws.com",
                        "Prefix": "myprefix"
                    },
                    "Aliases" : [
                        "mysite.example.com",
                        "*.yoursite.example.com"
                    ],
                    "DefaultCacheBehavior" : {
                        "TargetOriginId" : "myCustomOrigin",
                        "SmoothStreaming" : "false",
                        "ForwardedValues" : {
                            "QueryString" : "false",
                            "Cookies" : { "Forward" : "all" }
                        },
                        "TrustedSigners" : [
```

```
                            "1234567890EX",
                            "1234567891EX"
                        ],
                        "ViewerProtocolPolicy" : "allow-all"
                    },
                    "CustomErrorResponses" : [ {
                        "ErrorCode" : "404",
                        "ResponsePagePath" : "/error-pages/404.html",
                        "ResponseCode" : "200",
                        "ErrorCachingMinTTL" : "30"
                    } ],
                   "PriceClass" : "PriceClass_200",
                   "Restrictions" : {
                       "GeoRestriction" : {
                           "RestrictionType" : "whitelist",
                           "Locations" : [ "AQ", "CV" ]
                       }
                   },
                   "ViewerCertificate": { "CloudFrontDefaultCertificate" : "true" }
                }
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  myDistribution:
    Type: 'AWS::CloudFront::Distribution'
    Properties:
      DistributionConfig:
        Origins:
        - DomainName: www.example.com
          Id: myCustomOrigin
          CustomOriginConfig:
            HTTPPort: '80'
            HTTPSPort: '443'
            OriginProtocolPolicy: http-only
        Enabled: 'true'
        Comment: Somecomment
        DefaultRootObject: index.html
        Logging:
          IncludeCookies: 'true'
          Bucket: mylogs.s3.amazonaws.com
          Prefix: myprefix
        Aliases:
        - mysite.example.com
        - "*.yoursite.example.com"
        DefaultCacheBehavior:
          TargetOriginId: myCustomOrigin
          SmoothStreaming: 'false'
          ForwardedValues:
            QueryString: 'false'
            Cookies:
              Forward: all
          TrustedSigners:
          - 1234567890EX
          - 1234567891EX
          ViewerProtocolPolicy: allow-all
        CustomErrorResponses:
        - ErrorCode: '404'
          ResponsePagePath: "/error-pages/404.html"
```

```
          ResponseCode: '200'
          ErrorCachingMinTTL: '30'
        PriceClass: PriceClass_200
        Restrictions:
          GeoRestriction:
            RestrictionType: whitelist
            Locations:
            - AQ
            - CV
        ViewerCertificate:
          CloudFrontDefaultCertificate: 'true'
```

Amazon CloudFront Distribution with Multi-origin Support.

The following example template shows how to declare a CloudFront Distribution (p. 492) with multi-origin support. In the DistributionConfig (p. 1040), a list of origins is provided and a DefaultCacheBehavior (p. 1046) is set.

JSON

```
{
    "AWSTemplateFormatVersion" : "2010-09-09",
    "Resources" : {
        "myDistribution" : {
            "Type" : "AWS::CloudFront::Distribution",
            "Properties" : {
                "DistributionConfig" : {
                    "Origins" : [ {
                        "Id" : "myS3Origin",
                        "DomainName" : "mybucket.s3.amazonaws.com",
                        "S3OriginConfig" : {
                            "OriginAccessIdentity" : "origin-access-identity/cloudfront/
E127EXAMPLE51Z"
                        }
                     },
                     {
                         "Id" : "myCustomOrigin",
                         "DomainName" : "www.example.com",
                         "CustomOriginConfig" : {
                             "HTTPPort" : "80",
                             "HTTPSPort" : "443",
                             "OriginProtocolPolicy" : "http-only"
                         }
                     }
                   ],
                   "Enabled" : "true",
                   "Comment" : "Some comment",
                   "DefaultRootObject" : "index.html",
                   "Logging" : {
                       "IncludeCookies" : "true",
                       "Bucket" : "mylogs.s3.amazonaws.com",
                       "Prefix" : "myprefix"
                   },
                   "Aliases" : [ "mysite.example.com", "yoursite.example.com" ],
                   "DefaultCacheBehavior" : {
                       "TargetOriginId" : "myS3Origin",
                       "ForwardedValues" : {
                           "QueryString" : "false",
                           "Cookies" : { "Forward" : "all" }
                        },
                       "TrustedSigners" : [ "1234567890EX", "1234567891EX"  ],
                       "ViewerProtocolPolicy" : "allow-all",
                       "MinTTL" : "100",
```

```
                    "SmoothStreaming" : "true"
                },
                "CacheBehaviors" : [ {
                         "AllowedMethods" : [ "DELETE", "GET", "HEAD", "OPTIONS",
 "PATCH", "POST", "PUT" ],
                         "TargetOriginId" : "myS3Origin",
                         "ForwardedValues" : {
                             "QueryString" : "true",
                             "Cookies" : { "Forward" : "none" }
                         },
                         "TrustedSigners" : [ "1234567890EX", "1234567891EX" ],
                         "ViewerProtocolPolicy" : "allow-all",
                         "MinTTL" : "50",
                         "PathPattern" : "images1/*.jpg"
                     },
                     {
                         "AllowedMethods" : [ "DELETE", "GET", "HEAD", "OPTIONS",
 "PATCH", "POST", "PUT" ],
                         "TargetOriginId" : "myCustomOrigin",
                         "ForwardedValues" : {
                             "QueryString" : "true",
                             "Cookies" : { "Forward" : "none" }
                         },
                         "TrustedSigners" : [ "1234567890EX", "1234567891EX"  ],
                         "ViewerProtocolPolicy" : "allow-all",
                         "MinTTL" : "50",
                         "PathPattern" : "images2/*.jpg"
                     }
                ],
                "CustomErrorResponses" : [ {
                    "ErrorCode" : "404",
                    "ResponsePagePath" : "/error-pages/404.html",
                    "ResponseCode" : "200",
                    "ErrorCachingMinTTL" : "30"
                } ],
                "PriceClass" : "PriceClass_All",
                "ViewerCertificate" : { "CloudFrontDefaultCertificate" : "true" }
            }
        }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  myDistribution:
    Type: AWS::CloudFront::Distribution
    Properties:
      DistributionConfig:
        Origins:
        - Id: myS3Origin
          DomainName: mybucket.s3.amazonaws.com
          S3OriginConfig:
            OriginAccessIdentity: origin-access-identity/cloudfront/E127EXAMPLE51Z
        - Id: myCustomOrigin
          DomainName: www.example.com
          CustomOriginConfig:
            HTTPPort: '80'
            HTTPSPort: '443'
            OriginProtocolPolicy: http-only
        Enabled: 'true'
        Comment: Some comment
```

```
        DefaultRootObject: index.html
        Logging:
          IncludeCookies: 'true'
          Bucket: mylogs.s3.amazonaws.com
          Prefix: myprefix
        Aliases:
        - mysite.example.com
        - yoursite.example.com
        DefaultCacheBehavior:
          TargetOriginId: myS3Origin
          ForwardedValues:
            QueryString: 'false'
            Cookies:
              Forward: all
          TrustedSigners:
          - 1234567890EX
          - 1234567891EX
          ViewerProtocolPolicy: allow-all
          MinTTL: '100'
          SmoothStreaming: 'true'
        CacheBehaviors:
        - AllowedMethods:
          - DELETE
          - GET
          - HEAD
          - OPTIONS
          - PATCH
          - POST
          - PUT
          TargetOriginId: myS3Origin
          ForwardedValues:
            QueryString: 'true'
            Cookies:
              Forward: none
          TrustedSigners:
          - 1234567890EX
          - 1234567891EX
          ViewerProtocolPolicy: allow-all
          MinTTL: '50'
          PathPattern: images1/*.jpg
        - AllowedMethods:
          - DELETE
          - GET
          - HEAD
          - OPTIONS
          - PATCH
          - POST
          - PUT
          TargetOriginId: myCustomOrigin
          ForwardedValues:
            QueryString: 'true'
            Cookies:
              Forward: none
          TrustedSigners:
          - 1234567890EX
          - 1234567891EX
          ViewerProtocolPolicy: allow-all
          MinTTL: '50'
          PathPattern: images2/*.jpg
        CustomErrorResponses:
        - ErrorCode: '404'
          ResponsePagePath: "/error-pages/404.html"
          ResponseCode: '200'
          ErrorCachingMinTTL: '30'
        PriceClass: PriceClass_All
        ViewerCertificate:
```

```
        CloudFrontDefaultCertificate: 'true'
```

Amazon CloudWatch Template Snippets

Topics

Billing Alarm

In the following sample, CloudWatch sends an email notification when charges to your AWS account exceed the alarm threshold. Note that you'll need to enable billing alerts to receive notifications about your usage.

JSON

```
"SpendingAlarm": {
  "Type": "AWS::CloudWatch::Alarm",
  "Properties": {
    "AlarmDescription": { "Fn::Join": ["", [
      "Alarm if AWS spending is over $",
      { "Ref": "AlarmThreshold" }
    ]]},
    "Namespace": "AWS/Billing",
    "MetricName": "EstimatedCharges",
    "Dimensions": [{
      "Name": "Currency",
      "Value" : "USD"
    }],
    "Statistic": "Maximum",
    "Period": "21600",
    "EvaluationPeriods": "1",
    "Threshold": { "Ref": "AlarmThreshold" },
    "ComparisonOperator": "GreaterThanThreshold",
    "AlarmActions": [{
      "Ref": "BillingAlarmNotification"
    }],
    "InsufficientDataActions": [{
      "Ref": "BillingAlarmNotification"
    }]
  }
}
```

YAML

```
SpendingAlarm:
  Type: AWS::CloudWatch::Alarm
  Properties:
    AlarmDescription: !Sub >
      "Alarm if AWS spending is over $${AlarmThreshold}"
    Namespace: AWS/Billing
    MetricName: EstimatedCharges
    Dimensions:
    - Name: Currency
      Value: USD
    Statistic: Maximum
    Period: '21600'
```

```
    EvaluationPeriods: '1'
    Threshold:
      Ref: "AlarmThreshold"
    ComparisonOperator: GreaterThanThreshold
    AlarmActions:
    - Ref: "BillingAlarmNotification"
    InsufficientDataActions:
    - Ref: "BillingAlarmNotification"
```

CPU Utilization Alarm

The following sample snippet creates an alarm that sends a notification when the average CPU utilization of an Amazon EC2 instance exceeds 90 percent for more than 60 seconds over three evaluation periods.

JSON

```
"CPUAlarm" : {
  "Type" : "AWS::CloudWatch::Alarm",
  "Properties" : {
    "AlarmDescription" : "CPU alarm for my instance",
    "AlarmActions" : [ { "Ref" : "logical name of an AWS::SNS::Topic resource" } ],
    "MetricName" : "CPUUtilization",
    "Namespace" : "AWS/EC2",
    "Statistic" : "Average",
    "Period" : "60",
    "EvaluationPeriods" : "3",
    "Threshold" : "90",
    "ComparisonOperator" : "GreaterThanThreshold",
    "Dimensions" : [ {
      "Name" : "InstanceId",
      "Value" : { "Ref" : "logical name of an AWS::EC2::Instance resource" }
    } ]
  }
}
```

YAML

```
CPUAlarm:
  Type: AWS::CloudWatch::Alarm
  Properties:
    AlarmDescription: CPU alarm for my instance
    AlarmActions:
    - Ref: "logical name of an AWS::SNS::Topic resource"
    MetricName: CPUUtilization
    Namespace: AWS/EC2
    Statistic: Average
    Period: '60'
    EvaluationPeriods: '3'
    Threshold: '90'
    ComparisonOperator: GreaterThanThreshold
    Dimensions:
    - Name: InstanceId
      Value:
        Ref: "logical name of an AWS::EC2::Instance resource"
```

Recover an Amazon Elastic Compute Cloud instance

The following CloudWatch alarm recovers an EC2 instance when it has any status check failures for 15 consecutive minutes. For more information about alarm actions, see Create Alarms That Stop, Terminate, or Recover an Instance in the *Amazon CloudWatch User Guide*.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters" : {
    "RecoveryInstance" : {
      "Description" : "The EC2 instance ID to associate this alarm with.",
      "Type" : "AWS::EC2::Instance::Id"
    }
  },
  "Resources": {
    "RecoveryTestAlarm": {
      "Type": "AWS::CloudWatch::Alarm",
      "Properties": {
        "AlarmDescription": "Trigger a recovery when instance status check fails for 15
 consecutive minutes.",
        "Namespace": "AWS/EC2" ,
        "MetricName": "StatusCheckFailed_System",
        "Statistic": "Minimum",
        "Period": "60",
        "EvaluationPeriods": "15",
        "ComparisonOperator": "GreaterThanThreshold",
        "Threshold": "0",
        "AlarmActions": [ {"Fn::Join" : ["", ["arn:aws:automate:", { "Ref" :
 "AWS::Region" }, ":ec2:recover" ]]} ],
        "Dimensions": [{"Name": "InstanceId","Value": {"Ref": "RecoveryInstance"}}]
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Parameters:
  RecoveryInstance:
    Description: The EC2 instance ID to associate this alarm with.
    Type: AWS::EC2::Instance::Id
Resources:
  RecoveryTestAlarm:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmDescription: Trigger a recovery when instance status check fails for 15
        consecutive minutes.
      Namespace: AWS/EC2
      MetricName: StatusCheckFailed_System
      Statistic: Minimum
      Period: '60'
      EvaluationPeriods: '15'
      ComparisonOperator: GreaterThanThreshold
      Threshold: '0'
      AlarmActions: !Sub >
        "arn:aws:automate:${AWS::Region}:ec2:recover"
      Dimensions:
      - Name: InstanceId
        Value:
          Ref: RecoveryInstance
```

Amazon CloudWatch Logs Template Snippets

Topics

Send Logs to CloudWatch Logs from an Instance

Amazon CloudWatch Logs can monitor your system, application, and custom log files from Amazon EC2 instances or other sources. You can use AWS CloudFormation to provision and manage log groups and metric filters. For more information about getting started with Amazon CloudWatch Logs, see Monitoring System, Application, and Custom Log Files in the *Amazon CloudWatch User Guide*.

The following template describes a web server and its custom metrics. Log events from the web server's log provides the data for the custom metrics. To send log events to a custom metric, the `UserData` field installs a CloudWatch Logs agent on the Amazon EC2 instance. The configuration information for the agent, such as the location of the server log file, the log group name, and the log stream name, are defined in the `/tmp/cwlogs/apacheaccess.conf` file. The log stream is created after the web server starts sending log events to the `/var/log/httpd/access_log` file.

The two metric filters describe how the log information is transformed into CloudWatch metrics. The 404 metric counts the number of 404 occurrences. The size metric tracks the size of a request. The two CloudWatch alarms will send notifications if there are more than two 404s within two minutes or if the average request size is over 3500 KB over 10 minutes.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Description": "AWS CloudFormation Sample Template for CloudWatch Logs.",
    "Parameters": {
        "KeyName": {
          "Description": "Name of an existing EC2 KeyPair to enable SSH access to the
 instances",
          "Type": "AWS::EC2::KeyPair::KeyName",
          "ConstraintDescription" : "must be the name of an existing EC2 KeyPair."
        },
        "SSHLocation" : {
          "Description" : "The IP address range that can be used to SSH to the EC2
 instances",
          "Type": "String",
          "MinLength": "9",
          "MaxLength": "18",
          "Default": "0.0.0.0/0",
          "AllowedPattern": "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\\d{1,2})",
          "ConstraintDescription": "must be a valid IP CIDR range of the form x.x.x.x/x."
       },
       "OperatorEmail": {
          "Description": "Email address to notify if there are any scaling operations",
          "Type": "String"
       }
    },
    "Mappings": {
        "RegionMap": {
            "us-east-1": {
                "AMI": "ami-fb8e9292"
            },
            "us-west-1": {
                "AMI": "ami-7aba833f"
            },
            "us-west-2": {
                "AMI": "ami-043a5034"
            },
            "eu-west-1": {
```

```
                "AMI": "ami-2918e35e"
            },
            "ap-southeast-1": {
                "AMI": "ami-b40d5ee6"
            },
            "ap-southeast-2": {
                "AMI": "ami-3b4bd301"
            },
            "ap-northeast-1": {
                "AMI": "ami-c9562fc8"
            },
            "sa-east-1": {
                "AMI": "ami-215dff3c"
            },
            "eu-central-1": {
                "AMI" : "ami-a03503bd"
            }
        }
    },
    "Resources": {
        "LogRole": {
            "Type": "AWS::IAM::Role",
            "Properties": {
                "AssumeRolePolicyDocument": {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {
                                "Service": [
                                    "ec2.amazonaws.com"
                                ]
                            },
                            "Action": [
                                "sts:AssumeRole"
                            ]
                        }
                    ]
                },
                "Path": "/",
                "Policies": [
                    {
                        "PolicyName": "LogRolePolicy",
                        "PolicyDocument": {
                            "Version": "2012-10-17",
                            "Statement": [
                                {
                                    "Effect": "Allow",
                                    "Action": [
                                      "logs:Create*",
                                      "logs:PutLogEvents",
                                      "s3:GetObject"
                                    ],
                                    "Resource": [
                                        "arn:aws:logs:*:*:*",
                                        "arn:aws:s3:::*"
                                    ]
                                }
                            ]
                        }
                    }
                ]
            }
        },
        "LogRoleInstanceProfile": {
            "Type": "AWS::IAM::InstanceProfile",
```

```
            "Properties": {
                "Path": "/",
                "Roles": [
                    {
                        "Ref": "LogRole"
                    }
                ]
            }
        },
        "WebServerSecurityGroup": {
          "Type": "AWS::EC2::SecurityGroup",
          "Properties": {
            "GroupDescription": "Enable HTTP access via port 80 and SSH access via port
22",
            "SecurityGroupIngress" : [
              {"IpProtocol" : "tcp", "FromPort" : "80", "ToPort" : "80", "CidrIp" :
"0.0.0.0/0"},
              {"IpProtocol" : "tcp", "FromPort" : "22", "ToPort" : "22", "CidrIp" :
{ "Ref" : "SSHLocation"}}
            ]
          }
        },
        "WebServerHost": {
            "Type": "AWS::EC2::Instance",
            "Metadata": {
                "Comment": "Install a simple PHP application",
                "AWS::CloudFormation::Init": {
                    "config": {
                        "packages": {
                            "yum": {
                                "httpd": [],
                                "php": []
                            }
                        },
                        "files": {
                            "/tmp/cwlogs/apacheaccess.conf": {
                                "content": {
                                    "Fn::Join": [
                                        "",
                                        [
                                            "[general]\n",
                                            "state_file= /var/awslogs/agent-state\n",
                                            "[/var/log/httpd/access_log]\n",
                                            "file = /var/log/httpd/access_log\n",
                                            "log_group_name = ", {"Ref":
"WebServerLogGroup"}, "\n",
                                            "log_stream_name = {instance_id}/apache.log\n",
                                            "datetime_format = %d/%b/%Y:%H:%M:%S"
                                        ]
                                    ]
                                },
                                "mode": "000400",
                                "owner": "apache",
                                "group": "apache"
                            },
                            "/var/www/html/index.php": {
                                "content": {
                                    "Fn::Join": [
                                        "",
                                        [
                                            "<?php\n",
                                            "echo '<h1>AWS CloudFormation sample PHP
application</h1>';\n",
                                            "?>\n"
                                        ]
                                    ]
```

```
                    },
                    "mode": "000644",
                    "owner": "apache",
                    "group": "apache"
                },
                "/etc/cfn/cfn-hup.conf": {
                    "content": {
                        "Fn::Join": [
                            "",
                            [
                                "[main]\n",
                                "stack=",
                                {
                                    "Ref": "AWS::StackId"
                                },
                                "\n",
                                "region=",
                                {
                                    "Ref": "AWS::Region"
                                },
                                "\n"
                            ]
                        ]
                    },
                    "mode": "000400",
                    "owner": "root",
                    "group": "root"
                },
                "/etc/cfn/hooks.d/cfn-auto-reloader.conf": {
                    "content": {
                        "Fn::Join": [
                            "",
                            [
                                "[cfn-auto-reloader-hook]\n",
                                "triggers=post.update\n",

"path=Resources.WebServerHost.Metadata.AWS::CloudFormation::Init\n",
                                "action=/opt/aws/bin/cfn-init -s ",
                                {
                                    "Ref": "AWS::StackId"
                                },
                                " -r WebServerHost ",
                                " --region     ",
                                {
                                    "Ref": "AWS::Region"
                                },
                                "\n",
                                "runas=root\n"
                            ]
                        ]
                    }
                }
            },
            "services": {
                "sysvinit": {
                    "httpd": {
                        "enabled": "true",
                        "ensureRunning": "true"
                    },
                    "sendmail": {
                        "enabled": "false",
                        "ensureRunning": "false"
                    }
                }
            }
        }
```

```
                }
            },
            "CreationPolicy" : {
                "ResourceSignal" : { "Timeout" : "PT5M" }
            },
            "Properties": {
                "ImageId": {
                    "Fn::FindInMap": [
                        "RegionMap",
                        {
                            "Ref": "AWS::Region"
                        },
                        "AMI"
                    ]
                },
                "KeyName": {
                    "Ref": "KeyName"
                },
                "InstanceType": "t1.micro",
                "SecurityGroups": [ { "Ref": "WebServerSecurityGroup" } ],
                "IamInstanceProfile": { "Ref": "LogRoleInstanceProfile" },
                "UserData": {
                    "Fn::Base64": {
                        "Fn::Join": [
                            "",
                            [
                                "#!/bin/bash -xe\n",

                                "# Get the latest CloudFormation package\n",
                                "yum install -y aws-cfn-bootstrap\n",

                                "# Start cfn-init\n",
                                "/opt/aws/bin/cfn-init -s ", { "Ref": "AWS::StackId" }, " -
r WebServerHost ", " --region ", { "Ref": "AWS::Region" },
                                " || error_exit 'Failed to run cfn-init'\n",

                                "# Start up the cfn-hup daemon to listen for changes to the
 EC2 instance metadata\n",
                                "/opt/aws/bin/cfn-hup || error_exit 'Failed to start cfn-
hup'\n",

                                "# Get the CloudWatch Logs agent\n",
                                "wget https://s3.amazonaws.com/aws-cloudwatch/downloads/
latest/awslogs-agent-setup.py\n",

                                "# Install the CloudWatch Logs agent\n",
                                "python awslogs-agent-setup.py -n -r ", { "Ref" :
 "AWS::Region" }, " -c /tmp/cwlogs/apacheaccess.conf || error_exit 'Failed to run
 CloudWatch Logs agent setup'\n",

                                "# All done so signal success\n",
                                "/opt/aws/bin/cfn-signal -e $? ",
                                "         --stack ", { "Ref" : "AWS::StackName" },
                                "         --resource WebServerHost ",
                                "         --region ", { "Ref" : "AWS::Region" }, "\n"
                            ]
                        ]
                    }
                }
            }
        },
        "WebServerLogGroup": {
            "Type": "AWS::Logs::LogGroup",
            "Properties": {
                "RetentionInDays": 7
            }
```

```
        },
        "404MetricFilter": {
            "Type": "AWS::Logs::MetricFilter",
            "Properties": {
                "LogGroupName": {
                    "Ref": "WebServerLogGroup"
                },
                "FilterPattern": "[ip, identity, user_id, timestamp, request, status_code =
404, size, ...]",
                "MetricTransformations": [
                    {
                        "MetricValue": "1",
                        "MetricNamespace": "test/404s",
                        "MetricName": "test404Count"
                    }
                ]
            }
        },
        "BytesTransferredMetricFilter": {
            "Type": "AWS::Logs::MetricFilter",
            "Properties": {
                "LogGroupName": {
                    "Ref": "WebServerLogGroup"
                },
                "FilterPattern": "[ip, identity, user_id, timestamp, request, status_code,
size, ...]",
                "MetricTransformations": [
                    {
                        "MetricValue": "$size",
                        "MetricNamespace": "test/BytesTransferred",
                        "MetricName": "testBytesTransferred"
                    }
                ]
            }
        },
        "404Alarm": {
            "Type": "AWS::CloudWatch::Alarm",
            "Properties": {
                "AlarmDescription": "The number of 404s is greater than 2 over 2 minutes",
                "MetricName": "test404Count",
                "Namespace": "test/404s",
                "Statistic": "Sum",
                "Period": "60",
                "EvaluationPeriods": "2",
                "Threshold": "2",
                "AlarmActions": [
                    {
                        "Ref": "AlarmNotificationTopic"
                    }
                ],
                "ComparisonOperator": "GreaterThanThreshold"
            }
        },
        "BandwidthAlarm": {
            "Type": "AWS::CloudWatch::Alarm",
            "Properties": {
                "AlarmDescription": "The average volume of traffic is greater 3500 KB over
10 minutes",
                "MetricName": "testBytesTransferred",
                "Namespace": "test/BytesTransferred",
                "Statistic": "Average",
                "Period": "300",
                "EvaluationPeriods": "2",
                "Threshold": "3500",
                "AlarmActions": [
                    {
```

```
                        "Ref": "AlarmNotificationTopic"
                    }
                ],
                "ComparisonOperator": "GreaterThanThreshold"
            }
        },
        "AlarmNotificationTopic": {
          "Type": "AWS::SNS::Topic",
          "Properties": {
            "Subscription": [
                {
                    "Endpoint": { "Ref": "OperatorEmail" },
                    "Protocol": "email"
                }
             ]
          }
        }
    },
    "Outputs": {
        "InstanceId": {
            "Description": "The instance ID of the web server",
            "Value": {
                "Ref": "WebServerHost"
            }
        },
        "WebsiteURL" : {
          "Value" : { "Fn::Join" : ["", ["http://", { "Fn::GetAtt" : [ "WebServerHost",
 "PublicDnsName" ]}]] },
          "Description" : "URL for newly created LAMP stack"
        },
        "PublicIP": {
            "Description": "Public IP address of the web server",
            "Value": {
                "Fn::GetAtt": [
                    "WebServerHost",
                    "PublicIp"
                ]
            }
        },
        "CloudWatchLogGroupName": {
            "Description": "The name of the CloudWatch log group",
            "Value": {
                "Ref": "WebServerLogGroup"
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: AWS CloudFormation Sample Template for CloudWatch Logs.
Parameters:
  KeyName:
    Description: Name of an existing EC2 KeyPair to enable SSH access to the instances
    Type: AWS::EC2::KeyPair::KeyName
    ConstraintDescription: must be the name of an existing EC2 KeyPair.
  SSHLocation:
    Description: The IP address range that can be used to SSH to the EC2 instances
    Type: String
    MinLength: '9'
    MaxLength: '18'
    Default: 0.0.0.0/0
    AllowedPattern: "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\\d{1,2})"
```

```
    ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.
  OperatorEmail:
    Description: Email address to notify if there are any scaling operations
    Type: String
Mappings:
  RegionMap:
    us-east-1:
      AMI: ami-fb8e9292
    us-west-1:
      AMI: ami-7aba833f
    us-west-2:
      AMI: ami-043a5034
    eu-west-1:
      AMI: ami-2918e35e
    ap-southeast-1:
      AMI: ami-b40d5ee6
    ap-southeast-2:
      AMI: ami-3b4bd301
    ap-northeast-1:
      AMI: ami-c9562fc8
    sa-east-1:
      AMI: ami-215dff3c
    eu-central-1:
      AMI: ami-a03503bd
Resources:
  LogRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
        - Effect: Allow
          Principal:
            Service:
            - ec2.amazonaws.com
          Action:
          - sts:AssumeRole
      Path: "/"
      Policies:
      - PolicyName: LogRolePolicy
        PolicyDocument:
          Version: '2012-10-17'
          Statement:
          - Effect: Allow
            Action:
            - logs:Create*
            - logs:PutLogEvents
            - s3:GetObject
            Resource:
            - arn:aws:logs:*:*:*
            - arn:aws:s3:::*
  LogRoleInstanceProfile:
    Type: AWS::IAM::InstanceProfile
    Properties:
      Path: "/"
      Roles:
      - Ref: LogRole
  WebServerSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Enable HTTP access via port 80 and SSH access via port 22
      SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: '80'
        ToPort: '80'
        CidrIp: 0.0.0.0/0
```

```
    - IpProtocol: tcp
      FromPort: '22'
      ToPort: '22'
      CidrIp:
        Ref: SSHLocation
WebServerHost:
  Type: AWS::EC2::Instance
  Metadata:
    Comment: Install a simple PHP application
    AWS::CloudFormation::Init:
      config:
        packages:
          yum:
            httpd: []
            php: []
        files:
          "/tmp/cwlogs/apacheaccess.conf":
            content: !Sub |
              [general]
              state_file= /var/awslogs/agent-state
              [/var/log/httpd/access_log]
              file = /var/log/httpd/access_log
              log_group_name = ${WebServerLogGroup}
              log_stream_name = {instance_id}/apache.log
              datetime_format = %d/%b/%Y:%H:%M:%S
            mode: '000400'
            owner: apache
            group: apache
          "/var/www/html/index.php":
            content: !Sub |
              "<?php"
              "echo '<h1>AWS CloudFormation sample PHP application</h1>';"
              "?>"
            mode: '000644'
            owner: apache
            group: apache
          "/etc/cfn/cfn-hup.conf":
            content: !Sub |
              [main]
              stack= ${AWS::StackId}
              region=${AWS::Region}
            mode: "000400"
            owner: "root"
            group: "root"
          "/etc/cfn/hooks.d/cfn-auto-reloader.conf":
            content: !Sub |
              [cfn-auto-reloader-hook]
              triggers=post.update
              path=Resources.WebServerHost.Metadata.AWS::CloudFormation::Init
              action=/opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --resource
WebServerHost --region ${AWS::Region}
            mode: "000400"
            owner: "root"
            group: "root"
        services:
          sysvinit:
            httpd:
              enabled: 'true'
              ensureRunning: 'true'
            sendmail:
              enabled: 'false'
              ensureRunning: 'false'
  CreationPolicy:
    ResourceSignal:
      Timeout: PT5M
  Properties:
```

```
      ImageId:
        Fn::FindInMap:
        - RegionMap
        - Ref: AWS::Region
        - AMI
      KeyName:
        Ref: KeyName
      InstanceType: t1.micro
      SecurityGroups:
      - Ref: WebServerSecurityGroup
      IamInstanceProfile:
        Ref: LogRoleInstanceProfile
      UserData:
        "Fn::Base64":
          !Sub |
            #!/bin/bash -xe
            # Get the latest CloudFormation package
            yum update -y aws-cfn-bootstrap
            # Start cfn-init
            /opt/aws/bin/cfn-init -s ${AWS::StackId} -r WebServerHost --region
 ${AWS::Region} || error_exit 'Failed to run cfn-init'
            # Start up the cfn-hup daemon to listen for changes to the EC2 instance
 metadata
            /opt/aws/bin/cfn-hup || error_exit 'Failed to start cfn-hup'
            # Get the CloudWatch Logs agent
            wget https://s3.amazonaws.com/aws-cloudwatch/downloads/latest/awslogs-agent-
setup.py
            # Install the CloudWatch Logs agent
            python awslogs-agent-setup.py -n -r ${AWS::Region} -c /tmp/cwlogs/
apacheaccess.conf || error_exit 'Failed to run CloudWatch Logs agent setup'
            # All done so signal success
            /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackId} --resource WebServerHost
 --region ${AWS::Region}
  WebServerLogGroup:
    Type: AWS::Logs::LogGroup
    Properties:
      RetentionInDays: 7
  404MetricFilter:
    Type: AWS::Logs::MetricFilter
    Properties:
      LogGroupName:
        Ref: WebServerLogGroup
      FilterPattern: "[ip, identity, user_id, timestamp, request, status_code = 404,
 size, ...]"
      MetricTransformations:
      - MetricValue: '1'
        MetricNamespace: test/404s
        MetricName: test404Count
  BytesTransferredMetricFilter:
    Type: AWS::Logs::MetricFilter
    Properties:
      LogGroupName:
        Ref: WebServerLogGroup
      FilterPattern: "[ip, identity, user_id, timestamp, request, status_code, size, ...]"
      MetricTransformations:
      - MetricValue: "$size"
        MetricNamespace: test/BytesTransferred
        MetricName: testBytesTransferred
  404Alarm:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmDescription: The number of 404s is greater than 2 over 2 minutes
      MetricName: test404Count
      Namespace: test/404s
      Statistic: Sum
      Period: '60'
```

```
      EvaluationPeriods: '2'
      Threshold: '2'
      AlarmActions:
      - Ref: AlarmNotificationTopic
      ComparisonOperator: GreaterThanThreshold
  BandwidthAlarm:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmDescription: The average volume of traffic is greater 3500 KB over 10 minutes
      MetricName: testBytesTransferred
      Namespace: test/BytesTransferred
      Statistic: Average
      Period: '300'
      EvaluationPeriods: '2'
      Threshold: '3500'
      AlarmActions:
      - Ref: AlarmNotificationTopic
      ComparisonOperator: GreaterThanThreshold
  AlarmNotificationTopic:
    Type: AWS::SNS::Topic
    Properties:
      Subscription:
      - Endpoint:
          Ref: OperatorEmail
        Protocol: email
Outputs:
  InstanceId:
    Description: The instance ID of the web server
    Value:
      Ref: WebServerHost
  WebsiteURL:
    Value:
      !Sub 'http://${WebServerHost.PublicDnsName}'
    Description: URL for newly created LAMP stack
  PublicIP:
    Description: Public IP address of the web server
    Value:
      !GetAtt WebServerHost.PublicIp
  CloudWatchLogGroupName:
    Description: The name of the CloudWatch log group
    Value: !Ref WebServerLogGroup
```

See Also

For more information about CloudWatch Logs resources, see AWS::Logs::LogGroup (p. 835) or AWS::Logs::MetricFilter (p. 839).

Amazon EC2 Template Snippets

Topics

EC2 Block Device Mapping Examples

EC2 Instance with Block Device Mapping

JSON

```
    "Ec2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "ImageId" : { "Fn::FindInMap" : [ "AWSRegionArch2AMI", { "Ref" : "AWS::Region" },
                                          { "Fn::FindInMap" : [ "AWSInstanceType2Arch",
 { "Ref" : "InstanceType" }, "Arch" ] } ] },
        "KeyName" : { "Ref" : "KeyName" },
        "InstanceType" : { "Ref" : "InstanceType" },
        "SecurityGroups" : [{ "Ref" : "Ec2SecurityGroup" }],
        "BlockDeviceMappings" : [
          {
            "DeviceName" : "/dev/sda1",
            "Ebs" : { "VolumeSize" : "50" }
          },{
            "DeviceName" : "/dev/sdm",
            "Ebs" : { "VolumeSize" : "100" }
          }
        ]
      }
    }
```

YAML

```
EC2Instance:
    Type: AWS::EC2::Instance
    Properties:
        ImageId: !FindInMap [ AWSRegionArch2AMI, !Ref 'AWS::Region' , !FindInMap
 [ AWSInstanceType2Arch, !Ref InstanceType, Arch ] ]
        KeyName: !Ref KeyName
        InstanceType: !Ref InstanceType
        SecurityGroups:
        - !Ref Ec2SecurityGroup
        BlockDeviceMappings:
        -
          DeviceName: /dev/sda1
          Ebs:
            VolumeSize: 50
        -
          DeviceName: /dev/sdm
          Ebs:
            VolumeSize: 100
```

EC2 Instance with Ephemeral Drives

JSON

```
    "Ec2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "ImageId" : { "Fn::FindInMap" : [ "AWSRegionArch2AMI", { "Ref" : "AWS::Region" },
 "PV64" ]},
        "KeyName" : { "Ref" : "KeyName" },
        "InstanceType" : "m1.small",
        "SecurityGroups" : [{ "Ref" : "Ec2SecurityGroup" }],
        "BlockDeviceMappings" : [
          {
            "DeviceName"  : "/dev/sdc",
            "VirtualName" : "ephemeral0"
          }
        ]
      }
    }
```

YAML

```
EC2Instance:
 Type: AWS::EC2::Instance
 Properties:
   ImageId: !FindInMap [ AWSRegionArch2AMI, !Ref 'AWS::Region', PV64 ]
   KeyName: !Ref KeyName
   InstanceType: m1.small
   SecurityGroups:
   - !Ref Ec2SecurityGroup
   BlockDeviceMappings:
     -
       DeviceName: /dev/sdc
       VirtualName: ephemeral0
```

Assigning an Amazon EC2 Elastic IP Using AWS::EC2::EIP Snippet

This example shows how to allocate an Amazon EC2 Elastic IP address and assign it to an Amazon EC2 instance using a AWS::EC2::EIP resource (p. 564).

JSON

```
"MyEIP" : {
 "Type" : "AWS::EC2::EIP",
 "Properties" : {
     "InstanceId" : { "Ref" : "logical name of an AWS::EC2::Instance resource" }
 }
}
```

YAML

```
MyEIP:
  Type: AWS::EC2::EIP
  Properties:
    InstanceId: !Ref Logical name of an AWS::EC2::Instance resource
```

Assigning an Existing Elastic IP to an Amazon EC2 instance using AWS::EC2::EIPAssociation Snippet

This example shows how to assign an existing Amazon EC2 Elastic IP address to an Amazon EC2 instance using an AWS::EC2::EIPAssociation resource (p. 566).

JSON

```
"IPAssoc" : {
         "Type" : "AWS::EC2::EIPAssociation",
         "Properties" : {
             "InstanceId" : { "Ref" : "logical name of an AWS::EC2::Instance resource" },
             "EIP" : "existing Elastic IP address"
         }
     }
```

YAML

```
IPAssoc:
  Type: AWS::EC2::EIPAssociation
  Properties:
    InstanceId: !Ref Logical name of an AWS::EC2::Instance resource
    EIP: existing Elastic IP Address
```

Assigning an Existing VPC Elastic IP to an Amazon EC2 instance using AWS::EC2::EIPAssociation Snippet

This example shows how to assign an existing VPC Elastic IP address to an Amazon EC2 instance using an AWS::EC2::EIPAssociation resource (p. 566).

JSON

```
"VpcIPAssoc" : {
         "Type" : "AWS::EC2::EIPAssociation",
         "Properties" : {
             "InstanceId" : { "Ref" : "logical name of an AWS::EC2::Instance resource" },
             "AllocationId" : "existing VPC Elastic IP allocation ID"
         }
     }
```

YAML

```
VpcIPAssoc:
  Type: AWS::EC2::EIPAssociation
  Properties:
    InstanceId: !Ref Logical name of an AWS::EC2::Instance resource
    AllocationId: Existing VPC Elastic IP allocation ID
```

Elastic Network Interface (ENI) Template Snippets

VPC_EC2_Instance_With_ENI

Sample template showing how to create an instance with two elastic network interface (ENI). The sample assumes you have already created a VPC.

JSON

```
 "Resources" : {
   "ControlPortAddress" : {
     "Type" : "AWS::EC2::EIP",
     "Properties" : {
       "Domain" : "vpc"
     }
   },
   "AssociateControlPort" : {
     "Type" : "AWS::EC2::EIPAssociation",
     "Properties" : {
       "AllocationId" : { "Fn::GetAtt" : [ "ControlPortAddress", "AllocationId" ]},
       "NetworkInterfaceId" : { "Ref" : "controlXface" }
     }
   },
   "WebPortAddress" : {
     "Type" : "AWS::EC2::EIP",
     "Properties" : {
       "Domain" : "vpc"
     }
   },
   "AssociateWebPort" : {
     "Type" : "AWS::EC2::EIPAssociation",
     "Properties" : {
       "AllocationId" : { "Fn::GetAtt" : [ "WebPortAddress", "AllocationId" ]},
       "NetworkInterfaceId" : { "Ref" : "webXface" }
     }
   },
   "SSHSecurityGroup" : {
     "Type" : "AWS::EC2::SecurityGroup",
     "Properties" : {
       "VpcId" : { "Ref" : "VpcId" },
       "GroupDescription" : "Enable SSH access via port 22",
       "SecurityGroupIngress" : [ { "IpProtocol" : "tcp", "FromPort" : "22", "ToPort" :
"22", "CidrIp" : "0.0.0.0/0" } ]
     }
   },
   "WebSecurityGroup" : {
     "Type" : "AWS::EC2::SecurityGroup",
     "Properties" : {
       "VpcId" : { "Ref" : "VpcId" },
       "GroupDescription" : "Enable HTTP access via user defined port",
       "SecurityGroupIngress" : [ { "IpProtocol" : "tcp", "FromPort" : 80, "ToPort" : 80,
"CidrIp" : "0.0.0.0/0" } ]
     }
   },
   "controlXface" : {
     "Type" : "AWS::EC2::NetworkInterface",
     "Properties" : {
       "SubnetId" : { "Ref" : "SubnetId" },
       "Description" :"Interface for control traffic such as SSH",
       "GroupSet" : [ {"Ref" : "SSHSecurityGroup"} ],
       "SourceDestCheck" : "true",
       "Tags" : [ {"Key" : "Network", "Value" : "Control"}]
     }
   },
  "webXface" : {
     "Type" : "AWS::EC2::NetworkInterface",
     "Properties" : {
       "SubnetId" : { "Ref" : "SubnetId" },
       "Description" :"Interface for web traffic",
       "GroupSet" : [ {"Ref" : "WebSecurityGroup"} ],
       "SourceDestCheck" : "true",
       "Tags" : [ {"Key" : "Network", "Value" : "Web"}]
```

```
      }
    },
    "Ec2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "ImageId" : { "Fn::FindInMap" : [ "RegionMap", { "Ref" : "AWS::Region" }, "AMI" ]},
        "KeyName" : { "Ref" : "KeyName" },
        "NetworkInterfaces" : [ { "NetworkInterfaceId" : {"Ref" : "controlXface"},
 "DeviceIndex" : "0" },
        { "NetworkInterfaceId" : {"Ref" : "webXface"}, "DeviceIndex" : "1" }],
        "Tags" : [ {"Key" : "Role", "Value" : "Test Instance"}],
        "UserData" : {"Fn::Base64" : { "Fn::Join" : ["",[
   "#!/bin/bash -ex","\n",
            "\n","yum install ec2-net-utils -y","\n",
   "ec2ifup eth1","\n",
   "service httpd start"]]}
  }
   }
    }
  }
```

YAML

```
Resources:
  ControlPortAddress:
    Type: AWS::EC2::EIP
    Properties:
      Domain: vpc
  AssociateControlPort:
    Type: AWS::EC2::EIPAssociation
    Properties:
      AllocationId: !GetAtt ControlPortAddress.AllocationId
      NetworkInterfaceId: !Ref controlXface
  WebPortAddress:
    Type: AWS::EC2::EIP
    Properties:
      Domain: vpc
  AssociateWebPort:
    Type: AWS::EC2::EIPAssociation
    Properties:
      AllocationId: !GetAtt WebPortAddress.AllocationId
      NetworkInterfaceId: !Ref webXface
  SSHSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      VpcId: !Ref VpcId
      GroupDescription: Enable SSH access via port 22
      SecurityGroupIngress:
      - CidrIp: 0.0.0.0/0
        FromPort: 22
        IpProtocol: tcp
        ToPort: 22
  WebSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      VpcId: !Ref VpcId
      GroupDescription: Enable HTTP access via user defined port
      SecurityGroupIngress:
      - CidrIp: 0.0.0.0/0
        FromPort: 80
        IpProtocol: tcp
        ToPort: 80
  controlXface:
    Type: AWS::EC2::NetworkInterface
```

```
    Properties:
      SubnetId: !Ref SubnetId
      Description: Interface for controlling traffic such as SSH
      GroupSet:
      - !Ref SSHSecurityGroup
      SourceDestCheck: true
      Tags:
        -
          Key: Network
          Value: Control
  webXface:
    Type: AWS::EC2::NetworkInterface
    Properties:
      SubnetId: !Ref SubnetId
      Description: Interface for controlling traffic such as SSH
      GroupSet:
      - !Ref WebSecurityGroup
      SourceDestCheck: true
      Tags:
        -
          Key: Network
          Value: Web
  Ec2Instance:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !FindInMap [ RegionMap, !Ref 'AWS::Region', AMI ]
      KeyName: !Ref KeyName
      NetworkInterfaces:
        -
          NetworkInterfaceId: !Ref controlXface
          DeviceIndex: 0
        -
          NetworkInterfaceId: !Ref webXface
          DeviceIndex: 1
      Tags:
        -
          Key: Role
          Value: Test Instance
      UserData:
        Fn::Base64: !Sub |
          #!/bin/bash -xe
          yum install ec2-net-utils -y
          ec2ifup eth1
          service httpd start
```

Amazon EC2 Instance Resource

This snippet shows a simple AWS::EC2::Instance resource.

JSON

```
"MyInstance" : {
 "Type" : "AWS::EC2::Instance",
 "Properties" : {
     "AvailabilityZone" : "us-east-1a",
     "ImageId" : "ami-20b65349"
 }
}
```

YAML

```
MyInstance:
```

```
  Type: AWS::EC2::Instance
  Properties:
    AvailabilityZone: us-east-1a
    ImageId: ami-20b65349
```

Amazon EC2 Instance with Volume, Tag, and UserData Properties

This snippet shows an AWS::EC2::Instance resource with one Amazon EC2 volume, one tag, and a user data property. An AWS::EC2::SecurityGroup resource, an AWS::SNS::Topic resource, and an AWS::EC2::Volume resource all must be defined in the same template. Also, the reference to `KeyName` is a parameters that must be defined in the Parameters section of the template.

JSON

```
"MyInstance" : {
 "Type" : "AWS::EC2::Instance",
 "Properties" : {
     "KeyName" : { "Ref" : "KeyName" },
     "SecurityGroups" : [ {
         "Ref" : "logical name of AWS::EC2::SecurityGroup resource"
     } ],
     "UserData" : {
         "Fn::Base64" : {
             "Fn::Join" : [ ":", [
                 "PORT=80",
                 "TOPIC=", {
                     "Ref" : "logical name of an AWS::SNS::Topic resource"
                 } ]
             ]
         }
      },
     "InstanceType" : "m1.small",
     "AvailabilityZone" : "us-east-1a",
     "ImageId" : "ami-1e817677",
     "Volumes" : [
        { "VolumeId" : {
             "Ref" : "logical name of AWS::EC2::Volume resource"
        },
        "Device" : "/dev/sdk" }
     ],

     "Tags" : [ {
         "Key" : "Name",
         "Value" : "MyTag"
     } ]
 }
}
```

YAML

```
MyInstance:
  Type: AWS::EC2::Instance
  Properties:
    KeyName: !Ref KeyName
    SecurityGroups:
    - !Ref logical name of AWS::EC2::SecurityGroup resource
    UserData:
      Fn::Base64: !Sub |
        PORT=80
        TOPIC=${ logical name of an AWS::SNS::Topic resource }
```

```
    InstanceType: m1.small
    AvailabilityZone: us-east-1a
    ImageId: ami-1e817677
    Volumes:
      -
        VolumeId: !Ref logical name of AWS::EC2::Volume resource
        Device: /dev/sdk
    Tags:
      -
        Key: Name
        Value: MyTag
```

Amazon EC2 Instance Resource with an Amazon SimpleDB Domain

This snippet shows an AWS::EC2::Instance resource with an Amazon SimpleDB domain specified in the UserData.

JSON

```
"MyInstance" : {
 "Type" : "AWS::EC2::Instance",
 "Properties" : {
     "UserData" : {
         "Fn::Base64" : {
             "Fn::Join" : [ "",
                 [ "Domain=", {
                     "Ref" : "logical name of an AWS::SDB::Domain resource"
                 } ]
             ]
         }
      },
     "AvailabilityZone" : "us-east-1a",
     "ImageId" : "ami-20b65349"
 }
}
```

YAML

```
MyInstance:
  Type: AWS::EC2::Instance
  Properties:
    UserData:
      Fn::Base64: !Sub |
        Domain=${ logical name of an AWS::SDB::Domain resource }
    AvailabilityZone: us-east-1a
    ImageId: ami-20b65349
```

Amazon EC2 Security Group Resource with Two CIDR Range Ingress Rules

This snippet shows an AWS::EC2::SecurityGroup resource that describes two ingress rules giving access to a specified CIDR range for the TCP protocol on the specified ports.

JSON

```
"ServerSecurityGroup" : {
 "Type" : "AWS::EC2::SecurityGroup",
```

```
 "Properties" : {
     "GroupDescription" : "allow connections from specified CIDR ranges",
     "SecurityGroupIngress" : [
         {
             "IpProtocol" : "tcp",
             "FromPort" : "80",
             "ToPort" : "80",
             "CidrIp" : "0.0.0.0/0"
         },{
             "IpProtocol" : "tcp",
             "FromPort" : "22",
             "ToPort" : "22",
             "CidrIp" : "192.168.1.1/32"
         }
     ]
 }
}
```

YAML

```
ServerSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: allow connections from specified CIDR ranges
    SecurityGroupIngress:
    - IpProtocol: tcp
      FromPort: 80
      ToPort: 80
      CidrIp: 0.0.0.0/0
    - IpProtocol: tcp
      FromPort: 22
      ToPort: 22
      CidrIp: 192.168.1.1/32
```

Amazon EC2 Security Group Resource with Two Security Group Ingress Rules

This snippet shows an AWS::EC2::SecurityGroup resource that describes two security group ingress rules. The first ingress rule grants access to the existing security group myadminsecuritygroup, which is owned by the 1234-5678-9012 AWS account, for the TCP protocol on port 22. The second ingress rule grants access to the security group mysecuritygroupcreatedincfn for TCP on port 80. This ingress rule uses the Ref intrinsic function to refer to a security group (whose logical name is mysecuritygroupcreatedincfn) created in the same template. You must declare a value for both the `SourceSecurityGroupName` and `SourceSecurityGroupOwnerId` properties.

JSON

```
"ServerSecurityGroupBySG" : {
 "Type" : "AWS::EC2::SecurityGroup",
 "Properties" : {
     "GroupDescription" : "allow connections from specified source security group",
     "SecurityGroupIngress" : [
         {
            "IpProtocol" : "tcp",
            "FromPort" : "22",
            "ToPort" : "22",
            "SourceSecurityGroupName" : "myadminsecuritygroup",
            "SourceSecurityGroupOwnerId" : "123456789012"
         },
         {
```

```
            "IpProtocol" : "tcp",
            "FromPort" : "80",
            "ToPort" : "80",
            "SourceSecurityGroupName" : {"Ref" : "mysecuritygroupcreatedincfn"}
         }
     ]
 }
}
```

YAML

```
ServerSecurityGroupBySG:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: allow connections from specified source security group
    SecurityGroupIngress:
    - IpProtocol: tcp
      FromPort: 80
      ToPort: 80
      SourceSecurityGroupName: myadminsecuritygroup
      SourceSecurityGroupOwnerId: 123456789012
    - IpProtocol: tcp
      FromPort: 80
      ToPort: 80
      SourceSecurityGroupName: !Ref mysecuritygroupcreatedincfn
```

Amazon EC2 Security Group Resource with LoadBalancer Ingress Rule

This snippet shows an AWS::EC2::SecurityGroup resource that contains a security group ingress rule that grants access to the LoadBalancer myELB for TCP on port 80. Note that the rule uses the `SourceSecurityGroup.OwnerAlias` and `SourceSecurityGroup.GroupName` properties of the myELB resource to specify the source security group of the LoadBalancer.

JSON

```
        "myELB" : {
                 "Type" : "AWS::ElasticLoadBalancing::LoadBalancer",
                 "Properties" : {
                     "AvailabilityZones" : [ "us-east-1a" ],
                     "Listeners" : [ {
                         "LoadBalancerPort" : "80",
                         "InstancePort" : "80",
                         "Protocol" : "HTTP"
                     } ]
                 }
             },
     "ELBIngressGroup" : {
         "Type" : "AWS::EC2::SecurityGroup",
         "Properties" : {
             "GroupDescription" : "ELB ingress group",
             "SecurityGroupIngress" : [
                 {
                    "IpProtocol" : "tcp",
                    "FromPort" : "80",
                    "ToPort" : "80",
                    "SourceSecurityGroupOwnerId" : {"Fn::GetAtt" : ["myELB",
 "SourceSecurityGroup.OwnerAlias"]},
                    "SourceSecurityGroupName" : {"Fn::GetAtt" : ["myELB",
 "SourceSecurityGroup.GroupName"]}
                 }
```

```
        ]
    }
```

YAML

```
myELB:
  Type: AWS::ElasticLoadBalancing::LoadBalancer
  Properties:
    AvailabilityZones:
    - us-east-1a
    Listeners:
      -
        LoadBalancerPort: 80
        InstancePort: 80
        Protocol: HTTP
ELBIngressGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: ELB ingress group
    SecurityGroupIngress:
    - IpProtocol: tcp
      FromPort: 80
      ToPort: 80
      SourceSecurityGroupOwnerId: !GetAtt myELB.SourceSecurityGroup.OwnerAlias
      SourceSecurityGroupName: !GetAtt myELB.SourceSecurityGroup.GroupName
```

Using AWS::EC2::SecurityGroupIngress to Create Mutually Referencing Amazon EC2 Security Group Resources

This snippet shows two AWS::EC2::SecurityGroupIngress resources that add mutual ingress rules to the EC2 security groups SGroup1 and SGroup2. The SGroup1Ingress resource enables ingress from SGroup2 through TCP/IP port 80 to SGroup1. The SGroup2Ingress resource enables ingress from SGroup1 through TCP/IP port 80 to SGroup2.

> **Note**
> If you are using an Amazon VPC, use the `AWS::EC2::SecurityGroup` resource and specify the `VpcId` property.

JSON

```
        "SGroup1" : {
         "Type" : "AWS::EC2::SecurityGroup",
         "Properties" : {
             "GroupDescription" : "EC2 Instance access"
         }
     },
     "SGroup2" : {
         "Type" : "AWS::EC2::SecurityGroup",
         "Properties" : {
             "GroupDescription" : "EC2 Instance access"
         }
     },
     "SGroup1Ingress" : {
         "Type" : "AWS::EC2::SecurityGroupIngress",
         "Properties" : {
             "GroupName" : { "Ref" : "SGroup1" },
             "IpProtocol" : "tcp",
             "ToPort" : "80",
             "FromPort" : "80",
             "SourceSecurityGroupName" : { "Ref" : "SGroup2" }
         }
```

```
    },
    "SGroup2Ingress" : {
        "Type" : "AWS::EC2::SecurityGroupIngress",
        "Properties" : {
            "GroupName" : { "Ref" : "SGroup2" },
            "IpProtocol" : "tcp",
            "ToPort" : "80",
            "FromPort" : "80",
            "SourceSecurityGroupName" : { "Ref" : "SGroup1" }
        }
    }
```

YAML

```
SGroup1:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: EC2 Instance access
SGroup2:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: EC2 Instance access
SGroup1Ingress:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupName: !Ref SGroup1
    IpProtocol: tcp
    ToPort: 80
    FromPort: 80
    SourceSecurityGroupName: !Ref SGroup2
SGroup2Ingress:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupName: !Ref SGroup2
    IpProtocol: tcp
    ToPort: 80
    FromPort: 80
    SourceSecurityGroupName: !Ref SGroup1
```

Amazon EC2 Volume Resource

This snippet shows a simple Amazon EC2 volume resource with a DeletionPolicy attribute set to Snapshot. With the Snapshot DeletionPolicy set, AWS CloudFormation will take a snapshot of this volume before deleting it during stack deletion. Make sure you specify a value for `SnapShotId`, or a value for `Size`, but not both. Remove the one you don't need.

JSON

```
"MyEBSVolume" : {
 "Type" : "AWS::EC2::Volume",
 "Properties" : {
     "Size" : "specify a size if no SnapShotId",
     "SnapshotId" : "specify a SnapShotId if no Size",
     "AvailabilityZone" : { "Ref" : "AvailabilityZone" }
 },
 "DeletionPolicy" : "Snapshot"
}
```

YAML

```
MyEBSVolume:
```

```
  Type: AWS::EC2::Volume
  Properties:
    Size: specify a size if no SnapshotId
    SnapshotId: specify a SnapShotId if no Size
    AvailabilityZone: !Ref AvailabilityZone
  DeletionPolicy: Snapshot
```

Amazon EC2 VolumeAttachment Resource

This snippet shows the following resources: an Amazon EC2 instance using an Amazon Linux AMI from the US-East (Northern Virginia) Region, an EC2 security group that allows SSH access to IP addresses, a new Amazon EBS volume sized at 100 GB and in the same Availability Zone as the EC2 instance, and a volume attachment that attaches the new volume to the EC2 instance.

JSON

```
"Resources" : {
 "Ec2Instance" : {
   "Type" : "AWS::EC2::Instance",
   "Properties" : {
     "SecurityGroups" : [ { "Ref" : "InstanceSecurityGroup" } ],
     "ImageId" : "ami-76f0061f"
   }
 },

 "InstanceSecurityGroup" : {
   "Type" : "AWS::EC2::SecurityGroup",
   "Properties" : {
     "GroupDescription" : "Enable SSH access via port 22",
     "SecurityGroupIngress" : [ {
       "IpProtocol" : "tcp",
       "FromPort" : "22",
       "ToPort" : "22",
       "CidrIp" : "0.0.0.0/0"
     } ]
   }
 },

 "NewVolume" : {
   "Type" : "AWS::EC2::Volume",
   "Properties" : {
     "Size" : "100",
     "AvailabilityZone" : { "Fn::GetAtt" : [ "Ec2Instance", "AvailabilityZone" ]},
   }
 },

 "MountPoint" : {
   "Type" : "AWS::EC2::VolumeAttachment",
   "Properties" : {
     "InstanceId" : { "Ref" : "Ec2Instance" },
     "VolumeId"  : { "Ref" : "NewVolume" },
     "Device" : "/dev/sdh"
   }
 }
}
```

YAML

```
Resources:
  Ec2Instance:
    Type: AWS::EC2::Instance
```

```
    Properties:
      SecurityGroups:
      - !Ref InstanceSecurityGroup
      ImageId: ami-76f0061f
  InstanceSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Enable SSH access via port 22
      SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: 22
        ToPort: 22
        CidrIp: 0.0.0.0/0
  NewVolume:
    Type: AWS::EC2::Volume
    Properties:
      Size: 100
      AvailabilityZone: !GetAtt Ec2Instance.AvailabilityZone
  MountPoint:
    Type: AWS::EC2::VolumeAttachment
    Properties:
      InstanceId: !Ref Ec2Instance
      VolumeId: !Ref NewVolume
      Device: /dev/sdh
```

Amazon EC2 Instance in a Default VPC Security Group

Whenever you create a VPC, AWS automatically creates default resources for that VPC, such as a security group. However, when you define a VPC in AWS CloudFormation templates, you don't yet have the physical IDs of those default resources. To obtain the IDs, use the `Fn::GetAtt` (p. 1324) intrinsic function. That way, you can use the default resources instead of creating new ones in your template. For example, the following template snippet associates the default security group of the `myVPC` VPC with the `myInstance` Amazon EC2 instance.

JSON

```
"myVPC": {
  "Type": "AWS::EC2::VPC",
  "Properties": {
    "CidrBlock": {"Ref": "myVPCCIDRRange"},
    "EnableDnsSupport": false,
    "EnableDnsHostnames": false,
    "InstanceTenancy": "default"
  }
},
"myInstance" : {
  "Type" : "AWS::EC2::Instance",
  "Properties" : {
    "ImageId": {
      "Fn::FindInMap": ["AWSRegionToAMI",{"Ref": "AWS::Region"},"64"]
    },
    "SecurityGroupIds" : [{"Fn::GetAtt": ["myVPC", "DefaultSecurityGroup"]}],
    "SubnetId" : {"Ref" : "mySubnet"}
  }
}
```

YAML

```
myVPC:
  Type: AWS::EC2::VPC
  Properties:
```

```
    CidrBlock: !Ref myVPCCIDRRange
    EnableDnsSupport: false
    EnableDnsHostnames: false
    InstanceTenancy: default
myInstance:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !FindInMap [ AWSRegionToAMI , !Ref 'AWS::Region', 64 ]
    SecurityGroupIds:
    - !GetAtt myVPC.DefaultSecurityGroup
    SubnetId: !Ref mySubnet
```

Amazon EC2 Container Service Template Snippets

Amazon EC2 Container Service (Amazon ECS) is a container management service that makes it easy to run, stop, and manage Docker containers on a cluster of Amazon Elastic Compute Cloud (Amazon EC2) instances.

The following example template deploys a web application in an Amazon ECS container with autoscaling and an application load balancer. For more information, see Getting Started with Amazon ECS in the *Amazon EC2 Container Service Developer Guide*.

Important
For the latest AMI IDs, see Amazon ECS-optimized AMI in the *Amazon EC2 Container Service Developer Guide*.

JSON

```
{
  "AWSTemplateFormatVersion":"2010-09-09",
  "Parameters":{
    "KeyName":{
      "Type":"AWS::EC2::KeyPair::KeyName",
      "Description":"Name of an existing EC2 KeyPair to enable SSH access to the ECS
 instances."
    },
    "VpcId":{
      "Type":"AWS::EC2::VPC::Id",
      "Description":"Select a VPC that allows instances to access the Internet."
    },
    "SubnetID":{
      "Type":"List<AWS::EC2::Subnet::Id>",
      "Description":"Select at two subnets in your selected VPC."
    },
    "DesiredCapacity":{
      "Type":"Number",
      "Default":"1",
      "Description":"Number of instances to launch in your ECS cluster."
    },
    "MaxSize":{
      "Type":"Number",
      "Default":"1",
      "Description":"Maximum number of instances that can be launched in your ECS cluster."
    },
    "InstanceType":{
      "Description":"EC2 instance type",
      "Type":"String",
      "Default":"t2.micro",
      "AllowedValues":[
        "t2.micro",
        "t2.small",
```

```
        "t2.medium",
        "t2.large",
        "m3.medium",
        "m3.large",
        "m3.xlarge",
        "m3.2xlarge",
        "m4.large",
        "m4.xlarge",
        "m4.2xlarge",
        "m4.4xlarge",
        "m4.10xlarge",
        "c4.large",
        "c4.xlarge",
        "c4.2xlarge",
        "c4.4xlarge",
        "c4.8xlarge",
        "c3.large",
        "c3.xlarge",
        "c3.2xlarge",
        "c3.4xlarge",
        "c3.8xlarge",
        "r3.large",
        "r3.xlarge",
        "r3.2xlarge",
        "r3.4xlarge",
        "r3.8xlarge",
        "i2.xlarge",
        "i2.2xlarge",
        "i2.4xlarge",
        "i2.8xlarge"
      ],
      "ConstraintDescription":"Please choose a valid instance type."
    }
  },
  "Mappings":{
    "AWSRegionToAMI":{
      "us-east-1":{
        "AMIID":"ami-eca289fb"
      },
      "us-east-2":{
        "AMIID":"ami-446f3521"
      },
      "us-west-1":{
        "AMIID":"ami-9fadf8ff"
      },
      "us-west-2":{
        "AMIID":"ami-7abc111a"
      },
      "eu-west-1":{
        "AMIID":"ami-a1491ad2"
      },
      "eu-central-1":{
        "AMIID":"ami-54f5303b"
      },
      "ap-northeast-1":{
        "AMIID":"ami-9cd57ffd"
      },
      "ap-southeast-1":{
        "AMIID":"ami-a900a3ca"
      },
      "ap-southeast-2":{
        "AMIID":"ami-5781be34"
      }
    }
  },
  "Resources":{
```

```
"ECSCluster":{
  "Type":"AWS::ECS::Cluster"
},
"EcsSecurityGroup":{
  "Type":"AWS::EC2::SecurityGroup",
  "Properties":{
    "GroupDescription":"ECS Security Group",
    "VpcId":{
      "Ref":"VpcId"
    }
  }
},
"EcsSecurityGroupHTTPinbound":{
  "Type":"AWS::EC2::SecurityGroupIngress",
  "Properties":{
    "GroupId":{
      "Ref":"EcsSecurityGroup"
    },
    "IpProtocol":"tcp",
    "FromPort":"80",
    "ToPort":"80",
    "CidrIp":"0.0.0.0/0"
  }
},
"EcsSecurityGroupSSHinbound":{
  "Type":"AWS::EC2::SecurityGroupIngress",
  "Properties":{
    "GroupId":{
      "Ref":"EcsSecurityGroup"
    },
    "IpProtocol":"tcp",
    "FromPort":"22",
    "ToPort":"22",
    "CidrIp":"0.0.0.0/0"
  }
},
"EcsSecurityGroupALBports":{
  "Type":"AWS::EC2::SecurityGroupIngress",
  "Properties":{
    "GroupId":{
      "Ref":"EcsSecurityGroup"
    },
    "IpProtocol":"tcp",
    "FromPort":"31000",
    "ToPort":"61000",
    "SourceSecurityGroupId":{
      "Ref":"EcsSecurityGroup"
    }
  }
},
"CloudwatchLogsGroup":{
  "Type":"AWS::Logs::LogGroup",
  "Properties":{
    "LogGroupName":{
      "Fn::Join":[
        "-",
        [
          "ECSLogGroup",
          {
            "Ref":"AWS::StackName"
          }
        ]
      ]
    },
    "RetentionInDays":14
  }
```

```
    },
    "taskdefinition":{
      "Type":"AWS::ECS::TaskDefinition",
      "Properties":{
        "Family":{
          "Fn::Join":[
            "",
            [
              {
                "Ref":"AWS::StackName"
              },
              "-ecs-demo-app"
            ]
          ]
        },
        "ContainerDefinitions":[
          {
            "Name":"simple-app",
            "Cpu":"10",
            "Essential":"true",
            "Image":"httpd:2.4",
            "Memory":"300",
            "LogConfiguration":{
              "LogDriver":"awslogs",
              "Options":{
                "awslogs-group":{
                  "Ref":"CloudwatchLogsGroup"
                },
                "awslogs-region":{
                  "Ref":"AWS::Region"
                },
                "awslogs-stream-prefix":"ecs-demo-app"
              }
            },
            "MountPoints":[
              {
                "ContainerPath":"/usr/local/apache2/htdocs",
                "SourceVolume":"my-vol"
              }
            ],
            "PortMappings":[
              {
                "ContainerPort":80
              }
            ]
          },
          {
            "Name":"busybox",
            "Cpu":10,
            "Command":[
              "/bin/sh -c \"while true; do echo '<html> <head> <title>Amazon ECS
Sample App</title> <style>body {margin-top: 40px; background-color: #333;} </style>
</head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
<h2>Congratulations!</h2> <p>Your application is now running on a container in Amazon
ECS.</p>' > top; /bin/date > date ; echo '</div></body></html>' > bottom; cat top date
bottom > /usr/local/apache2/htdocs/index.html ; sleep 1; done\""
            ],
            "EntryPoint":[
              "sh",
              "-c"
            ],
            "Essential":false,
            "Image":"busybox",
            "Memory":200,
            "LogConfiguration":{
              "LogDriver":"awslogs",
```

```
            "Options":{
              "awslogs-group":{
                "Ref":"CloudwatchLogsGroup"
              },
              "awslogs-region":{
                "Ref":"AWS::Region"
              },
              "awslogs-stream-prefix":"ecs-demo-app"
            }
          },
          "VolumesFrom":[
            {
              "SourceContainer":"simple-app"
            }
          ]
        }
      ],
      "Volumes":[
        {
          "Name":"my-vol"
        }
      ]
    }
  },
  "ECSALB":{
    "Type":"AWS::ElasticLoadBalancingV2::LoadBalancer",
    "Properties":{
      "Name":"ECSALB",
      "Scheme":"internet-facing",
      "LoadBalancerAttributes":[
        {
          "Key":"idle_timeout.timeout_seconds",
          "Value":"30"
        }
      ],
      "Subnets":{
        "Ref":"SubnetID"
      },
      "SecurityGroups":[
        {
          "Ref":"EcsSecurityGroup"
        }
      ]
    }
  },
  "ALBListener":{
    "Type":"AWS::ElasticLoadBalancingV2::Listener",
    "DependsOn":"ECSServiceRole",
    "Properties":{
      "DefaultActions":[
        {
          "Type":"forward",
          "TargetGroupArn":{
            "Ref":"ECSTG"
          }
        }
      ],
      "LoadBalancerArn":{
        "Ref":"ECSALB"
      },
      "Port":"80",
      "Protocol":"HTTP"
    }
  },
  "ECSALBListenerRule":{
    "Type":"AWS::ElasticLoadBalancingV2::ListenerRule",
```

```
      "DependsOn":"ALBListener",
      "Properties":{
        "Actions":[
          {
            "Type":"forward",
            "TargetGroupArn":{
              "Ref":"ECSTG"
            }
          }
        ],
        "Conditions":[
          {
            "Field":"path-pattern",
            "Values":[
              "/"
            ]
          }
        ],
        "ListenerArn":{
          "Ref":"ALBListener"
        },
        "Priority":1
      }
    },
    "ECSTG":{
      "Type":"AWS::ElasticLoadBalancingV2::TargetGroup",
      "DependsOn":"ECSALB",
      "Properties":{
        "HealthCheckIntervalSeconds":10,
        "HealthCheckPath":"/",
        "HealthCheckProtocol":"HTTP",
        "HealthCheckTimeoutSeconds":5,
        "HealthyThresholdCount":2,
        "Name":"ECSTG",
        "Port":80,
        "Protocol":"HTTP",
        "UnhealthyThresholdCount":2,
        "VpcId":{
          "Ref":"VpcId"
        }
      }
    },
    "ECSAutoScalingGroup":{
      "Type":"AWS::AutoScaling::AutoScalingGroup",
      "Properties":{
        "VPCZoneIdentifier":{
          "Ref":"SubnetID"
        },
        "LaunchConfigurationName":{
          "Ref":"ContainerInstances"
        },
        "MinSize":"1",
        "MaxSize":{
          "Ref":"MaxSize"
        },
        "DesiredCapacity":{
          "Ref":"DesiredCapacity"
        }
      },
      "CreationPolicy":{
        "ResourceSignal":{
          "Timeout":"PT15M"
        }
      },
      "UpdatePolicy":{
        "AutoScalingReplacingUpdate":{
```

```
            "WillReplace":"true"
          }
        }
      },
      "ContainerInstances":{
        "Type":"AWS::AutoScaling::LaunchConfiguration",
        "Properties":{
          "ImageId":{
            "Fn::FindInMap":[
              "AWSRegionToAMI",
              {
                "Ref":"AWS::Region"
              },
              "AMIID"
            ]
          },
          "SecurityGroups":[
            {
              "Ref":"EcsSecurityGroup"
            }
          ],
          "InstanceType":{
            "Ref":"InstanceType"
          },
          "IamInstanceProfile":{
            "Ref":"EC2InstanceProfile"
          },
          "KeyName":{
            "Ref":"KeyName"
          },
          "UserData":{
            "Fn::Base64":{
              "Fn::Join":[
                "",
                [
                  "#!/bin/bash -xe\n",
                  "echo ECS_CLUSTER=",
                  {
                    "Ref":"ECSCluster"
                  },
                  " >> /etc/ecs/ecs.config\n",
                  "yum install -y aws-cfn-bootstrap\n",
                  "/opt/aws/bin/cfn-signal -e $? ",
                  "         --stack ",
                  {
                    "Ref":"AWS::StackName"
                  },
                  "         --resource ECSAutoScalingGroup ",
                  "         --region ",
                  {
                    "Ref":"AWS::Region"
                  },
                  "\n"
                ]
              ]
            }
          }
        }
      },
      "service":{
        "Type":"AWS::ECS::Service",
        "DependsOn":"ALBListener",
        "Properties":{
          "Cluster":{
            "Ref":"ECSCluster"
          },
```

```
        "DesiredCount":"1",
        "LoadBalancers":[
          {
            "ContainerName":"simple-app",
            "ContainerPort":"80",
            "TargetGroupArn":{
              "Ref":"ECSTG"
            }
          }
        ],
        "Role":{
          "Ref":"ECSServiceRole"
        },
        "TaskDefinition":{
          "Ref":"taskdefinition"
        }
      }
    },
    "ECSServiceRole":{
      "Type":"AWS::IAM::Role",
      "Properties":{
        "AssumeRolePolicyDocument":{
          "Statement":[
            {
              "Effect":"Allow",
              "Principal":{
                "Service":[
                  "ecs.amazonaws.com"
                ]
              },
              "Action":[
                "sts:AssumeRole"
              ]
            }
          ]
        },
        "Path":"/",
        "Policies":[
          {
            "PolicyName":"ecs-service",
            "PolicyDocument":{
              "Statement":[
                {
                  "Effect":"Allow",
                  "Action":[
                    "elasticloadbalancing:DeregisterInstancesFromLoadBalancer",
                    "elasticloadbalancing:DeregisterTargets",
                    "elasticloadbalancing:Describe*",
                    "elasticloadbalancing:RegisterInstancesWithLoadBalancer",
                    "elasticloadbalancing:RegisterTargets",
                    "ec2:Describe*",
                    "ec2:AuthorizeSecurityGroupIngress"
                  ],
                  "Resource":"*"
                }
              ]
            }
          }
        ]
      }
    },
    "ServiceScalingTarget":{
      "Type":"AWS::ApplicationAutoScaling::ScalableTarget",
      "DependsOn":"service",
      "Properties":{
        "MaxCapacity":2,
```

```
        "MinCapacity":1,
        "ResourceId":{
          "Fn::Join":[
            "",
            [
              "service/",
              {
                "Ref":"ECSCluster"
              },
              "/",
              {
                "Fn::GetAtt":[
                  "service",
                  "Name"
                ]
              }
            ]
          ]
        },
        "RoleARN":{
          "Fn::GetAtt":[
            "AutoscalingRole",
            "Arn"
          ]
        },
        "ScalableDimension":"ecs:service:DesiredCount",
        "ServiceNamespace":"ecs"
      }
    },
    "ServiceScalingPolicy":{
      "Type":"AWS::ApplicationAutoScaling::ScalingPolicy",
      "Properties":{
        "PolicyName":"AStepPolicy",
        "PolicyType":"StepScaling",
        "ScalingTargetId":{
          "Ref":"ServiceScalingTarget"
        },
        "StepScalingPolicyConfiguration":{
          "AdjustmentType":"PercentChangeInCapacity",
          "Cooldown":60,
          "MetricAggregationType":"Average",
          "StepAdjustments":[
            {
              "MetricIntervalLowerBound":0,
              "ScalingAdjustment":200
            }
          ]
        }
      }
    },
    "ALB500sAlarmScaleUp":{
      "Type":"AWS::CloudWatch::Alarm",
      "Properties":{
        "EvaluationPeriods":"1",
        "Statistic":"Average",
        "Threshold":"10",
        "AlarmDescription":"Alarm if our ALB generates too many HTTP 500s.",
        "Period":"60",
        "AlarmActions":[
          {
            "Ref":"ServiceScalingPolicy"
          }
        ],
        "Namespace":"AWS/ApplicationELB",
        "Dimensions":[
          {
```

```
            "Name":"ECSService",
            "Value":{
              "Ref":"service"
            }
          }
        ],
        "ComparisonOperator":"GreaterThanThreshold",
        "MetricName":"HTTPCode_ELB_5XX_Count"
      }
    },
    "EC2Role":{
      "Type":"AWS::IAM::Role",
      "Properties":{
        "AssumeRolePolicyDocument":{
          "Statement":[
            {
              "Effect":"Allow",
              "Principal":{
                "Service":[
                  "ec2.amazonaws.com"
                ]
              },
              "Action":[
                "sts:AssumeRole"
              ]
            }
          ]
        },
        "Path":"/",
        "Policies":[
          {
            "PolicyName":"ecs-service",
            "PolicyDocument":{
              "Statement":[
                {
                  "Effect":"Allow",
                  "Action":[
                    "ecs:CreateCluster",
                    "ecs:DeregisterContainerInstance",
                    "ecs:DiscoverPollEndpoint",
                    "ecs:Poll",
                    "ecs:RegisterContainerInstance",
                    "ecs:StartTelemetrySession",
                    "ecs:Submit*",
                    "logs:CreateLogStream",
                    "logs:PutLogEvents"
                  ],
                  "Resource":"*"
                }
              ]
            }
          }
        ]
      }
    },
    "AutoscalingRole":{
      "Type":"AWS::IAM::Role",
      "Properties":{
        "AssumeRolePolicyDocument":{
          "Statement":[
            {
              "Effect":"Allow",
              "Principal":{
                "Service":[
                  "application-autoscaling.amazonaws.com"
                ]
```

```
              },
              "Action":[
                "sts:AssumeRole"
              ]
            }
          ]
        },
        "Path":"/",
        "Policies":[
          {
            "PolicyName":"service-autoscaling",
            "PolicyDocument":{
              "Statement":[
                {
                  "Effect":"Allow",
                  "Action":[
                    "application-autoscaling:*",
                    "cloudwatch:DescribeAlarms",
                    "cloudwatch:PutMetricAlarm",
                    "ecs:DescribeServices",
                    "ecs:UpdateService"
                  ],
                  "Resource":"*"
                }
              ]
            }
          }
        ]
      }
    },
    "EC2InstanceProfile":{
      "Type":"AWS::IAM::InstanceProfile",
      "Properties":{
        "Path":"/",
        "Roles":[
          {
            "Ref":"EC2Role"
          }
        ]
      }
    }
  },
  "Outputs":{
    "ecsservice":{
      "Value":{
        "Ref":"service"
      }
    },
    "ecscluster":{
      "Value":{
        "Ref":"ECSCluster"
      }
    },
    "ECSALB":{
      "Description":"Your ALB DNS URL",
      "Value":{
        "Fn::Join":[
          "",
          [
            {
              "Fn::GetAtt":[
                "ECSALB",
                "DNSName"
              ]
            }
          ]
```

```
        ]
      }
    },
    "taskdef":{
      "Value":{
        "Ref":"taskdefinition"
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Parameters:
  KeyName:
    Type: AWS::EC2::KeyPair::KeyName
    Description: Name of an existing EC2 KeyPair to enable SSH access to the ECS instances.
  VpcId:
    Type: AWS::EC2::VPC::Id
    Description: Select a VPC that allows instances access to the Internet.
  SubnetID:
    Type: List<AWS::EC2::Subnet::Id>
    Description: Select at two subnets in your selected VPC.
  DesiredCapacity:
    Type: Number
    Default: '1'
    Description: Number of instances to launch in your ECS cluster.
  MaxSize:
    Type: Number
    Default: '1'
    Description: Maximum number of instances that can be launched in your ECS cluster.
  InstanceType:
    Description: EC2 instance type
    Type: String
    Default: t2.micro
    AllowedValues: [t2.micro, t2.small, t2.medium, t2.large, m3.medium, m3.large,
      m3.xlarge, m3.2xlarge, m4.large, m4.xlarge, m4.2xlarge, m4.4xlarge, m4.10xlarge,
      c4.large, c4.xlarge, c4.2xlarge, c4.4xlarge, c4.8xlarge, c3.large, c3.xlarge,
      c3.2xlarge, c3.4xlarge, c3.8xlarge, r3.large, r3.xlarge, r3.2xlarge, r3.4xlarge,
      r3.8xlarge, i2.xlarge, i2.2xlarge, i2.4xlarge, i2.8xlarge]
    ConstraintDescription: Please choose a valid instance type.
Mappings:
  AWSRegionToAMI:
    us-east-1:
      AMIID: ami-eca289fb
    us-east-2:
      AMIID: ami-446f3521
    us-west-1:
      AMIID: ami-9fadf8ff
    us-west-2:
      AMIID: ami-7abc111a
    eu-west-1:
      AMIID: ami-a1491ad2
    eu-central-1:
      AMIID: ami-54f5303b
    ap-northeast-1:
      AMIID: ami-9cd57ffd
    ap-southeast-1:
      AMIID: ami-a900a3ca
    ap-southeast-2:
      AMIID: ami-5781be34
Resources:
  ECSCluster:
```

```
    Type: AWS::ECS::Cluster
  EcsSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: ECS Security Group
      VpcId: !Ref 'VpcId'
  EcsSecurityGroupHTTPinbound:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !Ref 'EcsSecurityGroup'
      IpProtocol: tcp
      FromPort: '80'
      ToPort: '80'
      CidrIp: 0.0.0.0/0
  EcsSecurityGroupSSHinbound:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !Ref 'EcsSecurityGroup'
      IpProtocol: tcp
      FromPort: '22'
      ToPort: '22'
      CidrIp: 0.0.0.0/0
  EcsSecurityGroupALBports:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !Ref 'EcsSecurityGroup'
      IpProtocol: tcp
      FromPort: '31000'
      ToPort: '61000'
      SourceSecurityGroupId: !Ref 'EcsSecurityGroup'
  CloudwatchLogsGroup:
    Type: AWS::Logs::LogGroup
    Properties:
      LogGroupName: !Join ['-', [ECSLogGroup, !Ref 'AWS::StackName']]
      RetentionInDays: 14
  taskdefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      Family: !Join ['', [!Ref 'AWS::StackName', -ecs-demo-app]]
      ContainerDefinitions:
      - Name: simple-app
        Cpu: '10'
        Essential: 'true'
        Image: httpd:2.4
        Memory: '300'
        LogConfiguration:
          LogDriver: awslogs
          Options:
            awslogs-group: !Ref 'CloudwatchLogsGroup'
            awslogs-region: !Ref 'AWS::Region'
            awslogs-stream-prefix: ecs-demo-app
        MountPoints:
        - ContainerPath: /usr/local/apache2/htdocs
          SourceVolume: my-vol
        PortMappings:
        - ContainerPort: 80
      - Name: busybox
        Cpu: 10
        Command: ['/bin/sh -c "while true; do echo ''<html> <head> <title>Amazon ECS
            Sample App</title> <style>body {margin-top: 40px; background-color: #333;}
            </style> </head><body> <div style=color:white;text-align:center> <h1>Amazon
            ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your application is now
            running on a container in Amazon ECS.</p>'' > top; /bin/date > date ;
            echo ''</div></body></html>'' > bottom; cat top date bottom > /usr/local/
apache2/htdocs/index.html
            ; sleep 1; done"']
```

```
        EntryPoint: [sh, -c]
        Essential: false
        Image: busybox
        Memory: 200
        LogConfiguration:
          LogDriver: awslogs
          Options:
            awslogs-group: !Ref 'CloudwatchLogsGroup'
            awslogs-region: !Ref 'AWS::Region'
            awslogs-stream-prefix: ecs-demo-app
        VolumesFrom:
        - SourceContainer: simple-app
      Volumes:
      - Name: my-vol
  ECSALB:
    Type: AWS::ElasticLoadBalancingV2::LoadBalancer
    Properties:
      Name: ECSALB
      Scheme: internet-facing
      LoadBalancerAttributes:
      - Key: idle_timeout.timeout_seconds
        Value: '30'
      Subnets: !Ref 'SubnetID'
      SecurityGroups: [!Ref 'EcsSecurityGroup']
  ALBListener:
    Type: AWS::ElasticLoadBalancingV2::Listener
    DependsOn: ECSServiceRole
    Properties:
      DefaultActions:
      - Type: forward
        TargetGroupArn: !Ref 'ECSTG'
      LoadBalancerArn: !Ref 'ECSALB'
      Port: '80'
      Protocol: HTTP
  ECSALBListenerRule:
    Type: AWS::ElasticLoadBalancingV2::ListenerRule
    DependsOn: ALBListener
    Properties:
      Actions:
      - Type: forward
        TargetGroupArn: !Ref 'ECSTG'
      Conditions:
      - Field: path-pattern
        Values: [/]
      ListenerArn: !Ref 'ALBListener'
      Priority: 1
  ECSTG:
    Type: AWS::ElasticLoadBalancingV2::TargetGroup
    DependsOn: ECSALB
    Properties:
      HealthCheckIntervalSeconds: 10
      HealthCheckPath: /
      HealthCheckProtocol: HTTP
      HealthCheckTimeoutSeconds: 5
      HealthyThresholdCount: 2
      Name: ECSTG
      Port: 80
      Protocol: HTTP
      UnhealthyThresholdCount: 2
      VpcId: !Ref 'VpcId'
  ECSAutoScalingGroup:
    Type: AWS::AutoScaling::AutoScalingGroup
    Properties:
      VPCZoneIdentifier: !Ref 'SubnetID'
      LaunchConfigurationName: !Ref 'ContainerInstances'
      MinSize: '1'
```

```
      MaxSize: !Ref 'MaxSize'
      DesiredCapacity: !Ref 'DesiredCapacity'
    CreationPolicy:
      ResourceSignal:
        Timeout: PT15M
    UpdatePolicy:
      AutoScalingReplacingUpdate:
        WillReplace: 'true'
  ContainerInstances:
    Type: AWS::AutoScaling::LaunchConfiguration
    Properties:
      ImageId: !FindInMap [AWSRegionToAMI, !Ref 'AWS::Region', AMIID]
      SecurityGroups: [!Ref 'EcsSecurityGroup']
      InstanceType: !Ref 'InstanceType'
      IamInstanceProfile: !Ref 'EC2InstanceProfile'
      KeyName: !Ref 'KeyName'
      UserData:
        Fn::Base64: !Sub |
          #!/bin/bash -xe
          echo ECS_CLUSTER=${ECSCluster} >> /etc/ecs/ecs.config
          yum install -y aws-cfn-bootstrap
          /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --resource
 ECSAutoScalingGroup --region ${AWS::Region}
  service:
    Type: AWS::ECS::Service
    DependsOn: ALBListener
    Properties:
      Cluster: !Ref 'ECSCluster'
      DesiredCount: '1'
      LoadBalancers:
      - ContainerName: simple-app
        ContainerPort: '80'
        TargetGroupArn: !Ref 'ECSTG'
      Role: !Ref 'ECSServiceRole'
      TaskDefinition: !Ref 'taskdefinition'
  ECSServiceRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
        - Effect: Allow
          Principal:
            Service: [ecs.amazonaws.com]
          Action: ['sts:AssumeRole']
      Path: /
      Policies:
      - PolicyName: ecs-service
        PolicyDocument:
          Statement:
          - Effect: Allow
            Action: ['elasticloadbalancing:DeregisterInstancesFromLoadBalancer',
 'elasticloadbalancing:DeregisterTargets',
              'elasticloadbalancing:Describe*',
 'elasticloadbalancing:RegisterInstancesWithLoadBalancer',
              'elasticloadbalancing:RegisterTargets', 'ec2:Describe*',
 'ec2:AuthorizeSecurityGroupIngress']
            Resource: '*'
  ServiceScalingTarget:
    Type: AWS::ApplicationAutoScaling::ScalableTarget
    DependsOn: service
    Properties:
      MaxCapacity: 2
      MinCapacity: 1
      ResourceId: !Join ['', [service/, !Ref 'ECSCluster', /, !GetAtt [service, Name]]]
      RoleARN: !GetAtt [AutoscalingRole, Arn]
      ScalableDimension: ecs:service:DesiredCount
```

```
      ServiceNamespace: ecs
  ServiceScalingPolicy:
    Type: AWS::ApplicationAutoScaling::ScalingPolicy
    Properties:
      PolicyName: AStepPolicy
      PolicyType: StepScaling
      ScalingTargetId: !Ref 'ServiceScalingTarget'
      StepScalingPolicyConfiguration:
        AdjustmentType: PercentChangeInCapacity
        Cooldown: 60
        MetricAggregationType: Average
        StepAdjustments:
        - MetricIntervalLowerBound: 0
          ScalingAdjustment: 200
  ALB500sAlarmScaleUp:
    Type: AWS::CloudWatch::Alarm
    Properties:
      EvaluationPeriods: '1'
      Statistic: Average
      Threshold: '10'
      AlarmDescription: Alarm if our ALB generates too many HTTP 500s.
      Period: '60'
      AlarmActions: [!Ref 'ServiceScalingPolicy']
      Namespace: AWS/ApplicationELB
      Dimensions:
      - Name: ECSService
        Value: !Ref 'service'
      ComparisonOperator: GreaterThanThreshold
      MetricName: HTTPCode_ELB_5XX_Count
  EC2Role:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
        - Effect: Allow
          Principal:
            Service: [ec2.amazonaws.com]
          Action: ['sts:AssumeRole']
      Path: /
      Policies:
      - PolicyName: ecs-service
        PolicyDocument:
          Statement:
          - Effect: Allow
            Action: ['ecs:CreateCluster', 'ecs:DeregisterContainerInstance',
'ecs:DiscoverPollEndpoint',
              'ecs:Poll', 'ecs:RegisterContainerInstance', 'ecs:StartTelemetrySession',
              'ecs:Submit*', 'logs:CreateLogStream', 'logs:PutLogEvents']
            Resource: '*'
  AutoscalingRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
        - Effect: Allow
          Principal:
            Service: [application-autoscaling.amazonaws.com]
          Action: ['sts:AssumeRole']
      Path: /
      Policies:
      - PolicyName: service-autoscaling
        PolicyDocument:
          Statement:
          - Effect: Allow
            Action: ['application-autoscaling:*', 'cloudwatch:DescribeAlarms',
'cloudwatch:PutMetricAlarm',
```

```
              'ecs:DescribeServices', 'ecs:UpdateService']
            Resource: '*'
  EC2InstanceProfile:
    Type: AWS::IAM::InstanceProfile
    Properties:
      Path: /
      Roles: [!Ref 'EC2Role']
Outputs:
  ecsservice:
    Value: !Ref 'service'
  ecscluster:
    Value: !Ref 'ECSCluster'
  ECSALB:
    Description: Your ALB DNS URL
    Value: !Join ['', [!GetAtt [ECSALB, DNSName]]]
  taskdef:
    Value: !Ref 'taskdefinition'
```

Amazon Elastic File System Sample Template

Amazon Elastic File System (Amazon EFS) is a file storage service for Amazon Elastic Compute Cloud (Amazon EC2) instances. With Amazon EFS, your applications have storage when they need it because storage capacity grows and shrinks automatically as you add and remove files.

The following sample template deploys EC2 instances (in an Auto Scaling group) that are associated with an Amazon EFS file system. To associate the instances with the file system, the instances run the cfn-init helper script, which downloads and installs the `nfs-utils` yum package, creates a new directory, and then uses the file system's DNS name to mount the file system at that directory. The file system's DNS name resolves to a mount target's IP address in the Amazon EC2 instance's Availability Zone. For more information about the DNS name structure, see Mounting File Systems in the *Amazon Elastic File System User Guide*.

To measure Network File System activity, the template includes custom Amazon CloudWatch metrics. The template also creates a VPC, subnet, and security groups. To allow the instances to communicate with the file system, the VPC must have DNS enabled, and the mount target and the EC2 instances must be in the same Availability Zone (AZ), which is specified by the subnet.

The security group of the mount target enables a network connection to TCP port 2049, which is required for an NFSv4 client to mount a file system. For more information on security groups for EC2 instances and mount targets, see Security in the *Amazon Elastic File System User Guide*.

Note

If you make an update to the mount target that causes it to be replaced, instances or applications that use the associated file system might be disrupted. This can cause uncommitted writes to be lost. To avoid disruption, stop your instances when you update the mount target by setting the desired capacity to zero. This allows the instances to unmount the file system before the mount target is deleted. After the mount update has completed, start your instances in a subsequent update by setting the desired capacity.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "This template creates an Amazon EFS file system and mount target and
 associates it with Amazon EC2 instances in an Auto Scaling group. **WARNING** This
 template creates Amazon EC2 instances and related resources. You will be billed for the
 AWS resources used if you create a stack from this template.",
  "Parameters": {
    "InstanceType" : {
```

```
      "Description" : "WebServer EC2 instance type",
      "Type" : "String",
      "Default" : "m1.small",
      "AllowedValues" : [ "t1.micro", "t2.micro", "t2.small", "t2.medium", "m1.small",
"m1.medium", "m1.large", "m1.xlarge", "m2.xlarge", "m2.2xlarge", "m2.4xlarge",
"m3.medium", "m3.large", "m3.xlarge", "m3.2xlarge", "c1.medium", "c1.xlarge", "c3.large",
"c3.xlarge", "c3.2xlarge", "c3.4xlarge", "c3.8xlarge", "c4.large", "c4.xlarge",
"c4.2xlarge", "c4.4xlarge", "c4.8xlarge", "g2.2xlarge", "r3.large", "r3.xlarge",
"r3.2xlarge", "r3.4xlarge", "r3.8xlarge", "i2.xlarge", "i2.2xlarge", "i2.4xlarge",
"i2.8xlarge", "d2.xlarge", "d2.2xlarge", "d2.4xlarge", "d2.8xlarge", "hi1.4xlarge",
"hs1.8xlarge", "cr1.8xlarge", "cc2.8xlarge", "cg1.4xlarge"],
      "ConstraintDescription" : "Must be a valid EC2 instance type."
    },
    "KeyName": {
      "Type": "AWS::EC2::KeyPair::KeyName",
      "Description": "Name of an existing EC2 key pair to enable SSH access to the ECS
instances"
    },
    "AsgMaxSize": {
      "Type": "Number",
      "Description": "Maximum size and initial desired capacity of Auto Scaling Group",
      "Default": "2"
    },
    "SSHLocation" : {
      "Description" : "The IP address range that can be used to connect to the EC2
instances by using SSH",
      "Type": "String",
      "MinLength": "9",
      "MaxLength": "18",
      "Default": "0.0.0.0/0",
      "AllowedPattern": "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\\d{1,2})",
      "ConstraintDescription": "must be a valid IP CIDR range of the form x.x.x.x/x."
    },
    "VolumeName" : {
      "Description" : "The name to be used for the EFS volume",
      "Type": "String",
      "MinLength": "1",
      "Default": "myEFSvolume"
    },
    "MountPoint" : {
      "Description" : "The Linux mount point for the EFS volume",
      "Type": "String",
      "MinLength": "1",
      "Default": "myEFSvolume"
    }
  },
  "Mappings" : {
    "AWSInstanceType2Arch" : {
      "t1.micro"    : { "Arch" : "PV64"   },
      "t2.micro"    : { "Arch" : "HVM64"  },
      "t2.small"    : { "Arch" : "HVM64"  },
      "t2.medium"   : { "Arch" : "HVM64"  },
      "m1.small"    : { "Arch" : "PV64"   },
      "m1.medium"   : { "Arch" : "PV64"   },
      "m1.large"    : { "Arch" : "PV64"   },
      "m1.xlarge"   : { "Arch" : "PV64"   },
      "m2.xlarge"   : { "Arch" : "PV64"   },
      "m2.2xlarge"  : { "Arch" : "PV64"   },
      "m2.4xlarge"  : { "Arch" : "PV64"   },
      "m3.medium"   : { "Arch" : "HVM64"  },
      "m3.large"    : { "Arch" : "HVM64"  },
      "m3.xlarge"   : { "Arch" : "HVM64"  },
      "m3.2xlarge"  : { "Arch" : "HVM64"  },
      "c1.medium"   : { "Arch" : "PV64"   },
      "c1.xlarge"   : { "Arch" : "PV64"   },
      "c3.large"    : { "Arch" : "HVM64"  },
```

```
      "c3.xlarge"   : { "Arch" : "HVM64"  },
      "c3.2xlarge"  : { "Arch" : "HVM64"  },
      "c3.4xlarge"  : { "Arch" : "HVM64"  },
      "c3.8xlarge"  : { "Arch" : "HVM64"  },
      "c4.large"    : { "Arch" : "HVM64"  },
      "c4.xlarge"   : { "Arch" : "HVM64"  },
      "c4.2xlarge"  : { "Arch" : "HVM64"  },
      "c4.4xlarge"  : { "Arch" : "HVM64"  },
      "c4.8xlarge"  : { "Arch" : "HVM64"  },
      "g2.2xlarge"  : { "Arch" : "HVMG2"  },
      "r3.large"    : { "Arch" : "HVM64"  },
      "r3.xlarge"   : { "Arch" : "HVM64"  },
      "r3.2xlarge"  : { "Arch" : "HVM64"  },
      "r3.4xlarge"  : { "Arch" : "HVM64"  },
      "r3.8xlarge"  : { "Arch" : "HVM64"  },
      "i2.xlarge"   : { "Arch" : "HVM64"  },
      "i2.2xlarge"  : { "Arch" : "HVM64"  },
      "i2.4xlarge"  : { "Arch" : "HVM64"  },
      "i2.8xlarge"  : { "Arch" : "HVM64"  },
      "d2.xlarge"   : { "Arch" : "HVM64"  },
      "d2.2xlarge"  : { "Arch" : "HVM64"  },
      "d2.4xlarge"  : { "Arch" : "HVM64"  },
      "d2.8xlarge"  : { "Arch" : "HVM64"  },
      "hi1.4xlarge" : { "Arch" : "HVM64"  },
      "hs1.8xlarge" : { "Arch" : "HVM64"  },
      "cr1.8xlarge" : { "Arch" : "HVM64"  },
      "cc2.8xlarge" : { "Arch" : "HVM64"  }
    },
    "AWSRegionArch2AMI" : {
      "us-east-1"        : {"PV64" : "ami-1ccae774", "HVM64" : "ami-1ecae776", "HVMG2" :
 "ami-8c6b40e4"},
      "us-west-2"        : {"PV64" : "ami-ff527ecf", "HVM64" : "ami-e7527ed7", "HVMG2" :
 "ami-abbe919b"},
      "us-west-1"        : {"PV64" : "ami-d514f291", "HVM64" : "ami-d114f295", "HVMG2" :
 "ami-f31ffeb7"},
      "eu-west-1"        : {"PV64" : "ami-bf0897c8", "HVM64" : "ami-a10897d6", "HVMG2" :
 "ami-d5bc24a2"},
      "eu-central-1"     : {"PV64" : "ami-ac221fb1", "HVM64" : "ami-a8221fb5", "HVMG2" :
 "ami-7cd2ef61"},
      "ap-northeast-1"   : {"PV64" : "ami-27f90e27", "HVM64" : "ami-cbf90ecb", "HVMG2" :
 "ami-6318e863"},
      "ap-southeast-1"   : {"PV64" : "ami-acd9e8fe", "HVM64" : "ami-68d8e93a", "HVMG2" :
 "ami-3807376a"},
      "ap-southeast-2"   : {"PV64" : "ami-ff9cecc5", "HVM64" : "ami-fd9cecc7", "HVMG2" :
 "ami-89790ab3"},
      "sa-east-1"        : {"PV64" : "ami-bb2890a6", "HVM64" : "ami-b52890a8", "HVMG2" :
 "NOT_SUPPORTED"},
      "cn-north-1"       : {"PV64" : "ami-fa39abc3", "HVM64" : "ami-f239abcb", "HVMG2" :
 "NOT_SUPPORTED"}
    }
  },
  "Resources": {
    "CloudWatchPutMetricsRole" : {
      "Type"  : "AWS::IAM::Role",
      "Properties" : {
          "AssumeRolePolicyDocument" : {
              "Statement" : [ {
                  "Effect" : "Allow",
                  "Principal" : {
                      "Service" : [ "ec2.amazonaws.com" ]
                  },
                  "Action" : [ "sts:AssumeRole" ]
              } ]
          },
          "Path" : "/"
      }
```

```
    },
    "CloudWatchPutMetricsRolePolicy" : {
        "Type" : "AWS::IAM::Policy",
        "Properties" : {
            "PolicyName" : "CloudWatch_PutMetricData",
            "PolicyDocument" : {
              "Version": "2012-10-17",
              "Statement": [
                {
                  "Sid": "CloudWatchPutMetricData",
                  "Effect": "Allow",
                  "Action": ["cloudwatch:PutMetricData"],
                  "Resource": ["*"]
                }
              ]
            },
            "Roles" : [ { "Ref" : "CloudWatchPutMetricsRole" } ]
        }
    },
    "CloudWatchPutMetricsInstanceProfile" : {
      "Type" : "AWS::IAM::InstanceProfile",
      "Properties" : {
        "Path" : "/",
        "Roles" : [ { "Ref" : "CloudWatchPutMetricsRole" } ]
      }
    },
    "VPC": {
      "Type": "AWS::EC2::VPC",
      "Properties": {
        "EnableDnsSupport" : "true",
        "EnableDnsHostnames" : "true",
        "CidrBlock": "10.0.0.0/16",
        "Tags": [ {"Key": "Application", "Value": { "Ref": "AWS::StackId"} } ]
      }
    },
    "InternetGateway" : {
      "Type" : "AWS::EC2::InternetGateway",
      "Properties" : {
        "Tags" : [
          { "Key" : "Application", "Value" : { "Ref" : "AWS::StackName" } },
          { "Key" : "Network", "Value" : "Public" }
        ]
      }
    },
    "GatewayToInternet" : {
      "Type" : "AWS::EC2::VPCGatewayAttachment",
      "Properties" : {
        "VpcId" : { "Ref" : "VPC" },
        "InternetGatewayId" : { "Ref" : "InternetGateway" }
      }
    },
    "RouteTable":{
      "Type":"AWS::EC2::RouteTable",
      "Properties":{
        "VpcId": {"Ref":"VPC"}
      }
    },
    "SubnetRouteTableAssoc": {
      "Type" : "AWS::EC2::SubnetRouteTableAssociation",
      "Properties" : {
        "RouteTableId" : {"Ref":"RouteTable"},
        "SubnetId" : {"Ref":"Subnet"}
      }
    },
    "InternetGatewayRoute": {
        "Type":"AWS::EC2::Route",
```

```
        "Properties":{
            "DestinationCidrBlock":"0.0.0.0/0",
            "RouteTableId":{"Ref":"RouteTable"},
            "GatewayId":{"Ref":"InternetGateway"}
        }
    },
    "Subnet": {
      "Type": "AWS::EC2::Subnet",
      "Properties": {
        "VpcId": { "Ref": "VPC" },
        "CidrBlock": "10.0.0.0/24",
        "Tags": [ { "Key": "Application", "Value": { "Ref": "AWS::StackId" } } ]
      }
    },
    "InstanceSecurityGroup": {
      "Type": "AWS::EC2::SecurityGroup",
      "Properties": {
        "VpcId": { "Ref": "VPC" },
        "GroupDescription": "Enable SSH access via port 22",
        "SecurityGroupIngress": [
          { "IpProtocol": "tcp", "FromPort": "22", "ToPort": "22", "CidrIp": { "Ref":
"SSHLocation" } },
          { "IpProtocol": "tcp", "FromPort": "80", "ToPort": "80", "CidrIp": "0.0.0.0/0" }
         ]
      }
    },
    "MountTargetSecurityGroup": {
      "Type": "AWS::EC2::SecurityGroup",
      "Properties": {
        "VpcId": { "Ref": "VPC" },
        "GroupDescription": "Security group for mount target",
        "SecurityGroupIngress": [
          {
            "IpProtocol": "tcp",
            "FromPort": "2049",
            "ToPort": "2049",
            "CidrIp": "0.0.0.0/0"
          }
        ]
      }
    },
    "FileSystem": {
      "Type": "AWS::EFS::FileSystem",
      "Properties": {
        "PerformanceMode": "generalPurpose",
        "FileSystemTags": [
          {
            "Key": "Name",
            "Value": { "Ref" : "VolumeName" }
          }
        ]
      }
    },
    "MountTarget": {
      "Type": "AWS::EFS::MountTarget",
      "Properties": {
        "FileSystemId": { "Ref": "FileSystem" },
        "SubnetId": { "Ref": "Subnet" },
        "SecurityGroups": [ { "Ref": "MountTargetSecurityGroup" } ]
      }
    },
    "LaunchConfiguration": {
      "Type": "AWS::AutoScaling::LaunchConfiguration",
      "Metadata" : {
        "AWS::CloudFormation::Init" : {
          "configSets" : {
```

```
          "MountConfig" : [ "setup", "mount" ]
        },
        "setup" : {
          "packages" : {
            "yum" : {
              "nfs-utils" : []
            }
          },
          "files" : {
            "/home/ec2-user/post_nfsstat" : {
              "content" : { "Fn::Join" : [ "", [
                    "#!/bin/bash\n",
                    "\n",
                    "INPUT=\"$(cat)\"\n",
                    "CW_JSON_OPEN='{ \"Namespace\": \"EFS\", \"MetricData\": [ '\n",
                    "CW_JSON_CLOSE=' ] }'\n",
                    "CW_JSON_METRIC=''\n",
                    "METRIC_COUNTER=0\n",
                    "\n",
                    "for COL in 1 2 3 4 5 6; do\n",
                    "\n",
                    " COUNTER=0\n",
                    " METRIC_FIELD=$COL\n",
                    " DATA_FIELD=$(($COL+($COL-1)))\n",
                    "\n",
                    " while read line; do\n",
                    "   if [[ COUNTER -gt 0 ]]; then\n",
                    "\n",
                    "     LINE=`echo $line | tr -s ' ' `\n",
                    "     AWS_COMMAND=\"aws cloudwatch put-metric-data --region ",
 { "Ref": "AWS::Region" }, "\"\n",
                    "     MOD=$(( $COUNTER % 2))\n",
                    "\n",
                    "     if [ $MOD -eq 1 ]; then\n",
                    "       METRIC_NAME=`echo $LINE | cut -d ' ' -f $METRIC_FIELD`\n",
                    "     else\n",
                    "       METRIC_VALUE=`echo $LINE | cut -d ' ' -f $DATA_FIELD`\n",
                    "     fi\n",
                    "\n",
                    "     if [[ -n \"$METRIC_NAME\" && -n \"$METRIC_VALUE\" ]]; then\n",
                    "       INSTANCE_ID=$(curl -s http://169.254.169.254/latest/meta-
data/instance-id)\n",
                    "       CW_JSON_METRIC=\"$CW_JSON_METRIC { \\\"MetricName\\\": \\
\"$METRIC_NAME\\\", \\\"Dimensions\\\": [{\\\"Name\\\": \\\"InstanceId\\\", \\\"Value\\\":
 \\\"$INSTANCE_ID\\\"} ], \\\"Value\\\": $METRIC_VALUE },\"\n",
                    "       unset METRIC_NAME\n",
                    "       unset METRIC_VALUE\n",
                    "\n",
                    "       METRIC_COUNTER=$((METRIC_COUNTER+1))\n",
                    "       if [ $METRIC_COUNTER -eq 20 ]; then\n",
                    "         # 20 is max metric collection size, so we have to submit
 here\n",
                    "         aws cloudwatch put-metric-data --region ", { "Ref":
 "AWS::Region" }, " --cli-input-json \"`echo $CW_JSON_OPEN ${CW_JSON_METRIC%?}
 $CW_JSON_CLOSE`\"\n",
                    "\n",
                    "         # reset\n",
                    "         METRIC_COUNTER=0\n",
                    "         CW_JSON_METRIC=''\n",
                    "       fi\n",
                    "     fi  \n",
                    "\n",
                    "\n",
                    "\n",
                    "     COUNTER=$((COUNTER+1))\n",
                    "   fi\n",
```

```
                        "\n",
                        "   if [[ \"$line\" == \"Client nfs v4:\" ]]; then\n",
                        "     # the next line is the good stuff \n",
                        "     COUNTER=$((COUNTER+1))\n",
                        "   fi\n",
                        " done <<< \"$INPUT\"\n",
                        "done\n",
                        "\n",
                        "# submit whatever is left\n",
                        "aws cloudwatch put-metric-data --region ", { "Ref": "AWS::Region" },
 " --cli-input-json \"`echo $CW_JSON_OPEN ${CW_JSON_METRIC%?} $CW_JSON_CLOSE`\""
                      ] ] },
                "mode": "000755",
                "owner": "ec2-user",
                "group": "ec2-user"
              },
              "/home/ec2-user/crontab" : {
                "content" : { "Fn::Join" : [ "", [
                  "* * * * * /usr/sbin/nfsstat | /home/ec2-user/post_nfsstat\n"
                ] ] },
                "owner": "ec2-user",
                "group": "ec2-user"
              }
            },
            "commands" : {
              "01_createdir" : {
                "command" : {"Fn::Join" : [ "", [ "mkdir /", { "Ref" : "MountPoint" }]]}
              }
            }
          },
          "mount" : {
            "commands" : {
              "01_mount" : {
                "command" : { "Fn::Sub": "sudo mount -t nfs4 -o
 nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2 ${FileSystem}.efs.
${AWS::Region}.amazonaws.com:/ /${MountPoint}"}
              },
              "02_permissions" : {
                "command" : {"Fn::Join" : [ "", [ "chown ec2-user:ec2-user /", { "Ref" :
 "MountPoint" }]]}
              }
            }
          }
        }
      },
      "Properties": {
        "AssociatePublicIpAddress" : true,
        "ImageId": {
          "Fn::FindInMap": [ "AWSRegionArch2AMI", { "Ref": "AWS::Region" }, {
            "Fn::FindInMap": [ "AWSInstanceType2Arch", { "Ref": "InstanceType" }, "Arch" ]
          } ]
        },
        "InstanceType": { "Ref": "InstanceType" },
        "KeyName": { "Ref": "KeyName" },
        "SecurityGroups": [ { "Ref": "InstanceSecurityGroup" } ],
        "IamInstanceProfile" : { "Ref" : "CloudWatchPutMetricsInstanceProfile" },
        "UserData"       : { "Fn::Base64" : { "Fn::Join" : ["", [
             "#!/bin/bash -xe\n",
             "yum install -y aws-cfn-bootstrap\n",

             "/opt/aws/bin/cfn-init -v ",
             "         --stack ", { "Ref" : "AWS::StackName" },
             "         --resource LaunchConfiguration ",
             "         --configsets MountConfig ",
             "         --region ", { "Ref" : "AWS::Region" }, "\n",
```

```
            "crontab /home/ec2-user/crontab\n",

            "/opt/aws/bin/cfn-signal -e $? ",
            "         --stack ", { "Ref" : "AWS::StackName" },
            "         --resource AutoScalingGroup ",
            "         --region ", { "Ref" : "AWS::Region" }, "\n"
        ]]}}
      }
    },
    "AutoScalingGroup": {
      "Type": "AWS::AutoScaling::AutoScalingGroup",
      "DependsOn": ["MountTarget", "GatewayToInternet"],
      "CreationPolicy" : {
        "ResourceSignal" : {
          "Timeout" : "PT15M",
          "Count"   : { "Ref": "AsgMaxSize" }
        }
      },
      "Properties": {
        "VPCZoneIdentifier": [ { "Ref": "Subnet" } ],
        "LaunchConfigurationName": { "Ref": "LaunchConfiguration" },
        "MinSize": "1",
        "MaxSize": { "Ref": "AsgMaxSize" },
        "DesiredCapacity": { "Ref": "AsgMaxSize" },
        "Tags": [ {
          "Key": "Name",
          "Value": "EFS FileSystem Mounted Instance",
          "PropagateAtLaunch": "true"
        } ]
      }
    }
  },
  "Outputs" : {
    "MountTargetID" : {
      "Description" : "Mount target ID",
      "Value" :  { "Ref" : "MountTarget" }
    },
    "FileSystemID" : {
      "Description" : "File system ID",
      "Value" :  { "Ref" : "FileSystem" }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: This template creates an Amazon EFS file system and mount target and
  associates it with Amazon EC2 instances in an Auto Scaling group. **WARNING** This
  template creates Amazon EC2 instances and related resources. You will be billed
  for the AWS resources used if you create a stack from this template.
Parameters:
  InstanceType:
    Description: WebServer EC2 instance type
    Type: String
    Default: m1.small
    AllowedValues:
    - t1.micro
    - t2.micro
    - t2.small
    - t2.medium
    - m1.small
    - m1.medium
    - m1.large
```

```
    - m1.xlarge
    - m2.xlarge
    - m2.2xlarge
    - m2.4xlarge
    - m3.medium
    - m3.large
    - m3.xlarge
    - m3.2xlarge
    - c1.medium
    - c1.xlarge
    - c3.large
    - c3.xlarge
    - c3.2xlarge
    - c3.4xlarge
    - c3.8xlarge
    - c4.large
    - c4.xlarge
    - c4.2xlarge
    - c4.4xlarge
    - c4.8xlarge
    - g2.2xlarge
    - r3.large
    - r3.xlarge
    - r3.2xlarge
    - r3.4xlarge
    - r3.8xlarge
    - i2.xlarge
    - i2.2xlarge
    - i2.4xlarge
    - i2.8xlarge
    - d2.xlarge
    - d2.2xlarge
    - d2.4xlarge
    - d2.8xlarge
    - hi1.4xlarge
    - hs1.8xlarge
    - cr1.8xlarge
    - cc2.8xlarge
    - cg1.4xlarge
    ConstraintDescription: Must be a valid EC2 instance type.
  KeyName:
    Type: AWS::EC2::KeyPair::KeyName
    Description: Name of an existing EC2 key pair to enable SSH access to the ECS
      instances
  AsgMaxSize:
    Type: Number
    Description: Maximum size and initial desired capacity of Auto Scaling Group
    Default: '2'
  SSHLocation:
    Description: The IP address range that can be used to connect to the EC2 instances
      by using SSH
    Type: String
    MinLength: '9'
    MaxLength: '18'
    Default: 0.0.0.0/0
    AllowedPattern: "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\\d{1,2})"
    ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.
  VolumeName:
    Description: The name to be used for the EFS volume
    Type: String
    MinLength: '1'
    Default: myEFSvolume
  MountPoint:
    Description: The Linux mount point for the EFS volume
    Type: String
    MinLength: '1'
```

```
    Default: myEFSvolume
Mappings:
  AWSInstanceType2Arch:
    t1.micro:
      Arch: PV64
    t2.micro:
      Arch: HVM64
    t2.small:
      Arch: HVM64
    t2.medium:
      Arch: HVM64
    m1.small:
      Arch: PV64
    m1.medium:
      Arch: PV64
    m1.large:
      Arch: PV64
    m1.xlarge:
      Arch: PV64
    m2.xlarge:
      Arch: PV64
    m2.2xlarge:
      Arch: PV64
    m2.4xlarge:
      Arch: PV64
    m3.medium:
      Arch: HVM64
    m3.large:
      Arch: HVM64
    m3.xlarge:
      Arch: HVM64
    m3.2xlarge:
      Arch: HVM64
    c1.medium:
      Arch: PV64
    c1.xlarge:
      Arch: PV64
    c3.large:
      Arch: HVM64
    c3.xlarge:
      Arch: HVM64
    c3.2xlarge:
      Arch: HVM64
    c3.4xlarge:
      Arch: HVM64
    c3.8xlarge:
      Arch: HVM64
    c4.large:
      Arch: HVM64
    c4.xlarge:
      Arch: HVM64
    c4.2xlarge:
      Arch: HVM64
    c4.4xlarge:
      Arch: HVM64
    c4.8xlarge:
      Arch: HVM64
    g2.2xlarge:
      Arch: HVMG2
    r3.large:
      Arch: HVM64
    r3.xlarge:
      Arch: HVM64
    r3.2xlarge:
      Arch: HVM64
    r3.4xlarge:
```

```
      Arch: HVM64
    r3.8xlarge:
      Arch: HVM64
    i2.xlarge:
      Arch: HVM64
    i2.2xlarge:
      Arch: HVM64
    i2.4xlarge:
      Arch: HVM64
    i2.8xlarge:
      Arch: HVM64
    d2.xlarge:
      Arch: HVM64
    d2.2xlarge:
      Arch: HVM64
    d2.4xlarge:
      Arch: HVM64
    d2.8xlarge:
      Arch: HVM64
    hi1.4xlarge:
      Arch: HVM64
    hs1.8xlarge:
      Arch: HVM64
    cr1.8xlarge:
      Arch: HVM64
    cc2.8xlarge:
      Arch: HVM64
  AWSRegionArch2AMI:
    us-east-1:
      PV64: ami-1ccae774
      HVM64: ami-1ecae776
      HVMG2: ami-8c6b40e4
    us-west-2:
      PV64: ami-ff527ecf
      HVM64: ami-e7527ed7
      HVMG2: ami-abbe919b
    us-west-1:
      PV64: ami-d514f291
      HVM64: ami-d114f295
      HVMG2: ami-f31ffeb7
    eu-west-1:
      PV64: ami-bf0897c8
      HVM64: ami-a10897d6
      HVMG2: ami-d5bc24a2
    eu-central-1:
      PV64: ami-ac221fb1
      HVM64: ami-a8221fb5
      HVMG2: ami-7cd2ef61
    ap-northeast-1:
      PV64: ami-27f90e27
      HVM64: ami-cbf90ecb
      HVMG2: ami-6318e863
    ap-southeast-1:
      PV64: ami-acd9e8fe
      HVM64: ami-68d8e93a
      HVMG2: ami-3807376a
    ap-southeast-2:
      PV64: ami-ff9cecc5
      HVM64: ami-fd9cecc7
      HVMG2: ami-89790ab3
    sa-east-1:
      PV64: ami-bb2890a6
      HVM64: ami-b52890a8
      HVMG2: NOT_SUPPORTED
    cn-north-1:
      PV64: ami-fa39abc3
```

```
      HVM64: ami-f239abcb
      HVMG2: NOT_SUPPORTED
Resources:
  CloudWatchPutMetricsRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
        - Effect: Allow
          Principal:
            Service:
            - ec2.amazonaws.com
          Action:
          - sts:AssumeRole
      Path: "/"
  CloudWatchPutMetricsRolePolicy:
    Type: AWS::IAM::Policy
    Properties:
      PolicyName: CloudWatch_PutMetricData
      PolicyDocument:
        Version: '2012-10-17'
        Statement:
        - Sid: CloudWatchPutMetricData
          Effect: Allow
          Action:
          - cloudwatch:PutMetricData
          Resource:
          - "*"
      Roles:
      - Ref: CloudWatchPutMetricsRole
  CloudWatchPutMetricsInstanceProfile:
    Type: AWS::IAM::InstanceProfile
    Properties:
      Path: "/"
      Roles:
      - Ref: CloudWatchPutMetricsRole
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      EnableDnsSupport: 'true'
      EnableDnsHostnames: 'true'
      CidrBlock: 10.0.0.0/16
      Tags:
      - Key: Application
        Value:
          Ref: AWS::StackId
  InternetGateway:
    Type: AWS::EC2::InternetGateway
    Properties:
      Tags:
      - Key: Application
        Value:
          Ref: AWS::StackName
      - Key: Network
        Value: Public
  GatewayToInternet:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      VpcId:
        Ref: VPC
      InternetGatewayId:
        Ref: InternetGateway
  RouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId:
```

```
      Ref: VPC
  SubnetRouteTableAssoc:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId:
        Ref: RouteTable
      SubnetId:
        Ref: Subnet
  InternetGatewayRoute:
    Type: AWS::EC2::Route
    Properties:
      DestinationCidrBlock: 0.0.0.0/0
      RouteTableId:
        Ref: RouteTable
      GatewayId:
        Ref: InternetGateway
  Subnet:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId:
        Ref: VPC
      CidrBlock: 10.0.0.0/24
      Tags:
      - Key: Application
        Value:
          Ref: AWS::StackId
  InstanceSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      VpcId:
        Ref: VPC
      GroupDescription: Enable SSH access via port 22
      SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: '22'
        ToPort: '22'
        CidrIp:
          Ref: SSHLocation
      - IpProtocol: tcp
        FromPort: '80'
        ToPort: '80'
        CidrIp: 0.0.0.0/0
  MountTargetSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      VpcId:
        Ref: VPC
      GroupDescription: Security group for mount target
      SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: '2049'
        ToPort: '2049'
        CidrIp: 0.0.0.0/0
  FileSystem:
    Type: AWS::EFS::FileSystem
    Properties:
      PerformanceMode: generalPurpose
      FileSystemTags:
      - Key: Name
        Value:
          Ref: VolumeName
  MountTarget:
    Type: AWS::EFS::MountTarget
    Properties:
      FileSystemId:
        Ref: FileSystem
```

```
      SubnetId:
        Ref: Subnet
      SecurityGroups:
      - Ref: MountTargetSecurityGroup
  LaunchConfiguration:
    Type: AWS::AutoScaling::LaunchConfiguration
    Metadata:
      AWS::CloudFormation::Init:
        configSets:
          MountConfig:
          - setup
          - mount
        setup:
          packages:
            yum:
              nfs-utils: []
          files:
            "/home/ec2-user/post_nfsstat":
              content: !Sub |
                #!/bin/bash

                INPUT="$(cat)"
                CW_JSON_OPEN='{ "Namespace": "EFS", "MetricData": [ '
                CW_JSON_CLOSE=' ] }'
                CW_JSON_METRIC=''
                METRIC_COUNTER=0

                for COL in 1 2 3 4 5 6; do

                 COUNTER=0
                 METRIC_FIELD=$COL
                 DATA_FIELD=$(($COL+($COL-1)))

                 while read line; do
                   if [[ COUNTER -gt 0 ]]; then

                     LINE=`echo $line | tr -s ' ' `
                     AWS_COMMAND="aws cloudwatch put-metric-data --region ${AWS::Region}"
                     MOD=$(( $COUNTER % 2))

                     if [ $MOD -eq 1 ]; then
                       METRIC_NAME=`echo $LINE | cut -d ' ' -f $METRIC_FIELD`
                     else
                       METRIC_VALUE=`echo $LINE | cut -d ' ' -f $DATA_FIELD`
                     fi

                     if [[ -n "$METRIC_NAME" && -n "$METRIC_VALUE" ]]; then
                       INSTANCE_ID=$(curl -s http://169.254.169.254/latest/meta-data/
instance-id)
                       CW_JSON_METRIC="$CW_JSON_METRIC { \"MetricName\": \"$METRIC_NAME\",
 \"Dimensions\": [{\"Name\": \"InstanceId\", \"Value\": \"$INSTANCE_ID\"} ], \"Value\":
 $METRIC_VALUE },"
                       unset METRIC_NAME
                       unset METRIC_VALUE

                       METRIC_COUNTER=$((METRIC_COUNTER+1))
                       if [ $METRIC_COUNTER -eq 20 ]; then
                         # 20 is max metric collection size, so we have to submit here
                         aws cloudwatch put-metric-data --region ${AWS::Region} --cli-
input-json "`echo $CW_JSON_OPEN ${!CW_JSON_METRIC%?} $CW_JSON_CLOSE`"

                         # reset
                         METRIC_COUNTER=0
                         CW_JSON_METRIC=''
                       fi
                     fi
```

```
                  COUNTER=$((COUNTER+1))
                fi

                if [[ "$line" == "Client nfs v4:" ]]; then
                  # the next line is the good stuff
                  COUNTER=$((COUNTER+1))
                fi
              done <<< "$INPUT"
            done

            # submit whatever is left
            aws cloudwatch put-metric-data --region ${AWS::Region} --cli-input-json
 "`echo $CW_JSON_OPEN ${!CW_JSON_METRIC%?} $CW_JSON_CLOSE`"
          mode: '000755'
          owner: ec2-user
          group: ec2-user
        "/home/ec2-user/crontab":
          content: "* * * * * /usr/sbin/nfsstat | /home/ec2-user/post_nfsstat\n"
          owner: ec2-user
          group: ec2-user
      commands:
        01_createdir:
          command: !Sub "mkdir /${MountPoint}"
    mount:
      commands:
        01_mount:
          command: !Sub >
            mount -t nfs4 -o nfsvers=4.1 ${FileSystem}.efs.
${AWS::Region}.amazonaws.com:/ /${MountPoint}
        02_permissions:
          command: !Sub "chown ec2-user:ec2-user /${MountPoint}"
    Properties:
      AssociatePublicIpAddress: true
      ImageId:
        Fn::FindInMap:
        - AWSRegionArch2AMI
        - Ref: AWS::Region
        - Fn::FindInMap:
          - AWSInstanceType2Arch
          - Ref: InstanceType
          - Arch
      InstanceType:
        Ref: InstanceType
      KeyName:
        Ref: KeyName
      SecurityGroups:
      - Ref: InstanceSecurityGroup
      IamInstanceProfile:
        Ref: CloudWatchPutMetricsInstanceProfile
      UserData:
        Fn::Base64: !Sub |
          #!/bin/bash -xe
          yum install -y aws-cfn-bootstrap
          /opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --resource LaunchConfiguration
 --configsets MountConfig --region ${AWS::Region}
          crontab /home/ec2-user/crontab
          /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --resource
 AutoScalingGroup --region ${AWS::Region}
  AutoScalingGroup:
    Type: AWS::AutoScaling::AutoScalingGroup
    DependsOn:
    - MountTarget
    - GatewayToInternet
```

```
    CreationPolicy:
      ResourceSignal:
        Timeout: PT15M
        Count:
          Ref: AsgMaxSize
    Properties:
      VPCZoneIdentifier:
      - Ref: Subnet
      LaunchConfigurationName:
        Ref: LaunchConfiguration
      MinSize: '1'
      MaxSize:
        Ref: AsgMaxSize
      DesiredCapacity:
        Ref: AsgMaxSize
      Tags:
      - Key: Name
        Value: EFS FileSystem Mounted Instance
        PropagateAtLaunch: 'true'
Outputs:
  MountTargetID:
    Description: Mount target ID
    Value:
      Ref: MountTarget
  FileSystemID:
    Description: File system ID
    Value:
      Ref: FileSystem
```

Elastic Beanstalk Template Snippets

With Elastic Beanstalk, you can quickly deploy and manage applications in AWS without worrying about the infrastructure that runs those applications. The following sample template can help you describe Elastic Beanstalk resources in your AWS CloudFormation template.

Elastic Beanstalk Sample PHP

The following sample template deploys a sample PHP web application that is stored in an Amazon S3 bucket. The Elastic Beanstalk environment is 64-bit Amazon Linux running PHP 5.3. The environment is also an autoscaling, load-balancing environment, with a minimum of two Amazon EC2 instances and a maximum of six.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Resources": {
    "sampleApplication": {
      "Type": "AWS::ElasticBeanstalk::Application",
      "Properties": {
        "Description": "AWS Elastic Beanstalk Sample Application"
      }
    },
    "sampleApplicationVersion": {
      "Type": "AWS::ElasticBeanstalk::ApplicationVersion",
      "Properties": {
        "ApplicationName": { "Ref": "sampleApplication" },
        "Description": "AWS ElasticBeanstalk Sample Application Version",
        "SourceBundle": {
          "S3Bucket": { "Fn::Join": [ "-", [ "elasticbeanstalk-samples", { "Ref":
 "AWS::Region" } ] ] },
```

```
          "S3Key": "php-sample.zip"
        }
      }
    },
    "sampleConfigurationTemplate": {
      "Type": "AWS::ElasticBeanstalk::ConfigurationTemplate",
      "Properties": {
        "ApplicationName": { "Ref": "sampleApplication" },
        "Description": "AWS ElasticBeanstalk Sample Configuration Template",
        "OptionSettings": [
          {
            "Namespace": "aws:autoscaling:asg",
            "OptionName": "MinSize",
            "Value": "2"
          },
          {
            "Namespace": "aws:autoscaling:asg",
            "OptionName": "MaxSize",
            "Value": "6"
          },
          {
            "Namespace": "aws:elasticbeanstalk:environment",
            "OptionName": "EnvironmentType",
            "Value": "LoadBalanced"
          }
        ],
        "SolutionStackName": "64bit Amazon Linux running PHP 5.3"
      }
    },
    "sampleEnvironment": {
      "Type": "AWS::ElasticBeanstalk::Environment",
      "Properties": {
        "ApplicationName": { "Ref": "sampleApplication" },
        "Description": "AWS ElasticBeanstalk Sample Environment",
        "TemplateName": { "Ref": "sampleConfigurationTemplate" },
        "VersionLabel": { "Ref": "sampleApplicationVersion" }
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  sampleApplication:
    Type: AWS::ElasticBeanstalk::Application
    Properties:
      Description: AWS Elastic Beanstalk Sample Application
  sampleApplicationVersion:
    Type: AWS::ElasticBeanstalk::ApplicationVersion
    Properties:
      ApplicationName:
        Ref: sampleApplication
      Description: AWS ElasticBeanstalk Sample Application Version
      SourceBundle:
        S3Bucket: !Sub "elasticbeanstalk-samples-${AWS::Region}"
        S3Key: php-sample.zip
  sampleConfigurationTemplate:
    Type: AWS::ElasticBeanstalk::ConfigurationTemplate
    Properties:
      ApplicationName:
        Ref: sampleApplication
      Description: AWS ElasticBeanstalk Sample Configuration Template
```

```
      OptionSettings:
      - Namespace: aws:autoscaling:asg
        OptionName: MinSize
        Value: '2'
      - Namespace: aws:autoscaling:asg
        OptionName: MaxSize
        Value: '6'
      - Namespace: aws:elasticbeanstalk:environment
        OptionName: EnvironmentType
        Value: LoadBalanced
      SolutionStackName: 64bit Amazon Linux running PHP 5.3
  sampleEnvironment:
    Type: AWS::ElasticBeanstalk::Environment
    Properties:
      ApplicationName:
        Ref: sampleApplication
      Description: AWS ElasticBeanstalk Sample Environment
      TemplateName:
        Ref: sampleConfigurationTemplate
      VersionLabel:
        Ref: sampleApplicationVersion
```

Elastic Load Balancing Template Snippets

Topics

Elastic Load Balancing Load Balancer Resource

This example shows an Elastic Load Balancing load balancer with a single listener, and no instances.

JSON

```
"MyLoadBalancer" : {
    "Type" : "AWS::ElasticLoadBalancing::LoadBalancer",
    "Properties" : {
        "AvailabilityZones" : [ "us-east-1a" ],
        "Listeners" : [ {
            "LoadBalancerPort" : "80",
            "InstancePort" : "80",
            "Protocol" : "HTTP"
        } ]
    }
}
```

YAML

```
MyLoadBalancer:
  Type: AWS::ElasticLoadBalancing::LoadBalancer
  Properties:
    AvailabilityZones:
    - "us-east-1a"
    Listeners:
    - LoadBalancerPort: '80'
      InstancePort: '80'
      Protocol: HTTP
```

Elastic Load Balancing Load Balancer Resource with Health Check

This example shows an Elastic Load Balancing load balancer with two Amazon EC2 instances, a single listener and a health check.

JSON

```
"MyLoadBalancer" : {
    "Type" : "AWS::ElasticLoadBalancing::LoadBalancer",
    "Properties" : {
        "AvailabilityZones" : [ "us-east-1a" ],
        "Instances" : [
            { "Ref" : "logical name of AWS::EC2::Instance resource 1" },
            { "Ref" : "logical name of AWS::EC2::Instance resource 2" }
        ],
        "Listeners" : [ {
            "LoadBalancerPort" : "80",
            "InstancePort" : "80",
            "Protocol" : "HTTP"
        } ],

        "HealthCheck" : {
            "Target" : "HTTP:80/",
            "HealthyThreshold" : "3",
            "UnhealthyThreshold" : "5",
            "Interval" : "30",
            "Timeout" : "5"
        }
    }
}
```

YAML

```
MyLoadBalancer:
  Type: AWS::ElasticLoadBalancing::LoadBalancer
  Properties:
    AvailabilityZones:
    - "us-east-1a"
    Instances:
    - Ref: logical name of AWS::EC2::Instance resource 1
    - Ref: logical name of AWS::EC2::Instance resource 2
    Listeners:
    - LoadBalancerPort: '80'
      InstancePort: '80'
      Protocol: HTTP
    HealthCheck:
      Target: HTTP:80/
      HealthyThreshold: '3'
      UnhealthyThreshold: '5'
      Interval: '30'
      Timeout: '5'
```

AWS Identity and Access Management Template Snippets

This section contains AWS Identity and Access Management template snippets.

Topics

Important
When creating or updating a stack using a template containing IAM resources, you must acknowledge the use of IAM capabilities. For more information about using IAM resources in templates, see Controlling Access with AWS Identity and Access Management (p. 8).

Declaring an IAM User Resource

This snippet shows how to declare an `AWS::IAM::User` (p. 793) resource to create an IAM user. The user is declared with the path (`"/"`) and a login profile with the password (`myP@ssW0rd`).

The policy document named `giveaccesstoqueueonly` gives the user permission to perform all Amazon SQS actions on the Amazon SQS queue resource `myqueue`, and denies access to all other Amazon SQS queue resources. The `Fn::GetAtt` (p. 1324) function gets the Arn attribute of the `AWS::SQS::Queue` (p. 958) resource `myqueue`.

The policy document named `giveaccesstotopiconly` is added to the user to give the user permission to perform all Amazon SNS actions on the Amazon SNS topic resource `mytopic` and to deny access to all other Amazon SNS resources. The `Ref` (p. 1343) function gets the ARN of the `AWS::SNS::Topic` (p. 954) resource `mytopic`.

JSON

```
"myuser" : {
   "Type" : "AWS::IAM::User",
   "Properties" : {
      "Path" : "/",
      "LoginProfile" : {
         "Password" : "myP@ssW0rd"
      },
      "Policies" : [ {
         "PolicyName" : "giveaccesstoqueueonly",
         "PolicyDocument" : {
            "Version": "2012-10-17",
            "Statement" : [ {
               "Effect" : "Allow",
               "Action" : [ "sqs:*" ],
               "Resource" : [ {
                  "Fn::GetAtt" : [ "myqueue", "Arn" ]
               } ]
            }, {
               "Effect" : "Deny",
               "Action" : [ "sqs:*" ],
               "NotResource" : [ {
                  "Fn::GetAtt" : [ "myqueue", "Arn" ]
               } ]
            }
```

```
            ] }
        }, {
            "PolicyName" : "giveaccesstotopiconly",
            "PolicyDocument" : {
                "Version": "2012-10-17",
                "Statement" : [ {
                    "Effect" : "Allow",
                    "Action" : [ "sns:*" ],
                    "Resource" : [ { "Ref" : "mytopic" } ]
                }, {
                    "Effect" : "Deny",
                    "Action" : [ "sns:*" ],
                    "NotResource" : [ { "Ref" : "mytopic" } ]
                } ]
            }
        } ]
    }
}
```

YAML

```
myuser:
  Type: AWS::IAM::User
  Properties:
    Path: "/"
    LoginProfile:
      Password: myP@ssW0rd
    Policies:
    - PolicyName: giveaccesstoqueueonly
      PolicyDocument:
        Version: '2012-10-17'
        Statement:
        - Effect: Allow
          Action:
          - sqs:*
          Resource:
          - !GetAtt myqueue.Arn
        - Effect: Deny
          Action:
          - sqs:*
          NotResource:
          - !GetAtt myqueue.Arn
    - PolicyName: giveaccesstotopiconly
      PolicyDocument:
        Version: '2012-10-17'
        Statement:
        - Effect: Allow
          Action:
          - sns:*
          Resource:
          - !Ref: mytopic
        - Effect: Deny
          Action:
          - sns:*
          NotResource:
          - !Ref: mytopic
```

Declaring an IAM Access Key Resource

This snippet shows an `AWS::IAM::AccessKey` (p. 775) resource. The `myaccesskey` resource creates an access key and assigns it to an IAM user that is declared as an `AWS::IAM::User` (p. 793) resource in the template.

JSON

```
"myaccesskey" : {
   "Type" : "AWS::IAM::AccessKey",
   "Properties" : {
      "UserName" : { "Ref" : "myuser" }
   }
}
```

YAML

```
myaccesskey:
  Type: AWS::IAM::AccessKey
  Properties:
    UserName:
      !Ref: myuser
```

You can get the secret key for an `AWS::IAM::AccessKey` resource using the `Fn::GetAtt` (p. 1324) function. The only time that you can get the secret key for an AWS access key is when it is created. One way to retrieve the secret key is to put it into an `Output` value. You can get the access key using the `Ref` function. The following `Output` value declarations get the access key and secret key for `myaccesskey`.

JSON

```
"AccessKeyformyaccesskey" : {
   "Value" : { "Ref" : "myaccesskey" }
},
"SecretKeyformyaccesskey" : {
   "Value" : {
      "Fn::GetAtt" : [ "myaccesskey", "SecretAccessKey" ]
   }
}
```

YAML

```
AccessKeyformyaccesskey:
  Value:
    !Ref: myaccesskey
SecretKeyformyaccesskey:
  Value: !GetAtt myaccesskey.SecretAccessKey
```

You can also pass the AWS access key and secret key to an EC2 instance or Auto Scaling group defined in the template. The following `AWS::EC2::Instance` (p. 574) declaration uses the `UserData` property to pass the access key and secret key for the `myaccesskey` resource.

JSON

```
"myinstance" : {
   "Type" : "AWS::EC2::Instance",
   "Properties" : {
      "AvailabilityZone" : "us-east-1a",
      "ImageId" : "ami-20b65349",
      "UserData" : {
         "Fn::Base64" : {
            "Fn::Join" : [
```

```
                "", [
                   "ACCESS_KEY=", {
                      "Ref" : "myaccesskey"
                   },
                   "&",
                   "SECRET_KEY=",
                   {
                      "Fn::GetAtt" : [
                         "myaccesskey",
                         "SecretAccessKey"
                      ]
                   }
                ]
             ]
          }
       }
    }
}
```

YAML

```
myinstance:
  Type: AWS::EC2::Instance
  Properties:
    AvailabilityZone: "us-east-1a"
    ImageId: ami-20b65349
    UserData:
      Fn::Base64: !Sub "ACCESS_KEY=${myaccesskey}&SECRET_KEY=${myaccesskey.SecretAccessKey}
```

Declaring an IAM Group Resource

This snippet shows an `AWS::IAM::Group` (p. 777) resource. The group has a path ("`/myapplication/`"). The policy document named `myapppolicy` is added to the group to allow the group's users to perform all Amazon SQS actions on the Amazon SQS queue resource myqueue and deny access to all other Amazon SQS resources except `myqueue`.

To assign a policy to a resource, IAM requires the Amazon Resource Name (ARN) for the resource. In the snippet, the `Fn::GetAtt` (p. 1324) function gets the ARN of the `AWS::SQS::Queue` (p. 958) resource queue.

JSON

```

"mygroup" : {
   "Type" : "AWS::IAM::Group",
   "Properties" : {
      "Path" : "/myapplication/",
      "Policies" : [ {
         "PolicyName" : "myapppolicy",
         "PolicyDocument" : {
            "Version": "2012-10-17",
            "Statement" : [ {
               "Effect" : "Allow",
               "Action" : [ "sqs:*" ],
               "Resource" : [ {
                  "Fn::GetAtt" : [ "myqueue", "Arn" ]
               } ]
            },
            {
               "Effect" : "Deny",
               "Action" : [ "sqs:*" ],
```

```
                "NotResource" : [ { "Fn::GetAtt" : [ "myqueue", "Arn" ] } ]
            }
        ] }
    } ]
  }
}
```

YAML

```
mygroup:
  Type: AWS::IAM::Group
  Properties:
    Path: "/myapplication/"
    Policies:
    - PolicyName: myapppolicy
      PolicyDocument:
        Version: '2012-10-17'
        Statement:
        - Effect: Allow
          Action:
          - sqs:*
          Resource: !GetAtt myqueue.Arn
        - Effect: Deny
          Action:
          - sqs:*
          NotResource: !GetAtt myqueue.Arn
```

Adding Users to a Group

The `AWS::IAM::UserToGroupAddition` (p. 796) resource adds users to a group. In the following snippet, the `addUserToGroup` resource adds the following users to an existing group named `myexistinggroup2`: the existing user `existinguser1` and the user `myuser` which is declared as an `AWS::IAM::User` (p. 793) resource in the template.

JSON

```
"addUserToGroup" : {
   "Type" : "AWS::IAM::UserToGroupAddition",
   "Properties" : {
      "GroupName" : "myexistinggroup2",
      "Users" : [ "existinguser1", { "Ref" : "myuser" } ]
   }
}
```

YAML

```
addUserToGroup:
  Type: AWS::IAM::UserToGroupAddition
  Properties:
    GroupName: myexistinggroup2
    Users:
    - existinguser1
    - !Ref: myuser
```

Declaring an IAM Policy

This snippet shows how to create a policy and apply it to multiple groups using an `AWS::IAM::Policy` (p. 784) resource named `mypolicy`. The `mypolicy` resource contains a

`PolicyDocument` property that allows `GetObject`, `PutObject`, and `PutObjectAcl` actions on the objects in the S3 bucket represented by the ARN `arn:aws:s3:::myAWSBucket`. The `mypolicy` resource applies the policy to an existing group named `myexistinggroup1` and a group `mygroup` that is declared in the template as an `AWS::IAM::Group` (p. 777) resource. This example shows how to apply a policy to a group using the `Groups` property; however, you can alternatively use the `Users` property to add a policy document to a list of users.

Important

The Amazon SNS policy actions that are declared in the `AWS::IAM::Policy` resource (p. 322) differ from the Amazon SNS topic policy actions that are declared in the `AWS::SNS::TopicPolicy` resource (p. 324). For example, the policy actions `sns:Unsubscribe` and `sns:SetSubscriptionAttributes` are valid for the `AWS::IAM::Policy` resource, but are invalid for the `AWS::SNS::TopicPolicy` resource. For more information about valid Amazon SNS policy actions that you can use with the `AWS::IAM::Policy` resource, see Special Information for Amazon SNS Policies in the *Amazon Simple Notification Service Developer Guide*.

JSON

```
"mypolicy" : {
   "Type" : "AWS::IAM::Policy",
   "Properties" : {
      "PolicyName" : "mygrouppolicy",
      "PolicyDocument" : {
         "Version": "2012-10-17",
         "Statement" : [ {
            "Effect" : "Allow",
            "Action" : [
               "s3:GetObject" , "s3:PutObject" , "s3:PutObjectAcl" ],
            "Resource" : "arn:aws:s3:::myAWSBucket/*"
         } ]
      },
      "Groups" : [ "myexistinggroup1", { "Ref" : "mygroup" } ]
   }
}
```

YAML

```
mypolicy:
  Type: AWS::IAM::Policy
  Properties:
    PolicyName: mygrouppolicy
    PolicyDocument:
      Version: '2012-10-17'
      Statement:
      - Effect: Allow
        Action:
        - s3:GetObject
        - s3:PutObject
        - s3:PutObjectAcl
        Resource: arn:aws:s3:::myAWSBucket/*
    Groups:
    - myexistinggroup1
    - !Ref: mygroup
```

Declaring an Amazon S3 Bucket Policy

This snippet shows how to create a policy and apply it to an Amazon S3 bucket using the `AWS::S3::BucketPolicy` (p. 950) resource. The `mybucketpolicy` resource declares a policy document that allows the `user1` IAM user to perform the `GetObject` action on all objects in the S3 bucket to which this policy is applied. In the snippet, the `Fn::GetAtt` (p. 1324) function gets the ARN of the `user1` resource. The `mybucketpolicy` resource applies the policy to the `AWS::S3::Bucket` (p. 937) resource mybucket. The `Ref` (p. 1343) function gets the bucket name of the `mybucket` resource.

JSON

```
"mybucketpolicy" : {
   "Type" : "AWS::S3::BucketPolicy",
   "Properties" : {
      "PolicyDocument" : {
         "Id" : "MyPolicy",
         "Version": "2012-10-17",
         "Statement" : [ {
            "Sid" : "ReadAccess",
            "Action" : [ "s3:GetObject" ],
            "Effect" : "Allow",
            "Resource" : { "Fn::Join" : [
                  "", [ "arn:aws:s3:::", { "Ref" : "mybucket" } , "/*" ]
               ] },
            "Principal" : {
               "AWS" : { "Fn::GetAtt" : [ "user1", "Arn" ] }
            }
         } ]
      },
      "Bucket" : { "Ref" : "mybucket" }
   }
}
```

YAML

```
mybucketpolicy:
  Type: AWS::S3::BucketPolicy
  Properties:
    PolicyDocument:
      Id: MyPolicy
      Version: '2012-10-17'
      Statement:
      - Sid: ReadAccess
        Action:
        - s3:GetObject
        Effect: Allow
        Resource: !Sub "arn:aws:s3:::${mybucket}/*"
        Principal:
          AWS: !GetAtt user1.Arn
    Bucket: !Ref mybucket
```

Declaring an Amazon SNS Topic Policy

This snippet shows how to create a policy and apply it to an Amazon SNS topic using the `AWS::SNS::TopicPolicy` (p. 957) resource. The `mysnspolicy` resource contains a `PolicyDocument` property that allows the `AWS::IAM::User` (p. 793) resource `myuser` to perform the `Publish` action on an `AWS::SNS::Topic` (p. 954) resource `mytopic`. In the snippet, the `Fn::GetAtt` (p. 1324) function gets the ARN for the `myuser` resource and the `Ref` (p. 1343) function gets the ARN for the `mytopic` resource.

> **Important**
> The Amazon SNS policy actions that are declared in the `AWS::IAM::Policy` resource (p. 322) differ from the Amazon SNS topic policy actions that are declared in the `AWS::SNS::TopicPolicy` resource (p. 324). For example, the policy actions `sns:Unsubscribe` and `sns:SetSubscriptionAttributes` are valid for the `AWS::IAM::Policy` resource, but are

invalid for the `AWS::SNS::TopicPolicy` resource. For more information about valid Amazon SNS policy actions that you can use with the `AWS::IAM::Policy` resource, see Special Information for Amazon SNS Policies in the *Amazon Simple Notification Service Developer Guide*.

JSON

```
"mysnspolicy" : {
   "Type" : "AWS::SNS::TopicPolicy",
   "Properties" : {
      "PolicyDocument" :  {
         "Id" : "MyTopicPolicy",
         "Version" : "2012-10-17",
         "Statement" : [ {
            "Sid" : "My-statement-id",
            "Effect" : "Allow",
            "Principal" : {
               "AWS" : { "Fn::GetAtt" : [ "myuser", "Arn" ] }
            },
            "Action" : "sns:Publish",
            "Resource" : "*"
         } ]
      },
      "Topics" : [ { "Ref" : "mytopic" } ]
   }
}
```

YAML

```
mysnspolicy:
  Type: AWS::SNS::TopicPolicy
  Properties:
    PolicyDocument:
      Id: MyTopicPolicy
      Version: '2012-10-17'
      Statement:
      - Sid: My-statement-id
        Effect: Allow
        Principal:
          AWS: !GetAtt myuser.Arn
        Action: sns:Publish
        Resource: "*"
    Topics:
    - !Ref: mytopic
```

Declaring an Amazon SQS Policy

This snippet shows how to create a policy and apply it to an Amazon SQS queue using the `AWS::SQS::QueuePolicy` (p. 965) resource. The `PolicyDocument` property allows the existing user `myapp` (specified by its ARN) to perform the `SendMessage` action on an existing queue, which is specified by its URL, and an `AWS::SQS::Queue` (p. 958) resource myqueue. The Ref (p. 1343) function gets the URL for the `myqueue` resource.

JSON

```
"mysqspolicy" : {
   "Type" : "AWS::SQS::QueuePolicy",
   "Properties" : {
      "PolicyDocument" : {
```

```
         "Id" : "MyQueuePolicy",
         "Version" : "2012-10-17",
         "Statement" : [ {
            "Sid" : "Allow-User-SendMessage",
            "Effect" : "Allow",
            "Principal" : {
               "AWS" : "arn:aws:iam::123456789012:user/myapp"
            },
            "Action" : [ "sqs:SendMessage" ],
            "Resource" : "*"
         } ]
      },
      "Queues" : [
         "https://sqs.us-east-1.amazonaws.com/123456789012/myexistingqueue",
         { "Ref" : "myqueue" }
      ]
   }
}
```

YAML

```
mysqspolicy:
  Type: AWS::SQS::QueuePolicy
  Properties:
    PolicyDocument:
      Id: MyQueuePolicy
      Version: '2012-10-17'
      Statement:
      - Sid: Allow-User-SendMessage
        Effect: Allow
        Principal:
          AWS: arn:aws:iam::123456789012:user/myapp
        Action:
        - sqs:SendMessage
        Resource: "*"
    Queues:
    - https://sqs.us-east-1.amazonaws.com/123456789012/myexistingqueue
    - !Ref: myqueue
```

IAM Role Template Examples

This section provides CloudFormation template examples for IAM Roles for EC2 Instances.

For more information about IAM roles, see Working with Roles in the *AWS Identity and Access Management User Guide*.

IAM Role with EC2

In this example, the instance profile is referenced by the `IamInstanceProfile` property of the EC2 Instance. Both the instance policy and role policy reference `AWS::IAM::Role` (p. 787).

JSON

```
{
   "AWSTemplateFormatVersion": "2010-09-09",
   "Resources": {
      "myEC2Instance": {
         "Type": "AWS::EC2::Instance",
         "Version": "2009-05-15",
         "Properties": {
            "ImageId": "ami-205fba49",
```

```
            "InstanceType": "m1.small",
            "Monitoring": "true",
            "DisableApiTermination": "false",
            "IamInstanceProfile": {
               "Ref": "RootInstanceProfile"
            }
         }
      },
      "RootRole": {
         "Type": "AWS::IAM::Role",
         "Properties": {
            "AssumeRolePolicyDocument": {
               "Version" : "2012-10-17",
               "Statement": [ {
                  "Effect": "Allow",
                  "Principal": {
                     "Service": [ "ec2.amazonaws.com" ]
                  },
                  "Action": [ "sts:AssumeRole" ]
               } ]
            },
            "Path": "/"
         }
      },
      "RolePolicies": {
         "Type": "AWS::IAM::Policy",
         "Properties": {
            "PolicyName": "root",
            "PolicyDocument": {
               "Version" : "2012-10-17",
               "Statement": [ {
                  "Effect": "Allow",
                  "Action": "*",
                  "Resource": "*"
               } ]
            },
            "Roles": [ { "Ref": "RootRole" } ]
         }
      },
      "RootInstanceProfile": {
         "Type": "AWS::IAM::InstanceProfile",
         "Properties": {
            "Path": "/",
            "Roles": [ { "Ref": "RootRole" } ]
         }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  myEC2Instance:
    Type: AWS::EC2::Instance
    Version: '2009-05-15'
    Properties:
      ImageId: ami-205fba49
      InstanceType: m1.small
      Monitoring: 'true'
      DisableApiTermination: 'false'
      IamInstanceProfile:
        !Ref: RootInstanceProfile
  RootRole:
```

```
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
        - Effect: Allow
          Principal:
            Service:
            - ec2.amazonaws.com
          Action:
          - sts:AssumeRole
      Path: "/"
  RolePolicies:
    Type: AWS::IAM::Policy
    Properties:
      PolicyName: root
      PolicyDocument:
        Version: '2012-10-17'
        Statement:
        - Effect: Allow
          Action: "*"
          Resource: "*"
      Roles:
      - !Ref: RootRole
  RootInstanceProfile:
    Type: AWS::IAM::InstanceProfile
    Properties:
      Path: "/"
      Roles:
      - !Ref: RootRole
```

IAM Role with AutoScaling Group

In this example, the instance profile is referenced by the `IamInstanceProfile` property of an AutoScaling Group Launch Configuration.

JSON

```
{
   "AWSTemplateFormatVersion": "2010-09-09",
   "Resources": {
      "myLCOne": {
         "Type": "AWS::AutoScaling::LaunchConfiguration",
         "Version": "2009-05-15",
         "Properties": {
            "ImageId": "ami-205fba49",
            "InstanceType": "m1.small",
            "InstanceMonitoring": "true",
            "IamInstanceProfile": { "Ref": "RootInstanceProfile" }
         }
      },
      "myASGrpOne": {
         "Type": "AWS::AutoScaling::AutoScalingGroup",
         "Version": "2009-05-15",
         "Properties": {
            "AvailabilityZones": [ "us-east-1a" ],
            "LaunchConfigurationName": { "Ref": "myLCOne" },
            "MinSize": "0",
            "MaxSize": "0",
            "HealthCheckType": "EC2",
            "HealthCheckGracePeriod": "120"
         }
      },
      "RootRole": {
```

```
        "Type": "AWS::IAM::Role",
        "Properties": {
           "AssumeRolePolicyDocument": {
              "Version" : "2012-10-17",
              "Statement": [ {
                 "Effect": "Allow",
                 "Principal": {
                    "Service": [ "ec2.amazonaws.com" ]
                 },
                 "Action": [ "sts:AssumeRole" ]
              } ]
           },
           "Path": "/"
        }
     },
     "RolePolicies": {
        "Type": "AWS::IAM::Policy",
        "Properties": {
           "PolicyName": "root",
           "PolicyDocument": {
              "Version" : "2012-10-17",
              "Statement": [ {
                 "Effect": "Allow",
                 "Action": "*",
                 "Resource": "*"
              } ]
           },
           "Roles": [ { "Ref": "RootRole" } ]
        }
     },
     "RootInstanceProfile": {
        "Type": "AWS::IAM::InstanceProfile",
        "Properties": {
           "Path": "/",
           "Roles": [ { "Ref": "RootRole" } ]
        }
     }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  myLCOne:
    Type: AWS::AutoScaling::LaunchConfiguration
    Version: '2009-05-15'
    Properties:
      ImageId: ami-205fba49
      InstanceType: m1.small
      InstanceMonitoring: 'true'
      IamInstanceProfile:
        !Ref: RootInstanceProfile
  myASGrpOne:
    Type: AWS::AutoScaling::AutoScalingGroup
    Version: '2009-05-15'
    Properties:
      AvailabilityZones:
      - "us-east-1a"
      LaunchConfigurationName:
        !Ref: myLCOne
      MinSize: '0'
      MaxSize: '0'
      HealthCheckType: EC2
```

```
      HealthCheckGracePeriod: '120'
  RootRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
        - Effect: Allow
          Principal:
            Service:
            - ec2.amazonaws.com
          Action:
          - sts:AssumeRole
      Path: "/"
  RolePolicies:
    Type: AWS::IAM::Policy
    Properties:
      PolicyName: root
      PolicyDocument:
        Version: '2012-10-17'
        Statement:
        - Effect: Allow
          Action: "*"
          Resource: "*"
      Roles:
      - !Ref: RootRole
  RootInstanceProfile:
    Type: AWS::IAM::InstanceProfile
    Properties:
      Path: "/"
      Roles:
      - !Ref: RootRole
```

AWS Lambda Template

The following template uses an AWS Lambda (Lambda) function and custom resource to append a new security group to a list of existing security groups. This function is useful when you want to build a list of security groups dynamically, so that your list includes both new and existing security groups. For example, you can pass a list of existing security groups as a parameter value, append the new value to the list, and then associate all your values with an EC2 instance. For more information about the Lambda function resource type, see AWS::Lambda::Function (p. 824).

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters" : {
    "ExistingSecurityGroups" : {
      "Type" : "List<AWS::EC2::SecurityGroup::Id>"
    },
    "ExistingVPC" : {
      "Type" : "AWS::EC2::VPC::Id",
      "Description" : "The VPC ID that includes the security groups in the
 ExistingSecurityGroups parameter."
    },
    "InstanceType" : {
      "Type" : "String",
      "Default" : "t2.micro",
      "AllowedValues" : ["t2.micro", "m1.small"]
    }
  },
  "Mappings": {
```

```
    "AWSInstanceType2Arch" : {
      "t2.micro"    : { "Arch" : "HVM64"  },
      "m1.small"    : { "Arch" : "PV64"   }
    },
    "AWSRegionArch2AMI" : {
      "us-east-1"        : {"PV64" : "ami-1ccae774", "HVM64" : "ami-1ecae776"},
      "us-west-2"        : {"PV64" : "ami-ff527ecf", "HVM64" : "ami-e7527ed7"},
      "us-west-1"        : {"PV64" : "ami-d514f291", "HVM64" : "ami-d114f295"},
      "eu-west-1"        : {"PV64" : "ami-bf0897c8", "HVM64" : "ami-a10897d6"},
      "eu-central-1"     : {"PV64" : "ami-ac221fb1", "HVM64" : "ami-a8221fb5"},
      "ap-northeast-1"   : {"PV64" : "ami-27f90e27", "HVM64" : "ami-cbf90ecb"},
      "ap-southeast-1"   : {"PV64" : "ami-acd9e8fe", "HVM64" : "ami-68d8e93a"},
      "ap-southeast-2"   : {"PV64" : "ami-ff9cecc5", "HVM64" : "ami-fd9cecc7"},
      "sa-east-1"        : {"PV64" : "ami-bb2890a6", "HVM64" : "ami-b52890a8"},
      "cn-north-1"       : {"PV64" : "ami-fa39abc3", "HVM64" : "ami-f239abcb"}
    }
  },
  "Resources" : {
    "SecurityGroup" : {
      "Type" : "AWS::EC2::SecurityGroup",
      "Properties" : {
        "GroupDescription" : "Allow HTTP traffic to the host",
        "VpcId" : {"Ref" : "ExistingVPC"},
        "SecurityGroupIngress" : [{
          "IpProtocol" : "tcp",
          "FromPort" : "80",
          "ToPort" : "80",
          "CidrIp" : "0.0.0.0/0"
        }],
        "SecurityGroupEgress" : [{
          "IpProtocol" : "tcp",
          "FromPort" : "80",
          "ToPort" : "80",
          "CidrIp" : "0.0.0.0/0"
        }]
      }
    },
    "AllSecurityGroups": {
      "Type": "Custom::Split",
      "Properties": {
        "ServiceToken": { "Fn::GetAtt" : ["AppendItemToListFunction", "Arn"] },
        "List": { "Ref" : "ExistingSecurityGroups" },
        "AppendedItem": { "Ref" : "SecurityGroup" }
      }
    },
    "AppendItemToListFunction": {
      "Type": "AWS::Lambda::Function",
      "Properties": {
        "Handler": "index.handler",
        "Role": { "Fn::GetAtt" : ["LambdaExecutionRole", "Arn"] },
        "Code": {
          "ZipFile":  { "Fn::Join": ["", [
            "var response = require('cfn-response');",
            "exports.handler = function(event, context) {",
            "   var responseData = {Value: event.ResourceProperties.List};",
            "   responseData.Value.push(event.ResourceProperties.AppendedItem);",
            "   response.send(event, context, response.SUCCESS, responseData);",
            "};"
          ]]}
        },
        "Runtime": "nodejs4.3"
      }
    },
    "MyEC2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
```

```
        "ImageId": { "Fn::FindInMap": [ "AWSRegionArch2AMI", { "Ref": "AWS::Region" },
 { "Fn::FindInMap": [
          "AWSInstanceType2Arch", { "Ref": "InstanceType" }, "Arch" ] } ]
        },
        "SecurityGroupIds" : { "Fn::GetAtt": [ "AllSecurityGroups", "Value" ] },
        "InstanceType" : { "Ref" : "InstanceType" }
      }
    },
    "LambdaExecutionRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [{ "Effect": "Allow", "Principal": {"Service":
 ["lambda.amazonaws.com"]}, "Action": ["sts:AssumeRole"] }]
        },
        "Path": "/",
        "Policies": [{
          "PolicyName": "root",
          "PolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [{ "Effect": "Allow", "Action": ["logs:*"], "Resource":
 "arn:aws:logs:*:*:*" }]
          }
        }]
      }
    }
  },
  "Outputs" : {
    "AllSecurityGroups" : {
      "Description" : "Security Groups that are associated with the EC2 instance",
      "Value" : { "Fn::Join" : [ ", ", { "Fn::GetAtt": [ "AllSecurityGroups", "Value" ] }]}
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Parameters:
  ExistingSecurityGroups:
    Type: List<AWS::EC2::SecurityGroup::Id>
  ExistingVPC:
    Type: AWS::EC2::VPC::Id
    Description: The VPC ID that includes the security groups in the ExistingSecurityGroups
      parameter.
  InstanceType:
    Type: String
    Default: t2.micro
    AllowedValues:
    - t2.micro
    - m1.small
Mappings:
  AWSInstanceType2Arch:
    t2.micro:
      Arch: HVM64
    m1.small:
      Arch: PV64
  AWSRegionArch2AMI:
    us-east-1:
      PV64: ami-1ccae774
      HVM64: ami-1ecae776
    us-west-2:
      PV64: ami-ff527ecf
```

```
      HVM64: ami-e7527ed7
    us-west-1:
      PV64: ami-d514f291
      HVM64: ami-d114f295
    eu-west-1:
      PV64: ami-bf0897c8
      HVM64: ami-a10897d6
    eu-central-1:
      PV64: ami-ac221fb1
      HVM64: ami-a8221fb5
    ap-northeast-1:
      PV64: ami-27f90e27
      HVM64: ami-cbf90ecb
    ap-southeast-1:
      PV64: ami-acd9e8fe
      HVM64: ami-68d8e93a
    ap-southeast-2:
      PV64: ami-ff9cecc5
      HVM64: ami-fd9cecc7
    sa-east-1:
      PV64: ami-bb2890a6
      HVM64: ami-b52890a8
    cn-north-1:
      PV64: ami-fa39abc3
      HVM64: ami-f239abcb
Resources:
  SecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Allow HTTP traffic to the host
      VpcId:
        Ref: ExistingVPC
      SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: '80'
        ToPort: '80'
        CidrIp: 0.0.0.0/0
      SecurityGroupEgress:
      - IpProtocol: tcp
        FromPort: '80'
        ToPort: '80'
        CidrIp: 0.0.0.0/0
  AllSecurityGroups:
    Type: Custom::Split
    Properties:
      ServiceToken: !GetAtt AppendItemToListFunction.Arn
      List:
        Ref: ExistingSecurityGroups
      AppendedItem:
        Ref: SecurityGroup
  AppendItemToListFunction:
    Type: AWS::Lambda::Function
    Properties:
      Handler: index.handler
      Role: !GetAtt LambdaExecutionRole.Arn
      Code:
        ZipFile: !Sub |
          var response = require('cfn-response');
          exports.handler = function(event, context) {
             var responseData = {Value: event.ResourceProperties.List};
             responseData.Value.push(event.ResourceProperties.AppendedItem);
             response.send(event, context, response.SUCCESS, responseData);
          };
      Runtime: nodejs4.3
  MyEC2Instance:
    Type: AWS::EC2::Instance
```

```
    Properties:
      ImageId:
        Fn::FindInMap:
        - AWSRegionArch2AMI
        - Ref: AWS::Region
        - Fn::FindInMap:
          - AWSInstanceType2Arch
          - Ref: InstanceType
          - Arch
      SecurityGroupIds: !GetAtt AllSecurityGroups.Value
      InstanceType:
        Ref: InstanceType
  LambdaExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
        - Effect: Allow
          Principal:
            Service:
            - lambda.amazonaws.com
          Action:
          - sts:AssumeRole
      Path: "/"
      Policies:
      - PolicyName: root
        PolicyDocument:
          Version: '2012-10-17'
          Statement:
          - Effect: Allow
            Action:
            - logs:*
            Resource: arn:aws:logs:*:*:*
Outputs:
  AllSecurityGroups:
    Description: Security Groups that are associated with the EC2 instance
    Value:
      Fn::Join:
      - ", "
      - Fn::GetAtt:
        - AllSecurityGroups
        - Value
```

In the example, when AWS CloudFormation creates the `AllSecurityGroups` custom resource, AWS CloudFormation invokes the `AppendItemToListFunction` Lambda function. AWS CloudFormation passes the list of existing security groups and a new security group (`NewSecurityGroup`) to the function, which appends the new security group to the list and then returns the modified list. AWS CloudFormation uses the modified list to associate all security groups with the `MyEC2Instance` resource.

AWS OpsWorks Template Snippets

AWS OpsWorks is an application management service that simplifies a wide range of tasks such as software configuration, application deployment, scaling, and monitoring. AWS CloudFormation is a resource management service that you can use to manage AWS OpsWorks resources, such as AWS OpsWorks stacks, layers, apps, and instances.

AWS OpsWorks Sample PHP App

The following sample template deploys a sample AWS OpsWorks PHP web application that is stored in public Git repository. The AWS OpsWorks stack includes two application servers with a load balancer that distributes incoming traffic evenly across the servers. The AWS OpsWorks stack also includes a back-end MySQL database server to store data. For more information about the sample AWS OpsWorks application, see Walkthrough: Learn AWS AWS OpsWorks Basics by Creating an Application Server Stack in the *AWS OpsWorks User Guide*.

Note

The `ServiceRoleArn` and `DefaultInstanceProfileArn` properties reference IAM roles that are created after you use AWS OpsWorks for the first time.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters": {
    "ServiceRole": {
      "Default": "aws-opsworks-service-role",
      "Description": "The OpsWorks service role",
      "Type": "String",
      "MinLength": "1",
      "MaxLength": "64",
      "AllowedPattern": "[a-zA-Z][a-zA-Z0-9-]*",
      "ConstraintDescription": "must begin with a letter and contain only alphanumeric
 characters."
    },
    "InstanceRole": {
      "Default": "aws-opsworks-ec2-role",
      "Description": "The OpsWorks instance role",
      "Type": "String",
      "MinLength": "1",
      "MaxLength": "64",
      "AllowedPattern": "[a-zA-Z][a-zA-Z0-9-]*",
      "ConstraintDescription": "must begin with a letter and contain only alphanumeric
 characters."
    },
    "AppName": {
      "Default": "myapp",
      "Description": "The app name",
      "Type": "String",
      "MinLength": "1",
      "MaxLength": "64",
      "AllowedPattern": "[a-zA-Z][a-zA-Z0-9]*",
      "ConstraintDescription": "must begin with a letter and contain only alphanumeric
 characters."
    },
    "MysqlRootPassword" : {
      "Description" : "MysqlRootPassword",
      "NoEcho" : "true",
      "Type" : "String"
    }
  },
  "Resources": {
    "myStack": {
      "Type": "AWS::OpsWorks::Stack",
      "Properties": {
        "Name": {
          "Ref": "AWS::StackName"
        },
        "ServiceRoleArn": {
          "Fn::Join": [
            "", ["arn:aws:iam::", {"Ref": "AWS::AccountId"},
                  ":role/", {"Ref": "ServiceRole"}]
          ]
        },
        "DefaultInstanceProfileArn": {
```

```
          "Fn::Join": [
            "", ["arn:aws:iam::", {"Ref": "AWS::AccountId"},
              ":instance-profile/", {"Ref": "InstanceRole"}]
          ]
        },
        "UseCustomCookbooks": "true",
        "CustomCookbooksSource": {
          "Type": "git",
          "Url": "git://github.com/amazonwebservices/opsworks-example-cookbooks.git"
        }
      }
    },
    "myLayer": {
      "Type": "AWS::OpsWorks::Layer",
      "DependsOn": "myApp",
      "Properties": {
        "StackId": {"Ref": "myStack"},
        "Type": "php-app",
     "Shortname" : "php-app",
        "EnableAutoHealing" : "true",
        "AutoAssignElasticIps" : "false",
        "AutoAssignPublicIps" : "true",
        "Name": "MyPHPApp",
        "CustomRecipes" : {
          "Configure" : ["phpapp::appsetup"]
        }
      }
    },
    "DBLayer" : {
      "Type" : "AWS::OpsWorks::Layer",
      "DependsOn": "myApp",
      "Properties" : {
        "StackId" : {"Ref":"myStack"},
        "Type" : "db-master",
     "Shortname" : "db-layer",
        "EnableAutoHealing" : "true",
        "AutoAssignElasticIps" : "false",
        "AutoAssignPublicIps" : "true",
        "Name" : "MyMySQL",
        "CustomRecipes" : {
          "Setup" : ["phpapp::dbsetup"]
        },
        "Attributes" : {
          "MysqlRootPassword" : {"Ref":"MysqlRootPassword"},
          "MysqlRootPasswordUbiquitous": "true"
        },
        "VolumeConfigurations":[{"MountPoint":"/vol/mysql","NumberOfDisks":1,"Size":10}]
      }
    },
    "ELBAttachment" : {
      "Type" : "AWS::OpsWorks::ElasticLoadBalancerAttachment",
      "Properties" : {
        "ElasticLoadBalancerName" : { "Ref" : "ELB" },
        "LayerId" : { "Ref" : "myLayer" }
      }
    },
    "ELB" : {
      "Type": "AWS::ElasticLoadBalancing::LoadBalancer",
      "Properties": {
        "AvailabilityZones": { "Fn::GetAZs" : "" } ,
        "Listeners": [{
          "LoadBalancerPort": "80",
          "InstancePort": "80",
          "Protocol": "HTTP",
          "InstanceProtocol": "HTTP"
        }],
```

```
        "HealthCheck": {
          "Target": "HTTP:80/",
          "HealthyThreshold": "2",
          "UnhealthyThreshold": "10",
          "Interval": "30",
          "Timeout": "5"
        }
      }
    },
    "myAppInstance1": {
      "Type": "AWS::OpsWorks::Instance",
      "Properties": {
        "StackId": {"Ref": "myStack"},
        "LayerIds": [{"Ref": "myLayer"}],
        "InstanceType": "m1.small"
      }
    },
    "myAppInstance2": {
      "Type": "AWS::OpsWorks::Instance",
      "Properties": {
        "StackId": {"Ref": "myStack"},
        "LayerIds": [{"Ref": "myLayer"}],
        "InstanceType": "m1.small"
      }
    },
    "myDBInstance": {
      "Type": "AWS::OpsWorks::Instance",
      "Properties": {
        "StackId": {"Ref": "myStack"},
        "LayerIds": [{"Ref": "DBLayer"}],
        "InstanceType": "m1.small"
      }
    },
    "myApp" : {
      "Type" : "AWS::OpsWorks::App",
      "Properties" : {
        "StackId" : {"Ref":"myStack"},
        "Type" : "php",
        "Name" : {"Ref": "AppName"},
        "AppSource" : {
          "Type" : "git",
          "Url" : "git://github.com/amazonwebservices/opsworks-demo-php-simple-app.git",
          "Revision" : "version2"
        },
        "Attributes" : {
          "DocumentRoot" : "web"
        }
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Parameters:
  ServiceRole:
    Default: aws-opsworks-service-role
    Description: The OpsWorks service role
    Type: String
    MinLength: '1'
    MaxLength: '64'
    AllowedPattern: "[a-zA-Z][a-zA-Z0-9-]*"
    ConstraintDescription: must begin with a letter and contain only alphanumeric
```

```
      characters.
  InstanceRole:
    Default: aws-opsworks-ec2-role
    Description: The OpsWorks instance role
    Type: String
    MinLength: '1'
    MaxLength: '64'
    AllowedPattern: "[a-zA-Z][a-zA-Z0-9-]*"
    ConstraintDescription: must begin with a letter and contain only alphanumeric
      characters.
  AppName:
    Default: myapp
    Description: The app name
    Type: String
    MinLength: '1'
    MaxLength: '64'
    AllowedPattern: "[a-zA-Z][a-zA-Z0-9]*"
    ConstraintDescription: must begin with a letter and contain only alphanumeric
      characters.
  MysqlRootPassword:
    Description: MysqlRootPassword
    NoEcho: 'true'
    Type: String
Resources:
  myStack:
    Type: AWS::OpsWorks::Stack
    Properties:
      Name:
        Ref: AWS::StackName
      ServiceRoleArn: !Sub "arn:aws:iam::${AWS::AccountId}:role/${ServiceRole}"
      DefaultInstanceProfileArn: !Sub "arn:aws:iam::${AWS::AccountId}:instance-profile/
${InstanceRole}"
      UseCustomCookbooks: 'true'
      CustomCookbooksSource:
        Type: git
        Url: git://github.com/amazonwebservices/opsworks-example-cookbooks.git
  myLayer:
    Type: AWS::OpsWorks::Layer
    DependsOn: myApp
    Properties:
      StackId:
        Ref: myStack
      Type: php-app
      Shortname: php-app
      EnableAutoHealing: 'true'
      AutoAssignElasticIps: 'false'
      AutoAssignPublicIps: 'true'
      Name: MyPHPApp
      CustomRecipes:
        Configure:
        - phpapp::appsetup
  DBLayer:
    Type: AWS::OpsWorks::Layer
    DependsOn: myApp
    Properties:
      StackId:
        Ref: myStack
      Type: db-master
      Shortname: db-layer
      EnableAutoHealing: 'true'
      AutoAssignElasticIps: 'false'
      AutoAssignPublicIps: 'true'
      Name: MyMySQL
      CustomRecipes:
        Setup:
        - phpapp::dbsetup
```

```
    Attributes:
      MysqlRootPassword:
        Ref: MysqlRootPassword
      MysqlRootPasswordUbiquitous: 'true'
    VolumeConfigurations:
    - MountPoint: "/vol/mysql"
      NumberOfDisks: 1
      Size: 10
ELBAttachment:
  Type: AWS::OpsWorks::ElasticLoadBalancerAttachment
  Properties:
    ElasticLoadBalancerName:
      Ref: ELB
    LayerId:
      Ref: myLayer
ELB:
  Type: AWS::ElasticLoadBalancing::LoadBalancer
  Properties:
    AvailabilityZones:
      Fn::GetAZs: ''
    Listeners:
    - LoadBalancerPort: '80'
      InstancePort: '80'
      Protocol: HTTP
      InstanceProtocol: HTTP
    HealthCheck:
      Target: HTTP:80/
      HealthyThreshold: '2'
      UnhealthyThreshold: '10'
      Interval: '30'
      Timeout: '5'
myAppInstance1:
  Type: AWS::OpsWorks::Instance
  Properties:
    StackId:
      Ref: myStack
    LayerIds:
    - Ref: myLayer
    InstanceType: m1.small
myAppInstance2:
  Type: AWS::OpsWorks::Instance
  Properties:
    StackId:
      Ref: myStack
    LayerIds:
    - Ref: myLayer
    InstanceType: m1.small
myDBInstance:
  Type: AWS::OpsWorks::Instance
  Properties:
    StackId:
      Ref: myStack
    LayerIds:
    - Ref: DBLayer
    InstanceType: m1.small
myApp:
  Type: AWS::OpsWorks::App
  Properties:
    StackId:
      Ref: myStack
    Type: php
    Name:
      Ref: AppName
    AppSource:
      Type: git
      Url: git://github.com/amazonwebservices/opsworks-demo-php-simple-app.git
```

```
      Revision: version2
    Attributes:
      DocumentRoot: web
```

Amazon Redshift Template Snippets

Amazon Redshift is a fully managed, petabyte-scale data warehouse service in the cloud. You can use AWS CloudFormation to provision and manage Amazon Redshift clusters.

Amazon Redshift Cluster

The following sample template creates an Amazon Redshift cluster according to the parameter values that are specified when the stack is created. The cluster parameter group that is associated with the Amazon Redshift cluster enables user activity logging. The template also launches the Amazon Redshift clusters in an Amazon VPC that is defined in the template. The VPC includes an internet gateway so that you can access the Amazon Redshift clusters from the Internet. However, the communication between the cluster and the Internet gateway must also be enabled, which is done by the route table entry.

Note

The template includes the `IsMultiNodeCluster` condition so that the `NumberOfNodes` parameter is declared only when the `ClusterType` parameter value is set to `multi-node`.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters" : {
    "DatabaseName" : {
      "Description" : "The name of the first database to be created when the cluster is
 created",
      "Type" : "String",
      "Default" : "dev",
      "AllowedPattern" : "([a-z]|[0-9])+"
    },
    "ClusterType" : {
      "Description" : "The type of cluster",
      "Type" : "String",
      "Default" : "single-node",
      "AllowedValues" : [ "single-node", "multi-node" ]
    },
    "NumberOfNodes" : {
      "Description" : "The number of compute nodes in the cluster. For multi-node clusters,
 the NumberOfNodes parameter must be greater than 1",
      "Type" : "Number",
      "Default" : "1"
    },
    "NodeType" : {
      "Description" : "The type of node to be provisioned",
      "Type" : "String",
      "Default" : "dw1.xlarge",
      "AllowedValues" : [ "dw1.xlarge", "dw1.8xlarge", "dw2.large", "dw2.8xlarge" ]
    },
    "MasterUsername" : {
      "Description" : "The user name that is associated with the master user account for
 the cluster that is being created",
      "Type" : "String",
      "Default" : "defaultuser",
      "AllowedPattern" : "([a-z])([a-z]|[0-9])*"
    },
    "MasterUserPassword" :  {
      "Description" : "The password that is associated with the master user account for the
 cluster that is being created.",
```

```
      "Type" : "String",
      "NoEcho" : "true"
    },
    "InboundTraffic" : {
      "Description" : "Allow inbound traffic to the cluster from this CIDR range.",
      "Type" : "String",
      "MinLength": "9",
      "MaxLength": "18",
      "Default" : "0.0.0.0/0",
      "AllowedPattern" : "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\\d{1,2})",
      "ConstraintDescription" : "must be a valid CIDR range of the form x.x.x.x/x."
    },
    "PortNumber" : {
      "Description" : "The port number on which the cluster accepts incoming connections.",
      "Type" : "Number",
      "Default" : "5439"
    }
  },
  "Conditions" : {
    "IsMultiNodeCluster" : {
      "Fn::Equals" : [{ "Ref" : "ClusterType" }, "multi-node" ]
    }
  },
  "Resources" : {
    "RedshiftCluster" : {
      "Type" : "AWS::Redshift::Cluster",
      "DependsOn" : "AttachGateway",
      "Properties" : {
        "ClusterType" : { "Ref" : "ClusterType" },
        "NumberOfNodes" : { "Fn::If" : [ "IsMultiNodeCluster",  { "Ref" :
"NumberOfNodes" }, { "Ref" : "AWS::NoValue" }]},
        "NodeType" : { "Ref" : "NodeType" },
        "DBName" : { "Ref" : "DatabaseName" },
        "MasterUsername" : { "Ref" : "MasterUsername" },
        "MasterUserPassword" : { "Ref" : "MasterUserPassword" },
        "ClusterParameterGroupName" : { "Ref" : "RedshiftClusterParameterGroup" },
        "VpcSecurityGroupIds" : [ { "Ref" : "SecurityGroup" } ],
        "ClusterSubnetGroupName" : { "Ref" : "RedshiftClusterSubnetGroup" },
        "PubliclyAccessible" : "true",
        "Port" : { "Ref" : "PortNumber" }
      }
    },
    "RedshiftClusterParameterGroup" : {
      "Type" : "AWS::Redshift::ClusterParameterGroup",
      "Properties" : {
        "Description" : "Cluster parameter group",
        "ParameterGroupFamily" : "redshift-1.0",
        "Parameters" : [{
          "ParameterName" : "enable_user_activity_logging",
          "ParameterValue" : "true"
        }]
      }
    },
    "RedshiftClusterSubnetGroup" : {
      "Type" : "AWS::Redshift::ClusterSubnetGroup",
      "Properties" : {
        "Description" : "Cluster subnet group",
        "SubnetIds" : [ { "Ref" : "PublicSubnet" } ]
      }
    },
    "VPC" : {
      "Type" : "AWS::EC2::VPC",
      "Properties" : {
        "CidrBlock" : "10.0.0.0/16"
      }
    },
```

```
    "PublicSubnet" : {
      "Type" : "AWS::EC2::Subnet",
      "Properties" : {
        "CidrBlock" : "10.0.0.0/24",
        "VpcId" : { "Ref" : "VPC" }
      }
    },
    "SecurityGroup" : {
      "Type" : "AWS::EC2::SecurityGroup",
      "Properties" : {
        "GroupDescription" : "Security group",
        "SecurityGroupIngress" : [ {
          "CidrIp" : { "Ref": "InboundTraffic" },
          "FromPort" : { "Ref" : "PortNumber" },
          "ToPort" : { "Ref" : "PortNumber" },
          "IpProtocol" : "tcp"
        } ],
        "VpcId" : { "Ref" : "VPC" }
      }
    },
    "myInternetGateway" : {
      "Type" : "AWS::EC2::InternetGateway"
    },
    "AttachGateway" : {
      "Type" : "AWS::EC2::VPCGatewayAttachment",
      "Properties" : {
        "VpcId" : { "Ref" : "VPC" },
        "InternetGatewayId" : { "Ref" : "myInternetGateway" }
      }
    },
    "PublicRouteTable" : {
      "Type" : "AWS::EC2::RouteTable",
      "Properties" : {
        "VpcId" : {
          "Ref" : "VPC"
        }
      }
    },
    "PublicRoute" : {
      "Type" : "AWS::EC2::Route",
      "DependsOn" : "AttachGateway",
      "Properties"  : {
        "RouteTableId" : {
          "Ref" : "PublicRouteTable"
        },
        "DestinationCidrBlock" : "0.0.0.0/0",
        "GatewayId" : {
          "Ref" : "myInternetGateway"
        }
      }
    },
    "PublicSubnetRouteTableAssociation" : {
      "Type" : "AWS::EC2::SubnetRouteTableAssociation",
      "Properties" : {
        "SubnetId" : {
          "Ref" : "PublicSubnet"
        },
        "RouteTableId" : {
          "Ref" : "PublicRouteTable"
        }
      }
    }
  },
  "Outputs" : {
    "ClusterEndpoint" : {
      "Description" : "Cluster endpoint",
```

```
      "Value" : { "Fn::Join" : [ ":", [ { "Fn::GetAtt" : [ "RedshiftCluster",
 "Endpoint.Address" ] }, { "Fn::GetAtt" : [ "RedshiftCluster", "Endpoint.Port" ] } ] ] }
    },
    "ClusterName" : {
      "Description" : "Name of cluster",
      "Value" : { "Ref" : "RedshiftCluster" }
    },
    "ParameterGroupName" : {
      "Description" : "Name of parameter group",
      "Value" : { "Ref" : "RedshiftClusterParameterGroup" }
    },
    "RedshiftClusterSubnetGroupName" : {
      "Description" : "Name of cluster subnet group",
      "Value" : { "Ref" : "RedshiftClusterSubnetGroup" }
    },
    "RedshiftClusterSecurityGroupName" : {
      "Description" : "Name of cluster security group",
      "Value" : { "Ref" : "SecurityGroup" }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Parameters:
  DatabaseName:
    Description: The name of the first database to be created when the cluster is
      created
    Type: String
    Default: dev
    AllowedPattern: "([a-z]|[0-9])+"
  ClusterType:
    Description: The type of cluster
    Type: String
    Default: single-node
    AllowedValues:
    - single-node
    - multi-node
  NumberOfNodes:
    Description: The number of compute nodes in the cluster. For multi-node clusters,
      the NumberOfNodes parameter must be greater than 1
    Type: Number
    Default: '1'
  NodeType:
    Description: The type of node to be provisioned
    Type: String
    Default: dw1.xlarge
    AllowedValues:
    - dw1.xlarge
    - dw1.8xlarge
    - dw2.large
    - dw2.8xlarge
  MasterUsername:
    Description: The user name that is associated with the master user account for
      the cluster that is being created
    Type: String
    Default: defaultuser
    AllowedPattern: "([a-z])([a-z]|[0-9])*"
  MasterUserPassword:
    Description: The password that is associated with the master user account for
      the cluster that is being created.
    Type: String
    NoEcho: 'true'
```

```
  InboundTraffic:
    Description: Allow inbound traffic to the cluster from this CIDR range.
    Type: String
    MinLength: '9'
    MaxLength: '18'
    Default: 0.0.0.0/0
    AllowedPattern: "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\\d{1,2})"
    ConstraintDescription: must be a valid CIDR range of the form x.x.x.x/x.
  PortNumber:
    Description: The port number on which the cluster accepts incoming connections.
    Type: Number
    Default: '5439'
Conditions:
  IsMultiNodeCluster:
    Fn::Equals:
    - Ref: ClusterType
    - multi-node
Resources:
  RedshiftCluster:
    Type: AWS::Redshift::Cluster
    DependsOn: AttachGateway
    Properties:
      ClusterType:
        Ref: ClusterType
      NumberOfNodes:
        Fn::If:
        - IsMultiNodeCluster
        - Ref: NumberOfNodes
        - Ref: AWS::NoValue
      NodeType:
        Ref: NodeType
      DBName:
        Ref: DatabaseName
      MasterUsername:
        Ref: MasterUsername
      MasterUserPassword:
        Ref: MasterUserPassword
      ClusterParameterGroupName:
        Ref: RedshiftClusterParameterGroup
      VpcSecurityGroupIds:
      - Ref: SecurityGroup
      ClusterSubnetGroupName:
        Ref: RedshiftClusterSubnetGroup
      PubliclyAccessible: 'true'
      Port:
        Ref: PortNumber
  RedshiftClusterParameterGroup:
    Type: AWS::Redshift::ClusterParameterGroup
    Properties:
      Description: Cluster parameter group
      ParameterGroupFamily: redshift-1.0
      Parameters:
      - ParameterName: enable_user_activity_logging
        ParameterValue: 'true'
  RedshiftClusterSubnetGroup:
    Type: AWS::Redshift::ClusterSubnetGroup
    Properties:
      Description: Cluster subnet group
      SubnetIds:
      - Ref: PublicSubnet
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: 10.0.0.0/16
  PublicSubnet:
    Type: AWS::EC2::Subnet
```

```
    Properties:
      CidrBlock: 10.0.0.0/24
      VpcId:
        Ref: VPC
  SecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Security group
      SecurityGroupIngress:
      - CidrIp:
          Ref: InboundTraffic
        FromPort:
          Ref: PortNumber
        ToPort:
          Ref: PortNumber
        IpProtocol: tcp
      VpcId:
        Ref: VPC
  myInternetGateway:
    Type: AWS::EC2::InternetGateway
  AttachGateway:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      VpcId:
        Ref: VPC
      InternetGatewayId:
        Ref: myInternetGateway
  PublicRouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId:
        Ref: VPC
  PublicRoute:
    Type: AWS::EC2::Route
    DependsOn: AttachGateway
    Properties:
      RouteTableId:
        Ref: PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId:
        Ref: myInternetGateway
  PublicSubnetRouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      SubnetId:
        Ref: PublicSubnet
      RouteTableId:
        Ref: PublicRouteTable
Outputs:
  ClusterEndpoint:
    Description: Cluster endpoint
    Value: !Sub "${RedshiftCluster.Endpoint.Address}:${RedshiftCluster.Endpoint.Port}"
  ClusterName:
    Description: Name of cluster
    Value:
      Ref: RedshiftCluster
  ParameterGroupName:
    Description: Name of parameter group
    Value:
      Ref: RedshiftClusterParameterGroup
  RedshiftClusterSubnetGroupName:
    Description: Name of cluster subnet group
    Value:
      Ref: RedshiftClusterSubnetGroup
  RedshiftClusterSecurityGroupName:
    Description: Name of cluster security group
```

```
    Value:
      Ref: SecurityGroup
```

See Also

AWS::Redshift::Cluster (p. 911)

Amazon RDS Template Snippets

Topics

Amazon RDS DB Instance Resource

This example shows an Amazon RDS DB Instance resource. Because the optional EngineVersion property is not specified, the default engine version is used for this DB Instance. For details about the default engine version and other default settings, see CreateDBInstance. The DBSecurityGroups property authorizes network ingress to the AWS::RDS::DBSecurityGroup resources named MyDbSecurityByEC2SecurityGroup and MyDbSecurityByCIDRIPGroup. For details, see AWS::RDS::DBInstance (p. 881). The DB Instance resource also has a DeletionPolicy attribute set to Snapshot. With the Snapshot DeletionPolicy set, AWS CloudFormation will take a snapshot of this DB Instance before deleting it during stack deletion.

JSON

```
"MyDB" : {
 "Type" : "AWS::RDS::DBInstance",
 "Properties" : {
     "DBSecurityGroups" : [
        {"Ref" : "MyDbSecurityByEC2SecurityGroup"}, {"Ref" :
 "MyDbSecurityByCIDRIPGroup"} ],
     "AllocatedStorage" : "5",
     "DBInstanceClass" : "db.m1.small",
     "Engine" : "MySQL",
     "MasterUsername" : "MyName",
     "MasterUserPassword" : "MyPassword"
 },
 "DeletionPolicy" : "Snapshot"
}
```

YAML

```
MyDB:
  Type: AWS::RDS::DBInstance
  Properties:
    DBSecurityGroups:
    - Ref: MyDbSecurityByEC2SecurityGroup
    - Ref: MyDbSecurityByCIDRIPGroup
    AllocatedStorage: '5'
    DBInstanceClass: db.m1.small
    Engine: MySQL
```

```
    MasterUsername: MyName
    MasterUserPassword: MyPassword
  DeletionPolicy: Snapshot
```

Amazon RDS Oracle Database DB Instance Resource

This example creates an Oracle Database DB Instance resource by specifying the Engine as oracle-ee with a license model of bring-your-own-license. For details about the settings for Oracle Database DB instances, see CreateDBInstance. The DBSecurityGroups property authorizes network ingress to the AWS::RDS::DBSecurityGroup resources named MyDbSecurityByEC2SecurityGroup and MyDbSecurityByCIDRIPGroup. For details, see AWS::RDS::DBInstance (p. 881). The DB Instance resource also has a DeletionPolicy attribute set to Snapshot. With the Snapshot DeletionPolicy set, AWS CloudFormation will take a snapshot of this DB Instance before deleting it during stack deletion.

JSON

```
"MyDB" : {
 "Type" : "AWS::RDS::DBInstance",
 "Properties" : {
     "DBSecurityGroups" : [
        {"Ref" : "MyDbSecurityByEC2SecurityGroup"}, {"Ref" :
 "MyDbSecurityByCIDRIPGroup"} ],
     "AllocatedStorage" : "5",
     "DBInstanceClass" : "db.m1.small",
     "Engine" : "oracle-ee",
     "LicenseModel" : "bring-your-own-license",
     "MasterUsername" : "master",
     "MasterUserPassword" : "SecretPassword01"
 },
 "DeletionPolicy" : "Snapshot"
}
```

YAML

```
MyDB:
  Type: AWS::RDS::DBInstance
  Properties:
    DBSecurityGroups:
    - Ref: MyDbSecurityByEC2SecurityGroup
    - Ref: MyDbSecurityByCIDRIPGroup
    AllocatedStorage: '5'
    DBInstanceClass: db.m1.small
    Engine: oracle-ee
    LicenseModel: bring-your-own-license
    MasterUsername: master
    MasterUserPassword: SecretPassword01
  DeletionPolicy: Snapshot
```

Amazon RDS DBSecurityGroup Resource for CIDR Range

This example shows an Amazon RDS DBSecurityGroup resource with ingress authorization for the specified CIDR range in the format ddd.ddd.ddd.ddd/dd. For details, see AWS::RDS::DBSecurityGroup (p. 898) and Amazon RDS Security Group Rule (p. 1239).

JSON

```
"MyDbSecurityByCIDRIPGroup" : {
 "Type" : "AWS::RDS::DBSecurityGroup",
```

```
 "Properties" : {
     "GroupDescription" : "Ingress for CIDRIP",
     "DBSecurityGroupIngress" : {
         "CIDRIP" : "192.168.0.0/32"
     }
 }
}
```

YAML

```
MyDbSecurityByCIDRIPGroup:
  Type: AWS::RDS::DBSecurityGroup
  Properties:
    GroupDescription: Ingress for CIDRIP
    DBSecurityGroupIngress:
      CIDRIP: "192.168.0.0/32"
```

Amazon RDS DBSecurityGroup with an Amazon EC2 security group

This example shows an AWS::RDS::DBSecurityGroup (p. 898) resource with ingress authorization from an Amazon EC2 security group referenced by MyEc2SecurityGroup.

To do this, you define an EC2 security group and then use the intrinsic Ref function to refer to the EC2 security group within your DBSecurityGroup.

JSON

```

"DBInstance" : {
   "Type": "AWS::RDS::DBInstance",
   "Properties": {
      "DBName"            : { "Ref" : "DBName" },
      "Engine"            : "MySQL",
      "MasterUsername"    : { "Ref" : "DBUsername" },
      "DBInstanceClass"   : { "Ref" : "DBClass" },
      "DBSecurityGroups"  : [ { "Ref" : "DBSecurityGroup" } ],
      "AllocatedStorage"  : { "Ref" : "DBAllocatedStorage" },
      "MasterUserPassword": { "Ref" : "DBPassword" }
   }
},

"DBSecurityGroup": {
   "Type": "AWS::RDS::DBSecurityGroup",
   "Properties": {
      "DBSecurityGroupIngress": { "EC2SecurityGroupName": { "Ref":
 "WebServerSecurityGroup" } },
      "GroupDescription"      : "Frontend Access"
   }
},

"WebServerSecurityGroup" : {
   "Type" : "AWS::EC2::SecurityGroup",
   "Properties" : {
      "GroupDescription" : "Enable HTTP access via port 80 and SSH access",
      "SecurityGroupIngress" : [
         {"IpProtocol" : "tcp", "FromPort" : "80", "ToPort" : "80", "CidrIp" :
 "0.0.0.0/0"},
         {"IpProtocol" : "tcp", "FromPort" : "22", "ToPort" : "22", "CidrIp" : "0.0.0.0/0"}
      ]
```

```
    }
}
```

YAML

This example is extracted from the following full example: Drupal_Single_Instance_With_RDS.template

```
DBInstance:
  Type: AWS::RDS::DBInstance
  Properties:
    DBName:
      Ref: DBName
    Engine: MySQL
    MasterUsername:
      Ref: DBUsername
    DBInstanceClass:
      Ref: DBClass
    DBSecurityGroups:
    - Ref: DBSecurityGroup
    AllocatedStorage:
      Ref: DBAllocatedStorage
    MasterUserPassword:
      Ref: DBPassword
DBSecurityGroup:
  Type: AWS::RDS::DBSecurityGroup
  Properties:
    DBSecurityGroupIngress:
      EC2SecurityGroupName:
        Ref: WebServerSecurityGroup
    GroupDescription: Frontend Access
WebServerSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Enable HTTP access via port 80 and SSH access
    SecurityGroupIngress:
    - IpProtocol: tcp
      FromPort: '80'
      ToPort: '80'
      CidrIp: 0.0.0.0/0
    - IpProtocol: tcp
      FromPort: '22'
      ToPort: '22'
      CidrIp: 0.0.0.0/0
```

Multiple VPC security groups

This example shows an AWS::RDS::DBSecurityGroup (p. 898) resource with ingress authorization for multiple Amazon EC2 VPC security groups in AWS::RDS::DBSecurityGroupIngress (p. 901).

JSON

```
{
   "Resources" : {
      "DBinstance" : {
         "Type" : "AWS::RDS::DBInstance",
         "Properties" : {
            "AllocatedStorage" : "5",
            "DBInstanceClass" : "db.m1.small",
           "DBName" : {"Ref": "MyDBName" },
            "DBSecurityGroups" : [ { "Ref" : "DbSecurityByEC2SecurityGroup" } ],
            "DBSubnetGroupName" : { "Ref" : "MyDBSubnetGroup" },
            "Engine" : "MySQL",
```

```
            "MasterUserPassword": { "Ref" : "MyDBPassword" },
            "MasterUsername"    : { "Ref" : "MyDBUsername" }
         },
         "DeletionPolicy" : "Snapshot"
      },
      "DbSecurityByEC2SecurityGroup" : {
         "Type" : "AWS::RDS::DBSecurityGroup",
         "Properties" : {
            "GroupDescription" : "Ingress for Amazon EC2 security group",
           "EC2VpcId" : { "Ref" : "MyVPC" },
            "DBSecurityGroupIngress" : [ {
               "EC2SecurityGroupId" : "sg-b0ff1111",
               "EC2SecurityGroupOwnerId" : "111122223333"
            }, {
               "EC2SecurityGroupId" : "sg-ffd722222",
               "EC2SecurityGroupOwnerId" : "111122223333"
            } ]
         }
      }
   }
}
```

YAML

```
Resources:
  DBinstance:
    Type: AWS::RDS::DBInstance
    Properties:
      AllocatedStorage: '5'
      DBInstanceClass: db.m1.small
      DBName:
        Ref: MyDBName
      DBSecurityGroups:
      - Ref: DbSecurityByEC2SecurityGroup
      DBSubnetGroupName:
        Ref: MyDBSubnetGroup
      Engine: MySQL
      MasterUserPassword:
        Ref: MyDBPassword
      MasterUsername:
        Ref: MyDBUsername
    DeletionPolicy: Snapshot
  DbSecurityByEC2SecurityGroup:
    Type: AWS::RDS::DBSecurityGroup
    Properties:
      GroupDescription: Ingress for Amazon EC2 security group
      EC2VpcId:
        Ref: MyVPC
      DBSecurityGroupIngress:
      - EC2SecurityGroupId: sg-b0ff1111
        EC2SecurityGroupOwnerId: '111122223333'
      - EC2SecurityGroupId: sg-ffd722222
        EC2SecurityGroupOwnerId: '111122223333'
```

Amazon RDS Database Instance in a VPC Security Group

This example shows an Amazon RDS database instance associated with an Amazon EC2 VPC security group.

JSON

```
{
  "DBEC2SecurityGroup": {
    "Type": "AWS::EC2::SecurityGroup",
    "Properties" : {
      "GroupDescription": "Open database for access",
      "SecurityGroupIngress" : [{
        "IpProtocol" : "tcp",
        "FromPort" : "3306",
        "ToPort" : "3306",
        "SourceSecurityGroupName" : { "Ref" : "WebServerSecurityGroup" }
      }]
    }
  },
  "DBInstance" : {
    "Type": "AWS::RDS::DBInstance",
    "Properties": {
      "DBName"            : { "Ref" : "DBName" },
      "Engine"            : "MySQL",
      "MultiAZ"           : { "Ref": "MultiAZDatabase" },
      "MasterUsername"    : { "Ref" : "DBUser" },
      "DBInstanceClass"   : { "Ref" : "DBClass" },
      "AllocatedStorage"  : { "Ref" : "DBAllocatedStorage" },
      "MasterUserPassword": { "Ref" : "DBPassword" },
      "VPCSecurityGroups" : [ { "Fn::GetAtt": [ "DBEC2SecurityGroup", "GroupId" ] } ]
    }
  }
}
```

YAML

```
DBEC2SecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Open database for access
    SecurityGroupIngress:
    - IpProtocol: tcp
      FromPort: '3306'
      ToPort: '3306'
      SourceSecurityGroupName:
        Ref: WebServerSecurityGroup
DBInstance:
  Type: AWS::RDS::DBInstance
  Properties:
    DBName:
      Ref: DBName
    Engine: MySQL
    MultiAZ:
      Ref: MultiAZDatabase
    MasterUsername:
      Ref: DBUser
    DBInstanceClass:
      Ref: DBClass
    AllocatedStorage:
      Ref: DBAllocatedStorage
    MasterUserPassword:
      Ref: DBPassword
    VPCSecurityGroups:
    - !GetAtt DBEC2SecurityGroup.GroupId
```

Amazon Route 53 Template Snippets

Topics

Amazon Route 53 Resource Record Set Using Hosted Zone Name or ID

When you create an Amazon Route 53 resource record set, you must specify the hosted zone where you want to add it. AWS CloudFormation provides two ways to do this. You can explicitly specify the hosted zone using the HostedZoneId property or have AWS CloudFormation find the hosted zone using the HostedZoneName property. If you use the HostedZoneName property and there are multiple hosted zones with the same domain name, AWS CloudFormation doesn't create the stack.

Adding RecordSet using HostedZoneId

This example adds an Amazon Route 53 resource record set containing an SPF record for the domain name `mysite.example.com` that uses the `HostedZoneId` property to specify the hosted zone.

JSON

```
"myDNSRecord" : {
  "Type" : "AWS::Route53::RecordSet",
  "Properties" :
  {
    "HostedZoneId" : "Z3DG6IL3SJCGPX",
    "Name" : "mysite.example.com.",
    "Type" : "SPF",
    "TTL" : "900",
    "ResourceRecords" : [ "\"v=spf1 ip4:192.168.0.1/16 -all\"" ]
  }
}
```

YAML

```
myDNSRecord:
  Type: AWS::Route53::RecordSet
  Properties:
    HostedZoneId: Z3DG6IL3SJCGPX
    Name: mysite.example.com.
    Type: SPF
    TTL: '900'
    ResourceRecords:
    - '"v=spf1 ip4:192.168.0.1/16 -all"'
```

Adding RecordSet using HostedZoneName

This example adds an Amazon Route 53 resource record set containing A records for the domain name "mysite.example.com" using the HostedZoneName property to specify the hosted zone.

JSON

```
"myDNSRecord2" : {
            "Type" : "AWS::Route53::RecordSet",
            "Properties" : {
```

```
                "HostedZoneName" : "example.com.",
                "Name" : "mysite.example.com.",
                "Type" : "A",
                "TTL" : "900",
                "ResourceRecords" : [
                    "192.168.0.1",
                    "192.168.0.2"
                ]
            }
        }
```

YAML

```
myDNSRecord2:
  Type: AWS::Route53::RecordSet
  Properties:
    HostedZoneName: example.com.
    Name: mysite.example.com.
    Type: A
    TTL: '900'
    ResourceRecords:
    - 192.168.0.1
    - 192.168.0.2
```

Using RecordSetGroup to Set Up Weighted Resource Record Sets

This example uses an AWS::Route53::RecordSetGroup (p. 935) to set up two CNAME records for the "example.com." hosted zone. The `RecordSets` property contains the CNAME record sets for the "mysite.example.com" DNS name. Each record set contains an identifier (SetIdentifier) and weight (Weight). The weighting for Frontend One is 40% (4 of 10) and Frontend Two is 60% (6 of 10). For more information about weighted resource record sets, see Setting Up Weighted Resource Record Sets in Amazon Route 53 Developer Guide.

JSON

```
        "myDNSOne" : {
            "Type" : "AWS::Route53::RecordSetGroup",
            "Properties" : {
                "HostedZoneName" : "example.com.",
                "Comment" : "Weighted RR for my frontends.",
                "RecordSets" : [
                  {
                    "Name" : "mysite.example.com.",
                    "Type" : "CNAME",
                    "TTL" : "900",
                    "SetIdentifier" : "Frontend One",
                    "Weight" : "4",
                    "ResourceRecords" : ["example-ec2.amazonaws.com"]
                  },
                  {
                    "Name" : "mysite.example.com.",
                    "Type" : "CNAME",
                    "TTL" : "900",
                    "SetIdentifier" : "Frontend Two",
                    "Weight" : "6",
                    "ResourceRecords" : ["example-ec2-larger.amazonaws.com"]
                  }
                  ]
            }
```

```
        }
```

YAML

```
myDNSOne:
  Type: AWS::Route53::RecordSetGroup
  Properties:
    HostedZoneName: example.com.
    Comment: Weighted RR for my frontends.
    RecordSets:
    - Name: mysite.example.com.
      Type: CNAME
      TTL: '900'
      SetIdentifier: Frontend One
      Weight: '4'
      ResourceRecords:
      - example-ec2.amazonaws.com
    - Name: mysite.example.com.
      Type: CNAME
      TTL: '900'
      SetIdentifier: Frontend Two
      Weight: '6'
      ResourceRecords:
      - example-ec2-larger.amazonaws.com
```

Using RecordSetGroup to Set Up an Alias Resource Record Set

This example uses an AWS::Route53::RecordSetGroup (p. 935) to set up an alias resource record set for the "example.com." hosted zone. The `RecordSets` property contains the A record for the zone apex "example.com." The AliasTarget (p. 1240) property specifies the hosted zone ID and DNS name for the myELB LoadBalancer by using the GetAtt (p. 1324) intrinsic function to retrieve the CanonicalHostedZoneNameID and DNSName properties of myELB resource. For more information about alias resource record sets, see Creating Alias Resource Record Sets in the *Amazon Route 53 Developer Guide*.

JSON

```
      "myELB" : {
        "Type" : "AWS::ElasticLoadBalancing::LoadBalancer",
        "Properties" : {
            "AvailabilityZones" : [ "us-east-1a" ],
            "Listeners" : [ {
                "LoadBalancerPort" : "80",
                "InstancePort" : "80",
                "Protocol" : "HTTP"
            } ]
        }
      },
      "myDNS" : {
        "Type" : "AWS::Route53::RecordSetGroup",
        "Properties" : {
          "HostedZoneName" : "example.com.",
          "Comment" : "Zone apex alias targeted to myELB LoadBalancer.",
          "RecordSets" : [
            {
              "Name" : "example.com.",
              "Type" : "A",
              "AliasTarget" : {
                  "HostedZoneId" : { "Fn::GetAtt" : ["myELB",
 "CanonicalHostedZoneNameID"] },
                  "DNSName" : { "Fn::GetAtt" : ["myELB","DNSName"] }
```

```
            }
          }
        ]
      }
  }
```

YAML

```
myELB:
  Type: AWS::ElasticLoadBalancing::LoadBalancer
  Properties:
    AvailabilityZones:
    - "us-east-1a"
    Listeners:
    - LoadBalancerPort: '80'
      InstancePort: '80'
      Protocol: HTTP
myDNS:
  Type: AWS::Route53::RecordSetGroup
  Properties:
    HostedZoneName: example.com.
    Comment: Zone apex alias targeted to myELB LoadBalancer.
    RecordSets:
    - Name: example.com.
      Type: A
      AliasTarget:
        HostedZoneId: !GetAtt myELB.CanonicalHostedZoneNameID
        DNSName: !GetAtt myELB.DNSName
```

An Alias Resource Record Set for a CloudFront Distribution

The following example creates an alias record set that routes queries to the specified CloudFront distribution domain name.

JSON

```
"myDNS" : {
    "Type" : "AWS::Route53::RecordSetGroup",
    "Properties" : {
        "HostedZoneId" : { "Ref" : "myHostedZoneID" },
        "RecordSets" : [{
            "Name" : { "Ref" : "myRecordSetDomainName" },
            "Type" : "A",
            "AliasTarget" : {
                "HostedZoneId" : "Z2FDTNDATAQYW2",
                "DNSName" : { "Ref" : "myCloudFrontDistributionDomainName" }
            }
        }]
    }
}
```

YAML

```
myDNS:
  Type: AWS::Route53::RecordSetGroup
  Properties:
    HostedZoneId:
      Ref: myHostedZoneID
    RecordSets:
    - Name:
```

```
        Ref: myRecordSetDomainName
      Type: A
      AliasTarget:
        HostedZoneId: Z2FDTNDATAQYW2
        DNSName:
          Ref: myCloudFrontDistributionDomainName
```

Amazon S3 Template Snippets

Topics

Creating an Amazon S3 Bucket with Defaults

This example uses a AWS::S3::Bucket (p. 937) to create a bucket with default settings.

JSON

```
"myS3Bucket" : {
      "Type" : "AWS::S3::Bucket"
      }
```

YAML

```
MyS3Bucket:
    Type: AWS::S3::Bucket
```

Creating an Amazon S3 Bucket for Website Hosting and with a DeletionPolicy

This example creates a bucket as a website. The AccessControl property is set to the canned ACL PublicRead (public read permissions are required for buckets set up for website hosting). Because this bucket resource has a DeletionPolicy attribute (p. 1297) set to `Retain`, AWS CloudFormation will not delete this bucket when it deletes the stack. The Output section uses `Fn::GetAtt` to retrieve the WebsiteURL attribute and DomainName attribute of the S3Bucket resource.

JSON

```
{
      "AWSTemplateFormatVersion" : "2010-09-09",
      "Resources" : {
          "S3Bucket" : {
             "Type" : "AWS::S3::Bucket",
             "Properties" : {
                 "AccessControl" : "PublicRead",
                 "WebsiteConfiguration" : {
                     "IndexDocument" : "index.html",
                     "ErrorDocument" : "error.html"
                 }
             },
             "DeletionPolicy" : "Retain"
```

```
            }
        },
        "Outputs" : {
            "WebsiteURL" : {
                  "Value" : { "Fn::GetAtt" : [ "S3Bucket", "WebsiteURL" ] },
                  "Description" : "URL for website hosted on S3"
            },
            "S3BucketSecureURL" : {
                  "Value" : { "Fn::Join" : [ "", [ "https://", { "Fn::GetAtt" : [ "S3Bucket",
 "DomainName" ] } ] ] },
                  "Description" : "Name of S3 bucket to hold website content"
            }
        }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  S3Bucket:
    Type: AWS::S3::Bucket
    Properties:
      AccessControl: PublicRead
      WebsiteConfiguration:
        IndexDocument: index.html
        ErrorDocument: error.html
    DeletionPolicy: Retain
Outputs:
  WebsiteURL:
    Value: !GetAtt S3Bucket.WebsiteURL
    Description: URL for the website hosted on S3
  S3BucketSecureURL:
    Value: !Sub
        - https://${Domain}
        - Domain: !GetAtt S3Bucket.DomainName
    Description: Name of the S3 bucket to hold website content
```

Creating a Static Website Using a Custom Domain

You can use Amazon Route 53 with a registered domain. The following sample assumes that you have already created a hosted zone in Amazon Route 53 for your domain. The example creates two buckets for website hosting. The root bucket hosts the content, and the other bucket redirects `www.domainname.com` requests to the root bucket. The record sets map your domain name to Amazon S3 endpoints.

For more information about using a custom domain, see Setting Up a Static Website Using a Custom Domain in the *Amazon Simple Storage Service Developer Guide*.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Mappings" : {
        "RegionMap" : {
            "us-east-1" : { "S3hostedzoneID" : "Z3AQBSTGFYJSTF", "websiteendpoint" : "s3-
website-us-east-1.amazonaws.com" },
            "us-west-1" : { "S3hostedzoneID" : "Z2F56UZL2M1ACD", "websiteendpoint" : "s3-
website-us-west-1.amazonaws.com" },
            "us-west-2" : { "S3hostedzoneID" : "Z3BJ6K6RIION7M", "websiteendpoint" : "s3-
website-us-west-2.amazonaws.com" },
            "eu-west-1" : { "S3hostedzoneID" : "Z1BKCTXD74EZPE", "websiteendpoint" : "s3-
website-eu-west-1.amazonaws.com" },
```

```
            "ap-southeast-1" : { "S3hostedzoneID" : "Z3O0J2DXBE1FTB", "websiteendpoint" :
 "s3-website-ap-southeast-1.amazonaws.com" },
            "ap-southeast-2" : { "S3hostedzoneID" : "Z1WCIGYICN2BYD", "websiteendpoint" :
 "s3-website-ap-southeast-2.amazonaws.com" },
            "ap-northeast-1" : { "S3hostedzoneID" : "Z2M4EHUR26P7ZW", "websiteendpoint" :
 "s3-website-ap-northeast-1.amazonaws.com" },
            "sa-east-1" : { "S3hostedzoneID" : "Z31GFT0UA1I2HV", "websiteendpoint" : "s3-
website-sa-east-1.amazonaws.com" }
        }
    },
    "Parameters": {
        "RootDomainName": {
            "Description": "Domain name for your website (example.com)",
            "Type": "String"
        }
    },
    "Resources": {
        "RootBucket": {
            "Type": "AWS::S3::Bucket",
            "Properties": {
                "BucketName" : {"Ref":"RootDomainName"},
                "AccessControl": "PublicRead",
                "WebsiteConfiguration": {
                    "IndexDocument":"index.html",
                    "ErrorDocument":"404.html"
                }
            }
        },
        "WWWBucket": {
            "Type": "AWS::S3::Bucket",
            "Properties": {
                "BucketName": {
                    "Fn::Join": ["", ["www.", {"Ref":"RootDomainName"}]]
                },
                "AccessControl": "BucketOwnerFullControl",
                "WebsiteConfiguration": {
                    "RedirectAllRequestsTo": {
                        "HostName": {"Ref": "RootBucket"}
                    }
                }
            }
        },
        "myDNS": {
            "Type": "AWS::Route53::RecordSetGroup",
            "Properties": {
                "HostedZoneName": {
                    "Fn::Join": ["", [{"Ref": "RootDomainName"}, "."]]
                },
                "Comment": "Zone apex alias.",
                "RecordSets": [
                    {
                        "Name": {"Ref": "RootDomainName"},
                        "Type": "A",
                        "AliasTarget": {
                            "HostedZoneId": {"Fn::FindInMap" : [ "RegionMap", { "Ref" :
 "AWS::Region" }, "S3hostedzoneID"]},
                            "DNSName": {"Fn::FindInMap" : [ "RegionMap", { "Ref" :
 "AWS::Region" }, "websiteendpoint"]}
                        }
                    },
                    {
                        "Name": {
                            "Fn::Join": ["", ["www.", {"Ref":"RootDomainName"}]]
                        },
                        "Type": "CNAME",
                        "TTL" : "900",
```

```
                        "ResourceRecords" : [
                            {"Fn::GetAtt":["WWWBucket", "DomainName"]}
                        ]
                    }
                ]
            }
        }
    },
    "Outputs": {
        "WebsiteURL": {
            "Value": {"Fn::GetAtt": ["RootBucket", "WebsiteURL"]},
            "Description": "URL for website hosted on S3"
        }
    }
}
```

YAML

```
Parameters:
  RootDomainName:
    Description: Domain name for your website (example.com)
    Type: String
Mappings:
  RegionMap:
    us-east-1:
      S3hostedzoneID: Z3AQBSTGFYJSTF
      websiteendpoint: s3-website-us-east-1.amazonaws.com
    us-west-1:
      S3hostedzoneID: Z2F56UZL2M1ACD
      websiteendpoint: s3-website-us-west-1.amazonaws.com
    us-west-2:
      S3hostedzoneID: Z3BJ6K6RIION7M
      websiteendpoint: s3-website-us-west-2.amazonaws.com
    eu-west-1:
      S3hostedzoneID: Z1BKCTXD74EZPE
      websiteendpoint: s3-website-eu-west-1.amazonaws.com
    ap-southeast-1:
      S3hostedzoneID: Z3O0J2DXBE1FTB
      websiteendpoint: s3-website-ap-southeast-1.amazonaws.com
    ap-southeast-2:
      S3hostedzoneID: Z1WCIGYICN2BYD
      websiteendpoint: s3-website-ap-southeast-2.amazonaws.com
    ap-northeast-1:
      S3hostedzoneID: Z2M4EHUR26P7ZW
      websiteendpoint: s3-website-ap-northeast-1.amazonaws.com
    sa-east-1:
      S3hostedzoneID: Z31GFT0UA1I2HV
      websiteendpoint: s3-website-sa-east-1.amazonaws.com
Resources:
  RootBucket:
    Type: AWS::S3::Bucket
    Properties:
      BucketName: !Ref RootDomainName
      AccessControl: PublicRead
      WebsiteConfiguration:
        IndexDocument: index.html
        ErrorDocument: 404.html
  WWWBucket:
    Type: AWS::S3::Bucket
    Properties:
      BucketName: !Sub
          - www.${Domain}
          - Domain: !Ref RootDomainName
      AccessControl: BucketOwnerFullControl
```

```
      WebsiteConfiguration:
        RedirectAllRequestsTo:
          HostName: !Ref RootBucket
  myDNS:
    Type: AWS::Route53::RecordSetGroup
    Properties:
      HostedZoneName: !Sub
          - ${Domain}.
          - Domain: !Ref RootDomainName
      Comment: Zone apex alias.
      RecordSets:
      -
        Name: !Ref RootDomainName
        Type: A
        AliasTarget:
          HostedZoneId: !FindInMap [ RegionMap, !Ref 'AWS::Region', S3hostedzoneID]
          DNSName: !FindInMap [ RegionMap, !Ref 'AWS::Region', websiteendpoint]
      -
        Name: !Sub
            - www.${Domain}
            - Domain: !Ref RootDomainName
        Type: CNAME
        TTL: 900
        ResourceRecords:
        - !GetAtt WWWBucket.DomainName
Outputs:
  WebsiteURL:
    Value: !GetAtt RootBucket.WebsiteURL
    Description: URL for website hosted on S3
```

Amazon SNS Template Snippets

This example shows an Amazon SNS topic resource. It requires a valid email address.

JSON

```
"MySNSTopic" : {
    "Type" : "AWS::SNS::Topic",
    "Properties" : {
        "Subscription" : [ {
            "Endpoint" : "add valid email address",
            "Protocol" : "email"
        } ]
    }
}
```

YAML

```
MySNSTopic:
  Type: AWS::SNS::Topic
  Properties:
    Subscription:
    - Endpoint: "add valid email address"
      Protocol: email
```

Amazon SQS Template Snippets

This example shows an Amazon SQS queue.

JSON

```
"MyQueue" : {
    "Type" : "AWS::SQS::Queue",
    "Properties" : {
        "VisibilityTimeout" : "value"
    }
}
```

YAML

```
MyQueue:
  Type: AWS::SQS::Queue
  Properties:
    VisibilityTimeout: value
```

Custom Resources

Custom resources enable you to write custom provisioning logic in templates that AWS CloudFormation runs anytime you create, update (if you changed the custom resource), or delete stacks. For example, you might want to include resources that aren't available as AWS CloudFormation resource types (p. 390). You can include those resources by using custom resources. That way you can still manage all your related resources in a single stack.

Use the `AWS::CloudFormation::CustomResource` (p. 467) or `Custom::String` (p. 468) resource type to define custom resources in your templates. Custom resources require one property: the service token, which specifies where AWS CloudFormation sends requests to, such as an Amazon SNS topic.

> **Note**
> If you use the VPC endpoint feature, custom resources in the VPC must have access to AWS CloudFormation-specific S3 buckets. Custom resources must send responses to a pre-signed Amazon S3 URL. If they can't send responses to Amazon S3, AWS CloudFormation won't receive a response and the stack operation fails. For more information, see AWS CloudFormation and VPC Endpoints (p. 19).

How Custom Resources Work

Any action taken for a custom resource involves three parties.

template developer

Creates a template that includes a custom resource type. The template developer specifies the service token and any input data in the template.

custom resource provider

Owns the custom resource and determines how to handle and respond to requests from AWS CloudFormation. The custom resource provider must provide a service token that the template developer uses.

AWS CloudFormation

During a stack operation, sends a request to a service token that is specified in the template, and then waits for a response before proceeding with the stack operation.

The template developer and custom resource provider can be the same person or entity, but the process is the same. The following steps describe the general process:

1. The template developer defines a custom resource in his or her template, which includes a service token and any input data parameters. Depending on the custom resource, the input data might be required; however, the service token is always required.

   The service token specifies where AWS CloudFormation sends requests to, such as to an Amazon SNS topic ARN or to an AWS Lambda function ARN. For more information, see AWS::CloudFormation::CustomResource (p. 467). The service token and the structure of the input data is defined by the custom resource provider.
2. Whenever anyone uses the template to create, update, or delete a custom resource, AWS CloudFormation sends a request to the specified service token. The service token must be in the same region in which you are creating the stack.

   In the request, AWS CloudFormation includes information such as the request type and a pre-signed Amazon Simple Storage Service URL, where the custom resource sends responses to. For more information about what's included in the request, see Custom Resource Request Objects (p. 373).

   The following sample data shows what AWS CloudFormation includes in a request:

```
{
   "RequestType" : "Create",
   "ResponseURL" : "http://pre-signed-S3-url-for-response",
   "StackId" : "arn:aws:cloudformation:us-west-2:EXAMPLE/stack-name/guid",
   "RequestId" : "unique id for this create request",
   "ResourceType" : "Custom::TestResource",
   "LogicalResourceId" : "MyTestResource",
   "ResourceProperties" : {
      "Name" : "Value",
      "List" : [ "1", "2", "3" ]
   }
}
```

   Note
   In this example, `ResourceProperties` allows AWS CloudFormation to create a custom payload to send to the Lambda function.
3. The custom resource provider processes the AWS CloudFormation request and returns a response of `SUCCESS` or `FAILED` to the pre-signed URL. The custom resource provider provides the response in a JSON-formatted file and uploads it to the pre-signed S3 URL. For more information, see Uploading Objects Using Pre-Signed URLs in the *Amazon Simple Storage Service Developer Guide*.

   In the response, the custom resource provider can also include name-value pairs that the template developer can access. For example, the response can include output data if the request succeeded or an error message if the request failed. For more information about responses, see Custom Resource Response Objects (p. 375).

   The custom resource provider is responsible for listening and responding to the request. For example, for Amazon SNS notifications, the custom resource provider must listen and respond to notifications that are sent to a specific topic ARN. AWS CloudFormation waits and listens for a response in the pre-signed URL location.

   The following sample data shows what a custom resource might include in a response:

```
{
   "Status" : "SUCCESS",
   "PhysicalResourceId" : "TestResource1",
   "StackId" : "arn:aws:cloudformation:us-west-2:EXAMPLE:stack/stack-name/guid",
   "RequestId" : "unique id for this create request",
```

```
   "LogicalResourceId" : "MyTestResource",
   "Data" : {
      "OutputName1" : "Value1",
      "OutputName2" : "Value2",
   }
}
```

4. After getting a `SUCCESS` response, AWS CloudFormation proceeds with the stack operation. If a `FAILURE` or no response is returned, the operation fails. Any output data from the custom resource is stored in the pre-signed URL location. The template developer can retrieve that data by using the `Fn::GetAtt` (p. 1324) function.

Topics

Amazon Simple Notification Service-backed Custom Resources

When you associate an Amazon SNS topic with a custom resource, you use Amazon SNS notifications to trigger custom provisioning logic. With custom resources and Amazon SNS, you can enable scenarios such as adding new resources to a stack and injecting dynamic data into a stack. For example, when you create a stack, AWS CloudFormation can send a `create` request to a topic that's monitored by an application that's running on an Amazon Elastic Compute Cloud instance. The Amazon SNS notification triggers the application to carry out additional provisioning tasks, such as retrieve a pool of white-listed Elastic IPs. After it's done, the application sends a response (and any output data) that notifies AWS CloudFormation to proceed with the stack operation.

Walkthrough: Using Amazon Simple Notification Service to Create Custom Resources

This walkthrough will step through the custom resource process, explaining the sequence of events and messages sent and received as a result of custom resource stack creation, updates, and deletion.

Step 1: Stack Creation

1. The template developer creates an AWS CloudFormation stack that contains a custom resource; in the template example below, we use the custom resource type name `Custom::SeleniumTester` for the custom resource `MySeleniumTest`.

   The custom resource type is declared with a *service token*, optional *provider-specific properties*, and optional `Fn::GetAtt` (p. 1324) attributes that are defined by the custom resource provider. These properties and attributes can be used to pass information from the template developer to the custom resource provider and vice-versa. Custom resource type names must be alphanumeric and can have a maximum length of 60 characters.

   The following example shows a template that has both custom properties and return attributes:

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "MySeleniumTest" : {
         "Type": "Custom::SeleniumTester",
         "Version" : "1.0",
```

```
         "Properties" : {
            "ServiceToken": "arn:aws:sns:us-west-2:123456789012:CRTest",
            "seleniumTester" : "SeleniumTest()",
            "endpoints" : [ "http://mysite.com", "http://myecommercesite.com/", "http://
search.mysite.com" ],
            "frequencyOfTestsPerHour" : [ "3", "2", "4" ]
         }
      }
   },
   "Outputs" : {
      "topItem" : {
         "Value" : { "Fn::GetAtt" : ["MySeleniumTest", "resultsPage"] }
      },
      "numRespondents" : {
         "Value" : { "Fn::GetAtt" : ["MySeleniumTest", "lastUpdate"] }
      }
   }
}
```

Note

The names and values of the data accessed with `Fn::GetAtt` are returned by the custom resource provider during the provider's response to AWS CloudFormation. If the custom resource provider is a third-party, then the template developer must obtain the names of these return values from the custom resource provider.

2. AWS CloudFormation sends an Amazon SNS notification to the resource provider with a `"RequestType" : "Create"` that contains information about the stack, the custom resource properties from the stack template, and an S3 URL for the response.

   The SNS topic that is used to send the notification is embedded in the template in the `ServiceToken` property. To avoid using a hard-coded value, a template developer can use a template parameter so that the value is entered at the time the stack is launched.

   The following example shows a custom resource `Create` request which includes a custom resource type name, `Custom::SeleniumTester`, created with a `LogicalResourceId` of `MySeleniumTester`:

```
{
   "RequestType" : "Create",
   "ResponseURL" : "http://pre-signed-S3-url-for-response",
   "StackId" : "arn:aws:cloudformation:us-west-2:123456789012:stack/stack-name/guid",
   "RequestId" : "unique id for this create request",
   "ResourceType" : "Custom::SeleniumTester",
   "LogicalResourceId" : "MySeleniumTester",
   "ResourceProperties" : {
      "seleniumTester" : "SeleniumTest()",
      "endpoints" : [ "http://mysite.com", "http://myecommercesite.com/", "http://
search.mysite.com" ],
      "frequencyOfTestsPerHour" : [ "3", "2", "4" ]
   }
}
```

3. The custom resource provider processes the data sent by the template developer and determines whether the `Create` request was successful. The resource provider then uses the S3 URL sent by AWS CloudFormation to send a response of either `SUCCESS` or `FAILED`.

   Depending on the response type, different response fields will be expected by AWS CloudFormation. Refer to the Responses section in the reference topic for the RequestType that is being processed.

   In response to a create or update request, the custom resource provider can return data elements in the Data (p. 375) field of the response. These are name/value pairs, and the *names* correspond to the `Fn::GetAtt` attributes used with the custom resource in the stack template. The *values* are the data that is returned when the template developer calls `Fn::GetAtt` on the resource with the attribute name.

The following is an example of a custom resource response:

```
{
   "Status" : "SUCCESS",
   "PhysicalResourceId" : "Tester1",
   "StackId" : "arn:aws:cloudformation:us-west-2:123456789012:stack/stack-name/guid",
   "RequestId" : "unique id for this create request",
   "LogicalResourceId" : "MySeleniumTester",
   "Data" : {
      "resultsPage" : "http://www.myexampledomain/test-results/guid",
      "lastUpdate" : "2012-11-14T03:30Z",
   }
}
```

The `StackId`, `RequestId`, and `LogicalResourceId` fields must be copied verbatim from the request.

4. AWS CloudFormation declares the stack status as `CREATE_COMPLETE` or `CREATE_FAILED`. If the stack was successfully created, the template developer can use the output values of the created custom resource by accessing them with `Fn::GetAtt` (p. 1324).

   For example, the custom resource template used for illustration used `Fn::GetAtt` to copy resource outputs into the stack outputs:

```
"Outputs" : {
   "topItem" : {
      "Value" : { "Fn::GetAtt" : ["MySeleniumTest", "resultsPage"] }
   },
   "numRespondents" : {
      "Value" : { "Fn::GetAtt" : ["MySeleniumTest", "lastUpdate"] }
   }
}
```

For detailed information about the request and response objects involved in `Create` requests, see Create (p. 376) in the Custom Resource Reference (p. 372).

Step 2: Stack Updates

To update an existing stack, you must submit a template that specifies updates for the properties of resources in the stack, as shown in the example below. AWS CloudFormation updates only the resources that have changes specified in the template. For more information about updating stacks, see AWS CloudFormation Stacks Updates (p. 89).

You can update custom resources that require a replacement of the underlying physical resource. When you update a custom resource in an AWS CloudFormation template, AWS CloudFormation sends an update request to that custom resource. If a custom resource requires a replacement, the new custom resource must send a response with the new physical ID. When AWS CloudFormation receives the response, it compares the `PhysicalResourceId` between the old and new custom resources. If they are different, AWS CloudFormation recognizes the update as a replacement and sends a delete request to the old resource, as shown in Step 3: Stack Deletion (p. 367).

> **Note**
> If you didn't make changes to the custom resource, AWS CloudFormation won't send requests to it during a stack update.

1. The template developer initiates an update to the stack that contains a custom resource. During an update, the template developer can specify new Properties in the stack template.

The following is an example of an `Update` to the stack template using a custom resource type:

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "MySeleniumTest" : {
         "Type": "Custom::SeleniumTester",
         "Version" : "1.0",
         "Properties" : {
            "ServiceToken": "arn:aws:sns:us-west-2:123456789012:CRTest",
            "seleniumTester" : "SeleniumTest()",
            "endpoints" : [ "http://mysite.com", "http://myecommercesite.com/", "http://
search.mysite.com",
               "http://mynewsite.com" ],
            "frequencyOfTestsPerHour" : [ "3", "2", "4", "3" ]
         }
      }
   },
   "Outputs" : {
      "topItem" : {
         "Value" : { "Fn::GetAtt" : ["MySeleniumTest", "resultsPage"] }
      },
      "numRespondents" : {
         "Value" : { "Fn::GetAtt" : ["MySeleniumTest", "lastUpdate"] }
      }
   }
}
```

2. AWS CloudFormation sends an Amazon SNS notification to the resource provider with a `"RequestType" : "Update"` that contains similar information to the `Create` call, except that the `OldResourceProperties` field contains the old resource properties, and ResourceProperties contains the updated (if any) resource properties.

   The following is an example of an `Update` request:

```
{
   "RequestType" : "Update",
   "ResponseURL" : "http://pre-signed-S3-url-for-response",
   "StackId" : "arn:aws:cloudformation:us-west-2:123456789012:stack/stack-name/guid",
   "RequestId" : "uniqueid for this update request",
   "LogicalResourceId" : "MySeleniumTester",
   "ResourceType" : "Custom::SeleniumTester"
   "PhysicalResourceId" : "Tester1",
   "ResourceProperties" : {
      "seleniumTester" : "SeleniumTest()",
      "endpoints" : [ "http://mysite.com", "http://myecommercesite.com/", "http://
search.mysite.com",
         "http://mynewsite.com" ],
      "frequencyOfTestsPerHour" : [ "3", "2", "4", "3" ]
   }
   "OldResourceProperties" : {
      "seleniumTester" : "SeleniumTest()",
      "endpoints" : [ "http://mysite.com", "http://myecommercesite.com/", "http://
search.mysite.com" ],
      "frequencyOfTestsPerHour" : [ "3", "2", "4" ]
   }
}
```

3. The custom resource provider processes the data sent by AWS CloudFormation. The custom resource performs the update and sends a response of either `SUCCESS` or `FAILED` to the S3 URL. AWS

CloudFormation then compares the `PhysicalResourceIDs` of old and new custom resources. If they are different, AWS CloudFormation recognizes that the update requires a replacement and sends a delete request to the old resource. The following example demonstrates the custom resource provider response to an `Update` request.

```
{
   "Status" : "SUCCESS",
   "StackId" : "arn:aws:cloudformation:us-west-2:123456789012:stack/stack-name/guid",
   "RequestId" : "uniqueid for this update request",
   "LogicalResourceId" : "MySeleniumTester",
   "PhysicalResourceId" : "Tester2"
}
```

The `StackId`, `RequestId`, and `LogicalResourceId` fields must be copied verbatim from the request.

4. AWS CloudFormation declares the stack status as `UPDATE_COMPLETE` or `UPDATE_FAILED`. If the update fails, the stack rolls back. If the stack was successfully updated, the template developer can access any new output values of the created custom resource with `Fn::GetAtt`.

For detailed information about the request and response objects involved in `Update` requests, see Update (p. 381) in the Custom Resource Reference (p. 372).

Step 3: Stack Deletion

1. The template developer deletes a stack that contains a custom resource. AWS CloudFormation gets the current properties specified in the stack template along with the SNS topic, and prepares to make a request to the custom resource provider.
2. AWS CloudFormation sends an Amazon SNS notification to the resource provider with a `"RequestType" : "Delete"` that contains current information about the stack, the custom resource properties from the stack template, and an S3 URL for the response.

   Whenever you delete a stack or make an update that removes or replaces the custom resource, AWS CloudFormation compares the `PhysicalResourceId` between the old and new custom resources. If they are different, AWS CloudFormation recognizes the update as a replacement and sends a delete request for the old resource (`OldPhysicalResource`), as shown in the following example of a `Delete` request.

```
{
   "RequestType" : "Delete",
   "ResponseURL" : "http://pre-signed-S3-url-for-response",
   "StackId" : "arn:aws:cloudformation:us-west-2:123456789012:stack/stack-name/guid",
   "RequestId" : "unique id for this delete request",
   "ResourceType" : "Custom::SeleniumTester",
   "LogicalResourceId" : "MySeleniumTester",
   "PhysicalResourceId" : "Tester1",
   "ResourceProperties" : {
      "seleniumTester" : "SeleniumTest()",
      "endpoints" : [ "http://mysite.com", "http://myecommercesite.com/", "http://
search.mysite.com",
         "http://mynewsite.com" ],
      "frequencyOfTestsPerHour" : [ "3", "2", "4", "3" ]
   }
}
```

`DescribeStackResource`, `DescribeStackResources`, and `ListStackResources` display the user-defined name if it has been specified.

3. The custom resource provider processes the data sent by AWS CloudFormation and determines whether the `Delete` request was successful. The resource provider then uses the S3 URL sent by AWS CloudFormation to send a response of either `SUCCESS` or `FAILED`. To successfully delete a stack with a custom resource, the custom resource provider must respond successfully to a delete request.

   The following is an example of a custom resource provider response to a `Delete` request:

```
{
   "Status" : "SUCCESS",
   "StackId" : "arn:aws:cloudformation:us-west-2:123456789012:stack/stack-name/guid",
   "RequestId" : "unique id for this delete request",
   "LogicalResourceId" : "MySeleniumTester",
   "PhysicalResourceId" : "Tester1"
}
```

   The `StackId`, `RequestId`, and `LogicalResourceId` fields must be copied verbatim from the request.
4. AWS CloudFormation declares the stack status as `DELETE_COMPLETE` or `DELETE_FAILED`.

For detailed information about the request and response objects involved in `Delete` requests, see Delete (p. 378) in the Custom Resource Reference (p. 372).

See Also

- AWS CloudFormation Custom Resource Reference (p. 372)
- AWS::CloudFormation::CustomResource (p. 467)
- Fn::GetAtt (p. 1324)

AWS Lambda-backed Custom Resources

When you associate a Lambda function with a custom resource, the function is invoked whenever the custom resource is created, updated, or deleted. AWS CloudFormation calls a Lambda API to invoke the function and to pass all the request data (such as the request type and resource properties) to the function, The power and customizability of Lambda functions in combination with AWS CloudFormation enable a wide range of scenarios, such as dynamically looking up AMI IDs during stack creation, or implementing and using utility functions, such as string reversal functions.

Topics

Walkthrough: Looking Up Amazon Machine Image IDs

AWS CloudFormation templates that declare an Amazon Elastic Compute Cloud (Amazon EC2) instance must also specify an Amazon Machine Image (AMI) ID, which includes an operating system and other software and configuration information used to launch the instance. The correct AMI ID depends on the instance type and region in which you're launching your stack. And IDs can change regularly, such as when an AMI is updated with software updates.

Normally, you might map AMI IDs to specific instance types and regions. To update the IDs, you manually change them in each of your templates. By using custom resources and AWS Lambda (Lambda), you can create a function that gets the IDs of the latest AMIs for the region and instance type that you're using so that you don't have to maintain mappings.

This walkthrough shows you how to create a custom resource and associate a Lambda function with it to look up AMI IDs. Note that the walkthrough assumes that you understand how to use custom resources and Lambda. For more information, see Custom Resources (p. 361) or the AWS Lambda Developer Guide.

Walkthrough Overview

For this walkthrough, you'll create a stack with a custom resource, a Lambda function, and an EC2 instance. The walkthough provides sample code and a sample template that you'll use to create the stack.

The sample template uses the custom resource type to invoke and send input values to the Lambda function. When you use the template, AWS CloudFormation invokes the function and sends information to it, such as the request type, input data, and a pre-signed Amazon Simple Storage Service (Amazon S3) URL. The function uses that information to look up the AMI ID, and then sends a response to the pre-signed URL.

After AWS CloudFormation gets a response in the pre-signed URL location, it proceeds with creating the stack. When AWS CloudFormation creates the instance, it uses the Lambda function's response to specify the instance's AMI ID.

The following list summarizes the process. You need AWS Identity and Access Management (IAM) permissions to use all the corresponding services, such as Lambda, Amazon EC2, and AWS CloudFormation.

> **Note**
> AWS CloudFormation is a free service; however, you are charged for the AWS resources, such as the Lambda function and EC2 instance, that you include in your stacks at the current rate for each. For more information about AWS pricing, see the detail page for each product at http://aws.amazon.com.

1. Save the sample Lambda package in an Amazon Simple Storage Service (Amazon S3) bucket. (p. 369)

   The sample package contains everything that's required to create the Lambda function. You must save the package in a bucket that's in the same region in which you will create your stack.
2. Use the sample template to create a stack. (p. 370)

   The stack demonstrates how you associate the Lambda function with a custom resource and how to use the results from the function to specify an AMI ID. The stack also creates an IAM role (execution role), which Lambda uses to make calls to Amazon EC2.
3. Delete the stack. (p. 372)

   Delete the stack to clean up all the stack resources that you created so that you aren't charged for unnecessary resources.

Step 1: Downloading and Saving the Sample Package in Amazon S3

When you create a stack with a Lambda function, you must specify the location of the Amazon S3 bucket that contains the function's source code. The bucket must be in the same region in which you create your stack.

This walkthrough provides a sample package (a `.zip` file) that's required to create the Lambda function. A Lambda package contains the source code for the function and required libraries. For this walkthrough, the function doesn't require additional libraries.

The function takes an instance's architecture and region as inputs from an AWS CloudFormation custom resource request and returns the latest AMI ID to a pre-signed Amazon S3 URL.

To download and save the package in Amazon S3

1. Download the sample package from Amazon S3. When you save the file, use the same file name as the sample, `amilookup.zip` or `amilookup-win.zip`.

   Look up Linux AMI IDs

   https://s3.amazonaws.com/cloudformation-examples/lambda/amilookup.zip

   Look up Windows AMI IDs

   https://s3.amazonaws.com/cloudformation-examples/lambda/amilookup-win.zip
2. Open the Amazon S3 console at https://console.aws.amazon.com/s3/home.
3. Choose or create a bucket that's located in the same region in which you'll create your AWS CloudFormation stack. Record the bucket name.

   You'll save the sample package in this bucket. For more information about creating a bucket, see Creating a Bucket in the *Amazon Simple Storage Service Console User Guide*.
4. Upload the sample package to the bucket that you chose or created.

   For more information about uploading objects, see Uploading Objects in the *Amazon Simple Storage Service Console User Guide*.

With the package in Amazon S3, you can now specify its location in the Lambda resource declaration of the AWS CloudFormation template. The next step demonstrates how you declare the function and invoke it by using a custom resource. You'll also see how to use the results of the function to specify the AMI ID of an EC2 instance.

Step 2: Creating the Stack

To create the sample Amazon EC2 stack, you'll use a sample template that includes a Lambda function, an IAM execution role, a custom resource that invokes the function, and an EC2 instance that uses the results from the function.

During stack creation, the custom resource invokes the Lambda function and waits until the function sends a response to the pre-signed Amazon S3 URL. In the response, the function returns the ID of the latest AMI that corresponds to the EC2 instance type and region in which you are creating the instance. The data from the function's response is stored as an attribute of the custom resource, which is used to specify the AMI ID of the EC2 instance.

The following snippets explain relevant parts of the sample template to help you understand how to associate a Lambda function with a custom resource and how to use the function's response. To view the entire sample template, see:

Linux template

https://s3.amazonaws.com/cloudformation-examples/lambda/LambdaAMILookupSample.template

Windows template

https://s3.amazonaws.com/cloudformation-examples/lambda/LambdaAMILookupSample-win.template

Stack Template Snippets

To create the Lambda function, you declare the `AWS::Lambda::Function` resource, which requires the function's source code, handler name, runtime environment, and execution role ARN.

The `Code` property specifies the Amazon S3 location (bucket name and file name) where you uploaded the sample package. The sample template uses input parameters (`"Ref": "S3Bucket"` and `"Ref": "S3Key"`) to set the bucket and file names so that you are able to specify the names when you create the stack. Similarly, the handler name, which corresponds to the name of the source file (the JavaScript file) in the `.zip` package, also uses an input parameter (`"Ref": "ModuleName"`). Because the source file is JavaScript code, the runtime is specified as `nodejs4.3`.

For this walkthrough, the execution time for the function exceeds the default value of `3` seconds, so the timeout is set to `30` seconds. If you don't specify a sufficiently long timeout, Lambda might cause a timeout before the function can complete, causing stack creation to fail.

The execution role, which is declared elsewhere in the template, is specified by using the `Fn::GetAtt` intrinsic function in the `Role` property. The execution role grants the Lambda function permission to send logs to AWS and to call the EC2 `DescribeImages` API. The following snippet shows the role and policy that grant the appropriate permission:

For both the Linux and Windows templates, the custom resource invokes the Lambda function that is associated with it. To associate a function with a custom resource, you specify the Amazon Resource Name (ARN) of the function for the `ServiceToken` property, using the `Fn::GetAtt` intrinsic function. AWS CloudFormation sends the additional properties that are included in the custom resource declaration, such as `Region` and `Architecture`, to the Lambda function as inputs. The Lambda function determines the correct names and values for these input properties.

For Windows, the custom resource provides the Windows version to the Lambda function instead of the instance's architecture.

When AWS CloudFormation invokes the Lambda function, the function calls the EC2 `DescribeImages` API, using the region and instance architecture or the OS name to filter the list of images. Then the function sorts the list of images by date and returns the ID of the latest AMI.

When returning the ID of the latest AMI, the function sends the ID to a pre-signed URL in the `Data` property of the response object (p. 375). The data is structured as a name-value pair, as shown in the following example:

```
"Data": {
  "Id": "ami-43795473"
}
```

The following snippet shows how to get the data from a Lambda function. It uses the `Fn::GetAtt` intrinsic function, providing the name of the custom resource and the attribute name of the value that you want to get. In this walkthrough, the custom resource name is `AMIInfo` and the attribute name is `Id`.

Now that you understand what the template does, use the sample template to create a stack.

To create the stack

1. Open the AWS CloudFormation console at https://console.aws.amazon.com/cloudformation/.
2. Choose **Create Stack**.
3. In the **Template** section, choose **Specify an Amazon S3 template URL**, and then copy and paste the following URL in the text box:

   Linux template

   `https://s3.amazonaws.com/cloudformation-examples/lambda/LambdaAMILookupSample.template`

   Windows template

   `https://s3.amazonaws.com/cloudformation-examples/lambda/LambdaAMILookupSample-win.template`
4. Choose **Next**.

5. In the **Stack name** field, type **`SampleEC2Instance`**.
6. In the **Parameters** section, specify the name of the Amazon S3 bucket that you created, and then choose **Next**.

   The default values for the other parameters are the same names that are used in the sample `.zip` package.
7. For this walkthrough, you don't need to add tags or specify advanced settings, so choose **Next**.
8. Ensure that the stack name and template URL are correct, and then choose **Create**.

It might take several minutes for AWS CloudFormation to create your stack. To monitor progress, view the stack events. For more information, see Viewing Stack Data and Resources (p. 78).

If stack creation succeeds, all resources in the stack, such as the Lambda function, custom resource, and EC2 instance, were created. You successfully used a Lambda function and custom resource to specify the AMI ID of an EC2 instance. You don't need to create and maintain a mapping of AMI IDs in this template.

To see which AMI ID AWS CloudFormation used to create the EC2 instance, view the stack outputs.

If the Lambda function returns an error, view the function's logs in the Amazon CloudWatch Logs console. The name of the log stream is the the physical ID of the custom resource, which you can find by viewing the stack's resources. For more information, see Viewing Log Data in the *Amazon CloudWatch User Guide*.

Step 3: Clean Up Resources

To make sure that you are not charged for unwanted services, delete your stack.

To delete the stack

1. From the AWS CloudFormation console, choose the **SampleEC2Instance** stack.
2. Choose **Actions** and then **Delete Stack**.
3. In the confirmation message, choose **Yes, Delete**.

All the resources that you created are deleted.

Now that you understand how to create and use Lambda functions with AWS CloudFormation, you can use the sample template and code from this walkthrough to build other stacks and functions.

Related Information

- AWS CloudFormation Custom Resource Reference (p. 372)

Custom Resource Reference

This section provides detail about:

- The JSON request and response fields that are used in messages sent to and from AWS CloudFormation when providing a custom resource.
- Expected fields for requests to, and responses to, the custom resource provider in response to stack creation, stack updates, and stack deletion.

In This Section

Custom Resource Request Objects

Template Developer Request Properties

The template developer uses the AWS CloudFormation resource, AWS::CloudFormation::CustomResource (p. 467), to specify a custom resource in a template.

In `AWS::CloudFormation::CustomResource`, all properties are defined by the custom resource provider. There is only one required property: `ServiceToken`.

ServiceToken

The service token (an Amazon SNS topic or AWS Lambda function Amazon Resource Name) that is obtained from the custom resource provider to access the service. The service token must be in the same region in which you are creating the stack.

Required: Yes

Type: String

All other fields in the resource properties are optional and are sent, verbatim, to the custom resource provider in the request's `ResourceProperties` field. The provider defines both the names and the valid contents of these fields.

Custom Resource Provider Request Fields

These fields are sent in JSON requests from AWS CloudFormation to the custom resource provider in the SNS topic that the provider has configured for this purpose.

RequestType

The request type is set by the AWS CloudFormation stack operation (create-stack, update-stack, or delete-stack) that was initiated by the template developer for the stack that contains the custom resource.

Must be one of: `Create`, `Update`, or `Delete`. For more information, see Custom Resource Request Types (p. 376).

Required: Yes

Type: String

ResponseURL

The response URL identifies a pre-signed S3 bucket that receives responses from the custom resource provider to AWS CloudFormation.

Required: Yes

Type: String

StackId

The Amazon Resource Name (ARN) that identifies the stack that contains the custom resource.

Combining the `StackId` with the `RequestId` forms a value that you can use to uniquely identify a request on a particular custom resource.

Required: Yes

Type: String

RequestId

A unique ID for the request.

Combining the `StackId` with the `RequestId` forms a value that you can use to uniquely identify a request on a particular custom resource.

Required: Yes

Type: String

ResourceType

The template developer-chosen resource type of the custom resource in the AWS CloudFormation template. Custom resource type names can be up to 60 characters long and can include alphanumeric and the following characters: `_@-`.

Required: Yes

Type: String

LogicalResourceId

The template developer-chosen name (logical ID) of the custom resource in the AWS CloudFormation template. This is provided to facilitate communication between the custom resource provider and the template developer.

Required: Yes

Type: String

PhysicalResourceId

A required custom resource provider-defined physical ID that is unique for that provider.

Required: Always sent with `Update` and `Delete` requests; never sent with `Create`.

Type: String

ResourceProperties

This field contains the contents of the `Properties` object sent by the template developer. Its contents are defined by the custom resource provider.

Required: No

Type: JSON object

OldResourceProperties

Used only for `Update` requests. Contains the resource properties that were declared previous to the update request.

Required: Yes

Type: JSON object

Custom Resource Response Objects

Custom Resource Provider Response Fields

The properties that the custom resource provider includes when sending the JSON file to the pre-signed URL. For more information about uploading objects by using pre-signed URLs, see the related topic in the *Amazon Simple Storage Service Developer Guide*.

Status

The status value sent by the custom resource provider in response to an AWS CloudFormation-generated request.

Must be either `SUCCESS` or `FAILED`.

Required: Yes

Type: String

Reason

Describes the reason for a failure response.

Required: Required if `Status` is `FAILED`; optional otherwise.

Type: String

PhysicalResourceId

This value should be an identifier unique to the custom resource vendor, and can be up to 1 Kb in size. The value must be a non-empty string and must be identical for all responses for the same resource.

Required: Yes

Type: String

StackId

The Amazon Resource Name (ARN) that identifies the stack that contains the custom resource. This response value should be copied *verbatim* from the request.

Required: Yes

Type: String

RequestId

A unique ID for the request. This response value should be copied *verbatim* from the request.

Required: Yes

Type: String

LogicalResourceId

The template developer-chosen name (logical ID) of the custom resource in the AWS CloudFormation template. This response value should be copied *verbatim* from the request.

Required: Yes

Type: String

Data

Optional, custom resource provider-defined name-value pairs to send with the response. You can access the values provided here by name in the template with `Fn::GetAtt`.

Required: No

Type: JSON object

Custom Resource Request Types

The request type is sent in the `RequestType` field in the vendor request object (p. 373) sent by AWS CloudFormation when the template developer creates, updates, or deletes a stack that contains a custom resource.

Each request type has a particular set of fields that are sent with the request, including an S3 URL for the response by the custom resource provider. The provider must respond to the S3 bucket with either a `SUCCESS` or `FAILED` result within one hour. After one hour, the request times out. Each result also has a particular set of fields expected by AWS CloudFormation.

This section provides information about the request and response fields, with examples, for each request type.

In This Section

Create

Custom resource provider requests with `RequestType` set to `"Create"` are sent when the template developer creates a stack that contains a custom resource.

Request

Create requests contain the following fields:

RequestType

Will be "Create".

RequestId

A unique ID for the request.

ResponseURL

The response URL identifies a pre-signed S3 bucket that receives responses from the custom resource provider to AWS CloudFormation.

ResourceType

The template developer-chosen resource type of the custom resource in the AWS CloudFormation template. Custom resource type names can be up to 60 characters long and can include alphanumeric and the following characters: `_@-`.

LogicalResourceId

The template developer-chosen name (logical ID) of the custom resource in the AWS CloudFormation template.

StackId

The Amazon Resource Name (ARN) that identifies the stack that contains the custom resource.

ResourceProperties

This field contains the contents of the `Properties` object sent by the template developer. Its contents are defined by the custom resource provider.

Example

```
{
   "RequestType" : "Create",
   "RequestId" : "unique id for this create request",
   "ResponseURL" : "pre-signed-url-for-create-response",
   "ResourceType" : "Custom::MyCustomResourceType",
   "LogicalResourceId" : "name of resource in template",
   "StackId" : "arn:aws:cloudformation:us-east-1:namespace:stack/stack-name/guid",
   "ResourceProperties" : {
      "key1" : "string",
      "key2" : [ "list" ],
      "key3" : { "key4" : "map" }
   }
}
```

Responses

Success

When the create request is successful, a response must be sent to the S3 bucket with the following fields:

Status

Must be "SUCCESS".

LogicalResourceId

The template developer-chosen name (logical ID) of the custom resource in the AWS CloudFormation template. This response value should be copied *verbatim* from the request.

RequestId

A unique ID for the request. This response value should be copied *verbatim* from the request.

StackId

The Amazon Resource Name (ARN) that identifies the stack that contains the custom resource. This response value should be copied *verbatim* from the request.

PhysicalResourceId

This value should be an identifier unique to the custom resource vendor, and can be up to 1 Kb in size. The value must be a non-empty string and must be identical for all responses for the same resource.

Data

Optional, custom resource provider-defined name-value pairs to send with the response. You can access the values provided here by name in the template with `Fn::GetAtt`.

Example

```
{
   "Status" : "SUCCESS",
   "LogicalResourceId" : "name of resource in template (copied from request)",
```

```
   "RequestId" : "unique id for this create request (copied from request)",
   "StackId" : "arn:aws:cloudformation:us-east-1:namespace:stack/stack-name/guid (copied
 from request)",
   "PhysicalResourceId" : "required vendor-defined physical id that is unique for that
 vendor",
   "Data" : {
      "keyThatCanBeUsedInGetAtt1" : "data for key 1",
      "keyThatCanBeUsedInGetAtt2" : "data for key 2"
   }
}
```

Failed

When the create request fails, a response must be sent to the S3 bucket with the following fields:

Status

Must be "FAILED".

Reason

Describes the reason for a failure response.

LogicalResourceId

The template developer-chosen name (logical ID) of the custom resource in the AWS CloudFormation template. This response value should be copied *verbatim* from the request.

RequestId

A unique ID for the request. This response value should be copied *verbatim* from the request.

StackId

The Amazon Resource Name (ARN) that identifies the stack that contains the custom resource. This response value should be copied *verbatim* from the request.

Example

```
{
   "Status" : "FAILED",
   "Reason" : "Required failure reason string",
   "LogicalResourceId" : "name of resource in template (copied from request)",
   "RequestId" : "unique id for this create request (copied from request)",
   "StackId" : "arn:aws:cloudformation:us-east-1:namespace:stack/stack-name/guid (copied
 from request)"
}
```

Delete

Custom resource provider requests with `RequestType` set to `"Delete"` are sent when the template developer deletes a stack that contains a custom resource. To successfully delete a stack with a custom resource, the custom resource provider must respond successfully to a delete request.

Request

Delete requests contain the following fields:

RequestType

Will be "Delete".

RequestId

A unique ID for the request.

ResourceType

The template developer-chosen resource type of the custom resource in the AWS CloudFormation template. Custom resource type names can be up to 60 characters long and can include alphanumeric and the following characters: `_@-`.

ResponseURL

The response URL identifies a pre-signed S3 bucket that receives responses from the custom resource provider to AWS CloudFormation.

LogicalResourceId

The template developer-chosen name (logical ID) of the custom resource in the AWS CloudFormation template.

StackId

The Amazon Resource Name (ARN) that identifies the stack that contains the custom resource.

PhysicalResourceId

A required custom resource provider-defined physical ID that is unique for that provider.

ResourceProperties

This field contains the contents of the `Properties` object sent by the template developer. Its contents are defined by the custom resource provider.

Example

```
{
   "RequestType" : "Delete",
   "RequestId" : "unique id for this delete request",
   "ResponseURL" : "pre-signed-url-for-delete-response",
   "StackId" : "arn:aws:cloudformation:us-east-1:namespace:stack/stack-name/guid",
   "ResourceType" : "Custom::MyCustomResourceType",
   "LogicalResourceId" : "name of resource in template",
   "PhysicalResourceId" : "custom resource provider-defined physical id",
   "ResourceProperties" : {
      "key1" : "string",
      "key2" : [ "list" ],
      "key3" : { "key4" : "map" }
   }
}
```

Responses

Success

When the delete request is successful, a response must be sent to the S3 bucket with the following fields:

Status

Must be "SUCCESS".

LogicalResourceId

The template developer-chosen name (logical ID) of the custom resource in the AWS CloudFormation template. This response value should be copied *verbatim* from the request.

RequestId

A unique ID for the request. This response value should be copied *verbatim* from the request.

StackId

The Amazon Resource Name (ARN) that identifies the stack that contains the custom resource. This response value should be copied *verbatim* from the request.

PhysicalResourceId

This value should be an identifier unique to the custom resource vendor, and can be up to 1 Kb in size. The value must be a non-empty string and must be identical for all responses for the same resource.

Example

```
{
   "Status" : "SUCCESS",
   "LogicalResourceId" : "name of resource in template (copied from request)",
   "RequestId" : "unique id for this delete request (copied from request)",
   "StackId" : "arn:aws:cloudformation:us-east-1:namespace:stack/stack-name/guid (copied
 from request)",
   "PhysicalResourceId" : "custom resource provider-defined physical id"
}
```

Failed

When the delete request fails, a response must be sent to the S3 bucket with the following fields:

Status

Must be "FAILED".

Reason

The reason for the failure.

LogicalResourceId

The `LogicalResourceId` value copied from the delete request (p. 378).

RequestId

The `RequestId` value copied from the delete request (p. 378).

StackId

The `StackId` value copied from the delete request (p. 378).

PhysicalResourceId

A required custom resource provider-defined physical ID that is unique for that provider.

Example

```
{
  "Status" : "FAILED",
  "Reason" : "Required failure reason string",
  "LogicalResourceId" : "name of resource in template (copied from request)",
  "RequestId" : "unique id for this delete request (copied from request)",
  "StackId" : "arn:aws:cloudformation:us-east-1:namespace:stack/stack-name/guid (copied
 from request)",
  "PhysicalResourceId" : "custom resource provider-defined physical id"
}
```

Update

Custom resource provider requests with `RequestType` set to `"Update"` are sent when the template developer updates a stack that contains a custom resource.

Request

Update requests contain the following fields:

RequestType

Will be "Update".

RequestId

A unique ID for the request.

ResponseURL

The response URL identifies a pre-signed S3 bucket that receives responses from the custom resource provider to AWS CloudFormation.

StackId

The Amazon Resource Name (ARN) that identifies the stack that contains the custom resource.

ResourceType

The template developer-chosen resource type of the custom resource in the AWS CloudFormation template. Custom resource type names can be up to 60 characters long and can include alphanumeric and the following characters: `_@-`. You cannot change the type during an update.

LogicalResourceId

The template developer-chosen name (logical ID) of the custom resource in the AWS CloudFormation template.

PhysicalResourceId

A required custom resource provider-defined physical ID that is unique for that provider.

ResourceProperties

The new resource property values declared by the template developer in the updated AWS CloudFormation template.

OldResourceProperties

The resource property values that were previously declared by the template developer in the AWS CloudFormation template.

Example

```
{
   "RequestType" : "Update",
   "RequestId" : "unique id for this update request",
   "ResponseURL" : "pre-signed-url-for-update-response",
   "StackId" : "arn:aws:cloudformation:us-east-1:namespace:stack/stack-name/guid",
   "ResourceType" : "Custom::MyCustomResourceType",
   "LogicalResourceId" : "name of resource in template",
   "PhysicalResourceId" : "custom resource provider-defined physical id",
   "ResourceProperties" : {
      "key1" : "new-string",
      "key2" : [ "new-list" ],
      "key3" : { "key4" : "new-map" }
```

```
   },
   "OldResourceProperties" : {
      "key1" : "string",
      "key2" : [ "list" ],
      "key3" : { "key4" : "map" }
   }
}
```

Responses

Success

If the custom resource provider is able to successfully update the resource, AWS CloudFormation expects status to be set to `"SUCCESS"` in the response.

Status

Must be "SUCCESS".

StackId

The Amazon Resource Name (ARN) that identifies the stack that contains the custom resource. This response value should be copied *verbatim* from the request.

RequestId

A unique ID for the request. This response value should be copied *verbatim* from the request.

LogicalResourceId

The template developer-chosen name (logical ID) of the custom resource in the AWS CloudFormation template. This response value should be copied *verbatim* from the request.

PhysicalResourceId

This value should be an identifier unique to the custom resource vendor, and can be up to 1 Kb in size. The value must be a non-empty string and must be identical for all responses for the same resource.

Data

Optional, custom resource provider-defined name-value pairs to send with the response. You can access the values provided here by name in the template with `Fn::GetAtt`.

Example

```
{
   "Status" : "SUCCESS",
   "StackId" : "arn:aws:cloudformation:us-east-1:namespace:stack/stack-name/guid (copied
 from request)",
   "RequestId" : "unique id for this update request (copied from request)",
   "LogicalResourceId" : "name of resource in template (copied from request)",
   "PhysicalResourceId" : "custom resource provider-defined physical id",
   "Data" : {
      "keyThatCanBeUsedInGetAtt1" : "data for key 1",
      "keyThatCanBeUsedInGetAtt2" : "data for key 2"
   }
}
```

Failed

If the resource cannot be updated with new set of properties, AWS CloudFormation expects the status to be set to "FAILED", along with a failure reason in the response.

Status

Must be "FAILED".

Reason

Describes the reason for a failure response.

LogicalResourceId

The template developer-chosen name (logical ID) of the custom resource in the AWS CloudFormation template. This response value should be copied *verbatim* from the request.

RequestId

A unique ID for the request. This response value should be copied *verbatim* from the request.

StackId

The Amazon Resource Name (ARN) that identifies the stack that contains the custom resource. This response value should be copied *verbatim* from the request.

PhysicalResourceId

This value should be an identifier unique to the custom resource vendor, and can be up to 1 Kb in size. The value must be a non-empty string and must be identical for all responses for the same resource.

Example

```
{
   "Status" : "FAILED",
   "Reason" : "Required failure reason string",
   "LogicalResourceId" : "name of resource in template (copied from request)",
   "RequestId" : "unique id for this update request (copied from request)",
   "StackId" : "arn:aws:cloudformation:us-east-1:namespace:stack/stack-name/guid (copied
 from request)",
   "PhysicalResourceId" : "custom resource provider-defined physical id"
}
```

Using Regular Expressions in AWS CloudFormation Templates

Regular expressions (commonly known as regexes) can be specified in a number of places within an AWS CloudFormation template, such as for the `AllowedPattern` property when creating a template parameter (p. 132).

Regular expressions in AWS CloudFormation conform to the Java regular expression syntax. A full description of this syntax and its constructs can be viewed in the Java documentation, here: java.util.regex.Pattern.

> **Important**
> Since AWS CloudFormation templates use the JSON syntax for specifying objects and data, you will need to add an additional backslash to any backslash characters in your regular expression, or JSON will interpret these as escape characters.
> For example, if you include a `\d` in your regular expression to match a digit character, you will need to write it as `\\d` in your template.

Using CloudFormer to Create AWS CloudFormation Templates from Existing AWS Resources

CloudFormer is a template creation beta tool that creates an AWS CloudFormation template from existing AWS resources in your account. You select any supported AWS resources that are running in your account, and CloudFormer creates a template in an Amazon S3 bucket.

> **Note**
> Use CloudFormer to produce templates that you can use as a starting point. Not all AWS resources are supported. .

The following list outlines the basic procedure for using CloudFormer:

1. Provision and configure the required resources using your existing processes and tools.
2. Create and launch a CloudFormer stack.

   CloudFormer is an AWS CloudFormation stack. You run CloudFormer by launching the stack from your AWS environment. It runs on a t2.medium Amazon EC2 instance and requires no other resources.
3. Use CloudFormer to create a template using any of your existing AWS resources and save it to an Amazon S3 bucket.
4. Shut down the CloudFormer stack.

   You usually don't need CloudFormer beyond this point, so you can avoid additional charges by shutting it down, which terminates the associated Amazon EC2 instance.
5. Use the template to launch the stack, as needed.

The following topics describes how to use CloudFormer by walking you through a basic scenario (a simple website on an Amazon EC2 instance) that creates a template with multiple resources. However, this example is just one of many possible scenarios; CloudFormer can create a template from any collection of AWS resources.

Topics

Step 1: Create a CloudFormer Stack

CloudFormer is itself an AWS CloudFormation stack, so the first step is to create and launch the stack from the AWS CloudFormation console.

To create a CloudFormer stack using the AWS CloudFormation Console

1. Log in to the AWS CloudFormation console and click **Create New Stack** to launch the stack creation wizard. For instructions on how to log in, see Logging in to the AWS CloudFormation Console.
2. In the **Choose a template** section, select **Select a sample template** and then select **CloudFormer** from the drop-down list.
3. Click **Next** to specify the stack name and input parameters.
4. Specify a name for the CloudFormer stack in the **Name** field.
5. In the **Parameters** section, type a password and user name that you'll use to log in to CloudFormer, and then click **Next**.

Important
You can't use special characters for the password (such as `; & ! " £ $ % ^ ( ) / \`) or leave the password blank.

6. Click **Next**.

   For CloudFormer, you don't need to specify any additional options.
7. Review the information about the stack and select **I acknowledge that this template may create IAM resources**.
8. After you finish reviewing the stack information, click **Create** to start creating the CloudFormer stack.

   CloudFormer is an AWS CloudFormation stack, so it must go through the normal stack creation process, which can take a few minutes.

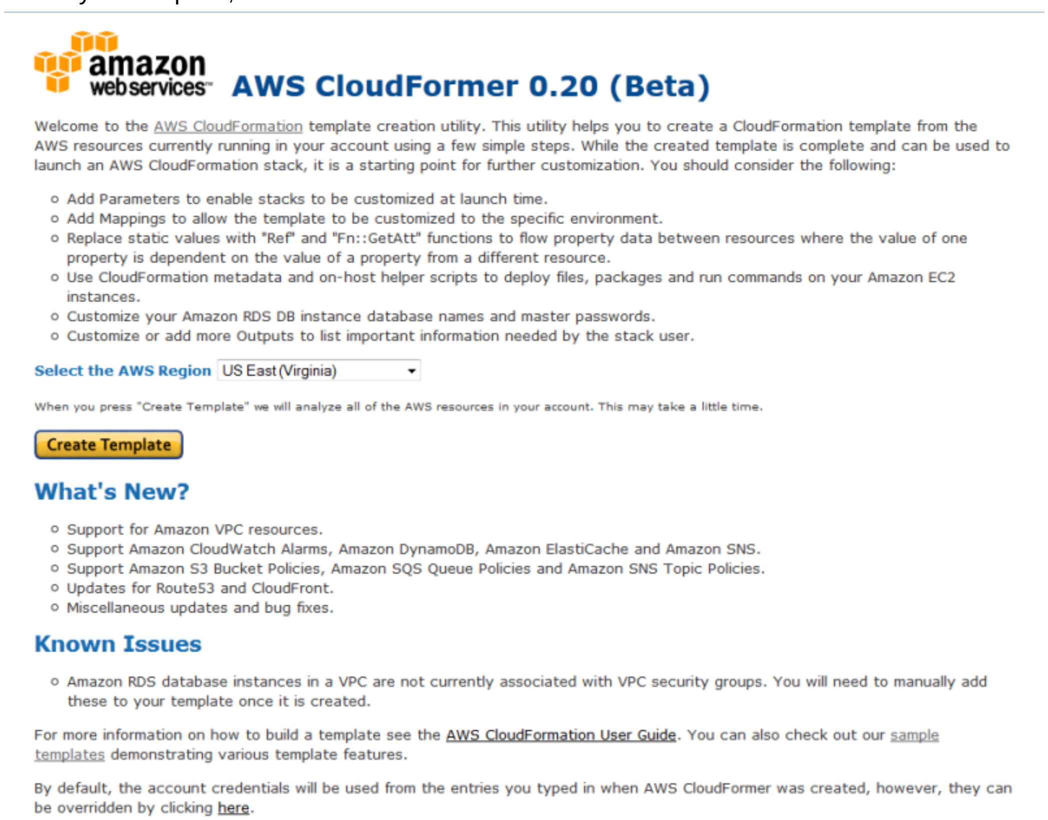
Step 2: Launch the CloudFormer Stack

After the CloudFormer stack's status is **CREATE_COMPLETE**, you can launch the stack.

To launch the CloudFormer stack

1. Click the CloudFormer stack's entry in the AWS CloudFormation Console, and select the **Outputs** tab in the stack information pane.
2. In the **Value** column, click the URL to launch the CloudFormer tool.
3. Type the user name and password that you specified when you created the CloudFormer stack.

When you log in to CloudFormer, it displays the first page of the tool in your browser, where you can start to create your template, as described in the next section.



Note
The CloudFormer stack launches a t2.medium Amazon EC2 instance, which you must manually terminate after you have finished.

After you create a CloudFormer stack, it becomes one of your account's collection of stacks. To create another template, just launch the CloudFormer stack again.

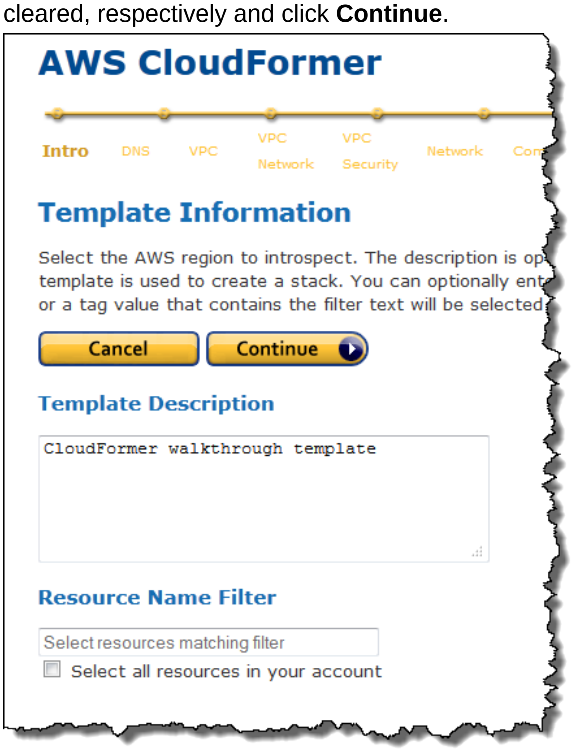
Step 3: Use CloudFormer to Create a Template

Before you start using CloudFormer to create a template, first ensure that your account has all the AWS resources that you want to include in your template. This walkthrough assumes that your account has:

- An Amazon EC2 instance (`AWS::EC2::Instance`).
- An Amazon EC2 security group (`AWS::EC2::SecurityGroup`). You should associate the security group with the instance.
- An Elastic IP Address(`AWS::EC2::EIP`). You should associate the address with the instance.

To use CloudFormer to create a template from your AWS resources

1. Under **Select the AWS Region**, select the template's region from the list, and click **Create Template**. The tool must first analyze your account, so it might take a few minutes before the **Intro** page is displayed.
2. On the **Intro** page, enter a description for your template. You can also use this page to select resources with a filter or select all resources in your account. This walkthrough specifies resources manually, so leave **Resource Name Filter** and **Select all resources in your account** blank and cleared, respectively and click **Continue**.

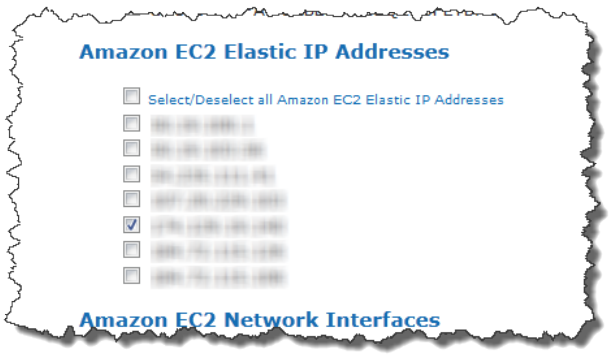


3. The following pages are for resources that are not used by this walkthrough, so just examine the page for future reference and click **Continue**. In order:

   1. **DNS Names** allows you to include Route 53 records.

   2. The **Virtual Private Clouds** allows you to include Amazon VPCs.
   3. **Virtual Private Cloud Network Topologies** allows you to include Amazon VPC subnets, gateways, DHCP configurations, and VPN connections.
   4. **Virtual Private Cloud Security Configuration** allows you to include network ACLS and route tables.

4. **Network Resources** allows you to include Elastic Load Balancing load balancers, Elastic IP Addresses, CloudFront distributions, and Amazon EC2 network interfaces. Select the Elastic IP address you want to include in the template.



5. The **Compute Resources** page allows you to include Auto Scaling groups and Amazon EC2 instances. Before you started creating the template, you associated an Elastic IP Address with your Amazon EC2 instance, creating a dependent resource. When you reach **Compute Resources**, CloudFormer automatically selects dependent instances, so just ensure that your instance is selected and click **Continue**.

   Note
   You can manually include additional instances, as needed. If you don't want to include an automatically selected instance, just clear the check box.

6. The following pages are for resources that are not used by this walkthrough, so just examine the page for future reference and click **Continue**. In order:

   1. **Storage** allows you to include Amazon EBS volumes, Amazon RDS instances, DynamoDB tables, and Amazon S3 buckets.
   2. **Application Services** allows you to include ElastiCache clusters, Amazon SQS queues, Amazon SimpleDB domains, and Amazon SNS topics.

      System Configuration allows you to include Auto Scaling launch configurations, Amazon RDS subnet groups, ElastiCache parameter groups, and Amazon RDS parameter groups.

7. The **Security Groups** page allows you include security groups. Before you started creating the template, you associated an Amazon EC2 security group with your Amazon EC2 instance, creating a dependent resource. When you reach **Security Groups**, CloudFormer automatically selects dependent security groups, so just ensure that your group is selected and click **Continue**.

Note

You can manually include additional security groups—including Amazon EC2 security groups, Amazon RDS security groups, and so on—as appropriate. If you don't want to include an automatically selected security group, just clear the check box.

8. The **Operational Resources** page allows you to include Auto Scaling policies and CloudWatch alarms. This walkthrough uses neither, so just click **Continue**.
9. The **Summary** page serves several purposes:

   - It allows you to review the resources you've added to your template.

     To modify your resources, click **Back** to return to the appropriate pages and modify your selections as needed.
   - It allows you to change your the auto-generated logical names that were assigned to your resources.

     To modify a logical name, click **Modify** and enter the name in the **Logical Name** field.
   - It allows you to specify outputs that provide necessary information, such as your site's IP address or URL.

     To modify an output, click **Modify** and select the appropriate output from the list.

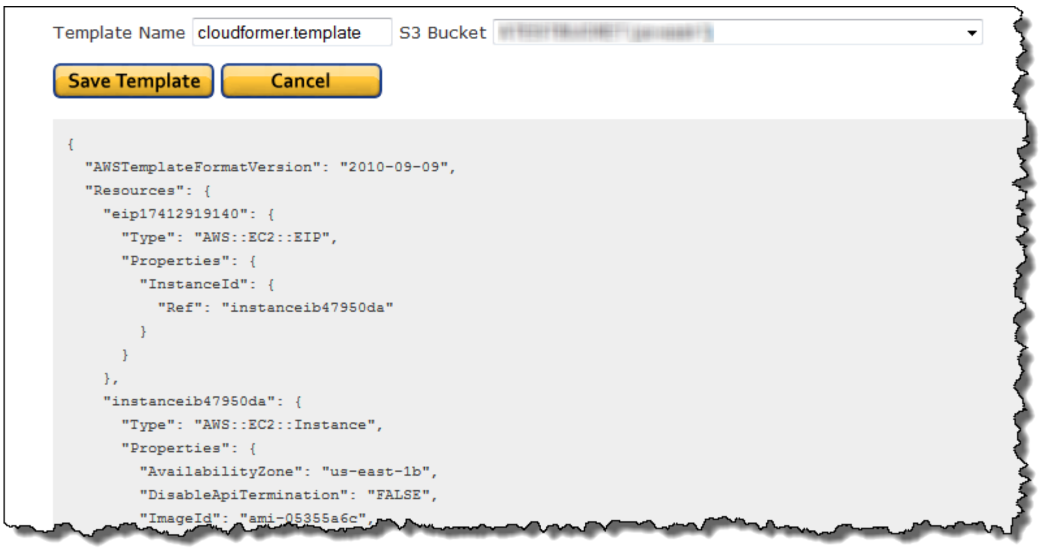
Examine the resources you've selected and make any necessary changes. You should have one Elastic IP Address, one Amazon EC2 instance, and one Amazon EC2 security group. When you are satisfied, click **Continue** to generate the template.

10. The **AWS CloudFormation Template** page displays the generated template. You can use the template to deploy your resources as a combined set with AWS CloudFormation, or as a base template for further modification.

    Note

    In addition to the resources that you explicitly specified, the template includes values that are associated with those resources such as Amazon EC2 instances' Availability Zones.

    Select an Amazon S3 bucket from the **S3 Bucket** list and click **Save Template** to save the template to the bucket and add it to your accounts collection of stacks.



    Save Template gives you two options:

    - **Launch Stack** saves the template to the specified Amazon S3 bucket and also launches the stack immediately.
    - **Create Template** simply saves the template to the specified Amazon S3 bucket.

      You can launch the stack later just like you would with any other template, for example, by using the AWS CloudFormation console.

11. Now that you have the template, you don't need the CloudFormer stack any more. To avoid unnecessary charges to your account, go to the Amazon EC2 console and delete the CloudFormer Amazon EC2 instance.

Template Reference

This section details the supported resources, type names, intrinsic functions and pseudo parameters used in AWS CloudFormation templates.

Topics

AWS Resource Types Reference

This section contains reference information for all AWS resources that are supported by AWS CloudFormation

Resource type identifiers always take the following form:

```
AWS::aws-product-name::data-type-name
```

Topics

AWS::ApiGateway::Account

The `AWS::ApiGateway::Account` resource specifies the AWS Identity and Access Management (IAM) role that Amazon API Gateway (API Gateway) uses to write API logs to Amazon CloudWatch Logs (CloudWatch Logs).

> **Important**
> If an API Gateway resource has never been created in your AWS account, you must add a dependency on another API Gateway resource, such as an AWS::ApiGateway::RestApi (p. 417) or AWS::ApiGateway::ApiKey (p. 397) resource.
> If an API Gateway resource has been created in your AWS account, no dependency is required (even if the resource was deleted).

Topics

Syntax

The syntax for declaring this resource:

JSON

```
{
  "Type" : "AWS::ApiGateway::Account",
  "Properties" : {
    "CloudWatchRoleArn": String
  }
}
```

YAML

```
Type: "AWS::ApiGateway::Account"
Properties:
  CloudWatchRoleArn: String
```

Properties

`CloudWatchRoleArn`

The Amazon Resource Name (ARN) of an IAM role that has write access to CloudWatch Logs in your account.

Required: No

Type: String

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the ID of the resource, such as `mysta-accou-01234b567890example`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates an IAM role that API Gateway can assume to push logs to CloudWatch Logs. The example associates the role with the `AWS::ApiGateway::Account` resource.

JSON

```
"CloudWatchRole": {
  "Type": "AWS::IAM::Role",
  "Properties": {
    "AssumeRolePolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [{
        "Effect": "Allow",
        "Principal": { "Service": [ "apigateway.amazonaws.com" ] },
        "Action": "sts:AssumeRole"
      }]
    },
    "Path": "/",
    "ManagedPolicyArns": ["arn:aws:iam::aws:policy/service-role/
AmazonAPIGatewayPushToCloudWatchLogs"]
  }
},
"Account": {
  "Type": "AWS::ApiGateway::Account",
  "Properties": {
    "CloudWatchRoleArn": { "Fn::GetAtt": ["CloudWatchRole", "Arn"] }
  }
}
```

YAML

```
CloudWatchRole:
 Type: "AWS::IAM::Role"
 Properties:
  AssumeRolePolicyDocument:
   Version: "2012-10-17"
   Statement:
    - Effect: Allow
      Principal:
       Service:
        - "apigateway.amazonaws.com"
      Action: "sts:AssumeRole"
  Path: "/"
  ManagedPolicyArns:
   - "arn:aws:iam::aws:policy/service-role/AmazonAPIGatewayPushToCloudWatchLogs"
Account:
 Type: "AWS::ApiGateway::Account"
```

```
Properties:
 CloudWatchRoleArn:
  "Fn::GetAtt":
   - CloudWatchRole
   - Arn
```

AWS::ApiGateway::ApiKey

The `AWS::ApiGateway::ApiKey` resource creates a unique key that you can distribute to clients who are executing Amazon API Gateway (API Gateway) `Method` resources that require an API key. To specify which API key clients must use, map the API key with the `RestApi` and `Stage` resources that include the methods requiring a key.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ApiGateway::ApiKey",
  "Properties" : {
    "Description" : String,
    "Enabled" : Boolean,
    "Name" : String,
    "StageKeys" : [ StageKey (p. 1010), ... ]
  }
}
```

YAML

```
Type: "AWS::ApiGateway::ApiKey"
Properties:
  Description: String
  Enabled: Boolean
  Name: String
  StageKeys:
    - StageKey (p. 1010)
    - ...
```

Properties

`Description`

A description of the purpose of the API key.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Enabled`

Indicates whether the API key can be used by clients.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`Name`

A name for the API key. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the API key name. For more information, see Name Type (p. 1217).

> **Important**
> If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

`StageKeys`

A list of stages to associated with this API key.

Required: No

Type: List of Amazon API Gateway ApiKey StageKey (p. 1010)

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the API key ID, such as `m2m1k7sybf`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates an API key and associates it with the `Test` stage of the `TestAPIDeployment` deployment. To ensure that AWS CloudFormation creates the stage and deployment (which are declared elsewhere in the same template) before the API key, the example adds an explicit dependency on the deployment and stage. Without this dependency, AWS CloudFormation might create the API key first, causing the association to fail because the deployment and stage wouldn't exist.

JSON

```
"ApiKey": {
  "Type": "AWS::ApiGateway::ApiKey",
```

```
  "DependsOn": ["TestAPIDeployment", "Test"],
  "Properties": {
    "Name": "TestApiKey",
    "Description": "CloudFormation API Key V1",
    "Enabled": "true",
    "StageKeys": [{
      "RestApiId": { "Ref": "RestApi" },
      "StageName": "Test"
    }]
  }
}
```

YAML

```
ApiKey:
  Type: "AWS::ApiGateway::ApiKey"
  DependsOn:
    - "TestAPIDeployment"
    - "Test"
  Properties:
    Name: "TestApiKey"
    Description: "CloudFormation API Key V1"
    Enabled: "true"
    StageKeys:
      - RestApiId:
          Ref: "RestApi"
        StageName: "Test"
```

AWS::ApiGateway::Authorizer

The `AWS::ApiGateway::Authorizer` resource creates an authorization layer that Amazon API Gateway (API Gateway) activates for methods that have authorization enabled. API Gateway activates the authorizer when a client calls those methods.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ApiGateway::Authorizer",
  "Properties" : {
    "AuthorizerCredentials" : String,
    "AuthorizerResultTtlInSeconds" : Integer,
    "AuthorizerUri" : String,
    "IdentitySource" : String,
    "IdentityValidationExpression" : String,
    "Name" : String,
    "ProviderARNs" : [ String, ... ],
    "RestApiId" : String,
```

```
    "Type" : String
  }
}
```

YAML

```
Type: "AWS::ApiGateway::Authorizer"
Properties:
  AuthorizerCredentials: String
  AuthorizerResultTtlInSeconds: Integer
  AuthorizerUri: String
  IdentitySource: String
  IdentityValidationExpression: String
  Name: String
  ProviderARNs:  - String
  RestApiId: String
  Type: String
```

Properties

`AuthorizerCredentials`

The credentials required for the authorizer. To specify an AWS Identity and Access Management (IAM) role that API Gateway assumes, specify the role's Amazon Resource Name (ARN). To use resource-based permissions on the AWS Lambda (Lambda) function, specify null.

Required: No

Type: String

Update requires: No interruption (p. 90)

`AuthorizerResultTtlInSeconds`

The time-to-live (TTL) period, in seconds, that specifies how long API Gateway caches authorizer results. If you specify a value greater than `0`, API Gateway caches the authorizer responses. By default, API Gateway sets this property to `300`. The maximum value is `3600`, or 1 hour.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`AuthorizerUri`

The authorizer's Uniform Resource Identifier (URI). If you specify `TOKEN` for the authorizer's `Type` property, specify a Lambda function URI, which has the form `arn:aws:apigateway:region:lambda:path/path`. The path usually has the form `/2015-03-31/functions/LambdaFunctionARN/invocations`.

Required: Conditional. Specify this property for Lambda functions only.

Type: String

Update requires: No interruption (p. 90)

`IdentitySource`

The source of the identity in an incoming request. If you specify `TOKEN` for the authorizer's `Type` property, specify a mapping expression. The custom header mapping expression has the form `method.request.header.name`, where `name` is the name of a custom authorization header that clients submit as part of their requests.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`IdentityValidationExpression`

A validation expression for the incoming identity. If you specify `TOKEN` for the authorizer's `Type` property, specify a regular expression. API Gateway uses the expression to attempt to match the incoming client token, and proceeds if the token matches. If the token doesn't match, API Gateway responds with a 401 (unauthorized request) error code.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Name`

The name of the authorizer.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`ProviderARNs`

A list of the Amazon Cognito user pool Amazon Resource Names (ARNs) to associate with this authorizer. For more information, see Use Amazon Cognito Your User Pool in the *API Gateway Developer Guide*.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`RestApiId`

The ID of the `RestApi` resource in which API Gateway creates the authorizer.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Type`

The type of authorizer:

- For a custom authorizer that uses a Lambda function, use `TOKEN`.
- For an authorizer that uses Amazon Cognito user pools, use `COGNITO_USER_POOLS`.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the authorizer's ID, such as `abcde1`.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

The following examples create a custom authorizer that is an AWS Lambda function.

JSON

```
"Authorizer": {
  "Type": "AWS::ApiGateway::Authorizer",
  "Properties": {
    "AuthorizerCredentials": { "Fn::GetAtt": ["LambdaInvocationRole", "Arn"] },
    "AuthorizerResultTtlInSeconds": "300",
    "AuthorizerUri" : {"Fn::Join" : ["", [
      "arn:aws:apigateway:",
      {"Ref" : "AWS::Region"},
      ":lambda:path/2015-03-31/functions/",
      {"Fn::GetAtt" : ["LambdaAuthorizer", "Arn"]}, "/invocations"
    ]]},
    "Type": "TOKEN",
    "IdentitySource": "method.request.header.Auth",
    "Name": "DefaultAuthorizer",
    "RestApiId": {
      "Ref": "RestApi"
    }
  }
}
```

YAML

```
Authorizer:
  Type: "AWS::ApiGateway::Authorizer"
  Properties:
    AuthorizerCredentials:
      Fn::GetAtt:
        - "LambdaInvocationRole"
        - "Arn"
    AuthorizerResultTtlInSeconds: "300"
    AuthorizerUri:
      Fn::Join:
        - ""
        -
          - "arn:aws:apigateway:"
          - Ref: "AWS::Region"
          - ":lambda:path/2015-03-31/functions/"
          - Fn::GetAtt:
              - "LambdaAuthorizer"
              - "Arn"
          - "/invocations"
    Type: "TOKEN"
```

```
    IdentitySource: "method.request.header.Auth"
    Name: "DefaultAuthorizer"
    RestApiId:
      Ref: "RestApi"
```

AWS::ApiGateway::BasePathMapping

The `AWS::ApiGateway::BasePathMapping` resource creates a base path that clients who call your Amazon API Gateway API must use in the invocation URL.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ApiGateway::BasePathMapping",
  "Properties" : {
    "BasePath" : String,
    "DomainName" : String,
    "RestApiId" : String,
    "Stage" : String
  }
}
```

YAML

```
Type: "AWS::ApiGateway::BasePathMapping"
Properties:
  BasePath: String
  DomainName: String
  RestApiId: String
  Stage: String
```

Properties

`BasePath`

The base path name that callers of the API must provide in the URL after the domain name.

Required: No

Type: String

Update requires: No interruption (p. 90)

`DomainName`

The name of a `DomainName` resource.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`RestApiId`

The name of the API.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`Stage`

The name of the API's stage.

Required: No

Type: String

Update requires: No interruption (p. 90)

AWS::ApiGateway::ClientCertificate

The `AWS::ApiGateway::ClientCertificate` resource creates a client certificate that Amazon API Gateway (API Gateway) uses to configure client-side SSL authentication for sending requests to the integration endpoint.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ApiGateway::ClientCertificate",
  "Properties" : {
    "Description" : String
  }
}
```

YAML

```
Type: "AWS::ApiGateway::ClientCertificate"
Properties:
  Description: String
```

Properties

`Description`

A description of the client certificate.

Required: No

Type: String

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the client certificate name, such as `abc123`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a client certificate that you can use with an API Gateway deployment and stage.

JSON

```
"TestClientCertificate": {
  "Type": "AWS::ApiGateway::ClientCertificate",
  "Properties": {
    "Description": "A test client certificate"
  }
}
```

YAML

```
TestClientCertificate:
  Type: "AWS::ApiGateway::ClientCertificate"
  Properties:
    Description: "A test client certificate"
```

AWS::ApiGateway::Deployment

The `AWS::ApiGateway::Deployment` resource deploys an Amazon API Gateway (API Gateway) `RestApi (p. 417)` resource to a stage so that clients can call the API over the Internet. The stage acts as an environment.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ApiGateway::Deployment",
  "Properties" : {
    "Description" : String,
    "RestApiId" : String,
    "StageDescription" : StageDescription (p. 1011),
    "StageName" : String
  }
}
```

YAML

```
Type: "AWS::ApiGateway::Deployment"
Properties:
  Description: String
  RestApiId: String
  StageDescription: StageDescription (p. 1011)
  StageName: String
```

Properties

`Description`

A description of the purpose of the API Gateway deployment.

Required: No

Type: String

Update requires: No interruption (p. 90)

`RestApiId`

The ID of the RestApi (p. 417) resource to deploy.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`StageDescription`

Configures the stage that API Gateway creates with this deployment.

Required: No

Type: Amazon API Gateway Deployment StageDescription (p. 1011)

Update requires: No interruption (p. 90)

`StageName`

A name for the stage that API Gateway creates with this deployment. Use only alphanumeric characters.

Required: No

Type: String

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the deployment ID, such as `123abc`.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

The following sections provide examples for declaring API Gateway deployments.

Deployment with an Empty Embedded Stage

The following example deploys the `MyApi` API to a stage named `DummyStage`.

JSON

```
"Deployment": {
  "Type": "AWS::ApiGateway::Deployment",
  "Properties": {
    "RestApiId": { "Ref": "MyApi" },
    "Description": "My deployment",
    "StageName": "DummyStage"
  }
}
```

YAML

```
Deployment:
  Type: "AWS::ApiGateway::Deployment"
  Properties:
    RestApiId:
      Ref: "MyApi"
    Description: "My deployment"
    StageName: "DummyStage"
```

`AWS::ApiGateway::Method` Dependency

If you create a `AWS::ApiGateway::RestApi` resource and its methods (using `AWS::ApiGateway::Method`) in the same template as your deployment, the deployment must depend on the `RestApi`'s methods. To create a dependency, add a `DependsOn` attribute to the deployment. If you don't, AWS CloudFormation creates the deployment right after it creates the `RestApi` resource that doesn't contain any methods, and AWS CloudFormation encounters the following error: `The REST API doesn't contain any methods`.

JSON

```
"Deployment": {
  "DependsOn": "MyMethod",
```

```
  "Type": "AWS::ApiGateway::Deployment",
  "Properties": {
    "RestApiId": { "Ref": "MyApi" },
    "Description": "My deployment",
    "StageName": "DummyStage"
  }
}
```

YAML

```
Deployment:
  DependsOn: "MyMethod"
  Type: "AWS::ApiGateway::Deployment"
  Properties:
    RestApiId:
      Ref: "MyApi"
    Description: "My deployment"
    StageName: "DummyStage"
```

AWS::ApiGateway::Method

The `AWS::ApiGateway::Method` resource creates Amazon API Gateway (API Gateway) methods that define the parameters and body that clients must send in their requests.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ApiGateway::Method",
  "Properties" : {
    "ApiKeyRequired" : Boolean,
    "AuthorizationType" : String,
    "AuthorizerId" : String,
    "HttpMethod" : String,
    "Integration" : Integration (p. 1016),
    "MethodResponses" : [ MethodResponse (p. 1020), ... ],
    "RequestModels" : { String:String, ... },
    "RequestParameters" : { String:Boolean, ... },
    "ResourceId" : String,
    "RestApiId" : String
  }
}
```

YAML

```
Type: "AWS::ApiGateway::Method"
Properties:
```

```
  ApiKeyRequired: Boolean
  AuthorizationType: String
  AuthorizerId: String
  HttpMethod: String
  Integration: Integration (p. 1016)
  MethodResponses:
    - MethodResponse (p. 1020)
  RequestModels:
    String: String
  RequestParameters:
    String: Boolean
  ResourceId: String
  RestApiId: String
```

Properties

`ApiKeyRequired`

Indicates whether the method requires clients to submit a valid API key.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`AuthorizationType`

The method's authorization type.

Required: Yes. If you specify the `AuthorizerId` property, specify `CUSTOM` for this property.

Type: String

Update requires: No interruption (p. 90)

`AuthorizerId`

The identifier of the authorizer (p. 399) to use on this method. If you specify this property, specify `CUSTOM` for the `AuthorizationType` property.

Required: No

Type: String

Update requires: No interruption (p. 90)

`HttpMethod`

The HTTP method that clients will use to call this method.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`Integration`

The back-end system that the method calls when it receives a request.

Required: No

Type: Amazon API Gateway Method Integration (p. 1016)

Update requires: No interruption (p. 90)

`MethodResponses`

The responses that can be sent to the client who calls the method.

Required: No

Type: List of Amazon API Gateway Method MethodResponse (p. 1020)

Update requires: No interruption (p. 90)

`RequestModels`

The resources used for the response's content type. Specify response models as key-value pairs (string-to-string map), with a content type as the key and a `Model` resource name as the value.

Required: No

Type: Mapping of key-value pairs

Update requires: No interruption (p. 90)

`RequestParameters`

Request parameters that API Gateway accepts. Specify request parameters as key-value pairs (string-to-Boolean map), with a source as the key and a Boolean as the value. The Boolean specifies whether a parameter is required. A source must match the following format `method.request.`*`location`*`.`*`name`*, where the *`location`* is `querystring`, `path`, or `header`, and *`name`* is a valid, unique parameter name.

Required: No

Type: Mapping of key-value pairs

Update requires: No interruption (p. 90)

`ResourceId`

The ID of an API Gateway resource (p. 415). For root resource methods, specify the RestApi root resource ID, such as `{ "Fn::GetAtt": ["MyRestApi", "RootResourceId"] }`.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`RestApiId`

The ID of the RestApi (p. 417) resource in which API Gateway creates the method.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the method ID, such as `mysta-metho-01234b567890example`.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

Mock Method

The following example creates a mock GET method for the `MyApi` API.

JSON

```
"MockMethod": {
  "Type": "AWS::ApiGateway::Method",
  "Properties": {
    "RestApiId": { "Ref": "MyApi" },
    "ResourceId": { "Fn::GetAtt": ["RestApi", "RootResourceId"] },
    "HttpMethod": "GET",
    "AuthorizationType": "NONE",
    "Integration": { "Type": "MOCK" }
  }
}
```

YAML

```
MockMethod:
  Type: "AWS::ApiGateway::Method"
  Properties:
    RestApiId:
      Ref: "MyApi"
    ResourceId:
      Fn::GetAtt:
        - "RestApi"
        - "RootResourceId"
    HttpMethod: "GET"
    AuthorizationType: "NONE"
    Integration:
      Type: "MOCK"
```

Lambda Proxy

The following example creates a proxy resource to enable clients to call a Lambda function with a single integration setup on a catch-all `ANY` method. The `Uri` property specifies the Lambda function. For more information about Lambda proxy integration and a sample Lambda function, see Create an API with Lambda Proxy Integration through a Proxy Resource in the *API Gateway Developer Guide*.

> **Note**
> Use the AWS::Lambda::Permission (p. 829) resource to grant API Gateway permission to invoke your Lambda function.

JSON

```
"ProxyResource": {
  "Type": "AWS::ApiGateway::Resource",
  "Properties": {
    "RestApiId": { "Ref":"LambdaSimpleProxy"},
    "ParentId": { "Fn::GetAtt" : [
      "LambdaSimpleProxy",
      "RootResourceId"
    ]},
    "PathPart": "{proxy+}"
```

```
  }
},
"ProxyResourceANY": {
  "Type": "AWS::ApiGateway::Method",
  "Properties": {
    "RestApiId": {"Ref":"LambdaSimpleProxy"},
    "ResourceId": {"Ref":"ProxyResource"},
    "HttpMethod": "ANY",
    "AuthorizationType": "NONE",
    "Integration": {
      "Type": "AWS_PROXY",
      "IntegrationHttpMethod": "POST",
      "Uri": { "Fn::Sub":"arn:aws:apigateway:${AWS::Region}:lambda:path/2015-03-31/
functions/${LambdaForSimpleProxy.Arn}/invocations"}
    }
  }
}
```

YAML

```
ProxyResource:
  Type: AWS::ApiGateway::Resource
  Properties:
    RestApiId: !Ref LambdaSimpleProxy
    ParentId: !GetAtt [LambdaSimpleProxy, RootResourceId]
    PathPart: '{proxy+}'
ProxyResourceANY:
  Type: AWS::ApiGateway::Method
  Properties:
    RestApiId: !Ref LambdaSimpleProxy
    ResourceId: !Ref ProxyResource
    HttpMethod: ANY
    AuthorizationType: NONE
    Integration:
      Type: AWS_PROXY
      IntegrationHttpMethod: POST
      Uri: !Sub arn:aws:apigateway:${AWS::Region}:lambda:path/2015-03-31/functions/
${LambdaForSimpleProxy.Arn}/invocations
```

AWS::ApiGateway::Model

The `AWS::ApiGateway::Model` resource defines the structure of a request or response payload for an Amazon API Gateway (API Gateway) method.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
```

```
  "Type" : "AWS::ApiGateway::Model",
  "Properties" : {
    "ContentType" : String,
    "Description" : String,
    "Name" : String,
    "RestApiId" : String,
    "Schema" : JSON object
  }
}
```

YAML

```
Type: "AWS::ApiGateway::Model"
Properties:
  ContentType: String
  Description: String
  Name: String
  RestApiId: String
  Schema: JSON object
```

Properties

`ContentType`

The content type for the model.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Description`

A description that identifies this model.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Name`

A name for the mode. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the model name. For more information, see Name Type (p. 1217).

Important
If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

`RestApiId`

The ID of a REST API with which to associate this model.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Schema`

The schema to use to transform data to one or more output formats. Specify null (`{}`) if you don't want to specify a schema.

Required: Yes

Type: JSON object

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the model name, such as `myModel`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a model that transforms input data into the described schema.

JSON

```
"PetsModelNoFlatten": {
  "Type": "AWS::ApiGateway::Model",
  "Properties": {
    "RestApiId": { "Ref": "RestApi" },
    "ContentType": "application/json",
    "Description": "Schema for Pets example",
    "Name": "PetsModelNoFlatten",
    "Schema": {
      "$schema": "http://json-schema.org/draft-04/schema#",
      "title": "PetsModelNoFlatten",
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "number": { "type": "integer" },
          "class": { "type": "string" },
          "salesPrice": { "type": "number" }
        }
      }
    }
  }
}
```

YAML

```
PetsModelNoFlatten:
```

```
  Type: "AWS::ApiGateway::Model"
  Properties:
    RestApiId:
      Ref: "RestApi"
    ContentType: "application/json"
    Description: "Schema for Pets example"
    Name: "PetsModelNoFlatten"
    Schema:
      Fn::Join:
        - ""
        -
          - "{"
          - "   \"$schema\": \"http://json-schema.org/draft-04/schema#\","
          - "   \"title\": \"PetsModelNoFlatten\","
          - "   \"type\": \"array\","
          - "   \"items\": {"
          - "       \"type\": \"object\","
          - "       \"properties\": {"
          - "           \"number\": { \"type\": \"integer\" },"
          - "           \"class\": { \"type\": \"string\" },"
          - "           \"salesPrice\": { \"type\": \"number\" }"
          - "       }"
          - "   }"
          - "}"
```

AWS::ApiGateway::Resource

The `AWS::ApiGateway::Resource` resource creates a resource in an Amazon API Gateway (API Gateway) API.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ApiGateway::Resource",
  "Properties" : {
    "ParentId" : String,
    "PathPart" : String,
    "RestApiId" : String
  }
}
```

YAML

```
Type: "AWS::ApiGateway::Resource"
Properties:
  ParentId: String
  PathPart: String
```

```
  RestApiId: String
```

Properties

`ParentId`

If you want to create a child resource, the ID of the parent resource. For resources without a parent, specify the RestApi root resource ID, such as `{ "Fn::GetAtt": ["MyRestApi", "RootResourceId"] }`.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`PathPart`

A path name for the resource.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`RestApiId`

The ID of the `RestApi` resource in which you want to create this resource.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource ID, such as `abc123`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a `stack` resource for the `MyApi` API.

JSON

```
"Stack": {
  "Type": "AWS::ApiGateway::Resource",
  "Properties": {
    "RestApiId": { "Ref": "MyApi" },
    "ParentId": { "Fn::GetAtt": ["MyApi", "RootResourceId"] },
    "PathPart": "stack"
  }
}
```

YAML

```
Stack:
  Type: "AWS::ApiGateway::Resource"
  Properties:
    RestApiId:
      Ref: "MyApi"
    ParentId:
      Fn::GetAtt:
        - "MyApi"
        - "RootResourceId"
    PathPart: "stack"
```

AWS::ApiGateway::RestApi

The `AWS::ApiGateway::RestApi` resource contains a collection of Amazon API Gateway (API Gateway) resources and methods that can be invoked through HTTPS endpoints.

> **Note**
> On January 1, 2016, the Swagger Specification was donated to the OpenAPI initiative, becoming the foundation of the OpenAPI Specification.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ApiGateway::RestApi",
  "Properties" : {
    "Body" : JSON object,
    "BodyS3Location" : S3Location (p. 1021),
    "CloneFrom" : String,
    "Description" : String,
    "FailOnWarnings" : Boolean,
    "Name" : String,
    "Parameters" : [ String, ... ]
  }
}
```

YAML

```
Type: "AWS::ApiGateway::RestApi"
Properties:
  Body: JSON object
  BodyS3Location: S3Location (p. 1021)
  CloneFrom: String
  Description: String
  FailOnWarnings: Boolean
  Name: String
```

```
Parameters:
  - String
```

Properties

`Body`

An OpenAPI specification that defines a set of RESTful APIs in the JSON format. For YAML templates, you can also specify the specification in the YAML format.

Required: No

Type: JSON object

Update requires: No interruption (p. 90)

`BodyS3Location`

The Amazon Simple Storage Service (Amazon S3) location that points to a OpenAPI file, which defines a set of RESTful APIs in JSON or YAML format.

Required: No

Type: Amazon API Gateway RestApi S3Location (p. 1021)

Update requires: No interruption (p. 90)

`CloneFrom`

The ID of the API Gateway `RestApi` resource that you want to clone.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Description`

A description of the purpose of this API Gateway `RestApi` resource.

Required: No

Type: String

Update requires: No interruption (p. 90)

`FailOnWarnings`

If a warning occurs while API Gateway is creating the `RestApi` resource, indicates whether to roll back the resource.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`Name`

A name for the API Gateway `RestApi` resource.

Required: Conditional. Required if you don't specify a OpenAPI definition.

Type: String

Update requires: No interruption (p. 90)

`Parameters`

Custom header parameters for the request.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the `RestApi` ID, such as `a1bcdef2gh`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attribute and a sample return value.

`RootResourceId`

The root resource ID for a `RestApi` resource, such as `a0bc123d4e`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Examples

The following example create an API Gateway `RestApi` resource.

JSON

```
"MyRestApi": {
  "Type": "AWS::ApiGateway::RestApi",
  "Properties": {
    "Body": {
      OpenAPI specification
    }
    "Description": "A test API",
    "Name": "MyRestAPI"
  }
}
```

YAML

```
MyRestApi:
  Type: "AWS::ApiGateway::RestApi"
  Properties:
    Body:
      OpenAPI specification
    Description: "A test API"
    Name: "MyRestAPI"
```

AWS::ApiGateway::Stage

The `AWS::ApiGateway::Stage` resource creates a stage for an Amazon API Gateway (API Gateway) deployment.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ApiGateway::Stage",
  "Properties" : {
    "CacheClusterEnabled" : Boolean,
    "CacheClusterSize" : String,
    "ClientCertificateId" : String,
    "DeploymentId" : String,
    "Description" : String,
    "MethodSettings" : [ MethodSetting (p. 1022) ],
    "RestApiId" : String,
    "StageName" : String,
    "Variables" : { String:String, ... }
  }
}
```

YAML

```
Type: "AWS::ApiGateway::Stage"
Properties:
  CacheClusterEnabled: Boolean
  CacheClusterSize: String
  ClientCertificateId: String
  DeploymentId: String
  Description: String
  MethodSettings:
    - MethodSetting (p. 1022)
  RestApiId: String
  StageName: String
  Variables:
    String: String
```

Properties

`CacheClusterEnabled`

Indicates whether cache clustering is enabled for the stage.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`CacheClusterSize`

The stage's cache cluster size.

Required: No

Type: String

Update requires: No interruption (p. 90)

`ClientCertificateId`

The identifier of the client certificate that API Gateway uses to call your integration endpoints in the stage.

Required: No

Type: String

Update requires: No interruption (p. 90)

`DeploymentId`

The ID of the deployment that the stage points to.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`Description`

A description of the stage's purpose.

Required: No

Type: String

Update requires: No interruption (p. 90)

`MethodSettings`

Settings for all methods in the stage.

Required: No

Type: Amazon API Gateway Stage MethodSetting (p. 1022)

Update requires: No interruption (p. 90)

`RestApiId`

The ID of the `RestApi` resource that you're deploying with this stage.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`StageName`

The name of the stage, which API Gateway uses as the first path segment in the invoke Uniform Resource Identifier (URI).

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Variables`

A map (string to string map) that defines the stage variables, where the variable name is the key and the variable value is the value. Variable names are limited to alphanumeric characters. Values must match the following regular expression: `[A-Za-z0-9-._~:/?#&=,]+`.

Required: No

Type: Mapping of key-value pairs

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the name of the stage, such as `MyTestStage`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a stage for the `TestDeployment` deployment. The stage also specifies method settings for the `MyRestApi` API.

JSON

```
"Prod": {
  "Type": "AWS::ApiGateway::Stage",
  "Properties": {
    "StageName": "Prod",
    "Description": "Prod Stage",
    "RestApiId": { "Ref": "MyRestApi" },
    "DeploymentId": { "Ref": "TestDeployment" },
    "ClientCertificateId": { "Ref": "ClientCertificate" },
    "Variables": { "Stack": "Prod" },
    "MethodSettings": [{
      "ResourcePath": "/",
      "HttpMethod": "GET",
      "MetricsEnabled": "true",
      "DataTraceEnabled": "true"
    }, {
      "ResourcePath": "/stack",
      "HttpMethod": "POST",
      "MetricsEnabled": "true",
      "DataTraceEnabled": "true",
      "ThrottlingBurstLimit": "999"
    }, {
      "ResourcePath": "/stack",
      "HttpMethod": "GET",
      "MetricsEnabled": "true",
      "DataTraceEnabled": "true",
      "ThrottlingBurstLimit": "555"
    }]
  }
}
```

YAML

```
Prod:
  Type: "AWS::ApiGateway::Stage"
  Properties:
    StageName: "Prod"
    Description: "Prod Stage"
    RestApiId:
      Ref: "MyRestApi"
    DeploymentId:
      Ref: "TestDeployment"
    ClientCertificateId:
      Ref: "ClientCertificate"
    Variables:
      Stack: "Prod"
    MethodSettings:
      -
        ResourcePath: "/"
        HttpMethod: "GET"
        MetricsEnabled: "true"
        DataTraceEnabled: "true"
      -
        ResourcePath: "/stack"
        HttpMethod: "POST"
        MetricsEnabled: "true"
        DataTraceEnabled: "true"
        ThrottlingBurstLimit: "999"
      -
        ResourcePath: "/stack"
        HttpMethod: "GET"
        MetricsEnabled: "true"
        DataTraceEnabled: "true"
        ThrottlingBurstLimit: "555"
```

AWS::ApiGateway::UsagePlan

The `AWS::ApiGateway::UsagePlan` resource specifies a usage plan for deployed Amazon API Gateway (API Gateway) APIs. A usage plan enforces throttling and quota limits on individual client API keys. For more information, see Creating and Using API Usage Plans in Amazon API Gateway in the *API Gateway Developer Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ApiGateway::UsagePlan",
  "Properties" : {
    "ApiStages" : [ ApiStage (p. 1024), ... ],
    "Description" : String,
```

```
    "Quota" : QuotaSettings (p. 1024),
    "Throttle" : ThrottleSettings (p. 1025),
    "UsagePlanName" : String
  }
}
```

YAML

```
Type: "AWS::ApiGateway::UsagePlan"
Properties:
  ApiStages:
  - ApiStage (p. 1024)
  Description: String
  Quota: QuotaSettings (p. 1024)
  Throttle: ThrottleSettings (p. 1025)
  UsagePlanName: String
```

Properties

`ApiStages`

The API stages to associate with this usage plan.

Required: No

Type: List of Amazon API Gateway UsagePlan ApiStage (p. 1024)

Update requires: No interruption (p. 90)

`Description`

The purpose of this usage plan.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Quota`

Configures the number of requests that users can make within a given interval.

Required: No

Type: Amazon API Gateway UsagePlan QuotaSettings (p. 1024)

Update requires: No interruption (p. 90)

`Throttle`

Configures the overall request rate (average requests per second) and burst capacity.

Required: No

Type: Amazon API Gateway UsagePlan ThrottleSettings (p. 1025)

Update requires: No interruption (p. 90)

`UsagePlanName`

A name for this usage plan.

Required: No

Type: String

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the usage plan ID, such as `MyUsagePlan`.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

The following examples create a usage plan for the `Prod` API stage, with a quota of 5000 requests per month and a rate limit of 100 requests per second.

JSON

```
"usagePlan" : {
  "Type" : "AWS::ApiGateway::UsagePlan",
  "Properties" : {
    "ApiStages" : [ {"ApiId" : { "Ref" : "MyRestApi" }, "Stage" : { "Ref" : "Prod" }} ],
    "Description" : "Customer ABC's usage plan",
    "Quota" : {
      "Limit" : 5000,
      "Period" : "MONTH"
    },
    "Throttle" : {
      "BurstLimit" : 200,
      "RateLimit" : 100
    },
    "UsagePlanName" : "Plan_ABC"
  }
}
```

YAML

```
usagePlan:
  Type: AWS::ApiGateway::UsagePlan
  Properties:
    ApiStages:
    - ApiId: !Ref 'MyRestApi'
      Stage: !Ref 'Prod'
    Description: Customer ABC's usage plan
    Quota:
      Limit: 5000
      Period: MONTH
    Throttle:
      BurstLimit: 200
      RateLimit: 100
    UsagePlanName: Plan_ABC
```

AWS::ApiGateway::UsagePlanKey

The `AWS::ApiGateway::UsagePlanKey` resource associates an Amazon API Gateway API key with an API Gateway usage plan. This association determines which users the usage plan is applied to.

Topics

Syntax

JSON

```
{
  "Type" : "AWS::ApiGateway::UsagePlanKey",
  "Properties" : {
    "KeyId" : String,
    "KeyType" : String,
    "UsagePlanId" : String
  }
}
```

YAML

```
Type: "AWS::ApiGateway::UsagePlanKey"
Properties:
  KeyId: String
  KeyType: String
  UsagePlanId: String
```

Properties

`KeyId`

The ID of the usage plan key.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`KeyType`

The type of usage plan key. Currently, the valid key type is `API_KEY`.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`UsagePlanId`

The value of the usage plan key.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "MyProfile" }
```

For the `IAM::InstanceProfile` with the logical ID `MyProfile`, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

JSON

```
“usagePlanKey” : {
  "Type": "AWS::ApiGateway::UsagePlanKey",
  "Properties": {
    "KeyId" : {"Ref" : "myApiKey"},
    "KeyType" : "API_KEY",
    "UsagePlanId" : {"Ref" : "myUsagePlan"}
  }
}
```

YAML

```
usagePlanKey:
  Type: "AWS::ApiGateway::UsagePlanKey"
  Properties :
    KeyId: !Ref 'myApiKey'
    KeyType: API_KEY
    UsagePlanId: !Ref 'myUsagePlan'
```

AWS::ApplicationAutoScaling::ScalableTarget

The `AWS::ApplicationAutoScaling::ScalableTarget` resource specifies a resource that Application Auto Scaling can scale up or down. For more information, see the RegisterScalableTarget action in the *Application Auto Scaling API Reference*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ApplicationAutoScaling::ScalableTarget",
  "Properties" : {
    "MaxCapacity" : Integer,
    "MinCapacity" : Integer,
    "ResourceId" : String,
    "RoleARN" : String,
    "ScalableDimension" : String,
    "ServiceNamespace" : String
  }
}
```

YAML

```
Type: "AWS::ApplicationAutoScaling::ScalableTarget"
Properties:
  MaxCapacity: Integer
  MinCapacity: Integer
  ResourceId: String
  RoleARN: String
  ScalableDimension: String
  ServiceNamespace: String
```

Properties

`MaxCapacity`

The maximum value that Application Auto Scaling can use to scale a target during a scaling activity.

Required: Yes

Type: Integer

Update requires: No interruption (p. 90)

`MinCapacity`

The minimum value that Application Auto Scaling can use to scale a target during a scaling activity.

Required: Yes

Type: Integer

Update requires: No interruption (p. 90)

`ResourceId`

The unique resource identifier to associate with this scalable target. For more information, see the `ResourceId` parameter for the RegisterScalableTarget action in the *Application Auto Scaling API Reference*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`RoleARN`

The Amazon Resource Name (ARN) of an AWS Identity and Access Management (IAM) role that allows Application Auto Scaling to modify your scalable target.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`ScalableDimension`

The scalable dimension associated with the scalable target. Specify the service namespace, resource type, and scaling property, such as `ecs:service:DesiredCount` for the desired task count of an Amazon EC2 Container Service service. For valid values, see the `ScalableDimension` content for the ScalingPolicy data type in the *Application Auto Scaling API Reference*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`ServiceNamespace`

The AWS service namespace of the scalable target. For a list of service namespaces, see AWS Service Namespaces in the *AWS General Reference*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the AWS CloudFormation-generated ID of the resource, such as `service/ecsStack-MyECSCluster-AB12CDE3F4GH/ecsStack-MyECSService-AB12CDE3F4GH|ecs:service:DesiredCount|ecs`.

AWS CloudFormation uses the following format to generate the ID: `service/`*`resource_ID (p. 428)`*`|`*`scalable_dimension`*`|`*`service_namespace`*.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a scalable target for an Amazon EC2 Container Service service. Application Auto Scaling scales the number of tasks at a minimum of 1 task and a maximum of 2.

JSON

```
"scalableTarget" : {
  "Type" : "AWS::ApplicationAutoScaling::ScalableTarget",
  "Properties" : {
    "MaxCapacity" : 2,
    "MinCapacity" : 1,
    "ResourceId" : "service/ecsStack-MyECSCluster-AB12CDE3F4GH/ecsStack-MyECSService-
AB12CDE3F4GH",
    "RoleARN" : {"Fn::GetAtt" : ["ApplicationAutoScalingRole", "Arn"] },
    "ScalableDimension" : "ecs:service:DesiredCount",
    "ServiceNamespace" : "ecs"
```

```
  }
}
```

YAML

```
scalableTarget:
  Type: AWS::ApplicationAutoScaling::ScalableTarget
  Properties:
    MaxCapacity: 2
    MinCapacity: 1
    ResourceId: service/ecsStack-MyECSCluster-AB12CDE3F4GH/ecsStack-MyECSService-
AB12CDE3F4GH
    RoleARN: !GetAtt [ ApplicationAutoScalingRole, Arn ]
    ScalableDimension: ecs:service:DesiredCount
    ServiceNamespace: ecs
```

AWS::ApplicationAutoScaling::ScalingPolicy

The `AWS::ApplicationAutoScaling::ScalingPolicy` resource defines an Application Auto Scaling scaling policy that Application Auto Scaling uses to adjust your application resources.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ApplicationAutoScaling::ScalingPolicy",
  "Properties" : {
    "PolicyName" : String,
    "PolicyType" : String,
    "ResourceId" : String,
    "ScalableDimension" : String,
    "ScalingTargetId" : String,
    "ServiceNamespace" : String,
    "StepScalingPolicyConfiguration" : StepScalingPolicyConfiguration (p. 1026)
  }
}
```

YAML

```
Type : "AWS::ApplicationAutoScaling::ScalingPolicy"
Properties:
  PolicyName: String
  PolicyType: String
  ResourceId: String
  ScalableDimension: String
  ScalingTargetId: String
  ServiceNamespace: String
```

```
  StepScalingPolicyConfiguration: StepScalingPolicyConfiguration (p. 1026)
```

Properties

`PolicyName`

A name for the scaling policy.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`PolicyType`

An Application Auto Scaling policy type. For valid values, see the `PolicyType` parameter for the PutScalingPolicy action in the *Application Auto Scaling API Reference*.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`ResourceId`

The unique resource identifier for the scalable target that this scaling policy applies to. For more information, see the `ResourceId` parameter for the PutScalingPolicy action in the *Application Auto Scaling API Reference*.

Required: Conditional. You must specify either the `ScalingTargetId` property or the `ResourceId`, `ScalableDimension`, and `ServiceNamespace` properties. If you specify the `ResourceId`, `ScalableDimension`, and `ServiceNamespace` properties, don't specify the `ScalingTargetId` property.

Type: String

Update requires: Replacement (p. 90)

`ScalableDimension`

The scalable dimension of the scalable target that this scaling policy applies to. The scalable dimension contains the service namespace, resource type, and scaling property, such as ecs:service:DesiredCount for the desired task count of an Amazon ECS service.

Required: Conditional. You must specify either the `ScalingTargetId` property or the `ResourceId`, `ScalableDimension`, and `ServiceNamespace` properties. If you specify the `ResourceId`, `ScalableDimension`, and `ServiceNamespace` properties, don't specify the `ScalingTargetId` property.

Type: String

Update requires: Replacement (p. 90)

`ServiceNamespace`

The AWS service namespace of the scalable target that this scaling policy applies to. For a list of service namespaces, see AWS Service Namespaces in the *AWS General Reference*.

Required: Conditional. You must specify either the `ScalingTargetId` property or the `ResourceId`, `ScalableDimension`, and `ServiceNamespace` properties. If you specify the `ResourceId`, `ScalableDimension`, and `ServiceNamespace` properties, don't specify the `ScalingTargetId` property.

Type: String

Update requires: Replacement (p. 90)

`ScalingTargetId`

The AWS CloudFormation-generated ID of an Application Auto Scaling scalable target. For more information about the ID, see the Return Value section of the AWS::ApplicationAutoScaling::ScalableTarget (p. 427) resource.

Required: Conditional. You must specify either the `ScalingTargetId` property or the `ResourceId`, `ScalableDimension`, and `ServiceNamespace` properties. If you specify this property, don't specify the `ResourceId`, `ScalableDimension`, and `ServiceNamespace` properties.

Type: String

Update requires: Replacement (p. 90)

`StepScalingPolicyConfiguration`

A step policy that configures when Application Auto Scaling scales resources up or down, and by how much.

Required: No

Type: Application Auto Scaling ScalingPolicy StepScalingPolicyConfiguration (p. 1026)

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the Application Auto Scaling scaling policy Amazon Resource Name (ARN), such as `arn:aws:autoscaling:us-east-1:123456789012:scalingPolicy:12ab3c4d-56789-0ef1-2345-6ghi7jk8lm90:resource/ecs/service/ecsStack-MyECSCluster-AB12CDE3F4GH/ecsStack-MyECSService-AB12CDE3F4GH:policyName/MyStepPolicy`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates an Application Auto Scaling scaling policy with a step policy configuration. When an associated alarm is breached, the policy increases the desired count of the scalable target by 200%, with a cooldown period of 60 seconds.

JSON

```
"scalingPolicy" : {
  "Type" : "AWS::ApplicationAutoScaling::ScalingPolicy",
  "Properties" : {
    "PolicyName" : "AStepPolicy",
    "PolicyType" : "StepScaling",
    "ScalingTargetId" : {"Ref": "scalableTarget"},
    "StepScalingPolicyConfiguration" : {
      "AdjustmentType" : "PercentChangeInCapacity",
      "Cooldown" : 60,
      "MetricAggregationType" : "Average",
      "StepAdjustments" : [{
        "MetricIntervalLowerBound" : 0,
        "ScalingAdjustment" : 200
```

```
      }]
    }
  }
}
```

YAML

```
scalingPolicy:
  Type: AWS::ApplicationAutoScaling::ScalingPolicy
  Properties:
    PolicyName: AStepPolicy
    PolicyType: StepScaling
    ScalingTargetId:
      Ref: scalableTarget
    StepScalingPolicyConfiguration:
      AdjustmentType: PercentChangeInCapacity
      Cooldown: 60
      MetricAggregationType: Average
      StepAdjustments:
      - MetricIntervalLowerBound: 0
        ScalingAdjustment: 200
```

AWS::AutoScaling::AutoScalingGroup

The `AWS::AutoScaling::AutoScalingGroup` type creates an Auto Scaling group.

You can add an UpdatePolicy (p. 1303) attribute to your Auto Scaling group to control how rolling updates are performed when a change has been made to the Auto Scaling group's launch configuration (p. 440) or subnet group membership (p. 437).

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::AutoScaling::AutoScalingGroup",
   "Properties" : {
      "AvailabilityZones (p. 434)" : [ String, ... ],
      "Cooldown (p. 434)" : String,
      "DesiredCapacity (p. 435)" : String,
      "HealthCheckGracePeriod (p. 435)" : Integer,
      "HealthCheckType (p. 435)" : String,
      "InstanceId (p. 435)" : String,
      "LaunchConfigurationName (p. 435)" : String,
      "LoadBalancerNames (p. 436)" : [ String, ... ],
      "MaxSize (p. 436)" : String,
```

```
      "MetricsCollection" : [ MetricsCollection, ... ],
      "MinSize (p. 436)" : String,
      "NotificationConfigurations" : [ NotificationConfigurations, ... ],
      "PlacementGroup" : String,
      "Tags (p. 437)" : [ Auto Scaling Tag, ... ],
      "TargetGroupARNs" : [ String, ... ],
      "TerminationPolicies" : [ String, ..., ],
      "VPCZoneIdentifier (p. 437)" : [ String, ... ]
   }
}
```

YAML

```
Type: "AWS::AutoScaling::AutoScalingGroup"
Properties:
  AvailabilityZones (p. 434):
    - String
  Cooldown (p. 434): String
  DesiredCapacity (p. 435): String
  HealthCheckGracePeriod (p. 435): Integer
  HealthCheckType (p. 435): String
  InstanceId (p. 435): String
  LaunchConfigurationName (p. 435): String
  LoadBalancerNames (p. 436):
    - String
  MaxSize (p. 436): String
  MetricsCollection:
    - MetricsCollection
  MinSize (p. 436): String
  NotificationConfigurations:
    - NotificationConfigurations
  PlacementGroup: String
  Tags (p. 437):
    - Auto Scaling Tag
  TargetGroupARNs:
    - String
  TerminationPolicies:
    - String
  VPCZoneIdentifier (p. 437):
    - String
```

Properties

`AvailabilityZones`

Contains a list of availability zones for the group.

Required: Conditional. If you don't specify the `VPCZoneIdentifier` property, you must specify this property.

Type: List of strings

Update requires: No interruption (p. 90)

`Cooldown`

The number of seconds after a scaling activity is completed before any further scaling activities can start.

Required: No

Type: String

Update requires: No interruption (p. 90)

`DesiredCapacity`

Specifies the desired capacity for the Auto Scaling group.

If `SpotPrice` is not set in the AWS::AutoScaling::LaunchConfiguration (p. 440) for this Auto Scaling group, then Auto Scaling will begin to bring instances online based on `DesiredCapacity`. CloudFormation will not mark the Auto Scaling group as successful (by setting its status to CREATE_COMPLETE) until the desired capacity is reached.

If `SpotPrice` *is* set, then `DesiredCapacity` will not be used as a criteria for success, since instances will only be started when the spot price has been matched. After the spot price has been matched, however, Auto Scaling uses `DesiredCapacity` as the target capacity for the group.

Required: No

Type: String

Update requires: No interruption (p. 90)

`HealthCheckGracePeriod`

The length of time in seconds after a new EC2 instance comes into service that Auto Scaling starts checking its health.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`HealthCheckType`

The service you want the health status from, Amazon EC2 or Elastic Load Balancer. Valid values are `EC2` or `ELB`.

Required: No

Type: String

Update requires: No interruption (p. 90)

`InstanceId`

The ID of the Amazon EC2 instance you want to use to create the Auto Scaling group. Use this property if you want to create an Auto Scaling group that uses an existing Amazon EC2 instance instead of a launch configuration.

When you use an Amazon EC2 instance to create an Auto Scaling group, a new launch configuration is first created and then associated with the Auto Scaling group. The new launch configuration derives all its properties from the instance, with the exception of `BlockDeviceMapping` and `AssociatePublicIpAddress`.

Required: Conditional. You must specify this property if you don't specify the `LaunchConfigurationName` property.

Type: String

Update requires: Replacement (p. 90)

`LaunchConfigurationName`

Specifies the name of the associated AWS::AutoScaling::LaunchConfiguration (p. 440).

Note

If this resource has a public IP address and is also in a VPC that is defined in the same template, you must use the `DependsOn` attribute to declare a dependency on the VPC-gateway attachment. For more information, see DependsOn Attribute (p. 1298).

Required: Conditional; you must specify this property if you don't specify the `InstanceId` property.

Type: String

Update requires: No interruption (p. 90)

Important

When you update the `LaunchConfigurationName`, existing Amazon EC2 instances continue to run with the configuration that they were originally launched with. To update existing instances, specify an update policy attribute for this Auto Scaling group. For more information, see UpdatePolicy (p. 1303).

`LoadBalancerNames`

A list of Classic load balancers associated with this Auto Scaling group. To specify Application load balancers, use `TargetGroupARNs`.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`MaxSize`

The maximum size of the Auto Scaling group.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`MetricsCollection`

Enables the monitoring of group metrics of an Auto Scaling group.

Required: No

Type: A list of Auto Scaling MetricsCollection (p. 1031)

Update requires: No interruption (p. 90)

`MinSize`

The minimum size of the Auto Scaling group.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`NotificationConfigurations`

An embedded property that configures an Auto Scaling group to send notifications when specified events take place.

Required: No

Type: List of Auto Scaling NotificationConfigurations (p. 1032)

Update requires: No interruption (p. 90)

`PlacementGroup`

The name of an existing cluster placement group into which you want to launch your instances. A placement group is a logical grouping of instances within a single Availability Zone. You cannot specify multiple Availability Zones and a placement group.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Tags`

The tags you want to attach to this resource.

For more information about tags, go to Tagging Auto Scaling Groups and Amazon EC2 Instances in the *Auto Scaling User Guide*.

Required: No

Type: List of Auto Scaling Tags (p. 1034)

Update requires: No interruption (p. 90)

`TargetGroupARNs`

A list of Amazon Resource Names (ARN) of target groups to associate with the Auto Scaling group.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`TerminationPolicies`

A policy or a list of policies that are used to select the instances to terminate. The policies are executed in the order that you list them.

For more information on configuring a termination policy for your Auto Scaling group, see Instance Termination Policy for Your Auto Scaling Group in the *Auto Scaling User Guide*.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`VPCZoneIdentifier`

A list of subnet identifiers of Amazon Virtual Private Cloud (Amazon VPCs).

If you specify the `AvailabilityZones` property, the subnets that you specify for this property must reside in those Availability Zones.

For more information, go to Using EC2 Dedicated Instances Within Your VPC in the *Auto Scaling User Guide*.

Required: Conditional. If you don't specify the `AvailabilityZones` property, you must specify this property.

Type: List of strings

Update requires: Some interruptions (p. 90)

Note
When you update VPCZoneIdentifier, the instances are replaced, but not the Auto Scaling group.

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

In the following sample, the `Ref` function returns the name of the `MyASGroup` Auto Scaling group, such as `mystack-myasgroup-NT5EUXTNTXXD`.

```
{ "Ref": "MyASGroup" }
```

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

To view more Auto Scaling examples, see Auto Scaling Template Snippets (p. 239).

Auto Scaling Group with an Elastic Load Balancing Load Balancer, Launch Configuration, and Metric Collection

JSON

```
"WebServerGroup" : {
   "Type" : "AWS::AutoScaling::AutoScalingGroup",
   "Properties" : {
      "AvailabilityZones" : { "Fn::GetAZs" : "" },
      "LaunchConfigurationName" : { "Ref" : "LaunchConfig" },
      "MinSize" : "2",
      "MaxSize" : "2",
      "LoadBalancerNames" : [ { "Ref" : "ElasticLoadBalancer" } ],
      "MetricsCollection": [
         {
            "Granularity": "1Minute",
            "Metrics": [
               "GroupMinSize",
               "GroupMaxSize"
            ]
         }
      ]
   }
}
```

YAML

```
WebServerGroup:
  Type: "AWS::AutoScaling::AutoScalingGroup"
  Properties:
    AvailabilityZones:
      Fn::GetAZs: ""
    LaunchConfigurationName:
      Ref: "LaunchConfig"
    MinSize: "2"
    MaxSize: "2"
    LoadBalancerNames:
      - Ref: "ElasticLoadBalancer"
    MetricsCollection:
```

```
      -
        Granularity: "1Minute"
        Metrics:
          - "GroupMinSize"
          - "GroupMaxSize"
```

Batch Update Instances in an Auto Scaling Group

The following example shows how to configure updates by including an UpdatePolicy (p. 1303) attribute. The attribute contains an `AutoScalingRollingUpdate` embedded object with three attributes that specify the update policy settings.

```
"ASG1" : {
   "UpdatePolicy" : {
      "AutoScalingRollingUpdate" : {
         "MinInstancesInService" : "1",
         "MaxBatchSize" : "1",
         "PauseTime" : "PT12M5S"
      }
   },
   "Type" : "AWS::AutoScaling::AutoScalingGroup",
   "Properties" : {
      "AvailabilityZones" : { "Fn::GetAZs" : { "Ref" : "AWS::Region" } },
      "LaunchConfigurationName" : { "Ref" : "ASLC" },
      "MaxSize" : "3",
      "MinSize" : "1"
   }
}
```

Auto Scaling Group Wait on Signals From New Instances

In the following example, the Auto Scaling group waits for new Amazon EC2 instances to signal the group before Auto Scaling proceeds to update the next batch of instances. In the UpdatePolicy (p. 1303) attribute, the `WaitOnResourceSignals` flag is set to `true`. You can use the cfn-signal (p. 1358) helper script on each instance to signal the Auto Scaling group.

JSON

```
"ASG1" : {
   "UpdatePolicy" : {
      "AutoScalingRollingUpdate" : {
         "MinInstancesInService" : "1",
         "MaxBatchSize" : "1",
         "PauseTime" : "PT12M5S",
         "WaitOnResourceSignals" : "true"
      }
   },
   "Type" : "AWS::AutoScaling::AutoScalingGroup",
   "Properties" : {
      "AvailabilityZones" : { "Fn::GetAZs" : { "Ref" : "AWS::Region" } },
      "LaunchConfigurationName" : { "Ref" : "ASLC" },
      "MaxSize" : "3",
      "MinSize" : "1"
   }
}
```

YAML

```
ASG1:
```

```
UpdatePolicy:
  AutoScalingRollingUpdate:
    MinInstancesInService: "1"
    MaxBatchSize: "1"
    PauseTime: "PT12M5S"
    WaitOnResourceSignals: "true"
Type: "AWS::AutoScaling::AutoScalingGroup"
Properties:
  AvailabilityZones:
    Fn::GetAZs:
      Ref: "AWS::Region"
  LaunchConfigurationName:
    Ref: "ASLC"
  MaxSize: "3"
  MinSize: "1"
```

See Also

- UpdatePolicy (p. 1303)
- UpdateAutoScalingGroup in the *Auto Scaling API Reference*
- AWS CloudFormation Stacks Updates (p. 89)

AWS::AutoScaling::LaunchConfiguration

The `AWS::AutoScaling::LaunchConfiguration` type creates an Auto Scaling launch configuration that can be used by an Auto Scaling group to configure Amazon EC2 instances in the Auto Scaling group.

Important
When you update a property of the `LaunchConfiguration` resource, AWS CloudFormation deletes that resource and creates a new launch configuration with the updated properties and a new name. This update action does not deploy any change across the running Amazon EC2 instances in the auto scaling group. In other words, an update simply replaces the `LaunchConfiguration` so that when the auto scaling group launches new instances, they will get the updated configuration, but existing instances continue to run with the configuration that they were originally launched with. This works the same way as if you made similar changes manually to an auto scaling group.
If you want to update existing instances when you update the `LaunchConfiguration` resource, you must specify an update policy attribute for the `AWS::AutoScaling::AutoScalingGroup` resource. For more information, see UpdatePolicy (p. 1303).

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::AutoScaling::LaunchConfiguration",
```

```
   "Properties" : {
      "AssociatePublicIpAddress" : Boolean,
      "BlockDeviceMappings" : [ BlockDeviceMapping, ... ],
      "ClassicLinkVPCId" : String,
      "ClassicLinkVPCSecurityGroups" : [ String, ... ],
      "EbsOptimized" : Boolean,
      "IamInstanceProfile" : String,
      "ImageId" : String,
      "InstanceId" : String,
      "InstanceMonitoring" : Boolean,
      "InstanceType" : String,
      "KernelId" : String,
      "KeyName" : String,
      "PlacementTenancy" : String,
      "RamDiskId" : String,
      "SecurityGroups" : [ SecurityGroup, ... ],
      "SpotPrice" : String,
      "UserData" : String
   }
}
```

YAML

```
Type: "AWS::AutoScaling::LaunchConfiguration"
Properties:
  AssociatePublicIpAddress: Boolean
  BlockDeviceMappings:
    - BlockDeviceMapping
  ClassicLinkVPCId: String
  ClassicLinkVPCSecurityGroups:
    - String
  EbsOptimized: Boolean
  IamInstanceProfile: String
  ImageId: String
  InstanceId: String
  InstanceMonitoring: Boolean
  InstanceType: String
  KernelId: String
  KeyName: String
  PlacementTenancy: String
  RamDiskId: String
  SecurityGroups:
    - SecurityGroup
  SpotPrice: String
  UserData: String
```

Properties

`AssociatePublicIpAddress`

For Amazon EC2 instances in a VPC, indicates whether instances in the Auto Scaling group receive public IP addresses. If you specify `true`, each instance in the Auto Scaling receives a unique public IP address.

Note

If this resource has a public IP address and is also in a VPC that is defined in the same template, you must use the `DependsOn` attribute to declare a dependency on the VPC-gateway attachment. For more information, see DependsOn Attribute (p. 1298).

Required: No

Type: Boolean

Update requires: Replacement (p. 90)

`BlockDeviceMappings`

Specifies how block devices are exposed to the instance. You can specify virtual devices and EBS volumes.

Required: No

Type: A list of BlockDeviceMappings (p. 1029).

Update requires: Replacement (p. 90)

`ClassicLinkVPCId`

The ID of a ClassicLink-enabled VPC to link your EC2-Classic instances to. You can specify this property only for EC2-Classic instances. For more information, see ClassicLink in the *Amazon Elastic Compute Cloud User Guide*.

Required: No

Type: String

Update requires: Replacement (p. 90)

`ClassicLinkVPCSecurityGroups`

The IDs of one or more security groups for the VPC that you specified in the `ClassicLinkVPCId` property.

Required: Conditional. If you specified the `ClassicLinkVPCId` property, you must specify this property.

Type: List of strings

Update requires: Replacement (p. 90)

`EbsOptimized`

Specifies whether the launch configuration is optimized for EBS I/O. This optimization provides dedicated throughput to Amazon EBS and an optimized configuration stack to provide optimal EBS I/O performance.

Additional fees are incurred when using EBS-optimized instances. For more information about fees and supported instance types, see EBS-Optimized Instances in the *Amazon EC2 User Guide for Linux Instances*.

Required: No If this property is not specified, "false" is used.

Type: Boolean

Update requires: Replacement (p. 90)

`IamInstanceProfile`

Provides the name or the Amazon Resource Name (ARN) of the instance profile associated with the IAM role for the instance. The instance profile contains the IAM role.

Required: No

Type: String (1–1600 chars)

Update requires: Replacement (p. 90)

`ImageId`

Provides the unique ID of the Amazon Machine Image (AMI) that was assigned during registration.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`InstanceId`

The ID of the Amazon EC2 instance you want to use to create the launch configuration. Use this property if you want the launch configuration to use settings from an existing Amazon EC2 instance.

When you use an instance to create a launch configuration, all properties are derived from the instance with the exception of `BlockDeviceMapping` and `AssociatePublicIpAddress`. You can override any properties from the instance by specifying them in the launch configuration.

Required: No

Type: String

Update requires: Replacement (p. 90)

`InstanceMonitoring`

Indicates whether detailed instance monitoring is enabled for the Auto Scaling group. By default, this property is set to `true` (enabled).

When detailed monitoring is enabled, Amazon CloudWatch (CloudWatch) generates metrics every minute and your account is charged a fee. When you disable detailed monitoring, CloudWatch generates metrics every 5 minutes. For more information, see Monitor Your Auto Scaling Instances in the *Auto Scaling Developer Guide*.

Required: No

Type: Boolean

Update requires: Replacement (p. 90)

`InstanceType`

Specifies the instance type of the EC2 instance.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`KernelId`

Provides the ID of the kernel associated with the EC2 AMI.

Required: No

Type: String

Update requires: Replacement (p. 90)

`KeyName`

Provides the name of the EC2 key pair.

Required: No

Type: String

Update requires: Replacement (p. 90)

`PlacementTenancy`

The tenancy of the instance. An instance with a tenancy of `dedicated` runs on single-tenant hardware and can only be launched in a VPC. You must set the value of this parameter to `dedicated` if want to launch dedicated instances in a shared tenancy VPC (a VPC with the instance placement tenancy attribute set to default). For more information, see CreateLaunchConfiguration in the *Auto Scaling API Reference*.

If you specify this property, you must specify at least one subnet in the VPCZoneIdentifier property of the AWS::AutoScaling::AutoScalingGroup (p. 433) resource.

Required: No

Type: String

Update requires: Replacement (p. 90)

`RamDiskId`

The ID of the RAM disk to select. Some kernels require additional drivers at launch. Check the kernel requirements for information about whether you need to specify a RAM disk. To find kernel requirements, refer to the AWS Resource Center and search for the kernel ID.

Required: No

Type: String

Update requires: Replacement (p. 90)

`SecurityGroups`

A list that contains the EC2 security groups to assign to the Amazon EC2 instances in the Auto Scaling group. The list can contain the name of existing EC2 security groups or references to `AWS::EC2::SecurityGroup` resources created in the template. If your instances are launched within VPC, specify Amazon VPC security group IDs.

Required: No

Type: A list of EC2 security groups.

Update requires: Replacement (p. 90)

`SpotPrice`

The spot price for this autoscaling group. If a spot price is set, then the autoscaling group will launch when the current spot price is less than the amount specified in the template.

When you have specified a spot price for an auto scaling group, the group will only launch when the spot price has been met, regardless of the setting in the autoscaling group's `DesiredCapacity`.

For more information about configuring a spot price for an autoscaling group, see Using Auto Scaling to Launch Spot Instances in the *AutoScaling Developer Guide*.

Required: No

Type: String

Update requires: Replacement (p. 90)

> **Note**
> When you change your bid price by creating a new launch configuration, running instances will continue to run as long as the bid price for those running instances is higher than the current Spot price.

`UserData`

The user data available to the launched EC2 instances.

Required: No

Type: String

Update requires: Replacement (p. 90)

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "LaunchConfig" }
```

For the resource with the logical ID `LaunchConfig`, `Ref` will return the Auto Scaling launch configuration name, such as `mystack-mylaunchconfig-1DDYF1E3B3I`.

For more information about using the `Ref` function, see Ref (p. 1343).

Template Examples

Example LaunchConfig with block device

This example shows a launch configuration that describes two Amazon Elastic Block Store mappings.

JSON

```
"LaunchConfig" : {
   "Type" : "AWS::AutoScaling::LaunchConfiguration",
   "Properties" : {
      "KeyName" : { "Ref" : "KeyName" },
      "ImageId" : {
         "Fn::FindInMap" : [
            "AWSRegionArch2AMI",
            { "Ref" : "AWS::Region" },
            {
               "Fn::FindInMap" : [
                  "AWSInstanceType2Arch", { "Ref" : "InstanceType" }, "Arch"
               ]
            }
         ]
      },
      "UserData" : { "Fn::Base64" : { "Ref" : "WebServerPort" }},
      "SecurityGroups" : [ { "Ref" : "InstanceSecurityGroup" } ],
      "InstanceType" : { "Ref" : "InstanceType" },
      "BlockDeviceMappings" : [
         {
           "DeviceName" : "/dev/sda1",
           "Ebs" : { "VolumeSize" : "50", "VolumeType" : "io1", "Iops" : 200 }
         },
         {
           "DeviceName" : "/dev/sdm",
           "Ebs" : { "VolumeSize" : "100", "DeleteOnTermination" : "true"}
         }
      ]
   }
```

```
}
```

YAML

```
LaunchConfig:
  Type: "AWS::AutoScaling::LaunchConfiguration"
  Properties:
    KeyName:
      Ref: "KeyName"
    ImageId:
      Fn::FindInMap:
        - "AWSRegionArch2AMI"
        - Ref: "AWS::Region"
        - Fn::FindInMap:
            - "AWSInstanceType2Arch"
            - Ref: "InstanceType"
            - "Arch"
    UserData:
      Fn::Base64:
        Ref: "WebServerPort"
    SecurityGroups:
      - Ref: "InstanceSecurityGroup"
    InstanceType:
      Ref: "InstanceType"
    BlockDeviceMappings:
      - DeviceName: "/dev/sda1"
        Ebs:
          VolumeSize: "50"
          VolumeType: "io1"
          Iops: 200
      - DeviceName: "/dev/sdm"
        Ebs:
          VolumeSize: "100"
          DeleteOnTermination: "true"
```

Example LaunchConfig with Spot Price in Autoscaling Group

This example shows a launch configuration that features a spot price in the AutoScaling group. This launch configuration will only be active if the current spot price is less than the amount in the template specification (0.05).

JSON

```
"LaunchConfig" : {
   "Type" : "AWS::AutoScaling::LaunchConfiguration",
   "Properties" : {
      "KeyName" : { "Ref" : "KeyName" },
      "ImageId" : {
         "Fn::FindInMap" : [
            "AWSRegionArch2AMI",
            { "Ref" : "AWS::Region" },
            {
               "Fn::FindInMap" : [
                  "AWSInstanceType2Arch", { "Ref" : "InstanceType" }, "Arch"
               ]
            }
         ]
      },
      "SecurityGroups" : [ { "Ref" : "InstanceSecurityGroup" } ],
      "SpotPrice" :  "0.05",
      "InstanceType" : { "Ref" : "InstanceType" }
   }
```

```
}
```

YAML

```
LaunchConfig:
  Type: "AWS::AutoScaling::LaunchConfiguration"
  Properties:
    KeyName:
      Ref: "KeyName"
    ImageId:
      Fn::FindInMap:
        - "AWSRegionArch2AMI"
        - Ref: "AWS::Region"
        - Fn::FindInMap:
            - "AWSInstanceType2Arch"
            - Ref: "InstanceType"
            - "Arch"
    SecurityGroups:
      - Ref: "InstanceSecurityGroup"
    SpotPrice: "0.05"
    InstanceType:
      Ref: "InstanceType"
```

Example LaunchConfig with IAM Instance Profile

Here's a launch configuration using the IamInstanceProfile (p. 442) property.

Only the `AWS::AutoScaling::LaunchConfiguration` specification is shown. For the full template, including the definition of, and further references from the AWS::IAM::InstanceProfile (p. 779) object referenced here as "RootInstanceProfile", see: auto_scaling_with_instance_profile.template.

JSON

```
"myLCOne": {
   "Type": "AWS::AutoScaling::LaunchConfiguration",
   "Properties": {
      "ImageId": {
         "Fn::FindInMap": [
            "AWSRegionArch2AMI",
            { "Ref": "AWS::Region" },
            {
               "Fn::FindInMap": [
                  "AWSInstanceType2Arch", { "Ref": "InstanceType" }, "Arch"
               ]
            }
         ]
      },
      "InstanceType": { "Ref": "InstanceType" },
      "IamInstanceProfile": { "Ref": "RootInstanceProfile" }
   }
}
```

YAML

```
myLCOne:
  Type: "AWS::AutoScaling::LaunchConfiguration"
  Properties:
    ImageId:
      Fn::FindInMap:
        - "AWSRegionArch2AMI"
```

```
        - Ref: "AWS::Region"
        - Fn::FindInMap:
            - "AWSInstanceType2Arch"
            - Ref: "InstanceType"
            - "Arch"
    InstanceType:
      Ref: "InstanceType"
    IamInstanceProfile:
      Ref: "RootInstanceProfile"
```

Example EBS-optimized volume with specified PIOPS

You can create an AWS CloudFormation stack with auto scaled instances that contain EBS-optimized volumes with a specified PIOPS. This can increase the performance of your EBS-backed instances as explained in Increasing EBS Performance in the *Amazon Elastic Compute Cloud User Guide*.

> **Caution**
> Additional fees are incurred when using EBS-optimized instances. For more information, see EBS-Optimized Instances in the *Amazon Elastic Compute Cloud User Guide*.

Because you cannot override PIOPS settings in an auto scaling launch configuration, the AMI in your launch configuration must have been configured with a block device mapping that specifies the desired PIOPS. You can do this by creating your own EC2 AMI with the following characteristics:

- An instance type of `m1.large` or greater. This is required for EBS optimization.
- An EBS-backed AMI with a volume type of "io1" and the number of IOPS you want for the Auto Scaling-launched instances.
- The size of the EBS volume must accommodate the IOPS you need. There is a 10 : 1 ratio between IOPS and Gibibytes (GiB) of storage, so for 100 PIOPS, you need at least 10 GiB storage on the root volume.

Use this AMI in your Auto Scaling launch configuration. For example, an EBS-optimized AMI with PIOPS that has the AMI ID `ami-7430ba44` would be used in your launch configuration like this:

JSON

```
"LaunchConfig" : {
   "Type" : "AWS::AutoScaling::LaunchConfiguration",
   "Properties" : {
      "KeyName" : { "Ref" : "KeyName" },
      "ImageId" : "ami-7430ba44",
      "UserData" : { "Fn::Base64" : { "Ref" : "WebServerPort" } },
      "SecurityGroups" : [ { "Ref" : "InstanceSecurityGroup" } ],
      "InstanceType" : "m1.large",
      "EbsOptimized" : "true"
   }
}
```

YAML

```
LaunchConfig:
  Type: "AWS::AutoScaling::LaunchConfiguration"
  Properties:
    KeyName:
      Ref: "KeyName"
    ImageId: "ami-7430ba44"
    UserData:
      Fn::Base64:
        Ref: "WebServerPort"
    SecurityGroups:
```

```
      - Ref: "InstanceSecurityGroup"
    InstanceType: "m1.large"
    EbsOptimized: "true"
```

Be sure to set the `InstanceType` to at least *m1.large* and set `EbsOptimized` to *true*.

When you create a launch configuration such as this one, your launched instances will contain optimized EBS root volumes with the PIOPS that you selected when creating the AMI.

To view more LaunchConfiguration snippets, see Auto Scaling Launch Configuration Resource (p. 240).

See Also

- Creating Your Own AMIs in the *Amazon Elastic Compute Cloud User Guide*.
- Block Device Mapping in the *Amazon Elastic Compute Cloud User Guide*.

AWS::AutoScaling::LifecycleHook

Use `AWS::AutoScaling::LifecycleHook` to control the state of an instance in an Auto Scaling group after it is launched or terminated. When you use a lifecycle hook, the Auto Scaling group either pauses the instance after it is launched (before it is put into service) or pauses the instance as it is terminated (before it is fully terminated). For more information, see Examples of How to Use Lifecycle Hooks in the *Auto Scaling User Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::AutoScaling::LifecycleHook",
  "Properties" : {
    "AutoScalingGroupName" : String,
    "DefaultResult" : String,
    "HeartbeatTimeout" : Integer,
    "LifecycleTransition" : String,
    "NotificationMetadata" : String,
    "NotificationTargetARN" : String,
    "RoleARN" : String
  }
}
```

YAML

```
Type: "AWS::AutoScaling::LifecycleHook"
Properties:
  AutoScalingGroupName: String
```

```
  DefaultResult: String
  HeartbeatTimeout: Integer
  LifecycleTransition: String
  NotificationMetadata: String
  NotificationTargetARN: String
  RoleARN: String
```

Properties

For information about valid and default values, see LifecycleHook in the *Auto Scaling API Reference*.

`AutoScalingGroupName`

The name of the Auto Scaling group for the lifecycle hook.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`DefaultResult`

The action the Auto Scaling group takes when the lifecycle hook timeout elapses or if an unexpected failure occurs.

Required: No

Type: String

Update requires: No interruption (p. 90)

`HeartbeatTimeout`

The amount of time that can elapse before the lifecycle hook times out. When the lifecycle hook times out, Auto Scaling performs the action that you specified in the DefaultResult property.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`LifecycleTransition`

The state of the Amazon EC2 instance to which you want to attach the lifecycle hook. For valid values, see the `LifecycleTransition` content for the LifecycleHook data type in the *Auto Scaling API Reference*.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`NotificationMetadata`

Additional information that you want to include when Auto Scaling sends a message to the notification target.

Required: No

Type: String

Update requires: No interruption (p. 90)

`NotificationTargetARN`

The Amazon resource name (ARN) of the notification target that Auto Scaling uses to notify you when an instance is in the transition state for the lifecycle hook. You can specify an Amazon SQS queue or an Amazon SNS topic. The notification message includes the following information: lifecycle action token, user account ID, Auto Scaling group name, lifecycle hook name, instance ID, lifecycle transition, and notification metadata.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`RoleARN`

The ARN of the IAM role that allows the Auto Scaling group to publish to the specified notification target. The role requires permissions to Amazon SNS and Amazon SQS.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "MyLifeCycleHook" }
```

`Ref` returns the lifecycle hook name, such as `mylifecyclehookname`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

In the following template snippet, the Auto Scaling pauses instances before completely terminating them. While in the pending state, you can, for example, connect to the instance and download logs or any other data before the instance is terminated.

JSON

```
"myLifecycleHook": {
  "Type": "AWS::AutoScaling::LifecycleHook",
  "Properties": {
    "AutoScalingGroupName": { "Ref": "myAutoScalingGroup" },
    "LifecycleTransition": "autoscaling:EC2_INSTANCE_TERMINATING",
    "NotificationTargetARN": { "Ref": "lifecycleHookTopic" },
    "RoleARN": { "Fn::GetAtt": [ "lifecycleHookRole", "Arn" ] }
  }
}
```

YAML

```
myLifecycleHook:
  Type: "AWS::AutoScaling::LifecycleHook"
  Properties:
```

```
    AutoScalingGroupName:
      Ref: myAutoScalingGroup
    LifecycleTransition: "autoscaling:EC2_INSTANCE_TERMINATING"
    NotificationTargetARN:
      Ref: lifecycleHookTopic
    RoleARN:
      Fn::GetAtt:
        - lifecycleHookRole
        - Arn
```

AWS::AutoScaling::ScalingPolicy

The `AWS::AutoScaling::ScalingPolicy` resource adds a scaling policy to an auto scaling group. A scaling policy specifies whether to scale the auto scaling group up or down, and by how much. For more information on scaling policies, see Scaling by Policy in the Auto Scaling Developer Guide.

You can use a scaling policy together with a CloudWatch alarm. A CloudWatch alarm can automatically initiate actions on your behalf, based on parameters you specify. A scaling policy is one type of action that an alarm can initiate. For a snippet showing how to create an Auto Scaling policy that is triggered by a CloudWatch alarm, see Auto Scaling Policy Triggered by CloudWatch Alarm (p. 241). Note that you can only associate one scaling policy with an alarm.

This type supports updates. For more information about updating this resource, see PutScalingPolicy.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::AutoScaling::ScalingPolicy",
   "Properties" : {
      "AdjustmentType (p. 453)" : String,
      "AutoScalingGroupName (p. 453)" : String,
      "Cooldown (p. 453)" : String,
      "EstimatedInstanceWarmup" : Integer,
      "MetricAggregationType" : String,
      "MinAdjustmentMagnitude" : Integer,
      "PolicyType" : String,
      "ScalingAdjustment (p. 454)" : Integer,
      "StepAdjustments" : [ StepAdjustments, ... ]
   }
}
```

YAML

```
Type: "AWS::AutoScaling::ScalingPolicy"
Properties:
  AdjustmentType (p. 453): String
```

```
  AutoScalingGroupName (p. 453): String
  Cooldown (p. 453): String
  EstimatedInstanceWarmup: Integer
  MetricAggregationType: String
  MinAdjustmentMagnitude: Integer
  PolicyType: String
  ScalingAdjustment (p. 454): Integer
  StepAdjustments:
    - StepAdjustments
```

Properties

`AdjustmentType`

Specifies whether the `ScalingAdjustment` is an absolute number or a percentage of the current capacity. Valid values are `ChangeInCapacity`, `ExactCapacity`, and `PercentChangeInCapacity`.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`AutoScalingGroupName`

The name or Amazon Resource Name (ARN) of the Auto Scaling Group that you want to attach the policy to.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`Cooldown`

The amount of time, in seconds, after a scaling activity completes before any further trigger-related scaling activities can start.

Do not specify this property if you are using the `StepScaling` policy type.

Required: No

Type: String

Update requires: No interruption (p. 90)

`EstimatedInstanceWarmup`

The estimated time, in seconds, until a newly launched instance can send metrics to CloudWatch. By default, Auto Scaling uses the cooldown period, as specified in the `Cooldown` property.

Do not specify this property if you are using the `SimpleScaling` policy type.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`MetricAggregationType`

The aggregation type for the CloudWatch metrics. You can specify `Minimum`, `Maximum`, or `Average`. By default, AWS CloudFormation specifies `Average`.

Do not specify this property if you are using the `SimpleScaling` policy type.

Required: No

Type: String

Update requires: No interruption (p. 90)

`MinAdjustmentMagnitude`

For the `PercentChangeInCapacity` adjustment type, the minimum number of instances to scale. The scaling policy changes the desired capacity of the Auto Scaling group by a minimum of this many instances. This property replaces the `MinAdjustmentStep` property.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`PolicyType`

An Auto Scaling policy type. You can specify `SimpleScaling` or `StepScaling`. By default, AWS CloudFormation specifies `SimpleScaling`. For more information, see Scaling Policy Types in the *Auto Scaling User Guide*.

Required: No

Type: String

Update requires: No interruption (p. 90)

`ScalingAdjustment`

The number of instances by which to scale. The `AdjustmentType` property determines if AWS CloudFormation interprets this number as an absolute number (when the `ExactCapacity` value is specified), increase or decrease capacity by a specified number (when the `ChangeInCapacity` value is specified), or increase or decrease capacity as a percentage of the existing Auto Scaling group size (when the `PercentChangeInCapacity` value is specified). A positive value adds to the current capacity and a negative value subtracts from the current capacity. For exact capacity, you must specify a positive value.

Required: Conditional. This property is required if the policy type is `SimpleScaling`. This property is not supported with any other policy type.

Type: Integer

Update requires: No interruption (p. 90)

`StepAdjustments`

A set of adjustments that enable you to scale based on the size of the alarm breach.

Required: Conditional. This property is required if the policy type is `StepScaling`. This property is not supported with any other policy type.

Type: List of Auto Scaling ScalingPolicy StepAdjustments (p. 1033)

Update requires: No interruption (p. 90)

Return Value

When you specify an `AWS::AutoScaling::ScalingPolicy` type as an argument to the `Ref` function, AWS CloudFormation returns the policy Amazon Resource Name (ARN), such as `arn:aws:autoscaling:us-east-1:123456789012:scalingPolicy:ab12c4d5-a1b2-a1b2-a1b2-ab12c4d56789:autoScalingGroupName/myStack-AutoScalingGroup-AB12C4D5E6:policyName/myStack-myScalingPolicy-AB12C4D5E6`.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

Simple policy type

The following example is a simple scaling policy that increases the number instances by one when it is triggered.

JSON

```
"SimpleScaling" : {
  "Type" : "AWS::AutoScaling::ScalingPolicy",
  "Properties" : {
    "AdjustmentType" : "ChangeInCapacity",
    "PolicyType" : "SimpleScaling",
    "Cooldown" : "60",
    "AutoScalingGroupName" : { "Ref" : "ASG" },
    "ScalingAdjustment" : 1
  }
}
```

YAML

```
SimpleScaling:
  Type: "AWS::AutoScaling::ScalingPolicy"
  Properties:
    AdjustmentType: "ChangeInCapacity"
    PolicyType: "SimpleScaling"
    Cooldown: "60"
    AutoScalingGroupName:
      Ref: "ASG"
    ScalingAdjustment: 1
```

Step policy type

The following example is a step scaling policy that increases the number instances by one or two, depending on the size of the alarm breach. For a breach that is less than 50 units than the threshold value, the policy increases the number of instances by one. For a breach that is 50 units or more higher than the threshold, the policy increases the number of instances by two.

JSON

```
"StepScaling" : {
  "Type" : "AWS::AutoScaling::ScalingPolicy",
  "Properties" : {
    "AdjustmentType" : "ChangeInCapacity",
    "AutoScalingGroupName" : { "Ref" : "ASG" },
    "PolicyType" : "StepScaling",
    "MetricAggregationType" : "Average",
    "EstimatedInstanceWarmup" : "60",
    "StepAdjustments": [
      {
        "MetricIntervalLowerBound": "0",
        "MetricIntervalUpperBound" : "50",
        "ScalingAdjustment": "1"
      },
      {
        "MetricIntervalLowerBound": "50",
```

```
          "ScalingAdjustment": "2"
        }
      ]
    }
  }
```

YAML

```
StepScaling:
  Type: "AWS::AutoScaling::ScalingPolicy"
  Properties:
    AdjustmentType: "ChangeInCapacity"
    AutoScalingGroupName:
      Ref: "ASG"
    PolicyType: "StepScaling"
    MetricAggregationType: "Average"
    EstimatedInstanceWarmup: "60"
    StepAdjustments:
      -
        MetricIntervalLowerBound: "0"
        MetricIntervalUpperBound: "50"
        ScalingAdjustment: "1"
      -
        MetricIntervalLowerBound: "50"
        ScalingAdjustment: "2"
```

AWS::AutoScaling::ScheduledAction

Creates a scheduled scaling action for an Auto Scaling group, changing the number of servers available for your application in response to predictable load changes.

Important
Note the following:

- If you have rolling updates enabled, you must suspend scheduled actions before you can update the Auto Scaling group. You can suspend processes by using the AWS CLI or Auto Scaling API. For more information, see Suspend and Resume Auto Scaling Process in the *Auto Scaling User Guide*.
- When you update a stack with an Auto Scaling group and scheduled action, AWS CloudFormation always sets the min size, max size, and desired capacity properties of your Auto Scaling group to the values that are defined in the `AWS::AutoScaling::AutoScalingGroup` resource of your template, even if a scheduled action is in effect. However, you might not want AWS CloudFormation to change any of the group size property values, such as when you have a scheduled action in effect. You can use an UpdatePolicy attribute (p. 1303) to prevent AWS CloudFormation from changing the min size, max size, or desired capacity property values during a stack update unless you modified the individual values in your template.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::AutoScaling::ScheduledAction",
  "Properties" : {
    "AutoScalingGroupName" : String,
    "DesiredCapacity" : Integer,
    "EndTime" : Time stamp,
    "MaxSize" : Integer,
    "MinSize" : Integer,
    "Recurrence" : String,
    "StartTime" : Time stamp
  }
}
```

YAML

```
Type: AWS::AutoScaling::ScheduledAction
Properties:
  AutoScalingGroupName: String
  DesiredCapacity: Integer
  EndTime: Time stamp
  MaxSize: Integer
  MinSize: Integer
  Recurrence: String
  StartTime: Time stamp
```

Properties

`AutoScalingGroupName`

The name or ARN of the Auto Scaling group.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`DesiredCapacity`

The number of Amazon EC2 instances that should be running in the Auto Scaling group.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`EndTime`

The time in UTC for this schedule to end. For example, `2010-06-01T00:00:00Z`.

Required: No

Type: Time stamp

Update requires: No interruption (p. 90)

`MaxSize`

The maximum number of Amazon EC2 instances in the Auto Scaling group.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`MinSize`

The minimum number of Amazon EC2 instances in the Auto Scaling group.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`Recurrence`

The time in UTC when recurring future actions will start. You specify the start time by following the Unix cron syntax format. For more information about cron syntax, go to http://en.wikipedia.org/wiki/Cron.

Specifying the `StartTime` and `EndTime` properties with `Recurrence` property forms the start and stop boundaries of the recurring action.

Required: No

Type: String

Update requires: No interruption (p. 90)

`StartTime`

The time in UTC for this schedule to start. For example, `2010-06-01T00:00:00Z`.

Required: No

Type: Time stamp

Update requires: No interruption (p. 90)

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "MyScheduledAction" }
```

For a scheduled Auto Scaling action with the logical ID `MyScheduledAction`, `Ref` returns the scheduled action name. For example:

```
mystack-myscheduledaction-NT5EUXTNTXXD
```

For more information about using the `Ref` function, see Ref (p. 1343).

Auto Scaling Scheduled Action Snippet

The following template snippet includes two scheduled actions that scale the number of instances in an Auto Scaling group. The `ScheduledActionUp` action starts at 7 AM every day and sets the Auto Scaling group to a minimum of five Amazon EC2 instances with a maximum of 10. The `ScheduledActionDown` action starts at 7 PM every day and sets the Auto Scaling group to a minimum and maximum of one Amazon EC2 instance.

JSON

```
"ScheduledActionUp": {
  "Type": "AWS::AutoScaling::ScheduledAction",
  "Properties": {
    "AutoScalingGroupName": {
      "Ref": "WebServerGroup"
    },
    "MaxSize": "10",
    "MinSize": "5",
    "Recurrence": "0 7 * * *"
  }
},
"ScheduledActionDown": {
  "Type": "AWS::AutoScaling::ScheduledAction",
  "Properties": {
    "AutoScalingGroupName": {
      "Ref": "WebServerGroup"
    },
    "MaxSize": "1",
    "MinSize": "1",
    "Recurrence": "0 19 * * *"
  }
}
```

YAML

```
ScheduledActionUp:
  Type: "AWS::AutoScaling::ScheduledAction"
  Properties:
    AutoScalingGroupName:
      Ref: "WebServerGroup"
    MaxSize: 10
    MinSize: 5
    Recurrence: "0 7 * * *"
ScheduledActionDown:
  Type: "AWS::AutoScaling::ScheduledAction"
  Properties:
    AutoScalingGroupName:
      Ref: "WebServerGroup"
    MaxSize: 1
    MinSize: 1
    Recurrence: "0 19 * * *"
```

AWS::CertificateManager::Certificate

The `AWS::CertificateManager::Certificate` resource requests an AWS Certificate Manager (ACM) certificate that you can use with AWS services to enable secure connections. For example, you can deploy an ACM certificate to an Elastic Load Balancing load balancer to enable HTTPS support. For more information, see the `RequestCertificate` action in the *AWS Certificate Manager API Reference*.

> **Important**
> When you use the `AWS::CertificateManager::Certificate` resource in an AWS CloudFormation stack, the stack will remain in the `CREATE_IN_PROGRESS` state and any further stack operations will be delayed until you act upon the instructions in the certificate validation email.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::CertificateManager::Certificate",
  "Properties" : {
    "DomainName" : String,
    "DomainValidationOptions" : [ DomainValidationOptions (p. 1035), ... ],
    "SubjectAlternativeNames" : [ String, ... ],
    "Tags" : [ Resource Tag, ... ]
  }
}
```

YAML

```
Type: "AWS::CertificateManager::Certificate"
Properties:
  DomainName: String
  DomainValidationOptions:
    - DomainValidationOptions (p. 1035)
  SubjectAlternativeNames:
    - String
  Tags:
    - Resource Tag
```

Properties

`DomainName`

Fully qualified domain name (FQDN), such as `www.example.com`, of the site that you want to secure with the ACM certificate. To protect several sites in the same domain, use an asterisk (`*`) to specify a wildcard. For example, `*.example.com` protects `www.example.com`, `site.example.com`, and `images.example.com`.

For constraints, see the `DomainName` parameter for the RequestCertificate action in the *AWS Certificate Manager API Reference*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`DomainValidationOptions`

Domain information that domain name registrars use to verify your identity. For more information and the default values, see Configure Email for Your Domain and Validate Domain Ownership in the *AWS Certificate Manager User Guide*.

Required: No

Type: List of AWS Certificate Manager Certificate DomainValidationOption (p. 1035)

Update requires: Replacement (p. 90)

`SubjectAlternativeNames`

FQDNs to be included in the Subject Alternative Name extension of the ACM certificate. For example, you can add `www.example.net` to a certificate for the `www.example.com` domain name so that users can reach your site by using either name.

Required: No

Type: List of strings

Update requires: Replacement (p. 90)

`Tags`

An arbitrary set of tags (key–value pairs) for this ACM certificate.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90).

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the certificate Amazon Resource Name (ARN), such as `arn:aws:acm:us-east-1:123456789012:certificate/12ab3c4d-56789-0ef1-2345-3dab6fa3ee50`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates an ACM certificate for the `example.com` domain name. ACM sends validation emails to the email address that is registered to the `example.com` domain.

JSON

```
"mycert" : {
  "Type" : "AWS::CertificateManager::Certificate",
  "Properties" : {
    "DomainName" : "example.com",
    "DomainValidationOptions" : [{
      "DomainName" : "example.com",
      "ValidationDomain" : "example.com"
    }]
  }
}
```

YAML

```
mycert:
```

```
  Type: AWS::CertificateManager::Certificate
  Properties:
    DomainName: example.com
    DomainValidationOptions:
    - DomainName: example.com
      ValidationDomain: example.com
```

AWS::CloudFormation::Authentication

Use the AWS::CloudFormation::Authentication resource to specify authentication credentials for files or sources that you specify with the AWS::CloudFormation::Init (p. 470) resource.

To include authentication information for a file or source that you specify with AWS::CloudFormation::Init, use the `uris` property if the source is a URI or the buckets property if the source is an Amazon S3 bucket. For more information about files, see Files (p. 476). For more information about sources, see Sources (p. 481).

You can also specify authentication information for files directly in the AWS::CloudFormation::Init resource. The files key of the resource contains a property named `authentication`. You can use the `authentication` property to associate authentication information defined in an AWS::CloudFormation::Authentication resource directly with a file.

For files, AWS CloudFormation looks for authentication information in the following order:

1. The `authentication` property of the AWS::CloudFormation::Init `files` key.
2. The `uris` or `buckets` property of the AWS::CloudFormation::Authentication resource.

For sources, AWS CloudFormation looks for authentication information in the `uris` or `buckets` property of the AWS::CloudFormation::Authentication resource.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

Unlike most AWS CloudFormation resources, the AWS::CloudFormation::Authentication type does not contain a block called "Properties", but instead contains a list of user-named blocks, each containing its own authentication properties.

Not all properties pertain to each authentication type; see the type (p. 463) property for more details.

JSON

```
{
  "Type" : "AWS::CloudFormation::Authentication" {
    "String" : {
      "accessKeyId (p. 463)" : String,
      "buckets (p. 463)" : [ String, ... ],
      "password (p. 463)" : String,
      "secretKey (p. 463)" : String,
      "type (p. 463)" : String,
```

```
      "uris (p. 464)" : [ String, ... ],
      "username (p. 464)" : String,
      "roleName (p. 464)" : String
    }
  }
}
```

YAML

```
Type: "AWS::CloudFormation::Authentication"
String:
  accessKeyId (p. 463): String
  buckets (p. 463):
    - String
  password (p. 463): String
  secretKey (p. 463): String
  type (p. 463): String
  uris (p. 464):
    - String
  username (p. 464): String
  roleName (p. 464): String
```

Properties

`accessKeyId`

Specifies the access key ID for S3 authentication.

Required: Conditional Can be specified only if the `type` property is set to `"S3"`.

Type: String

`buckets`

A comma-delimited list of Amazon S3 buckets to be associated with the S3 authentication credentials.

Required: Conditional Can be specified only if the `type` property is set to `"S3"`.

Type: List of strings

`password`

Specifies the password for basic authentication.

Required: Conditional Can be specified only if the type property is set to `"basic"`.

Type: String

`secretKey`

Specifies the secret key for S3 authentication.

Required: Conditional Can be specified only if the `type` property is set to `"S3"`.

Type: String

`type`

Specifies whether the authentication scheme uses a user name and password ("basic") or an access key ID and secret key ("S3").

If you specify `"basic"`, specify the `username`, `password`, and `uris` properties.

If you specify `"S3"`, specify the `accessKeyId`, `secretKey`, and `buckets` (optional) properties.

Required: Yes

Type: String Valid values are "basic" or "S3"

`uris`

A comma-delimited list of URIs to be associated with the basic authentication credentials. The authorization applies to the specified URIs and any more specific URI. For example, if you specify `http://www.example.com`, the authorization will also apply to `http://www.example.com/test`.

Required: Conditional Can be specified only if the `type` property is set to `"basic"`.

Type: List of strings

`username`

Specifies the user name for basic authentication.

Required: Conditional Can be specified only if the type property is set to `"basic"`.

Type: String

`roleName`

Describes the role for role-based authentication.

Important
The EC2 instance must be able to access this role using an instance profile.

Required: Conditional Can be specified only if the `type` property is set to `"S3"`.

Type: String.

Examples

EC2 Web Server Authentication

This template snippet shows how to get a file from a private S3 bucket within an EC2 instance. The credentials used for authentication are defined in the AWS::CloudFormation::Authentication resource, and referenced by the AWS::CloudFormation::Init resource in the *files* section.

JSON

```
"WebServer": {
   "Type": "AWS::EC2::Instance",
   "DependsOn" : "BucketPolicy",
   "Metadata" : {
      "AWS::CloudFormation::Init" : {
         "config" : {
            "packages" : { "yum" : { "httpd" : [] } },
            "files" : {
               "/var/www/html/index.html" : {
                  "source" : {
                     "Fn::Join" : [
                        "", [ "http://s3.amazonaws.com/", { "Ref" : "BucketName" }, "/
index.html" ]
                     ]
                  },
                  "mode"   : "000400",
                  "owner"  : "apache",
```

```
                  "group"  : "apache",
                  "authentication" : "S3AccessCreds"
               }
            },
            "services" : {
               "sysvinit" : {
                  "httpd" : { "enabled" : "true", "ensureRunning" : "true" }
               }
            }
         }
      },
      "AWS::CloudFormation::Authentication" : {
         "S3AccessCreds" : {
            "type" : "S3",
            "accessKeyId" : { "Ref" : "CfnKeys" },
            "secretKey" : { "Fn::GetAtt": [ "CfnKeys", "SecretAccessKey" ] }
         }
      }
   },
   "Properties": {
      EC2 Resource Properties ...
   }
}
```

YAML

```
WebServer:
  Type: "AWS::EC2::Instance"
  DependsOn: "BucketPolicy"
  Metadata:
    AWS::CloudFormation::Init:
      config:
        packages:
          yum:
            httpd: []
        files:
          /var/www/html/index.html:
            source:
              Fn::Join:
                - ""
                -
                  - "http://s3.amazonaws.com/"
                  - Ref: "BucketName"
                  - "/index.html"
            mode: "000400"
            owner: "apache"
            group: "apache"
            authentication: "S3AccessCreds"
        services:
          sysvinit:
            httpd:
              enabled: "true"
              ensureRunning: "true"
    AWS::CloudFormation::Authentication:
      S3AccessCreds:
        type: "S3"
        accessKeyId:
          Ref: "CfnKeys"
        secretKey:
          Fn::GetAtt:
            - "CfnKeys"
            - "SecretAccessKey"
Properties:
  EC2 Resource Properties ...
```

Specifying Both Basic and S3 Authentication

The following example template snippet includes both *basic* and *S3* authentication types.

JSON

```
"AWS::CloudFormation::Authentication" : {
   "testBasic" : {
      "type" : "basic",
      "username" : { "Ref" : "UserName" },
      "password" : { "Ref" : "Password" },
      "uris" : [ "http://www.example.com/test" ]
   },
   "testS3" : {
      "type" : "S3",
      "accessKeyId" : { "Ref" : "AccessKeyID" },
      "secretKey" : { "Ref" : "SecretAccessKeyID" },
      "buckets" : [ "myawsbucket" ]
   }
}
```

YAML

```
AWS::CloudFormation::Authentication:
  testBasic:
    type: "basic"
    username:
      Ref: "UserName"
    password:
      Ref: "Password"
    uris:
      - "http://www.example.com/test"
  testS3:
    type: "S3"
    accessKeyId:
      Ref: "AccessKeyID"
    secretKey:
      Ref: "SecretAccessKeyID"
    buckets:
      - "myawsbucket"
```

IAM Roles

The following example shows how to use IAM roles:

- `myRole` is an AWS::IAM::Role (p. 787) resource.
- The Amazon EC2 instance that runs `cfn-init` is associated with `myRole` through an instance profile.
- The example specifies the authentication by using the `buckets` property, like in Amazon S3 authentication. You can also specify authentication by name.

JSON

```
"AWS::CloudFormation::Authentication": {
    "rolebased" : {
        "type": "S3",
        "buckets": [ "myBucket" ],
        "roleName": { "Ref": "myRole" }
    }
```

```
}
```

YAML

```
AWS::CloudFormation::Authentication:
  rolebased:
    type: "S3"
    buckets:
      - "myBucket"
    roleName:
      Ref: "myRole"
```

AWS::CloudFormation::CustomResource

In an AWS CloudFormation template, you use the `AWS::CloudFormation::CustomResource` or `Custom::`*`String`* (p. 468) resource type to specify custom resources.

Custom resources provide a way for you to write custom provisioning logic in AWS CloudFormation template and have AWS CloudFormation run it during a stack operation, such as when you create, update or delete a stack. For more information, see Custom Resources (p. 361).

> **Note**
> If you use the VPC endpoint feature, custom resources in the VPC must have access to AWS CloudFormation-specific Amazon Simple Storage Service (Amazon S3) buckets. Custom resources must send responses to a pre-signed Amazon S3 URL. If they can't send responses to Amazon S3, AWS CloudFormation won't receive a response and the stack operation fails. For more information, see AWS CloudFormation and VPC Endpoints (p. 19).

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "Custom::String",
   "Version" : "1.0",
   "Properties" : {
      "ServiceToken" : String,
      ... provider-defined properties ...
   }
}
```

YAML

```
Type: "Custom::String"
Version: "1.0"
```

```
Properties:
  ServiceToken: String
    ... provider-defined properties ...
```

Custom::*String*

For custom resources, you can specify `AWS::CloudFormation::CustomResource` as the resource type, or you can specify your own resource type name. For example, instead of using `AWS::CloudFormation::CustomResource`, you can use `Custom::MyCustomResourceTypeName`.

Custom resource type names can include alphanumeric characters and the following characters: `_@-`. You can specify a custom resource type name up to a maximum length of 60 characters. You cannot change the type during an update.

Using your own resource type names helps you quickly differentiate the types of custom resources in your stack. For example, if you had two custom resources that conduct two different ping tests, you could name their type as `Custom::PingTester` to make them easily identifiable as ping testers (instead of using `AWS::CloudFormation::CustomResource`).

Properties

> **Note**
> Only one property is defined by AWS for a custom resource: `ServiceToken`. All other properties are defined by the service provider.

`ServiceToken`

The service token that was given to the template developer by the service provider to access the service, such as an Amazon SNS topic ARN or Lambda function ARN. The service token must be from the same region in which you are creating the stack.

Required: Yes

Type: String

Update requires: Updates are not supported.

Return Values

For a custom resource, return values are defined by the custom resource provider, and are retrieved by calling `Fn::GetAtt` (p. 1324) on the provider-defined attributes.

Examples

Creating a custom resource definition in a template

The following example demonstrates how to create a custom resource definition in a template.

All properties other than `ServiceToken`, and all `Fn::GetAtt` resource attributes, are defined by the custom resource provider.

JSON

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "MyFrontEndTest" : {
         "Type": "Custom::PingTester",
         "Version" : "1.0",
```

```
         "Properties" : {
            "ServiceToken": "arn:aws:sns:us-east-1:84969EXAMPLE:CRTest",
            "key1" : "string",
            "key2" : [ "list" ],
            "key3" : { "key4" : "map" }
         }
      }
   },
   "Outputs" : {
      "CustomResourceAttribute1" : {
         "Value" : { "Fn::GetAtt" : ["MyFrontEndTest", "responseKey1"] }
      },
      "CustomResourceAttribute2" : {
         "Value" : { "Fn::GetAtt" : ["MyFrontEndTest", "responseKey2"] }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  MyFrontEndTest:
    Type: "Custom::PingTester"
    Version: "1.0"
    Properties:
      ServiceToken: "arn:aws:sns:us-east-1:84969EXAMPLE:CRTest"
      key1: string
      key2:
        - list
      key3:
        key4: map
Outputs:
  CustomResourceAttribute1:
    Value:
      Fn::GetAtt:
        - MyFrontEndTest
        - responseKey1
  CustomResourceAttribute2:
    Value:
      Fn::GetAtt:
        - MyFrontEndTest
        - responseKey2
```

Using an AWS Lambda function in a custom resource

With Lambda functions and custom resources, you can run custom code in response to stack events (create, update, and delete). The following custom resource invokes a Lambda function and sends it the `StackName` property as input. The function uses this property to get outputs from the appropriate stack.

JSON

```
"MyCustomResource" : {
  "Type" : "Custom::TestLambdaCrossStackRef",
  "Properties" : {
    "ServiceToken": { "Fn::Join": [ "", [ "arn:aws:lambda:", { "Ref": "AWS::Region" }, ":",
 { "Ref": "AWS::AccountId" }, ":function:", {"Ref" : "LambdaFunctionName"} ] ] },
    "StackName": {
      "Ref": "NetworkStackName"
    }
  }
}
```

YAML

```
MyCustomResource:
  Type: "Custom::TestLambdaCrossStackRef"
  Properties:
    ServiceToken:
      !Sub |
        arn:aws:lambda:${AWS::Region}:${AWS::AccountId}:function:${LambdaFunctionName}
    StackName:
      Ref: "NetworkStackName"
```

Replacing a Custom Resource During an Update

You can update custom resources that require a replacement of the underlying physical resource. When you update a custom resource in an AWS CloudFormation template, AWS CloudFormation sends an update request to that custom resource. If the custom resource requires a replacement, the new custom resource must send a response with the new physical ID. When AWS CloudFormation receives the response, it compares the `PhysicalResourceId` between the old and new custom resources. If they are different, AWS CloudFormation recognizes the update as a replacement and sends a delete request to the old resource. For a step-by-step walkthrough of this process, see Stack Updates (p. 365).

Note the following:

- You can monitor the progress of the update in the **Events** tab. For more information, see Viewing Stack Data and Resources (p. 78).
- For more information about resource behavior during updates, see AWS CloudFormation Stacks Updates (p. 89).

AWS::CloudFormation::Init

Topics

Use the AWS::CloudFormation::Init type to include metadata on an Amazon EC2 instance for the cfn-init helper script. If your template calls the cfn-init script, the script looks for resource metadata rooted in the AWS::CloudFormation::Init metadata key. For more information about cfn-init, see cfn-init (p. 1355).

cfn-init supports all metadata types for Linux systems. It supports metadata types for Windows with conditions that are described in the sections that follow.

For an example of using AWS::CloudFormation::Init and the cfn-init helper script, see Deploying Applications on Amazon EC2 with AWS CloudFormation (p. 212).

For an example that shows how to use cfn-init to create a Windows stack, see Bootstrapping AWS CloudFormation Windows Stacks (p. 122).

Syntax

The configuration is separated into sections. The following template snippet shows how you can attach metadata for cfn-init to an Amazon EC2 instance resource within the template.

The metadata is organized into config keys, which you can group into configsets. You can specify a configset when you call cfn-init in your template. If you don't specify a configset, cfn-init looks for a single config key named *config*.

> **Note**
> The cfn-init helper script processes these configuration sections in the following order: packages, groups, users, sources, files, commands, and then services. If you require a different order, separate your sections into different config keys, and then use a configset that specifies the order in which the config keys should be processed.

JSON

```
"Resources": {
  "MyInstance": {
    "Type": "AWS::EC2::Instance",
    "Metadata" : {
      "AWS::CloudFormation::Init" : {
        "config" : {
          "packages" : {
            :
          },
          "groups" : {
            :
          },
          "users" : {
            :
          },
          "sources" : {
            :
          },
          "files" : {
            :
          },
          "commands" : {
            :
          },
          "services" : {
            :
          }
        }
      }
    },
    "Properties": {
      :
    }
  }
}
```

YAML

```
Resources:
  MyInstance:
    Type: "AWS::EC2::Instance"
    Metadata:
      AWS::CloudFormation::Init:
        config:
```

```
        packages:
          :
        groups:
          :
        users:
          :
        sources:
          :
        files:
          :
        commands:
          :
        services:
          :
  Properties:
    :
```

Configsets

If you want to create more than one config key and to have cfn-init process them in a specific order, create a configset that contains the config keys in the desired order.

Single Configset

The following template snippet creates configsets named `ascending` and `descending` that each contain two config keys.

JSON

```
"AWS::CloudFormation::Init" : {
    "configSets" : {
        "ascending" : [ "config1" , "config2" ],
        "descending" : [ "config2" , "config1" ]
    },
    "config1" : {
        "commands" : {
            "test" : {
                "command" : "echo \"$CFNTEST\" > test.txt",
                "env" : { "CFNTEST" : "I come from config1." },
                "cwd" : "~"
            }
        }
    },
    "config2" : {
        "commands" : {
            "test" : {
                "command" : "echo \"$CFNTEST\" > test.txt",
                "env" : { "CFNTEST" : "I come from config2" },
                "cwd" : "~"
            }
        }
    }
}
```

YAML

```
AWS::CloudFormation::Init:
  configSets:
    ascending:
      - "config1"
      - "config2"
```

```
    descending:
      - "config2"
      - "config1"
  config1:
    commands:
      test:
        command: "echo \"$CFNTEST\" > test.txt"
        env:
          CFNTEST: "I come from config1."
        cwd: "~"
  config2:
    commands:
      test:
        command: "echo \"$CFNTEST\" > test.txt"
        env:
          CFNTEST: "I come from config2"
        cwd: "~"
```

Related cfn-init Calls

The following example calls to cfn-init refer to the preceding example configsets. The example calls are abbreviated for clarity, see cfn-init (p. 1355) for the complete syntax.

- If a call to cfn-init specifies the `ascending` configset:

  ```
  cfn-init -c ascending
  ```

  the script processes `config1` and then processes `config2` and the test.txt file would contain the text `I come from config2`.
- If a call to cfn-init specifies the `descending` configset:

  ```
  cfn-init -c descending
  ```

  the script processes `config2` and then processes `config1` and the test.txt file would contain the text `I come from config1`.

Multiple Configsets

You can create multiple configsets, and call a series of them using your cfn-init script. Each configset can contain a list of config keys or references to other configsets. For example, the following template snippet creates three configsets. The first configset, `test1`, contains one config key named `1`. The second configset, `test2`, contains a reference to the `test1` configset and one config key named `2`. The third configset, default, contains a reference to the configset `test2`.

JSON

```
"AWS::CloudFormation::Init" : {
    "configSets" : {
        "test1" : [ "1" ],
        "test2" : [ { "ConfigSet" : "test1" }, "2" ],
        "default" : [ { "ConfigSet" : "test2" } ]
    },
    "1" : {
        "commands" : {
            "test" : {
                "command" : "echo \"$MAGIC\" > test.txt",
                "env" : { "MAGIC" : "I come from the environment!" },
                "cwd" : "~"
            }
```

```
        }
    },
    "2" : {
        "commands" : {
            "test" : {
                "command" : "echo \"$MAGIC\" >> test.txt",
                "env" : { "MAGIC" : "I am test 2!" },
                "cwd" : "~"
            }
        }
    }
}
```

YAML

```
AWS::CloudFormation::Init:
  1:
    commands:
      test:
        command: "echo \"$MAGIC\" > test.txt"
        env:
          MAGIC: "I come from the environment!"
        cwd: "~"
  2:
    commands:
      test:
        command: "echo \"$MAGIC\" >> test.txt"
        env:
          MAGIC: "I am test 2!"
        cwd: "~"
  configSets:
    test1:
      - "1"
    test2:
      -
        ConfigSet: "test1"
      - "2"
    default:
      -
        ConfigSet: "test2"
```

Related cfn-init Calls

The following calls to cfn-init refer to the configSets declared in the preceding template snippet. The example calls are abbreviated for clarity, see cfn-init (p. 1355) for the complete syntax.

- If you specify `test1` only:

  ```
  cfn-init -c test1
  ```

  cfn-init processes config key `1` only.
- If you specify `test2` only:

  ```
  cfn-init -c test2
  ```

  cfn-init processes config key `1` and then processes config key `2`.
- If you specify the `default` configset (or no configsets at all):

  ```
  cfn-init -c default
  ```

you get the same behavior that you would if you specify configset `test2`.

Commands

You can use the commands key to execute commands on the EC2 instance. The commands are processed in alphabetical order by name.

Key	Description
command	Required. Either an array or a string specifying the command to run. If you use an array, you do not need to escape space characters or enclose command parameters in quotes. Don't use the array to specify multiple commands.
env	Optional. Sets environment variables for the command. This property overwrites, rather than appends, the existing environment.
cwd	Optional. The working directory
test	Optional. A test command that determines whether cfn-init runs commands that are specified in the command key. If the test passes, cfn-init runs the commands. The cfn-init script runs the test in a command interpreter, such as Bash or `cmd.exe`. Whether a test passes depends on the exit code that the interpreter returns. For Linux, the test command must return an exit code of `0` for the test to pass. For Windows, the test command must return an %ERRORLEVEL% of `0`.
ignoreErrors	Optional. A Boolean value that determines whether cfn-init continues to run if the command in contained in the command key fails (returns a non-zero value). Set to `true` if you want cfn-init to continue running even if the command fails. Set to `false` if you want cfn-init to stop running if the command fails. The default value is `false`.
waitAfterCompletion	Optional. For Windows systems only. Specifies how long to wait (in seconds) after a command has finished in case the command causes a reboot. The default value is 60 seconds and a value of "forever" directs cfn-init to exit and resume only after the reboot is complete.

Example

The following example snippet calls the echo command if the `~/test.txt` file doesn't exist.

JSON

```
"commands" : {
    "test" : {
        "command" : "echo \"$MAGIC\" > test.txt",
        "env" : { "MAGIC" : "I come from the environment!" },
        "cwd" : "~",
        "test" : "test ! -e ~/test.txt",
        "ignoreErrors" : "false"
    },
    "test2" : {
        "command" : "echo \"$MAGIC2\" > test2.txt",
        "env" : { "MAGIC2" : "I come from the environment!" },
        "cwd" : "~",
        "test" : "test ! -e ~/test2.txt",
```

```
        "ignoreErrors" : "false"
    }
}
```

YAML

```
commands:
  test:
    command: "echo \"$MAGIC\" > test.txt"
    env:
      MAGIC: "I come from the environment!"
    cwd: "~"
    test: "test ! -e ~/test.txt"
    ignoreErrors: "false"
  test2:
    command: "echo \"$MAGIC2\" > test2.txt"
    env:
      MAGIC2: "I come from the environment!"
    cwd: "~"
    test: "test ! -e ~/test2.txt"
    ignoreErrors: "false"
```

Files

You can use the `files` key to create files on the EC2 instance. The content can be either inline in the template or the content can be pulled from a URL. The files are written to disk in lexicographic order. The following table lists the supported keys.

Key	Description
content	Either a string or a properly formatted JSON object. If you use a JSON object as your content, the JSON will be written to a file on disk. Any intrinsic functions such as Fn::GetAtt or Ref are evaluated before the JSON object is written to disk. When you create a symlink, specify the symlink target as the content. **Note** If you create a symlink, the helper script modifies the permissions of the target file. Currently, you can't create a symlink without modifying the permissions of the target file.
source	A URL to load the file from. This option cannot be specified with the content key.
encoding	The encoding format. Only used if the content is a string. Encoding is not applied if you are using a source. Valid values: `plain` | `base64`
group	The name of the owning group for this file. Not supported for Windows systems.
owner	The name of the owning user for this file. Not supported for Windows systems.
mode	A six-digit octal value representing the mode for this file. Not supported for Windows systems. Use the first three digits for symlinks and the last three digits for setting permissions. To create a symlink, specify **`120`*`xxx`***, where `xxx` defines the permissions of the target file. To specify permissions for a file, use the last three digits, such as **`000644`**.

Key	Description
authentication	The name of an authentication method to use. This overrides any default authentication. You can use this property to select an authentication method you define with the AWS::CloudFormation::Authentication (p. 462) resource.
context	Specifies a context for files that are to be processed as Mustache templates. To use this key, you must have installed aws-cfn-bootstrap 1.3-11 or later as well as pystache.

Examples

The following example snippet creates a file named setup.mysql as part of a larger installation.

The full template is available at: https://s3.amazonaws.com/cloudformation-templates-us-east-1/Drupal_Single_Instance.template

The following example snippet creates a symlink `/tmp/myfile2.txt` that points at an existing file `/tmp/myfile1.txt`. The permissions of the target file `/tmp/myfile1.txt` is defined by the mode value `644`.

Mustache templates are used primarily to create configuration files. For example, you can store a configuration file in an S3 bucket and interpolate Refs and GetAtts from the template, instead of using `Fn::Join` (p. 1336). The following example snippet outputs "Content for test9" to `/tmp/test9.txt`.

When working with Mustache templates, note the following:

- The context key must be present for the files to be processed.
- The context key must be a key-value map, but it can be nested.
- You can process files with inline content by using the content key and remote files by using the source key.
- Mustache support depends on the pystache version. Version 0.5.2 supports the Mustache 1.1.2 specification.

Groups

You can use the groups key to create Linux/UNIX groups and to assign group IDs. The groups key is not supported for Windows systems.

To create a group, add a new key-value pair that maps a new group name to an optional group ID. The groups key can contain one or more group names. The following table lists the available keys.

Key	Description
`gid`	A group ID number. If a group ID is specified, and the group already exists by name, the group creation will fail. If another group has the specified group ID, the OS may reject the group creation. Example: `{ "gid" : "23" }`

Example snippet

The following snippet specifies a group named `groupOne` without assigning a group ID and a group named `groupTwo` that specified a group ID value of `45`.

JSON

```
"groups" : {
    "groupOne" : {},
    "groupTwo" : { "gid" : "45" }
}
```

YAML

```
groups:
  groupOne: {}
  groupTwo:
    gid: "45"
```

Packages

You can use the packages key to download and install pre-packaged applications and components. On Windows systems, the packages key supports only the MSI installer.

Supported package formats

The cfn-init script currently supports the following package formats: apt, msi, python, rpm, rubygems, and yum. Packages are processed in the following order: rpm, yum/apt, and then rubygems and python. There is no ordering between rubygems and python, and packages within each package manager are not guaranteed to be installed in any order.

Specifying versions

Within each package manager, each package is specified as a package name and a list of versions. The version can be a string, a list of versions, or an empty string or list. An empty string or list indicates that you want the latest version. For rpm manager, the version is specified as a path to a file on disk or a URL.

If you specify a version of a package, cfn-init will attempt to install that version even if a newer version of the package is already installed on the instance. Some package managers support multiple versions, but others may not. Please check the documentation for your package manager for more information. If you do not specify a version and a version of the package is already installed, the cfn-init script will not install a new version—it will assume that you want to keep and use the existing version.

Example snippets

RPM, yum, and Rubygems

The following snippet specifies a version URL for rpm, requests the latest versions from yum, and version 0.10.2 of chef from rubygems:

JSON

```
"rpm" : {
  "epel" : "http://download.fedoraproject.org/pub/epel/5/i386/epel-release-5-4.noarch.rpm"
},
"yum" : {
  "httpd" : [],
  "php" : [],
  "wordpress" : []
},
"rubygems" : {
  "chef" : [ "0.10.2" ]
}
```

YAML

```
rpm:
  epel: "http://download.fedoraproject.org/pub/epel/5/i386/epel-release-5-4.noarch.rpm"
yum:
  httpd: []
  php: []
  wordpress: []
rubygems:
  chef:
    - "0.10.2"
```

MSI Package

The following snippet specifies a URL for an MSI package:

JSON

```
"msi" : {
  "awscli" : "https://s3.amazonaws.com/aws-cli/AWSCLI64.msi"
}
```

YAML

```
msi:
  awscli: "https://s3.amazonaws.com/aws-cli/AWSCLI64.msi"
```

Services

You can use the services key to define which services should be enabled or disabled when the instance is launched. On Linux systems, this key is supported by using sysvinit. On Windows systems, it is supported by using the Windows service manager.

The services key also allows you to specify dependencies on sources, packages and files so that if a restart is needed due to files being installed, cfn-init will take care of the service restart. For example, if you download the Apache HTTP Server package, the package installation will automatically start the Apache HTTP Server during the stack creation process. However, if the Apache HTTP Server configuration is updated later in the stack creation process, the update won't take effect unless the Apache server is restarted. You can use the services key to ensure that the Apache HTTP service is restarted.

The following table lists the supported keys.

Key	Description
ensureRunning	Set to true to ensure that the service is running after cfn-init finishes. Set to false to ensure that the service is not running after cfn-init finishes. Omit this key to make no changes to the service state.
enabled	Set to true to ensure that the service will be started automatically upon boot. Set to false to ensure that the service will not be started automatically upon boot. Omit this key to make no changes to this property.
files	A list of files. If cfn-init changes one directly via the files block, this service will be restarted

Key	Description
sources	A list of directories. If cfn-init expands an archive into one of these directories, this service will be restarted.
packages	A map of package manager to list of package names. If cfn-init installs or updates one of these packages, this service will be restarted.
commands	A list of command names. If cfn-init runs the specified command, this service will be restarted.

Examples

Linux

The following Linux snippet configures the services as follows:

- The nginx service will be restarted if either /etc/nginx/nginx.conf or /var/www/html are modified by cfn-init.
- The php-fastcgi service will be restarted if cfn-init installs or updates php or spawn-fcgi using yum.
- The sendmail service will be stopped and disabled.

JSON

```
"services" : {
  "sysvinit" : {
    "nginx" : {
      "enabled" : "true",
      "ensureRunning" : "true",
      "files" : ["/etc/nginx/nginx.conf"],
      "sources" : ["/var/www/html"]
    },
    "php-fastcgi" : {
      "enabled" : "true",
      "ensureRunning" : "true",
      "packages" : { "yum" : ["php", "spawn-fcgi"] }
    },
    "sendmail" : {
      "enabled" : "false",
      "ensureRunning" : "false"
    }
  }
}
```

YAML

```
services:
  sysvinit:
    nginx:
      enabled: "true"
      ensureRunning: "true"
      files:
        - "/etc/nginx/nginx.conf"
      sources:
        - "/var/www/html"
    php-fastcgi:
      enabled: "true"
      ensureRunning: "true"
      packages:
        yum:
```

```
          - "php"
          - "spawn-fcgi"
    sendmail:
      enabled: "false"
      ensureRunning: "false"
```

Windows

The following Windows snippet starts the `cfn-hup` service, sets it to automatic, and restarts the service if `cfn-init` modifies the specified configuration files:

JSON

```
"services" : {
  "windows" : {
    "cfn-hup" : {
      "enabled" : "true",
      "ensureRunning" : "true",
      "files" : ["c:\\cfn\\cfn-hup.conf", "c:\\cfn\\hooks.d\\cfn-auto-reloader.conf"]
    }
  }
}
```

YAML

```
services:
  windows:
    cfn-hup:
      enabled: "true"
      ensureRunning: "true"
      files:
        - "c:\\cfn\\cfn-hup.conf"
        - "c:\\cfn\\hooks.d\\cfn-auto-reloader.conf"
```

Sources

You can use the sources key to download an archive file and unpack it in a target directory on the EC2 instance. This key is fully supported for both Linux and Windows systems.

Supported formats

Supported formats are tar, tar+gzip, tar+bz2 and zip.

Examples

GitHub

If you use GitHub as a source control system, you can use cfn-init and the sources package mechanism to pull a specific version of your application. GitHub allows you to create a zip or a tar from a specific version via a URL as follows:

```
https://github.com/<your directory>/(zipball|tarball)/<version>
```

For example, the following snippet pulls down version *master* as a `.tar` file.

JSON

```
"sources" : {
  "/etc/puppet" : "https://github.com/user1/cfn-demo/tarball/master"
```

```
  }
```

YAML

```
sources:
  /etc/puppet: "https://github.com/user1/cfn-demo/tarball/master"
```

S3 Bucket

The following example downloads a zip file from an Amazon S3 bucket and unpacks it into /etc/myapp:

> **Note**
> You can use authentication credentials for a source. However, you cannot put an authentication key in the sources block. Instead, include a buckets key in your S3AccessCreds block. For an example, see the example template. For more information on Amazon S3 authentication credentials, see AWS::CloudFormation::Authentication (p. 462).

JSON

```
"sources" : {
  "/etc/myapp" : "https://s3.amazonaws.com/mybucket/myapp.tar.gz"
  }
```

YAML

```
sources:
  /etc/myapp: "https://s3.amazonaws.com/mybucket/myapp.tar.gz"
```

Users

You can use the users key to create Linux/UNIX users on the EC2 instance. The users key is not supported for Windows systems.

The following table lists the supported keys.

Key	Description
uid	A user ID. The creation process fails if the user name exists with a different user ID. If the user ID is already assigned to an existing user the operating system may reject the creation request.
groups	A list of group names. The user will be added to each group in the list.
homeDir	The user's home directory.

Example

Users are created as non-interactive system users with a shell of /sbin/nologin. This is by design and cannot be modified.

JSON

```

"users" : {
    "myUser" : {
        "groups" : ["groupOne", "groupTwo"],
        "uid" : "50",
        "homeDir" : "/tmp"
```

```
    }
}
```

YAML

```
users:
  myUser:
    groups:
      - "groupOne"
      - "groupTwo"
    uid: "50"
    homeDir: "/tmp"
```

AWS::CloudFormation::Interface

`AWS::CloudFormation::Interface` is a metadata key that defines how parameters are grouped and sorted in the AWS CloudFormation console. When you create or update stacks in the console, the console lists input parameters in alphabetical order by their logical IDs. By using this key, you can define your own parameter grouping and ordering so that users can efficiently specify parameter values. For example, you could group all EC2-related parameters in one group and all VPC-related parameters in another group.

In addition to grouping and ordering parameters, you can define labels for parameters. A label is a friendly name or description that the console displays instead of a parameter's logical ID. Labels are useful for helping users understand the values to specify for each parameter. For example, you could label a `KeyPair` parameter `Select an EC2 key pair`.

> **Note**
> Only the AWS CloudFormation console uses the `AWS::CloudFormation::Interface` metadata key. AWS CloudFormation CLI and API calls do not use this key.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
"Metadata" : {
  "AWS::CloudFormation::Interface" : {
    "ParameterGroups" : [ ParameterGroup, ... ],
    "ParameterLabels" : ParameterLabel
  }
}
```

YAML

```
Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
      - ParameterGroup
    ParameterLabels:
      ParameterLabel
```

Properties

`ParameterGroups`

A list of parameter group types, where you specify group names, the parameters in each group, and the order in which the parameters are shown.

Required: No

Type: AWS CloudFormation Interface ParameterGroup (p. 1038)

Update requires: No interruption (p. 90)

`ParameterLabels`

A mapping of parameters and their friendly names that the AWS CloudFormation console shows when a stack is created or updated.

Required: No

Type: AWS CloudFormation Interface ParameterLabel (p. 1039)

Update requires: No interruption (p. 90)

Example

The following example defines two parameter groups: `Network Configuration` and `Amazon EC2 Configuration`. The `Network Configuration` group includes the `VPCID`, `SubnetId`, and `SecurityGroupID` parameters, which are defined in the `Parameters` section of the template (not shown). The order in which the console shows these parameters is defined by the order in which the parameters are listed, starting with the `VPCID` parameter. The example similarly groups and orders the `Amazon EC2 Configuration` parameters.

The example also defines a label for the `VPCID` parameter. The console will show **Which VPC should this be deployed to?** instead of the parameter's logical ID (`VPCID`).

JSON

```
"Metadata" : {
  "AWS::CloudFormation::Interface" : {
    "ParameterGroups" : [
      {
        "Label" : { "default" : "Network Configuration" },
        "Parameters" : [ "VPCID", "SubnetId", "SecurityGroupID" ]
      },
      {
        "Label" : { "default":"Amazon EC2 Configuration" },
        "Parameters" : [ "InstanceType", "KeyName" ]
      }
    ],
    "ParameterLabels" : {
      "VPCID" : { "default" : "Which VPC should this be deployed to?" }
    }
  }
}
```

YAML

```
Metadata:
  AWS::CloudFormation::Interface:
```

```
    ParameterGroups:
      -
        Label:
          default: "Network Configuration"
        Parameters:
          - VPCID
          - SubnetId
          - SecurityGroupID
      -
        Label:
          default: "Amazon EC2 Configuration"
        Parameters:
          - InstanceType
          - KeyName
    ParameterLabels:
      VPCID:
        default: "Which VPC should this be deployed to?"
```

Parameter Groups in the Console

Using the metadata key from this example, the following figure shows how the console displays parameter groups when a stack is created or updated:

AWS::CloudFormation::Stack

The AWS::CloudFormation::Stack type nests a stack as a resource in a top-level template.

You can add output values from a nested stack within the containing template. You use the GetAtt (p. 1324) function with the nested stack's logical name and the name of the output value in the nested stack in the format Outputs.*NestedStackOutputName*.

When you apply template changes to update a top-level stack, AWS CloudFormation updates the top-level stack and initiates an update to its nested stacks. AWS CloudFormation updates the resources of modified nested stacks, but does not update the resources of unmodified nested stacks. For more information, see AWS CloudFormation Stacks Updates (p. 89).

> **Note**
> You must acknowledge IAM capabilities for nested stacks that contain IAM resources. Also, verify that you have cancel update stack permissions, which is required if an update rolls back. For more information about IAM and AWS CloudFormation, see Controlling Access with AWS Identity and Access Management (p. 8).

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::CloudFormation::Stack",
  "Properties" : {
    "NotificationARNs" : [ String, ... ],
```

```
    "Parameters" : { CloudFormation Stack Parameters Property Type },
    "Tags" : [ Resource Tag, ... ],
    "TemplateURL" : String,
    "TimeoutInMinutes" : Integer
  }
}
```

YAML

```
Type: "AWS::CloudFormation::Stack"
Properties:
  NotificationARNs:
    - String
  Parameters:
    CloudFormation Stack Parameters Property Type
  Tags:
    - Resource Tag
  TemplateURL: String
  TimeoutInMinutes: Integer
```

Properties

`NotificationARNs`

A list of existing Amazon SNS topics where notifications about stack events are sent.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`Parameters`

The set of parameters passed to AWS CloudFormation when this nested stack is created.

Note
If you use the `ref` function to pass a parameter value to a nested stack, comma-delimited list parameters must be of type `String`. In other words, you cannot pass values that are of type `CommaDelimitedList` to nested stacks.

Required: Conditional (required if the nested stack requires input parameters).

Type: CloudFormation Stack Parameters Property Type (p. 1036)

Update requires: Whether an update causes interruptions depends on the resources that are being update. An update never causes a nested stack to be replaced.

`Tags`

An arbitrary set of tags (key–value pairs) to describe this stack.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90).

`TemplateURL`

The URL of a template that specifies the stack that you want to create as a resource. The template must be stored on an Amazon S3 bucket, so the URL must have the form: `https://s3.amazonaws.com/.../TemplateName.template`

Required: Yes

Type: String

Update requires: Whether an update causes interruptions depends on the resources that are being update. An update never causes a nested stack to be replaced.

`TimeoutInMinutes`

The length of time, in minutes, that AWS CloudFormation waits for the nested stack to reach the CREATE_COMPLETE state. The default is no timeout. When AWS CloudFormation detects that the nested stack has reached the CREATE_COMPLETE state, it marks the nested stack resource as CREATE_COMPLETE in the parent stack and resumes creating the parent stack. If the timeout period expires before the nested stack reaches CREATE_COMPLETE, AWS CloudFormation marks the nested stack as failed and rolls back both the nested stack and parent stack.

Required: No

Type: Integer

Update requires: Updates are not supported.

Return Values

Ref

For AWS::CloudFormation::Stack, `Ref` returns the Stack ID. For example:

```
arn:aws:cloudformation:us-east-1:123456789012:stack/mystack-mynestedstack-sggfrhxhum7w/
f449b250-b969-11e0-a185-5081d0136786
```

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Outputs.`*`NestedStackOutputName`*

Returns: The output value from the specified nested stack where *`NestedStackOutputName`* is the name of the output value.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Related Information

- For sample template snippets, see Nested Stacks in AWS CloudFormation Template Snippets (p. 244).
- If you have nested stacks that are stuck in an in-progress operation, see Troubleshooting Errors in Troubleshooting AWS CloudFormation (p. 1368).

AWS::CloudFormation::WaitCondition

Important
For Amazon EC2 and Auto Scaling resources, we recommend that you use a CreationPolicy attribute instead of wait conditions. Add a CreationPolicy attribute to those resources, and use the cfn-signal helper script to signal when an instance creation process has completed successfully.

You can use a wait condition for situations like the following:

- To coordinate stack resource creation with configuration actions that are external to the stack creation
- To track the status of a configuration process

For these situations, we recommend that you associate a CreationPolicy (p. 1293) attribute with the wait condition so that you don't have to use a wait condition handle. For more information and an example, see Creating Wait Conditions in a Template (p. 228). If you use a CreationPolicy with a wait condition, do not specify any of the wait condition's properties.

> **Note**
> If you use the VPC endpoint feature, resources in the VPC that respond to wait conditions must have access to AWS CloudFormation-specific Amazon Simple Storage Service (Amazon S3) buckets. Resources must send wait condition responses to a pre-signed Amazon S3 URL. If they can't send responses to Amazon S3, AWS CloudFormation won't receive a response and the stack operation fails. For more information, see AWS CloudFormation and VPC Endpoints (p. 19).

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::CloudFormation::WaitCondition",
   "Properties" : {
      "Count (p. 488)" : Integer,
      "Handle (p. 489)" : String,
      "Timeout (p. 489)" : String
   }
}
```

YAML

```
Type: "AWS::CloudFormation::WaitCondition"
Properties:
  Count (p. 488): Integer
  Handle (p. 489): String
  Timeout (p. 489): String
```

Properties

`Count`

The number of success signals that AWS CloudFormation must receive before it continues the stack creation process. When the wait condition receives the requisite number of success signals, AWS CloudFormation resumes the creation of the stack. If the wait condition does not receive the specified number of success signals before the Timeout period expires, AWS CloudFormation assumes that the wait condition has failed and rolls the stack back.

Required: No

Type: Integer

Update requires: Updates are not supported.

`Handle`

A reference to the wait condition handle used to signal this wait condition. Use the `Ref` intrinsic function to specify an AWS::CloudFormation::WaitConditionHandle (p. 491) resource.

Anytime you add a WaitCondition resource during a stack update, you must associate the wait condition with a new WaitConditionHandle resource. Do not reuse an old wait condition handle that has already been defined in the template. If you reuse a wait condition handle, the wait condition might evaluate old signals from a previous create or update stack command.

Required: Yes

Type: String

Update requires: Updates are not supported.

`Timeout`

The length of time (in seconds) to wait for the number of signals that the `Count` property specifies. `Timeout` is a minimum-bound property, meaning the timeout occurs no sooner than the time you specify, but can occur shortly thereafter. The maximum time that can be specified for this property is 12 hours (43200 seconds).

Required: Yes

Type: String

Update requires: Updates are not supported.

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`Data`

Returns: A JSON object that contains the `UniqueId` and `Data` values from the wait condition signal(s) for the specified wait condition. For more information about wait condition signals, see Wait Condition Signal JSON Format (p. 231).

Example return value for a wait condition with 2 signals:

```
{ "Signal1" : "Step 1 complete." , "Signal2" : "Step 2 complete." }
```

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Examples

WaitCondition that waits for the desired number of instances in a web server group

JSON

```
"WebServerGroup" : {
   "Type" : "AWS::AutoScaling::AutoScalingGroup",
   "Properties" : {
      "AvailabilityZones" : { "Fn::GetAZs" : "" },
      "LaunchConfigurationName" : { "Ref" : "LaunchConfig" },
      "MinSize" : "1",
      "MaxSize" : "5",
      "DesiredCapacity" : { "Ref" : "WebServerCapacity" },
      "LoadBalancerNames" : [ { "Ref" : "ElasticLoadBalancer" } ]
   }
},

"WaitHandle" : {
   "Type" : "AWS::CloudFormation::WaitConditionHandle"
},

"WaitCondition" : {
   "Type" : "AWS::CloudFormation::WaitCondition",
   "DependsOn" : "WebServerGroup",
   "Properties" : {
      "Handle"  : { "Ref" : "WaitHandle" },
      "Timeout" : "300",
      "Count"   : { "Ref" : "WebServerCapacity" }
   }
}
```

YAML

```
WebServerGroup:
  Type: "AWS::AutoScaling::AutoScalingGroup"
  Properties:
    AvailabilityZones:
      Fn::GetAZs: ""
    LaunchConfigurationName:
      Ref: "LaunchConfig"
    MinSize: "1"
    MaxSize: "5"
    DesiredCapacity:
      Ref: "WebServerCapacity"
    LoadBalancerNames:
      -
        Ref: "ElasticLoadBalancer"
WaitHandle:
  Type: "AWS::CloudFormation::WaitConditionHandle"
WaitCondition:
  Type: "AWS::CloudFormation::WaitCondition"
  DependsOn: "WebServerGroup"
  Properties:
    Handle:
```

```
      Ref: "WaitHandle"
    Timeout: "300"
    Count:
      Ref: "WebServerCapacity"
```

See Also

- Creating Wait Conditions in a Template (p. 228)
- DependsOn Attribute (p. 1298)

AWS::CloudFormation::WaitConditionHandle

> **Important**
> For Amazon EC2 and Auto Scaling resources, we recommend that you use a CreationPolicy attribute instead of wait conditions. Add a CreationPolicy attribute to those resources, and use the cfn-signal helper script to signal when an instance creation process has completed successfully. For more information, see Deploying Applications on Amazon EC2 with AWS CloudFormation (p. 212).

The AWS::CloudFormation::WaitConditionHandle type has no properties. When you reference the WaitConditionHandle resource by using the `Ref` function, AWS CloudFormation returns a presigned URL. You pass this URL to applications or scripts that are running on your Amazon EC2 instances to send signals to that URL. An associated AWS::CloudFormation::WaitCondition (p. 487) resource checks the URL for the required number of success signals or for a failure signal.

> **Important**
> Anytime you add a `WaitCondition` resource during a stack update or update a resource with a wait condition, you must associate the wait condition with a new `WaitConditionHandle` resource. Do not reuse an old wait condition handle that has already been defined in the template. If you reuse a wait condition handle, the wait condition might evaluate old signals from a previous create or update stack command.

> **Note**
> Updates are not supported for this resource.

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::CloudFormation::WaitConditionHandle",
   "Properties" : {
   }
}
```

YAML

```
Type: "AWS::CloudFormation::WaitConditionHandle"
Properties:
```

Related Resources

For information about how to use wait conditions, see Creating Wait Conditions in a Template (p. 228).

AWS::CloudFront::Distribution

Creates an Amazon CloudFront web distribution. For general information about CloudFront distributions, see the Introduction to Amazon CloudFront in the *Amazon CloudFront Developer Guide*. For specific information about creating CloudFront web distributions, see CreateDistribution in the *Amazon CloudFront API Reference*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::CloudFront::Distribution",
   "Properties" : {
      "DistributionConfig (p. 492)" : DistributionConfig
   }
}
```

YAML

```
Type: "AWS::CloudFront::Distribution"
Properties:
  DistributionConfig (p. 492):
    DistributionConfig
```

Properties

`DistributionConfig`

The distribution's configuration information.

Required: Yes

Type: DistributionConfig (p. 1040) type

Update requires: No interruption (p. 90)

Return Values

Ref

Returns: The CloudFront distribution ID. For example: `E27LVI50CSW06W`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`DomainName`

Returns: The domain name of the resource. For example: `d2fadu0nynjpfn.cloudfront.net`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Template Examples

To view AWS::CloudFront::Distribution snippets, see Amazon CloudFront Template Snippets (p. 248).

AWS::CloudTrail::Trail

The `AWS::CloudTrail::Trail` resource creates a trail and specifies where logs are published. An AWS CloudTrail (CloudTrail) trail can capture AWS API calls made by your AWS account and publishes the logs to an Amazon S3 bucket. For more information, see What is AWS CloudTrail? in the *AWS CloudTrail User Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::CloudTrail::Trail",
  "Properties" : {
    "CloudWatchLogsLogGroupArn" : String,
    "CloudWatchLogsRoleArn" : String,
    "EnableLogFileValidation" : Boolean,
    "IncludeGlobalServiceEvents" : Boolean,
    "IsLogging" : Boolean,
    "IsMultiRegionTrail" : Boolean,
    "KMSKeyId" : String,
    "S3BucketName" : String,
    "S3KeyPrefix" : String,
    "SnsTopicName" : String,
    "Tags" : [ Resource Tag, ... ]
  }
}
```

YAML

```
Type: "AWS::CloudTrail::Trail"
Properties:
```

```
  CloudWatchLogsLogGroupArn: String
  CloudWatchLogsRoleArn: String
  EnableLogFileValidation: Boolean
  IncludeGlobalServiceEvents: Boolean
  IsLogging: Boolean
  IsMultiRegionTrail: Boolean
  KMSKeyId: String
  S3BucketName: String
  S3KeyPrefix: String
  SnsTopicName: String
  Tags:
    - Resource Tag
```

Properties

`CloudWatchLogsLogGroupArn`

The Amazon Resource Name (ARN) of a log group to which CloudTrail logs will be delivered.

Required: Conditional. This property is required if you specify the `CloudWatchLogsRoleArn` property.

Type: String

Update requires: No interruption (p. 90)

`CloudWatchLogsRoleArn`

The role ARN that Amazon CloudWatch Logs (CloudWatch Logs) assumes to write logs to a log group. For more information, see Role Policy Document for CloudTrail to Use CloudWatch Logs for Monitoring in the *AWS CloudTrail User Guide*.

Required: No

Type: String

Update requires: No interruption (p. 90)

`EnableLogFileValidation`

Indicates whether CloudTrail validates the integrity of log files. By default, AWS CloudFormation sets this value to `false`. When you disable log file integrity validation, CloudTrail stops creating digest files. For more information, see CreateTrail in the *AWS CloudTrail API Reference*.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`IncludeGlobalServiceEvents`

Indicates whether the trail is publishing events from global services, such as IAM, to the log files. By default, AWS CloudFormation sets this value to `false`.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`IsLogging`

Indicates whether the CloudTrail trail is currently logging AWS API calls.

Required: Yes

Type: Boolean

Update requires: No interruption (p. 90)

`IsMultiRegionTrail`

Indicates whether the CloudTrail trail is created in the region in which you create the stack (`false`) or in all regions (`true`). By default, AWS CloudFormation sets this value to `false`. For more information, see How Does CloudTrail Behave Regionally and Globally? in the *AWS CloudTrail User Guide*.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`KMSKeyId`

The AWS Key Management Service (AWS KMS) key ID that you want to use to encrypt CloudTrail logs. You can specify an alias name (prefixed with `alias/`), an alias ARN, a key ARN, or a globally unique identifier.

Required: No

Type: String

Update requires: No interruption (p. 90)

`S3BucketName`

The name of the Amazon S3 bucket where CloudTrail publishes log files.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`S3KeyPrefix`

An Amazon S3 object key prefix that precedes the name of all log files.

Required: No

Type: String

Update requires: No interruption (p. 90)

`SnsTopicName`

The name of an Amazon SNS topic that is notified when new log files are published.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Tags`

An arbitrary set of tags (key–value pairs) for this trail.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90).

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a CloudTrail trail, an Amazon S3 bucket where logs are published, and an Amazon SNS topic where notifications are sent. The bucket and topic policies allow CloudTrail (from the specified regions) to publish logs to the Amazon S3 bucket and to send notifications to an email that you specify. Because CloudTrail automatically writes to the *`bucket_name`*`/AWSLogs/`*`account_ID`*`/` folder, the bucket policy grants write privileges for that prefix. For information about CloudTrail bucket policies, see Amazon S3 Bucket Policy in the *AWS CloudTrail User Guide*.

For more information about the regions that CloudTrail supports, see Supported Regions in the *AWS CloudTrail User Guide*.

JSON

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Parameters" : {
    "OperatorEmail": {
      "Description": "Email address to notify when new logs are published.",
      "Type": "String"
    }
  },
  "Resources" : {
    "S3Bucket": {
      "DeletionPolicy" : "Retain",
      "Type": "AWS::S3::Bucket",
      "Properties": {
      }
    },
    "BucketPolicy" : {
      "Type" : "AWS::S3::BucketPolicy",
      "Properties" : {
        "Bucket" : {"Ref" : "S3Bucket"},
        "PolicyDocument" : {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Sid": "AWSCloudTrailAclCheck",
              "Effect": "Allow",
              "Principal": { "Service":"cloudtrail.amazonaws.com"},
              "Action": "s3:GetBucketAcl",
              "Resource": { "Fn::Join" : ["", ["arn:aws:s3:::", {"Ref":"S3Bucket"}]]}
            },
            {
              "Sid": "AWSCloudTrailWrite",
              "Effect": "Allow",
              "Principal": { "Service":"cloudtrail.amazonaws.com"},
              "Action": "s3:PutObject",
              "Resource": { "Fn::Join" : ["", ["arn:aws:s3:::", {"Ref":"S3Bucket"}, "/
AWSLogs/", {"Ref":"AWS::AccountId"}, "/*"]]},
              "Condition": {
```

```
                "StringEquals": {
                  "s3:x-amz-acl": "bucket-owner-full-control"
                }
              }
            }
          ]
        }
      }
    },
    "Topic": {
      "Type": "AWS::SNS::Topic",
      "Properties": {
        "Subscription": [ {
          "Endpoint": { "Ref": "OperatorEmail" },
          "Protocol": "email" } ]
      }
    },
    "TopicPolicy" : {
      "Type" : "AWS::SNS::TopicPolicy",
      "Properties" : {
        "Topics" : [{"Ref":"Topic"}],
        "PolicyDocument" : {
          "Version": "2008-10-17",
          "Statement": [
            {
              "Sid": "AWSCloudTrailSNSPolicy",
              "Effect": "Allow",
              "Principal": { "Service":"cloudtrail.amazonaws.com"},
              "Resource": "*",
              "Action": "SNS:Publish"
            }
          ]
        }
      }
    },
    "myTrail" : {
      "DependsOn" : ["BucketPolicy", "TopicPolicy"],
      "Type" : "AWS::CloudTrail::Trail",
      "Properties" : {
        "S3BucketName" : {"Ref":"S3Bucket"},
        "SnsTopicName" : {"Fn::GetAtt":["Topic","TopicName"]},
        "IsLogging" : true
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Parameters:
  OperatorEmail:
    Description: "Email address to notify when new logs are published."
    Type: String
Resources:
  S3Bucket:
    DeletionPolicy: Retain
    Type: "AWS::S3::Bucket"
    Properties: {}
  BucketPolicy:
    Type: "AWS::S3::BucketPolicy"
    Properties:
      Bucket:
        Ref: S3Bucket
```

```
      PolicyDocument:
        Version: "2012-10-17"
        Statement:
          -
            Sid: "AWSCloudTrailAclCheck"
            Effect: "Allow"
            Principal:
              Service: "cloudtrail.amazonaws.com"
            Action: "s3:GetBucketAcl"
            Resource:
              !Sub |-
                arn:aws:s3:::${S3Bucket}
          -
            Sid: "AWSCloudTrailWrite"
            Effect: "Allow"
            Principal:
              Service: "cloudtrail.amazonaws.com"
            Action: "s3:PutObject"
            Resource:
              !Sub |-
                arn:aws:s3:::${S3Bucket}/AWSLogs/${AWS::AccountId}/*
            Condition:
              StringEquals:
                s3:x-amz-acl: "bucket-owner-full-control"
  Topic:
    Type: "AWS::SNS::Topic"
    Properties:
      Subscription:
        -
          Endpoint:
            Ref: OperatorEmail
          Protocol: email
  TopicPolicy:
    Type: "AWS::SNS::TopicPolicy"
    Properties:
      Topics:
        - Ref: "Topic"
      PolicyDocument:
        Version: "2008-10-17"
        Statement:
          -
            Sid: "AWSCloudTrailSNSPolicy"
            Effect: "Allow"
            Principal:
              Service: "cloudtrail.amazonaws.com"
            Resource: "*"
            Action: "SNS:Publish"
  myTrail:
    DependsOn:
      - BucketPolicy
      - TopicPolicy
    Type: "AWS::CloudTrail::Trail"
    Properties:
      S3BucketName:
        Ref: S3Bucket
      SnsTopicName:
        Fn::GetAtt:
          - Topic
          - TopicName
      IsLogging: true
```

AWS::CloudWatch::Alarm

The `AWS::CloudWatch::Alarm` type creates a CloudWatch alarm.

This type supports updates. For more information about updating this resource, see PutMetricAlarm. For more information about updating stacks, see AWS CloudFormation Stacks Updates (p. 89).

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::CloudWatch::Alarm",
   "Properties" : {
      "ActionsEnabled" : Boolean,
      "AlarmActions" : [ String, ... ],
      "AlarmDescription" : String,
      "AlarmName" : String,
      "ComparisonOperator" : String,
      "Dimensions" : [ Metric dimension, ... ],
      "EvaluationPeriods" : Integer,
      "InsufficientDataActions" : [ String, ... ],
      "MetricName" : String,
      "Namespace" : String,
      "OKActions" : [ String, ... ],
      "Period" : Integer,
      "Statistic" : String,
      "Threshold" : Double,
      "Unit" : String
   }
}
```

YAML

```
Type: "AWS::CloudWatch::Alarm"
Properties:
  ActionsEnabled: Boolean
  AlarmActions:
    - String
  AlarmDescription: String
  AlarmName: String
  ComparisonOperator: String
  Dimensions:
    - Metric dimension
  EvaluationPeriods: Integer
  InsufficientDataActions:
    - String
  MetricName: String
  Namespace: String
  OKActions:
    - String
  Period: Integer
  Statistic: String
```

```
  Threshold: Double
  Unit: String
```

Properties

`ActionsEnabled`

Indicates whether or not actions should be executed during any changes to the alarm's state.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`AlarmActions`

The list of actions to execute when this alarm transitions into an ALARM state from any other state. Each action is specified as an Amazon Resource Number (ARN). For more information about creating alarms and the actions you can specify, see Creating Amazon CloudWatch Alarms in the *Amazon CloudWatch User Guide*.

Note
For Auto Scaling scaling polices, you can specify only one policy. If you associate more than one policy, CloudWatch executes only the first scaling policy.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`AlarmDescription`

The description for the alarm.

Required: No

Type: String

Update requires: No interruption (p. 90)

`AlarmName`

A name for the alarm. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the alarm name. For more information, see Name Type (p. 1217).

Important
If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

`ComparisonOperator`

The arithmetic operation to use when comparing the specified Statistic and Threshold. The specified Statistic value is used as the first operand.

You can specify the following values: `GreaterThanOrEqualToThreshold` | `GreaterThanThreshold` | `LessThanThreshold` | `LessThanOrEqualToThreshold`

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`Dimensions`

The dimensions for the alarm's associated metric.

Required: No

Type: List of Metric Dimension (p. 1059)

Update requires: No interruption (p. 90)

`EvaluationPeriods`

The number of periods over which data is compared to the specified threshold.

Required: Yes

Type: Integer

Update requires: No interruption (p. 90)

`InsufficientDataActions`

The list of actions to execute when this alarm transitions into an INSUFFICIENT_DATA state from any other state. Each action is specified as an Amazon Resource Number (ARN). Currently the only action supported is publishing to an Amazon SNS topic or an Amazon Auto Scaling policy.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`MetricName`

The name for the alarm's associated metric. For more information about the metrics that you can specify, see Amazon CloudWatch Namespaces, Dimensions, and Metrics Reference in the *Amazon CloudWatch User Guide*.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`Namespace`

The namespace for the alarm's associated metric.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`OKActions`

The list of actions to execute when this alarm transitions into an OK state from any other state. Each action is specified as an Amazon Resource Number (ARN). Currently the only action supported is publishing to an Amazon SNS topic or an Amazon Auto Scaling policy.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`Period`

The time over which the specified statistic is applied. You must specify a time in seconds that is also a multiple of 60.

Required: Yes

Type: Integer

Update requires: No interruption (p. 90)

`Statistic`

The statistic to apply to the alarm's associated metric.

You can specify the following values: `SampleCount` | `Average` | `Sum` | `Minimum` | `Maximum`

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`Threshold`

The value against which the specified statistic is compared.

Required: Yes

Type: Double

Update requires: No interruption (p. 90)

`Unit`

The unit for the alarm's associated metric.

You can specify the following values: Seconds | Microseconds | Milliseconds | Bytes | Kilobytes | Megabytes | Gigabytes | Terabytes | Bits | Kilobits | Megabits | Gigabits | Terabits | Percent | Count | Bytes/Second | Kilobytes/Second | Megabytes/Second | Gigabytes/Second | Terabytes/Second | Bits/Second | Kilobits/Second | Megabits/Second | Gigabits/Second | Terabits/Second | Count/Second | None

Required: No

Type: String

Update requires: No interruption (p. 90)

Return Values

Ref

When you specify an AWS::CloudWatch::Alarm type as an argument to the `Ref` function, AWS CloudFormation returns the value of the `AlarmName`.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

For sample template snippets, see Amazon CloudWatch Template Snippets (p. 255).

AWS::CodeBuild::Project

The `AWS::CodeBuild::Project` resource configures how AWS CodeBuild builds your source code. For example, it tells AWS CodeBuild where to get the source code and which build environment to use.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::CodeBuild::Project",
  "Properties" : {
    "Artifacts" : Artifacts (p. 1062),
    "Description" : String,
    "EncryptionKey" : String,
    "Environment" : Environment (p. 1064),
    "Name" : String,
    "ServiceRole" : String,
    "Source" : Source (p. 1066),
    "Tags" : [ Resource Tag, ... ],
    "TimeoutInMinutes" : Integer
  }
}
```

YAML

```
Type: "AWS::CodeBuild::Project"
Properties:
  Artifacts:
    Artifacts (p. 1062)
  Description: String
  EncryptionKey: String
  Environment:
    Environment (p. 1064)
  Name: String
  ServiceRole: String
  Source:
    Source (p. 1066)
  Tags:
    - Resource Tag
  TimeoutInMinutes: Integer
```

Properties

`Artifacts`

The output settings for artifacts that the project generates during a build.

Required: Yes

Type: AWS CodeBuild Project Artifacts (p. 1062)

Update requires: No interruption (p. 90)

`Description`

A description of the project. Use the description to identify the purpose of the project.

Required: No

Type: String

Update requires: No interruption (p. 90)

`EncryptionKey`

The alias or Amazon Resource Name (ARN) of the AWS Key Management Service (AWS KMS) customer master key (CMK) that AWS CodeBuild uses to encrypt the build output. If you don't specify a value, AWS CodeBuild uses the AWS-managed CMK for Amazon Simple Storage Service.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Environment`

The build environment settings for the project, such as the environment type or the environment variables to use for the build environment.

Required: Yes

Type: AWS CodeBuild Project Environment (p. 1064)

Update requires: No interruption (p. 90)

`Name`

A name for the project. The name must be unique across all of the projects in your AWS account.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`ServiceRole`

The ARN of the service role that AWS CodeBuild uses to interact with services on your behalf.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`Source`

The source code settings for the project, such as the source code's repository type and location.

Required: Yes

Type: AWS CodeBuild Project Source (p. 1066)

Update requires: No interruption (p. 90)

`Tags`

An arbitrary set of tags (key-value pairs) for the AWS CodeBuild project.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90)

`TimeoutInMinutes`

The number of minutes after which AWS CodeBuild stops the build if it's not complete. For valid values, see the `timeoutInMinutes` field in the *AWS CodeBuild User Guide*.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the name of the AWS CodeBuild project, such as `myProjectName`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attribute and a sample return value.

`Arn`

The ARN of the AWS CodeBuild project, such as `arn:aws:codebuild:us-west-2:123456789012:project/myProjectName`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Example

The following example creates an AWS CodeBuild project.

JSON

```
{
  "Project": {
```

```
    "Type": "AWS::CodeBuild::Project",
    "Properties": {
      "Name": "myProjectName",
      "Description": "A description about my project",
      "ServiceRole": { "Fn::GetAtt": [ "ServiceRole", "Arn" ] },
      "Artifacts": {
        "Type": "no_artifacts"
      },
      "Environment": {
        "Type": "LINUX_CONTAINER",
        "ComputeType": "BUILD_GENERAL1_SMALL",
        "Image": "aws/codebuild/java:openjdk-8",
        "EnvironmentVariables": [
          {
            "Name": "varName",
            "Value": "varValue"
          }
        ]
      },
      "Source": {
        "Location": "codebuild-demo-test/0123ab9a371ebf0187b0fe5614fbb72c",
        "Type": "S3"
      },
      "TimeoutInMinutes": 10,
      "Tags": [
        {
          "Key": "Key1",
          "Value": "Value1"
        },
        {
          "Key": "Key2",
          "Value": "Value2"
        }
      ]
    }
  }
}
```

YAML

```
Project:
  Type: AWS::CodeBuild::Project
  Properties:
    Name: myProjectName
    Description: A description about my project
    ServiceRole: !GetAtt ServiceRole.Arn
    Artifacts:
      Type: no_artifacts
    Environment:
      Type: LINUX_CONTAINER
      ComputeType: BUILD_GENERAL1_SMALL
      Image: aws/codebuild/java:openjdk-8
      EnvironmentVariables:
      - Name: varName
        Value: varValue
    Source:
      Location: codebuild-demo-test/0123ab9a371ebf0187b0fe5614fbb72c
      Type: S3
    TimeoutInMinutes: 10
    Tags:
      - Key: Key1
        Value: Value1
      - Key: Key2
        Value: Value2
```

AWS::CodeCommit::Repository

The `AWS::CodeCommit::Repository` resource creates an AWS CodeCommit repository that is hosted by Amazon Web Services. For more information, see Create an AWS CodeCommit Repository in the *AWS CodeCommit User Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::CodeCommit::Repository",
  "Properties" : {
    "RepositoryDescription" : String, required,
    "RepositoryName" : String,
    "Triggers" : [ Trigger (p. 1067) ]
  }
}
```

YAML

```
Type: "AWS::CodeCommit::Repository"
Properties:
  RepositoryDescription: String
  RepositoryName: String
  Triggers:
  - Trigger (p. 1067)
```

Properties

`RepositoryDescription`

A description about the AWS CodeCommit repository. For constraints, see the CreateRepository action in the *AWS CodeCommit API Reference*.

Required: No

Type: String

Update requires: No interruption (p. 90)

`RepositoryName`

A name for the AWS CodeCommit repository.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`Triggers`

Defines the actions to take in response to events that occur in the repository. For example, you can send email notifications when someone pushes to the repository.

Required: No

Type: List of AWS CodeCommit Repository Trigger (p. 1067)

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the repository ID, such as `12a345b6-bbb7-4bb6-90b0-8c9577a2d2b9`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`Arn`

The Amazon Resource Name (ARN) of the repository, such as `arn:aws:codecommit:us-east-1:123456789012:MyDemoRepo`.

`CloneUrlHttp`

The URL to use for cloning the repository over HTTPS, such as `https://codecommit.us-east-1.amazonaws.com/v1/repos/MyDemoRepo`.

`CloneUrlSsh`

The URL to use for cloning the repository over SSH, such as `ssh://git-codecommit.us-east-1.amazonaws.com/v1/repos//v1/repos/MyDemoRepo`.

`Name`

The name of the repository, such `MyDemoRepo`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Example

The following example creates an AWS CodeCommit repository with a trigger for all events in the `Master` branch.

JSON

```
"MyRepo" : {
  "Type" : "AWS::CodeCommit::Repository",
  "Properties" : {
    "RepositoryName" : "MyRepoName",
```

```
    "RepositoryDescription" : "a description",
    "Triggers" : [
      {
        "Name" : "MasterTrigger",
        "CustomData" : "Project ID 12345",
        "DestinationArn" : { "Ref":"SNSarn" },
        "Branches" : ["Master"],
        "Events" : ["all"]
      }
    ]
  }
}
```

YAML

```
MyRepo:
  Type: AWS::CodeCommit::Repository
  Properties:
    RepositoryName: MyRepoName
    RepositoryDescription: a description
    Triggers:
    - Name: MasterTrigger
      CustomData: Project ID 12345
      DestinationArn:
        Ref: SNSarn
      Branches:
      - Master
      Events:
      - all
```

AWS::CodeDeploy::Application

The `AWS::CodeDeploy::Application` resource creates an AWS CodeDeploy application. You can use the `AWS::CodeDeploy::DeploymentGroup` resource to associate the application with an AWS CodeDeploy deployment group. For more information, see AWS CodeDeploy Deployments in the *AWS CodeDeploy User Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::CodeDeploy::Application",
  "Properties" : {
    "ApplicationName" : String
  }
}
```

YAML

```
Type: "AWS::CodeDeploy::Application"
Properties:
  ApplicationName: String
```

Properties

`ApplicationName`

A name for the application. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the application name. For more information, see Name Type (p. 1217).

Required: No

Type: String

Update requires: Updates are not supported.

Return Value

Ref

When you pass the logical ID of an `AWS::CodeDeploy::Application` resource to the intrinsic `Ref` function, the function returns the application name, such as `myapplication-a123d0d1`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates an AWS CodeDeploy application:

JSON

```
"MyApplication" : {
  "Type" : "AWS::CodeDeploy::Application"
}
```

YAML

```
MyApplication:
  Type: "AWS::CodeDeploy::Application"
```

Related Resources

For configuring your deployment and specifying your application revisions, see AWS::CodeDeploy::DeploymentConfig (p. 510) and AWS::CodeDeploy::DeploymentGroup (p. 512).

AWS::CodeDeploy::DeploymentConfig

The `AWS::CodeDeploy::DeploymentConfig` resource creates a set of deployment rules, deployment success conditions, and deployment failure conditions that AWS CodeDeploy uses during a deployment.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::CodeDeploy::DeploymentConfig",
  "Properties" : {
    "DeploymentConfigName" : String,
    "MinimumHealthyHosts" : MinimumHealthyHosts
  }
}
```

YAML

```
Type: "AWS::CodeDeploy::DeploymentConfig"
Properties:
  DeploymentConfigName: String
  MinimumHealthyHosts:
    MinimumHealthyHosts
```

Properties

`DeploymentConfigName`

A name for the deployment configuration. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the deployment configuration name. For more information, see Name Type (p. 1217).

Important
If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

`MinimumHealthyHosts`

The minimum number of healthy instances that must be available at any time during an AWS CodeDeploy deployment. For example, for a fleet of nine instances, if you specify a minimum of six healthy instances, AWS CodeDeploy deploys your application up to three instances at a time so that you always have six healthy instances. The deployment succeeds if your application successfully deploys to six or more instances; otherwise, the deployment fails.

For more information about instance health, see AWS CodeDeploy Instance Health in the *AWS CodeDeploy User Guide*.

Required: Yes

Type: AWS CodeDeploy DeploymentConfig MinimumHealthyHosts (p. 1068)

Update requires: Replacement (p. 90)

Return Value

Ref

When you pass the logical ID of an `AWS::CodeDeploy::DeploymentConfig` resource to the intrinsic `Ref` function, the function returns the deployment configuration name, such as `mydeploymentconfig-a123d0d1`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example requires at least 75% of the fleet to be healthy. For example, if you had a fleet of four instances, the deployment proceeds one instance at a time.

JSON

```
"TwentyFivePercentAtATime" : {
  "Type" : "AWS::CodeDeploy::DeploymentConfig",
  "Properties" : {
    "MinimumHealthyHosts" : {
      "Type" : "FLEET_PERCENT",
      "Value" : "75"
    }
  }
}
```

YAML

```
TwentyFivePercentAtATime:
  Type: "AWS::CodeDeploy::DeploymentConfig"
  Properties:
    MinimumHealthyHosts:
      Type: "FLEET_PERCENT"
      Value: 75
```

AWS::CodeDeploy::DeploymentGroup

The `AWS::CodeDeploy::DeploymentGroup` resource creates an AWS CodeDeploy deployment group that details which application revision to use and which instances your application revisions are deployed to.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::CodeDeploy::DeploymentGroup",
  "Properties" : {
    "ApplicationName" : String,
    "AutoScalingGroups" : [ String, ... ],
    "Deployment" : Deployment,
    "DeploymentConfigName" : String,
    "DeploymentGroupName" : String,
    "Ec2TagFilters" : [ Ec2TagFilters, ... ],
    "OnPremisesInstanceTagFilters" : [ OnPremisesInstanceTagFilters, ... ],
    "ServiceRoleArn" : String
  }
}
```

YAML

```
Type: "AWS::CodeDeploy::DeploymentGroup"
Properties:
  ApplicationName: String
  AutoScalingGroups:
    - String
  Deployment:
    Deployment
  DeploymentConfigName: String
  DeploymentGroupName: String
  Ec2TagFilters:
    - Ec2TagFilters
  OnPremisesInstanceTagFilters:
    - OnPremisesInstanceTagFilters
  ServiceRoleArn: String
```

Properties

`ApplicationName`

The name of an AWS CodeDeploy application for this deployment group.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`AutoScalingGroups`

A list of associated Auto Scaling groups that AWS CodeDeploy automatically deploys revisions to when new instances are created.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`Deployment`

The application revision that will be deployed to this deployment group.

Required: No

Type: AWS CodeDeploy DeploymentGroup Deployment (p. 1069)

Update requires: No interruption (p. 90)

`DeploymentConfigName`

A deployment configuration name or a predefined configuration name. With predefined configurations, you can deploy application revisions to one instance at a time, half of the instances at a time, or all the instances at once. For more information and valid values, see the `DeploymentConfigName` parameter for the CreateDeploymentGroup action in the *AWS CodeDeploy API Reference*.

Required: No

Type: String

Update requires: No interruption (p. 90)

`DeploymentGroupName`

A name for the deployment group. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the deployment group name. For more information, see Name Type (p. 1217).

Important
If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Ec2TagFilters`

The Amazon EC2 tags to filter on. AWS CodeDeploy includes all instances that match the tag filter with this deployment group.

Required: No

Type: AWS CodeDeploy DeploymentGroup Ec2TagFilters (p. 1072)

Update requires: No interruption (p. 90)

`OnPremisesInstanceTagFilters`

The on-premises instance tags to filter on. AWS CodeDeploy includes all on-premises instances that match the tag filter with this deployment group. To register on-premises instances with AWS CodeDeploy, see Configure Existing On-Premises Instances by Using AWS CodeDeploy in the *AWS CodeDeploy User Guide*.

Required: No

Type: AWS CodeDeploy DeploymentGroup OnPremisesInstanceTagFilters (p. 1073)

Update requires: No interruption (p. 90)

`ServiceRoleArn`

A service role Amazon Resource Name (ARN) that grants AWS CodeDeploy permission to make calls to AWS services on your behalf. For more information, see Create a Service Role for AWS CodeDeploy in the *AWS CodeDeploy User Guide.*

Note
In some cases, you might need to add a dependency on the service role's policy. For more information, see IAM role policy in DependsOn Attribute (p. 1298).

Required: Yes

Type: String

Update requires: No interruption (p. 90)

Return Value

Ref

When you pass the logical ID of an `AWS::CodeDeploy::DeploymentGroup` resource to the intrinsic `Ref` function, the function returns the deployment group name, such as `mydeploymentgroup-a123d0d1`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

Revision in GitHub

The following example creates a deployment group that is associated with Auto Scaling groups and uses an application revision that is stored in a GitHub repository. You specify the repository information as input parameters.

JSON

```
"DeploymentGroup" : {
  "Type" : "AWS::CodeDeploy::DeploymentGroup",
  "Properties" : {
    "ApplicationName" : {"Ref" : "ApplicationName"},
    "AutoScalingGroups" : [ {"Ref" : "CodeDeployAutoScalingGroups" } ],
    "Deployment" : {
      "Description" : "A sample deployment",
      "IgnoreApplicationStopFailures" : "true",
      "Revision" : {
        "RevisionType" : "GitHub",
        "GitHubLocation" : {
          "CommitId" : {"Ref" : "CommitId"},
          "Repository" : {"Ref" : "Repository"}
        }
      }
    },
    "ServiceRoleArn" : {"Ref" : "RoleArn"}
  }
}
```

YAML

```
DeploymentGroup:
  Type: "AWS::CodeDeploy::DeploymentGroup"
  Properties:
    ApplicationName:
      Ref: "ApplicationName"
    AutoScalingGroups:
      - Ref: CodeDeployAutoScalingGroups
    Deployment:
```

```
      Description: "A sample deployment"
      IgnoreApplicationStopFailures: true
      Revision:
        RevisionType: GitHub
        GitHubLocation:
          CommitId:
            Ref: CommitId
          Repository:
            Ref: Repository
    ServiceRoleArn:
      Ref: RoleArn
```

Associate EC2 Instances

The following example creates a deployment group that uses instance tags to associate EC2 instances with the deployment group. The deployment group uses an application revision that is stored in an S3 bucket.

JSON

```
"DeploymentGroup" : {
  "Type" : "AWS::CodeDeploy::DeploymentGroup",
  "Properties" : {
    "ApplicationName" : {"Ref" : "Application"},
    "Deployment" : {
      "Description" : "First time",
      "IgnoreApplicationStopFailures" : "true",
      "Revision" : {
        "RevisionType" : "S3",
        "S3Location" : {
          "Bucket" : {"Ref" : "Bucket"},
          "Key" : {"Ref" : "Key"},
          "BundleType" : "Zip",
          "ETag" : {"Ref" : "ETag"},
          "Version" : {"Ref" : "Version"}
        }
      }
    },
    "Ec2TagFilters" : [{
      "Key" : {"Ref" : "TagKey"},
      "Value" : {"Ref" : "TagValue"},
      "Type" : "KEY_AND_VALUE"
    }],
    "ServiceRoleArn" : {"Ref" : "RoleArn"}
  }
}
```

YAML

```
DeploymentGroup:
  Type: "AWS::CodeDeploy::DeploymentGroup"
  Properties:
    ApplicationName:
      Ref: "Application"
    Deployment:
      Description: "First time"
      IgnoreApplicationStopFailures: true
      Revision:
        RevisionType: S3
        S3Location:
          Bucket:
            Ref: Bucket
          Key:
```

```
          Ref: Key
        BundleType: Zip
        ETag:
          Ref: ETag
        Version:
          Ref: Version
    Ec2TagFilters:
      -
        Key:
          Ref: TagKey
        Value:
          Ref: TagValue
        Type: "KEY_AND_VALUE"
    ServiceRoleArn:
      Ref: RoleArn
```

AWS::CodePipeline::CustomActionType

The `AWS::CodePipeline::CustomActionType` resource creates a custom action for activities that aren't included in the AWS CodePipeline default actions, such as running an internally developed build process or a test suite. You can use these custom actions in the stage of a pipeline (p. 520). For more information, see Create and Add a Custom Action in AWS CodePipeline in the *AWS CodePipeline User Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::CodePipeline::CustomActionType",
  "Properties" : {
    "Category" : String,
    "ConfigurationProperties" : [ ConfigurationProperties, ... ],
    "InputArtifactDetails" : ArtifactDetails,
    "OutputArtifactDetails" : ArtifactDetails,
    "Provider" : String,
    "Settings" : Settings,
    "Version" : String
  }
}
```

YAML

```
Type: "AWS::CodePipeline::CustomActionType"
Properties:
  Category: String,
  ConfigurationProperties:
    - ConfigurationProperties
  InputArtifactDetails:
    ArtifactDetails
```

```
  OutputArtifactDetails:
    ArtifactDetails
  Provider: String
  Settings:
    Settings
  Version: String
```

Properties

`Category`

The category of the custom action, such as a source action or a build action. For valid values, see CreateCustomActionType in the *AWS CodePipeline API Reference*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`ConfigurationProperties`

The configuration properties for the custom action.

Required: No

Type: List of AWS CodePipeline CustomActionType ConfigurationProperties (p. 1075)

Update requires: Replacement (p. 90)

`InputArtifactDetails`

The input artifact details for this custom action.

Required: Yes

Type: AWS CodePipeline CustomActionType ArtifactDetails (p. 1074)

Update requires: Replacement (p. 90)

`OutputArtifactDetails`

The output artifact details for this custom action.

Required: Yes

Type: AWS CodePipeline CustomActionType ArtifactDetails (p. 1074)

Update requires: Replacement (p. 90)

`Provider`

The name of the service provider that AWS CodePipeline uses for this custom action.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Settings`

URLs that provide users information about this custom action.

Required: No

Type: AWS CodePipeline CustomActionType Settings (p. 1076)

Update requires: Replacement (p. 90)

`Version`

The version number of this custom action.

Required: No

Type: String

Update requires: Replacement (p. 90)

Return Value

Ref

When you pass the logical ID of an `AWS::CodePipeline::CustomActionType` resource to the intrinsic `Ref` function, the function returns the custom action name, such as `custo-MyCus-A1BCDEFGHIJ2`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example is a custom build action that requires users to specify one property: a project name.

JSON

```
"MyCustomActionType": {
  "Type": "AWS::CodePipeline::CustomActionType",
  "Properties": {
    "Category": "Build",
    "Provider": "My-Build-Provider-Name",
    "Version": { "Ref" : "Version" },
    "ConfigurationProperties": [
      {
        "Description": "The name of the build project must be provided when this action is
 added to the pipeline.",
        "Key": "true",
        "Name": "MyProjectName",
        "Queryable": "false",
        "Required": "true",
        "Secret": "false",
        "Type": "String"
      }
    ],
    "InputArtifactDetails": {
      "MaximumCount": "1",
      "MinimumCount": "1"
    },
    "OutputArtifactDetails": {
      "MaximumCount": { "Ref" : "MaximumCountForOutputArtifactDetails" },
      "MinimumCount": "0"
    },
    "Settings": {
      "EntityUrlTemplate": "https://my-build-instance/job/{Config:ProjectName}/",
      "ExecutionUrlTemplate": "https://my-build-instance/job/{Config:ProjectName}/
lastSuccessfulBuild/{ExternalExecutionId}/"
    }
  }
```

```
}
```

YAML

```
MyCustomActionType:
  Type: "AWS::CodePipeline::CustomActionType"
  Properties:
    Category: Build
    Provider: "My-Build-Provider-Name"
    Version:
      Ref: Version
    ConfigurationProperties:
      -
        Description: "The name of the build project must be provided when this action is
 added to the pipeline."
        Key: true
        Name: MyProjectName
        Queryable: false
        Required: true
        Secret: false
        Type: String
    InputArtifactDetails:
      MaximumCount: 1
      MinimumCount: 1
    OutputArtifactDetails:
      MaximumCount:
        Ref: MaximumCountForOutputArtifactDetails
      MinimumCount: 0
    Settings:
      EntityUrlTemplate: "https://my-build-instance/job/{Config:ProjectName}/"
      ExecutionUrlTemplate: "https://my-build-instance/job/{Config:ProjectName}/
lastSuccessfulBuild/{ExternalExecutionId}/"
```

AWS::CodePipeline::Pipeline

The `AWS::CodePipeline::Pipeline` resource creates an AWS CodePipeline pipeline that describes how software changes go through a release process. For more information, see What Is AWS CodePipeline? in the *AWS CodePipeline User Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::CodePipeline::Pipeline",
  "Properties" : {
    "ArtifactStore" : ArtifactStore,
    "DisableInboundStageTransitions" : [ DisableInboundStageTransitions, ... ],
    "Name" : String,
```

```
    "RestartExecutionOnUpdate" : Boolean,
    "RoleArn" : String,
    "Stages" : [ Stages, ... ]
  }
}
```

YAML

```
Type: "AWS::CodePipeline::Pipeline"
Properties:
  ArtifactStore:
    ArtifactStore
  DisableInboundStageTransitions:
    - DisableInboundStageTransitions
  Name: String
  RestartExecutionOnUpdate: Boolean
  RoleArn: String
  Stages:
    - Stages
```

Properties

`ArtifactStore`

The Amazon Simple Storage Service (Amazon S3) location where AWS CodePipeline stores pipeline artifacts. For more information, see Create an Amazon S3 Bucket for Your Application in the *AWS CodePipeline User Guide*.

Required: Yes

Type: AWS CodePipeline Pipeline ArtifactStore (p. 1077)

Update requires: No interruption (p. 90)

`DisableInboundStageTransitions`

Prevents artifacts in a pipeline from transitioning to the stage that you specified. This enables you to manually control transitions.

Required: No

Type: List of AWS CodePipeline Pipeline DisableInboundStageTransitions (p. 1079)

Update requires: No interruption (p. 90)

`Name`

The name of your AWS CodePipeline pipeline.

Required: No

Type: String

Update requires: No interruption (p. 90)

`RestartExecutionOnUpdate`

Indicates whether to rerun the AWS CodePipeline pipeline after you update it.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`RoleArn`

A service role Amazon Resource Name (ARN) that grants AWS CodePipeline permission to make calls to AWS services on your behalf. For more information, see AWS CodePipeline Access Permissions Reference in the *AWS CodePipeline User Guide*.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`Stages`

Defines the AWS CodePipeline pipeline stages.

Required: Yes

Type: AWS CodePipeline Pipeline Stages (p. 1080)

Update requires: No interruption (p. 90)

Return Value

Ref

When you pass the logical ID of an `AWS::CodePipeline::Pipeline` resource to the intrinsic `Ref` function, the function returns the pipeline name, such as `mysta-MyPipeline-A1BCDEFGHIJ2`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a pipeline with a source, beta, and release stage. For the source stage, AWS CodePipeline detects changes to the application that is stored in the S3 bucket and pulls them into the pipeline. The beta stage deploys those changes to EC2 instances by using AWS CodeDeploy. For the release stage, inbound transitions are disabled, which enables you to control when the changes are ready to be deployed to release.

JSON

```
"AppPipeline": {
  "Type": "AWS::CodePipeline::Pipeline",
  "Properties": {
    "RoleArn": { "Ref" : "CodePipelineServiceRole" },
    "Stages": [
      {
        "Name": "Source",
        "Actions": [
          {
            "Name": "SourceAction",
            "ActionTypeId": {
              "Category": "Source",
              "Owner": "AWS",
              "Version": "1",
              "Provider": "S3"
            },
            "OutputArtifacts": [
              {
```

```
                "Name": "SourceOutput"
              }
            ],
            "Configuration": {
              "S3Bucket": { "Ref" : "SourceS3Bucket" },
              "S3ObjectKey": { "Ref" : "SourceS3ObjectKey" }
            },
            "RunOrder": 1
          }
        ]
      },
      {
        "Name": "Beta",
        "Actions": [
          {
            "Name": "BetaAction",
            "InputArtifacts": [
              {
                "Name": "SourceOutput"
              }
            ],
            "ActionTypeId": {
              "Category": "Deploy",
              "Owner": "AWS",
              "Version": "1",
              "Provider": "CodeDeploy"
            },
            "Configuration": {
              "ApplicationName": {"Ref" : "ApplicationName"},
              "DeploymentGroupName": {"Ref" : "DeploymentGroupName"}
            },
            "RunOrder": 1
          }
        ]
      },
      {
        "Name": "Release",
        "Actions": [
          {
            "Name": "ReleaseAction",
            "InputArtifacts": [
              {
                "Name": "SourceOutput"
              }
            ],
            "ActionTypeId": {
              "Category": "Deploy",
              "Owner": "AWS",
              "Version": "1",
              "Provider": "CodeDeploy"
            },
            "Configuration": {
              "ApplicationName": {"Ref" : "ApplicationName"},
              "DeploymentGroupName": {"Ref" : "DeploymentGroupName"}
            },
            "RunOrder": 1
          }
        ]
      }
    ],
    "ArtifactStore": {
      "Type": "S3",
      "Location": { "Ref" : "ArtifactStoreS3Location" }
    },
    "DisableInboundStageTransitions": [
      {
```

```
        "StageName": "Release",
        "Reason": "Disabling the transition until integration tests are completed"
      }
    ]
  }
}
```

YAML

```
AppPipeline:
  Type: "AWS::CodePipeline::Pipeline"
  Properties:
    RoleArn:
      Ref: CodePipelineServiceRole
    Stages:
      -
        Name: Source
        Actions:
          -
            Name: SourceAction
            ActionTypeId:
              Category: Source
              Owner: AWS
              Version: 1
              Provider: S3
            OutputArtifacts:
              -
                Name: SourceOutput
            Configuration:
              S3Bucket:
                Ref: SourceS3Bucket
              S3ObjectKey:
                Ref: SourceS3ObjectKey
            RunOrder: 1
      -
        Name: Beta
        Actions:
          -
            Name: BetaAction
            InputArtifacts:
              -
                Name: SourceOutput
            ActionTypeId:
              Category: Deploy
              Owner: AWS
              Version: 1
              Provider: CodeDeploy
            Configuration:
              ApplicationName:
                Ref: ApplicationName
              DeploymentGroupName:
                Ref: DeploymentGroupName
            RunOrder: 1
      -
        Name: Release
        Actions:
          -
            Name: ReleaseAction
            InputArtifacts:
              -
                Name: SourceOutput
            ActionTypeId:
              Category: Deploy
              Owner: AWS
```

```
            Version: 1
            Provider: CodeDeploy
          Configuration:
            ApplicationName:
              Ref: ApplicationName
            DeploymentGroupName:
              Ref: DeploymentGroupName
          RunOrder: 1
  ArtifactStore:
    Type: S3
    Location:
      Ref: ArtifactStoreS3Location
  DisableInboundStageTransitions:
    -
      StageName: Release
      Reason: "Disabling the transition until integration tests are completed"
```

AWS::Config::ConfigRule

The `AWS::Config::ConfigRule` resource uses an AWS Lambda (Lambda) function that evaluates configuration items to assess whether your AWS resources comply with your specified configurations. This function can run when AWS Config detects a configuration change or delivers a configuration snapshot. The resources this function evaluates must be in the recording group. For more information, see Evaluating AWS Resource Configurations with AWS Config in the *AWS Config Developer Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Config::ConfigRule",
  "Properties" : {
    "ConfigRuleName" : String,
    "Description" : String,
    "InputParameters" : { ParameterName : Value },
    "MaximumExecutionFrequency" : String,
    "Scope" : Scope,
    "Source" : Source
  }
}
```

YAML

```
Type: "AWS::Config::ConfigRule"
Properties:
  ConfigRuleName: String
  Description: String
  InputParameters:
    ParameterName : Value
```

```
  MaximumExecutionFrequency: String
  Scope:
    Scope
  Source:
    Source
```

Properties

`ConfigRuleName`

A name for the AWS Config rule. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the rule name. For more information, see Name Type (p. 1217).

Required: No

Type: String

Update requires: Replacement (p. 90)

`Description`

A description about this AWS Config rule.

Required: No

Type: String

Update requires: No interruption (p. 90)

`InputParameters`

Input parameter values that are passed to the AWS Config rule (Lambda function).

Required: No

Type: JSON object

Update requires: No interruption (p. 90)

`MaximumExecutionFrequency`

The maximum frequency at which the AWS Config rule runs evaluations. For valid values, see the ConfigRule data type in the *AWS Config API Reference*.

If the rule runs an evaluation when AWS Config delivers a configuration snapshot, the rule cannot run more frequently than the snapshot delivery frequency. Set an execution frequency value that is equal to or greater than the value of the snapshot delivery frequency, which is a property the AWS::Config::DeliveryChannel (p. 533) resource.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Scope`

Defines which AWS resources will trigger an evaluation when their configurations change. The scope can include one or more resource types, a combination of a tag key and value, or a combination of one resource type and one resource ID. Specify a scope to constrain the resources that are evaluated. If you don't specify a scope, the rule evaluates all resources in the recording group.

Required: No

Type: AWS Config ConfigRule Scope (p. 1085)

Update requires: No interruption (p. 90)

`Source`

Specifies the rule owner, the rule identifier, and the events that cause the function to evaluate your AWS resources.

Required: Yes

Type: AWS Config ConfigRule Source (p. 1086)

Update requires: No interruption (p. 90)

Return Values

Ref

When you pass the logical ID of an `AWS::Config::ConfigRule` resource to the intrinsic `Ref` function, the function returns the rule name, such as `mystack-MyConfigRule-12ABCFPXHV4OV`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`Arn`

The Amazon Resource Name (ARN) of the AWS Config rule, such as `arn:aws:config:us-east-1:123456789012:config-rule/config-rule-a1bzhi`.

`ConfigRuleId`

The ID of the AWS Config rule, such as `config-rule-a1bzhi`.

`Compliance.Type`

The compliance status of an AWS Config rule, such as `COMPLIANT` or `NON_COMPLIANT`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Examples

The following example uses an AWS managed rule that checks whether EC2 volumes resource types have a `CostCenter` tag.

JSON

```
"ConfigRuleForVolumeTags": {
  "Type": "AWS::Config::ConfigRule",
  "Properties": {
    "InputParameters": {"tag1Key": "CostCenter"},
    "Scope": {
      "ComplianceResourceTypes": ["AWS::EC2::Volume"]
    },
    "Source": {
```

```
      "Owner": "AWS",
      "SourceIdentifier": "REQUIRED_TAGS"
    }
  }
}
```

YAML

```
ConfigRuleForVolumeTags:
  Type: "AWS::Config::ConfigRule"
  Properties:
    InputParameters:
      tag1Key: CostCenter
    Scope:
      ComplianceResourceTypes:
        - "AWS::EC2::Volume"
    Source:
      Owner: AWS
      SourceIdentifier: "REQUIRED_TAGS"
```

Rule Using Lambda Function

The following example creates a custom configuration rule that uses a Lambda function. The function checks whether an EC2 volume has the `AutoEnableIO` property set to true. Note that the configuration rule has a dependency on the Lambda policy so that the rule calls the function only after it's permitted to do so.

JSON

```
"ConfigPermissionToCallLambda": {
  "Type": "AWS::Lambda::Permission",
  "Properties": {
    "FunctionName": {"Fn::GetAtt": ["VolumeAutoEnableIOComplianceCheck", "Arn"]},
    "Action": "lambda:InvokeFunction",
    "Principal": "config.amazonaws.com"
  }
},
"VolumeAutoEnableIOComplianceCheck": {
  "Type": "AWS::Lambda::Function",
  "Properties": {
    "Code": {
      "ZipFile":  {"Fn::Join": ["\n", [
        "var aws  = require('aws-sdk');",
        "var config = new aws.ConfigService();",
        "var ec2 = new aws.EC2();",

        "exports.handler = function(event, context) {",
        "    compliance = evaluateCompliance(event, function(compliance, event) {",
        "        var configurationItem =
 JSON.parse(event.invokingEvent).configurationItem;",

        "        var putEvaluationsRequest = {",
        "            Evaluations: [{",
        "                ComplianceResourceType: configurationItem.resourceType,",
        "                ComplianceResourceId: configurationItem.resourceId,",
        "                ComplianceType: compliance,",
        "                OrderingTimestamp:
 configurationItem.configurationItemCaptureTime",
        "            }],",
        "            ResultToken: event.resultToken",
        "        };",

        "        config.putEvaluations(putEvaluationsRequest, function(err, data) {",
```

```
        "            if (err) context.fail(err);",
        "            else context.succeed(data);",
        "        });",
        "    });",
        "};",

        "function evaluateCompliance(event, doReturn) {",
        "    var configurationItem = JSON.parse(event.invokingEvent).configurationItem;",
        "    var status = configurationItem.configurationItemStatus;",
        "    if (configurationItem.resourceType !== 'AWS::EC2::Volume' ||
 event.eventLeftScope || (status !== 'OK' && status !== 'ResourceDiscovered'))",
        "        doReturn('NOT_APPLICABLE', event);",
        "    else ec2.describeVolumeAttribute({VolumeId: configurationItem.resourceId,
 Attribute: 'autoEnableIO'}, function(err, data) {",
        "        if (err) context.fail(err);",
        "        else if (data.AutoEnableIO.Value) doReturn('COMPLIANT', event);",
        "        else doReturn('NON_COMPLIANT', event);",
        "    });",
        "}"
      ]]}
    },
    "Handler": "index.handler",
    "Runtime": "nodejs4.3",
    "Timeout": "30",
    "Role": {"Fn::GetAtt": ["LambdaExecutionRole", "Arn"]}
  }
},
"ConfigRuleForVolumeAutoEnableIO": {
  "Type": "AWS::Config::ConfigRule",
  "Properties": {
    "ConfigRuleName": "ConfigRuleForVolumeAutoEnableIO",
    "Scope": {
      "ComplianceResourceId": {"Ref": "Ec2Volume"},
      "ComplianceResourceTypes": ["AWS::EC2::Volume"]
    },
    "Source": {
      "Owner": "CUSTOM_LAMBDA",
      "SourceDetails": [{
          "EventSource": "aws.config",
          "MessageType": "ConfigurationItemChangeNotification"
      }],
      "SourceIdentifier": {"Fn::GetAtt": ["VolumeAutoEnableIOComplianceCheck", "Arn"]}
    }
  },
  "DependsOn": "ConfigPermissionToCallLambda"
}
```

YAML

```
ConfigPermissionToCallLambda:
  Type: "AWS::Lambda::Permission"
  Properties:
    FunctionName:
      Fn::GetAtt:
        - VolumeAutoEnableIOComplianceCheck
        - Arn
    Action: "lambda:InvokeFunction"
    Principal: "config.amazonaws.com"
VolumeAutoEnableIOComplianceCheck:
  Type: "AWS::Lambda::Function"
  Properties:
    Code:
      ZipFile:
        !Sub |
```

```
          var aws  = require('aws-sdk');
          var config = new aws.ConfigService();
          var ec2 = new aws.EC2();
          exports.handler = function(event, context) {
              compliance = evaluateCompliance(event, function(compliance, event) {
                    var configurationItem =
 JSON.parse(event.invokingEvent).configurationItem;
                    var putEvaluationsRequest = {
                        Evaluations: [{
                            ComplianceResourceType: configurationItem.resourceType,
                            ComplianceResourceId: configurationItem.resourceId,
                            ComplianceType: compliance,
                            OrderingTimestamp:
 configurationItem.configurationItemCaptureTime
                        }],
                        ResultToken: event.resultToken
                    };
                    config.putEvaluations(putEvaluationsRequest, function(err, data) {
                        if (err) context.fail(err);
                        else context.succeed(data);
                    });
                });
            };
            function evaluateCompliance(event, doReturn) {
                var configurationItem = JSON.parse(event.invokingEvent).configurationItem;
                var status = configurationItem.configurationItemStatus;
                if (configurationItem.resourceType !== 'AWS::EC2::Volume' ||
 event.eventLeftScope || (status !== 'OK' && status !== 'ResourceDiscovered'))
                    doReturn('NOT_APPLICABLE', event);
                else ec2.describeVolumeAttribute({VolumeId: configurationItem.resourceId,
 Attribute: 'autoEnableIO'}, function(err, data) {
                    if (err) context.fail(err);
                    else if (data.AutoEnableIO.Value) doReturn('COMPLIANT', event);
                    else doReturn('NON_COMPLIANT', event);
                });
            }
    Handler: "index.handler"
    Runtime: nodejs4.3
    Timeout: 30
    Role:
      Fn::GetAtt:
        - LambdaExecutionRole
        - Arn
ConfigRuleForVolumeAutoEnableIO:
  Type: "AWS::Config::ConfigRule"
  Properties:
    ConfigRuleName: ConfigRuleForVolumeAutoEnableIO
    Scope:
      ComplianceResourceId:
        Ref: Ec2Volume
      ComplianceResourceTypes:
        - "AWS::EC2::Volume"
    Source:
      Owner: "CUSTOM_LAMBDA"
      SourceDetails:
        -
          EventSource: "aws.config"
          MessageType: "ConfigurationItemChangeNotification"
      SourceIdentifier:
        Fn::GetAtt:
          - VolumeAutoEnableIOComplianceCheck
          - Arn
  DependsOn: ConfigPermissionToCallLambda
```

AWS::Config::ConfigurationRecorder

The `AWS::Config::ConfigurationRecorder` resource describes the AWS resource types for which AWS Config records configuration changes. The configuration recorder stores the configurations of the supported resources in your account as configuration items.

AWS CloudFormation starts the recorder as soon as the delivery channel becomes available. To stop the recorder, delete the configuration recorder from your stack.

> **Note**
> If you create this resource, you must also create or have an `AWS::Config::DeliveryChannel` resource already running in your account. These two interdependent resources must be present to successfully create both resources.

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Config::ConfigurationRecorder",
  "Properties" : {
    "Name" : String,
    "RecordingGroup" : Recording group,
    "RoleARN" : String
  }
}
```

YAML

```
Type: "AWS::Config::ConfigurationRecorder"
Properties:
  Name: String
  RecordingGroup:
    Recording group
  RoleARN: String
```

Properties

`Name`

A name for the configuration recorder. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the configuration recorder name. For more information, see Name Type (p. 1217).

> **Note**
> After you create a configuration recorder, you cannot rename it. If you don't want a AWS CloudFormation-generated name, specify a value for this property.

If you specify the name of an existing configuration recorder, AWS CloudFormation uses that recorder.

Required: No

Type: String

Update requires: Updates are not supported.

`RecordingGroup`

Indicates whether to record configurations for all supported resources or for a list of resource types. The resource types that you list must be supported by AWS Config.

Required: No

Type: AWS Config ConfigurationRecorder RecordingGroup (p. 1088)

Update requires: No interruption (p. 90)

`RoleARN`

The Amazon Resource Name (ARN) of the AWS Identity and Access Management (IAM) role that is used to make read or write requests to the delivery channel that you specify and to get configuration details for supported AWS resources. For more information, see Permissions for the AWS Config IAM Role in the *AWS Config Developer Guide*.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

Return Values

Ref

When you pass the logical ID of an `AWS::Config::ConfigurationRecorder` resource to the intrinsic `Ref` function, the function returns the configuration recorder name, such as `default`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a configuration recorder for EC2 volumes.

JSON

```
"ConfigRecorder": {
  "Type": "AWS::Config::ConfigurationRecorder",
  "Properties": {
    "Name": "default",
    "RecordingGroup": {
      "ResourceTypes": ["AWS::EC2::Volume"]
    },
    "RoleARN": {"Fn::GetAtt": ["ConfigRole", "Arn"]}
  }
}
```

YAML

```
ConfigRecorder:
  Type: "AWS::Config::ConfigurationRecorder"
  Properties:
    Name: default
    RecordingGroup:
      ResourceTypes:
        - "AWS::EC2::Volume"
```

```
    RoleARN:
      Fn::GetAtt:
        - ConfigRole
        - Arn
```

AWS::Config::DeliveryChannel

The `AWS::Config::DeliveryChannel` resource describes where AWS Config stores configuration changes to an AWS resource.

Note

If you create this resource, you must also create or have an `AWS::Config::ConfigurationRecorder` resource already running in your account. These two interdependent resources must be present to successfully create both resources.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Config::DeliveryChannel",
  "Properties" : {
    "ConfigSnapshotDeliveryProperties" : Config snapshot delivery properties,
    "Name" : String,
    "S3BucketName" : String,
    "S3KeyPrefix" : String,
    "SnsTopicARN" : String
  }
}
```

YAML

```
Type: "AWS::Config::DeliveryChannel"
Properties:
  ConfigSnapshotDeliveryProperties:
    Config snapshot delivery properties
  Name: String
  S3BucketName: String
  S3KeyPrefix: String
  SnsTopicARN: String
```

Properties

`ConfigSnapshotDeliveryProperties`

Provides options for how AWS Config delivers configuration snapshots to the S3 bucket in your delivery channel.

Required: No

Type: AWS Config DeliveryChannel ConfigSnapshotDeliveryProperties (p. 1089)

Update requires: No interruption (p. 90)

`Name`

A name for the delivery channel. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the delivery channel name. For more information, see Name Type (p. 1217).

Required: No

Type: String

Update requires: Updates are not supported.. To change the name, you must run two separate updates. Delete this resource in the first update and then recreate it with a new name in the second update.

`S3BucketName`

The name of an S3 bucket where you want to store configuration history for the delivery channel.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`S3KeyPrefix`

A key prefix (folder) for the specified S3 bucket.

Required: No

Type: String

Update requires: No interruption (p. 90)

`SnsTopicARN`

The Amazon Resource Name (ARN) of the Amazon Simple Notification Service (Amazon SNS) topic that AWS Config delivers notifications to.

Required: No

Type: String

Update requires: No interruption (p. 90)

Return Values

Ref

When you pass the logical ID of an `AWS::Config::DeliveryChannel` resource to the intrinsic `Ref` function, the function returns the delivery channel name, such as `default`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a delivery channel that sends notifications to the specified Amazon SNS topic. The delivery channel also sends configuration changes and snapshots to the specified S3 bucket.

JSON

```
"DeliveryChannel": {
  "Type": "AWS::Config::DeliveryChannel",
  "Properties": {
    "ConfigSnapshotDeliveryProperties": {
      "DeliveryFrequency": "Six_Hours"
    },
    "S3BucketName": {"Ref": "ConfigBucket"},
    "SnsTopicARN": {"Ref": "ConfigTopic"}
  }
}
```

YAML

```
DeliveryChannel:
  Type: "AWS::Config::DeliveryChannel"
  Properties:
    ConfigSnapshotDeliveryProperties:
      DeliveryFrequency: "Six_Hours"
    S3BucketName:
      Ref: ConfigBucket
    SnsTopicARN:
      Ref: ConfigTopic
```

AWS::DataPipeline::Pipeline

Creates a data pipeline that you can use to automate the movement and transformation of data. In each pipeline, you define pipeline objects, such as activities, schedules, data nodes, and resources. For information about pipeline objects and components that you can use, see Pipeline Object Reference in the *AWS Data Pipeline Developer Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::DataPipeline::Pipeline",
  "Properties" : {
    "Activate" : Boolean,
    "Description" : String,
```

```
    "Name" : String,
    "ParameterObjects" : [ Parameter object, ... ],
    "ParameterValues" : [ Parameter value, ... ],
    "PipelineObjects" : [ Pipeline object, ... ],
    "PipelineTags" : [ Pipeline tag, ... ]
  }
}
```

YAML

```
Type: "AWS::DataPipeline::Pipeline"
Properties:
  Activate: Boolean
  Description: String
  Name: String
  ParameterObjects:
    - Parameter object
  ParameterValues:
    - Parameter value
  PipelineObjects:
    - Pipeline object
  PipelineTags:
    - Pipeline tag
```

Properties

`Activate`

Indicates whether to validate and start the pipeline or stop an active pipeline. By default, the value is set to `true`.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`Description`

A description for the pipeline.

Required: No

Type: String

Update requires: Replacement (p. 90).

`Name`

A name for the pipeline. Because AWS CloudFormation assigns each new pipeline a unique identifier, you can use the same name for multiple pipelines that are associated with your AWS account.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`ParameterObjects`

Defines the variables that are in the pipeline definition. For more information, see Creating a Pipeline Using Parameterized Templates in the *AWS Data Pipeline Developer Guide*.

Required: No

Type: AWS Data Pipeline Pipeline ParameterObjects (p. 1089)

Update requires: No interruption (p. 90)

`ParameterValues`

Defines the values for the parameters that are defined in the `ParameterObjects` property. For more information, see Creating a Pipeline Using Parameterized Templates in the *AWS Data Pipeline Developer Guide*.

Required: No

Type: AWS Data Pipeline Pipeline ParameterValues (p. 1091)

Update requires: No interruption (p. 90)

`PipelineObjects`

A list of pipeline objects that make up the pipeline. For more information about pipeline objects and a description of each object, see Pipeline Object Reference in the *AWS Data Pipeline Developer Guide*.

Required: Yes

Type: A list of AWS Data Pipeline PipelineObjects (p. 1091)

Update requires: Some interruptions (p. 90). Not all objects, fields, and values can be updated. Restrictions on what can be updated are documented in Editing Your Pipelines in the *AWS Data Pipeline Developer Guide*.

`PipelineTags`

A list of arbitrary tags (key-value pairs) to associate with the pipeline, which you can use to control permissions. For more information, see Controlling Access to Pipelines and Resources in the *AWS Data Pipeline Developer Guide*.

Required: No

Type: AWS Data Pipeline Pipeline PipelineTags (p. 1093)

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

When you specify an `AWS::DataPipeline::Pipeline` resource as an argument to the `Ref` function, AWS CloudFormation returns the pipeline ID.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following data pipeline backs up data from an Amazon DynamoDB (DynamoDB) table to an Amazon Simple Storage Service (Amazon S3) bucket. The pipeline uses the `HiveCopyActivity` activity to copy the data, and runs it once a day. The roles for the pipeline and the pipeline resource are declared elsewhere in the same template.

JSON

```
"DynamoDBInputS3OutputHive": {
  "Type": "AWS::DataPipeline::Pipeline",
  "Properties": {
    "Name": "DynamoDBInputS3OutputHive",
    "Description": "Pipeline to backup DynamoDB data to S3",
    "Activate": "true",
    "ParameterObjects": [
      {
        "Id": "myDDBReadThroughputRatio",
        "Attributes": [
          {
            "Key": "description",
            "StringValue": "DynamoDB read throughput ratio"
          },
          {
            "Key": "type",
            "StringValue": "Double"
          },
          {
            "Key": "default",
            "StringValue": "0.2"
          }
        ]
      },
      {
        "Id": "myOutputS3Loc",
        "Attributes": [
          {
            "Key": "description",
            "StringValue": "S3 output bucket"
          },
          {
            "Key": "type",
            "StringValue": "AWS::S3::ObjectKey"
          },
          {
            "Key": "default",
            "StringValue": { "Fn::Join" : [ "", [ "s3://", { "Ref": "S3OutputLoc" } ] ] }
          }
        ]
      },
      {
        "Id": "myDDBTableName",
        "Attributes": [
          {
            "Key": "description",
            "StringValue": "DynamoDB Table Name "
          },
          {
            "Key": "type",
            "StringValue": "String"
          }
        ]
      }
    ],
    "ParameterValues": [
      {
        "Id": "myDDBTableName",
        "StringValue": { "Ref": "TableName" }
      }
    ],
    "PipelineObjects": [
      {
```

```
        "Id": "S3BackupLocation",
        "Name": "Copy data to this S3 location",
        "Fields": [
          {
            "Key": "type",
            "StringValue": "S3DataNode"
          },
          {
            "Key": "dataFormat",
            "RefValue": "DDBExportFormat"
          },
          {
            "Key": "directoryPath",
            "StringValue": "#{myOutputS3Loc}/#{format(@scheduledStartTime, 'YYYY-MM-dd-HH-
mm-ss')}"
          }
        ]
      },
      {
        "Id": "DDBSourceTable",
        "Name": "DDBSourceTable",
        "Fields": [
          {
            "Key": "tableName",
            "StringValue": "#{myDDBTableName}"
          },
          {
            "Key": "type",
            "StringValue": "DynamoDBDataNode"
          },
          {
            "Key": "dataFormat",
            "RefValue": "DDBExportFormat"
          },
          {
            "Key": "readThroughputPercent",
            "StringValue": "#{myDDBReadThroughputRatio}"
          }
        ]
      },
      {
        "Id": "DDBExportFormat",
        "Name": "DDBExportFormat",
        "Fields": [
          {
            "Key": "type",
            "StringValue": "DynamoDBExportDataFormat"
          }
        ]
      },
      {
        "Id": "TableBackupActivity",
        "Name": "TableBackupActivity",
        "Fields": [
          {
            "Key": "resizeClusterBeforeRunning",
            "StringValue": "true"
          },
          {
            "Key": "type",
            "StringValue": "HiveCopyActivity"
          },
          {
            "Key": "input",
            "RefValue": "DDBSourceTable"
          },
```

```
          {
            "Key": "runsOn",
            "RefValue": "EmrClusterForBackup"
          },
          {
            "Key": "output",
            "RefValue": "S3BackupLocation"
          }
        ]
      },
      {
        "Id": "DefaultSchedule",
        "Name": "RunOnce",
        "Fields": [
          {
            "Key": "occurrences",
            "StringValue": "1"
          },
          {
            "Key": "startAt",
            "StringValue": "FIRST_ACTIVATION_DATE_TIME"
          },
          {
            "Key": "type",
            "StringValue": "Schedule"
          },
          {
            "Key": "period",
            "StringValue": "1 Day"
          }
        ]
      },
      {
        "Id": "Default",
        "Name": "Default",
        "Fields": [
          {
            "Key": "type",
            "StringValue": "Default"
          },
          {
            "Key": "scheduleType",
            "StringValue": "cron"
          },
          {
            "Key": "failureAndRerunMode",
            "StringValue": "CASCADE"
          },
          {
            "Key": "role",
            "StringValue": "DataPipelineDefaultRole"
          },
          {
            "Key": "resourceRole",
            "StringValue": "DataPipelineDefaultResourceRole"
          },
          {
            "Key": "schedule",
            "RefValue": "DefaultSchedule"
          }
        ]
      },
      {
        "Id": "EmrClusterForBackup",
        "Name": "EmrClusterForBackup",
        "Fields": [
```

```
          {
            "Key": "terminateAfter",
            "StringValue": "2 Hours"
          },
          {
            "Key": "amiVersion",
            "StringValue": "3.3.2"
          },
          {
            "Key": "masterInstanceType",
            "StringValue": "m1.medium"
          },
          {
            "Key": "coreInstanceType",
            "StringValue": "m1.medium"
          },
          {
            "Key": "coreInstanceCount",
            "StringValue": "1"
          },
          {
            "Key": "type",
            "StringValue": "EmrCluster"
          }
        ]
      }
    ]
  }
}
```

YAML

```
DynamoDBInputS3OutputHive:
  Type: "AWS::DataPipeline::Pipeline"
  Properties:
    Name: DynamoDBInputS3OutputHive
    Description: "Pipeline to backup DynamoDB data to S3"
    Activate: true
    ParameterObjects:
      -
        Id: "myDDBReadThroughputRatio"
        Attributes:
          -
            Key: "description"
            StringValue: "DynamoDB read throughput ratio"
          -
            Key: "type"
            StringValue: "Double"
          -
            Key: "default"
            StringValue: "0.2"
      -
        Id: "myOutputS3Loc"
        Attributes:
          -
            Key: "description"
            StringValue: "S3 output bucket"
          -
            Key: "type"
            StringValue: "AWS::S3::ObjectKey"
          -
            Key: "default"
            StringValue:
              Fn::Join:
```

```
                - ""
                -
                  - "s3://"
                  -
                    Ref: "S3OutputLoc"
        -
          Id: "myDDBTableName"
          Attributes:
            -
              Key: "description"
              StringValue: "DynamoDB Table Name "
            -
              Key: "type"
              StringValue: "String"
      ParameterValues:
        -
          Id: "myDDBTableName"
          StringValue:
            Ref: "TableName"
      PipelineObjects:
        -
          Id: "S3BackupLocation"
          Name: "Copy data to this S3 location"
          Fields:
            -
              Key: "type"
              StringValue: "S3DataNode"
            -
              Key: "dataFormat"
              RefValue: "DDBExportFormat"
            -
              Key: "directoryPath"
              StringValue: "#{myOutputS3Loc}/#{format(@scheduledStartTime, 'YYYY-MM-dd-HH-mm-
ss')}"
        -
          Id: "DDBSourceTable"
          Name: "DDBSourceTable"
          Fields:
            -
              Key: "tableName"
              StringValue: "#{myDDBTableName}"
            -
              Key: "type"
              StringValue: "DynamoDBDataNode"
            -
              Key: "dataFormat"
              RefValue: "DDBExportFormat"
            -
              Key: "readThroughputPercent"
              StringValue: "#{myDDBReadThroughputRatio}"
        -
          Id: "DDBExportFormat"
          Name: "DDBExportFormat"
          Fields:
            -
              Key: "type"
              StringValue: "DynamoDBExportDataFormat"
        -
          Id: "TableBackupActivity"
          Name: "TableBackupActivity"
          Fields:
            -
              Key: "resizeClusterBeforeRunning"
              StringValue: "true"
            -
              Key: "type"
```

```
          StringValue: "HiveCopyActivity"
        -
          Key: "input"
          RefValue: "DDBSourceTable"
        -
          Key: "runsOn"
          RefValue: "EmrClusterForBackup"
        -
          Key: "output"
          RefValue: "S3BackupLocation"
    -
      Id: "DefaultSchedule"
      Name: "RunOnce"
      Fields:
        -
          Key: "occurrences"
          StringValue: "1"
        -
          Key: "startAt"
          StringValue: "FIRST_ACTIVATION_DATE_TIME"
        -
          Key: "type"
          StringValue: "Schedule"
        -
          Key: "period"
          StringValue: "1 Day"
    -
      Id: "Default"
      Name: "Default"
      Fields:
        -
          Key: "type"
          StringValue: "Default"
        -
          Key: "scheduleType"
          StringValue: "cron"
        -
          Key: "failureAndRerunMode"
          StringValue: "CASCADE"
        -
          Key: "role"
          StringValue: "DataPipelineDefaultRole"
        -
          Key: "resourceRole"
          StringValue: "DataPipelineDefaultResourceRole"
        -
          Key: "schedule"
          RefValue: "DefaultSchedule"
    -
      Id: "EmrClusterForBackup"
      Name: "EmrClusterForBackup"
      Fields:
        -
          Key: "terminateAfter"
          StringValue: "2 Hours"
        -
          Key: "amiVersion"
          StringValue: "3.3.2"
        -
          Key: "masterInstanceType"
          StringValue: "m1.medium"
        -
          Key: "coreInstanceType"
          StringValue: "m1.medium"
        -
          Key: "coreInstanceCount"
```

```
          StringValue: "1"
        -
          Key: "type"
          StringValue: "EmrCluster"
```

AWS::DirectoryService::MicrosoftAD

The `AWS::DirectoryService::MicrosoftAD` resource creates a Microsoft Active Directory in AWS so that your directory users and groups can access the AWS Management Console and AWS applications using their existing credentials. For more information, see What Is AWS Directory Service? in the *AWS Directory Service Administration Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::DirectoryService::MicrosoftAD",
  "Properties" : {
    "CreateAlias" : Boolean,
    "EnableSso" : Boolean,
    "Name" : String,
    "Password" : String,
    "ShortName" : String,
    "VpcSettings" : VpcSettings
  }
}
```

YAML

```
Type: "AWS::DirectoryService::MicrosoftAD"
Properties:
  CreateAlias: Boolean
  EnableSso: Boolean
  Name: String
  Password: String
  ShortName: String
  VpcSettings:
    VpcSettings
```

Properties

`CreateAlias`

A unique alias to assign to the Microsoft Active Directory in AWS. AWS Directory Service uses the alias to construct the access URL for the directory, such as `http://alias.awsapps.com`. By default, AWS CloudFormation does not create an alias.

Required: No

Type: Boolean

Update requires: Replacement (p. 90)

`EnableSso`

Whether to enable single sign-on for a Microsoft Active Directory in AWS. Single sign-on allows users in your directory to access certain AWS services from a computer joined to the directory without having to enter their credentials separately. If you don't specify a value, AWS CloudFormation disables single sign-on by default.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`Name`

The fully qualified name for the Microsoft Active Directory in AWS, such as `corp.example.com`. The name doesn't need to be publicly resolvable; it will resolve inside your VPC only.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Password`

The password for the default administrative user, `Admin`.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`ShortName`

The NetBIOS name for your domain, such as `CORP`. If you don't specify a value, AWS Directory Service uses the first part of your directory DNS server name. For example, if your directory DNS server name is `corp.example.com`, AWS Directory Service specifies `CORP` for the NetBIOS name.

Required: No

Type: String

Update requires: Replacement (p. 90)

`VpcSettings`

Specifies the VPC settings of the Microsoft Active Directory server in AWS.

Required: Yes

Type: AWS Directory Service MicrosoftAD VpcSettings (p. 1094)

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource ID.

In the following sample, the `Ref` function returns the ID of the `myDirectory` directory, such as `d-12345ab592`.

```
{ "Ref": "myDirectory" }
```

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`Alias`

The alias for a directory. For example: `d-12373a053a` or `alias4-mydirectory-12345abcgmzsk` (if you have the `CreateAlias` property set to true).

`DnsIpAddresses`

The IP addresses of the DNS servers for the directory, such as `[ "192.0.2.1", "192.0.2.2" ]`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Example

The following example creates a Microsoft Active Directory in AWS, where the directory DNS name is `corp.example.com`:

JSON

```
"myDirectory" : {
  "Type" : "AWS::DirectoryService::MicrosoftAD",
  "Properties" : {
    "Name" : "corp.example.com",
    "Password" : { "Ref" : "MicrosoftADPW" },
    "ShortName" : { "Ref" : "MicrosoftADShortName" },
    "VpcSettings" : {
      "SubnetIds" : [ { "Ref" : "subnetID1" }, { "Ref" : "subnetID2" } ],
      "VpcId" : { "Ref" : "vpcID" }
    }
  }
}
```

YAML

```
myDirectory:
  Type: "AWS::DirectoryService::MicrosoftAD"
  Properties:
    Name: "corp.example.com"
    Password:
      Ref: MicrosoftADPW
```

```
    ShortName:
      Ref: MicrosoftADShortName
    VpcSettings:
      SubnetIds:
        - Ref: subnetID1
        - Ref: subnetID2
      VpcId:
        Ref: vpcID
```

AWS::DirectoryService::SimpleAD

The `AWS::DirectoryService::SimpleAD` resource creates an AWS Directory Service Simple Active Directory (Simple AD) in AWS so that your directory users and groups can access the AWS Management Console and AWS applications using their existing credentials. Simple AD is a Microsoft Active Directory–compatible directory. For more information, see What Is AWS Directory Service? in the *AWS Directory Service Administration Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::DirectoryService::SimpleAD",
  "Properties" : {
    "CreateAlias" : Boolean,
    "Description" : String,
    "EnableSso" : Boolean,
    "Name" : String,
    "Password" : String,
    "ShortName" : String,
    "Size" : String,
    "VpcSettings" : VpcSettings
  }
}
```

YAML

```
Type: "AWS::DirectoryService::SimpleAD"
Properties:
  CreateAlias: Boolean
  Description: String
  EnableSso: Boolean
  Name: String
  Password: String
  ShortName: String
  Size: String
  VpcSettings:
    VpcSettings
```

Properties

`CreateAlias`

A unique alias to assign to the directory. AWS Directory Service uses the alias to construct the access URL for the directory, such as `http://alias.awsapps.com`. By default, AWS CloudFormation does not create an alias.

Required: No

Type: Boolean

Update requires: Replacement (p. 90)

`Description`

A description of the directory.

Required: No

Type: String

Update requires: Replacement (p. 90)

`EnableSso`

Whether to enable single sign-on for a directory. If you don't specify a value, AWS CloudFormation disables single sign-on by default.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`Name`

The fully qualified name for the directory, such as `corp.example.com`.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Password`

The password for the directory administrator. AWS Directory Service creates a directory administrator account with the user name `Administrator` and this password.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`ShortName`

The NetBIOS name of the on-premises directory, such as `CORP`.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Size`

The size of the directory. For valid values, see CreateDirectory in the *AWS Directory Service API Reference*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`VpcSettings`

Specifies the VPC settings of the directory server.

Required: Yes

Type: AWS Directory Service SimpleAD VpcSettings (p. 1094)

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource ID.

In the following sample, the `Ref` function returns the ID of the `myDirectory` directory, such as `d-1a2b3c4d5e`.

```
{ "Ref": "myDirectory" }
```

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`Alias`

The alias for a directory. For example: `d-12373a053a` or `alias4-mydirectory-12345abcgmzsk` (if you have the `CreateAlias` property set to true).

`DnsIpAddresses`

The IP addresses of the DNS servers for the directory, such as `[ "172.31.3.154", "172.31.63.203" ]`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Example

The following example creates a Simple AD directory, where the directory DNS name is `corp.example.com`:

JSON

```
"myDirectory" : {
```

```
  "Type" : "AWS::DirectoryService::SimpleAD",
  "Properties" : {
    "Name" : "corp.example.com",
    "Password" : { "Ref" : "SimpleADPW" },
    "Size" : "Small",
    "VpcSettings" : {
      "SubnetIds" : [ { "Ref" : "subnetID1" }, { "Ref" : "subnetID2" } ],
      "VpcId" : { "Ref" : "vpcID" }
    }
  }
}
```

YAML

```
myDirectory:
  Type: "AWS::DirectoryService::SimpleAD"
  Properties:
    Name: "corp.example.com"
    Password:
      Ref: SimpleADPW
    Size: "Small"
    VpcSettings:
      SubnetIds:
        - Ref: subnetID1
        - Ref: subnetID2
      VpcId:
        Ref: vpcID
```

AWS::DynamoDB::Table

Creates a DynamoDB table.

> **Note**
> AWS CloudFormation typically creates DynamoDB tables in parallel. However, if your template includes multiple DynamoDB tables with indexes, you must declare dependencies so that the tables are created sequentially. DynamoDB limits the number of tables with secondary indexes that are in the creating state. If you create multiple tables with indexes at the same time, DynamoDB returns an error and the stack operation fails. For a sample snippet, see DynamoDB Table with a DependsOn Attribute (p. 556).

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::DynamoDB::Table",
  "Properties" : {
```

```
    "AttributeDefinitions" : [ AttributeDefinitions, ... ],
    "GlobalSecondaryIndexes" : [ GlobalSecondaryIndexes, ... ],
    "KeySchema" : [ KeySchema, ... ],
    "LocalSecondaryIndexes" : [ LocalSecondaryIndexes, ... ],
    "ProvisionedThroughput" : ProvisionedThroughput,
    "StreamSpecification" : StreamSpecification,
    "TableName" : String
  }
}
```

YAML

```
Type: "AWS::DynamoDB::Table"
Properties:
  AttributeDefinitions:
    - AttributeDefinitions
  GlobalSecondaryIndexes:
    - GlobalSecondaryIndexes
  KeySchema:
    - KeySchema
  LocalSecondaryIndexes:
    - LocalSecondaryIndexes
  ProvisionedThroughput:
    ProvisionedThroughput
  StreamSpecification:
    StreamSpecification
  TableName: String
```

Properties

`AttributeDefinitions`

A list of `AttributeName` and `AttributeType` objects that describe the key schema for the table and indexes.

Required: Yes

Type: List of DynamoDB Attribute Definitions (p. 1095)

Update requires: No interruption (p. 90)

`GlobalSecondaryIndexes`

Global secondary indexes to be created on the table. You can create up to 5 global secondary indexes.

Important
If you update a table to include a new global secondary index, AWS CloudFormation initiates the index creation and then proceeds with the stack update. AWS CloudFormation doesn't wait for the index to complete creation because the backfilling phase can take a long time, depending on the size of the table. You cannot use the index or update the table until the index's status is `ACTIVE`. You can track its status by using the DynamoDB `DescribeTable` command.
If you add or delete an index during an update, we recommend that you don't update any other resources. If your stack fails to update and is rolled back while adding a new index, you must manually delete the index.

Required: No

Type: List of DynamoDB Global Secondary Indexes (p. 1096)

Update requires: Updates are not supported. with the following exceptions:

- If you update only the provisioned throughput values of global secondary indexes, you can update the table without interruption (p. 90).
- You can delete or add one global secondary index without interruption (p. 90). If you do both in the same update (for example, by changing the index's logical ID), the update fails.

`KeySchema`

Specifies the attributes that make up the primary key for the table. The attributes in the `KeySchema` property must also be defined in the `AttributeDefinitions` property.

Required: Yes

Type: List of DynamoDB Key Schema (p. 1097)

Update requires: Replacement (p. 90)

`LocalSecondaryIndexes`

Local secondary indexes to be created on the table. You can create up to 5 local secondary indexes. Each index is scoped to a given hash key value. The size of each hash key can be up to 10 gigabytes.

Required: No

Type: List of DynamoDB Local Secondary Indexes (p. 1098)

Update requires: Replacement (p. 90)

`ProvisionedThroughput`

Throughput for the specified table, consisting of values for ReadCapacityUnits and WriteCapacityUnits. For more information about the contents of a provisioned throughput structure, see DynamoDB Provisioned Throughput (p. 1100).

Required: Yes

Type: DynamoDB Provisioned Throughput (p. 1100)

Update requires: No interruption (p. 90)

`StreamSpecification`

The settings for the DynamoDB table stream, which capture changes to items stored in the table.

Required: No

Type: DynamoDB Table StreamSpecification (p. 1100)

Update requires: No interruption (p. 90) to the table; however, the stream is replaced.

`TableName`

A name for the table. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the table name. For more information, see Name Type (p. 1217).

> **Important**
> If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

Note
For detailed information about the limits in DynamoDB, see Limits in Amazon DynamoDB in the *Amazon DynamoDB Developer Guide*.

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "MyResource" }
```

For the resource with the logical ID `myDynamoDBTable`, `Ref` will return the DynamoDB table name.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`StreamArn`

The Amazon Resource Name (ARN) of the DynamoDB stream, such as `arn:aws:dynamodb:us-east-1:123456789012:table/testddbstack-myDynamoDBTable-012A1SL7SMP5Q/stream/2015-11-30T20:10:00.000`.

Note
You must specify the `StreamSpecification` property to use this attribute.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

DynamoDB Table with Local and Secondary Indexes

The following sample creates an DynamoDB table with `Album`, `Artist`, `Sales`, `NumberOfSongs` as attributes. The primary key includes the `Album` attribute as the hash key and `Artist` attribute as the range key. The table also includes two global and one secondary index. For querying the number of sales for a given artist, the global secondary index uses the `Sales` attribute as the hash key and the `Artist` attribute as the range key.

For querying the sales based on the number of songs, the global secondary index uses the `NumberOfSongs` attribute as the hash key and the `Sales` attribute as the range key.

For querying the sales of an album, the local secondary index uses the same hash key as the table but uses the `Sales` attribute as the range key.

JSON

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Resources" : {
```

```
"myDynamoDBTable" : {
  "Type" : "AWS::DynamoDB::Table",
  "Properties" : {
    "AttributeDefinitions" : [
      {
        "AttributeName" : "Album",
        "AttributeType" : "S"
      },
      {
        "AttributeName" : "Artist",
        "AttributeType" : "S"
      },
      {
        "AttributeName" : "Sales",
        "AttributeType" : "N"
      },
      {
        "AttributeName" : "NumberOfSongs",
        "AttributeType" : "N"
      }
    ],
    "KeySchema" : [
      {
        "AttributeName" : "Album",
        "KeyType" : "HASH"
      },
      {
        "AttributeName" : "Artist",
        "KeyType" : "RANGE"
      }
    ],
    "ProvisionedThroughput" : {
      "ReadCapacityUnits" : "5",
      "WriteCapacityUnits" : "5"
    },
    "TableName" : "myTableName",
    "GlobalSecondaryIndexes" : [{
      "IndexName" : "myGSI",
      "KeySchema" : [
        {
          "AttributeName" : "Sales",
          "KeyType" : "HASH"
        },
        {
          "AttributeName" : "Artist",
          "KeyType" : "RANGE"
        }
      ],
      "Projection" : {
        "NonKeyAttributes" : ["Album","NumberOfSongs"],
        "ProjectionType" : "INCLUDE"
      },
      "ProvisionedThroughput" : {
        "ReadCapacityUnits" : "5",
        "WriteCapacityUnits" : "5"
      }
    },
    {
      "IndexName" : "myGSI2",
      "KeySchema" : [
        {
          "AttributeName" : "NumberOfSongs",
          "KeyType" : "HASH"
        },
        {
          "AttributeName" : "Sales",
```

```
            "KeyType" : "RANGE"
          }
        ],
        "Projection" : {
          "NonKeyAttributes" : ["Album","Artist"],
          "ProjectionType" : "INCLUDE"
        },
        "ProvisionedThroughput" : {
          "ReadCapacityUnits" : "5",
          "WriteCapacityUnits" : "5"
        }
      }],
      "LocalSecondaryIndexes" :[{
        "IndexName" : "myLSI",
        "KeySchema" : [
          {
            "AttributeName" : "Album",
            "KeyType" : "HASH"
          },
          {
            "AttributeName" : "Sales",
            "KeyType" : "RANGE"
          }
        ],
        "Projection" : {
          "NonKeyAttributes" : ["Artist","NumberOfSongs"],
          "ProjectionType" : "INCLUDE"
        }
      }]
    }
  }
 }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  myDynamoDBTable:
    Type: "AWS::DynamoDB::Table"
    Properties:
      AttributeDefinitions:
        -
          AttributeName: "Album"
          AttributeType: "S"
        -
          AttributeName: "Artist"
          AttributeType: "S"
        -
          AttributeName: "Sales"
          AttributeType: "N"
        -
          AttributeName: "NumberOfSongs"
          AttributeType: "N"
      KeySchema:
        -
          AttributeName: "Album"
          KeyType: "HASH"
        -
          AttributeName: "Artist"
          KeyType: "RANGE"
      ProvisionedThroughput:
        ReadCapacityUnits: "5"
        WriteCapacityUnits: "5"
```

```
      TableName: "myTableName"
      GlobalSecondaryIndexes:
        -
          IndexName: "myGSI"
          KeySchema:
            -
              AttributeName: "Sales"
              KeyType: "HASH"
            -
              AttributeName: "Artist"
              KeyType: "RANGE"
          Projection:
            NonKeyAttributes:
              - "Album"
              - "NumberOfSongs"
            ProjectionType: "INCLUDE"
          ProvisionedThroughput:
            ReadCapacityUnits: "5"
            WriteCapacityUnits: "5"
        -
          IndexName: "myGSI2"
          KeySchema:
            -
              AttributeName: "NumberOfSongs"
              KeyType: "HASH"
            -
              AttributeName: "Sales"
              KeyType: "RANGE"
          Projection:
            NonKeyAttributes:
              - "Album"
              - "Artist"
            ProjectionType: "INCLUDE"
          ProvisionedThroughput:
            ReadCapacityUnits: "5"
            WriteCapacityUnits: "5"
      LocalSecondaryIndexes:
        -
          IndexName: "myLSI"
          KeySchema:
            -
              AttributeName: "Album"
              KeyType: "HASH"
            -
              AttributeName: "Sales"
              KeyType: "RANGE"
          Projection:
            NonKeyAttributes:
              - "Artist"
              - "NumberOfSongs"
            ProjectionType: "INCLUDE"
```

DynamoDB Table with a DependsOn Attribute

If you include multiple DynamoDB tables with indexes in a single template, you must include dependencies so that the tables are created sequentially. DynamoDB limits the number of tables with secondary indexes that are in the creating state. If you create multiple tables with indexes at the same time, DynamoDB returns an error and the stack operation fails.

The following sample assumes that the `myFirstDDBTable` table is declared in the same template as the `mySecondDDBTable` table, and both tables include a secondary index. The `mySecondDDBTable` table includes a dependency on the `myFirstDDBTable` table so that AWS CloudFormation creates the tables one at a time.

JSON

```
"mySecondDDBTable" : {
  "Type" : "AWS::DynamoDB::Table",
  "DependsOn" : "myFirstDDBTable" ,
  "Properties" : {
    "AttributeDefinitions" : [
      {
        "AttributeName" : "ArtistId",
        "AttributeType" : "S"
      },
      {
        "AttributeName" : "Concert",
        "AttributeType" : "S"
      },
      {
        "AttributeName" : "TicketSales",
        "AttributeType" : "S"
      }
    ],
    "KeySchema" : [
      {
        "AttributeName" : "ArtistId",
        "KeyType" : "HASH"
      },
      {
        "AttributeName" : "Concert",
        "KeyType" : "RANGE"
      }
    ],
    "ProvisionedThroughput" : {
      "ReadCapacityUnits" : {"Ref" : "ReadCapacityUnits"},
      "WriteCapacityUnits" : {"Ref" : "WriteCapacityUnits"}
    },
    "GlobalSecondaryIndexes" : [{
      "IndexName" : "myGSI",
      "KeySchema" : [
        {
          "AttributeName" : "TicketSales",
          "KeyType" : "HASH"
        }
      ],
      "Projection" : {
        "ProjectionType" : "KEYS_ONLY"
      },
      "ProvisionedThroughput" : {
        "ReadCapacityUnits" : {"Ref" : "ReadCapacityUnits"},
        "WriteCapacityUnits" : {"Ref" : "WriteCapacityUnits"}
      }
    }]
  }
}
```

YAML

```
mySecondDDBTable:
  Type: "AWS::DynamoDB::Table"
  DependsOn: "myFirstDDBTable"
  Properties:
    AttributeDefinitions:
      -
        AttributeName: "ArtistId"
        AttributeType: "S"
```

```
      -
        AttributeName: "Concert"
        AttributeType: "S"
      -
        AttributeName: "TicketSales"
        AttributeType: "S"
    KeySchema:
      -
        AttributeName: "ArtistId"
        KeyType: "HASH"
      -
        AttributeName: "Concert"
        KeyType: "RANGE"
    ProvisionedThroughput:
      ReadCapacityUnits:
        Ref: "ReadCapacityUnits"
      WriteCapacityUnits:
        Ref: "WriteCapacityUnits"
    GlobalSecondaryIndexes:
      -
        IndexName: "myGSI"
        KeySchema:
          -
            AttributeName: "TicketSales"
            KeyType: "HASH"
        Projection:
          ProjectionType: "KEYS_ONLY"
        ProvisionedThroughput:
          ReadCapacityUnits:
            Ref: "ReadCapacityUnits"
          WriteCapacityUnits:
            Ref: "WriteCapacityUnits"
```

AWS::EC2::CustomerGateway

Provides information to AWS about your VPN customer gateway device.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::CustomerGateway",
   "Properties" : {
      "BgpAsn (p. 559)" : Number,
      "IpAddress (p. 559)" : String,
      "Tags" :  [ Resource Tag, ... ],
      "Type (p. 559)" : String
   }
```

```
}
```

YAML

```
Type: "AWS::EC2::CustomerGateway"
Properties:
  BgpAsn (p. 559): Number
  IpAddress (p. 559): String
  Tags:
    Resource Tag
  Type (p. 559): String
```

Properties

`BgpAsn`

The customer gateway's Border Gateway Protocol (BGP) Autonomous System Number (ASN).

Required: Yes

Type: Number BgpAsn is always an integer value.

Update requires: Replacement (p. 90)

`IpAddress`

The internet-routable IP address for the customer gateway's outside interface. The address must be static.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Tags`

The tags that you want to attach to the resource.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236).

Update requires: No interruption (p. 90).

`Type`

The type of VPN connection that this customer gateway supports.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Example: `ipsec.1`

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "MyResource" }
```

For the resource with the logical ID "MyResource", `Ref` will return the AWS resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

JSON

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "myCustomerGateway" : {
         "Type" : "AWS::EC2::CustomerGateway",
         "Properties" : {
            "Type" : "ipsec.1",
            "BgpAsn" : "64000",
            "IpAddress" : "1.1.1.1"
         }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  myCustomerGateway:
    Type: "AWS::EC2::CustomerGateway"
    Properties:
      Type: ipsec.1
      BgpAsn: 64000
      IpAddress: 1.1.1.1
```

See Also

- CreateCustomerGateway in the *Amazon EC2 API Reference*.

AWS::EC2::DHCPOptions

Creates a set of DHCP options for your VPC.

For more information, see CreateDhcpOptions in the *Amazon EC2 API Reference*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::DHCPOptions",
   "Properties" : {
      "DomainName (p. 561)" : String,
      "DomainNameServers (p. 561)" : [ String, ... ],
      "NetbiosNameServers (p. 562)" : [ String, ... ],
      "NetbiosNodeType (p. 562)" : Number,
      "NtpServers (p. 562)" : [ String, ... ],
      "Tags (p. 562)" : [ Resource Tag, ... ]
   }
}
```

YAML

```
Type: "AWS::EC2::DHCPOptions"
Properties:
  DomainName (p. 561): String
  DomainNameServers (p. 561):
    - String
  NetbiosNameServers (p. 562):
    - String
  NetbiosNodeType (p. 562): Number
  NtpServers (p. 562):
    - String
  Tags (p. 562):
    -Resource Tag
```

Properties

`DomainName`

A domain name of your choice.

Required: Conditional; see note (p. 563).

Type: String

Update requires: Replacement (p. 90)

Example: `"example.com"`

`DomainNameServers`

The IP (IPv4) address of a domain name server. You can specify up to four addresses.

Required: Conditional; see note (p. 563).

Type: List of strings

Update requires: Replacement (p. 90)

Example: `"DomainNameServers" : [ "10.0.0.1", "10.0.0.2" ]`

Example: To preserve the order of IP addresses, specify a comma delimited list as a single string: `"DomainNameServers" : [ "10.0.0.1, 10.0.0.2" ]`

`NetbiosNameServers`

The IP address (IPv4) of a NetBIOS name server. You can specify up to four addresses.

Required: Conditional; see note (p. 563).

Type: List of strings

Update requires: Replacement (p. 90)

Example: `"NetbiosNameServers" : [ "10.0.0.1", "10.0.0.2" ]`

Example: To preserve the order of IP addresses, specify a comma delimited list as a single string: `"NetbiosNameServers" : [ "10.0.0.1, 10.0.0.2" ]`

`NetbiosNodeType`

An integer value indicating the NetBIOS node type:

- **1**: Broadcast ("B")
- **2**: Point-to-point ("P")
- **4**: Mixed mode ("M")
- **8**: Hybrid ("H")

For more information about these values and about NetBIOS node types, see RFC 2132, RFC 1001, and RFC 1002. We recommend that you use only the value `2` at this time (broadcast and multicast are not currently supported).

Required: Required if `NetBiosNameServers` is specified; optional otherwise.

Type: List of numbers

Update requires: Replacement (p. 90)

Example: `"NetbiosNodeType" : 2`

`NtpServers`

The IP address (IPv4) of a Network Time Protocol (NTP) server. You can specify up to four addresses.

Required: Conditional; see note (p. 563).

Type: List of strings

Update requires: Replacement (p. 90)

Example: `"NtpServers" : [ "10.0.0.1" ]`

Example: To preserve the order of IP addresses, specify a comma delimited list as a single string: `"NtpServers" : [ "10.0.0.1, 10.0.0.2" ]`

`Tags`

An arbitrary set of tags (key–value pairs) for this resource.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90).

Conditional Properties

At least one of the following properties must be specified:

- DomainNameServers (p. 561)
- NetbiosNameServers (p. 562)
- NtpServers (p. 562)

After this condition has been fulfilled, the rest of these properties are optional.

If you specify `NetbiosNameServers`, then `NetbiosNodeType` is required.

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

JSON

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "myDhcpOptions" : {
         "Type" : "AWS::EC2::DHCPOptions",
         "Properties" : {
            "DomainName" : "example.com",
            "DomainNameServers" : [ "AmazonProvidedDNS" ],
            "NtpServers" : [ "10.2.5.1" ],
            "NetbiosNameServers" : [ "10.2.5.1" ],
            "NetbiosNodeType" : 2,
            "Tags" : [ { "Key" : "foo", "Value" : "bar" } ]
         }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  myDhcpOptions:
    Type: "AWS::EC2::DHCPOptions"
    Properties:
      DomainName: example.com
      DomainNameServers:
        - AmazonProvidedDNS
      NtpServers:
```

```
    - 10.2.5.1
  NetbiosNameServers:
    - 10.2.5.1
  NetbiosNodeType: 2
  Tags:
    -
      Key: foo
      Value: bar
```

See Also

- CreateDhcpOptions in the *Amazon EC2 API Reference*
- Using Tags in the *Amazon Elastic Compute Cloud User Guide*.
- RFC 2132 - *DHCP Options and BOOTP Vendor Extensions*, Network Working Group, 1997
- RFC 1001 - *Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Concepts and Methods*, Network Working Group, 1987
- RFC 1002 - *Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Detailed Specifications*, Network Working Group, 1987

AWS::EC2::EIP

The AWS::EC2::EIP resource allocates an Elastic IP (EIP) address and can, optionally, associate it with an Amazon EC2 instance.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::EIP",
   "Properties" : {
      "InstanceId (p. 565)" : String,
      "Domain (p. 565)" : String
   }
}
```

YAML

```
Type: "AWS::EC2::EIP"
Properties:
  InstanceId (p. 565): String
  Domain (p. 565): String
```

Properties

`InstanceId`

The Instance ID of the Amazon EC2 instance that you want to associate with this Elastic IP address.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Domain`

Set to `vpc` to allocate the address to your Virtual Private Cloud (VPC). No other values are supported.

> **Note**
> If you define an Elastic IP address and associate it with a VPC that is defined in the same template, you must declare a dependency on the VPC-gateway attachment by using the `DependsOn` attribute on this resource. For more information, see DependsOn Attribute (p. 1298).

For more information, see AllocateAddress in the *Amazon EC2 API Reference*. For more information about Elastic IP Addresses in VPC, go to IP Addressing in Your VPC in the *Amazon VPC User Guide*.

Required: Conditional. Required when allocating an address to a VPC

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When you specify the logical ID of an AWS::EC2::EIP object as an argument to the `Ref` function, AWS CloudFormation returns the value of the instance's `PublicIp`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`AllocationId`

The ID that AWS assigns to represent the allocation of the address for use with Amazon VPC. This is returned only for VPC elastic IP addresses. Example return value: `eipalloc-5723d13e`

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Examples

To view AWS::EC2::EIP snippets, see Assigning an Amazon EC2 Elastic IP Using AWS::EC2::EIP Snippet (p. 270).

AWS::EC2::EIPAssociation

The AWS::EC2::EIPAssociation resource type associates an Elastic IP address with an Amazon EC2 instance. The Elastic IP address can be an existing Elastic IP address or an Elastic IP address allocated through an AWS::EC2::EIP resource (p. 564).

For more information EC2-Classic and EC2-VPC, see AssociateAddress in the *Amazon EC2 API Reference*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type": "AWS::EC2::EIPAssociation",
   "Properties": {
      "AllocationId (p. 566)": String,
      "EIP (p. 567)": String,
      "InstanceId (p. 567)": String,
      "NetworkInterfaceId (p. 567)": String,
      "PrivateIpAddress (p. 567)": String
   }
}
```

YAML

```
Type: "AWS::EC2::EIPAssociation"
Properties:
  AllocationId (p. 566): String
  EIP (p. 567): String
  InstanceId (p. 567): String
  NetworkInterfaceId (p. 567): String
  PrivateIpAddress (p. 567): String
```

Properties

`AllocationId`

[EC2-VPC] Allocation ID for the VPC Elastic IP address you want to associate with an Amazon EC2 instance in your VPC.

Required: Conditional. Required for EC2-VPC.

Type: String

Update requires: Replacement (p. 90) if you also change the `InstanceId` or `NetworkInterfaceId` property. If not, update requires No interruption (p. 90).

`EIP`

Elastic IP address that you want to associate with the Amazon EC2 instance specified by the `InstanceId` property. You can specify an existing Elastic IP address or a reference to an Elastic IP address allocated with a AWS::EC2::EIP resource (p. 564).

Required: Conditional. Required for EC2-Classic.

Type: String

Update requires: Replacement (p. 90) if you also change the `InstanceId` or `NetworkInterfaceId` property. If not, update requires No interruption (p. 90).

`InstanceId`

Instance ID of the Amazon EC2 instance that you want to associate with the Elastic IP address specified by the EIP property. If the instance has more than one network interface, you must specify a network interface ID.

Required: Conditional. If you specify the `EIP` property, you must specify this property. If you specify the `AllocationId` property, you must specify this property or the `NetworkInterfaceId` property.

Type: String

Update requires: Replacement (p. 90) if you also change the `AllocationId` or `EIP` property. If not, update requires No interruption (p. 90).

`NetworkInterfaceId`

[EC2-VPC] The ID of the network interface to associate with the Elastic IP address. If the instance has more than one network interface, you must specify a network interface ID.

Required: Conditional. If you specify the `AllocationId` property, you must specify this property or the `InstanceId` property.

Type: String

Update requires: Replacement (p. 90) if you also change the `AllocationId` or `EIP` property. If not, update requires No interruption (p. 90).

`PrivateIpAddress`

[EC2-VPC] The private IP address that you want to associate with the Elastic IP address. The private IP address is restricted to the primary and secondary private IP addresses that are associated with the network interface. By default, the private IP address that is associated with the EIP is the primary private IP address of the network interface.

Required: No

Type: String

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

The following example creates an instance with two elastic network interfaces (ENI). The example assumes that you have an existing VPC.

For additional examples, see Assigning an Amazon EC2 Elastic IP Using AWS::EC2::EIP Snippet (p. 270).

JSON

```
 "Resources" : {
   "ControlPortAddress" : {
     "Type" : "AWS::EC2::EIP",
     "Properties" : {
       "Domain" : "vpc"
     }
   },
   "AssociateControlPort" : {
     "Type" : "AWS::EC2::EIPAssociation",
     "Properties" : {
       "AllocationId" : { "Fn::GetAtt" : [ "ControlPortAddress", "AllocationId" ]},
       "NetworkInterfaceId" : { "Ref" : "controlXface" }
     }
   },
   "WebPortAddress" : {
     "Type" : "AWS::EC2::EIP",
     "Properties" : {
       "Domain" : "vpc"
     }
   },
   "AssociateWebPort" : {
     "Type" : "AWS::EC2::EIPAssociation",
     "Properties" : {
       "AllocationId" : { "Fn::GetAtt" : [ "WebPortAddress", "AllocationId" ]},
       "NetworkInterfaceId" : { "Ref" : "webXface" }
     }
   },
   "SSHSecurityGroup" : {
     "Type" : "AWS::EC2::SecurityGroup",
     "Properties" : {
       "VpcId" : { "Ref" : "VpcId" },
       "GroupDescription" : "Enable SSH access via port 22",
       "SecurityGroupIngress" : [ { "IpProtocol" : "tcp", "FromPort" : "22", "ToPort" :
"22", "CidrIp" : "0.0.0.0/0" } ]
     }
   },
   "WebSecurityGroup" : {
     "Type" : "AWS::EC2::SecurityGroup",
     "Properties" : {
       "VpcId" : { "Ref" : "VpcId" },
       "GroupDescription" : "Enable HTTP access via user defined port",
       "SecurityGroupIngress" : [ { "IpProtocol" : "tcp", "FromPort" : 80, "ToPort" : 80,
"CidrIp" : "0.0.0.0/0" } ]
     }
   },
   "controlXface" : {
     "Type" : "AWS::EC2::NetworkInterface",
     "Properties" : {
       "SubnetId" : { "Ref" : "SubnetId" },
       "Description" :"Interface for control traffic such as SSH",
       "GroupSet" : [ {"Ref" : "SSHSecurityGroup"} ],
       "SourceDestCheck" : "true",
       "Tags" : [ {"Key" : "Network", "Value" : "Control"}]
     }
   },
```

```
    "webXface" : {
       "Type" : "AWS::EC2::NetworkInterface",
       "Properties" : {
         "SubnetId" : { "Ref" : "SubnetId" },
         "Description" :"Interface for web traffic",
         "GroupSet" : [ {"Ref" : "WebSecurityGroup"} ],
         "SourceDestCheck" : "true",
         "Tags" : [ {"Key" : "Network", "Value" : "Web"}]
       }
     },
     "Ec2Instance" : {
       "Type" : "AWS::EC2::Instance",
       "Properties" : {
         "ImageId" : { "Fn::FindInMap" : [ "RegionMap", { "Ref" : "AWS::Region" }, "AMI" ]},
         "KeyName" : { "Ref" : "KeyName" },
         "NetworkInterfaces" : [ { "NetworkInterfaceId" : {"Ref" : "controlXface"},
 "DeviceIndex" : "0" },
         { "NetworkInterfaceId" : {"Ref" : "webXface"}, "DeviceIndex" : "1" }],
         "Tags" : [ {"Key" : "Role", "Value" : "Test Instance"}],
         "UserData" : {"Fn::Base64" : { "Fn::Join" : ["",[
   "#!/bin/bash -ex","\n",
             "\n","yum install ec2-net-utils -y","\n",
   "ec2ifup eth1","\n",
   "service httpd start"]]}
  }
   }
    }
  }
```

YAML

```
Resources:
  ControlPortAddress:
    Type: AWS::EC2::EIP
    Properties:
      Domain: vpc
  AssociateControlPort:
    Type: AWS::EC2::EIPAssociation
    Properties:
      AllocationId: !GetAtt ControlPortAddress.AllocationId
      NetworkInterfaceId: !Ref controlXface
  WebPortAddress:
    Type: AWS::EC2::EIP
    Properties:
      Domain: vpc
  AssociateWebPort:
    Type: AWS::EC2::EIPAssociation
    Properties:
      AllocationId: !GetAtt WebPortAddress.AllocationId
      NetworkInterfaceId: !Ref webXface
  SSHSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      VpcId: !Ref VpcId
      GroupDescription: Enable SSH access via port 22
      SecurityGroupIngress:
      - CidrIp: 0.0.0.0/0
        FromPort: 22
        IpProtocol: tcp
        ToPort: 22
  WebSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      VpcId: !Ref VpcId
```

```
        GroupDescription: Enable HTTP access via user defined port
        SecurityGroupIngress:
        - CidrIp: 0.0.0.0/0
          FromPort: 80
          IpProtocol: tcp
          ToPort: 80
    controlXface:
      Type: AWS::EC2::NetworkInterface
      Properties:
        SubnetId: !Ref SubnetId
        Description: Interface for controlling traffic such as SSH
        GroupSet:
        - !Ref SSHSecurityGroup
        SourceDestCheck: true
        Tags:
          -
            Key: Network
            Value: Control
    webXface:
      Type: AWS::EC2::NetworkInterface
      Properties:
        SubnetId: !Ref SubnetId
        Description: Interface for controlling traffic such as SSH
        GroupSet:
        - !Ref WebSecurityGroup
        SourceDestCheck: true
        Tags:
          -
            Key: Network
            Value: Web
    Ec2Instance:
      Type: AWS::EC2::Instance
      Properties:
        ImageId: !FindInMap [ RegionMap, !Ref 'AWS::Region', AMI ]
        KeyName: !Ref KeyName
        NetworkInterfaces:
          -
            NetworkInterfaceId: !Ref controlXface
            DeviceIndex: 0
          -
            NetworkInterfaceId: !Ref webXface
            DeviceIndex: 1
        Tags:
          -
            Key: Role
            Value: Test Instance
        UserData:
          Fn::Base64: !Sub |
            #!/bin/bash -xe
            yum install ec2-net-utils -y
            ec2ifup eth1
            service httpd start
```

AWS::EC2::FlowLog

The `AWS::EC2::FlowLog` resource creates an Amazon Elastic Compute Cloud (Amazon EC2) flow log that captures IP traffic for a specified network interface, subnet, or VPC. To view the log data, use Amazon CloudWatch Logs (CloudWatch Logs) to help troubleshoot connection issues. For example, you can use a flow log to investigate why certain traffic isn't reaching an instance, which can help you diagnose overly restrictive security group rules. For more information, see VPC Flow Logs in the *Amazon VPC User Guide*.

Topics

- Properties (p. 571)
- Return Value (p. 572)
- Example (p. 572)

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::EC2::FlowLog",
  "Properties" : {
    "DeliverLogsPermissionArn" : String,
    "LogGroupName" : String,
    "ResourceId" : String,
    "ResourceType" : String,
    "TrafficType" : String
  }
}
```

YAML

```
Type: "AWS::EC2::FlowLog"
Properties:
  DeliverLogsPermissionArn : String
  LogGroupName : String
  ResourceId : String
  ResourceType : String
  TrafficType : String
```

Properties

`DeliverLogsPermissionArn`

The Amazon Resource Name (ARN) of an AWS Identity and Access Management (IAM) role that permits Amazon EC2 to publish flow logs to a CloudWatch Logs log group in your account.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`LogGroupName`

The name of a new or existing CloudWatch Logs log group where Amazon EC2 publishes your flow logs.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`ResourceId`

The ID of the subnet, network interface, or VPC for which you want to create a flow log.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`ResourceType`

The type of resource that you specified in the `ResourceId` property. For example, if you specified a VPC ID for the `ResourceId` property, specify `VPC` for this property. For valid values, see the `ResourceType` parameter for the CreateFlowLogs action in the *Amazon EC2 API Reference*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`TrafficType`

The type of traffic to log. You can log traffic that the resource accepts or rejects, or all traffic. For valid values, see the `TrafficType` parameter for the CreateFlowLogs action in the *Amazon EC2 API Reference*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the flow log ID, such as `fl-1a23b456`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a flow log for the VPC called `MyVPC` and logs all traffic types. Amazon EC2 publishes the logs to the `FlowLogsGroup` log group.

```
"MyFlowLog" : {
  "Type" : "AWS::EC2::FlowLog",
  "Properties" : {
    "DeliverLogsPermissionArn" : { "Fn::GetAtt" : ["FlowLogRole", "Arn"] },
    "LogGroupName" : "FlowLogsGroup",
    "ResourceId" : { "Ref" : "MyVPC" },
    "ResourceType" : "VPC",
    "TrafficType" : "ALL"
  }
}
```

AWS::EC2::Host

The `AWS::EC2::Host` resource allocates a fully dedicated physical server for launching EC2 instances. Because the host is fully dedicated for your use, it can help you address compliance requirements and reduce costs by allowing you to use your existing server-bound software licenses. For more information, see Dedicated Hosts in the *Amazon EC2 User Guide for Linux Instances*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::EC2::Host",
  "Properties" : {
    "AutoPlacement" : String,
    "AvailabilityZone" : String,
    "InstanceType" : String
  }
}
```

YAML

```
Type: "AWS::EC2::Host"
Properties:
  AutoPlacement: String
  AvailabilityZone: String
  InstanceType: String
```

Properties

`AutoPlacement`

Indicates if the host accepts EC2 instances with only matching configurations or if instances must also specify the host ID. Instances that don't specify a host ID can't launch onto a host with `AutoPlacement` set to `off`. By default, AWS CloudFormation sets this property to `on`. For more information, see Understanding Instance Placement and Host Affinity in the *Amazon EC2 User Guide for Linux Instances*.

Required: No

Type: String

Update requires: No interruption (p. 90)

`AvailabilityZone`

The Availability Zone (AZ) in which to launch the dedicated host.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`InstanceType`

The instance type that the dedicated host accepts. Only instances of this type can be launched onto the host. For more information, see Supported Instance Types in the *Amazon EC2 User Guide for Linux Instances*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the host ID, such as `h-0ab123c45d67ef89`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example allocates a dedicated host for `c3.large` instances in the `us-east-1a` Availability Zone.

```
"Host" : {
  "Type" : "AWS::EC2::Host",
  "Properties" : {
    "AutoPlacement" : "on",
    "AvailabilityZone" : "us-east-1a",
    "InstanceType" : "c3.large"
  }
}
```

AWS::EC2::Instance

The `AWS::EC2::Instance` resource creates an EC2 instance.

If an Elastic IP address is attached to your instance, AWS CloudFormation reattaches the Elastic IP address after it updates the instance. For more information about updating stacks, see AWS CloudFormation Stacks Updates (p. 89).

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::Instance",
   "Properties" : {
      "Affinity" : String,
      "AvailabilityZone" : String,
      "BlockDeviceMappings" : [ EC2 Block Device Mapping, ... ],
      "DisableApiTermination" : Boolean,
      "EbsOptimized" : Boolean,
      "HostId" : String,
      "IamInstanceProfile" : String,
      "ImageId" : String,
      "InstanceInitiatedShutdownBehavior" : String,
      "InstanceType" : String,
      "Ipv6AddressCount" : Integer,
      "Ipv6Addresses" : [ IPv6 Address Type, ... ],
      "KernelId" : String,
      "KeyName" : String,
      "Monitoring" : Boolean,
      "NetworkInterfaces" : [ EC2 Network Interface, ... ],
      "PlacementGroupName" : String,
      "PrivateIpAddress" : String,
      "RamdiskId" : String,
      "SecurityGroupIds" : [ String, ... ],
      "SecurityGroups" : [ String, ... ],
      "SourceDestCheck" : Boolean,
      "SsmAssociations" : [ SSMAssociation, ... ],
      "SubnetId" : String,
      "Tags" : [ Resource Tag, ... ],
      "Tenancy" : String,
      "UserData" : String,
      "Volumes" : [ EC2 MountPoint (p. 1106), ... ],
      "AdditionalInfo" : String
   }
}
```

YAML

```
Type: "AWS::EC2::Instance"
Properties:
  Affinity: String
  AvailabilityZone: String
  BlockDeviceMappings:
    - EC2 Block Device Mapping
  DisableApiTermination: Boolean
  EbsOptimized: Boolean
  HostId: String
  IamInstanceProfile: String
  ImageId: String
  InstanceInitiatedShutdownBehavior: String
  InstanceType: String
  Ipv6AddressCount: Integer
  Ipv6Addresses:
    - IPv6 Address Type
  KernelId: String
  KeyName: String
  Monitoring: Boolean
  NetworkInterfaces:
    - EC2 Network Interface"
  PlacementGroupName: String
  PrivateIpAddress: String
```

```
  RamdiskId: String
  SecurityGroupIds:
    - String
  SecurityGroups:
    - String
  SourceDestCheck: Boolean
  SsmAssociations:
    - SSMAssociation
  SubnetId: String
  Tags:
    - Resource Tag
  Tenancy: String
  UserData: String
  Volumes:
    - EC2 MountPoint
  AdditionalInfo: String
```

Properties

`Affinity`

Indicates whether Amazon Elastic Compute Cloud (Amazon EC2) always associates the instance with a dedicated host (p. 577). If you want Amazon EC2 to always restart the instance (if it was stopped) onto the same host on which it was launched, specify `host`. If you want Amazon EC2 to restart the instance on any available host, but to try to launch the instance onto the last host it ran on (on a best-effort basis), specify `default`.

Required: No

Type: String

Update requires: No interruption (p. 90)

`AvailabilityZone`

Specifies the name of the Availability Zone in which the instance is located.

For more information about AWS regions and Availability Zones, see Regions and Availability Zones in the *Amazon EC2 User Guide*.

Required: No. If not specified, an Availability Zone will be automatically chosen for you based on the load balancing criteria for the region.

Type: String

Update requires: Replacement (p. 90)

`BlockDeviceMappings`

Defines a set of Amazon Elastic Block Store block device mappings, ephemeral instance store block device mappings, or both. For more information, see Amazon Elastic Block Store or Amazon EC2 Instance Store in the *Amazon EC2 User Guide for Linux Instances*.

Required: No

Type: A list of Amazon EC2 Block Device Mapping Property (p. 1101).

Update requires: Replacement (p. 90). If you change only the `DeleteOnTermination` property for one or more block devices, update requires No interruption (p. 90).

`DisableApiTermination`

Specifies whether the instance can be terminated through the API.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`EbsOptimized`

Specifies whether the instance is optimized for Amazon Elastic Block Store I/O. This optimization provides dedicated throughput to Amazon EBS and an optimized configuration stack to provide optimal `EBS` I/O performance.

For more information about the instance types that can be launched as Amazon EBS optimized instances, see Amazon EBS-Optimized Instances in the *Amazon Elastic Compute Cloud User Guide*. Additional fees are incurred when using Amazon EBS-optimized instances.

Required: No. By default, AWS CloudFormation specifies `false`.

Type: Boolean

Update requires:

- *Update requires*: Some interruptions (p. 90) for Amazon EBS-backed instances
- *Update requires*: Replacement (p. 90) for instance store-backed instances

`HostId`

If you specify `host` for the `Affinity` property, the ID of a dedicated host that the instance is associated with. If you don't specify an ID, Amazon EC2 launches the instance onto any available, compatible dedicated host in your account. This type of launch is called an untargeted launch. Note that for untargeted launches, you must have a compatible, dedicated host available to successfully launch instances.

Required: No

Type: String

Update requires: No interruption (p. 90)

`IamInstanceProfile`

The physical ID of an instance profile or a reference to an AWS::IAM::InstanceProfile (p. 779) resource.

For more information about IAM roles, see Working with Roles in the *AWS Identity and Access Management User Guide*.

Required: No

Type: String

Update requires: Replacement (p. 90)

`ImageId`

Provides the unique ID of the Amazon Machine Image (AMI) that was assigned during registration.

Required: Yes

Type: String

Update requires: Updates are not supported.

`InstanceInitiatedShutdownBehavior`

Indicates whether an instance stops or terminates when you shut down the instance from the instance's operating system shutdown command. You can specify `stop` or `terminate`. For more information, see the RunInstances command in the *Amazon EC2 API Reference*.

Required: No

Type: String

Update requires: No interruption (p. 90)

`InstanceType`

The instance type, such as `t2.micro`. The default type is `"m3.medium"`. For a list of instance types, see Instance Families and Types.

Required: No

Type: String

Update requires:

- *Update requires*: Some interruptions (p. 90) for Amazon EBS-backed instances
- *Update requires*: Replacement (p. 90) for instance store-backed instances

`Ipv6AddressCount`

The number of IPv6 addresses to associate with the instance's primary network interface. Amazon EC2 automatically selects the IPv6 addresses from the subnet range. To specify specific IPv6 addresses, use the `Ipv6Addresses` property and don't specify this property.

For restrictions on which instance types support IPv6 addresses, see the RunInstances action in the *Amazon EC2 API Reference*.

Required: No

Type: Integer

Update requires: Replacement (p. 90)

`Ipv6Addresses`

One or more specific IPv6 addresses from the IPv6 CIDR block range of your subnet to associate with the instance's primary network interface. To specify a number of IPv6 addresses, use the `Ipv6AddressCount` property and don't specify this property.

For information about restrictions on which instance types support IPv6 addresses, see the RunInstances action in the *Amazon EC2 API Reference*.

Required: No

Type: List of EC2 NetworkInterface Ipv6Addresses (p. 1111)

Update requires: Replacement (p. 90)

`KernelId`

The kernel ID.

Required: No

Type: String

Update requires:

- *Update requires*: Some interruptions (p. 90) for Amazon EBS-backed instances
- *Update requires*: Replacement (p. 90) for instance store-backed instances

`KeyName`

Provides the name of the Amazon EC2 key pair.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Monitoring`

Specifies whether detailed monitoring is enabled for the instance.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`NetworkInterfaces`

A list of embedded objects that describes the network interfaces to associate with this instance.

Note
If you use this property to point to a network interface, you must terminate the original interface before attaching a new one to allow the update of the instance to succeed.
If this resource has a public IP address and is also in a VPC that is defined in the same template, you must use the `DependsOn` attribute to declare a dependency on the VPC-gateway attachment. For more information, see DependsOn Attribute (p. 1298).

Required: No

Type: A list of EC2 NetworkInterface Embedded Property Type (p. 1107)

Update requires: Replacement (p. 90)

`PlacementGroupName`

The name of an existing placement group that you want to launch the instance into (for cluster instances).

Required: No

Type: String

Update requires: Replacement (p. 90)

`PrivateIpAddress`

The private IP address for this instance.

Important
If you make an update to an instance that requires replacement, you must assign a new private IP address. During a replacement, AWS CloudFormation creates a new instance but doesn't delete the old instance until the stack has successfully updated. If the stack update

fails, AWS CloudFormation uses the old instance in order to roll back the stack to the previous working state. The old and new instances cannot have the same private IP address.

(Optional) If you're using Amazon VPC, you can use this parameter to assign the instance a specific available IP address from the subnet (for example, 10.0.0.25). By default, Amazon VPC selects an IP address from the subnet for the instance.

Required: No

Type: String

Update requires: Replacement (p. 90)

`RamdiskId`

The ID of the RAM disk to select. Some kernels require additional drivers at launch. Check the kernel requirements for information about whether you need to specify a RAM disk. To find kernel requirements, go to the AWS Resource Center and search for the kernel ID.

Required: No

Type: String

Update requires:

- *Update requires*: Some interruptions (p. 90) for Amazon EBS-backed instances
- *Update requires*: Replacement (p. 90) for instance store-backed instances

`SecurityGroupIds`

A list that contains the security group IDs for VPC security groups to assign to the Amazon EC2 instance. If you specified the `NetworkInterfaces` property, do not specify this property.

Required: Conditional. Required for VPC security groups.

Type: List of strings

Update requires:

- *Update requires*: No interruption (p. 90) for instances that are in a VPC.
- *Update requires*: Replacement (p. 90) for instances that are not in a VPC.

`SecurityGroups`

Valid only for Amazon EC2 security groups. A list that contains the Amazon EC2 security groups to assign to the Amazon EC2 instance. The list can contain both the name of existing Amazon EC2 security groups or references to AWS::EC2::SecurityGroup resources created in the template.

Required: No

Type: List of strings

Update requires: Replacement (p. 90).

`SourceDestCheck`

Controls whether source/destination checking is enabled on the instance. Also determines if an instance in a VPC will perform network address translation (NAT).

A value of `"true"` means that source/destination checking is enabled, and a value of `"false"` means that checking is disabled. For the instance to perform NAT, the value *must* be `"false"`. For more information, see NAT Instances in the *Amazon Virtual Private Cloud User Guide*.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`SsmAssociations`

The Amazon EC2 Systems Manager (SSM) document (p. 968) and parameter values to associate with this instance. To use this property, you must specify an IAM role for the instance. For more information, see Prerequisites for Remotely Running Commands on EC2 Instances in the *Amazon EC2 User Guide for Windows Instances*.

Note
You can currently associate only one document with an instance.

Required: No

Type: List of Amazon EC2 Instance SsmAssociations (p. 1105).

Update requires: No interruption (p. 90)

`SubnetId`

If you're using Amazon VPC, this property specifies the ID of the subnet that you want to launch the instance into. If you specified the `NetworkInterfaces` property, do not specify this property.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Tags`

An arbitrary set of tags (key–value pairs) for this instance.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90).

`Tenancy`

The tenancy of the instance that you want to launch, such as `default`, `dedicated`, or `host`. If you specify a tenancy value of `dedicated` or `host`, you must launch the instance in a VPC. For more information, see Dedicated Instances in the *Amazon VPC User Guide*.

Required: No

Type: String

Update requires:

- *Update requires*: No interruption (p. 90) if this property was set to `dedicated` and you change it to `host` or vice versa.
- *Update requires*: Replacement (p. 90) for all other changes.

`UserData`

Base64-encoded MIME user data that is made available to the instances.

Required: No

Type: String

Update requires:

- *Update requires*: Some interruptions (p. 90) for Amazon EBS-backed instances.

  Note
  For EBS-backed instances, changing the `UserData` stops and then starts the instance; however, Amazon EC2 doesn't automatically run the updated `UserData`. To update configurations on your instance, use the cfn-hup (p. 1363) helper script.
- *Update requires*: Replacement (p. 90) for instance store-backed instances.

`Volumes`

The Amazon EBS volumes to attach to the instance.

Note
Before detaching a volume, unmount any file systems on the device within your operating system. If you don't unmount the file system, a volume might get stuck in a busy state while detaching.

Required: No

Type: A list of EC2 MountPoints (p. 1106).

Update requires: No interruption (p. 90)

`AdditionalInfo`

Reserved.

Required: No

Type: String

Update requires:

- *Update requires*: Some interruptions (p. 90) for Amazon EBS-backed instances
- *Update requires*: Replacement (p. 90) for instance store-backed instances

Return Values

Ref

When you pass the logical ID of an AWS::EC2::Instance object to the intrinsic `Ref` function, the object's InstanceId is returned. For example: `i-636be302`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`AvailabilityZone`

The Availability Zone where the specified instance is launched. For example: `us-east-1b`.

You can retrieve a list of all Availability Zones for a region by using the `Fn::GetAZs` (p. 1332) intrinsic function.

`PrivateDnsName`

The private DNS name of the specified instance. For example: `ip-10-24-34-0.ec2.internal`.

`PublicDnsName`

The public DNS name of the specified instance. For example: `ec2-107-20-50-45.compute-1.amazonaws.com`.

`PrivateIp`

The private IP address of the specified instance. For example: `10.24.34.0`.

`PublicIp`

The public IP address of the specified instance. For example: `192.0.2.0`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Examples

EC2 Instance with an EBS Block Device Mapping

JSON

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Description" : "Ec2 block device mapping",
   "Resources" : {
      "MyEC2Instance" : {
         "Type" : "AWS::EC2::Instance",
         "Properties" : {
            "ImageId" : "ami-79fd7eee",
            "KeyName" : "testkey",
            "BlockDeviceMappings" : [
               {
                  "DeviceName" : "/dev/sdm",
                  "Ebs" : {
                     "VolumeType" : "io1",
                     "Iops" : "200",
                     "DeleteOnTermination" : "false",
                     "VolumeSize" : "20"
                  }
               },
               {
                  "DeviceName" : "/dev/sdk",
                  "NoDevice" : {}
               }
            ]
         }
      }
   }
}
```

YAML

```
  AWSTemplateFormatVersion: "2010-09-09"
  Description: "Ec2 block device mapping"
  Resources:
    MyEC2Instance:
      Type: "AWS::EC2::Instance"
```

```
      Properties:
        ImageId: "ami-79fd7eee"
        KeyName: "testkey"
        BlockDeviceMappings:
          - DeviceName: "/dev/sdm"
            Ebs:
              VolumeType: "io1"
              Iops: "200"
              DeleteOnTermination: "false"
              VolumeSize: "20"
          - DeviceName: "/dev/sdk"
            NoDevice: {}
```

Automatically Assign a Public IP Address

You can associate a public IP address with a network interface only if it has a device index of `0` and if it is a new network interface (not an existing one).

JSON

```
"Ec2Instance" : {
  "Type" : "AWS::EC2::Instance",
  "Properties" : {
    "ImageId" : { "Fn::FindInMap" : [ "RegionMap", { "Ref" : "AWS::Region" }, "AMI" ]},
    "KeyName" : { "Ref" : "KeyName" },
    "NetworkInterfaces": [ {
      "AssociatePublicIpAddress": "true",
      "DeviceIndex": "0",
      "GroupSet": [{ "Ref" : "myVPCEC2SecurityGroup" }],
      "SubnetId": { "Ref" : "PublicSubnet" }
    } ]
  }
}
```

YAML

```
  Ec2Instance:
    Type: "AWS::EC2::Instance"
    Properties:
      ImageId:
        Fn::FindInMap:
          - "RegionMap"
          - Ref: "AWS::Region"
          - "AMI"
      KeyName:
        Ref: "KeyName"
      NetworkInterfaces:
        - AssociatePublicIpAddress: "true"
          DeviceIndex: "0"
          GroupSet:
            - Ref: "myVPCEC2SecurityGroup"
          SubnetId:
            Ref: "PublicSubnet"
```

Other Examples

You can download templates that show how to use AWS::EC2::Instance to create a virtual private cloud (VPC):

- Single instance in a single subnet

- Multiple subnets with ELB and Auto Scaling group

For more information about an AWS::EC2::Instance that has an IAM instance profile, see: Create an EC2 instance with an associated instance profile.

For more information about Amazon EC2 template examples, see: Amazon EC2 Template Snippets (p. 268).

See Also

- RunInstances in the *Amazon Elastic Compute Cloud API Reference*
- EBS-Optimized Instances in the *Amazon Elastic Compute Cloud User Guide*

AWS::EC2::InternetGateway

Creates a new Internet gateway in your AWS account. After creating the Internet gateway, you then attach it to a VPC.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::InternetGateway",
   "Properties" : {
      "Tags (p. 585)" : [ Resource Tag, ... ]
   }
}
```

YAML

```
Type: "AWS::EC2::InternetGateway"
Properties:
  Tags (p. 585):
    - Resource Tag
```

Properties

`Tags`

An arbitrary set of tags (key–value pairs) for this resource.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90).

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

JSON

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "myInternetGateway" : {
         "Type" : "AWS::EC2::InternetGateway",
         "Properties" : {
            "Tags" : [ {"Key" : "foo", "Value" : "bar"}]
         }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  myInternetGateway:
    Type: "AWS::EC2::InternetGateway"
    Properties:
      Tags:
      - Key: foo
        Value: bar
```

Related Information

- CreateInternetGateway in the *Amazon EC2 API Reference*.
- Use the AWS::EC2::VPCGatewayAttachment (p. 647) resource to associate an Internet gateway with a VPC.

AWS::EC2::NatGateway

The `AWS::EC2::NatGateway` resource creates a network address translation (NAT) gateway in the specified public subnet. Use a NAT gateway to allow instances in a private subnet to connect to the Internet or to other AWS services, but prevent the Internet from initiating a connection with those instances. For more information and a sample architectural diagram, see NAT Gateways in the *Amazon VPC User Guide*.

Note

If you add a default route (`AWS::EC2::Route` resource) that points to a NAT gateway, specify NAT gateway's ID for the route's `NatGatewayId` property.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::EC2::NatGateway",
  "Properties" : {
    "AllocationId" : String,
    "SubnetId" : String
  }
}
```

YAML

```
Type: "AWS::EC2::NatGateway"
Properties:
  AllocationId: String
  SubnetId: String
```

Properties

`AllocationId`

The allocation ID of an Elastic IP address to associate with the NAT gateway. If the Elastic IP address is associated with another resource, you must first disassociate it.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`SubnetId`

The public subnet in which to create the NAT gateway.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Value

Ref

When you pass the logical ID of an `AWS::EC2::NatGateway` resource to the intrinsic `Ref` function, the function returns the ID of the NAT gateway, such as `nat-0a12bc456789de0fg`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a NAT gateway and a route that associates the NAT gateway with a route table. The route table must be associated with an Internet gateway so that the NAT gateway can connect to the Internet.

JSON

```
"NAT" : {
  "DependsOn" : "VPCGatewayAttach",
  "Type" : "AWS::EC2::NatGateway",
  "Properties" : {
    "AllocationId" : { "Fn::GetAtt" : ["EIP", "AllocationId"]},
    "SubnetId" : { "Ref" : "Subnet"}
  }
},
"EIP" : {
  "Type" : "AWS::EC2::EIP",
  "Properties" : {
    "Domain" : "vpc"
  }
},
"Route" : {
  "Type" : "AWS::EC2::Route",
  "Properties" : {
    "RouteTableId" : { "Ref" : "RouteTable" },
    "DestinationCidrBlock" : "0.0.0.0/0",
    "NatGatewayId" : { "Ref" : "NAT" }
  }
}
```

YAML

```
NAT:
  DependsOn: VPCGatewayAttach
  Type: AWS::EC2::NatGateway
  Properties:
    AllocationId:
      Fn::GetAtt:
      - EIP
      - AllocationId
    SubnetId:
      Ref: Subnet
EIP:
  Type: AWS::EC2::EIP
  Properties:
    Domain: vpc
Route:
  Type: AWS::EC2::Route
  Properties:
    RouteTableId:
      Ref: RouteTable
```

```
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT
```

AWS::EC2::NetworkAcl

Creates a new network ACL in a VPC.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::NetworkAcl",
   "Properties" : {
      "Tags (p. 589)" : [ Resource Tag, ... ],
      "VpcId (p. 589)" : String
   }
}
```

YAML

```
Type: "AWS::EC2::NetworkAcl"
Properties:
  Tags (p. 589):
    - Resource Tag
  VpcId (p. 589): String
```

Properties

`Tags`

An arbitrary set of tags (key–value pairs) for this ACL.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90).

`VpcId`

The ID of the VPC where the network ACL will be created.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

JSON

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "myNetworkAcl" : {
         "Type" : "AWS::EC2::NetworkAcl",
         "Properties" : {
            "VpcId" : { "Ref" : "myVPC" },
            "Tags" : [ { "Key" : "foo", "Value" : "bar" } ]
         }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  myNetworkAcl:
    Type: AWS::EC2::NetworkAcl
    Properties:
      VpcId:
        Ref: myVPC
      Tags:
      - Key: foo
        Value: bar
```

See Also

- CreateNetworkAcl in the *Amazon EC2 API Reference*
- Network ACLs in the *Amazon Virtual Private Cloud User Guide*.

AWS::EC2::NetworkAclEntry

Creates an entry (i.e., a rule) in a network ACL with a rule number you specify. Each network ACL has a set of numbered ingress rules and a separate set of numbered egress rules.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::NetworkAclEntry",
   "Properties" : {
      "CidrBlock (p. 591)" : String,
      "Egress (p. 591)" : Boolean,
      "Icmp (p. 592)" : EC2 ICMP,
      "Ipv6CidrBlock" : String,
      "NetworkAclId (p. 592)" : String,
      "PortRange (p. 592)" : EC2 PortRange,
      "Protocol (p. 592)" : Integer,
      "RuleAction (p. 592)" : String,
      "RuleNumber (p. 593)" : Integer
   }
}
```

YAML

```
Type: "AWS::EC2::NetworkAclEntry"
Properties:
  CidrBlock (p. 591): String
  Egress (p. 591): Boolean
  Icmp (p. 592):
    EC2 ICMP
  Ipv6CidrBlock: String
  NetworkAclId (p. 592): String
  PortRange (p. 592):
    EC2 PortRange
  Protocol (p. 592): Integer
  RuleAction (p. 592) : String
  RuleNumber (p. 593) : Integer
```

Properties

`CidrBlock`

The IPv4 CIDR range to allow or deny, in CIDR notation (e.g., 172.16.0.0/24).

Required: Conditional. You must specify the `CidrBlock` or `Ipv6CidrBlock` property.

Type: String

Update requires: No interruption (p. 90)

`Egress`

Whether this rule applies to egress traffic from the subnet (`true`) or ingress traffic to the subnet (`false`). By default, AWS CloudFormation specifies `false`.

Required: No

Type: Boolean

Update requires: Replacement (p. 90).

`Icmp`

The Internet Control Message Protocol (ICMP) code and type.

Required: Conditional required if specifying 1 (ICMP) for the protocol parameter.

Type: EC2 NetworkAclEntry Icmp (p. 1110)

Update requires: No interruption (p. 90)

`Ipv6CidrBlock`

The IPv6 CIDR range to allow or deny, in CIDR notation.

Required: Conditional. You must specify the `CidrBlock` or `Ipv6CidrBlock` property.

Type: String

Update requires: No interruption (p. 90)

`NetworkAclId`

ID of the ACL where the entry will be created.

Required: Yes

Type: String

Update requires: Replacement (p. 90).

`PortRange`

The range of port numbers for the UDP/TCP protocol.

Required: Conditional Required if specifying 6 (TCP) or 17 (UDP) for the protocol parameter.

Type: EC2 NetworkAclEntry PortRange (p. 1111)

Update requires: No interruption (p. 90)

`Protocol`

The IP protocol that the rule applies to. You must specify `-1` or a protocol number (go to Protocol Numbers at iana.org). You can specify `-1` for all protocols.

Note
If you specify `-1`, all ports are opened and the `PortRange` property is ignored.

Required: Yes

Type: Number

Update requires: No interruption (p. 90)

`RuleAction`

Whether to allow or deny traffic that matches the rule; valid values are "allow" or "deny".

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`RuleNumber`

Rule number to assign to the entry, such as `100`. ACL entries are processed in ascending order by rule number. Entries can't use the same rule number unless one is an egress rule and the other is an ingress rule. For valid values, see the CreateNetworkAclEntry action in the *Amazon EC2 API Reference*.

Required: Yes

Type: Number

Update requires: Replacement (p. 90).

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

JSON

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "myNetworkAclEntry" : {
         "Type" : "AWS::EC2::NetworkAclEntry",
         "Properties" : {
            "NetworkAclId" : { "Ref" : "myNetworkAcl" },
            "RuleNumber" : "100",
            "Protocol" : "-1",
            "RuleAction" : "allow",
            "Egress" : "true",
            "CidrBlock" : "172.16.0.0/24",
            "Icmp" : { "Code" : "-1", "Type" : "-1" },
            "PortRange" : { "From" : "53", "To" : "53" }
         }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  myNetworkAclEntry:
    Type: AWS::EC2::NetworkAclEntry
    Properties:
      NetworkAclId:
        Ref: myNetworkAcl
      RuleNumber: '100'
      Protocol: "-1"
```

```
      RuleAction: allow
      Egress: 'true'
      CidrBlock: 172.16.0.0/24
      Icmp:
        Code: "-1"
        Type: "-1"
      PortRange:
        From: '53'
        To: '53'
```

See Also

- NetworkAclEntry in the *Amazon EC2 API Reference*
- Network ACLs in the *Amazon Virtual Private Cloud User Guide*.

AWS::EC2::NetworkInterface

Describes a network interface in an Elastic Compute Cloud (EC2) instance for AWS CloudFormation. This is provided in a list in the `NetworkInterfaces` property of AWS::EC2::Instance (p. 574).

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::NetworkInterface",
   "Properties" : {
      "Description (p. 595)" : String,
      "GroupSet (p. 595)" : [ String, ... ],
      "Ipv6AddressCount" : Integer,
      "Ipv6Addresses" : [ Ipv6Address, ... ],
      "PrivateIpAddress (p. 596)" : String,
      "PrivateIpAddresses (p. 596)" : [ PrivateIpAddressSpecification, ... ],
      "SecondaryPrivateIpAddressCount (p. 596)" : Integer,
      "SourceDestCheck (p. 596)" : Boolean,
      "SubnetId (p. 596)" : String,
      "Tags (p. 597)" : [ Resource Tag, ... ]
   }
}
```

YAML

```
Type: "AWS::EC2::NetworkInterface"
Properties:
  Description (p. 595): String
  GroupSet (p. 595):
    - String
  Ipv6AddressCount: Integer
  Ipv6Addresses:
    - Ipv6Address
  PrivateIpAddress (p. 596): String
  PrivateIpAddresses (p. 596):
    - PrivateIpAddressSpecification
  SecondaryPrivateIpAddressCount (p. 596): Integer
  SourceDestCheck (p. 596): Boolean
  SubnetId (p. 596): String
  Tags (p. 597):
    - Resource Tag
```

Properties

`Description`

The description of this network interface.

Required: No

Type: String

Update requires: No interruption (p. 90).

`GroupSet`

A list of security group IDs associated with this network interface.

Required: No

Type: List of strings.

Update requires: No interruption (p. 90)

`Ipv6AddressCount`

The number of IPv6 addresses to associate with the network interface. Amazon EC2 automatically selects the IPv6 addresses from the subnet range. To specify specific IPv6 addresses, use the `Ipv6Addresses` property and don't specify this property.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`Ipv6Addresses`

One or more specific IPv6 addresses from the IPv6 CIDR block range of your subnet to associate with the network interface. If you're specifying a number of IPv6 addresses, use the `Ipv6AddressCount` property and don't specify this property.

Required: No

Type: List of EC2 NetworkInterface Ipv6Addresses (p. 1111)

Update requires: No interruption (p. 90)

`PrivateIpAddress`

Assigns a single private IP address to the network interface, which is used as the primary private IP address. If you want to specify multiple private IP address, use the `PrivateIpAddresses` property.

Required: No

Type: String

Update requires: Replacement (p. 90).

`PrivateIpAddresses`

Assigns a list of private IP addresses to the network interface. You can specify a primary private IP address by setting the value of the `Primary` property to `true` in the `PrivateIpAddressSpecification` property. If you want Amazon EC2 to automatically assign private IP addresses, use the `SecondaryPrivateIpAddressCount` property and do not specify this property.

For information about the maximum number of private IP addresses, see Private IP Addresses Per ENI Per Instance Type in the *Amazon EC2 User Guide for Linux Instances*.

Required: No

Type: list of PrivateIpAddressSpecification (p. 1112).

Update requires: Replacement (p. 90) if you change the primary private IP address. If not, update requires No interruption (p. 90).

`SecondaryPrivateIpAddressCount`

The number of secondary private IP addresses that Amazon EC2 automatically assigns to the network interface. Amazon EC2 uses the value of the `PrivateIpAddress` property as the primary private IP address. If you don't specify that property, Amazon EC2 automatically assigns both the primary and secondary private IP addresses.

If you want to specify your own list of private IP addresses, use the `PrivateIpAddresses` property and do not specify this property.

For information about the maximum number of private IP addresses, see Private IP Addresses Per ENI Per Instance Type in the *Amazon EC2 User Guide for Linux Instances*.

Required: No

Type: Integer.

Update requires: No interruption (p. 90).

`SourceDestCheck`

Flag indicating whether traffic to or from the instance is validated.

Required: No

Type: Boolean

Update requires: No interruption (p. 90).

`SubnetId`

The ID of the subnet to associate with the network interface.

Required: Yes

Type: String

Update requires: Replacement (p. 90).

`Tags`

An arbitrary set of tags (key–value pairs) for this network interface.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90).

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`PrimaryPrivateIpAddress`

Returns the primary private IP address of the network interface. For example, `10.0.0.192`.

`SecondaryPrivateIpAddresses`

Returns the secondary private IP addresses of the network interface. For example, `["10.0.0.161", "10.0.0.162", "10.0.0.163"]`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Template Examples

Tip

For more NetworkInterface template examples, see Elastic Network Interface (ENI) Template Snippets (p. 271).

Simple Standalone ENI

This is a simple standalone Elastic Network Interface (ENI), using all of the available properties.

JSON

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Description" : "Simple Standalone ENI",
   "Resources" : {
      "myENI" : {
         "Type" : "AWS::EC2::NetworkInterface",
```

```
            "Properties" : {
               "Tags": [{"Key":"foo","Value":"bar"}],
               "Description": "A nice description.",
               "SourceDestCheck": "false",
               "GroupSet": ["sg-75zzz219"],
               "SubnetId": "subnet-3z648z53",
               "PrivateIpAddress": "10.0.0.16"
            }
         }
      }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: Simple Standalone ENI
Resources:
  myENI:
    Type: AWS::EC2::NetworkInterface
    Properties:
      Tags:
      - Key: foo
        Value: bar
      Description: A nice description.
      SourceDestCheck: 'false'
      GroupSet:
      - sg-75zzz219
      SubnetId: subnet-3z648z53
      PrivateIpAddress: 10.0.0.16
```

ENI on an EC2 instance

This is an example of an ENI on an EC2 instance. In this example, one ENI is added to the instance. If you want to add more than one ENI, you can specify a list for the `NetworkInterface` property. However, you can specify multiple ENIs only if all the ENIs have just private IP addresses (no associated public IP address). If you have an ENI with a public IP address, specify it and then use the `AWS::EC2::NetworkInterfaceAttachment` resource to add additional ENIs.

JSON

```

"Ec2Instance" : {
   "Type" : "AWS::EC2::Instance",
   "Properties" : {
      "ImageId" : { "Fn::FindInMap" : [ "RegionMap", { "Ref" : "AWS::Region" }, "AMI" ]},
      "KeyName" : { "Ref" : "KeyName" },
      "SecurityGroupIds" : [{ "Ref" : "WebSecurityGroup" }],
      "SubnetId" : { "Ref" : "SubnetId" },
      "NetworkInterfaces" : [ {
         "NetworkInterfaceId" : {"Ref" : "controlXface"}, "DeviceIndex" : "1" } ],
      "Tags" : [ {"Key" : "Role", "Value" : "Test Instance"}],
      "UserData" : { "Fn::Base64" : { "Ref" : "WebServerPort" }}
   }
}
```

YAML

```
Ec2Instance:
  Type: AWS::EC2::Instance
  Properties:
    ImageId:
```

```
      Fn::FindInMap:
      - RegionMap
      - Ref: AWS::Region
      - AMI
    KeyName:
      Ref: KeyName
    SecurityGroupIds:
    - Ref: WebSecurityGroup
    SubnetId:
      Ref: SubnetId
    NetworkInterfaces:
    - NetworkInterfaceId:
        Ref: controlXface
      DeviceIndex: '1'
    Tags:
    - Key: Role
      Value: Test Instance
    UserData:
      Fn::Base64:
        Ref: WebServerPort
```

More Info

- NetworkInterfaceType in the *Amazon Elastic Compute Cloud API Reference*

AWS::EC2::NetworkInterfaceAttachment

Attaches an elastic network interface (ENI) to an Amazon EC2 instance. You can use this resource type to attach additional network interfaces to an instances without interruption.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::NetworkInterfaceAttachment",
   "Properties" : {
      "DeleteOnTermination (p. 600)": Boolean,
      "DeviceIndex (p. 600)": String,
      "InstanceId (p. 600)": String,
      "NetworkInterfaceId (p. 600)": String
   }
}
```

YAML

```
Type: "AWS::EC2::NetworkInterfaceAttachment"
Properties:
  DeleteOnTermination (p. 600): Boolean
  DeviceIndex (p. 600): String
  InstanceId (p. 600): String
  NetworkInterfaceId (p. 600): String
```

Properties

`DeleteOnTermination`

Whether to delete the network interface when the instance terminates. By default, this value is set to `True`.

Required: No

Type: Boolean.

Update requires: No interruption (p. 90)

`DeviceIndex`

The network interface's position in the attachment order. For example, the first attached network interface has a `DeviceIndex` of `0`.

Required: Yes.

Type: String.

Update requires: No interruption (p. 90)

`InstanceId`

The ID of the instance to which you will attach the ENI.

Required: Yes.

Type: String.

Update requires: No interruption (p. 90)

`NetworkInterfaceId`

The ID of the ENI that you want to attach.

Required: Yes.

Type: String.

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

Attaching `MyNetworkInterface` to `MyInstance`

JSON

```
"NetworkInterfaceAttachment" : {
    "Type" : "AWS::EC2::NetworkInterfaceAttachment",
        "Properties" : {
            "InstanceId" : {"Ref" : "MyInstance"},
            "NetworkInterfaceId" : {"Ref" : "MyNetworkInterface"},
            "DeviceIndex" : "1"
        }
}
```

YAML

```
NetworkInterfaceAttachment:
  Type: AWS::EC2::NetworkInterfaceAttachment
  Properties:
    InstanceId:
      Ref: MyInstance
    NetworkInterfaceId:
      Ref: MyNetworkInterface
    DeviceIndex: 1
```

AWS::EC2::PlacementGroup

The `AWS::EC2::PlacementGroup` resource is a logical grouping of instances within a single Availability Zone (AZ) that enables applications to participate in a low-latency, 10 Gbps network. You create a placement group first, and then you can launch instances in the placement group.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::EC2::PlacementGroup",
  "Properties" : {
    "Strategy" : String
  }
}
```

YAML

```
Type: "AWS::EC2::PlacementGroup"
Properties:
  Strategy: String
```

Properties

`Strategy`

The placement strategy, which relates to the instance types that can be added to the placement group. For example, for the `cluster` strategy, you can cluster C4 instance types but not T2 instance types. For valid values, see CreatePlacementGroup in the *Amazon EC2 API Reference*. By default, AWS CloudFormation sets the value of this property to `cluster`.

Required: No

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a placement group with a `cluster` placement strategy.

JSON

```
"PlacementGroup" : {
  "Type" : "AWS::EC2::PlacementGroup",
  "Properties" : {
    "Strategy" : "cluster"
  }
}
```

YAML

```
PlacementGroup:
  Type: AWS::EC2::PlacementGroup
  Properties:
    Strategy: cluster
```

AWS::EC2::Route

Creates a new route in a route table within a VPC. The route's target can be either a gateway attached to the VPC or a NAT instance in the VPC.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::EC2::Route",
  "Properties" : {
    "DestinationCidrBlock (p. 603)" : String,
    "DestinationIpv6CidrBlock" : String,
    "GatewayId (p. 604)" : String,
    "InstanceId (p. 604)" : String,
    "NatGatewayId" : String,
    "NetworkInterfaceId (p. 604)" : String,
    "RouteTableId (p. 604)" : String,
    "VpcPeeringConnectionId" : String
  }
}
```

YAML

```
Type: "AWS::EC2::Route"
Properties:
  DestinationCidrBlock (p. 603): String
  DestinationIpv6CidrBlock: String
  GatewayId (p. 604): String
  InstanceId (p. 604): String
  NatGatewayId: String
  NetworkInterfaceId (p. 604): String
  RouteTableId (p. 604): String
  VpcPeeringConnectionId: String
```

Properties

`DestinationCidrBlock`

The IPv4 CIDR address block used for the destination match. For example, `0.0.0.0/0`. Routing decisions are based on the most specific match.

Required: Conditional. You must specify the `DestinationCidrBlock` or `DestinationIpv6CidrBlock` property.

Type: String

Update requires: Replacement (p. 90)

`DestinationIpv6CidrBlock`

The IPv6 CIDR address block used for the destination match. For example, `::/0`. Routing decisions are based on the most specific match.

Required: Conditional. You must specify the `DestinationCidrBlock` or `DestinationIpv6CidrBlock` property.

Type: String

Update requires: Replacement (p. 90)

`GatewayId`

The ID of an Internet gateway or virtual private gateway that is attached to your VPC. For example: `igw-eaad4883`.

For route entries that specify a gateway, you must specify a dependency on the gateway attachment resource. For more information, see DependsOn Attribute (p. 1298).

Required: Conditional. You must specify only one of the following properties: `GatewayId`, `InstanceId`, `NatGatewayId`, `NetworkInterfaceId`, or `VpcPeeringConnectionId`.

Type: String

Update requires: No interruption (p. 90)

`InstanceId`

The ID of a NAT instance in your VPC. For example, `i-1a2b3c4d`.

Required: Conditional. You must specify only one of the following properties: `GatewayId`, `InstanceId`, `NatGatewayId`, `NetworkInterfaceId`, or `VpcPeeringConnectionId`.

Type: String

Update requires: No interruption (p. 90)

`NatGatewayId`

The ID of a NAT gateway. For example, `nat-0a12bc456789de0fg`.

Required: Conditional. You must specify only one of the following properties: `GatewayId`, `InstanceId`, `NatGatewayId`, `NetworkInterfaceId`, or `VpcPeeringConnectionId`.

Type: String

Update requires: No interruption (p. 90)

`NetworkInterfaceId`

Allows the routing of network interface IDs.

Required: Conditional. You must specify only one of the following properties: `GatewayId`, `InstanceId`, `NatGatewayId`, `NetworkInterfaceId`, or `VpcPeeringConnectionId`.

Type: String

Update requires: No interruption (p. 90)

`RouteTableId`

The ID of the route table (p. 606) where the route will be added.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`VpcPeeringConnectionId`

The ID of a VPC peering connection.

Required: Conditional. You must specify only one of the following properties: `GatewayId`, `InstanceId`, `NatGatewayId`, `NetworkInterfaceId`, or `VpcPeeringConnectionId`.

Type: String

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

The following example creates a route that is added to a gateway.

JSON

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "myRoute" : {
         "Type" : "AWS::EC2::Route",
         "DependsOn" : "GatewayToInternet",
         "Properties" : {
            "RouteTableId" : { "Ref" : "myRouteTable" },
            "DestinationCidrBlock" : "0.0.0.0/0",
            "GatewayId" : { "Ref" : "myInternetGateway" }
         }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  myRoute:
    Type: AWS::EC2::Route
    DependsOn: GatewayToInternet
    Properties:
      RouteTableId:
        Ref: myRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId:
        Ref: myInternetGateway
```

More Info

- AWS::EC2::RouteTable (p. 606)

- CreateRoute in the *Amazon EC2 API Reference*
- Route Tables in the *Amazon VPC User Guide*.

AWS::EC2::RouteTable

Creates a new route table within a VPC. After you create a new route table, you can add routes and associate the table with a subnet.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::RouteTable",
   "Properties" : {
      "VpcId (p. 606)" : String,
      "Tags (p. 607)" : [ Resource Tag, ... ]
   }
}
```

YAML

```
Type: "AWS::EC2::RouteTable"
Properties:
  VpcId (p. 606): String
  Tags (p. 607):
    - Resource Tag
```

Properties

`VpcId`

The ID of the VPC where the route table will be created.

Example: vpc-11ad4878

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Tags`

An arbitrary set of tags (key–value pairs) for this route table.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90).

Return Values

Ref

When you specify an `AWS::EC2::RouteTable` type as an argument to the `Ref` function, AWS CloudFormation returns the route table ID, such as `rtb-12a34567`.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

The following example snippet uses the VPC ID from a VPC named *myVPC* that was declared elsewhere in the same template.

JSON

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "myRouteTable" : {
         "Type" : "AWS::EC2::RouteTable",
         "Properties" : {
            "VpcId" : { "Ref" : "myVPC" },
            "Tags" : [ { "Key" : "foo", "Value" : "bar" } ]
         }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  myRouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId:
        Ref: myVPC
      Tags:
      - Key: foo
        Value: bar
```

See Also

- AWS::EC2::Route (p. 602)

- CreateRouteTable in the *Amazon EC2 API Reference*
- Route Tables in the *Amazon VPC User Guide*
- Using Tags in the *Amazon Elastic Compute Cloud User Guide*

AWS::EC2::SecurityGroup

Creates an Amazon EC2 security group. To create a VPC security group, use the VpcId (p. 609) property.

This type supports updates. For more information about updating stacks, see AWS CloudFormation Stacks Updates (p. 89).

> **Important**
> If you want to cross-reference two security groups in the ingress and egress rules of those security groups, use the AWS::EC2::SecurityGroupEgress (p. 611) and AWS::EC2::SecurityGroupIngress (p. 616) resources to define your rules. Do not use the embedded ingress and egress rules in the `AWS::EC2::SecurityGroup`. If you do, it causes a circular dependency, which AWS CloudFormation doesn't allow.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::EC2::SecurityGroup",
  "Properties" : {
     "GroupDescription (p. 609)" : String,
     "SecurityGroupEgress (p. 609)" : [ Security Group Rule, ... ],
     "SecurityGroupIngress (p. 609)" : [ Security Group Rule, ... ],
     "Tags" :  [ Resource Tag, ... ],
     "VpcId (p. 609)" : String
  }
}
```

YAML

```
Type: "AWS::EC2::SecurityGroup"
Properties:
  GroupDescription (p. 609): String
  SecurityGroupEgress (p. 609):
    - Security Group Rule
  SecurityGroupIngress (p. 609):
    - Security Group Rule
  Tags:
```

```
    - Resource Tag
  VpcId (p. 609): String
```

Properties

`GroupDescription`

Description of the security group.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`SecurityGroupEgress`

A list of Amazon EC2 security group egress rules.

Required: No

Type: List of EC2 Security Group Rule (p. 1112)

Update requires: No interruption (p. 90)

`SecurityGroupIngress`

A list of Amazon EC2 security group ingress rules.

Required: No

Type: List of EC2 Security Group Rule (p. 1112)

Update requires: No interruption (p. 90)

`Tags`

The tags that you want to attach to the resource.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236).

Update requires: No interruption (p. 90).

`VpcId`

The physical ID of the VPC. Can be obtained by using a reference to an AWS::EC2::VPC (p. 639), such as: `{ "Ref" : "myVPC" }`.

For more information about using the `Ref` function, see Ref (p. 1343).

Required: Yes, for VPC security groups without a default VPC

Type: String

Update requires: Replacement (p. 90)

> **Note**
> For more information about VPC security groups, go to Security Groups in the *Amazon VPC User Guide*.

Return Values

Ref

When you specify an `AWS::EC2::SecurityGroup` type as an argument to the `Ref` function, AWS CloudFormation returns the security group name or the security group ID (for EC2-VPC security groups that are not in a default VPC).

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`GroupId`

The group ID of the specified security group, such as `sg-94b3a1f6`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Examples

Basic Ingress and Egress Rules

The following sample defines a security group with an ingress and egress rule:

JSON

```
"InstanceSecurityGroup" : {
   "Type" : "AWS::EC2::SecurityGroup",
   "Properties" : {
      "GroupDescription" : "Allow http to client host",
      "VpcId" : {"Ref" : "myVPC"},
      "SecurityGroupIngress" : [{
            "IpProtocol" : "tcp",
            "FromPort" : "80",
            "ToPort" : "80",
            "CidrIp" : "0.0.0.0/0"
         }],
      "SecurityGroupEgress" : [{
         "IpProtocol" : "tcp",
         "FromPort" : "80",
         "ToPort" : "80",
         "CidrIp" : "0.0.0.0/0"
      }]
   }
}
```

YAML

```
InstanceSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Allow http to client host
    VpcId:
      Ref: myVPC
    SecurityGroupIngress:
    - IpProtocol: tcp
```

```
      FromPort: '80'
      ToPort: '80'
      CidrIp: 0.0.0.0/0
    SecurityGroupEgress:
    - IpProtocol: tcp
      FromPort: '80'
      ToPort: '80'
      CidrIp: 0.0.0.0/0
```

Remove Default Rule

When you create a VPC security group, Amazon EC2 creates a default egress rule that allows egress traffic on all ports and IP protocols to any location. The default rule is removed only when you specify one or more egress rules. If you want to remove the default rule and limit egress traffic to just the localhost (`127.0.0.1/32`), you can use the following sample:

JSON

```
"sgwithoutegress": {
  "Type": "AWS::EC2::SecurityGroup",
  "Properties": {
    "GroupDescription": "Limits security group egress traffic",
    "SecurityGroupEgress": [
      {
        "CidrIp": "127.0.0.1/32",
        "IpProtocol": "-1"
      }
    ],
    "VpcId": { "Ref": "myVPC"}
  }
}
```

YAML

```
sgwithoutegress:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Limits security group egress traffic
    SecurityGroupEgress:
    - CidrIp: 127.0.0.1/32
      IpProtocol: "-1"
    VpcId:
      Ref: myVPC
```

See Also

- Using Security Groups in the *Amazon EC2 User Guide for Linux Instances*.
- Security Groups in the *Amazon VPC User Guide*.

AWS::EC2::SecurityGroupEgress

The `AWS::EC2::SecurityGroupEgress` resource adds an egress rule to an Amazon VPC security group.

> **Important**
> Use `AWS::EC2::SecurityGroupIngress` and `AWS::EC2::SecurityGroupEgress` only when necessary, typically to allow security groups to reference each other in ingress and egress rules. Otherwise, use the embedded ingress and egress rules of AWS::EC2::SecurityGroup (p. 608). For more information, see Amazon EC2 Security Groups.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::EC2::SecurityGroupEgress",
  "Properties" : {
    "CidrIp" : String,
    "CidrIpv6" : String,
    "DestinationPrefixListId" : String,
    "DestinationSecurityGroupId" : String,
    "FromPort" : Integer,
    "GroupId" : String,
    "IpProtocol" : String,
    "ToPort" : Integer
  }
}
```

YAML

```
Type: "AWS::EC2::SecurityGroupEgress"
Properties:
  CidrIp: String
  CidrIpv6: String
  DestinationPrefixListId: String
  DestinationSecurityGroupId:  String
  FromPort: Integer
  GroupId: String
  IpProtocol: String
  ToPort: Integer
```

Properties

For more information about adding egress rules to VPC security groups, go to AuthorizeSecurityGroupEgress in the *Amazon EC2 API Reference*.

> **Note**
> If you change this resource's logical ID, you must also update a property value in order to trigger an update for this resource.

`CidrIp`

An IPv4 CIDR range.

Required: Conditional. You must specify a destination security group (`DestinationPrefixListId` or `DestinationSecurityGroupId`) or a CIDR range (`CidrIp` or `CidrIpv6`).

Type: String

Update requires: Replacement (p. 90)

`CidrIpv6`

An IPv6 CIDR range.

Type: String

Required: Conditional. You must specify a destination security group (`DestinationPrefixListId` or `DestinationSecurityGroupId`) or a CIDR range (`CidrIp` or `CidrIpv6`).

Update requires: Replacement (p. 90)

`DestinationPrefixListId`

The AWS service prefix of an Amazon VPC endpoint. For more information, see VPC Endpoints in the *Amazon VPC User Guide*.

Required: Conditional. You must specify a destination security group (`DestinationPrefixListId` or `DestinationSecurityGroupId`) or a CIDR range (`CidrIp` or `CidrIpv6`).

Type: String

Update requires: Replacement (p. 90)

`DestinationSecurityGroupId`

Specifies the group ID of the destination Amazon VPC security group.

Required: Conditional. You must specify a destination security group (`DestinationPrefixListId` or `DestinationSecurityGroupId`) or a CIDR range (`CidrIp` or `CidrIpv6`).

Type: String

Update requires: Replacement (p. 90)

`FromPort`

Start of port range for the TCP and UDP protocols, or an ICMP type number. If you specify `icmp` for the `IpProtocol` property, you can specify -1 as a wildcard (i.e., any ICMP type number).

Required: Yes

Type: Integer

Update requires: Replacement (p. 90)

`GroupId`

ID of the Amazon VPC security group to modify. This value can be a reference to an AWS::EC2::SecurityGroup (p. 608) resource that has a valid `VpcId` property or the ID of an existing Amazon VPC security group.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`IpProtocol`

IP protocol name or number. For valid values, see the IpProtocol parameter in AuthorizeSecurityGroupIngress

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`ToPort`

End of port range for the TCP and UDP protocols, or an ICMP code. If you specify `icmp` for the `IpProtocol` property, you can specify -1 as a wildcard (i.e., any ICMP code).

Required: Yes

Type: Integer

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

VPC Security Groups Example

In some cases, you might have an originating (source) security group to which you want to add an outbound rule that allows traffic to a destination (target) security group. The target security group also needs an inbound rule that allows traffic from the source security group. Note that you cannot use the `Ref` function to specify the outbound and inbound rules for each security group. Doing so creates a circular dependency; you cannot have two resources that depend on each other. Instead, use the egress and ingress resources to declare these outbound and inbound rules, as shown in the following template snippet.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Resources": {
    "SourceSG": {
      "Type": "AWS::EC2::SecurityGroup",
      "Properties": {
        "VpcId" : "vpc-e063f789",
        "GroupDescription": "Sample source security group"
      }
    },
    "TargetSG": {
      "Type": "AWS::EC2::SecurityGroup",
      "Properties": {
        "VpcId" : "vpc-e063f789",
        "GroupDescription": "Sample target security group"
      }
    },
    "OutboundRule": {
      "Type": "AWS::EC2::SecurityGroupEgress",
      "Properties":{
        "IpProtocol": "tcp",
        "FromPort": "0",
        "ToPort": "65535",
        "DestinationSecurityGroupId": {
          "Fn::GetAtt": [
            "TargetSG",
            "GroupId"
          ]
        },
```

```
        "GroupId": {
          "Fn::GetAtt": [
            "SourceSG",
            "GroupId"
          ]
        }
      }
    },
    "InboundRule": {
      "Type": "AWS::EC2::SecurityGroupIngress",
      "Properties":{
        "IpProtocol": "tcp",
        "FromPort": "0",
        "ToPort": "65535",
        "SourceSecurityGroupId": {
          "Fn::GetAtt": [
            "SourceSG",
            "GroupId"
          ]
        },
        "GroupId": {
          "Fn::GetAtt": [
            "TargetSG",
            "GroupId"
          ]
        }
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  SourceSG:
    Type: AWS::EC2::SecurityGroup
    Properties:
      VpcId: vpc-e063f789
      GroupDescription: Sample source security group
  TargetSG:
    Type: AWS::EC2::SecurityGroup
    Properties:
      VpcId: vpc-e063f789
      GroupDescription: Sample target security group
  OutboundRule:
    Type: AWS::EC2::SecurityGroupEgress
    Properties:
      IpProtocol: tcp
      FromPort: '0'
      ToPort: '65535'
      DestinationSecurityGroupId:
        Fn::GetAtt:
        - TargetSG
        - GroupId
      GroupId:
        Fn::GetAtt:
        - SourceSG
        - GroupId
  InboundRule:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      IpProtocol: tcp
      FromPort: '0'
```

```
      ToPort: '65535'
      SourceSecurityGroupId:
        Fn::GetAtt:
        - SourceSG
        - GroupId
      GroupId:
        Fn::GetAtt:
        - TargetSG
        - GroupId
```

AWS::EC2::SecurityGroupIngress

The `AWS::EC2::SecurityGroupIngress` resource adds an ingress rule to an Amazon EC2 or Amazon VPC security group.

> **Important**
> Use `AWS::EC2::SecurityGroupIngress` and `AWS::EC2::SecurityGroupEgress` only when necessary, typically to allow security groups to reference each other in ingress and egress rules. Otherwise, use the embedded ingress and egress rules of AWS::EC2::SecurityGroup (p. 608). For more information, see Amazon EC2 Security Groups.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::EC2::SecurityGroupIngress",
  "Properties" : {
    "CidrIp" : String,
    "CidrIpv6" : String,
    "FromPort" : Integer,
    "GroupId" : String,
    "GroupName" : String,
    "IpProtocol" : String,
    "SourceSecurityGroupName" : String,
    "SourceSecurityGroupId" : String,
    "SourceSecurityGroupOwnerId" : String,
    "ToPort" : Integer
  }
}
```

YAML

```

Type: "AWS::EC2::SecurityGroupIngress"
Properties:
  CidrIp: String
  CidrIpv6: String
  FromPort: Integer
  GroupId: String
  GroupName: String
```

```
  IpProtocol: String
  SourceSecurityGroupName: String
  SourceSecurityGroupId: String
  SourceSecurityGroupOwnerId: String
  ToPort: Integer
```

Properties

For more information about adding ingress rules to Amazon EC2 or VPC security groups, see AuthorizeSecurityGroupIngress in the *Amazon EC2 API Reference*.

Note
If you change this resource's logical ID, you must also update a property value in order to trigger an update for this resource.

`CidrIp`

An IPv4 CIDR range.

For an overview of CIDR ranges, go to the Wikipedia Tutorial.

Type: String

Required: Conditional. You must specify a source security group (`SourceSecurityGroupName` or `SourceSecurityGroupId`) or a CIDR range (`CidrIp` or `CidrIpv6`).

Update requires: Replacement (p. 90)

`CidrIpv6`

An IPv6 CIDR range.

Type: String

Required: Conditional. You must specify a source security group (`SourceSecurityGroupName` or `SourceSecurityGroupId`) or a CIDR range (`CidrIp` or `CidrIpv6`).

Update requires: Replacement (p. 90)

`FromPort`

Start of port range for the TCP and UDP protocols, or an ICMP type number. If you specify `icmp` for the `IpProtocol` property, you can specify -1 as a wildcard (i.e., any ICMP type number).

Type: Integer

Required: Yes, for ICMP and any protocol that uses ports.

Update requires: Replacement (p. 90)

`GroupId`

ID of the Amazon EC2 or VPC security group to modify. The group must belong to your account.

Type: String
Required: Conditional. You must specify the `GroupName` property or the `GroupId` property. For security groups that are in a VPC, you must use the `GroupId` property. For example, EC2-VPC accounts must use the `GroupId` property.

Update requires: Replacement (p. 90)

`GroupName`

Name of the Amazon EC2 security group (non-VPC security group) to modify. This value can be a reference to an AWS::EC2::SecurityGroup (p. 608) resource or the name of an existing Amazon EC2 security group.

Type: String

Required: Conditional. You must specify the `GroupName` property or the `GroupId` property. For security groups that are in a VPC, you must use the `GroupId` property. For example, EC2-VPC accounts must use the `GroupId` property.

Update requires: Replacement (p. 90)

`IpProtocol`
IP protocol name or number. For valid values, see the IpProtocol parameter in AuthorizeSecurityGroupIngress

Type: String

Required: Yes

Update requires: Replacement (p. 90)

`SourceSecurityGroupId`

Specifies the ID of the source security group or uses the `Ref` intrinsic function to refer to the logical ID of a security group defined in the same template.

Type: String

Required: Conditional. You must specify a source security group (`SourceSecurityGroupName` or `SourceSecurityGroupId`) or a CIDR range (`CidrIp` or `CidrIpv6`).

Update requires: Replacement (p. 90)

`SourceSecurityGroupName`

Specifies the name of the Amazon EC2 security group (non-VPC security group) to allow access or use the `Ref` intrinsic function to refer to the logical ID of a security group defined in the same template. For instances in a VPC, specify the `SourceSecurityGroupId` property.

Type: String

Required: Conditional. You must specify a source security group (`SourceSecurityGroupName` or `SourceSecurityGroupId`) or a CIDR range (`CidrIp` or `CidrIpv6`).

Update requires: Replacement (p. 90)

`SourceSecurityGroupOwnerId`
Specifies the AWS Account ID of the owner of the Amazon EC2 security group specified in the `SourceSecurityGroupName` property.

Type: String

Required: Conditional. If you specify `SourceSecurityGroupName` and that security group is owned by a different account than the account creating the stack, you must specify the `SourceSecurityGroupOwnerId`; otherwise, this property is optional.

Update requires: Replacement (p. 90)

`ToPort`
End of port range for the TCP and UDP protocols, or an ICMP code. If you specify `icmp` for the `IpProtocol` property, you can specify -1 as a wildcard (i.e., any ICMP code).

Type: Integer

Required: Yes, for ICMP and any protocol that uses ports.

Update requires: Replacement (p. 90)

Examples

EC2 Security Group and Ingress Rule

To create an Amazon EC2 (non-VPC) security group and an ingress rule, use the `SourceSecurityGroupName` property in the ingress rule.

The following template snippet creates an EC2 security group with an ingress rule that allows incoming traffic on port 80 from any other host in the security group. The snippet uses the intrinsic function `Ref` (p. 1343) to specify the value for SourceSecurityGroupName.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Resources": {
        "SGBase": {
            "Type": "AWS::EC2::SecurityGroup",
            "Properties": {
                "GroupDescription": "Base Security Group",
                "SecurityGroupIngress": [
                    {
                        "IpProtocol": "tcp",
                        "CidrIp": "0.0.0.0/0",
                        "FromPort": "22",
                        "ToPort": "22"
                    }
                ]
            }
        },
        "SGBaseIngress": {
            "Type": "AWS::EC2::SecurityGroupIngress",
            "Properties": {
                "GroupId": { "Ref": "SGBase" },
                "IpProtocol": "tcp",
                "FromPort": "80",
                "ToPort": "80",
                "SourceSecurityGroupId": { "Ref": "SGBase" }
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  SGBase:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Base Security Group
      SecurityGroupIngress:
      - IpProtocol: tcp
        CidrIp: 0.0.0.0/0
        FromPort: '22'
        ToPort: '22'
  SGBaseIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
```

```
      GroupId:
        Ref: SGBase
      IpProtocol: tcp
      FromPort: '80'
      ToPort: '80'
      SourceSecurityGroupId:
        Ref: SGBase
```

VPC Security Groups with Egress and Ingress Rules

In some cases, you might have an originating (source) security group to which you want to add an outbound rule that allows traffic to a destination (target) security group. The target security group also needs an inbound rule that allows traffic from the source security group. Note that you cannot use the `Ref` function to specify the outbound and inbound rules for each security group. Doing so creates a circular dependency; you cannot have two resources that depend on each other. Instead, use the egress and ingress resources to declare these outbound and inbound rules, as shown in the following template snippet.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Resources": {
    "SourceSG": {
      "Type": "AWS::EC2::SecurityGroup",
      "Properties": {
        "VpcId" : "vpc-e063f789",
        "GroupDescription": "Sample source security group"
      }
    },
    "TargetSG": {
      "Type": "AWS::EC2::SecurityGroup",
      "Properties": {
        "VpcId" : "vpc-e063f789",
        "GroupDescription": "Sample target security group"
      }
    },
    "OutboundRule": {
      "Type": "AWS::EC2::SecurityGroupEgress",
      "Properties":{
        "IpProtocol": "tcp",
        "FromPort": "0",
        "ToPort": "65535",
        "DestinationSecurityGroupId": {
          "Fn::GetAtt": [
            "TargetSG",
            "GroupId"
          ]
        },
        "GroupId": {
          "Fn::GetAtt": [
            "SourceSG",
            "GroupId"
          ]
        }
      }
    },
    "InboundRule": {
      "Type": "AWS::EC2::SecurityGroupIngress",
      "Properties":{
        "IpProtocol": "tcp",
        "FromPort": "0",
        "ToPort": "65535",
        "SourceSecurityGroupId": {
```

```
          "Fn::GetAtt": [
            "SourceSG",
            "GroupId"
          ]
        },
        "GroupId": {
          "Fn::GetAtt": [
            "TargetSG",
            "GroupId"
          ]
        }
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  SourceSG:
    Type: AWS::EC2::SecurityGroup
    Properties:
      VpcId: vpc-e063f789
      GroupDescription: Sample source security group
  TargetSG:
    Type: AWS::EC2::SecurityGroup
    Properties:
      VpcId: vpc-e063f789
      GroupDescription: Sample target security group
  OutboundRule:
    Type: AWS::EC2::SecurityGroupEgress
    Properties:
      IpProtocol: tcp
      FromPort: '0'
      ToPort: '65535'
      DestinationSecurityGroupId:
        Fn::GetAtt:
        - TargetSG
        - GroupId
      GroupId:
        Fn::GetAtt:
        - SourceSG
        - GroupId
  InboundRule:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      IpProtocol: tcp
      FromPort: '0'
      ToPort: '65535'
      SourceSecurityGroupId:
        Fn::GetAtt:
        - SourceSG
        - GroupId
      GroupId:
        Fn::GetAtt:
        - TargetSG
        - GroupId
```

Allow Ping Requests

To allow ping requests, add the ICMP protocol type and specify `8` (echo request) for the ICMP type and either `0` or `-1` (all) for the ICMP code.

JSON

```
"SGPing" : {
  "Type" : "AWS::EC2::SecurityGroup",
  "DependsOn": "VPC",
  "Properties" : {
    "GroupDescription" : "SG to test ping",
    "VpcId" : {"Ref" : "VPC"},
    "SecurityGroupIngress" : [
      { "IpProtocol" : "tcp", "FromPort" : "22", "ToPort" : "22", "CidrIp" :
 "10.0.0.0/24" },
      { "IpProtocol" : "icmp", "FromPort" : "8", "ToPort" : "-1", "CidrIp" :
 "10.0.0.0/24" }
    ]
  }
}
```

YAML

```
SGPing:
  Type: AWS::EC2::SecurityGroup
  DependsOn: VPC
  Properties:
    GroupDescription: SG to test ping
    VpcId:
      Ref: VPC
    SecurityGroupIngress:
    - IpProtocol: tcp
      FromPort: '22'
      ToPort: '22'
      CidrIp: 10.0.0.0/24
    - IpProtocol: icmp
      FromPort: '8'
      ToPort: "-1"
      CidrIp: 10.0.0.0/24
```

AWS::EC2::SpotFleet

The `AWS::EC2::SpotFleet` resource creates a request for a collection of Spot instances. The Spot fleet attempts to launch the number of Spot instances to meet the target capacity that you specified. For more information, see Spot Instances in the *Amazon EC2 User Guide for Linux Instances*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
```

```
  "Type" : "AWS::EC2::SpotFleet",
  "Properties" : {
    "SpotFleetRequestConfigData" : SpotFleetRequestConfigData
  }
}
```

YAML

```
Type: "AWS::EC2::SpotFleet"
Properties:
  SpotFleetRequestConfigData:
    SpotFleetRequestConfigData
```

Properties

`SpotFleetRequestConfigData`

The configuration for a Spot fleet request.

Required: Yes

Type: Amazon EC2 SpotFleet SpotFleetRequestConfigData (p. 1118)

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a Spot fleet with two launch specifications. The weighted capacities are the same, so Amazon EC2 launches the same number of instances for each specification. For more information, see How Spot Fleet Works in the *Amazon EC2 User Guide for Linux Instances*.

JSON

```
"SpotFleet": {
  "Type": "AWS::EC2::SpotFleet",
  "Properties": {
    "SpotFleetRequestConfigData": {
      "IamFleetRole": { "Fn::GetAtt": [ "IAMFleetRole", "Arn"] },
      "SpotPrice": "1000",
      "TargetCapacity": { "Ref": "TargetCapacity" },
      "LaunchSpecifications": [
      {
        "EbsOptimized": "false",
        "InstanceType": { "Ref": "InstanceType" },
        "ImageId": { "Fn::FindInMap": [ "AWSRegionArch2AMI", { "Ref": "AWS::Region" },
                     { "Fn::FindInMap": [ "AWSInstanceType2Arch", { "Ref":
 "InstanceType" }, "Arch" ] }
```

```
                    ]},
        "SubnetId": { "Ref": "Subnet1" },
        "WeightedCapacity": "8"
      },
      {
        "EbsOptimized": "true",
        "InstanceType": { "Ref": "InstanceType" },
        "ImageId": { "Fn::FindInMap": [ "AWSRegionArch2AMI", { "Ref": "AWS::Region" },
                     { "Fn::FindInMap": [ "AWSInstanceType2Arch", { "Ref":
 "InstanceType" }, "Arch" ] }
                    ]},
        "Monitoring": { "Enabled": "true" },
        "SecurityGroups": [ { "GroupId": { "Fn::GetAtt": [ "SG0", "GroupId" ] } } ],
        "SubnetId": { "Ref": "Subnet0" },
        "IamInstanceProfile": { "Arn": { "Fn::GetAtt": [ "RootInstanceProfile",
 "Arn" ] } },
        "WeightedCapacity": "8"
      }
      ]
    }
  }
}
```

YAML

```
SpotFleet:
  Type: AWS::EC2::SpotFleet
  Properties:
    SpotFleetRequestConfigData:
      IamFleetRole: !GetAtt [IAMFleetRole, Arn]
      SpotPrice: '1000'
      TargetCapacity:
        Ref: TargetCapacity
      LaunchSpecifications:
      - EbsOptimized: 'false'
        InstanceType:
          Ref: InstanceType
        ImageId:
          Fn::FindInMap:
          - AWSRegionArch2AMI
          - Ref: AWS::Region
          - Fn::FindInMap:
            - AWSInstanceType2Arch
            - Ref: InstanceType
            - Arch
        SubnetId:
          Ref: Subnet1
        WeightedCapacity: '8'
      - EbsOptimized: 'true'
        InstanceType:
          Ref: InstanceType
        ImageId:
          Fn::FindInMap:
          - AWSRegionArch2AMI
          - Ref: AWS::Region
          - Fn::FindInMap:
            - AWSInstanceType2Arch
            - Ref: InstanceType
            - Arch
        Monitoring:
          Enabled: 'true'
        SecurityGroups:
        - GroupId:
            Fn::GetAtt:
```

```
          - SG0
          - GroupId
        SubnetId:
          Ref: Subnet0
        IamInstanceProfile:
          Arn:
            Fn::GetAtt:
            - RootInstanceProfile
            - Arn
        WeightedCapacity: '8'
```

Related Resources

To use Application Auto Scaling to scale an Amazon ECS service in response to CloudWatch alarms, use the AWS::ApplicationAutoScaling::ScalableTarget (p. 427) and AWS::ApplicationAutoScaling::ScalingPolicy (p. 430) resources.

AWS::EC2::Subnet

Creates a subnet in an existing VPC.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::EC2::Subnet",
  "Properties" : {
    "AvailabilityZone (p. 626)" : String,
    "CidrBlock (p. 626)" : String,
    "MapPublicIpOnLaunch" : Boolean,
    "Tags (p. 626)" : [ Resource Tag, ... ],
    "VpcId (p. 626)" : String
  }
}
```

YAML

```
Type: "AWS::EC2::Subnet"
Properties:
  AvailabilityZone (p. 626): String
  CidrBlock (p. 626): String
  MapPublicIpOnLaunch: Boolean
  Tags (p. 626):
    Resource Tag
```

```
  VpcId (p. 626): String
```

Properties

`AvailabilityZone`

The availability zone in which you want the subnet. Default: AWS selects a zone for you (recommended).

Required: No

Type: String

Update requires: Replacement (p. 90)

Note
If you update this property, you must also update the `CidrBlock` property.

`CidrBlock`

The CIDR block that you want the subnet to cover (for example, `"10.0.0.0/24"`).

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Note
If you update this property, you must also update the `AvailabilityZone` property.

`MapPublicIpOnLaunch`

Indicates whether instances that are launched in this subnet receive a public IP address. By default, the value is `false`.

Required: No

Type: Boolean

Update requires: No interruption (p. 90).

`Tags`

An arbitrary set of tags (key–value pairs) for this subnet.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90).

`VpcId`

A Ref structure that contains the ID of the VPC on which you want to create the subnet. The VPC ID is provided as the value of the "Ref" property, as: `{ "Ref": "VPCID" }`.

Required: Yes

Type: Ref ID

Update requires: Replacement (p. 90)

Note
If you update this property, you must also update the `CidrBlock` property.

Return Values

You can pass the logical ID of the resource to an intrinsic function to get a value back from the resource. The value that is returned depends on the function that you used.

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource ID, such as `subnet-e19f0178`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`AvailabilityZone`

Returns the availability zone (for example, `"us-east-1a"`) of this subnet.

Example:

```
{ "Fn::GetAtt" : [ "mySubnet", "AvailabilityZone" ] }
```

`Ipv6CidrBlocks`

A list of IPv6 CIDR blocks that are associated with the subnet, such as `[ 2001:db8:1234:1a00::/64 ]`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Example

The following example snippet uses the VPC ID from a VPC named *myVPC* that was declared elsewhere in the same template.

JSON

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "mySubnet" : {
         "Type" : "AWS::EC2::Subnet",
         "Properties" : {
            "VpcId" : { "Ref" : "myVPC" },
            "CidrBlock" : "10.0.0.0/24",
            "AvailabilityZone" : "us-east-1a",
            "Tags" : [ { "Key" : "foo", "Value" : "bar" } ]
         }
      }
```

```
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  mySubnet:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId:
        Ref: myVPC
      CidrBlock: 10.0.0.0/24
      AvailabilityZone: "us-east-1a"
      Tags:
      - Key: foo
        Value: bar
```

More Info

- CreateSubnet in the *Amazon EC2 API Reference*
- Using Tags in the *Amazon Elastic Compute Cloud User Guide*

AWS::EC2::SubnetCidrBlock

The `AWS::EC2::SubnetCidrBlock` resource associates a single IPv6 CIDR block with an Amazon VPC subnet.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::SubnetCidrBlock",
   "Properties" : {
      "Ipv6CidrBlock" : String,
      "SubnetId" : String
   }
}
```

YAML

```
Type: "AWS::EC2::SubnetCidrBlock"
Properties:
  Ipv6CidrBlock: String
  SubnetId: String
```

Properties

`Ipv6CidrBlock`

The IPv6 CIDR block for the subnet. The CIDR block must have a prefix length of /64.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`SubnetId`

The ID of the subnet to associate the IPv6 CIDR block with.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Example

The following example associates an IPv6 CIDR block (with a prefix length of /64) with the `Ipv6TestSubnet` subnet.

JSON

```
{
  "Ipv6TestSubnetCidrBlock": {
    "Type": "AWS::EC2::SubnetCidrBlock",
    "Properties": {
      "Ipv6CidrBlock": { "Ref" : "Ipv6SubnetCidrBlock" },
      "SubnetId": { "Ref" : "Ipv6TestSubnet" }
    }
  }
}
```

YAML

```
Ipv6TestSubnetCidrBlock:
  Type: "AWS::EC2::SubnetCidrBlock"
  Properties:
    Ipv6CidrBlock: !Ref Ipv6SubnetCidrBlock
    SubnetId: !Ref Ipv6TestSubnet
```

AWS::EC2::SubnetNetworkAclAssociation

Associates a subnet with a network ACL.

For more information, go to ReplaceNetworkAclAssociation in the *Amazon EC2 API Reference*.

> **Note**
> The EC2 API Reference refers to the `SubnetId` parameter as the `AssociationId`.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
"Type" : "AWS::EC2::SubnetNetworkAclAssociation",
"Properties" : {
   "SubnetId (p. 630)" : String,
   "NetworkAclId (p. 630)" : String
}
```

YAML

```
Type: "AWS::EC2::SubnetNetworkAclAssociation"
Properties:
  SubnetId (p. 630): String
  NetworkAclId (p. 630): String
```

Properties

`SubnetId`

The ID representing the current association between the original network ACL and the subnet.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`NetworkAclId`

The ID of the new ACL to associate with the subnet.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`AssociationId`

Returns the value of this object's SubnetId (p. 630) property.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Template Examples

JSON

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "mySubnetNetworkAclAssociation" : {
         "Type" : "AWS::EC2::SubnetNetworkAclAssociation",
         "Properties" : {
            "SubnetId" : { "Ref" : "mySubnet" },
            "NetworkAclId" : { "Ref" : "myNetworkAcl" }
         }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  mySubnetNetworkAclAssociation:
    Type: AWS::EC2::SubnetNetworkAclAssociation
    Properties:
      SubnetId:
        Ref: mySubnet
      NetworkAclId:
        Ref: myNetworkAcl
```

AWS::EC2::SubnetRouteTableAssociation

Associates a subnet with a route table.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::SubnetRouteTableAssociation",
   "Properties" : {
      "RouteTableId (p. 632)" : String,
      "SubnetId (p. 632)" : String
   }
}
```

YAML

```
Type: "AWS::EC2::SubnetRouteTableAssociation"
Properties:
  RouteTableId (p. 632): String
  SubnetId (p. 632): String
```

Properties

`RouteTableId`

The ID of the route table. This is commonly written as a reference to a route table declared elsewhere in the template. For example:

```
"RouteTableId" : { "Ref" : "myRouteTable" }
```

Required: Yes

Type: String

Update requires: No interruption (p. 90). However, the physical ID changes when the route table ID is changed.

`SubnetId`

The ID of the subnet. This is commonly written as a reference to a subnet declared elsewhere in the template. For example:

```
"SubnetId" : { "Ref" : "mySubnet" }
```

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "MyRTA" }
```

For the subnet route table association with the logical ID "MyRTA", Ref will return the AWS resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

JSON

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "mySubnetRouteTableAssociation" : {
         "Type" : "AWS::EC2::SubnetRouteTableAssociation",
         "Properties" : {
            "SubnetId" : { "Ref" : "mySubnet" },
            "RouteTableId" : { "Ref" : "myRouteTable" }
         }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  mySubnetRouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      SubnetId:
        Ref: mySubnet
      RouteTableId:
        Ref: myRouteTable
```

See Also

- AssociateRouteTable in the *Amazon EC2 API Reference*

AWS::EC2::Volume

The `AWS::EC2::Volume` type creates a new Amazon Elastic Block Store (Amazon EBS) volume.

To control how AWS CloudFormation handles the volume when the stack is deleted, set a deletion policy for your volume. You can choose to *retain* the volume, to *delete* the volume, or to *create a snapshot* of the volume. For more information, see DeletionPolicy Attribute (p. 1297).

> **Note**
> If you set a deletion policy that creates a snapshot, all tags on the volume are included in the snapshot.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type":"AWS::EC2::Volume",
   "Properties" : {
      "AutoEnableIO" : Boolean,
      "AvailabilityZone (p. 634)" : String,
      "Encrypted" : Boolean,
      "Iops (p. 635)" : Number,
      "KmsKeyId" : String,
      "Size (p. 635)" : Integer,
      "SnapshotId (p. 635)" : String,
      "Tags (p. 636)" : [ Resource Tag, ... ],
      "VolumeType (p. 636)" : String
   }
}
```

YAML

```
Type: "AWS::EC2::Volume"
Properties:
  AutoEnableIO: Boolean
  AvailabilityZone (p. 634): String
  Encrypted: Boolean
  Iops (p. 635): Number
  KmsKeyId: String
  Size (p. 635): Integer
  SnapshotId (p. 635): String
  Tags (p. 636):
    - Resource Tag
  VolumeType (p. 636): String
```

Properties

`AutoEnableIO`

Indicates whether the volume is auto-enabled for I/O operations. By default, Amazon EBS disables I/O to the volume from attached EC2 instances when it determines that a volume's data is potentially inconsistent. If the consistency of the volume is not a concern, and you prefer that the volume be made available immediately if it's impaired, you can configure the volume to automatically enable I/O. For more information, see Working with the AutoEnableIO Volume Attribute in the *Amazon EC2 User Guide for Linux Instances*.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`AvailabilityZone`

The Availability Zone in which to create the new volume.

Required: Yes

Type: String

Update requires: Updates are not supported.

`Encrypted`

Indicates whether the volume is encrypted. You can attach encrypted Amazon EBS volumes only to instance types that support Amazon EBS encryption. Volumes that are created from encrypted snapshots are automatically encrypted. You can't create an encrypted volume from an unencrypted snapshot, or vice versa. If your AMI uses encrypted volumes, you can launch the AMI only on supported instance types. For more information, see Amazon EBS encryption in the *Amazon EC2 User Guide for Linux Instances*.

Required: Conditional. If you specify the `KmsKeyId` property, you must enable encryption.

Type: Boolean

Update requires: Updates are not supported.

`Iops`

The number of I/O operations per second (IOPS) that the volume supports. For more information about the valid sizes for each volume type, see the `Iops` parameter for the `CreateVolume` action in the *Amazon EC2 API Reference*.

Required: Conditional. *Required* when the volume type is `io1`; not used with other volume types.

Type: Number

Update requires: Updates are not supported.

`KmsKeyId`

The Amazon Resource Name (ARN) of the AWS Key Management Service master key that is used to create the encrypted volume, such as `arn:aws:kms:us-east-1:012345678910:key/abcd1234-a123-456a-a12b-a123b4cd56ef`. If you create an encrypted volume and don't specify this property, AWS CloudFormation uses the default master key.

Required: No

Type: String

Update requires: Updates are not supported.

`Size`

The size of the volume, in gibibytes (GiBs). For more information about the valid sizes for each volume type, see the `Size` parameter for the `CreateVolume` action in the *Amazon EC2 API Reference*.

If you specify the `SnapshotId` property, specify a size that is equal to or greater than the size of the snapshot. If you don't specify a size, Amazon EC2 uses the size of the snapshot as the volume size.

Required: Conditional. If you don't specify a value for the `SnapshotId` property, you must specify this property.

Type: Integer

Update requires: Updates are not supported.

`SnapshotId`

The snapshot from which to create the new volume.

Required: No

Type: String

Update requires: Updates are not supported.

`Tags`

An arbitrary set of tags (key–value pairs) for this volume.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90)

`VolumeType`

The volume type. If you set the type to `io1`, you must also set the `Iops` property. For valid values, see the `VolumeType` parameter for the CreateVolume action in the *Amazon EC2 API Reference*.

Required: No

Type: String

Update requires: Updates are not supported.

Return Values

Ref

When you specify an `AWS::EC2::Volume` type as an argument to the `Ref` function, AWS CloudFormation returns the volume's physical ID. For example: `vol-5cb85026`.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

Example Encrypted Amazon EBS Volume with DeletionPolicy to Make a Snapshot on Delete

```
"NewVolume" : {
   "Type" : "AWS::EC2::Volume",
   "Properties" : {
      "Size" : "100",
      "Encrypted" : "true",
      "AvailabilityZone" : { "Fn::GetAtt" : [ "Ec2Instance", "AvailabilityZone" ] },
      "Tags" : [ {
         "Key" : "MyTag",
         "Value" : "TagValue"
      } ]
   },
   "DeletionPolicy" : "Snapshot"
}
```

Example Amazon EBS Volume with 100 Provisioned IOPS

```
"NewVolume" : {
   "Type" : "AWS::EC2::Volume",
   "Properties" : {
     "Size" : "100",
     "VolumeType" : "io1",
     "Iops" : "100",
     "AvailabilityZone" : { "Fn::GetAtt" : [ "EC2Instance", "AvailabilityZone" ] }
   }
}
```

More Info

- CreateVolume in the *Amazon Elastic Compute Cloud API Reference*
- DeletionPolicy Attribute (p. 1297)

AWS::EC2::VolumeAttachment

Attaches an Amazon EBS volume to a running instance and exposes it to the instance with the specified device name.

> **Important**
> Before this resource can be deleted (and therefore the volume detached), you must first unmount the volume in the instance. Failure to do so results in the volume being stuck in the busy state while it is trying to detach, which could possibly damage the file system or the data it contains.
> If an Amazon EBS volume is the root device of an instance, it cannot be detached while the instance is in the "running" state. To detach the root volume, stop the instance first.
> If the root volume is detached from an instance with an AWS Marketplace product code, then the AWS Marketplace product codes from that volume are no longer associated with the instance.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type":"AWS::EC2::VolumeAttachment",
   "Properties" : {
      "Device (p. 638)" : String,
      "InstanceId (p. 638)" : String,
      "VolumeId (p. 638)" : String
   }
}
```

YAML

```
Type: AWS::EC2::VolumeAttachment
Properties:
  Device (p. 638): String
  InstanceId (p. 638): String
  VolumeId (p. 638): String
```

Properties

`Device`

How the device is exposed to the instance (e.g., /dev/sdh, or xvdh).

Required: Yes

Type: String

Update requires: Updates are not supported.

`InstanceId`

The ID of the instance to which the volume attaches. This value can be a reference to an AWS::EC2::Instance (p. 574) resource, or it can be the physical ID of an existing EC2 instance.

Required: Yes

Type: String

Update requires: Updates are not supported.

`VolumeId`

The ID of the Amazon EBS volume. The volume and instance must be within the same Availability Zone. This value can be a reference to an AWS::EC2::Volume (p. 633) resource, or it can be the volume ID of an existing Amazon EBS volume.

Required: Yes

Type: String

Update requires: Updates are not supported.

Example

This example attaches an EC2 EBS volume to the EC2 instance with the logical name "Ec2Instance".

```
"NewVolume" : {
   "Type" : "AWS::EC2::Volume",
   "Properties" : {
      "Size" : "100",
      "AvailabilityZone" : { "Fn::GetAtt" : [ "Ec2Instance", "AvailabilityZone" ] },
      "Tags" : [ {
         "Key" : "MyTag",
         "Value" : "TagValue"
      } ]
   }
},

"MountPoint" : {
   "Type" : "AWS::EC2::VolumeAttachment",
```

```
   "Properties" : {
      "InstanceId" : { "Ref" : "Ec2Instance" },
      "VolumeId"  : { "Ref" : "NewVolume" },
      "Device" : "/dev/sdh"
   }
}
```

See Also

- Amazon Elastic Block Store (Amazon EBS) in the *Amazon Elastic Compute Cloud User Guide*.
- Attaching a Volume to an Instance in the *Amazon Elastic Compute Cloud User Guide*
- Detaching an Amazon EBS Volume from an Instance in the *Amazon Elastic Compute Cloud User Guide*
- AttachVolume in the *Amazon Elastic Compute Cloud API Reference*
- DetachVolume in the *Amazon Elastic Compute Cloud API Reference*

AWS::EC2::VPC

Creates a Virtual Private Cloud (VPC) with the CIDR block that you specify. To name a VPC resource, use the `Tags` property and specify a value for the `Name` key.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::VPC",
   "Properties" : {
      "CidrBlock (p. 640)" : String,
      "EnableDnsSupport (p. 640)" : Boolean,
      "EnableDnsHostnames (p. 640)" : Boolean,
      "InstanceTenancy (p. 640)" : String,
      "Tags (p. 640)" : [ Resource Tag, ... ]
   }
}
```

YAML

```
Type: "AWS::EC2::VPC"
Properties:
  CidrBlock (p. 640): String
  EnableDnsSupport (p. 640): Boolean
  EnableDnsHostnames (p. 640): Boolean
  InstanceTenancy (p. 640): String
```

```
  Tags (p. 640):
    - Resource Tag
```

Properties

`CidrBlock`

The CIDR block you want the VPC to cover. For example: `"10.0.0.0/16"`.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`EnableDnsSupport`

Specifies whether DNS resolution is supported for the VPC. If this attribute is `true`, the Amazon DNS server resolves DNS hostnames for your instances to their corresponding IP addresses; otherwise, it does not. By default the value is set to `true`.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`EnableDnsHostnames`

Specifies whether the instances launched in the VPC get DNS hostnames. If this attribute is `true`, instances in the VPC get DNS hostnames; otherwise, they do not. You can only set `EnableDnsHostnames` to `true` if you also set the `EnableDnsSupport` attribute to `true`. By default, the value is set to `false`.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`InstanceTenancy`

The allowed tenancy of instances launched into the VPC.

- `"default"`: Instances can be launched with any tenancy.
- `"dedicated"`: Any instance launched into the VPC automatically has dedicated tenancy, unless you launch it with the default tenancy.

Required: No

Type: String

Valid values: `"default"` or `"dedicated"`

Update requires: Replacement (p. 90)

`Tags`

An arbitrary set of tags (key–value pairs) for this VPC. To name a VPC resource, specify a value for the `Name` key.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90).

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource ID, such as `vpc-18ac277d`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`CidrBlock`

The set of IP addresses for the VPC. For example, `10.0.0.0/16`.

`DefaultNetworkAcl`

The default network ACL ID that is associated with the VPC. For example, `acl-814dafe3`.

`DefaultSecurityGroup`

The default security group ID that is associated with the VPC. For example, `sg-b178e0d3`.

`Ipv6CidrBlocks`

A list of IPv6 CIDR blocks that are associated with the VPC, such as `[ 2001:db8:1234:1a00::/56 ]`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Example

JSON

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "myVPC" : {
         "Type" : "AWS::EC2::VPC",
         "Properties" : {
            "CidrBlock" : "10.0.0.0/16",
         "EnableDnsSupport" : "false",
         "EnableDnsHostnames" : "false",
            "InstanceTenancy" : "dedicated",
            "Tags" : [ {"Key" : "foo", "Value" : "bar"} ]
         }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  myVPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: 10.0.0.0/16
```

```
      EnableDnsSupport: 'false'
      EnableDnsHostnames: 'false'
      InstanceTenancy: dedicated
      Tags:
      - Key: foo
        Value: bar
```

More Info

- CreateVpc in the *Amazon EC2 API Reference*.

AWS::EC2::VPCCidrBlock

The `AWS::EC2::VPCCidrBlock` resource associates a single Amazon-provided IPv6 CIDR block with an Amazon VPC VPC.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::VPCCidrBlock",
   "Properties" : {
      "AmazonProvidedIpv6CidrBlock" : Boolean,
      "VpcId" : String
   }
}
```

YAML

```
Type: "AWS::EC2::VPCCidrBlock"
Properties:
  AmazonProvidedIpv6CidrBlock: Boolean
  VpcId: String
```

Properties

`AmazonProvidedIpv6CidrBlock`

Whether to request an Amazon-provided IPv6 CIDR block with a /56 prefix length for the VPC. You can't specify the range of IPv6 addresses or the size of the CIDR block.

Required: No

Type: Boolean

Update requires: Replacement (p. 90)

`VpcId`

The ID of the VPC to associate the Amazon-provided IPv6 CIDR block with.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Example

The following example associates an Amazon-provided IPv6 CIDR block (with a prefix length of /56) with the `TestVPCIpv6` VPC.

JSON

```
{
  "Ipv6VPCCidrBlock": {
    "Type": "AWS::EC2::VPCCidrBlock",
    "Properties": {
      "AmazonProvidedIpv6CidrBlock": true,
      "VpcId": { "Ref" : "TestVPCIpv6" }
    }
  }
}
```

YAML

```
Ipv6VPCCidrBlock:
  Type: "AWS::EC2::VPCCidrBlock"
  Properties:
    AmazonProvidedIpv6CidrBlock: true
    VpcId: !Ref TestVPCIpv6
```

AWS::EC2::VPCDHCPOptionsAssociation

Associates a set of DHCP options (that you've previously created) with the specified VPC.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::VPCDHCPOptionsAssociation",
   "Properties" : {
      "DhcpOptionsId (p. 644)" : String,
      "VpcId (p. 644)" : String
   }
}
```

YAML

```
Type: "AWS::EC2::VPCDHCPOptionsAssociation"
Properties:
  DhcpOptionsId (p. 644): String
  VpcId (p. 644): String
```

Properties

`DhcpOptionsId`

The ID of the DHCP options you want to associate with the VPC. Specify `default` if you want the VPC to use no DHCP options.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`VpcId`

The ID of the VPC to associate with this DHCP options set.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following snippet uses the `Ref` intrinsic function to associate the `myDHCPOptions` DHCP options with the `myVPC` VPC. The VPC and DHCP options can be declared in the same template or added as input parameters. For more information about the VPC or the DHCP options resources, see AWS::EC2::VPC (p. 639) or AWS::EC2::DHCPOptions (p. 560).

JSON

```
"myVPCDHCPOptionsAssociation" : {
```

```
  "Type" : "AWS::EC2::VPCDHCPOptionsAssociation",
  "Properties" : {
    "VpcId" : {"Ref" : "myVPC"},
    "DhcpOptionsId" : {"Ref" : "myDHCPOptions"}
  }
}
```

YAML

```
myVPCDHCPOptionsAssociation:
  Type: AWS::EC2::VPCDHCPOptionsAssociation
  Properties:
    VpcId:
      Ref: myVPC
    DhcpOptionsId:
      Ref: myDHCPOptions
```

See Also

- AssociateDhcpOptions in the *Amazon EC2 API Reference*.

AWS::EC2::VPCEndpoint

The `AWS::EC2::VPCEndpoint` resource creates a VPC endpoint that you can use to establish a private connection between your VPC and another AWS service without requiring access over the Internet, a VPN connection, or AWS Direct Connect.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::EC2::VPCEndpoint",
  "Properties" : {
    "PolicyDocument" : JSON object,
    "RouteTableIds" : [ String, ... ],
    "ServiceName" : String,
    "VpcId" : String
  }
}
```

YAML

```
Type: "AWS::EC2::VPCEndpoint"
```

```
Properties:
  PolicyDocument: JSON object
  RouteTableIds:
    - String
  ServiceName: String
  VpcId: String
```

Properties

`PolicyDocument`

A policy to attach to the endpoint that controls access to the service. The policy must be valid JSON. The default policy allows full access to the AWS service. For more information, see Controlling Access to Services in the *Amazon VPC User Guide*.

Required: No

Type: JSON object

Update requires: No interruption (p. 90)

`RouteTableIds`

One or more route table IDs that are used by the VPC to reach the endpoint.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`ServiceName`

The AWS service to which you want to establish a connection. Specify the service name in the form of `com.amazonaws.region.service`.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`VpcId`

The ID of the VPC in which the endpoint is used.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Value

Ref

When you pass the logical ID of an `AWS::EC2::VPCEndpoint` resource to the intrinsic `Ref` function, the function returns the endpoint ID, such as `vpce-a123d0d1`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a VPC endpoint that allows only the `s3:GetObject` action on the `examplebucket` bucket. Traffic to S3 within subnets that are associated with the `routetableA` and `routetableB` route tables is automatically routed through the VPC endpoint.

JSON

```
"S3Endpoint" : {
  "Type" : "AWS::EC2::VPCEndpoint",
  "Properties" : {
    "PolicyDocument" : {
      "Version":"2012-10-17",
      "Statement":[{
        "Effect":"Allow",
        "Principal": "*",
        "Action":["s3:GetObject"],
        "Resource":["arn:aws:s3:::examplebucket/*"]
      }]
    },
    "RouteTableIds" : [ {"Ref" : "routetableA"}, {"Ref" : "routetableB"} ],
    "ServiceName" : { "Fn::Join": [ "", [ "com.amazonaws.", { "Ref": "AWS::Region" },
 ".s3" ] ] },
    "VpcId" : {"Ref" : "VPCID"}
  }
}
```

YAML

```

S3Endpoint:
  Type: "AWS::EC2::VPCEndpoint"
  Properties:
    PolicyDocument: {
      "Version":"2012-10-17",
      "Statement":[{
        "Effect":"Allow",
        "Principal": "*",
        "Action":["s3:GetObject"],
        "Resource":["arn:aws:s3:::examplebucket/*"]
      }]
    }
    RouteTableIds:
      - !Ref routetableA
      - !Ref routetableB
    ServiceName: !Join ['', [com.amazonaws., !Ref 'AWS::Region', .s3]]
    VpcId: !Ref VPCID
```

AWS::EC2::VPCGatewayAttachment

Attaches a gateway to a VPC.

Topics

- See Also (p. 649)

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::VPCGatewayAttachment",
   "Properties" : {
      "InternetGatewayId (p. 648)" : String,
      "VpcId (p. 648)" : String,
      "VpnGatewayId (p. 648)" : String
   }
}
```

YAML

```
Type: "AWS::EC2::VPCGatewayAttachment"
Properties:
  InternetGatewayId (p. 648): String
  VpcId (p. 648): String
  VpnGatewayId (p. 648): String
```

Properties

`InternetGatewayId`

The ID of the Internet gateway.

Required: Conditional You must specify either `InternetGatewayId` or `VpnGatewayId`, but not both.

Type: String

Update requires: No interruption (p. 90)

`VpcId`

The ID of the VPC to associate with this gateway.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`VpnGatewayId`

The ID of the virtual private network (VPN) gateway to attach to the VPC.

Required: Conditional You must specify either `InternetGatewayId` or `VpnGatewayId`, but not both.

Type: String

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

To attach both an Internet gateway and a VPN gateway to a VPC, you must specify two separate `AWS::EC2::VPCGatewayAttachment` resources:

JSON

```
"AttachGateway" : {
   "Type" : "AWS::EC2::VPCGatewayAttachment",
   "Properties" : {
      "VpcId" : { "Ref" : "VPC" },
      "InternetGatewayId" : { "Ref" : "myInternetGateway" }
   }
},

"AttachVpnGateway" : {
   "Type" : "AWS::EC2::VPCGatewayAttachment",
   "Properties" : {
      "VpcId" : { "Ref" : "VPC" },
      "VpnGatewayId" : { "Ref" : "myVPNGateway" }
   }
}
```

YAML

```
AttachGateway:
  Type: AWS::EC2::VPCGatewayAttachment
  Properties:
    VpcId:
      Ref: VPC
    InternetGatewayId:
      Ref: myInternetGateway
AttachVpnGateway:
  Type: AWS::EC2::VPCGatewayAttachment
  Properties:
    VpcId:
      Ref: VPC
    VpnGatewayId:
      Ref: myVPNGateway
```

See Also

- AttachVpnGateway in the *Amazon EC2 API Reference*.

AWS::EC2::VPCPeeringConnection

A VPC peering connection enables a network connection between two virtual private clouds (VPCs) so that you can route traffic between them using a private IP address. For more information about VPC peering and its limitations, see VPC Peering Overview in the *Amazon VPC Peering Guide*.

Note

You can create a peering connection with another AWS account. For a detailed walkthrough, see Walkthrough: Peer with an Amazon VPC in Another AWS Account (p. 195).

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::VPCPeeringConnection",
   "Properties" : {
      "PeerVpcId" : String,
      "Tags" : [ Resource Tag, ... ],
      "VpcId" : String,
      "PeerOwnerId" : String,
      "PeerRoleArn" :  String
   }
}
```

YAML

```
Type: "AWS::EC2::VPCPeeringConnection"
Properties:
  PeerVpcId: String
  Tags:
    - Resource Tag
  VpcId: String
  PeerOwnerId: String
  PeerRoleArn:  String
```

Properties

`PeerVpcId`

The ID of the VPC with which you are creating the peering connection.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Tags`

An arbitrary set of tags (key–value pairs) for this resource.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90).

`VpcId`

The ID of the VPC that is requesting a peering connection.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`PeerOwnerId`

The AWS account ID of the owner of the VPC that you want to peer with.

Required: No

Type: String

Update requires: Replacement (p. 90)

`PeerRoleArn`

The Amazon Resource Name (ARN) of the VPC peer role for the peering connection in another AWS account.

Required: No

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

The following example template creates two VPCs to demonstrate how to configure a peering connection. For a VPC peering connection, you must create a VPC peering route for each VPC route table, as shown in the example by `PeeringRoute1` and `PeeringRoute2`. If you launch the template, you can connect to the `myInstance` instance using SSH, and then ping the `myPrivateInstance` instance although both instances are in separate VPCs.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Description": "Creates a VPC that and then creates a peering connection with an
 existing VPC that you specify.",
    "Parameters": {
        "EC2KeyPairName": {
            "Description": "Name of an existing EC2 KeyPair to enable SSH access to the
 instances",
```

```
            "Type": "AWS::EC2::KeyPair::KeyName",
            "ConstraintDescription" : "must be the name of an existing EC2 KeyPair."
        },
        "InstanceType": {
            "Description": "EC2 instance type",
            "Type": "String",
            "Default": "t1.micro",
            "AllowedValues": [
                "t1.micro",
                "m1.small",
                "m3.medium",
                "m3.large",
                "m3.xlarge",
                "m3.2xlarge",
                "c3.large",
                "c3.xlarge",
                "c3.2xlarge",
                "c3.4xlarge",
                "c3.8xlarge"
            ],
            "ConstraintDescription": "must be a valid EC2 instance type."
        },
        "myVPCIDCIDRRange": {
            "Description": "The IP address range for your new VPC.",
            "Type": "String",
            "MinLength": "9",
            "MaxLength": "18",
            "Default": "10.1.0.0/16",
            "AllowedPattern": "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\
\d{1,2})",
            "ConstraintDescription": "must be a valid IP CIDR range of the form x.x.x.x/x."
        },
        "myPrivateVPCIDCIDRRange": {
            "Description": "The IP address range for your new Private VPC.",
            "Type": "String",
            "MinLength": "9",
            "MaxLength": "18",
            "Default": "10.0.0.0/16",
            "AllowedPattern": "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\
\d{1,2})",
            "ConstraintDescription": "must be a valid IP CIDR range of the form x.x.x.x/x."
        },
        "EC2SubnetCIDRRange": {
            "Description": "The IP address range for a subnet in myPrivateVPC.",
            "Type": "String",
            "MinLength": "9",
            "MaxLength": "18",
            "Default": "10.0.0.0/24",
            "AllowedPattern": "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\
\d{1,2})",
            "ConstraintDescription": "must be a valid IP CIDR range of the form x.x.x.x/x."
        },
        "EC2PublicSubnetCIDRRange": {
            "Description": "The IP address range for a subnet in myVPC.",
            "Type": "String",
            "MinLength": "9",
            "MaxLength": "18",
            "Default": "10.1.0.0/24",
            "AllowedPattern": "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\
\d{1,2})",
            "ConstraintDescription": "must be a valid IP CIDR range of the form x.x.x.x/x."
        }
    },
    "Mappings": {
        "AWSRegionToAMI": {
            "us-east-1": {
```

```
                "64": "ami-fb8e9292"
            },
            "us-west-2": {
                "64": "ami-043a5034"
            },
            "us-west-1": {
                "64": "ami-7aba833f"
            },
            "eu-west-1": {
                "64": "ami-2918e35e"
            },
            "ap-southeast-1": {
                "64": "ami-b40d5ee6"
            },
            "ap-southeast-2": {
                "64": "ami-3b4bd301"
            },
            "ap-northeast-1": {
                "64": "ami-c9562fc8"
            },
            "sa-east-1": {
                "64": "ami-215dff3c"
            }
        }
    },
    "Resources": {
        "myPrivateVPC": {
            "Type": "AWS::EC2::VPC",
            "Properties": {
                "CidrBlock": {"Ref": "myPrivateVPCIDCIDRRange"},
                "EnableDnsSupport": false,
                "EnableDnsHostnames": false,
                "InstanceTenancy": "default"
            }
        },
        "myPrivateEC2Subnet" : {
            "Type" : "AWS::EC2::Subnet",
            "Properties" : {
                "VpcId" : { "Ref" : "myPrivateVPC" },
                "CidrBlock" : {"Ref": "EC2SubnetCIDRRange"}
            }
        },
        "RouteTable" : {
            "Type" : "AWS::EC2::RouteTable",
            "Properties" : {
                "VpcId" : {"Ref" : "myPrivateVPC"}
            }
        },
        "PeeringRoute1" : {
            "Type" : "AWS::EC2::Route",
            "Properties" : {
                "DestinationCidrBlock": "0.0.0.0/0",
                "RouteTableId" : { "Ref" : "RouteTable" },
                "VpcPeeringConnectionId" : { "Ref" : "myVPCPeeringConnection" }
            }
        },
        "SubnetRouteTableAssociation" : {
            "Type" : "AWS::EC2::SubnetRouteTableAssociation",
            "Properties" : {
                "SubnetId" : { "Ref" : "myPrivateEC2Subnet" },
                "RouteTableId" : { "Ref" : "RouteTable" }
            }
        },
        "myVPC": {
            "Type": "AWS::EC2::VPC",
            "Properties": {
```

```
                "CidrBlock": {"Ref": "myVPCIDCIDRRange"},
                "EnableDnsSupport": true,
                "EnableDnsHostnames": true,
                "InstanceTenancy": "default"
            }
        },
        "PublicSubnet": {
            "Type": "AWS::EC2::Subnet",
            "Properties": {
                "CidrBlock": {"Ref": "EC2PublicSubnetCIDRRange"},
                "VpcId": {
                    "Ref": "myVPC"
                }
            }
        },
        "myInternetGateway": {
            "Type": "AWS::EC2::InternetGateway"
        },
        "AttachGateway": {
            "Type": "AWS::EC2::VPCGatewayAttachment",
            "Properties": {
                "VpcId": {
                    "Ref": "myVPC"
                },
                "InternetGatewayId": {
                    "Ref": "myInternetGateway"
                }
            }
        },
        "PublicRouteTable": {
            "Type": "AWS::EC2::RouteTable",
            "Properties": {
                "VpcId": {
                    "Ref": "myVPC"
                }
            }
        },
        "PeeringRoute2" : {
            "Type" : "AWS::EC2::Route",
            "Properties" : {
                "DestinationCidrBlock": { "Ref" : "myPrivateVPCIDCIDRRange" },
                "RouteTableId" : { "Ref" : "PublicRouteTable" },
                "VpcPeeringConnectionId" : { "Ref" : "myVPCPeeringConnection" }
            }
        },
        "PublicRoute": {
            "Type": "AWS::EC2::Route",
            "DependsOn": "AttachGateway",
            "Properties": {
                "RouteTableId": {
                    "Ref": "PublicRouteTable"
                },
                "DestinationCidrBlock": "0.0.0.0/0",
                "GatewayId": {
                    "Ref": "myInternetGateway"
                }
            }
        },
        "PublicSubnetRouteTableAssociation": {
            "Type": "AWS::EC2::SubnetRouteTableAssociation",
            "Properties": {
                "SubnetId": {
                    "Ref": "PublicSubnet"
                },
                "RouteTableId": {
                    "Ref": "PublicRouteTable"
```

```
                }
            }
        },
        "myPrivateVPCEC2SecurityGroup" : {
            "Type" : "AWS::EC2::SecurityGroup",
            "Properties" : {
                "GroupDescription": "Private instance security group",
                "VpcId" : { "Ref" : "myPrivateVPC" },
                "SecurityGroupIngress" : [
                    {"IpProtocol" : "-1", "FromPort" : "0", "ToPort" : "65535", "CidrIp" :
"0.0.0.0/0"}
                ]
            }
        },
        "myVPCEC2SecurityGroup" : {
            "Type" : "AWS::EC2::SecurityGroup",
            "Properties" : {
                "GroupDescription": "Public instance security group",
                "VpcId" : { "Ref" : "myVPC" },
                "SecurityGroupIngress" : [
                    {"IpProtocol" : "tcp", "FromPort" : "80", "ToPort" : "80", "CidrIp" :
"0.0.0.0/0"},
                    {"IpProtocol" : "tcp", "FromPort" : "22", "ToPort" : "22", "CidrIp" :
"0.0.0.0/0"}
                ]
            }
        },
        "myPrivateInstance" : {
            "Type" : "AWS::EC2::Instance",
            "Properties" : {
                "SecurityGroupIds" : [{ "Ref" : "myPrivateVPCEC2SecurityGroup" }],
                "SubnetId" : { "Ref" : "myPrivateEC2Subnet" },
                "KeyName": {
                    "Ref": "EC2KeyPairName"
                },
                "ImageId": {
                    "Fn::FindInMap": [
                        "AWSRegionToAMI",
                        {"Ref": "AWS::Region"},
                        "64"
                    ]
                }
            }
        },
        "myInstance" : {
            "Type" : "AWS::EC2::Instance",
            "Properties" : {
                "NetworkInterfaces": [ {
                    "AssociatePublicIpAddress": "true",
                    "DeviceIndex": "0",
                    "GroupSet": [{ "Ref" : "myVPCEC2SecurityGroup" }],
                    "SubnetId": { "Ref" : "PublicSubnet" }
                } ],
                "KeyName": {
                    "Ref": "EC2KeyPairName"
                },
                "ImageId": {
                    "Fn::FindInMap": [
                        "AWSRegionToAMI",
                        {"Ref": "AWS::Region"},
                        "64"
                    ]
                }
            }
        },
        "myVPCPeeringConnection": {
```

```
            "Type": "AWS::EC2::VPCPeeringConnection",
            "Properties": {
                "VpcId": {"Ref": "myVPC"},
                "PeerVpcId": {"Ref": "myPrivateVPC"}
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: Creates a VPC that and then creates a peering connection with an existing
  VPC that you specify.
Parameters:
  EC2KeyPairName:
    Description: Name of an existing EC2 KeyPair to enable SSH access to the instances
    Type: AWS::EC2::KeyPair::KeyName
    ConstraintDescription: must be the name of an existing EC2 KeyPair.
  InstanceType:
    Description: EC2 instance type
    Type: String
    Default: t1.micro
    AllowedValues:
    - t1.micro
    - m1.small
    - m3.medium
    - m3.large
    - m3.xlarge
    - m3.2xlarge
    - c3.large
    - c3.xlarge
    - c3.2xlarge
    - c3.4xlarge
    - c3.8xlarge
    ConstraintDescription: must be a valid EC2 instance type.
  myVPCIDCIDRRange:
    Description: The IP address range for your new VPC.
    Type: String
    MinLength: '9'
    MaxLength: '18'
    Default: 10.1.0.0/16
    AllowedPattern: "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\\d{1,2})"
    ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.
  myPrivateVPCIDCIDRRange:
    Description: The IP address range for your new Private VPC.
    Type: String
    MinLength: '9'
    MaxLength: '18'
    Default: 10.0.0.0/16
    AllowedPattern: "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\\d{1,2})"
    ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.
  EC2SubnetCIDRRange:
    Description: The IP address range for a subnet in myPrivateVPC.
    Type: String
    MinLength: '9'
    MaxLength: '18'
    Default: 10.0.0.0/24
    AllowedPattern: "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\\d{1,2})"
    ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.
  EC2PublicSubnetCIDRRange:
    Description: The IP address range for a subnet in myVPC.
    Type: String
    MinLength: '9'
```

```
    MaxLength: '18'
    Default: 10.1.0.0/24
    AllowedPattern: "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\\d{1,2})"
    ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.
Mappings:
  AWSRegionToAMI:
    us-east-1:
      '64': ami-fb8e9292
    us-west-2:
      '64': ami-043a5034
    us-west-1:
      '64': ami-7aba833f
    eu-west-1:
      '64': ami-2918e35e
    ap-southeast-1:
      '64': ami-b40d5ee6
    ap-southeast-2:
      '64': ami-3b4bd301
    ap-northeast-1:
      '64': ami-c9562fc8
    sa-east-1:
      '64': ami-215dff3c
Resources:
  myPrivateVPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock:
        Ref: myPrivateVPCIDCIDRRange
      EnableDnsSupport: false
      EnableDnsHostnames: false
      InstanceTenancy: default
  myPrivateEC2Subnet:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId:
        Ref: myPrivateVPC
      CidrBlock:
        Ref: EC2SubnetCIDRRange
  RouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId:
        Ref: myPrivateVPC
  PeeringRoute1:
    Type: AWS::EC2::Route
    Properties:
      DestinationCidrBlock: 0.0.0.0/0
      RouteTableId:
        Ref: RouteTable
      VpcPeeringConnectionId:
        Ref: myVPCPeeringConnection
  SubnetRouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      SubnetId:
        Ref: myPrivateEC2Subnet
      RouteTableId:
        Ref: RouteTable
  myVPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock:
        Ref: myVPCIDCIDRRange
      EnableDnsSupport: true
      EnableDnsHostnames: true
      InstanceTenancy: default
```

```
PublicSubnet:
  Type: AWS::EC2::Subnet
  Properties:
    CidrBlock:
      Ref: EC2PublicSubnetCIDRRange
    VpcId:
      Ref: myVPC
myInternetGateway:
  Type: AWS::EC2::InternetGateway
AttachGateway:
  Type: AWS::EC2::VPCGatewayAttachment
  Properties:
    VpcId:
      Ref: myVPC
    InternetGatewayId:
      Ref: myInternetGateway
PublicRouteTable:
  Type: AWS::EC2::RouteTable
  Properties:
    VpcId:
      Ref: myVPC
PeeringRoute2:
  Type: AWS::EC2::Route
  Properties:
    DestinationCidrBlock:
      Ref: myPrivateVPCIDCIDRRange
    RouteTableId:
      Ref: PublicRouteTable
    VpcPeeringConnectionId:
      Ref: myVPCPeeringConnection
PublicRoute:
  Type: AWS::EC2::Route
  DependsOn: AttachGateway
  Properties:
    RouteTableId:
      Ref: PublicRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    GatewayId:
      Ref: myInternetGateway
PublicSubnetRouteTableAssociation:
  Type: AWS::EC2::SubnetRouteTableAssociation
  Properties:
    SubnetId:
      Ref: PublicSubnet
    RouteTableId:
      Ref: PublicRouteTable
myPrivateVPCEC2SecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Private instance security group
    VpcId:
      Ref: myPrivateVPC
    SecurityGroupIngress:
    - IpProtocol: "-1"
      FromPort: '0'
      ToPort: '65535'
      CidrIp: 0.0.0.0/0
myVPCEC2SecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Public instance security group
    VpcId:
      Ref: myVPC
    SecurityGroupIngress:
    - IpProtocol: tcp
      FromPort: '80'
```

```
        ToPort: '80'
        CidrIp: 0.0.0.0/0
      - IpProtocol: tcp
        FromPort: '22'
        ToPort: '22'
        CidrIp: 0.0.0.0/0
  myPrivateInstance:
    Type: AWS::EC2::Instance
    Properties:
      SecurityGroupIds:
      - Ref: myPrivateVPCEC2SecurityGroup
      SubnetId:
        Ref: myPrivateEC2Subnet
      KeyName:
        Ref: EC2KeyPairName
      ImageId:
        Fn::FindInMap:
        - AWSRegionToAMI
        - Ref: AWS::Region
        - '64'
  myInstance:
    Type: AWS::EC2::Instance
    Properties:
      NetworkInterfaces:
      - AssociatePublicIpAddress: 'true'
        DeviceIndex: '0'
        GroupSet:
        - Ref: myVPCEC2SecurityGroup
        SubnetId:
          Ref: PublicSubnet
      KeyName:
        Ref: EC2KeyPairName
      ImageId:
        Fn::FindInMap:
        - AWSRegionToAMI
        - Ref: AWS::Region
        - '64'
  myVPCPeeringConnection:
    Type: AWS::EC2::VPCPeeringConnection
    Properties:
      VpcId:
        Ref: myVPC
      PeerVpcId:
        Ref: myPrivateVPC
```

AWS::EC2::VPNConnection

Creates a new VPN connection between an existing virtual private gateway and a VPN customer gateway.

For more information, go to CreateVpnConnection in the *Amazon EC2 API Reference*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::VPNConnection",
   "Properties" : {
      "Type (p. 660)" : String,
      "CustomerGatewayId (p. 660)" : GatewayID,
      "StaticRoutesOnly (p. 660)" : Boolean,
      "Tags" :  [ Resource Tag, ... ],
      "VpnGatewayId (p. 661)" : GatewayID
   }
}
```

YAML

```
Type: "AWS::EC2::VPNConnection"
Properties:
  Type (p. 660): String
  CustomerGatewayId (p. 660):
    GatewayID
  StaticRoutesOnly (p. 660): Boolean
  Tags:
    - Resource Tag
  VpnGatewayId (p. 661):
    GatewayID
```

Properties

`Type`

The type of VPN connection this virtual private gateway supports.

Example: "ipsec.1"

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`CustomerGatewayId`

The ID of the customer gateway. This can either be an embedded JSON object or a reference to a Gateway ID.

Required: Yes

Type: String
Update requires: Replacement (p. 90)

`StaticRoutesOnly`

Indicates whether the VPN connection requires static routes.

Required: Conditional: If you are creating a VPN connection for a device that does not support Border Gateway Protocol (BGP), you must specify `true`.

Type: Boolean

Update requires: Replacement (p. 90)

`Tags`

The tags that you want to attach to the resource.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236).

Update requires: No interruption (p. 90).

`VpnGatewayId`

The ID of the virtual private gateway. This can either be an embedded JSON object or a reference to a Gateway ID.

Required: Yes

Type: String
Update requires: Replacement (p. 90)

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "MyVPNConnection" }
```

For the VPNConnection with the logical ID "MyVPNConnection", `Ref` will return the VPN connection's resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Template Example

JSON

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "myVPNConnection" : {
         "Type" : "AWS::EC2::VPNConnection",
         "Properties" : {
            "Type" : "ipsec.1",
         "StaticRoutesOnly" : "true",
            "CustomerGatewayId" : {"Ref" : "myCustomerGateway"},
            "VpnGatewayId" : {"Ref" : "myVPNGateway"}
         }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  myVPNConnection:
```

```
    Type: "AWS::EC2::VPNConnection"
    Properties:
      Type: ipsec.1
      StaticRoutesOnly: true
      CustomerGatewayId:
        !Ref myCustomerGateway
      VpnGatewayId:
        !Ref myVPNGateway
```

AWS::EC2::VPNConnectionRoute

A static route that is associated with a VPN connection between an existing virtual private gateway and a VPN customer gateway. The static route allows traffic to be routed from the virtual private gateway to the VPN customer gateway.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::VPNConnectionRoute",
   "Properties" : {
      "DestinationCidrBlock (p. 662)" : String,
      "VpnConnectionId (p. 663)" : String
   }
}
```

YAML

```
Type: "AWS::EC2::VPNConnectionRoute"
Properties:
  DestinationCidrBlock (p. 662): String
  VpnConnectionId (p. 663): String
```

Properties

`DestinationCidrBlock`

The CIDR block that is associated with the local subnet of the customer network.

Required: Yes.

Type: String

Update requires: Replacement (p. 90)

`VpnConnectionId`

The ID of the VPN connection.

Required: Yes.

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

JSON

```
"MyConnectionRoute0" : {
   "Type" : "AWS::EC2::VPNConnectionRoute",
   "Properties" : {
      "DestinationCidrBlock" : "10.0.0.0/16",
      "VpnConnectionId" : {"Ref" : "Connection0"}
   }
}
```

YAML

```
MyConnectionRoute0:
  Type: "AWS::EC2::VPNConnectionRoute"
  Properties:
    DestinationCidrBlock: 10.0.0.0/16
    VpnConnectionId:
      !Ref Connection0
```

See Also

- CreateVpnConnectionRoute in the *Amazon EC2 API Reference*.

AWS::EC2::VPNGateway

Creates a virtual private gateway. A virtual private gateway is the VPC-side endpoint for your VPN connection.

Topics

- See Also (p. 665)

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::VPNGateway",
   "Properties" : {
      "Type (p. 664)" : String,
      "Tags (p. 664)" : [ Resource Tag, ... ]
   }
}
```

YAML

```
Type: "AWS::EC2::VPNGateway"
Properties:
  Type (p. 664): String
  Tags (p. 664):
    Resource Tag
```

Properties

`Type`

The type of VPN connection this virtual private gateway supports. The only valid value is `"ipsec.1"`.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Tags`

An arbitrary set of tags (key–value pairs) for this resource.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90).

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "MyVPNGateway" }
```

For the VPN gateway with the logical ID "MyVPNGateway", `Ref` will return the gateway's resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

JSON

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "myVPNGateway" : {
         "Type" : "AWS::EC2::VPNGateway",
         "Properties" : {
            "Type" : "ipsec.1",
            "Tags" : [ { "Key" : "Use", "Value" : "Test" } ]
         }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  myVPNGateway:
    Type: "AWS::EC2::VPNGateway"
    Properties:
      Type: ipsec.1
      Tags:
        -
          Key: Use
          Value: Test
```

See Also

- CreateVpnGateway in the *Amazon EC2 API Reference*.

AWS::EC2::VPNGatewayRoutePropagation

Enables a virtual private gateway (VGW) to propagate routes to the routing tables of a VPC.

> **Note**
> If you reference a VPN gateway that is in the same template as your VPN gateway route propagation, you must explicitly declare a dependency on the VPN gateway attachment. The `AWS::EC2::VPNGatewayRoutePropagation` resource cannot use the VPN gateway until it has successfully attached to the VPC. Add a DependsOn (p. 1298) attribute in the `AWS::EC2::VPNGatewayRoutePropagation` resource to explicitly declare a dependency on the VPN gateway attachment.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::EC2::VPNGatewayRoutePropagation",
   "Properties" : {
      "RouteTableIds (p. 666)" : [ String, ... ],
      "VpnGatewayId (p. 666)" : String
   }
}
```

YAML

```
Type: "AWS::EC2::VPNGatewayRoutePropagation"
Properties:
  RouteTableIds (p. 666):
    - String
  VpnGatewayId (p. 666): String
```

Properties

`RouteTableIds`
: A list of routing table IDs that are associated with a VPC. The routing tables must be associated with the same VPC that the virtual private gateway is attached to.

  Required: Yes

  Type: List of route table IDs

  Update requires: No interruption (p. 90)

`VpnGatewayId`
: The ID of the virtual private gateway that is attached to a VPC. The virtual private gateway must be attached to the same VPC that the routing tables are associated with.

  Required: Yes

  Type: String

  Update requires: No interruption (p. 90)

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "myVPNGatewayRouteProp" }
```

For the VPN gateway with the logical ID *myVPNGatewayRouteProp*, `Ref` will return the gateway's resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

JSON

```
"myVPNGatewayRouteProp" : {
  "Type" : "AWS::EC2::VPNGatewayRoutePropagation",
  "Properties" : {
    "RouteTableIds" : [{"Ref" : "PrivateRouteTable"}],
    "VpnGatewayId" : {"Ref" : "VPNGateway"}
  }
}
```

YAML

```
myVPNGatewayRouteProp:
  Type: "AWS::EC2::VPNGatewayRoutePropagation"
  Properties:
    RouteTableIds:
      - !Ref PrivateRouteTable
    VpnGatewayId:
      !Ref VPNGateway
```

See Also

- EnableVgwRoutePropagation in the *Amazon EC2 API Reference*.

AWS::ECR::Repository

The `AWS::ECR::Repository` resource creates an Amazon EC2 Container Registry (Amazon ECR) repository, where users can push and pull Docker images. For more information, see Amazon ECR Repositories in the *Amazon EC2 Container Registry User Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ECR::Repository",
  "Properties" : {
    "RepositoryName" : String,
    "RepositoryPolicyText" : JSON object
  }
```

```
}
```

YAML

```
Type: "AWS::ECR::Repository"
Properties:
  RepositoryName: String
  RepositoryPolicyText: JSON object
```

Properties

`RepositoryName`

A name for the image repository. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the repository name. For more information, see Name Type (p. 1217).

Important
If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

`RepositoryPolicyText`

A policy that controls who has access to the repository and which actions they can perform on it. For more information, see Amazon ECR Repository Policies in the *Amazon EC2 Container Registry User Guide*.

Required: No

Type: JSON object

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name, such as `test-repository`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a repository named `test-repository`. Its policy permits the users `Bob` and `Alice` to push and pull images.

JSON

```
"MyRepository": {
```

```
  "Type": "AWS::ECR::Repository",
  "Properties": {
    "RepositoryName" : "test-repository",
    "RepositoryPolicyText" : {
      "Version": "2008-10-17",
      "Statement": [
        {
          "Sid": "AllowPushPull",
          "Effect": "Allow",
          "Principal": {
            "AWS": [
              "arn:aws:iam::123456789012:user/Bob",
              "arn:aws:iam::123456789012:user/Alice"
            ]
          },
          "Action": [
            "ecr:GetDownloadUrlForLayer",
            "ecr:BatchGetImage",
            "ecr:BatchCheckLayerAvailability",
            "ecr:PutImage",
            "ecr:InitiateLayerUpload",
            "ecr:UploadLayerPart",
            "ecr:CompleteLayerUpload"
          ]
        }
      ]
    }
  }
}
```

YAML

```
MyRepository:
  Type: "AWS::ECR::Repository"
  Properties:
    RepositoryName: "test-repository"
    RepositoryPolicyText:
      Version: "2012-10-17"
      Statement:
        -
          Sid: AllowPushPull
          Effect: Allow
          Principal:
            AWS:
              - "arn:aws:iam::123456789012:user/Bob"
              - "arn:aws:iam::123456789012:user/Alice"
          Action:
            - "ecr:GetDownloadUrlForLayer"
            - "ecr:BatchGetImage"
            - "ecr:BatchCheckLayerAvailability"
            - "ecr:PutImage"
            - "ecr:InitiateLayerUpload"
            - "ecr:UploadLayerPart"
            - "ecr:CompleteLayerUpload"
```

AWS::ECS::Cluster

The `AWS::ECS::Cluster` resource creates an Amazon EC2 Container Service (Amazon ECS) cluster. This resource has no properties; use the Amazon ECS container agent to connect to the cluster. For more information, see Amazon ECS Container Agent in the *Amazon EC2 Container Service Developer Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ECS::Cluster",
  "Properties" : {
    "ClusterName" : String
  }
}
```

YAML

```
Type: "AWS::ECS::Cluster"
Properties:
  ClusterName: String
```

Properties

`ClusterName`

A name for the cluster. If you don't specify a name, AWS CloudFormation generates a unique physical ID for the name. For more information, see Name Type (p. 1217).

> **Important**
> If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

In the following sample, the `Ref` function returns the name of the `MyECSCluster` cluster, such as `MyStack-MyECSCluster-NT5EUXTNTXXD`.

```
{ "Ref": "MyECSCluster" }
```

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following sample declares an Amazon ECS cluster:

JSON

```
"MyCluster": {
  "Type": "AWS::ECS::Cluster"
}
```

YAML

```
MyCluster:
  Type: "AWS::ECS::Cluster"
```

AWS::ECS::Service

The `AWS::ECS::Service` resource creates an Amazon EC2 Container Service (Amazon ECS) service that runs and maintains the requested number of tasks and associated load balancers.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ECS::Service",
  "Properties" : {
    "Cluster" : String,
    "DeploymentConfiguration" : DeploymentConfiguration,
    "DesiredCount" : Integer,
    "LoadBalancers" : [ Load Balancer Objects, ... ],
    "Role" : String,
    "TaskDefinition" : String
  }
}
```

YAML

```
Type: "AWS::ECS::Service"
Properties:
```

```
  Cluster: String
  DeploymentConfiguration:
    DeploymentConfiguration
  DesiredCount: Integer
  LoadBalancers:
    - Load Balancer Objects, ...
  Role: String
  TaskDefinition: String
```

Properties

Note
When you use Auto Scaling or Amazon Elastic Compute Cloud (Amazon EC2) to create container instances for an Amazon ECS cluster, the Amazon ECS service resource must have a dependency on the Auto Scaling group or Amazon EC2 instances. That way the container instances are available and associated with the Amazon ECS cluster before AWS CloudFormation creates the Amazon ECS service.

`Cluster`

The name or Amazon Resource Name (ARN) of the cluster that you want to run your service on. If you do not specify a cluster, Amazon ECS uses the default cluster.

Required: No

Type: String

Update requires: Replacement (p. 90)

`DeploymentConfiguration`

Configures how many tasks run during a deployment.

Required: No

Type: Amazon EC2 Container Service Service DeploymentConfiguration (p. 1131)

Update requires: No interruption (p. 90)

`DesiredCount`

The number of simultaneous tasks that you want to run on the cluster. Specify the tasks with the `TaskDefinition` property,

Required: Yes

Type: Integer

Update requires: No interruption (p. 90)

`LoadBalancers`

A list of load balancer objects to associate with the cluster. For information about the number of load balancers you can specify per service, see Service Load Balancing in the *Amazon EC2 Container Service Developer Guide*.

Required: No

Type: List of Amazon EC2 Container Service Service LoadBalancers (p. 1132)

Update requires: Replacement (p. 90)

`Role`

The name or ARN of an AWS Identity and Access Management (IAM) role that allows your Amazon ECS container agent to make calls to your load balancer.

Note
In some cases, you might need to add a dependency on the service role's policy. For more information, see IAM role policy in DependsOn Attribute (p. 1298).

Required: Conditional. Required only if you specify the `LoadBalancers` property.

Type: String

Update requires: Replacement (p. 90)

`TaskDefinition`

The ARN of the task definition (including the revision number) that you want to run on the cluster, such as `arn:aws:ecs:us-east-1:123456789012:task-definition/mytask:3`. You can't use `:latest` to specify a revision because it's ambiguous. For example, if AWS CloudFormation needed to roll back an update, it wouldn't know which revision to roll back to.

Required: Yes

Type: String

Update requires: Some interruptions (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the ARN.

In the following sample, the `Ref` function returns the ARN of the `MyECSService` service, such as `arn:aws:ecs:us-west-2:123456789012:service/sample-webapp`.

```
{ "Ref": "MyECSService" }
```

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`Name`

The name of the Amazon ECS service, such as `sample-webapp`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Examples

Basic Amazon ECS Service

The following example defines an Amazon ECS service that uses a cluster and task definition which are declared elsewhere in the same template:

JSON

```
"WebApp": {
  "Type": "AWS::ECS::Service",
  "Properties" : {
    "Cluster": { "Ref": "cluster" },
    "DesiredCount": { "Ref": "desiredcount" },
    "TaskDefinition" : { "Ref": "taskdefinition" }
  }
}
```

YAML

```
WebApp:
  Type: "AWS::ECS::Service"
  Properties:
    Cluster:
      Ref: "cluster"
    DesiredCount:
      Ref: "desiredcount"
    TaskDefinition:
      Ref: "taskdefinition"
```

Application Load Balancer

The following sample associates an Application Load Balancer with an Amazon ECS service by referencing an `AWS::ElasticLoadBalancingV2::TargetGroup` resource.

> **Note**
> The Amazon ECS service requires an explicit dependency on the Application load balancer listener rule and the Application load balancer listener, so that the service isn't started before the listener is ready.

JSON

```
"service" : {
  "Type" : "AWS::ECS::Service",
  "DependsOn": ["Listener"],
  "Properties" : {
    "Role" : { "Ref" : "ECSServiceRole" },
    "TaskDefinition" : { "Ref" : "taskdefinition" },
    "DesiredCount" : "1",
    "LoadBalancers" : [{
      "TargetGroupArn" : { "Ref" : "TargetGroup" },
      "ContainerPort" : "80",
      "ContainerName" : "sample-app"
    }],
    "Cluster" : { "Ref" : "ECSCluster" }
  }
}
```

YAML

```
service:
  Type: AWS::ECS::Service
  DependsOn:
  - Listener
  Properties:
    Role:
```

```
      Ref: ECSServiceRole
    TaskDefinition:
      Ref: taskdefinition
    DesiredCount: 1
    LoadBalancers:
    - TargetGroupArn:
        Ref: TargetGroup
      ContainerPort: 80
      ContainerName: sample-app
    Cluster:
      Ref: ECSCluster
```

Related Resources

- To use Application Auto Scaling to scale an Amazon ECS service in response to CloudWatch alarms, use the AWS::ApplicationAutoScaling::ScalableTarget (p. 427) and AWS::ApplicationAutoScaling::ScalingPolicy (p. 430) resources.
- To use an Application Load Balancer to distribute incoming application traffic across multiple targets, use the AWS::ElasticLoadBalancingV2::TargetGroup (p. 739), AWS::ElasticLoadBalancingV2::Listener (p. 731), AWS::ElasticLoadBalancingV2::ListenerRule (p. 733), and AWS::ElasticLoadBalancingV2::LoadBalancer (p. 736) resources.
- For a complete sample template that shows how you can create an Amazon ECS cluster and service, see Amazon EC2 Container Service Template Snippets (p. 283).

AWS::ECS::TaskDefinition

The `AWS::ECS::TaskDefinition` resource describes the container and volume definitions of an Amazon EC2 Container Service (Amazon ECS) task. You can specify which Docker images to use, the required resources, and other configurations related to launching the task definition through an Amazon ECS service or task.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ECS::TaskDefinition",
  "Properties" : {
    "Volumes" : [ Volume Definition, ... ],
    "Family" : String,
    "NetworkMode" : String,
    "TaskRoleArn" : String,
    "ContainerDefinitions" : [ Container Definition, ... ]
  }
```

```
}
```

YAML

```
Type: "AWS::ECS::TaskDefinition"
Properties:
  Volumes:
    - Volume Definition
  Family: String
  NetworkMode: String
  TaskRoleArn: String
  ContainerDefinitions:
    - Container Definition
```

Properties

`Volumes`

A list of volume definitions in JSON format for volumes that you can use in your container definitions.

Required: Yes

Type: List of Amazon EC2 Container Service TaskDefinition Volumes (p. 1144)

Update requires: Replacement (p. 90)

`Family`

The name of a family that this task definition is registered to. A *family* groups multiple versions of a task definition. Amazon ECS gives the first task definition that you registered to a family a revision number of 1. Amazon ECS gives sequential revision numbers to each task definition that you add.

> **Note**
> To use revision numbers when you update a task definition, specify this property. If you don't specify a value, AWS CloudFormation generates a new task definition each time you update it.

Required: No

Type: String

Update requires: Replacement (p. 90)

`NetworkMode`

The Docker networking mode to use for the containers in the task, such as `none`, `bridge`, or `host`. For information about network modes, see `NetworkMode` in the Task Definition Parameters topic in the *Amazon EC2 Container Service Developer Guide*.

Required: No

Type: String

Update requires: Replacement (p. 90)

`TaskRoleArn`

The Amazon Resource Name (ARN) of an AWS Identity and Access Management (IAM) role that grants containers in the task permission to call AWS APIs on your behalf. For more information, see IAM Roles for Tasks in the *Amazon EC2 Container Service Developer Guide*.

Required: No

Type: String

Update requires: Replacement (p. 90)

`ContainerDefinitions`

A list of container definitions in JSON format that describe the containers that make up your task.

Required: Yes

Type: List of Amazon EC2 Container Service TaskDefinition ContainerDefinitions (p. 1133)

Update requires: Replacement (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the Amazon Resource Name (ARN).

In the following example, the `Ref` function returns the ARN of the `MyTaskDefinition` task, such as `arn:aws:ecs:us-west-2:123456789012:task/1abf0f6d-a411-4033-b8eb-a4eed3ad252a`.

```
{ "Ref": "MyTaskDefinition" }
```

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

The following examples define an Amazon ECS task definition, which includes two container definitions and one volume definition.

JSON

```
"taskdefinition": {
  "Type": "AWS::ECS::TaskDefinition",
  "Properties" : {
    "ContainerDefinitions" : [
    {
      "Name": {"Ref": "AppName"},
      "MountPoints": [
        {
          "SourceVolume": "my-vol",
          "ContainerPath": "/var/www/my-vol"
        }
      ],
      "Image":"amazon/amazon-ecs-sample",
      "Cpu": "10",
      "PortMappings":[
        {
          "ContainerPort": {"Ref":"AppContainerPort"},
          "HostPort": {"Ref":"AppHostPort"}
        }
      ],
      "EntryPoint": [
        "/usr/sbin/apache2",
```

```
          "-D",
          "FOREGROUND"
        ],
        "Memory":"500",
        "Essential": "true"
      },
      {
        "Name": "busybox",
        "Image": "busybox",
        "Cpu": "10",
        "EntryPoint": [
          "sh",
          "-c"
        ],
        "Memory": "500",
        "Command": [
          "/bin/sh -c \"while true; do /bin/date > /var/www/my-vol/date; sleep 1; done\""
        ],
        "Essential" : "false",
        "VolumesFrom": [
          {
            "SourceContainer": {"Ref":"AppName"}
          }
        ]
      }],
      "Volumes": [
      {
        "Host": {
          "SourcePath": "/var/lib/docker/vfs/dir/"
        },
        "Name": "my-vol"
      }]
    }
}
```

YAML

```
taskdefinition:
  Type: "AWS::ECS::TaskDefinition"
  Properties:
    ContainerDefinitions:
      -
        Name:
          Ref: "AppName"
        MountPoints:
          -
            SourceVolume: "my-vol"
            ContainerPath: "/var/www/my-vol"
        Image: "amazon/amazon-ecs-sample"
        Cpu: "10"
        PortMappings:
          -
            ContainerPort:
              Ref: "AppContainerPort"
            HostPort:
              Ref: "AppHostPort"
        EntryPoint:
          - "/usr/sbin/apache2"
          - "-D"
          - "FOREGROUND"
        Memory: "500"
        Essential: "true"
      -
        Name: "busybox"
```

```
        Image: "busybox"
        Cpu: "10"
        EntryPoint:
          - "sh"
          - "-c"
        Memory: "500"
        Command:
          - "/bin/sh -c \"while true; do /bin/date > /var/www/my-vol/date; sleep 1; done\""
        Essential: "false"
        VolumesFrom:
          -
            SourceContainer:
              Ref: "AppName"
    Volumes:
      -
        Host:
          SourcePath: "/var/lib/docker/vfs/dir/"
        Name: "my-vol"
```

Related Resources

For a complete sample template that shows how you can create an Amazon ECS cluster and service, see Amazon EC2 Container Service Template Snippets (p. 283).

AWS::EFS::FileSystem

The `AWS::EFS::FileSystem` resource creates a new, empty file system in Amazon Elastic File System (Amazon EFS). You must create a mount target (AWS::EFS::MountTarget (p. 681)) to mount your Amazon EFS file system on an Amazon Elastic Compute Cloud (Amazon EC2) instance. For more information, see the CreateFileSystem API in the *Amazon Elastic File System User Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::EFS::FileSystem",
  "Properties" : {
    "FileSystemTags" : [ FileSystemTags, ... ],
    "PerformanceMode" : String
  }
}
```

YAML

```
Type: "AWS::EFS::FileSystem"
```

```
Properties:
  FileSystemTags:
    - FileSystemTags
  PerformanceMode: String
```

Properties

`FileSystemTags`

Tags to associate with the file system.

Required: No

Type: Amazon Elastic File System FileSystem FileSystemTags (p. 1145)

Update requires: No interruption (p. 90)

`PerformanceMode`

The performance mode of the file system. For valid values, see the `PerformanceMode` parameter for the CreateFileSystem action in the *Amazon Elastic File System User Guide*.

For more information about performance modes, see Amazon EFS Performance in the *Amazon Elastic File System User Guide*.

Required: No

Type: String

Update requires: Replacement (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource ID, such as `fs-47a2c22e`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example declares a file system with a tag key `Name` and tag value `TestFileSystem`:

JSON

```
"WebServerFileSystem" : {
  "Type" : "AWS::EFS::FileSystem",
  "Properties" : {
    "FileSystemTags" : [
      {
        "Key" : "Name",
        "Value" : "TestFileSystem"
      }
    ]
  }
}
```

YAML

```
WebServerFileSystem:
  Type: "AWS::EFS::FileSystem"
  Properties:
    FileSystemTags:
      -
        Key: "Name"
        Value: "TestFileSystem"
```

Additional Resources

For a complete sample template, see Amazon Elastic File System Sample Template (p. 299).

AWS::EFS::MountTarget

The `AWS::EFS::MountTarget` resource creates a mount target for an Amazon Elastic File System (Amazon EFS) file system (AWS::EFS::FileSystem (p. 679)). Use the mount target to mount file systems on Amazon Elastic Compute Cloud (Amazon EC2) instances.

For more information on creating a mount target for a file system, see CreateMountTarget in the *Amazon Elastic File System User Guide*. For a detailed overview of deploying EC2 instances associated with an Amazon EFS file system, see Amazon Elastic File System Sample Template (p. 299).

> **Note**
> EC2 instances and the mount target that they connect to must be in a VPC with DNS enabled.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::EFS::MountTarget",
  "Properties" : {
    "FileSystemId" : String,
    "IpAddress" : String,
    "SecurityGroups" : [ String, ... ],
    "SubnetId" : String
  }
}
```

YAML

```
Type: "AWS::EFS::MountTarget"
```

```
Properties:
  FileSystemId: String
  IpAddress: String
  SecurityGroups:
    [ String, ... ]
  SubnetId: String
```

Properties

`FileSystemId`

The ID of the file system for which you want to create the mount target.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Before updating this property, stop EC2 instances that are using this mount target, and then restart them after the update is complete. This allows the instances to unmount the file system before the mount target is replaced. If you don't stop and restart them, instances or applications that are using those mounts might be disrupted when the mount target is deleted (uncommitted writes might be lost).

`IpAddress`

An IPv4 address that is within the address range of the subnet that is specified in the `SubnetId` property. If you don't specify an IP address, Amazon EFS automatically assigns an address that is within the range of the subnet.

Required: No

Type: String

Update requires: Replacement (p. 90)

Before updating this property, stop EC2 instances that are using this mount target, and then restart them after the update is complete. This allows the instances to unmount the file system before the mount target is replaced. If you don't stop and restart them, instances or applications that are using those mounts might be disrupted when the mount target is deleted (uncommitted writes might be lost).

`SecurityGroups`

A maximum of five VPC security group IDs that are in the same VPC as the subnet that is specified in the `SubnetId` property. For more information about security groups and mount targets, see Security in the *Amazon Elastic File System User Guide*.

Required: Yes

Type: List of strings

Update requires: No interruption (p. 90)

`SubnetId`

The ID of the subnet in which you want to add the mount target.

Note
For each file system, you can create only one mount target per Availability Zone (AZ). All EC2 instances in an AZ share a single mount target for a file system. If you create multiple mount

targets for a single file system, do not specify a subnet that is an AZ that already has a mount target associated with the same file system.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Before updating this property, stop EC2 instances that are using this mount target and then restart them after the update is complete. That way the instances can unmount the file system before the mount target is replaced. If you don't stop and restart them, instances or applications that are using those mounts might be disrupted when the mount target is deleted (uncommitted writes might be lost).

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource ID, such as `fsmt-55a4413c`.

For more information about using the `Ref` function, see Ref (p. 1343).

Template Example

The following example declares a mount target that is associated with a file system, subnet, and security group, which are all declared in the same template. EC2 instances that are in the same AZ as the mount target can use the mount target to connect to the associated file system. For information about mounting file systems on EC2 instances, see Mounting File Systems in the *Amazon Elastic File System User Guide*.

JSON

```
"MountTarget": {
  "Type": "AWS::EFS::MountTarget",
  "Properties": {
    "FileSystemId": { "Ref": "FileSystem" },
    "SubnetId": { "Ref": "Subnet" },
    "SecurityGroups": [ { "Ref": "MountTargetSecurityGroup" } ]
  }
}
```

YAML

```
MountTarget:
  Type: "AWS::EFS::MountTarget"
  Properties:
    FileSystemId:
      Ref: "FileSystem"
    SubnetId:
      Ref: "Subnet"
    SecurityGroups:
      -
        Ref: "MountTargetSecurityGroup"
```

Additional Resources

For a complete sample template, see Amazon Elastic File System Sample Template (p. 299).

AWS::ElastiCache::CacheCluster

The AWS::ElastiCache::CacheCluster type creates an Amazon ElastiCache cache cluster.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::ElastiCache::CacheCluster",
   "Properties" :
   {
      "AutoMinorVersionUpgrade (p. 685)" : Boolean,
      "AZMode" : String,
      "CacheNodeType (p. 685)" : String,
      "CacheParameterGroupName (p. 685)" : String,
      "CacheSecurityGroupNames (p. 686)" : [ String, ... ],
      "CacheSubnetGroupName (p. 686)" : String,
      "ClusterName" : String,
      "Engine (p. 686)" : String,
      "EngineVersion (p. 686)" : String,
      "NotificationTopicArn (p. 687)" : String,
      "NumCacheNodes (p. 687)" : Integer,
      "Port (p. 687)" : Integer,
      "PreferredAvailabilityZone (p. 687)" : String,
      "PreferredAvailabilityZones" : [String, ... ],
      "PreferredMaintenanceWindow (p. 688)" : String,
      "SnapshotArns (p. 688)" : [String, ... ],
      "SnapshotName" : String,
      "SnapshotRetentionLimit" : Integer,
      "SnapshotWindow" : String,
      "Tags" : [Resource Tag, ...],
      "VpcSecurityGroupIds (p. 689)" : [String, ...]
   }
}
```

YAML

```
Type: "AWS::ElastiCache::CacheCluster"
Properties:
  AutoMinorVersionUpgrade (p. 685): Boolean
  AZMode: String
  CacheNodeType (p. 685): String
  CacheParameterGroupName (p. 685): String
  CacheSecurityGroupNames (p. 686):
    - String
  CacheSubnetGroupName (p. 686): String
```

```
  ClusterName: String
  Engine (p. 686): String
  EngineVersion (p. 686): String
  NotificationTopicArn (p. 687): String
  NumCacheNodes (p. 687): Integer
  Port (p. 687): Integer
  PreferredAvailabilityZone (p. 687): String
  PreferredAvailabilityZones:
    - String
  PreferredMaintenanceWindow (p. 688): String
  SnapshotArns (p. 688):
    - String
  SnapshotName: String
  SnapshotRetentionLimit: Integer
  SnapshotWindow: String
  Tags:
    - Resource Tag
  VpcSecurityGroupIds (p. 689):
    - String
```

Properties

For valid values, see CreateCacheCluster in the *Amazon ElastiCache API Reference*.

`AutoMinorVersionUpgrade`

Indicates that minor engine upgrades will be applied automatically to the cache cluster during the maintenance window.

Required: No

Type: Boolean

Default: `true`

Update requires: No interruption (p. 90)

`AZMode`

For Memcached cache clusters, indicates whether the nodes are created in a single Availability Zone or across multiple Availability Zones in the cluster's region. For valid values, see CreateCacheCluster in the *Amazon ElastiCache API Reference*.

Required: Conditional. If you specify multiple Availability Zones in the `PreferredAvailabilityZones` property, you must specify cross Availability Zones for this property.

Type: String

Update requires: No interruption (p. 90)

`CacheNodeType`

The compute and memory capacity of nodes in a cache cluster.

Required: Yes

Type: String

Update requires: Some interruptions (p. 90)

`CacheParameterGroupName`

The name of the cache parameter group that is associated with this cache cluster.

Required: No

Type: String

Update requires: Some interruptions (p. 90)

`CacheSecurityGroupNames`

A list of cache security group names that are associated with this cache cluster. If your cache cluster is in a VPC, specify the `VpcSecurityGroupIds` property instead.

Required: Conditional: If your cache cluster isn't in a VPC, you must specify this property.

Type: List of strings

Update requires: No interruption (p. 90)

`CacheSubnetGroupName`

The cache subnet group that you associate with a cache cluster.

Required: Conditional. If you specified the `VpcSecurityGroupIds` property, you must specify this property.

Type: String

Update requires: Replacement (p. 90)

`ClusterName`

A name for the cache cluster. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the cache cluster. For more information, see Name Type (p. 1217).

> **Important**
> If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

The name must contain 1 to 20 alphanumeric characters or hyphens. The name must start with a letter and cannot end with a hyphen or contain two consecutive hyphens.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Engine`

The name of the cache engine to be used for this cache cluster, such as `memcached` or `redis`.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`EngineVersion`

The version of the cache engine to be used for this cluster.

Required: No

Type: String

Update requires: Some interruptions (p. 90)

`NotificationTopicArn`

The Amazon Resource Name (ARN) of the Amazon Simple Notification Service (SNS) topic to which notifications will be sent.

Required: No

Type: String

Update requires: No interruption (p. 90)

`NumCacheNodes`

The number of cache nodes that the cache cluster should have.

Required: Yes

Type: Integer

Update requires: No interruption (p. 90). However, if the `PreferredAvailabilityZone` and `PreferredAvailabilityZones` properties were not previously specified and you don't specify any new values, an update requires replacement (p. 90).

`Port`

The port number on which each of the cache nodes will accept connections.

Required: No

Type: Integer

Update requires: Replacement (p. 90)

`PreferredAvailabilityZone`

The Amazon EC2 Availability Zone in which the cache cluster is created.

Required: No

Type: String

Update requires: Replacement (p. 90)

`PreferredAvailabilityZones`

For Memcached cache clusters, the list of Availability Zones in which cache nodes are created. The number of Availability Zones listed must equal the number of cache nodes. For example, if you want to create three nodes in two different Availability Zones, you can specify `["us-east-1a", "us-east-1a", "us-east-1b"]`, which would create two nodes in us-east-1a and one node in us-east-1b.

If you specify a subnet group and you're creating your cache cluster in a VPC, you must specify Availability Zones that are associated with the subnets in the subnet group that you've chosen.

If you want all the nodes in the same Availability Zone, use the `PreferredAvailabilityZone` property or repeat the Availability Zone multiple times in the list.

Required: No

Type: List of strings

If you specify an Availability Zone that was previously specified in the template, such as in the `PreferredAvailabilityZone` property, the update requires some interruptions (p. 90). Also, if the `PreferredAvailabilityZones` property was already specified and you're updating its values (regardless of whether you specify the same Availability Zones), the update requires some interruptions (p. 90).

All other updates require replacement (p. 90).

`PreferredMaintenanceWindow`

The weekly time range (in UTC) during which system maintenance can occur.

Required: No

Type: String

Update requires: No interruption (p. 90)

`SnapshotArns`

The ARN of the snapshot file that you want to use to seed a new Redis cache cluster. If you manage a Redis instance outside of Amazon ElastiCache, you can create a new cache cluster in ElastiCache by using a snapshot file that is stored in an Amazon S3 bucket.

Required: No

Type: List of strings

Update requires: Replacement (p. 90)

`SnapshotName`

The name of a snapshot from which to restore data into a new Redis cache cluster.

Required: No

Type: String

Update requires: Replacement (p. 90)

`SnapshotRetentionLimit`

For Redis cache clusters, the number of days for which ElastiCache retains automatic snapshots before deleting them. For example, if you set the value to `5`, a snapshot that was taken today will be retained for 5 days before being deleted.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`SnapshotWindow`

For Redis cache clusters, the daily time range (in UTC) during which ElastiCache will begin taking a daily snapshot of your node group. For example, you can specify `05:00-09:00`.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Tags`

An arbitrary set of tags (key–value pairs) for this cache cluster.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90).

`VpcSecurityGroupIds`

A list of VPC security group IDs. If your cache cluster isn't in a VPC, specify the `CacheSecurityGroupNames` property instead.

Note
You must use the `AWS::EC2::SecurityGroup` resource instead of the `AWS::ElastiCache::SecurityGroup` resource in order to specify an ElastiCache security group that is in a VPC. In addition, if you use the default VPC for your AWS account, you must use the `Fn::GetAtt` function and the `GroupId` attribute to retrieve security group IDs (instead of the `Ref` function). To see a sample template, see the Template Snippet section.

Required: Conditional: If your cache cluster is in a VPC, you must specify this property.

Type: List of strings

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`ConfigurationEndpoint.Address`

The DNS address of the configuration endpoint for the Memcached cache cluster.

`ConfigurationEndpoint.Port`

The port number of the configuration endpoint for the Memcached cache cluster.

`RedisEndpoint.Address`

The DNS address of the configuration endpoint for the Redis cache cluster.

`RedisEndpoint.Port`

The port number of the configuration endpoint for the Redis cache cluster.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Template Snippets

Cluster in a Default VPC

The following snippet describes an ElastiCache cluster in a security group that is in a default VPC. Usually, a security group in a VPC requires the VPC ID to be specified. In this case, no VPC ID is needed because the security group uses the default VPC. If you want to specify a VPC for the security group, specify its `VpcId` property.

For the cache cluster, the `VpcSecurityGroupIds` property is used to associate the cluster with the security group. Because the `VpcSecurityGroupIds` property requires security group IDs (not security group names), the template snippet uses the `Fn::GetAtt` function instead of a `Ref` function on the `ElasticacheSecurityGroup` resource. The `Ref` function will return the security group name. If you specify a VPC ID for the security group, `Ref` returns the security group ID.

Note that `InstanceSecurityGroup` refers to the logical name of a security group that is not actually defined in this snippet. To learn more about the `SourceSecurityGroupName` property, see AWS::EC2::SecurityGroupIngress (p. 616).

JSON

```
"ElasticacheSecurityGroup": {
  "Type": "AWS::EC2::SecurityGroup",
  "Properties": {
    "GroupDescription": "Elasticache Security Group",
    "SecurityGroupIngress": [ {
      "IpProtocol": "tcp",
      "FromPort": "11211",
      "ToPort": "11211",
      "SourceSecurityGroupName": {"Ref": "InstanceSecurityGroup"}
    } ]
  }
},
"ElasticacheCluster": {
  "Type": "AWS::ElastiCache::CacheCluster",
  "Properties": {
    "AutoMinorVersionUpgrade": "true",
    "Engine": "memcached",
    "CacheNodeType": "cache.t2.micro",
    "NumCacheNodes": "1",
    "VpcSecurityGroupIds": [{"Fn::GetAtt": [ "ElasticacheSecurityGroup", "GroupId"]}]
  }
}
```

YAML

```
ElasticacheSecurityGroup:
  Type: "AWS::EC2::SecurityGroup"
  Properties:
    GroupDescription: "Elasticache Security Group"
    SecurityGroupIngress:
      -
        IpProtocol: "tcp"
        FromPort: "11211"
        ToPort: "11211"
        SourceSecurityGroupName:
          Ref: "InstanceSecurityGroup"
ElasticacheCluster:
  Type: "AWS::ElastiCache::CacheCluster"
  Properties:
    AutoMinorVersionUpgrade: "true"
```

```
    Engine: "memcached"
    CacheNodeType: "cache.t2.micro"
    NumCacheNodes: "1"
    VpcSecurityGroupIds:
      -
        Fn::GetAtt:
          - "ElasticacheSecurityGroup"
          - "GroupId"
```

Memcached Nodes in Multiple Availability Zones

The following example launches a cache cluster with three nodes, where two nodes are created in us-west-2a and one is created in us-west-2b.

JSON

```
"myCacheCluster" : {
  "Type": "AWS::ElastiCache::CacheCluster",
  "Properties" : {
    "AZMode" : "cross-az",
    "CacheNodeType" : "cache.m3.medium",
    "Engine" : "memcached",
    "NumCacheNodes" : "3",
    "PreferredAvailabilityZones" : [ "us-west-2a", "us-west-2a", "us-west-2b" ]
  }
}
```

YAML

```
myCacheCluster:
  Type: "AWS::ElastiCache::CacheCluster"
  Properties:
    AZMode: "cross-az"
    CacheNodeType: "cache.m3.medium"
    Engine: "memcached"
    NumCacheNodes: "3"
    PreferredAvailabilityZones:
      - "us-west-2a"
      - "us-west-2a"
      - "us-west-2b"
```

See Also

- CreateCacheCluster in the *Amazon ElastiCache API Reference Guide*
- ModifyCacheCluster in the *Amazon ElastiCache API Reference Guide*

AWS::ElastiCache::ParameterGroup

The AWS::ElastiCache::ParameterGroup type creates a new cache parameter group. Cache parameter groups control the parameters for a cache cluster.

Topics

- See Also (p. 693)

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type": "AWS::ElastiCache::ParameterGroup",
   "Properties": {
      "CacheParameterGroupFamily" : String,
      "Description" : String,
      "Properties" : { String:String, ... }
   }
}
```

YAML

```
Type: "AWS::ElastiCache::ParameterGroup"
Properties:
  CacheParameterGroupFamily: String
  Description: String
  Properties:
    String: String
```

Properties

`CacheParameterGroupFamily`

The name of the cache parameter group family that the cache parameter group can be used with.

Required: Yes

Type: String

Update requires: Updates are not supported.

`Description`

The description for the Cache Parameter Group.

Required: Yes

Type: String

Update requires: Updates are not supported.

`Properties`

A comma-delimited list of parameter name/value pairs. For more information, go to ModifyCacheParameterGroup in the *Amazon ElastiCache API Reference Guide*.

Example:

```
"Properties" : {
   "cas_disabled" : "1",
   "chunk_size_growth_factor" : "1.02"
```

```
}
```

Required: No

Type: Mapping of key-value pairs

Update requires: Updates are not supported.

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

JSON

```
"MyParameterGroup": {
   "Type": "AWS::ElastiCache::ParameterGroup",
   "Properties": {
      "Description": "MyNewParameterGroup",
      "CacheParameterGroupFamily": "memcached1.4",
      "Properties" : {
         "cas_disabled" : "1",
         "chunk_size_growth_factor" : "1.02"
      }
   }
}
```

YAML

```
MyParameterGroup:
  Type: "AWS::ElastiCache::ParameterGroup"
  Properties:
    Description: "MyNewParameterGroup"
    CacheParameterGroupFamily: "memcached1.4"
    Properties:
      cas_disabled: "1"
      chunk_size_growth_factor: "1.02"
```

See Also

- CreateCacheParameterGroup in the *Amazon ElastiCache API Reference Guide*
- ModifyCacheParameterGroup in the *Amazon ElastiCache API Reference Guide*
- AWS CloudFormation Stacks Updates (p. 89)

AWS::ElastiCache::ReplicationGroup

The `AWS::ElastiCache::ReplicationGroup` resource creates an Amazon ElastiCache Redis replication group. A *replication group* is a collection of cache clusters, where one of the clusters is a primary read-write cluster and the others are read-only replicas.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ElastiCache::ReplicationGroup",
  "Properties" : {
    "AutomaticFailoverEnabled" : Boolean,
    "AutoMinorVersionUpgrade" : Boolean,
    "CacheNodeType" : String,
    "CacheParameterGroupName" : String,
    "CacheSecurityGroupNames" : [ String, ... ],
    "CacheSubnetGroupName" : String,
    "Engine" : String,
    "EngineVersion" : String,
    "NodeGroupConfiguration" : [ NodeGroupConfiguration (p. 1149) ],
    "NotificationTopicArn" : String,
    "NumCacheClusters" : Integer,
    "NumNodeGroups" : Integer,
    "Port" : Integer,
    "PreferredCacheClusterAZs" : [ String, ... ],
    "PreferredMaintenanceWindow" : String,
    "PrimaryClusterId" : String,
    "ReplicasPerNodeGroup" : Integer,
    "ReplicationGroupDescription" : String,
    "ReplicationGroupId" : String,
    "SecurityGroupIds" : [ String, ... ],
    "SnapshotArns" : [ String, ... ],
    "SnapshotName" : String,
    "SnapshotRetentionLimit" : Integer,
    "SnapshottingClusterId" : String,
    "SnapshotWindow" : String,
    "Tags" : Resource Tag, ...
  }
}
```

YAML

```
Type: "AWS::ElastiCache::ReplicationGroup"
Properties:
  AutomaticFailoverEnabled: Boolean
  AutoMinorVersionUpgrade: Boolean
  CacheNodeType: String
  CacheParameterGroupName: String
  CacheSecurityGroupNames:
    - String
  CacheSubnetGroupName: String
  Engine: String
  EngineVersion: String
  NodeGroupConfiguration:
```

```
    - NodeGroupConfiguration (p. 1149)
  NotificationTopicArn: String
  NumCacheClusters: Integer
  NumNodeGroups: Integer
  Port: Integer
  PreferredCacheClusterAZs:
    - String
  PreferredMaintenanceWindow: String
  PrimaryClusterId: String
  ReplicasPerNodeGroup: Integer
  ReplicationGroupDescription: String
  ReplicationGroupId: String
  SecurityGroupIds:
    - String
  SnapshotArns:
    - String
  SnapshotName: String
  SnapshotRetentionLimit: Integer
  SnapshottingClusterId: String
  SnapshotWindow: String
  Tags
    - Resource Tag
```

Properties

For more information about each property and valid values, see CreateReplicationGroup in the *Amazon ElastiCache API Reference Guide*.

`AutomaticFailoverEnabled`

Indicates whether Multi-AZ is enabled. When Multi-AZ is enabled, a read-only replica is automatically promoted to a read-write primary cluster if the existing primary cluster fails. If you specify `true`, you must specify a value greater than `1` for the `NumCacheClusters` property. By default, AWS CloudFormation sets the value to `true`.

For Redis (clustered mode enabled) replication groups, you must enable automatic failover.

For information about Multi-AZ constraints, see Replication with Multi-AZ and Automatic Failover (Redis) in the *Amazon ElastiCache User Guide*.

Note
You cannot enable automatic failover for Redis versions earlier than 2.8.6 or for T1 and T2 cache node types.

Important
If you specify the `PrimaryClusterId`, you can use only the following additional parameters:

- `AutomaticFailoverEnabled`
- `NodeGroupConfiguration`
- `NumCacheClusters`
- `NumNodeGroups`
- `PreferredCacheClusterAZs`
- `ReplicationGroupDescription`

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`AutoMinorVersionUpgrade`

Currently, this property isn't used by ElastiCache.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`CacheNodeType`

The compute and memory capacity of nodes in the node group. To see valid values, see CreateReplicationGroup in the *Amazon ElastiCache API Reference Guide*.

Required: No

Type: String

Update requires: No interruption (p. 90)

`CacheParameterGroupName`

The name of the parameter group to associate with this replication group. For valid and default values, see CreateReplicationGroup in the *Amazon ElastiCache API Reference Guide*.

Required: No

Type: String

Update requires: Some interruptions (p. 90)

`CacheSecurityGroupNames`

A list of cache security group names to associate with this replication group.

> **Important**
> If you specify the `CacheSecurityGroupNames` property, don't also specify the `SecurityGroupIds` property.
> The `SecurityGroupIds` property is only for Amazon Virtual Private Cloud (Amazon VPC) security groups. If you specify an Amazon VPC security group, the deployment will fail.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`CacheSubnetGroupName`

The name of a cache subnet group to use for this replication group.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Engine`

The name of the cache engine to use for the cache clusters in this replication group. Currently, you can specify only `redis`.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`EngineVersion`

The version number of the cache engine to use for the cache clusters in this replication group.

Required: No

Type: String

Update requires: No interruption (p. 90)

`NodeGroupConfiguration`

Configuration options for the node group (shard).

Important
If you specify the `PrimaryClusterId`, you can use only the following additional parameters:

- `AutomaticFailoverEnabled`
- `NodeGroupConfiguration`
- `NumCacheClusters`
- `NumNodeGroups`
- `PreferredCacheClusterAZs`
- `ReplicationGroupDescription`

Required: No

Type: List of Amazon ElastiCache ReplicationGroup NodeGroupConfiguration (p. 1149)

Update requires: Replacement (p. 90)

`NotificationTopicArn`

The Amazon Resource Name (ARN) of the Amazon Simple Notification Service topic to which notifications are sent.

Required: No

Type: String

Update requires: No interruption (p. 90)

`NumCacheClusters`

The number of cache clusters for this replication group. If automatic failover is enabled, you must specify a value greater than `1`. For valid values, see CreateReplicationGroup in the *Amazon ElastiCache API Reference Guide*.

If you specify more than one node group (shard), this property is ignored. Use the `ReplicasPerNodeGroup` property instead.

Important
If you specify the `PrimaryClusterId`, you can use only the following additional parameters:

- `AutomaticFailoverEnabled`
- `NodeGroupConfiguration`

- `NumCacheClusters`
- `NumNodeGroups`
- `PreferredCacheClusterAZs`
- `ReplicationGroupDescription`

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`NumNodeGroups`

The number of node groups (shards) for this Redis (clustered mode enabled) replication group. For Redis (clustered mode disabled), omit this property.

Important
If you specify the `PrimaryClusterId`, you can use only the following additional parameters:

- `AutomaticFailoverEnabled`
- `NodeGroupConfiguration`
- `NumCacheClusters`
- `NumNodeGroups`
- `PreferredCacheClusterAZs`
- `ReplicationGroupDescription`

Required: No

Type: Integer

Update requires: Replacement (p. 90)

`Port`

The port number on which each member of the replication group accepts connections.

Required: No

Type: Integer

Update requires: Replacement (p. 90)

`PreferredCacheClusterAZs`

A list of Availability Zones (AZs) in which the cache clusters in this replication group are created.

Important
If you specify the `PrimaryClusterId`, you can use only the following additional parameters:

- `AutomaticFailoverEnabled`
- `NodeGroupConfiguration`
- `NumCacheClusters`
- `NumNodeGroups`
- `PreferredCacheClusterAZs`
- `ReplicationGroupDescription`

Required: No

Type: List of strings

Update requires: Replacement (p. 90)

`PreferredMaintenanceWindow`

The weekly time range during which system maintenance can occur. Use the following format to specify a time range: `ddd:hh24:mi-ddd:hh24:mi` (24H Clock UTC). For example, you can specify `sun:22:00-sun:23:30` for Sunday from 10 PM to 11:30 PM.

Required: No

Type: String

Update requires: No interruption (p. 90)

`PrimaryClusterId`

The cache cluster that ElastiCache uses as the primary cluster for the replication group. The cache cluster must have a status of `available`.

Important
If you specify the `PrimaryClusterId`, you can use only the following additional parameters:

- `AutomaticFailoverEnabled`
- `NodeGroupConfiguration`
- `NumCacheClusters`
- `NumNodeGroups`
- `PreferredCacheClusterAZs`
- `ReplicationGroupDescription`

Required: Conditional. This property is optional if you specify the `NumCacheClusters`, `NumNodeGroups`, or the `ReplicasPerNodeGroup` property.

Type: String

Update requires: No interruption (p. 90)

`ReplicasPerNodeGroup`

The number of replica nodes in each node group (shard). For valid values, see CreateReplicationGroup in the *Amazon ElastiCache API Reference Guide*.

Required: No

Type: Integer

Update requires: Replacement (p. 90)

`ReplicationGroupDescription`

The description of the replication group.

Important
If you specify the `PrimaryClusterId`, you can use only the following additional parameters:

- `AutomaticFailoverEnabled`
- `NodeGroupConfiguration`
- `NumCacheClusters`

- `NumNodeGroups`
- `PreferredCacheClusterAZs`
- `ReplicationGroupDescription`

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`ReplicationGroupId`

An ID for the replication group. If you don't specify an ID, AWS CloudFormation generates a unique physical ID. For more information, see Name Type (p. 1217).

Required: No

Type: String

Update requires: Replacement (p. 90)

`SecurityGroupIds`

A list of Amazon Virtual Private Cloud (Amazon VPC) security groups to associate with this replication group.

Important
If you specify the `SecurityGroupIds` property, don't also specify the `CacheSecurityGroupNames` property.
The `CacheSecurityGroupNames` property is only for EC2-Classic security groups. If you specify an EC2-Classic security group, the deployment will fail.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`SnapshotArns`

A single-element string list that specifies an ARN of a Redis `.rdb` snapshot file that is stored in Amazon Simple Storage Service (Amazon S3). The snapshot file populates the node group. The Amazon S3 object name in the ARN cannot contain commas. For example, you can specify `arn:aws:s3:::my_bucket/snapshot1.rdb`.

Required: No

Type: List of strings

Update requires: Replacement (p. 90)

`SnapshotName`

The name of a snapshot from which to restore data into the replication group.

Required: No

Type: String

Update requires: Replacement (p. 90)

`SnapshotRetentionLimit`

The number of days that ElastiCache retains automatic snapshots before deleting them.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`SnapshottingClusterId`

The ID of the cache cluster that ElastiCache uses as the daily snapshot source for the replication group.

Required: No

Type: String

Update requires: No interruption (p. 90)

`SnapshotWindow`

The time range (in UTC) when ElastiCache takes a daily snapshot of your node group that you specified in the `SnapshottingClusterId` property. For example, you can specify `05:00-09:00`.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Tags`

An arbitrary set of tags (key–value pairs) for this replication group.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90).

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

In the following sample, the `Ref` function returns the name of the `myReplicationGroup` replication group, such as `abc12xmy3d1w3hv6`.

```
{ "Ref": "myReplicationGroup" }
```

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`PrimaryEndPoint.Address`

The DNS address of the primary read-write cache node.

`PrimaryEndPoint.Port`

The number of the port that the primary read-write cache engine is listening on.

`ReadEndPoint.Addresses`

A string with a list of endpoints for the read-only replicas. The order of the addresses map to the order of the ports from the `ReadEndPoint.Ports` attribute.

`ReadEndPoint.Ports`

A string with a list of ports for the read-only replicas. The order of the ports map to the order of the addresses from the `ReadEndPoint.Addresses` attribute.

`ReadEndPoint.Addresses.List`

A list of endpoints for the read-only replicas. The order of the addresses map to the order of the ports from the `ReadEndPoint.Ports.List` attribute.

`ReadEndPoint.Ports.List`

A list of ports for the read-only replicas. The order of the ports map to the order of the addresses from the `ReadEndPoint.Addresses.List` attribute.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Examples

Replication Group with Two Nodes

The following example declares a replication group with two nodes and automatic failover enabled.

JSON

```
"myReplicationGroup" : {
  "Type": "AWS::ElastiCache::ReplicationGroup",
  "Properties": {
    "ReplicationGroupDescription" : "my description",
    "NumCacheClusters" : "2",
    "Engine" : "redis",
    "CacheNodeType" : "cache.m3.medium",
    "AutoMinorVersionUpgrade" : "true",
    "AutomaticFailoverEnabled" : "true",
    "CacheSubnetGroupName" : "subnetgroup",
    "EngineVersion" : "2.8.6",
    "PreferredMaintenanceWindow" : "wed:09:25-wed:22:30",
    "SnapshotRetentionLimit" : "4",
    "SnapshotWindow" : "03:30-05:30"
  }
}
```

YAML

```
myReplicationGroup:
  Type: "AWS::ElastiCache::ReplicationGroup"
  Properties:
    ReplicationGroupDescription: "my description"
```

```
    NumCacheClusters: "2"
    Engine: "redis"
    CacheNodeType: "cache.m3.medium"
    AutoMinorVersionUpgrade: "true"
    AutomaticFailoverEnabled: "true"
    CacheSubnetGroupName: "subnetgroup"
    EngineVersion: "2.8.6"
    PreferredMaintenanceWindow: "wed:09:25-wed:22:30"
    SnapshotRetentionLimit: "4"
    SnapshotWindow: "03:30-05:30"
```

Replication Group with Two Node Groups

The following example declares a replication group with two nodes groups (shards) with three replicas in each group.

JSON

```
"BasicReplicationGroup" : {
  "Type" : "AWS::ElastiCache::ReplicationGroup",
  "Properties" : {
    "AutomaticFailoverEnabled" : true,
    "AutoMinorVersionUpgrade" : true,
    "CacheNodeType" : "cache.r3.large",
    "CacheSubnetGroupName" : { "Ref" : "CacheSubnetGroup" },
    "Engine" : "redis",
    "EngineVersion" : "3.2",
    "NumNodeGroups" : "2",
    "ReplicasPerNodeGroup" : "3",
    "Port" : 6379,
    "PreferredMaintenanceWindow" : "sun:05:00-sun:09:00",
    "ReplicationGroupDescription" : "A sample replication group",
    "SecurityGroupIds" : [
      { "Ref" : "ReplicationGroupSG" }
    ],
    "SnapshotRetentionLimit" : 5,
    "SnapshotWindow" : "10:00-12:00"
  }
}
```

YAML

```
BasicReplicationGroup:
  Type: AWS::ElastiCache::ReplicationGroup
  Properties:
    AutomaticFailoverEnabled: true
    AutoMinorVersionUpgrade: true
    CacheNodeType: cache.r3.large
    CacheSubnetGroupName:
      Ref: CacheSubnetGroup
    Engine: redis
    EngineVersion: '3.2'
    NumNodeGroups: '2'
    ReplicasPerNodeGroup: '3'
    Port: 6379
    PreferredMaintenanceWindow: sun:05:00-sun:09:00
    ReplicationGroupDescription: A sample replication group
    SecurityGroupIds:
    - Ref: ReplicationGroupSG
    SnapshotRetentionLimit: 5
    SnapshotWindow: 10:00-12:00
```

AWS::ElastiCache::SecurityGroup

The `AWS::ElastiCache::SecurityGroup` resource creates a cache security group. For more information about cache security groups, go to Cache Security Groups in the *Amazon ElastiCache User Guide* or go to CreateCacheSecurityGroup in the *Amazon ElastiCache API Reference Guide*.

To create an ElastiCache cluster in a VPC, use the AWS::EC2::SecurityGroup (p. 608) resource. For more information, see the `VpcSecurityGroupIds` property in the AWS::ElastiCache::CacheCluster (p. 684) resource.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ElastiCache::SecurityGroup",
  "Properties" :
  {
    "Description" : String
  }
}
```

YAML

```
Type: "AWS::ElastiCache::SecurityGroup"
Properties:
  Description: String
```

Properties

`Description`

A description for the cache security group.

Type: String

Required: No

Update requires: Updates are not supported.

Return Values

Ref

When you specify the `AWS::ElastiCache::SecurityGroup` resource as an argument to the `Ref` function, AWS CloudFormation returns the `CacheSecurityGroupName` property of the cache security group.

For more information about using the `Ref` function, see Ref (p. 1343).

AWS::ElastiCache::SecurityGroupIngress

The AWS::ElastiCache::SecurityGroupIngress type authorizes ingress to a cache security group from hosts in specified Amazon EC2 security groups. For more information about ElastiCache security group ingress, go to AuthorizeCacheSecurityGroupIngress in the *Amazon ElastiCache API Reference Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ElastiCache::SecurityGroupIngress",
  "Properties" :
  {
    "CacheSecurityGroupName" : String,
    "EC2SecurityGroupName" : String,
    "EC2SecurityGroupOwnerId" : String
  }
}
```

YAML

```
Type: "AWS::ElastiCache::SecurityGroupIngress"
Properties:
  CacheSecurityGroupName: String
  EC2SecurityGroupName: String
  EC2SecurityGroupOwnerId: String
```

Properties

`CacheSecurityGroupName`

The name of the Cache Security Group to authorize.

Type: String

Required: Yes

Update requires: Updates are not supported.

`EC2SecurityGroupName`

Name of the EC2 Security Group to include in the authorization.

Type: String

Required: Yes

Update requires: Updates are not supported.

`EC2SecurityGroupOwnerId`

Specifies the AWS Account ID of the owner of the EC2 security group specified in the EC2SecurityGroupName property. The AWS access key ID is not an acceptable value.

Type: String

Required: No

Update requires: Updates are not supported.

AWS::ElastiCache::SubnetGroup

Creates a cache subnet group. For more information about cache subnet groups, go to Cache Subnet Groups in the *Amazon ElastiCache User Guide* or go to CreateCacheSubnetGroup in the *Amazon ElastiCache API Reference Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ElastiCache::SubnetGroup",
  "Properties" : {
    "CacheSubnetGroupName" : String,
    "Description (p. 707)" : String,
    "SubnetIds (p. 707)" : [ String, ... ]
  }
}
```

YAML

```
Type: "AWS::ElastiCache::SubnetGroup"
Properties:
  CacheSubnetGroupName: String
  Description (p. 707): String
  SubnetIds (p. 707):
    - String
```

Properties

`CacheSubnetGroupName`

A name for the cache subnet group. If you don't specify a name, AWS CloudFormation generates a unique physical ID. For more information, see Name Type (p. 1217).

Important
If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Description`

The description for the cache subnet group.

Type: String

Required: Yes

Update requires: No interruption (p. 90)

`SubnetIds`

The Amazon EC2 subnet IDs for the cache subnet group.

Type: String list

Required: Yes

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

JSON

```
"SubnetGroup" : {
    "Type" : "AWS::ElastiCache::SubnetGroup",
    "Properties" : {
        "Description" : "Cache Subnet Group",
        "SubnetIds" : [ { "Ref" : "Subnet1" }, { "Ref" : "Subnet2" } ]
    }
}
```

YAML

```
SubnetGroup:
  Type: "AWS::ElastiCache::SubnetGroup"
  Properties:
    Description: "Cache Subnet Group"
```

```
    SubnetIds:
      - Ref: "Subnet1"
      - Ref: "Subnet2"
```

AWS::ElasticBeanstalk::Application

Creates an Elastic Beanstalk application.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::ElasticBeanstalk::Application",
   "Properties" : {
      "ApplicationName" : String,
      "Description (p. 709)" : String
   }
}
```

YAML

```
Type: "AWS::ElasticBeanstalk::Application"
Properties:
  ApplicationName: String
  Description (p. 709): String
```

Properties

`ApplicationName`

A name for the Elastic Beanstalk application. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the application name. For more information, see Name Type (p. 1217).

> **Important**
> If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Description`

An optional description of this application.

Required: No

Type: String

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

JSON

```
{
   "Type" : "AWS::ElasticBeanstalk::Application",
   "Properties" : {
      "ApplicationName" : "SampleAWSElasticBeanstalkApplication",
      "Description" : "AWS Elastic Beanstalk PHP Sample Application"
   }
}
```

YAML

```
Type: "AWS::ElasticBeanstalk::Application"
Properties:
  ApplicationName: "SampleAWSElasticBeanstalkApplication"
  Description: "AWS Elastic Beanstalk PHP Sample Application"
```

See Also

- For a complete Elastic Beanstalk sample template, see Elastic Beanstalk Template Snippets (p. 314).

AWS::ElasticBeanstalk::ApplicationVersion

Creates an application version, an iteration of deployable code, for an Elastic Beanstalk application.

Topics

- See Also (p. 711)

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ElasticBeanstalk::ApplicationVersion",
  "Properties" : {
    "ApplicationName" : String,
    "Description" : String,
    "SourceBundle" : { SourceBundle }
  }
}
```

YAML

```
Type: AWS::ElasticBeanstalk::ApplicationVersion
Properties:
  ApplicationName: String
  Description: String
  SourceBundle:
    SourceBundle
```

Members

`ApplicationName`

Name of the Elastic Beanstalk application that is associated with this application version.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Description`

A description of this application version.

Required: No

Type: String

Update requires: Some interruptions (p. 90)

`SourceBundle`

The location of the source bundle for this version.

Required: Yes

Type: Source Bundle (p. 1148)

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

JSON

```
"myAppVersion" :{
  "Type" : "AWS::ElasticBeanstalk::ApplicationVersion",
  "Properties" : {
    "ApplicationName" : {"Ref" : "myApp"},
    "Description" : "my sample version",
    "SourceBundle" : {
      "S3Bucket" : { "Fn::Join" :
        ["-", [ "elasticbeanstalk-samples", { "Ref" : "AWS::Region" } ] ] },
      "S3Key" : "php-sample.zip"
    }
  }
}
```

YAML

```
myAppVersion:
  Type: "AWS::ElasticBeanstalk::ApplicationVersion"
  Properties:
    ApplicationName:
      Ref: "myApp"
    Description: "my sample version"
    SourceBundle:
      S3Bucket:
        Fn::Join:
          - "-"
          -
            - "elasticbeanstalk-samples"
            - Ref: "AWS::Region"
      S3Key: "php-sample.zip"
```

See Also

- For a complete Elastic Beanstalk sample template, see Elastic Beanstalk Template Snippets (p. 314).

AWS::ElasticBeanstalk::ConfigurationTemplate

Creates a configuration template for an Elastic Beanstalk application. You can use configuration templates to deploy different versions of an application by using the configuration settings that you define in the configuration template.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ElasticBeanstalk::ConfigurationTemplate",
  "Properties" : {
    "ApplicationName" : String,
    "Description" : String,
    "EnvironmentId" : String,
    "OptionSettings" : [ OptionSetting, ... ],
    "SolutionStackName" : String,
    "SourceConfiguration" : Source configuration
  }
}
```

YAML

```
Type: "AWS::ElasticBeanstalk::ConfigurationTemplate"
Properties:
  ApplicationName: String
  Description: String
  EnvironmentId: String
  OptionSettings:
    - OptionSetting
  SolutionStackName: String
  SourceConfiguration:
    Source configuration
```

Members

`ApplicationName`

Name of the Elastic Beanstalk application that is associated with this configuration template.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Description`

An optional description for this configuration.

Type: String

Required: No

Update requires: Some interruptions (p. 90)

`EnvironmentId`

An environment whose settings you want to use to create the configuration template. You must specify this property if you don't specify the `SolutionStackName` or `SourceConfiguration` properties.

Type: String

Required: Conditional

Update requires: Replacement (p. 90)

`OptionSettings`

A list of OptionSettings (p. 1147) for this Elastic Beanstalk configuration, such as the instance type. For a complete list of Elastic Beanstalk configuration options, see Option Values, in the *AWS Elastic Beanstalk Developer Guide*.

Type: A list of OptionSettings (p. 1147).

Required: No

Update requires: Some interruptions (p. 90)

`SolutionStackName`

The name of an Elastic Beanstalk solution stack that this configuration will use. A solution stack specifies the operating system, architecture, and application server for a configuration template, such as `64bit Amazon Linux 2013.09 running Tomcat 7 Java 7`. For more information, see Supported Platforms in the *AWS Elastic Beanstalk Developer Guide*.

You must specify this property if you don't specify the `EnvironmentId` or `SourceConfiguration` properties.

Type: String

Required: Conditional

Update requires: Replacement (p. 90)

`SourceConfiguration`

A configuration template that is associated with another Elastic Beanstalk application. If you specify the `SolutionStackName` property and the `SourceConfiguration` property, the solution stack in the source configuration template must match the value that you specified for the `SolutionStackName` property.

You must specify this property if you don't specify the `EnvironmentId` or `SolutionStackName` properties.

Type: Elastic Beanstalk SourceConfiguration Property Type (p. 1149)

Required: Conditional

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

This example of an ElasticBeanstalk `ConfigurationTemplate` is found in the AWS CloudFormation sample template ElasticBeanstalkSample.template, which also provides an example of its use within an `AWS::ElasticBeanstalk::Application`.

JSON

```
"myConfigTemplate" : {
  "Type" : "AWS::ElasticBeanstalk::ConfigurationTemplate",
  "Properties" : {
    "ApplicationName" :{"Ref" : "myApp"},
    "Description" : "my sample configuration template",
    "EnvironmentId" : "",
    "SourceConfiguration" : {
      "ApplicationName" : {"Ref" : "mySecondApp"},
      "TemplateName" : {"Ref" : "mySourceTemplate"}
    },
    "SolutionStackName" : "64bit Amazon Linux running PHP 5.3",
    "OptionSettings" : [ {
      "Namespace" : "aws:autoscaling:launchconfiguration",
      "OptionName" : "EC2KeyName",
      "Value" : { "Ref" : "KeyName" }
    } ]
  }
}
```

YAML

```
myConfigTemplate:
  Type: "AWS::ElasticBeanstalk::ConfigurationTemplate"
  Properties:
    ApplicationName:
      Ref: "myApp"
    Description: "my sample configuration template"
    EnvironmentId: ""
    SourceConfiguration:
      ApplicationName:
        Ref: "mySecondApp"
      TemplateName:
        Ref: "mySourceTemplate"
    SolutionStackName: "64bit Amazon Linux running PHP 5.3"
    OptionSettings:
      -
        Namespace: "aws:autoscaling:launchconfiguration"
        OptionName: "EC2KeyName"
        Value:
          Ref: "KeyName"
```

See Also

- AWS::ElasticBeanstalk::Application (p. 708)
- Option Values in the *AWS Elastic Beanstalk Developer Guide*
- For a complete Elastic Beanstalk sample template, see Elastic Beanstalk Template Snippets (p. 314).

AWS::ElasticBeanstalk::Environment

Creates or updates an AWS Elastic Beanstalk environment.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::ElasticBeanstalk::Environment",
   "Properties" : {
      "ApplicationName (p. 715)" : String,
      "CNAMEPrefix (p. 716)" : String,
      "Description (p. 716)" :  String,
      "EnvironmentName" :  String,
      "OptionSettings (p. 716)" : [ OptionSettings, ... ],
      "SolutionStackName (p. 716)" : String,
      "Tags" : [ Resource Tag, ... ],
      "TemplateName (p. 717)" : String,
      "Tier" : Environment Tier,
      "VersionLabel (p. 717)" : String
   }
}
```

YAML

```
Type: "AWS::ElasticBeanstalk::Environment"
Properties:
  ApplicationName (p. 715): String
  CNAMEPrefix (p. 716): String
  Description (p. 716): String
  EnvironmentName: String
  OptionSettings (p. 716):
    - OptionSettings
  SolutionStackName (p. 716): String
  Tags:
    - Resource Tag, ...
  TemplateName (p. 717): String
  Tier:
    Environment Tier
  VersionLabel (p. 717): String
```

Properties

`ApplicationName`

The name of the application that is associated with this environment.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`CNAMEPrefix`

A prefix for your Elastic Beanstalk environment URL.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Description`

A description that helps you identify this environment.

Required: No

Type: String

Update requires: No interruption (p. 90)

`EnvironmentName`

A name for the Elastic Beanstalk environment. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the environment name. For more information, see Name Type (p. 1217).

Important
If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

`OptionSettings`

Key-value pairs defining configuration options for this environment, such as the instance type. These options override the values that are defined in the solution stack or the configuration template (p. 711). If you remove any options during a stack update, the removed options revert to default values.

Required: No

Type: A list of OptionSettings (p. 1147).

Update requires: Some interruptions (p. 90)

`SolutionStackName`

The name of an Elastic Beanstalk solution stack that this configuration will use. For more information, see Supported Platforms in the *AWS Elastic Beanstalk Developer Guide*. You must specify either this parameter or an Elastic Beanstalk configuration template name.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Tags`

An arbitrary set of tags (key–value pairs) for this environment.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: You can update tags only if you update another property that requires that the environment be replaced, such as the `ApplicationName` property.

`TemplateName`

The name of the Elastic Beanstalk configuration template to use with the environment. You must specify either this parameter or a solution stack name.

Required: No

Type: String

Update requires: Some interruptions (p. 90)

`Tier`

Specifies the tier to use in creating this environment. The environment tier that you choose determines whether Elastic Beanstalk provisions resources to support a web application that handles HTTP(S) requests or a web application that handles background-processing tasks.

Required: No

Type: Elastic Beanstalk Environment Tier Property Type (p. 1146)

Update requires: See Elastic Beanstalk Environment Tier Property Type (p. 1146)

`VersionLabel`

The version to associate with the environment.

Required: No

Type: String

Update requires: Some interruptions (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`EndpointURL`

The URL to the load balancer for this environment.

Example:

awseb-myst-myen-132MQC4KRLAMD-1371280482.us-east-1.elb.amazonaws.com

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Examples

Simple Environment

JSON

```
{
   "Type" : "AWS::ElasticBeanstalk::Environment",
   "Properties" : {
      "ApplicationName" : { "Ref" : "sampleApplication" },
      "Description" :  "AWS Elastic Beanstalk Environment running PHP Sample Application",
      "EnvironmentName" :  "SamplePHPEnvironment",
      "TemplateName" : "DefaultConfiguration",
      "VersionLabel" : "Initial Version"
   }
}
```

YAML

```
Type: "AWS::ElasticBeanstalk::Environment"
Properties:
  ApplicationName:
    Ref: sampleApplication
  Description: "AWS Elastic Beanstalk Environment running PHP Sample Application"
  EnvironmentName: SamplePHPEnvironment
  TemplateName: DefaultConfiguration
  VersionLabel: "Initial Version"
```

Environment with Embedded Option Settings

JSON

```
{
   "Type" : "AWS::ElasticBeanstalk::Environment",
   "Properties" : {
      "ApplicationName" : { "Ref" : "sampleApplication" },
      "Description" :  "AWS Elastic Beanstalk Environment running Python Sample
 Application",
      "EnvironmentName" :  "SamplePythonEnvironment",
      "SolutionStackName" : "64bit Amazon Linux running Python",
      "OptionSettings" : [ {
         "Namespace" : "aws:autoscaling:launchconfiguration",
         "OptionName" : "EC2KeyName",
         "Value" : { "Ref" : "KeyName" }
      } ],
      "VersionLabel" : "Initial Version"
   }
```

```
}
```

YAML

```
Type: "AWS::ElasticBeanstalk::Environment"
Properties:
  ApplicationName:
    Ref: sampleApplication
  Description: "AWS Elastic Beanstalk Environment running Python Sample Application"
  EnvironmentName: SamplePythonEnvironment
  SolutionStackName: "64bit Amazon Linux running Python"
  OptionSettings:
    -
      Namespace: "aws:autoscaling:launchconfiguration"
      OptionName: EC2KeyName
      Value:
        Ref: KeyName
  VersionLabel: "Initial Version"
```

See Also

- Launching New Environments in the *AWS Elastic Beanstalk Developer Guide*
- Managing Environments in the *AWS Elastic Beanstalk Developer Guide*
- For a complete Elastic Beanstalk sample template, see Elastic Beanstalk Template Snippets (p. 314).

AWS::ElasticLoadBalancing::LoadBalancer

The AWS::ElasticLoadBalancing::LoadBalancer type creates a LoadBalancer.

> **Note**
> If this resource has a public IP address and is also in a VPC that is defined in the same template, you must use the `DependsOn` attribute to declare a dependency on the VPC-gateway attachment. For more information, see DependsOn Attribute (p. 1298).

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type": "AWS::ElasticLoadBalancing::LoadBalancer",
   "Properties": {
      "AccessLoggingPolicy" : AccessLoggingPolicy,
      "AppCookieStickinessPolicy (p. 721)" : [ AppCookieStickinessPolicy, ... ],
      "AvailabilityZones (p. 721)" : [ String, ... ],
      "ConnectionDrainingPolicy" : ConnectionDrainingPolicy,
```

```
      "ConnectionSettings" : ConnectionSettings,
      "CrossZone" : Boolean,
      "HealthCheck (p. 721)" : HealthCheck,
      "Instances (p. 722)" : [ String, ... ],
      "LBCookieStickinessPolicy (p. 722)" : [ LBCookieStickinessPolicy, ... ],
      "LoadBalancerName" : String,
      "Listeners (p. 722)" : [ Listener, ... ],
      "Policies (p. 723)" : [ ElasticLoadBalancing Policy, ... ],
      "Scheme (p. 723)" : String,
      "SecurityGroups (p. 723)" : [ Security Group, ... ],
      "Subnets (p. 723)" : [ String, ... ],
      "Tags" : [ Resource Tag, ... ]
   }
}
```

YAML

```
Type: "AWS::ElasticLoadBalancing::LoadBalancer"
Properties:
  AccessLoggingPolicy:
    AccessLoggingPolicy
  AppCookieStickinessPolicy (p. 721):
    - AppCookieStickinessPolicy
  AvailabilityZones (p. 721):
    - String
  ConnectionDrainingPolicy:
    ConnectionDrainingPolicy
  ConnectionSettings:
    ConnectionSettings
  CrossZone: Boolean
  HealthCheck (p. 721):
    HealthCheck
  Instances (p. 722):
    - String
  LBCookieStickinessPolicy (p. 722):
    - LBCookieStickinessPolicy
  LoadBalancerName: String
  Listeners (p. 722):
    - Listener
  Policies (p. 723):
    - ElasticLoadBalancing Policy
  Scheme (p. 723): String,
  SecurityGroups (p. 723):
    - Security Group
  Subnets (p. 723):
    - String
  Tags:
    - Resource Tag
```

Properties

`AccessLoggingPolicy`

Captures detailed information for all requests made to your load balancer, such as the time a request was received, client’s IP address, latencies, request path, and server responses.

Required: No

Type: Elastic Load Balancing AccessLoggingPolicy (p. 1151)

Update requires: No interruption (p. 90)

`AppCookieStickinessPolicy`

Generates one or more stickiness policies with sticky session lifetimes that follow that of an application-generated cookie. These policies can be associated only with HTTP/HTTPS listeners.

Required: No

Type: A list of AppCookieStickinessPolicy (p. 1152) objects.

Update requires: No interruption (p. 90)

`AvailabilityZones`

The Availability Zones in which to create the load balancer. You can specify the `AvailabilityZones` or `Subnets` property, but not both.

Note
For load balancers that are in a VPC, specify the `Subnets` property.

Required: No

Type: List of strings

Update requires: Replacement (p. 90) if you did not have an Availability Zone specified and you are adding one or if you are removing all Availability Zones. Otherwise, update requires no interruption (p. 90).

`ConnectionDrainingPolicy`

Whether deregistered or unhealthy instances can complete all in-flight requests.

Required: No

Type: Elastic Load Balancing ConnectionDrainingPolicy (p. 1152)

Update requires: No interruption (p. 90)

`ConnectionSettings`

Specifies how long front-end and back-end connections of your load balancer can remain idle.

Required: No

Type: Elastic Load Balancing ConnectionSettings (p. 1153)

Update requires: No interruption (p. 90)

`CrossZone`

Whether cross-zone load balancing is enabled for the load balancer. With cross-zone load balancing, your load balancer nodes route traffic to the back-end instances across all Availability Zones. By default the `CrossZone` property is `false`.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`HealthCheck`

Application health check for the instances.

Required: No

Type: ElasticLoadBalancing HealthCheck Type (p. 1154).

Update requires: Replacement (p. 90) if you did not have a health check specified and you are adding one or if you are removing a health check. Otherwise, update requires no interruption (p. 90).

`Instances`

A list of EC2 instance IDs for the load balancer.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`LBCookieStickinessPolicy`

Generates a stickiness policy with sticky session lifetimes controlled by the lifetime of the browser (user-agent), or by a specified expiration period. This policy can be associated only with HTTP/HTTPS listeners.

Required: No

Type: A list of LBCookieStickinessPolicy (p. 1155) objects.

Update requires: No interruption (p. 90)

`LoadBalancerName`

A name for the load balancer. For valid values, see the `LoadBalancerName` parameter for the `CreateLoadBalancer` action in the *Elastic Load Balancing API Reference version 2012-06-01*.

If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the load balancer. The name must be unique within your set of load balancers. For more information, see Name Type (p. 1217).

Important
If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Listeners`

One or more listeners for this load balancer. Each listener must be registered for a specific port, and you cannot have more than one listener for a given port.

Important
If you update the property values for a listener specified by the `Listeners` property, AWS CloudFormation will delete the existing listener and create a new one with the updated properties. During the time that AWS CloudFormation is performing this action, clients will not be able to connect to the load balancer.

Required: Yes

Type: A list of ElasticLoadBalancing Listener Property Type (p. 1156) objects.

Update requires: No interruption (p. 90)

`Policies`

A list of elastic load balancing policies to apply to this elastic load balancer. Specify only back-end server policies. For more information, see DescribeLoadBalancerPolicyTypes in the *Elastic Load Balancing API Reference version 2012-06-01*.

Required: No

Type: A list of ElasticLoadBalancing policy (p. 1158) objects.

Update requires: No interruption (p. 90)

`Scheme`

For load balancers attached to an Amazon VPC, this parameter can be used to specify the type of load balancer to use. Specify `internal` to create an internal load balancer with a DNS name that resolves to private IP addresses or `internet-facing` to create a load balancer with a publicly resolvable DNS name, which resolves to public IP addresses.

Note
If you specify `internal`, you must specify subnets to associate with the load balancer, not Availability Zones.

Required: No

Type: String

Update requires: Replacement (p. 90)

`SecurityGroups`

Required: No

Type: A list of security groups assigned to your load balancer within your virtual private cloud (VPC).

Update requires: No interruption (p. 90)

`Subnets`

A list of subnet IDs in your virtual private cloud (VPC) to attach to your load balancer. Do not specify multiple subnets that are in the same Availability Zone. You can specify the `AvailabilityZones` or `Subnets` property, but not both.

For more information about using Elastic Load Balancing in a VPC, see How Do I Use Elastic Load Balancing in Amazon VPC in the *Elastic Load Balancing Developer Guide*.

Required: No

Type: List of strings

Update requires: Replacement (p. 90) if you did not have an subnet specified and you are adding one or if you are removing all subnets. Otherwise, update requires no interruption (p. 90). To update the load balancer to another subnet that is in the same Availability Zone, you must do two updates. You must first update the load balancer to use a subnet in different Availability Zone. After the update is complete, update the load balancer to use the new subnet that is in the original Availability Zone.

`Tags`

An arbitrary set of tags (key-value pairs) for this load balancer.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example, `mystack-myelb-1WQN7BJGDB5YQ`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`CanonicalHostedZoneName`

The name of the Amazon Route 53 hosted zone that is associated with the load balancer.

> **Important**
> If you specify `internal` for the Elastic Load Balancing scheme, use `DNSName` instead. For an `internal` scheme, the load balancer doesn't have a `CanonicalHostedZoneName` value.

Example: `mystack-myelb-15HMABG9ZCN57-1013119603.us-east-1.elb.amazonaws.com`

`CanonicalHostedZoneNameID`

The ID of the Amazon Route 53 hosted zone name that is associated with the load balancer.

Example: `Z3DZXE0Q79N41H`

`DNSName`

The DNS name for the load balancer.

Example: `mystack-myelb-15HMABG9ZCN57-1013119603.us-east-1.elb.amazonaws.com`

`SourceSecurityGroup.GroupName`

The security group that you can use as part of your inbound rules for your load balancer's back-end Amazon EC2 application instances.

Example: `amazon-elb`

`SourceSecurityGroup.OwnerAlias`

The owner of the source security group.

Example: `amazon-elb-sg`

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Examples

A load balancer with a health check and access logs

JSON

```
"ElasticLoadBalancer" : {
  "Type" : "AWS::ElasticLoadBalancing::LoadBalancer",
  "Properties" : {
    "AvailabilityZones" : { "Fn::GetAZs" : "" },
    "Instances" : [ { "Ref" : "Ec2Instance1" },{ "Ref" : "Ec2Instance2" } ],
    "Listeners" : [ {
```

```
      "LoadBalancerPort" : "80",
      "InstancePort" : { "Ref" : "WebServerPort" },
      "Protocol" : "HTTP"
    } ],
    "HealthCheck" : {
      "Target" : {
        "Fn::Join" : [ "", [ "HTTP:", { "Ref" : "WebServerPort" }, "/" ] ]
      },
      "HealthyThreshold" : "3",
      "UnhealthyThreshold" : "5",
      "Interval" : "30",
      "Timeout" : "5"
    },
    "AccessLoggingPolicy": {
      "S3BucketName": {
        "Ref": "S3LoggingBucket"
      },
      "S3BucketPrefix": "MyELBLogs",
      "Enabled": "true",
      "EmitInterval" : "60"
    },
  "DependsOn": "S3LoggingBucketPolicy"
  }
}
```

YAML

```
ElasticLoadBalancer:
  Type: AWS::ElasticLoadBalancing::LoadBalancer
  Properties:
    AvailabilityZones:
      Fn::GetAZs: ''
    Instances:
    - Ref: Ec2Instance1
    - Ref: Ec2Instance2
    Listeners:
    - LoadBalancerPort: '80'
      InstancePort:
        Ref: WebServerPort
      Protocol: HTTP
    HealthCheck:
      Target:
        Fn::Join:
        - ''
        - - 'HTTP:'
          - Ref: WebServerPort
          - "/"
      HealthyThreshold: '3'
      UnhealthyThreshold: '5'
      Interval: '30'
      Timeout: '5'
    AccessLoggingPolicy:
      S3BucketName:
        Ref: S3LoggingBucket
      S3BucketPrefix: MyELBLogs
      Enabled: 'true'
      EmitInterval: '60'
    DependsOn: S3LoggingBucketPolicy
```

A load balancer with access logging enabled

The following sample snippet creates an Amazon S3 bucket with a bucket policy that allows the load balancer to store information in the `Logs/AWSLogs/`*`AWS account number`*`/` folder. The load balancer also includes an explicit dependency on the bucket policy, which is required before the load balancer can write to the bucket.

JSON

```
"S3LoggingBucket": {
  "Type": "AWS::S3::Bucket"
},
"S3LoggingBucketPolicy": {
  "Type": "AWS::S3::BucketPolicy",
  "Properties": {
    "Bucket": {
      "Ref": "S3LoggingBucket"
    },
    "PolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [ {
        "Sid": "ELBAccessLogs20130930",
        "Effect": "Allow",
        "Resource": {
          "Fn::Join": [
            "",
            [
              "arn:aws:s3:::",
              { "Ref": "S3LoggingBucket" },
              "/",
              "Logs",
              "/AWSLogs/",
              { "Ref": "AWS::AccountId" },
              "/*"
            ]
          ]
        },
        "Principal": { "Ref": "ElasticLoadBalancingAccountID" },
        "Action": [
          "s3:PutObject"
        ]
      } ]
    }
  }
},
"ElasticLoadBalancer": {
  "Type": "AWS::ElasticLoadBalancing::LoadBalancer",
  "Properties": {
    "AvailabilityZones": { "Fn::GetAZs": "" },
    "Listeners": [{
      "LoadBalancerPort": "80",
      "InstancePort": "80",
      "Protocol": "HTTP"
    }],
    "HealthCheck": {
      "Target": "HTTP:80/",
      "HealthyThreshold": "3",
      "UnhealthyThreshold": "5",
      "Interval": "30",
      "Timeout": "5"
    },
    "AccessLoggingPolicy": {
      "S3BucketName": {
        "Ref": "S3LoggingBucket"
      },
      "S3BucketPrefix": "Logs",
      "Enabled": "true",
      "EmitInterval" : "60"
    }
```

```
  },
  "DependsOn": "S3LoggingBucketPolicy"
}
```

YAML

```
S3LoggingBucket:
  Type: AWS::S3::Bucket
S3LoggingBucketPolicy:
  Type: AWS::S3::BucketPolicy
  Properties:
    Bucket:
      Ref: S3LoggingBucket
    PolicyDocument:
      Version: '2012-10-17'
      Statement:
      - Sid: ELBAccessLogs20130930
        Effect: Allow
        Resource:
          Fn::Join:
          - ''
          - - 'arn:aws:s3:::'
            - Ref: S3LoggingBucket
            - "/"
            - Logs
            - "/AWSLogs/"
            - Ref: AWS::AccountId
            - "/*"
        Principal:
          Ref: ElasticLoadBalancingAccountID
        Action:
        - s3:PutObject
ElasticLoadBalancer:
  Type: AWS::ElasticLoadBalancing::LoadBalancer
  Properties:
    AvailabilityZones:
      Fn::GetAZs: ''
    Listeners:
    - LoadBalancerPort: '80'
      InstancePort: '80'
      Protocol: HTTP
    HealthCheck:
      Target: HTTP:80/
      HealthyThreshold: '3'
      UnhealthyThreshold: '5'
      Interval: '30'
      Timeout: '5'
    AccessLoggingPolicy:
      S3BucketName:
        Ref: S3LoggingBucket
      S3BucketPrefix: Logs
      Enabled: 'true'
      EmitInterval: '60'
  DependsOn: S3LoggingBucketPolicy
```

A load balancer with a connection draining policy

The following snippet enables a connection draining policy that ends connections to a deregistered or unhealthy instance after 60 seconds.

JSON

```
"ElasticLoadBalancer" : {
```

```
  "Type" : "AWS::ElasticLoadBalancing::LoadBalancer",
  "Properties" : {
    "AvailabilityZones" : { "Fn::GetAZs" : "" },
    "Instances" : [ { "Ref" : "Ec2Instance1" },{ "Ref" : "Ec2Instance2" } ],
    "Listeners": [{
      "LoadBalancerPort": "80",
      "InstancePort": "80",
      "Protocol": "HTTP"
    }],
    "HealthCheck": {
      "Target": "HTTP:80/",
      "HealthyThreshold": "3",
      "UnhealthyThreshold": "5",
      "Interval": "30",
      "Timeout": "5"
    },
    "ConnectionDrainingPolicy": {
      "Enabled" : "true",
      "Timeout" : "60"
    }
  }
}
```

YAML

```
ElasticLoadBalancer:
  Type: AWS::ElasticLoadBalancing::LoadBalancer
  Properties:
    AvailabilityZones:
      Fn::GetAZs: ''
    Instances:
    - Ref: Ec2Instance1
    - Ref: Ec2Instance2
    Listeners:
    - LoadBalancerPort: '80'
      InstancePort: '80'
      Protocol: HTTP
    HealthCheck:
      Target: HTTP:80/
      HealthyThreshold: '3'
      UnhealthyThreshold: '5'
      Interval: '30'
      Timeout: '5'
    ConnectionDrainingPolicy:
      Enabled: 'true'
      Timeout: '60'
```

A load balancer with multiple policies

The following snippet creates a load balancer with listeners on port 80 and 443. The snippet applies a proxy on port 80 and a back-end server authentication policy on port 443.

JSON

```
"ElasticLoadBalancer": {
  "Type": "AWS::ElasticLoadBalancing::LoadBalancer",
  "Properties": {
    "SecurityGroups" : { "Ref" : "SecurityGroups" },
    "Scheme" : "internet-facing",
    "AvailabilityZones": { "Fn::GetAZs": "" },
    "Listeners": [
      {
        "LoadBalancerPort": "80",
```

```
        "InstancePort": "80",
        "Protocol": "TCP",
        "InstanceProtocol" : "TCP"
      },
      {
        "LoadBalancerPort": "443",
        "InstancePort": "443",
        "Protocol": "HTTPS",
        "SSLCertificateId" : { "Ref" : "CertARN" },
        "PolicyNames" : ["MySSLNegotiationPolicy", "MyAppCookieStickinessPolicy"]
      }
    ],
    "Policies" : [
      {
        "PolicyName" : "MySSLNegotiationPolicy",
        "PolicyType" : "SSLNegotiationPolicyType",
        "Attributes" : [
            { "Name" : "Protocol-TLSv1", "Value" : "true" },
            { "Name" : "Protocol-SSLv2", "Value" : "true" },
            { "Name" : "Protocol-SSLv3", "Value" : "false" },
            { "Name" : "DHE-RSA-AES256-SHA", "Value" : "true" }
        ]
      },
      {
        "PolicyName" : "MyAppCookieStickinessPolicy",
        "PolicyType" : "AppCookieStickinessPolicyType",
        "Attributes" : [
          { "Name" : "CookieName", "Value" : "MyCookie" }
        ]
      },
      {
        "PolicyName" : "MyPublicKeyPolicy",
        "PolicyType" : "PublicKeyPolicyType",
        "Attributes" : [
          { "Name" : "PublicKey", "Value" : { "Fn::Join" : [ "\n", [
            "MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDh/51Aohx5VrpmlfGHZCzciMBa",
            "fkHve+MQYYJcxmNUKMdsWnz9WtVfKxxWUU7Cfor4lorYmENGCG8FWqCoLDMFs7pN",
            "yGEtpsrlKhzZWtgY1d7eGrUrBil03bI90E2KW0j4qAwGYAC8xixOkNClicojeEz4",
            "f4rr3sUf+ZBSsuMEuwIDAQAB" ] ] }
          }
        ]
      },
      {
        "PolicyName" : "MyBackendServerAuthenticationPolicy",
        "PolicyType" : "BackendServerAuthenticationPolicyType",
        "Attributes" : [
          { "Name" : "PublicKeyPolicyName", "Value" : "MyPublicKeyPolicy" }
        ],
        "InstancePorts" : [ "443" ]
      },
      {
        "PolicyName" : "EnableProxyProtocol",
        "PolicyType" : "ProxyProtocolPolicyType",
        "Attributes" : [
          { "Name"  : "ProxyProtocol", "Value" : "true" }
        ],
        "InstancePorts" : ["80"]
      }
    ]
  }
}
```

YAML

```
ElasticLoadBalancer:
  Type: AWS::ElasticLoadBalancing::LoadBalancer
  Properties:
    SecurityGroups:
      Ref: SecurityGroups
    Scheme: internet-facing
    AvailabilityZones:
      Fn::GetAZs: ''
    Listeners:
    - LoadBalancerPort: '80'
      InstancePort: '80'
      Protocol: TCP
      InstanceProtocol: TCP
    - LoadBalancerPort: '443'
      InstancePort: '443'
      Protocol: HTTPS
      SSLCertificateId:
        Ref: CertARN
      PolicyNames:
      - MySSLNegotiationPolicy
      - MyAppCookieStickinessPolicy
    Policies:
    - PolicyName: MySSLNegotiationPolicy
      PolicyType: SSLNegotiationPolicyType
      Attributes:
      - Name: Protocol-TLSv1
        Value: 'true'
      - Name: Protocol-SSLv2
        Value: 'true'
      - Name: Protocol-SSLv3
        Value: 'false'
      - Name: DHE-RSA-AES256-SHA
        Value: 'true'
    - PolicyName: MyAppCookieStickinessPolicy
      PolicyType: AppCookieStickinessPolicyType
      Attributes:
      - Name: CookieName
        Value: MyCookie
    - PolicyName: MyPublicKeyPolicy
      PolicyType: PublicKeyPolicyType
      Attributes:
      - Name: PublicKey
        Value:
          Fn::Join:
          - "\n"
          - - MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDh/51Aohx5VrpmlfGHZCzciMBa
            - fkHve+MQYYJcxmNUKMdsWnz9WtVfKxxWUU7Cfor4lorYmENGCG8FWqCoLDMFs7pN
            - yGEtpsrlKhzZWtgY1d7eGrUrBil03bI90E2KW0j4qAwGYAC8xixOkNClicojeEz4
            - f4rr3sUf+ZBSsuMEuwIDAQAB
    - PolicyName: MyBackendServerAuthenticationPolicy
      PolicyType: BackendServerAuthenticationPolicyType
      Attributes:
      - Name: PublicKeyPolicyName
        Value: MyPublicKeyPolicy
      InstancePorts:
      - '443'
    - PolicyName: EnableProxyProtocol
      PolicyType: ProxyProtocolPolicyType
      Attributes:
      - Name: ProxyProtocol
        Value: 'true'
      InstancePorts:
```

```
      - '80'
```

Additional Examples

You can view additional examples from the AWS CloudFormation sample template collection: Sample Templates (p. 1367).

AWS::ElasticLoadBalancingV2::Listener

The `AWS::ElasticLoadBalancingV2::Listener` resource creates a listener for an Elastic Load Balancing Application load balancer. The listener checks for connection requests and forwards them to one or more target groups. For more information, see the Listeners for Your Application Load Balancers in the *Application Load Balancers Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ElasticLoadBalancingV2::Listener",
  "Properties" : {
    "Certificates" : [ Certificates (p. 1160), ... ],
    "DefaultActions" : [ DefaultActions (p. 1161), ... ],
    "LoadBalancerArn" : String,
    "Port" : Integer,
    "Protocol" : String,
    "SslPolicy" : String
  }
}
```

YAML

```
Type: "AWS::ElasticLoadBalancingV2::Listener"
Properties:
  Certificates:
    - Certificates (p. 1160)
  DefaultActions:
    - DefaultActions (p. 1161)
  LoadBalancerArn: String
  Port: Integer
  Protocol: String
  SslPolicy: String
```

Properties

`Certificates`

The SSL server certificate for the listener. With a certificate, you can encrypt traffic between the load balancer and the clients that initiate HTTPS sessions, and traffic between the load balancer and your targets.

Required: Conditional. If you specify `HTTPS` for the `Protocol` property, specify a certificate.

Type: List of Elastic Load Balancing Listener Certificates (p. 1160)

Update requires: No interruption (p. 90)

`DefaultActions`

The default actions that the listener takes when handling incoming requests.

Required: Yes

Type: List of Elastic Load Balancing Listener DefaultActions (p. 1161)

Update requires: No interruption (p. 90)

`LoadBalancerArn`

The Amazon Resource Name (ARN) of the load balancer to associate with the listener.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Port`

The port on which the listener listens for requests.

For valid values, see the `Port` parameter for the CreateListener action in the *Elastic Load Balancing API Reference version 2015-12-01*.

Required: Yes

Type: Integer

Update requires: No interruption (p. 90)

`Protocol`

The protocol that clients must use to send requests to the listener.

For valid values, see the `Protocol` parameter for the CreateListener action in the *Elastic Load Balancing API Reference version 2015-12-01*.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`SslPolicy`

The security policy that defines the ciphers and protocols that the load balancer supports.

Required: No

Type: String

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the listener's ARN, such as `arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a listener for the `myLoadBalancer` resource. The listener's default action is to forward requests to the `myTargetGroup` target group.

JSON

```
"Listener": {
  "Type": "AWS::ElasticLoadBalancingV2::Listener",
  "Properties": {
    "DefaultActions": [{
      "Type": "forward",
      "TargetGroupArn": { "Ref": "myTargetGroup" }
    }],
    "LoadBalancerArn": { "Ref": "myLoadBalancer" },
    "Port": "8000",
    "Protocol": "HTTP"
  }
}
```

YAML

```
Listener:
  Type: AWS::ElasticLoadBalancingV2::Listener
  Properties:
    DefaultActions:
    - Type: forward
      TargetGroupArn:
        Ref: myTargetGroup
    LoadBalancerArn:
      Ref: myLoadBalancer
    Port: '8000'
    Protocol: HTTP
```

AWS::ElasticLoadBalancingV2::ListenerRule

The `AWS::ElasticLoadBalancingV2::ListenerRule` resource defines which requests an Elastic Load Balancing listener takes action on and the action that it takes. For more information, see the Listeners for Your Application Load Balancers in the *Application Load Balancers Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ElasticLoadBalancingV2::ListenerRule",
  "Properties" : {
    "Actions" : [ Actions (p. 1161), ... ],
    "Conditions" : [ Conditions (p. 1162), ... ],
    "ListenerArn" : String,
    "Priority" : Integer
  }
}
```

YAML

```
Type: "AWS::ElasticLoadBalancingV2::ListenerRule"
Properties:
  Actions:
    - Actions (p. 1161)
  Conditions:
    - Conditions (p. 1162)
  ListenerArn: String
  Priority: Integer
```

Properties

`Actions`

The action that the listener takes when a request meets the specified condition.

Required: Yes

Type: List of Elastic Load Balancing ListenerRule Actions (p. 1161)

Update requires: No interruption (p. 90)

`Conditions`

The conditions under which a rule takes effect.

Required: Yes

Type: List of Elastic Load Balancing ListenerRule Conditions (p. 1162)

Update requires: No interruption (p. 90)

`ListenerArn`

The Amazon Resource Name (ARN) of the listener that the rule applies to.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Priority`

The priority for the rule. Elastic Load Balancing evaluates rules in priority order, from the lowest value to the highest value. If a request satisfies a rule, Elastic Load Balancing ignores all subsequent rules.

> **Note**
> A target group can have only one rule with a given priority.

For valid values, see the `Priority` parameter for the CreateRule action in the *Elastic Load Balancing API Reference version 2015-12-01*.

Required: Yes

Type: Integer

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the rule's ARN, such as `arn:aws:elasticloadbalancing:us-west-2:123456789012:listener-rule/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2/9683b2d02a6cabee`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a rule that forwards requests to the `TargetGroup` target group if the request URL contains the `/img/*` pattern.

JSON

```
"ListenerRule": {
  "Type": "AWS::ElasticLoadBalancingV2::ListenerRule",
  "Properties": {
    "Actions": [{
      "Type": "forward",
      "TargetGroupArn": { "Ref": "TargetGroup" }
    }],
    "Conditions": [{
      "Field": "path-pattern",
      "Values": [ "/img/*" ]
    }],
    "ListenerArn": { "Ref": "Listener" },
    "Priority": 1
  }
}
```

YAML

```
ListenerRule:
  Type: AWS::ElasticLoadBalancingV2::ListenerRule
  Properties:
    Actions:
    - Type: forward
```

```
      TargetGroupArn:
        Ref: TargetGroup
    Conditions:
    - Field: path-pattern
      Values:
      - "/img/*"
    ListenerArn:
      Ref: Listener
    Priority: 1
```

AWS::ElasticLoadBalancingV2::LoadBalancer

The `AWS::ElasticLoadBalancingV2::LoadBalancer` resource creates an Elastic Load Balancing Application load balancer that distributes incoming application traffic across multiple targets (such as EC2 instances) in multiple Availability Zones. For more information, see the *Application Load Balancers Guide*.

> **Note**
> AWS CloudFormation does not automatically create tags (key–value pairs) for an Elastic Load Balancing Application load balancer. You must use the `Tags` (p. 738) property to create tags to associate with the load balancer.

Elastic Load Balancing API Reference version 2015-12-01

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ElasticLoadBalancingV2::LoadBalancer",
  "Properties" : {
    "LoadBalancerAttributes" : [ LoadBalancerAttributes (p. 1163), ... ],
    "Name" : String,
    "Scheme" : String,
    "SecurityGroups" : [ String, ... ],
    "Subnets" : [ String, ... ],
    "Tags" : [ Resource Tag, ... ]
  }
}
```

YAML

```
Type: "AWS::ElasticLoadBalancingV2::LoadBalancer"
Properties:
  LoadBalancerAttributes:
    - LoadBalancerAttributes (p. 1163)
  Name: String
  Scheme: String
  SecurityGroups:
    - String
```

```
Subnets:
  - String
Tags:
  - Resource Tag
```

Properties

`LoadBalancerAttributes`

Specifies the load balancer configuration.

Required: No

Type: A list of Elastic Load Balancing LoadBalancer LoadBalancerAttributes (p. 1163)

Update requires: No interruption (p. 90)

`Name`

Specifies a name for the load balancer. This name must be unique within your AWS account and can have a maximum of 32 alphanumeric characters and hyphens. A name can't begin or end with a hyphen.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Scheme`

Specifies whether the load balancer is internal or Internet-facing. An internal load balancer routes requests to targets using private IP addresses. An Internet-facing load balancer routes requests from clients over the Internet to targets in your public subnets.

For valid and default values, see the `Scheme` parameter for the `CreateLoadBalancer` action in the *Elastic Load Balancing API Reference version 2015-12-01*.

Required: No

Type: String

Update requires: Replacement (p. 90)

`SecurityGroups`

Specifies a list of the IDs of the security groups to assign to the load balancer.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`Subnets`

Specifies a list of at least two IDs of the subnets to associate with the load balancer. The subnets must be in different Availability Zones.

Required: Yes

Type: List of strings

Update requires: No interruption (p. 90)

`Tags`

Specifies an arbitrary set of tags (key–value pairs) to associate with this load balancer. Use tags to manage your resources.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the ARN of the load balancer, for example:

```
arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-internal-load-
balancer/50dc6c495c0c9188
```

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for the following attributes.

`DNSName`

The DNS name for the Application load balancer, for example `my-load-balancer-424835706.us-west-2.elb.amazonaws.com`.

`CanonicalHostedZoneID`

The ID of the Amazon Route 53 hosted zone associated with the load balancer, for example `Z2P70J7EXAMPLE`.

`LoadBalancerFullName`

The full name of the Application load balancer, for example `app/my-load-balancer/50dc6c495c0c9188`.

`LoadBalancerName`

The name of the Application load balancer, for example `my-load-balancer`.

`SecurityGroups`

The IDs of the security groups for the Application load balancer, for example `sg-123456a`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Example

The following example creates an internal load balancer with an idle timeout period of `50` seconds.

JSON

```
"loadBalancer" : {
  "Type": "AWS::ElasticLoadBalancingV2::LoadBalancer",
```

```
  "Properties": {
    "Scheme" : "internal",
    "Subnets" : [ {"Ref": "SubnetAZ1"}, {"Ref" : "SubnetAZ2"}],
    "LoadBalancerAttributes" : [
      { "Key" : "idle_timeout.timeout_seconds", "Value" : "50" }
    ],
    "SecurityGroups": [{"Ref": "SecurityGroup1"}, {"Ref" : "SecurityGroup2"}],
    "Tags" : [
      { "Key" : "key", "Value" : "value" },
      { "Key" : "key2", "Value" : "value2" }
    ]
  }
}
```

YAML

```
loadBalancer:
  Type: AWS::ElasticLoadBalancingV2::LoadBalancer
  Properties:
    Scheme: internal
    Subnets:
    - Ref: SubnetAZ1
    - Ref: SubnetAZ2
    LoadBalancerAttributes:
    - Key: idle_timeout.timeout_seconds
      Value: '50'
    SecurityGroups:
    - Ref: SecurityGroup1
    - Ref: SecurityGroup2
    Tags:
    - Key: key
      Value: value
    - Key: key2
      Value: value2
```

AWS::ElasticLoadBalancingV2::TargetGroup

The `AWS::ElasticLoadBalancingV2::TargetGroup` resource creates an Elastic Load Balancing target group that routes requests to one or more registered targets, such as EC2 instances. For more information, see the Target Groups for Your Application Load Balancers in the *Application Load Balancers Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::ElasticLoadBalancingV2::TargetGroup",
  "Properties" : {
    "HealthCheckIntervalSeconds" : Integer,
    "HealthCheckPath" : String,
```

```
    "HealthCheckPort" : String,
    "HealthCheckProtocol" : String,
    "HealthCheckTimeoutSeconds" : Integer,
    "HealthyThresholdCount" : Integer,
    "Matcher" : Matcher (p. 1164),
    "Name" : String,
    "Port" : Integer,
    "Protocol" : String,
    "Tags" : [ Resource Tag (p. 1236), ... ],
    "TargetGroupAttributes" : [ TargetGroupAttributes (p. 1165), ... ],
    "Targets" : [ TargetDescription (p. 1164), ... ],
    "UnhealthyThresholdCount" : Integer,
    "VpcId" : String
  }
}
```

YAML

```
Type: "AWS::ElasticLoadBalancingV2::TargetGroup"
Properties:
  HealthCheckIntervalSeconds: Integer
  HealthCheckPath: String
  HealthCheckPort: String
  HealthCheckProtocol: String
  HealthCheckTimeoutSeconds: Integer
  HealthyThresholdCount: Integer
  Matcher: Matcher (p. 1164)
  Name: String
  Port: Integer
  Protocol: String
  Tags:
    - Resource Tag (p. 1236)
  TargetGroupAttributes:
    - TargetGroupAttributes (p. 1165)
  Targets:
    - TargetDescription (p. 1164)
  UnhealthyThresholdCount: Integer
  VpcId: String
```

Properties

`HealthCheckIntervalSeconds`

The approximate number of seconds between health checks for an individual target.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`HealthCheckPath`

The ping path destination where Elastic Load Balancing sends health check requests.

Required: No

Type: String

Update requires: No interruption (p. 90)

`HealthCheckPort`

The port that the load balancer uses when performing health checks on the targets.

For valid and default values, see the `HealthCheckPort` parameter for the CreateTargetGroup action in the *Elastic Load Balancing API Reference version 2015-12-01*.

Required: No

Type: String

Update requires: No interruption (p. 90)

`HealthCheckProtocol`

The protocol that the load balancer uses when performing health checks on the targets, such as `HTTP` or `HTTPS`.

For valid and default values, see the `HealthCheckProtocol` parameter for the CreateTargetGroup action in the *Elastic Load Balancing API Reference version 2015-12-01*.

Required: No

Type: String

Update requires: No interruption (p. 90)

`HealthCheckTimeoutSeconds`

The number of seconds to wait for a response before considering that a health check has failed.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`HealthyThresholdCount`

The number of consecutive successful health checks that are required before an unhealthy target is considered healthy.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`Matcher`

The HTTP codes that a healthy target uses when responding to a health check.

Required: No

Type: Elastic Load Balancing TargetGroup Matcher (p. 1164)

Update requires: No interruption (p. 90)

`Name`

A name for the target group.

> **Important**
> The target group name should be shorter than 22 characters because AWS CloudFormation uses the target group name to create the name of the load balancer.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Port`

The port on which the targets receive traffic.

Required: Yes

Type: Integer

Update requires: Replacement (p. 90)

`Protocol`

The protocol to use for routing traffic to the targets.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Tags`

An arbitrary set of tags (key–value pairs) for the target group. Use tags to help manage resources.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90).

`TargetGroupAttributes`

Target group configurations.

Required: No

Type: List of Elastic Load Balancing TargetGroup TargetGroupAttributes (p. 1165)

Update requires: No interruption (p. 90)

`Targets`

The targets to add to this target group.

Required: No

Type: List of Elastic Load Balancing TargetGroup TargetDescription (p. 1164)

Update requires: No interruption (p. 90)

`UnhealthyThresholdCount`

The number of consecutive failed health checks that are required before a target is considered unhealthy.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`VpcId`

The ID of the VPC in which your targets are located.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the target group's Amazon Resource Name (ARN), such as `arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`LoadBalancerArns`

A list of Amazon Resource Names (ARNs) of the load balancers that route traffic to this target group, such as `[ "arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188" ]`.

`TargetGroupFullName`

The full name of the target group, such as `targetgroup/my-target-group/cbf133c568e0d028`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Examples

Create a Target Group with EC2 Instances as Targets

The following examples creates a target group that includes the `Instance1` and `Instance2` EC2 instances as targets. The instances must respond with a `200` status code to pass health check requests.

JSON

```
"TargetGroup" : {
  "Type" : "AWS::ElasticLoadBalancingV2::TargetGroup",
  "Properties" : {
    "HealthCheckIntervalSeconds": 30,
    "HealthCheckProtocol": "HTTPS",
    "HealthCheckTimeoutSeconds": 10,
    "HealthyThresholdCount": 4,
    "Matcher" : {
      "HttpCode" : "200"
    },
    "Name": "MyTargets",
    "Port": 10,
    "Protocol": "HTTPS",
    "TargetGroupAttributes": [{
      "Key": "deregistration_delay.timeout_seconds",
      "Value": "20"
    }],
    "Targets": [
      { "Id": {"Ref" : "Instance1"}, "Port": 80 },
      { "Id": {"Ref" : "Instance2"}, "Port": 80 }
    ],
```

```
    "UnhealthyThresholdCount": 3,
    "VpcId": {"Ref" : "VPC"},
    "Tags" : [
      { "Key" : "key", "Value" : "value" },
      { "Key" : "key2", "Value" : "value2" }
    ]
  }
}
```

YAML

```
TargetGroup:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
    HealthCheckIntervalSeconds: 30
    HealthCheckProtocol: HTTPS
    HealthCheckTimeoutSeconds: 10
    HealthyThresholdCount: 4
    Matcher:
      HttpCode: '200'
    Name: MyTargets
    Port: 10
    Protocol: HTTPS
    TargetGroupAttributes:
    - Key: deregistration_delay.timeout_seconds
      Value: '20'
    Targets:
    - Id:
        Ref: Instance1
      Port: 80
    - Id:
        Ref: Instance2
      Port: 80
    UnhealthyThresholdCount: 3
    VpcId:
      Ref: VPC
    Tags:
    - Key: key
      Value: value
    - Key: key2
      Value: value2
```

Relate an Elastic Load Balancing Load Balancer to an Elastic Load Balancing Target Group

The following example creates an Elastic Load Balancing listener and associates it with a target group and a load balancer.

JSON

```
"ALBListener" : {
  "Type" : "AWS::ElasticLoadBalancingV2::Listener",
  "Properties" : {
    "DefaultActions" : [{
      "Type" : "forward",
      "TargetGroupArn" : { "Ref" : "ALBTargetGroup" }
    }],
    "LoadBalancerArn" : { "Ref" : "ApplicationLoadBalancer" },
    "Port" : "80",
    "Protocol" : "HTTP"
  }
},
```

```
"ApplicationLoadBalancer" : {
  "Type" : "AWS::ElasticLoadBalancingV2::LoadBalancer",
  "Properties" : {
    "Scheme" : "internet-facing",
    "Subnets" : [ {"Ref" : "PublicSubnetAz1"}, {"Ref" : "PublicSubnetAz2"}],
    "SecurityGroups" : [{"Ref": "ALBSecurityGroup"}]
  }
},
"ALBTargetGroup" : {
  "Type" : "AWS::ElasticLoadBalancingV2::TargetGroup",
  "Properties" : {
    "HealthCheckIntervalSeconds" : 60,
    "UnhealthyThresholdCount" : 10,
    "HealthCheckPath" : "/",
    "Name" : "MyTargetGroup",
    "Port" : 80,
    "Protocol" : "HTTP",
    "VpcId" : { "Ref": "MyVpc" }
  }
}
```

YAML

```
ALBListener:
  Type: AWS::ElasticLoadBalancingV2::Listener
  Properties:
    DefaultActions:
      Type: forward
      TargetGroupArn:
        Ref: ALBTargetGroup
      LoadBalancerArn:
        Ref: ApplicationLoadBalancer
      Port: 80
      Protocol: HTTP
ApplicationLoadBalancer:
  Type: AWS::ElasticLoadBalancingV2::LoadBalancer
  Properties:
    Scheme: internet-facing
    Subnets:
      Ref: PublicSubnetAz1
      Ref: PublicSubnetAz2
    SecurityGroups:
      Ref: ALBSecurityGroup
ALBTargetGroup:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
    HealthCheckIntervalSeconds: 60
    UnhealthyThresholdCount: 10
    HealthCheckPath: /
    Name: MyTargetGroup
    Port: 80
    Protocol: HTTP
    VpcId:
      Ref: MyVpc
```

AWS::Elasticsearch::Domain

The `AWS::Elasticsearch::Domain` resource creates an Amazon Elasticsearch Service (Amazon ES) domain that encapsulates the Amazon ES engine instances. For more information, see CreateElasticsearchDomain in the *Amazon Elasticsearch Service Developer Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Elasticsearch::Domain",
  "Properties" : {
    "AccessPolicies" : JSON object,
    "AdvancedOptions" : Advanced Options,
    "DomainName" : String,
    "EBSOptions" : EBS Options,
    "ElasticsearchClusterConfig" : Elasticsearch Cluster Config,
    "ElasticsearchVersion" : String,
    "SnapshotOptions" : Snapshot Options,
    "Tags" : [ Resource Tag, ... ]
  }
}
```

YAML

```
Type: "AWS::Elasticsearch::Domain"
Properties:
  AccessPolicies: JSON object
  AdvancedOptions:
    Advanced Options
  DomainName: String
  EBSOptions:
    EBS Options
  ElasticsearchClusterConfig:
    Elasticsearch Cluster Config
  ElasticsearchVersion: String
  SnapshotOptions:
    Snapshot Options
  Tags:
    - Resource Tag
```

Properties

`AccessPolicies`

An AWS Identity and Access Management (IAM) policy document that specifies who can access the Amazon ES domain and their permissions. For more information, see Configuring Access Policies in the *Amazon Elasticsearch Service Developer Guide*.

Required: No

Type: JSON object

Update requires: No interruption (p. 90)

`AdvancedOptions`

Additional options to specify for the Amazon ES domain. For more information, see Configuring Advanced Options in the *Amazon Elasticsearch Service Developer Guide*.

Required: No

Type: A JSON object consisting of a string key-value pair, such as:

```
{
  "rest.action.multi.allow_explicit_index": "true"
}
```

Update requires: Replacement (p. 90)

`DomainName`

A name for the Amazon ES domain. For valid values, see the DomainName data type in the *Amazon Elasticsearch Service Developer Guide*.

If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the domain name. For more information, see Name Type (p. 1217).

> **Important**
> If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

`EBSOptions`

The configurations of Amazon Elastic Block Store (Amazon EBS) volumes that are attached to data nodes in the Amazon ES domain. For more information, see Configuring EBS-based Storage in the *Amazon Elasticsearch Service Developer Guide*.

Required: No

Type: Amazon Elasticsearch Service Domain EBSOptions (p. 1166)

Update requires: No interruption (p. 90)

`ElasticsearchClusterConfig`

The cluster configuration for the Amazon ES domain. You can specify options such as the instance type and the number of instances. For more information, see Configuring Amazon ES Domains in the *Amazon Elasticsearch Service Developer Guide*.

Required: No

Type: Amazon Elasticsearch Service Domain ElasticsearchClusterConfig (p. 1167)

Update requires: No interruption (p. 90)

`ElasticsearchVersion`

The version of Elasticsearch to use, such as `2.3`. For information about the versions that Amazon ES supports, see the `Elasticsearch-Version` parameter for the CreateElasticsearchDomain action in the *Amazon Elasticsearch Service Developer Guide*.

Required: No

Type: String

Update requires: Replacement (p. 90)

`SnapshotOptions`

The automated snapshot configuration for the Amazon ES domain indices.

Required: No

Type: Amazon Elasticsearch Service Domain SnapshotOptions (p. 1168)

Update requires: No interruption (p. 90)

`Tags`

An arbitrary set of tags (key–value pairs) to associate with the Amazon ES domain.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name, such as `mystack-elasticsea-abc1d2efg3h4`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`DomainArn`

The Amazon Resource Name (ARN) of the domain, such as `arn:aws:es:us-west-2:123456789012:domain/mystack-elasti-1ab2cdefghij`.

`DomainEndpoint`

The domain-specific endpoint that is used to submit index, search, and data upload requests to an Amazon ES domain, such as `search-mystack-elasti-1ab2cdefghij-ab1c2deckoyb3hofw7wpqa3cm.us-west-2.es.amazonaws.com`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Examples

The following examples create an Amazon ES domain that contains two data nodes and three master nodes. Automated snapshots of the indices are taken daily between midnight and 1:00 AM (UTC). The access policy permits the IAM user `es-user` to take all Amazon ES actions on the domain, such as `es:UpdateElasticsearchDomainConfig`.

JSON

```
"ElasticsearchDomain": {
  "Type": "AWS::Elasticsearch::Domain",
  "Properties": {
    "DomainName": "test",
    "ElasticsearchClusterConfig": {
      "DedicatedMasterEnabled": "true",
      "InstanceCount": "2",
      "ZoneAwarenessEnabled": "true",
      "InstanceType": "m3.medium.elasticsearch",
      "DedicatedMasterType": "m3.medium.elasticsearch",
      "DedicatedMasterCount": "3"
    },
    "EBSOptions": {
      "EBSEnabled": true,
      "Iops": 0,
      "VolumeSize": 20,
      "VolumeType": "gp2"
    },
    "SnapshotOptions": {
      "AutomatedSnapshotStartHour": "0"
    },
    "AccessPolicies": {
      "Version": "2012-10-17",
      "Statement": [{
        "Effect": "Allow",
        "Principal": {
          "AWS": "arn:aws:iam::123456789012:user/es-user"
        },
        "Action": "es:*",
        "Resource": "arn:aws:es:us-east-1:123456789012:domain/test/*"
      }]
    },
    "AdvancedOptions": {
      "rest.action.multi.allow_explicit_index": "true"
    }
  }
}
```

YAML

```
ElasticsearchDomain:
  Type: "AWS::Elasticsearch::Domain"
  Properties:
    DomainName: "test"
    ElasticsearchClusterConfig:
      DedicatedMasterEnabled: "true"
      InstanceCount: "2"
      ZoneAwarenessEnabled: "true"
      InstanceType: "m3.medium.elasticsearch"
      DedicatedMasterType: "m3.medium.elasticsearch"
      DedicatedMasterCount: "3"
    EBSOptions:
      EBSEnabled: true
      Iops: 0
      VolumeSize: 20
      VolumeType: "gp2"
    SnapshotOptions:
      AutomatedSnapshotStartHour: "0"
    AccessPolicies:
      Version: "2012-10-17"
      Statement:
```

```
        -
          Effect: "Allow"
          Principal:
            AWS: "arn:aws:iam::123456789012:user/es-user"
          Action: "es:*"
          Resource: "arn:aws:es:us-east-1:846973539254:domain/test/*"
    AdvancedOptions:
      rest.action.multi.allow_explicit_index: "true"
```

AWS::EMR::Cluster

The `AWS::EMR::Cluster` resource creates an Amazon EMR (Amazon EMR) cluster, which is a collection of EC2 instances on which you can run big data frameworks to process and analyze vast amounts of data. For more information, see Plan an Amazon EMR Cluster in the *Amazon EMR Management Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::EMR::Cluster",
  "Properties" : {
    "AdditionalInfo" : JSON object,
    "Applications" : [ Applications, ... ],
    "BootstrapActions" [ Bootstrap Actions, ... ],
    "Configurations" : [ Configurations, ... ],
    "Instances" : JobFlowInstancesConfig,
    "JobFlowRole" : String,
    "LogUri" : String,
    "Name" : String,
    "ReleaseLabel" : String,
    "ServiceRole" : String,
    "Tags" : [ Resource Tag, ... ],
    "VisibleToAllUsers" : Boolean
  }
}
```

YAML

```
Type: "AWS::EMR::Cluster"
Properties:
  AdditionalInfo: JSON object
  Applications:
    - Applications
  BootstrapActions:
    - Bootstrap Actions
  Configurations:
    - Configurations
  Instances:
    JobFlowInstancesConfig
```

```
  JobFlowRole: String
  LogUri: String
  Name: String
  ReleaseLabel: String
  ServiceRole: String
  Tags:
    - Resource Tag
  VisibleToAllUsers: Boolean
```

Properties

`AdditionalInfo`

Additional features that you want to select.

Required: No

Type: JSON object

Update requires: Replacement (p. 90)

`Applications`

The software applications to deploy on the cluster, and the arguments that Amazon EMR passes to those applications.

Required: No

Type: List of Amazon EMR Cluster Application (p. 1169)

Update requires: Replacement (p. 90)

`BootstrapActions`

A list of bootstrap actions that Amazon EMR runs before starting applications on the cluster.

Required: No

Type: List of Amazon EMR Cluster BootstrapActionConfig (p. 1170)

Update requires: Replacement (p. 90)

`Configurations`

The software configuration of the Amazon EMR cluster.

Required: No

Type: List of Amazon EMR Cluster Configuration (p. 1171)

Update requires: Replacement (p. 90)

`Instances`

Configures the EC2 instances that will run jobs in the Amazon EMR cluster.

Required: Yes

Type: Amazon EMR Cluster JobFlowInstancesConfig (p. 1172)

Update requires: Replacement (p. 90)

`JobFlowRole`

Also called *instance profile* and *EC2 role*. Accepts an instance profile associated with the role that you want to use. All EC2 instances in the cluster assume this role.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`LogUri`

An S3 bucket location to which Amazon EMR writes logs files from a job flow. If you don't specify a value, Amazon EMR doesn't write any log files.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Name`

A name for the Amazon EMR cluster.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`ReleaseLabel`

The Amazon EMR software release label. A release is a set of software applications and components that you can install and configure on an Amazon EMR cluster. For more information, see About Amazon EMR Releases in the *Amazon EMR Release Guide*.

Currently, AWS CloudFormation supports only Amazon EMR 4.0 and later software releases.

Required: Conditional. If you specify the `Applications` property, you must specify this property.

Type: String

Update requires: Replacement (p. 90)

`ServiceRole`

The IAM role that Amazon EMR assumes to access AWS resources on your behalf. For more information, see Configure IAM Roles for Amazon EMR in the *Amazon EMR Management Guide*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Tags`

An arbitrary set of tags (key–value pairs) to help you identify the Amazon EMR cluster.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90).

`VisibleToAllUsers`

Indicates whether the instances in the cluster are visible to all IAM users in the AWS account. If you specify `true`, all IAM users can view and (if they have permissions) manage the instances. If you specify `false`, only the IAM user that created the cluster can view and manage it. By default, AWS CloudFormation sets this property to `false`.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the cluster ID, such as `j-1ABCD123AB1A`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`MasterPublicDNS`

The public DNS name of the master node (instance), such as `ec2-12-123-123-123.us-west-2.compute.amazonaws.com`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Examples

Cluster With Two Core Nodes

The following example creates an Amazon EMR cluster with one master node and two core nodes. The specified IAM roles are the default roles provided by Amazon EMR. This example also assumes the cluster is launched in a region with a default VPC and subnet. If you don't have these, use the `Ec2SubnetId` (p. 1172) property to specify the VPC and subnet for the cluster. Otherwise, AWS CloudFormation can't launch the cluster and returns the following status message: `ElasticMapReduce Cluster failed to stabilize`.

JSON

```
"TestCluster": {
  "Type": "AWS::EMR::Cluster",
  "Properties": {
    "Instances": {
      "MasterInstanceGroup": {
        "InstanceCount": 1,
        "InstanceType": "m3.xlarge",
        "Market": "ON_DEMAND",
        "Name": "Master"
      },
      "CoreInstanceGroup": {
        "InstanceCount": 2,
        "InstanceType": "m3.xlarge",
        "Market": "ON_DEMAND",
        "Name": "Core"
      },
      "TerminationProtected" : true
    },
    "Name": "TestCluster",
```

```
    "JobFlowRole" : "EMR_EC2_DefaultRole",
    "ServiceRole" : "EMR_DefaultRole",
    "ReleaseLabel" : "emr-4.2.0",
    "Tags": [
      {
        "Key": "IsTest",
        "Value": "True"
      }
    ]
  }
}
```

YAML

```
TestCluster:
  Type: "AWS::EMR::Cluster"
  Properties:
    Instances:
      MasterInstanceGroup:
        InstanceCount: 1
        InstanceType: "m3.xlarge"
        Market: "ON_DEMAND"
        Name: "Master"
      CoreInstanceGroup:
        InstanceCount: 2
        InstanceType: "m3.xlarge"
        Market: "ON_DEMAND"
        Name: "Core"
      TerminationProtected: true
    Name: "TestCluster"
    JobFlowRole: "EMR_EC2_DefaultRole"
    ServiceRole: "EMR_DefaultRole"
    ReleaseLabel: "emr-4.2.0"
    Tags:
      -
        Key: "IsTest"
        Value: "True"
```

Cluster With a Bootstrap Action

The following example creates an Amazon EMR cluster with a bootstrap action.

JSON

```
"TestCluster": {
  "Type": "AWS::EMR::Cluster",
  "Properties": {
    "BootstrapActions": [{
      "Name": "SomeBootStrapAction",
      "ScriptBootstrapAction": {
        "Path": "/path/to/s3"
      }
    }],
    "Instances": {
      "MasterInstanceGroup": {
        "InstanceCount": 1,
        "InstanceType": "m3.xlarge",
        "Market": "ON_DEMAND",
        "Name": "Master"
      },
      "CoreInstanceGroup": {
        "InstanceCount": 2,
        "InstanceType": "m3.xlarge",
```

```
        "Market": "ON_DEMAND",
        "Name": "Core"
      },
      "TerminationProtected" : true
    },
    "Name": "TestCluster",
    "JobFlowRole" : "EMR_EC2_DefaultRole",
    "ServiceRole" : "EMR_DefaultRole",
    "ReleaseLabel" : "emr-4.2.0",
    "Tags": [
      {
        "Key": "IsTest",
        "Value": "True"
      }
    ]
  }
}
```

YAML

```
TestCluster:
  Type: "AWS::EMR::Cluster"
  Properties:
    BootstrapActions:
      -
        Name: "SomeBootStrapAction"
        ScriptBootstrapAction:
          Path: "/path/to/s3"
    Instances:
      MasterInstanceGroup:
        InstanceCount: 1
        InstanceType: "m3.xlarge"
        Market: "ON_DEMAND"
        Name: "Master"
      CoreInstanceGroup:
        InstanceCount: 2
        InstanceType: "m3.xlarge"
        Market: "ON_DEMAND"
        Name: "Core"
      TerminationProtected: true
    Name: "TestCluster"
    JobFlowRole: "EMR_EC2_DefaultRole"
    ServiceRole: "EMR_DefaultRole"
    ReleaseLabel: "emr-4.2.0"
    Tags:
      -
        Key: "IsTest"
        Value: "True"
```

AWS::EMR::InstanceGroupConfig

The `AWS::EMR::InstanceGroupConfig` resource configures a task instance group for an Amazon EMR (Amazon EMR) cluster.

> **Note**
> You can't delete an instance group. If you remove an instance group, AWS CloudFormation sets the instance count to zero (`0`).

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::EMR::InstanceGroupConfig",
  "Properties" : {
    "BidPrice" : String,
    "Configurations" : [ Configuration, ... ],
    "EbsConfiguration" : EBSConfiguration,
    "InstanceCount" : Integer,
    "InstanceRole" : String,
    "InstanceType" : String,
    "JobFlowId": String,
    "Market" : String,
    "Name" : String
  }
}
```

YAML

```
Type: "AWS::EMR::InstanceGroupConfig"
Properties:
  BidPrice: String
  Configurations:
    - Configuration
  EbsConfiguration" : EBSConfiguration
  InstanceCount" : Integer
  InstanceRole" : String
  InstanceType" : String
  JobFlowId": String
  Market" : String
  Name" : String
```

Properties

`BidPrice`

The bid price in USD for each EC2 instance in the instance group when launching instances (nodes) as Spot Instances.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Configurations`

A list of configurations to apply to this instance group. For more information see, Configuring Applications in the *Amazon EMR Release Guide*.

Required: No

Type: List of Amazon EMR Cluster Configuration (p. 1171)

Update requires: Replacement (p. 90)

`EbsConfiguration`

Configures Amazon Elastic Block Store (Amazon EBS) storage volumes to attach to your instances.

Required: No

Type: Amazon EMR EbsConfiguration (p. 1177)

Update requires: Replacement (p. 90)

`InstanceCount`

The number of instances to launch in the instance group.

Required: Yes

Type: Integer

Update requires: No interruption (p. 90)

`InstanceRole`

The role of the servers in the Amazon EMR cluster, such as `TASK`. For more information, see Instance Groups in the *Amazon EMR Management Guide*.

> **Note**
> Currently, the only valid value is `TASK`. You configure the master and core instance groups as part of the AWS::EMR::Cluster (p. 750) resource.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`InstanceType`

The EC2 instance type for all instances in the instance group. For more information, see Instance Configurations in the *Amazon EMR Management Guide*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`JobFlowId`

The ID of an Amazon EMR cluster that you want to associate this instance group with.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Market`

The type of marketplace from which your instances are provisioned into this group, either `ON_DEMAND` or `SPOT`. For more information, see Amazon EC2 Purchasing Options.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Name`

A name for the instance group.

Required: No

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the instance group ID, such as `ig-ABC12DEF3456`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example adds a task instance group to the `TestCluster` cluster. The instance group contains two m3.xlarge instances.

JSON

```
"TestInstanceGroupConfig": {
  "Type": "AWS::EMR::InstanceGroupConfig",
  "Properties": {
    "InstanceCount": 2,
    "InstanceType": "m3.xlarge",
    "InstanceRole": "TASK",
    "Market": "ON_DEMAND",
    "Name": "cfnTask2",
    "JobFlowId": {
      "Ref": "cluster"
    }
  }
}
```

YAML

```
TestInstanceGroupConfig:
  Type: "AWS::EMR::InstanceGroupConfig"
  Properties:
    InstanceCount: 2
    InstanceType: "m3.xlarge"
    InstanceRole: "TASK"
    Market: "ON_DEMAND"
    Name: "cfnTask2"
    JobFlowId:
      Ref: "cluster"
```

AWS::EMR::Step

The `AWS::EMR::Step` resource creates a unit of work (a job flow step) that you submit to an Amazon EMR (Amazon EMR) cluster. The job flow step contains instructions for processing data on the cluster.

Note
You can't delete work flow steps. During a stack update, if you remove a step, AWS CloudFormation takes no action.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::EMR::Step",
  "Properties" : {
    "ActionOnFailure" : String,
    "HadoopJarStep" : HadoopJarStepConfig,
    "JobFlowId" : String,
    "Name" : String
  }
}
```

YAML

```
Type: "AWS::EMR::Step"
Properties:
  ActionOnFailure: String
  HadoopJarStep:
    HadoopJarStepConfig
  JobFlowId: String
  Name: String
```

Properties

`ActionOnFailure`

The action to take if the job flow step fails. Currently, AWS CloudFormation supports `CONTINUE` and `CANCEL_AND_WAIT`. For more information, see Managing Cluster Termination in the *Amazon EMR Management Guide*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`HadoopJarStep`

The JAR file that includes the main function that Amazon EMR executes.

Required: Yes

Type: Amazon EMR Step HadoopJarStepConfig (p. 1179)

Update requires: Replacement (p. 90)

`JobFlowId`

The ID of a cluster in which you want to run this job flow step.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Name`

A name for the job flow step.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the step ID, such as `s-1A2BC3D4EFG56`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a step that submits work to the `TestCluster` cluster. The step runs the `pi` program in the `hadoop-mapreduce-examples-2.6.0.jar` file with 5 maps and 10 samples, specified in the `Args` property.

JSON

```
"TestStep": {
  "Type": "AWS::EMR::Step",
  "Properties": {
    "ActionOnFailure": "CONTINUE",
    "HadoopJarStep": {
      "Args": [
        "5",
        "10"
      ],
      "Jar": "s3://emr-cfn-test/hadoop-mapreduce-examples-2.6.0.jar",
      "MainClass": "pi"
    },
    "Name": "TestStep",
    "JobFlowId": {
      "Ref": "TestCluster"
    }
  }
}
```

YAML

```
TestStep:
  Type: "AWS::EMR::Step"
```

```
  Properties:
    ActionOnFailure: "CONTINUE"
    HadoopJarStep:
      Args:
        - "5"
        - "10"
      Jar: "s3://emr-cfn-test/hadoop-mapreduce-examples-2.6.0.jar"
      MainClass: "pi"
    Name: "TestStep"
    JobFlowId:
      Ref: "TestCluster"
```

AWS::Events::Rule

The `AWS::Events::Rule` resource creates a rule that matches incoming Amazon CloudWatch Events (CloudWatch Events) events and routes them to one or more targets for processing. For more information, see Using CloudWatch Events in the *Amazon CloudWatch User Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Events::Rule",
  "Properties" : {
    "Description" : String,
    "EventPattern" : JSON object,
    "Name" : String,
    "RoleArn" : String,
    "ScheduleExpression" : String,
    "State" : String,
    "Targets" : [ Target (p. 1060), ... ]
  }
}
```

YAML

```
Type: "AWS::Events::Rule"
Properties:
  Description: String
  EventPattern: JSON object
  Name: String
  RoleArn: String
  ScheduleExpression: String
  State: String
  Targets:
    - Target (p. 1060)
```

Properties

`Description`

A description of the rule's purpose.

Required: No

Type: String

Update requires: No interruption (p. 90)

`EventPattern`

Describes which events CloudWatch Events routes to the specified target. These routed events are matched events. For more information, see Events and Event Patterns in the *Amazon CloudWatch User Guide*.

Required: Conditional. You must specify this property, the `ScheduleExpression` property, or both.

Type: JSON object

Update requires: No interruption (p. 90)

`Name`

A name for the rule. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the rule name. For more information, see Name Type (p. 1217).

> **Important**
> If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

`RoleArn`

The Amazon Resource Name (ARN) of the AWS Identity and Access Management (IAM) role that grants CloudWatch Events permission to make calls to target services, such as AWS Lambda (Lambda) or Amazon Kinesis streams.

Required: No

Type: String

Update requires: No interruption (p. 90)

`ScheduleExpression`

The schedule or rate (frequency) that determines when CloudWatch Events runs the rule. For more information, see Schedule Expression Syntax for Rules in the *Amazon CloudWatch User Guide*.

Required: Conditional. You must specify this property, the `EventPattern` property, or both.

Type: String

Update requires: No interruption (p. 90)

`State`

Indicates whether the rule is enabled. For valid values, see the `State` parameter for the PutRule action in the *Amazon CloudWatch Events API Reference*.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Targets`

The resources, such as Lambda functions or Amazon Kinesis streams, that CloudWatch Events routes events to and invokes when the rule is triggered. For information about valid targets, see the PutTargets action in the *Amazon CloudWatch Events API Reference*.

Note
Creating rules with built-in targets is supported only in the AWS Management Console.

Required: No

Type: List of Amazon CloudWatch Events Rule Target (p. 1060)

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the event rule ID, such as `mystack-ScheduledRule-ABCDEFGHIJK`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`Arn`

The event rule Amazon Resource Name (ARN), such as `arn:aws:events:us-east-1:123456789012:rule/example`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Examples

Regularly Invoke Lambda Function

The following example creates a rule that invokes the specified Lambda function every 10 minutes. The `PermissionForEventsToInvokeLambda` resource grants CloudWatch Events permission to invoke the associated function.

JSON

```
"ScheduledRule": {
  "Type": "AWS::Events::Rule",
  "Properties": {
    "Description": "ScheduledRule",
    "ScheduleExpression": "rate(10 minutes)",
    "State": "ENABLED",
    "Targets": [{
      "Arn": { "Fn::GetAtt": ["LambdaFunction", "Arn"] },
```

```
      "Id": "TargetFunctionV1"
    }]
  }
},
"PermissionForEventsToInvokeLambda": {
  "Type": "AWS::Lambda::Permission",
  "Properties": {
    "FunctionName": { "Ref": "LambdaFunction" },
    "Action": "lambda:InvokeFunction",
    "Principal": "events.amazonaws.com",
    "SourceArn": { "Fn::GetAtt": ["ScheduledRule", "Arn"] }
  }
}
```

YAML

```
ScheduledRule:
  Type: "AWS::Events::Rule"
  Properties:
    Description: "ScheduledRule"
    ScheduleExpression: "rate(10 minutes)"
    State: "ENABLED"
    Targets:
      -
        Arn:
          Fn::GetAtt:
            - "LambdaFunction"
            - "Arn"
        Id: "TargetFunctionV1"
PermissionForEventsToInvokeLambda:
  Type: "AWS::Lambda::Permission"
  Properties:
    FunctionName:
      Ref: "LambdaFunction"
    Action: "lambda:InvokeFunction"
    Principal: "events.amazonaws.com"
    SourceArn:
      Fn::GetAtt:
        - "ScheduledRule"
        - "Arn"
```

Invoke Lambda Function in Response to an Event

The following example creates a rule that invokes the specified Lambda function when any EC2 instance's state changes to `stopping`.

JSON

```
"EventRule": {
  "Type": "AWS::Events::Rule",
  "Properties": {
    "Description": "EventRule",
    "EventPattern": {
      "source": [
        "aws.ec2"
      ],
      "detail-type": [
        "EC2 Instance State-change Notification"
      ],
      "detail": {
        "state": [
          "stopping"
        ]
```

```
      }
    },
    "State": "ENABLED",
    "Targets": [{
      "Arn": { "Fn::GetAtt": ["LambdaFunction", "Arn"] },
      "Id": "TargetFunctionV1"
    }]
  }
},
"PermissionForEventsToInvokeLambda": {
  "Type": "AWS::Lambda::Permission",
  "Properties": {
    "FunctionName": { "Ref": "LambdaFunction" },
    "Action": "lambda:InvokeFunction",
    "Principal": "events.amazonaws.com",
    "SourceArn": { "Fn::GetAtt": ["EventRule", "Arn"] }
  }
}
```

YAML

```
EventRule:
  Type: "AWS::Events::Rule"
  Properties:
    Description: "EventRule"
    EventPattern:
      source:
        - "aws.ec2"
      detail-type:
        - "EC2 Instance State-change Notification"
      detail:
        state:
          - "stopping"
    State: "ENABLED"
    Targets:
      -
        Arn:
          Fn::GetAtt:
            - "LambdaFunction"
            - "Arn"
        Id: "TargetFunctionV1"
PermissionForEventsToInvokeLambda:
  Type: "AWS::Lambda::Permission"
  Properties:
    FunctionName:
      Ref: "LambdaFunction"
    Action: "lambda:InvokeFunction"
    Principal: "events.amazonaws.com"
    SourceArn:
      Fn::GetAtt:
        - "EventRule"
        - "Arn"
```

Notify a Topic in Response to a Log Entry

The following example creates a rule that notifies an Amazon Simple Notification Service topic if an AWS CloudTrail log entry contains a call by the `Root` user.

JSON

```
"OpsEventRule": {
  "Type": "AWS::Events::Rule",
  "Properties": {
```

```
    "Description": "EventRule",
    "EventPattern": {
      "detail-type": [ "AWS API Call via CloudTrail" ],
      "detail": {
        "userIdentity": {
          "type": [ "Root" ]
        }
      }
    },
    "State": "ENABLED",
    "Targets": [
      {
        "Arn": { "Ref": "MySNSTopic" },
        "Id": "OpsTopic"
      }
    ]
  }
}
```

YAML

```
OpsEventRule:
  Type: "AWS::Events::Rule"
  Properties:
    Description: "EventRule"
    EventPattern:
      detail-type:
        - "AWS API Call via CloudTrail"
      detail:
        userIdentity:
          type:
            - "Root"
    State: "ENABLED"
    Targets:
      -
        Arn:
          Ref: "MySNSTopic"
        Id: "OpsTopic"
```

AWS::GameLift::Alias

The `AWS::GameLift::Alias` resource creates an alias for an Amazon GameLift (GameLift) fleet, which you can use to anonymize your fleet. You can reference the alias instead of a specific fleet when you create game sessions. For more information, see the CreateAlias action in the *Amazon GameLift API Reference*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
```

```
  "Type" : "AWS::GameLift::Alias",
  "Properties" : {
    "Name" : String,
    "Description" : String,
    "RoutingStrategy" : RoutingStrategy (p. 1181)
  }
}
```

YAML

```
Type: "AWS::GameLift::Alias"
Properties:
  Name: String
  Description: String
  RoutingStrategy:
    RoutingStrategy (p. 1181)
```

Properties

`Description`

Information that helps you identify the purpose of this alias.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Name`

An identifier to associate with this alias. Alias names don't need to be unique.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`RoutingStrategy`

A routing configuration that specifies where traffic is directed for this alias, such as to a fleet or to a message.

Required: Yes

Type: Amazon GameLift Alias RoutingStrategy (p. 1181)

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the alias ID, such as `myalias-a01234b56-7890-1de2-f345-g67h8i901j2k`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a terminal alias named `TerminalAlias` with a generic terminal message.

JSON

```
"AliasResource": {
  "Type": "AWS::GameLift::Alias",
  "Properties": {
    "Name": "TerminalAlias",
    "Description": "A terminal alias",
    "RoutingStrategy": {
      "Type": "TERMINAL",
      "Message": "Terminal routing strategy message"
    }
  }
}
```

YAML

```
AliasResource:
  Type: "AWS::GameLift::Alias"
  Properties:
    Name: "TerminalAlias"
    Description: "A terminal alias"
    RoutingStrategy:
      Type: "TERMINAL"
      Message: "Terminal routing strategy message"
```

AWS::GameLift::Build

The `AWS::GameLift::Build` resource creates a build that includes all of the components to run your game server in an Amazon GameLift (GameLift) fleet.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::GameLift::Build",
  "Properties" : {
    "Name" : String,
    "StorageLocation" : StorageLocation (p. 1182),
    "Version" : String
  }
}
```

YAML

```
Type: "AWS::GameLift::Build"
Properties:
  Name: String
  StorageLocation:
    StorageLocation (p. 1182)
  Version: String
```

Properties

`Name`

An identifier to associate with this build. Build names don't need to be unique.

Required: No

Type: String

Update requires: No interruption (p. 90)

`StorageLocation`

The Amazon Simple Storage Service (Amazon S3) location where your build package files are located.

Required: No, but we recommend that you specify a location. If you don't specify this property, you must manually upload your build package files to GameLift.

Type: Amazon GameLift Build StorageLocation (p. 1182)

Update requires: Replacement (p. 90)

`Version`

A version to associate with this build. Version is useful if you want to track updates to your build package files. Versions don't need to be unique.

Required: No

Type: String

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the build ID, such as `mybuild-a01234b56-7890-1de2-f345-g67h8i901j2k`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a GameLift build named `MyGameServerBuild`. The build package is located in an S3 bucket, specified by the `S3Bucket` and `S3Key` input parameters. The example also creates the AWS Identity and Access Management (IAM) role that GameLift assumes so that it has permissions to download the build package files.

JSON

```
"BuildResource": {
  "Type": "AWS::GameLift::Build",
  "Properties": {
    "Name": "MyGameServerBuild",
    "Version": "v15",
    "StorageLocation": {
      "Bucket": "mybucket",
      "Key": "buildpackagefiles/",
      "RoleArn": { "Fn::GetAtt": [ "IAMRole", "Arn" ] }
    }
  }
},
"IAMRole": {
  "Type": "AWS::IAM::Role",
  "Properties": {
    "AssumeRolePolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Principal": { "Service": [ "gamelift.amazonaws.com" ] },
          "Action": [ "sts:AssumeRole" ]
        }
      ]
    },
    "Path": "/",
    "Policies": [
      {
        "PolicyName": "gamelift-s3-access-policy",
        "PolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Action": [ "s3:GetObject" ],
              "Resource": [ "arn:aws:s3:::mybucket/*" ]
            }
          ]
        }
      }
    ]
  }
}
```

YAML

```
BuildResource:
  Type: "AWS::GameLift::Build"
  Properties:
    Name: "MyGameServerBuild"
    Version: "v15"
    StorageLocation:
      Bucket: "mybucket"
      Key: "buildpackagefiles/"
      RoleArn:
        Fn::GetAtt:
          - "IAMRole"
          - "Arn"
IAMRole:
  Type: "AWS::IAM::Role"
  Properties:
```

```
    AssumeRolePolicyDocument:
      Version: "2012-10-17"
      Statement:
        -
          Effect: "Allow"
          Principal:
            Service:
              - "gamelift.amazonaws.com"
          Action:
            - "sts:AssumeRole"
    Path: "/"
    Policies:
      -
        PolicyName: "gamelift-s3-access-policy"
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
            -
              Effect: "Allow"
              Action:
                - "s3:GetObject"
              Resource:
                - "arn:aws:s3:::mybucket/*"
```

AWS::GameLift::Fleet

The `AWS::GameLift::Fleet` resource creates an Amazon GameLift (GameLift) fleet to host game servers. A fleet is a set of EC2 instances, each of which is a host in the fleet. For more information, see the CreateFleet action in the *Amazon GameLift API Reference*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::GameLift::Fleet",
  "Properties" : {
    "BuildId" : String,
    "Description" : String,
    "DesiredEC2Instances" : Integer,
    "EC2InboundPermissions" : [ EC2InboundPermission (p. 1183), ... ],
    "EC2InstanceType" : String,
    "LogPaths" : [ String, ... ],
    "MaxSize" : Integer,
    "MinSize" : Integer,
    "Name" : String,
    "ServerLaunchParameters" : String,
    "ServerLaunchPath" : String
  }
}
```

YAML

```
Type: "AWS::GameLift::Fleet"
Properties:
  BuildId: String
  Description: String
  DesiredEC2Instances: Integer
  EC2InboundPermissions:
    - EC2InboundPermission (p. 1183)
  EC2InstanceType: String
  LogPaths:
    [ String, ... ]
  MaxSize: Integer
  MinSize: Integer
  Name: String
  ServerLaunchParameters: String
  ServerLaunchPath: String
```

Properties

`BuildId`

The unique identifier for the build that you want to use with this fleet.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Description`

Information that helps you identify the purpose of this fleet.

Required: No

Type: String

Update requires: No interruption (p. 90)

`DesiredEC2Instances`

The number of EC2 instances that you want in this fleet.

Required: Yes

Type: Integer

Update requires: No interruption (p. 90)

`EC2InboundPermissions`

The incoming traffic, expressed as IP ranges and port numbers, that is permitted to access the game server. If you don't specify values, no traffic is permitted to your game servers.

Required: No

Type: List of Amazon GameLift Fleet EC2InboundPermission (p. 1183)

Update requires: No interruption (p. 90)

`EC2InstanceType`

The type of EC2 instances that the fleet uses. EC2 instance types define the CPU, memory, storage, and networking capacity of the fleet's hosts. For more information about the instance types that are supported by GameLift, see the EC2InstanceType parameter in the *Amazon GameLift API Reference*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`LogPaths`

The path to game-session log files that are generated by your game server, with the slashes (`\`) escaped. After a game session has been terminated, GameLift captures and stores the logs in an S3 bucket.

Required: No

Type: List of strings

Update requires: Replacement (p. 90)

`MaxSize`

The maximum number of EC2 instances that you want to allow in this fleet. By default, AWS CloudFormation, sets this property to `1`.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`MinSize`

The minimum number of EC2 instances that you want to allow in this fleet. By default, AWS CloudFormation, sets this property to `0`.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`Name`

An identifier to associate with this fleet. Fleet names don't need to be unique.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`ServerLaunchParameters`

The parameters that are required to launch your game server. Specify these parameters as a string of command-line parameters, such as `+sv_port 33435 +start_lobby`.

Required: No

Type: String

Update requires: Replacement (p. 90)

`ServerLaunchPath`

The location of your game server that GameLift launches. You must escape the slashes (`\`) and use the following pattern: `C:\\game\\launchpath`. For example, if your game server files are in the `MyGame` folder, the path should be `C:\\game\\MyGame\\server.exe`.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the fleet ID, such as `myfleet-a01234b56-7890-1de2-f345-g67h8i901j2k`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a GameLift fleet named `MyGameFleet` with two inbound permissions. The fleet uses a `Ref` intrinsic function to specify a build, which can be declared elsewhere in the same template. For the log path and server launch path, the example uses the escape character (\) to escape the slashes (\).

JSON

```
"FleetResource": {
  "Type": "AWS::GameLift::Fleet",
  "Properties": {
    "Name": "MyGameFleet",
    "Description": "A fleet for my game",
    "BuildId": { "Ref": "BuildResource" },
    "ServerLaunchPath": "c:\\game\\TestApplicationServer.exe",
    "LogPaths": [
      "c:\\game\\testlog.log",
      "c:\\game\\testlog2.log"
    ],
    "EC2InstanceType": "t2.small",
    "DesiredEC2Instances": "2",
    "EC2InboundPermissions": [
      {
        "FromPort": "1234",
        "ToPort": "1324",
        "IpRange": "0.0.0.0/24",
        "Protocol": "TCP"
      },
      {
        "FromPort": "1356",
        "ToPort": "1578",
        "IpRange": "192.168.0.0/24",
        "Protocol": "UDP"
      }
    ]
  }
}
```

YAML

```
FleetResource:
  Type: "AWS::GameLift::Fleet"
  Properties:
```

```
    Name: "MyGameFleet"
    Description: "A fleet for my game"
    BuildId:
      Ref: "BuildResource"
    ServerLaunchPath: "c:\\game\\TestApplicationServer.exe"
    LogPaths:
      - "c:\\game\\testlog.log"
      - "c:\\game\\testlog2.log"
    EC2InstanceType: "t2.small"
    DesiredEC2Instances: "2"
    EC2InboundPermissions:
      -
        FromPort: "1234"
        ToPort: "1324"
        IpRange: "0.0.0.0/24"
        Protocol: "TCP"
      -
        FromPort: "1356"
        ToPort: "1578"
        IpRange: "192.168.0.0/24"
        Protocol: "UDP"
```

AWS::IAM::AccessKey

The AWS::IAM::AccessKey resource type generates a secret access key and assigns it to an IAM user or AWS account.

This type supports updates. For more information about updating stacks, see AWS CloudFormation Stacks Updates (p. 89).

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type": "AWS::IAM::AccessKey",
   "Properties": {
      "Serial (p. 776)": Integer,
      "Status (p. 776)": String,
      "UserName (p. 776)": String
   }
}
```

YAML

```
Type: "AWS::IAM::AccessKey"
Properties:
  Serial (p. 776): Integer
```

```
  Status (p. 776): String
  UserName (p. 776): String
```

Properties

`Serial`

This value is specific to AWS CloudFormation and can only be *incremented*. Incrementing this value notifies AWS CloudFormation that you want to rotate your access key. When you update your stack, AWS CloudFormation will replace the existing access key with a new key.

Required: No

Type: Integer

Update requires: Replacement (p. 90)

`Status`

The status of the access key. By default, AWS CloudFormation sets this property value to `Active`.

Required: No

Type: String

Valid values: `Active` or `Inactive`

Update requires: No interruption (p. 90)

`UserName`

The name of the user that the new key will belong to.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

Specifying this resource ID to the intrinsic `Ref` function will return the `AccessKeyId`. For example: `AKIAIOSFODNN7EXAMPLE`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`SecretAccessKey`

Returns the secret access key for the specified `AWS::IAM::AccessKey` resource. For example: `wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Template Examples

To view AWS::IAM::AccessKey snippets, see Declaring an IAM Access Key Resource (p. 319).

AWS::IAM::Group

The `AWS::IAM::Group` resource creates an AWS Identity and Access Management (IAM) group.

This type supports updates. For more information about updating stacks, see AWS CloudFormation Stacks Updates (p. 89).

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type": "AWS::IAM::Group",
   "Properties": {
      "GroupName": String,
      "ManagedPolicyArns": [ String, ... ],
      "Path": String,
      "Policies": [ Policies, ... ]
   }
}
```

YAML

```
Type: "AWS::IAM::Group"
Properties:
  GroupName: String
  ManagedPolicyArns: [ String, ... ]
  Path: String
  Policies:
    - Policies
```

Properties

`GroupName`

A name for the IAM group. For valid values, see the `GroupName` parameter for the `CreateGroup` action in the *IAM API Reference*. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the group name.

> **Important**
> If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

If you specify a name, you must specify the `CAPABILITY_NAMED_IAM` value to acknowledge your template's capabilities. For more information, see Acknowledging IAM Resources in AWS CloudFormation Templates (p. 11).

> **Warning**
> Naming an IAM resource can cause an unrecoverable error if you reuse the same template in multiple regions. To prevent this, we recommend using `Fn::Join` and `AWS::Region` to create a region-specific name, as in the following example: `{"Fn::Join": ["", [{"Ref": "AWS::Region"}, {"Ref": "MyResourceName"}]]}`.

Required: No

Type: String

Update requires: Replacement (p. 90)

`ManagedPolicyArns`

One or more managed policy ARNs to attach to this group.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`Path`

The path to the group. For more information about paths, see IAM Identifiers in the *IAM User Guide*.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Policies`

The policies to associate with this group. For information about policies, see Overview of IAM Policies in the *IAM User Guide*.

Required: No

Type: List of IAM Policies (p. 1184)

Update requires: No interruption (p. 90)

Return Values

Ref

Specifying this resource ID to the intrinsic `Ref` function will return the `GroupName`. For example: `mystack-mygroup-1DZETITOWEKVO`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`Arn`

Returns the Amazon Resource Name (ARN) for the `AWS::IAM::Group` resource. For example: `arn:aws:iam::123456789012:group/mystack-mygroup-1DZETITOWEKVO`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Template Examples

To view AWS::IAM::Group snippets, see Declaring an IAM Group Resource (p. 321)

AWS::IAM::InstanceProfile

The `AWS::IAM::InstanceProfile` resource creates an AWS Identity and Access Management (IAM) instance profile that can be used with IAM roles for EC2 instances.

For more information about IAM roles, see Working with Roles in the *AWS Identity and Access Management User Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type": "AWS::IAM::InstanceProfile",
   "Properties": {
      "Path (p. 779)": String,
      "Roles (p. 780)": [ IAM Roles ],
      "InstanceProfileName (p. 780)": String
   }
}
```

YAML

```
Type: "AWS::IAM::InstanceProfile"
Properties:
  Path (p. 779): String
  Roles (p. 780):
    - IAM Roles
  InstanceProfileName (p. 780): String
```

Properties

`Path`

The path associated with this IAM instance profile. For information about IAM paths, see Friendly Names and Paths in the *AWS Identity and Access Management User Guide*.

By default, AWS CloudFormation specifies `/` for the path.
Required: No
Type: String

Update requires: Replacement (p. 90)

`Roles`

The name of an existing IAM role to associate with this instance profile. Currently, you can assign a maximum of one role to an instance profile.

Required: Yes

Type: List of strings

Update requires: No interruption (p. 90)

`InstanceProfileName`

The name of the instance profile that you want to create. This parameter allows (per its regex pattern) a string consisting of upper and lowercase alphanumeric characters with no spaces. You can also include any of the following characters: `= , . @ -`.

Required: No

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "MyProfile" }
```

For the `IAM::InstanceProfile` with the logical ID `MyProfile`, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`Arn`

Returns the Amazon Resource Name (ARN) for the instance profile. For example:

```
{"Fn::GetAtt" : ["MyProfile", "Arn"] }
```

This returns a value such as `“arn:aws:iam::1234567890:instance-profile/MyProfile-ASDNSDLKJ”`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

AWS::IAM::ManagedPolicy

`AWS::IAM::ManagedPolicy` creates an AWS Identity and Access Management (IAM) managed policy for your AWS account that you can use to apply permissions to IAM users, groups, and roles. For more information about managed policies, see Managed Policies and Inline Policies in the *IAM User Guide* guide.

Topics

- Properties (p. 781)
- Return Values (p. 782)
- Example (p. 783)

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type": "AWS::IAM::ManagedPolicy",
  "Properties": {
    "Description" : String,
    "Groups" : [ String, ... ],
    "Path" : String,
    "PolicyDocument" : JSON object,
    "Roles" : [ String, ... ],
    "Users" : [ String, ... ]
  }
}
```

YAML

```
Type: "AWS::IAM::ManagedPolicy"
Properties:
  Description: String
  Groups:
    - String
  Path: String
  PolicyDocument: JSON object
  Roles:
    - String
  Users:
    - String
```

Properties

`Description`

A description of the policy. For example, you can describe the permissions that are defined in the policy.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Groups`

The names of groups to attach to this policy.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`Path`

The path for the policy. By default, the path is `/`. For more information, see IAM Identifiers in the *IAM User Guide* guide.

Required: No

Type: String

Update requires: Replacement (p. 90)

`PolicyDocument`

Policies that define the permissions for this managed policy. For more information about policy syntax, see IAM Policy Elements Reference in *IAM User Guide*.

Required: Yes

Type: JSON object

Note
AWS Identity and Access Management (IAM) requires that policies be in JSON format. However, for templates formatted in YAML, you can create an IAM policy in either JSON or YAML format. AWS CloudFormation always converts a policy to JSON format before submitting it to IAM.

Update requires: No interruption (p. 90)

`Roles`

The names of roles to attach to this policy.

Note
If a policy has a `Ref` to a role and if a resource (such as `AWS::ECS::Service`) also has a `Ref` to the same role, add a `DependsOn` attribute to the resource so that the resource depends on the policy. This dependency ensures that the role's policy is available throughout the resource's lifecycle. For example, when you delete a stack with an `AWS::ECS::Service` resource, the `DependsOn` attribute ensures that the `AWS::ECS::Service` resource can complete its deletion before its role's policy is deleted.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`Users`

The names of users to attach to this policy.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the ARN.

In the following sample, the `Ref` function returns the ARN of the `CreateTestDBPolicy` managed policy, such as `arn:aws:iam::123456789012:policy/teststack-CreateTestDBPolicy-16M23YE3CS700`.

```
{ "Ref": "CreateTestDBPolicy" }
```

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following snippet creates a managed policy and associates it with the `TestDBGroup` group. The managed policy grants users permission to create t2.micro database instances. The database must use the MySQL database engine and the instance name must include the prefix `test`.

JSON

```
"CreateTestDBPolicy" : {
  "Type" : "AWS::IAM::ManagedPolicy",
  "Properties" : {
    "Description" : "Policy for creating a test database",
    "Path" : "/",
    "PolicyDocument" :   {
      "Version":"2012-10-17",
      "Statement" : [{
        "Effect" : "Allow",
        "Action" : "rds:CreateDBInstance",
        "Resource" : {"Fn::Join" : [ "", [ "arn:aws:rds:", { "Ref" : "AWS::Region" }, ":",
 { "Ref" : "AWS::AccountId" }, ":db:test*" ] ]},
        "Condition" : {
          "StringEquals" : { "rds:DatabaseEngine" : "mysql" }
        }
      },
      {
        "Effect" : "Allow",
        "Action" : "rds:CreateDBInstance",
        "Resource" : {"Fn::Join" : [ "", [ "arn:aws:rds:", { "Ref" : "AWS::Region" }, ":",
 { "Ref" : "AWS::Region" }, ":db:test*" ] ]},
        "Condition" : {
          "StringEquals" : { "rds:DatabaseClass" : "db.t2.micro" }
        }
      }]
    },
    "Groups" : ["TestDBGroup"]
  }
}
```

YAML

```
CreateTestDBPolicy:
  Type: "AWS::IAM::ManagedPolicy"
  Properties:
    Description: "Policy for creating a test database"
    Path: "/"
    PolicyDocument:
      Version: "2012-10-17"
      Statement:
        -
          Effect: "Allow"
          Action: "rds:CreateDBInstance"
          Resource:
            Fn::Join:
              - ""
```

```
              -
                - "arn:aws:rds:"
                -
                  Ref: "AWS::Region"
                - ":"
                -
                  Ref: "AWS::AccountId"
                - ":db:test*"
          Condition:
            StringEquals:
              rds:DatabaseEngine: "mysql"
        -
          Effect: "Allow"
          Action: "rds:CreateDBInstance"
          Resource:
            Fn::Join:
              - ""
              -
                - "arn:aws:rds:"
                -
                  Ref: "AWS::Region"
                - ":"
                -
                  Ref: "AWS::Region"
                - ":db:test*"
          Condition:
            StringEquals:
              rds:DatabaseClass: "db.t2.micro"
    Groups:
      - "TestDBGroup"
```

AWS::IAM::Policy

The `AWS::IAM::Policy` resource associates an IAM policy with IAM users, roles, or groups. For more information about IAM policies, see Overview of IAM Policies in the *IAM User Guide* guide.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::IAM::Policy",
  "Properties" : {
    "Groups (p. 785)" : [ String, ... ],
    "PolicyDocument (p. 785)" : JSON object,
    "PolicyName (p. 785)" : String,
    "Roles (p. 785)" : [ String, ... ],
    "Users (p. 786)" : [ String, ... ]
  }
}
```

YAML

```
Type: "AWS::IAM::Policy"
Properties:
  Groups (p. 785):
    - String
  PolicyDocument (p. 785): JSON object
  PolicyName (p. 785): String
  Roles (p. 785):
    - String
  Users (p. 786):
    - String
```

Properties

`Groups`

The names of groups to which you want to add the policy.

Required: Conditional. You must specify at least one of the following properties: `Groups`, `Roles`, or `Users`.

Type: List of strings

Update requires: No interruption (p. 90)

`PolicyDocument`

A policy document that contains permissions to add to the specified users or groups.

Required: Yes

Type: JSON object

Note
AWS Identity and Access Management (IAM) requires that policies be in JSON format. However, for templates formatted in YAML, you can create an IAM policy in either JSON or YAML format. AWS CloudFormation always converts a policy to JSON format before submitting it to IAM.

Update requires: No interruption (p. 90)

`PolicyName`

The name of the policy. If you specify multiple policies for an entity, specify unique names. For example, if you specify a list of policies for an IAM role, each policy must have a unique name.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`Roles`

The names of AWS::IAM::Role (p. 787)s to attach to this policy.

Note
If a policy has a `Ref` to a role and if a resource (such as `AWS::ECS::Service`) also has a `Ref` to the same role, add a `DependsOn` attribute to the resource so that the resource depends on the policy. This dependency ensures that the role's policy is available throughout the resource's lifecycle. For example, when you delete a stack with an `AWS::ECS::Service` resource, the

`DependsOn` attribute ensures that the `AWS::ECS::Service` resource can complete its deletion before its role's policy is deleted.

Required: Conditional. You must specify at least one of the following properties: `Groups`, `Roles`, or `Users`.

Type: List of strings

Update requires: No interruption (p. 90)

`Users`

The names of users for whom you want to add the policy.

Required: Conditional. You must specify at least one of the following properties: `Groups`, `Roles`, or `Users`.

Type: List of strings

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

IAM Policy with policy group

JSON

```
{
   "Type" : "AWS::IAM::Policy",
   "Properties" : {
      "PolicyName" : "CFNUsers",
      "PolicyDocument" : {
         "Version" : "2012-10-17",
         "Statement": [ {
         "Effect"   : "Allow",
         "Action"   : [
            "cloudformation:Describe*",
            "cloudformation:List*",
            "cloudformation:Get*"
         ],
         "Resource" : "*"
         } ]
      },
      "Groups" : [ { "Ref" : "CFNUserGroup" } ]
   }
}
```

YAML

```
Type: "AWS::IAM::Policy"
Properties:
  PolicyName: "CFNUsers"
```

```
  PolicyDocument:
    Version: "2012-10-17"
    Statement:
      -
        Effect: "Allow"
        Action:
          - "cloudformation:Describe*"
          - "cloudformation:List*"
          - "cloudformation:Get*"
        Resource: "*"
  Groups:
    -
      Ref: "CFNUserGroup"
```

IAM Policy with specified role

JSON

```
{
   "Type": "AWS::IAM::Policy",
   "Properties": {
      "PolicyName": "root",
      "PolicyDocument": {
         "Version" : "2012-10-17",
         "Statement": [
            { "Effect": "Allow", "Action": "*", "Resource": "*" }
         ]
      },
      "Roles": [ { "Ref": "RootRole" } ]
   }
}
```

YAML

```
Type: "AWS::IAM::Policy"
Properties:
  PolicyName: "root"
  PolicyDocument:
    Version: "2012-10-17"
    Statement:
      -
        Effect: "Allow"
        Action: "*"
        Resource: "*"
  Roles:
    -
      Ref: "RootRole"
```

AWS::IAM::Role

Creates an AWS Identity and Access Management (IAM) role. Use an IAM role to enable applications running on an EC2 instance to securely access your AWS resources.

For more information about IAM roles, see Working with Roles in the *AWS Identity and Access Management User Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type": "AWS::IAM::Role",
  "Properties": {
    "AssumeRolePolicyDocument (p. 788)": { JSON },
    "ManagedPolicyArns": [ String, ... ],
    "Path (p. 789)": String,
    "Policies (p. 789)": [ Policies, ... ],
    "RoleName": String
  }
}
```

YAML

```
Type: "AWS::IAM::Role"
Properties:
  AssumeRolePolicyDocument (p. 788):
    JSON object
  ManagedPolicyArns:
    - String
  Path (p. 789): String
  Policies (p. 789):
    - Policies
  RoleName: String
```

Properties

`AssumeRolePolicyDocument`

The trust policy that is associated with this role.

Required: Yes

Type: A JSON policy document

Note
AWS Identity and Access Management (IAM) requires that policies be in JSON format. However, for templates formatted in YAML, you can create an IAM policy in either JSON or YAML format. AWS CloudFormation always converts a policy to JSON format before submitting it to IAM.

Update requires: No interruption (p. 90)

Note
You can associate only one assume role policy with a role. For an example of an assume role policy, see Template Examples (p. 790).

`ManagedPolicyArns`

One or more managed policy ARNs to attach to this role.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`Path`

The path associated with this role. For information about IAM paths, see Friendly Names and Paths in *IAM User Guide*.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Policies`

The policies to associate with this role. For sample templates, see Template Examples (p. 790).

Important

The name of each policy for a role, user, or group must be unique. If you don't, updates to the IAM role will fail.

Note

If an external policy (such as `AWS::IAM::Policy` or `AWS::IAM::ManagedPolicy`) has a `Ref` to a role and if a resource (such as `AWS::ECS::Service`) also has a `Ref` to the same role, add a `DependsOn` attribute to the resource to make the resource depend on the external policy. This dependency ensures that the role's policy is available throughout the resource's lifecycle. For example, when you delete a stack with an `AWS::ECS::Service` resource, the `DependsOn` attribute ensures that AWS CloudFormation deletes the `AWS::ECS::Service` resource before deleting its role's policy.

Required: No

Type: List of IAM Policies (p. 1184)

Update requires: No interruption (p. 90)

`RoleName`

A name for the IAM role. For valid values, see the `RoleName` parameter for the `CreateRole` action in the *IAM API Reference*. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the group name.

Important

If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

If you specify a name, you must specify the `CAPABILITY_NAMED_IAM` value to acknowledge your template's capabilities. For more information, see Acknowledging IAM Resources in AWS CloudFormation Templates (p. 11).

Warning

Naming an IAM resource can cause an unrecoverable error if you reuse the same template in multiple regions. To prevent this, we recommend using `Fn::Join` and `AWS::Region` to create a region-specific name, as in the following example: `{"Fn::Join": ["", [{"Ref": "AWS::Region"}, {"Ref": "MyResourceName"}]]}`.

Required: No

Type: String

Update requires: Replacement (p. 90)

Notes on policies for IAM roles

For general information about IAM policies and policy documents, see How to Write a Policy in *IAM User Guide*.

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "RootRole" }
```

For the IAM::Role with the logical ID "RootRole", `Ref` will return the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`Arn`

Returns the Amazon Resource Name (ARN) for the instance profile. For example:

```
{"Fn::GetAtt" : ["MyRole", "Arn"] }
```

This will return a value such as `“arn:aws:iam::1234567890:role/MyRole-AJJHDSKSDF”`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Template Examples

IAM Role with Embedded Policy and Instance Profiles

This example shows an embedded Policy in the IAM::Role. The policy is specified inline in the IAM::Role Policies property.

JSON

```
{
   "AWSTemplateFormatVersion": "2010-09-09",
   "Resources": {
      "RootRole": {
         "Type": "AWS::IAM::Role",
         "Properties": {
            "AssumeRolePolicyDocument": {
               "Version" : "2012-10-17",
               "Statement": [ {
                  "Effect": "Allow",
                  "Principal": {
                     "Service": [ "ec2.amazonaws.com" ]
                  },
```

```
                  "Action": [ "sts:AssumeRole" ]
               } ]
            },
            "Path": "/",
            "Policies": [ {
               "PolicyName": "root",
               "PolicyDocument": {
                  "Version" : "2012-10-17",
                  "Statement": [ {
                     "Effect": "Allow",
                     "Action": "*",
                     "Resource": "*"
                  } ]
               }
               } ]
            }
      },
      "RootInstanceProfile": {
         "Type": "AWS::IAM::InstanceProfile",
         "Properties": {
            "Path": "/",
            "Roles": [ {
               "Ref": "RootRole"
            } ]
         }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  RootRole:
    Type: "AWS::IAM::Role"
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          -
            Effect: "Allow"
            Principal:
              Service:
                - "ec2.amazonaws.com"
            Action:
              - "sts:AssumeRole"
      Path: "/"
      Policies:
        -
          PolicyName: "root"
          PolicyDocument:
            Version: "2012-10-17"
            Statement:
              -
                Effect: "Allow"
                Action: "*"
                Resource: "*"
  RootInstanceProfile:
    Type: "AWS::IAM::InstanceProfile"
    Properties:
      Path: "/"
      Roles:
        -
          Ref: "RootRole"
```

IAM Role with External Policy and Instance Profiles

In this example, the Policy and InstanceProfile resources are specified externally to the IAM Role. They refer to the role by specifying its name, "RootRole", in their respective Roles properties.

JSON

```
{
   "AWSTemplateFormatVersion": "2010-09-09",
   "Resources": {
      "RootRole": {
         "Type": "AWS::IAM::Role",
         "Properties": {
            "AssumeRolePolicyDocument": {
               "Version" : "2012-10-17",
               "Statement": [ {
                  "Effect": "Allow",
                  "Principal": {
                     "Service": [ "ec2.amazonaws.com" ]
                  },
                  "Action": [ "sts:AssumeRole" ]
               } ]
            },
            "Path": "/"
         }
      },
      "RolePolicies": {
         "Type": "AWS::IAM::Policy",
         "Properties": {
            "PolicyName": "root",
            "PolicyDocument": {
               "Version" : "2012-10-17",
               "Statement": [ {
                  "Effect": "Allow",
                  "Action": "*",
                  "Resource": "*"
               } ]
            },
            "Roles": [ {
               "Ref": "RootRole"
            } ]
         }
      },
      "RootInstanceProfile": {
         "Type": "AWS::IAM::InstanceProfile",
         "Properties": {
            "Path": "/",
            "Roles": [ {
               "Ref": "RootRole"
            } ]
         }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  RootRole:
    Type: "AWS::IAM::Role"
    Properties:
      AssumeRolePolicyDocument:
```

```
        Version: "2012-10-17"
        Statement:
          -
            Effect: "Allow"
            Principal:
              Service:
                - "ec2.amazonaws.com"
            Action:
              - "sts:AssumeRole"
      Path: "/"
  RolePolicies:
    Type: "AWS::IAM::Policy"
    Properties:
      PolicyName: "root"
      PolicyDocument:
        Version: "2012-10-17"
        Statement:
          -
            Effect: "Allow"
            Action: "*"
            Resource: "*"
      Roles:
        -
          Ref: "RootRole"
  RootInstanceProfile:
    Type: "AWS::IAM::InstanceProfile"
    Properties:
      Path: "/"
      Roles:
        -
          Ref: "RootRole"
```

See Also

- AWS Identity and Access Management Template Snippets (p. 317)
- AWS::IAM::InstanceProfile (p. 779)

AWS::IAM::User

The `AWS::IAM::User` type creates a user.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type": "AWS::IAM::User",
  "Properties": {
    "Groups (p. 794)": [ String, ... ],
```

```
    "LoginProfile (p. 794)": LoginProfile Type,
    "ManagedPolicyArns": [ String, ... ],
    "Path (p. 794)": String,
    "Policies (p. 795)": [ Policies, ... ],
    "UserName": String
  }
}
```

YAML

```
Type: "AWS::IAM::User"
Properties:
  Groups (p. 794):
    - String
  LoginProfile (p. 794):
    LoginProfile Type
  ManagedPolicyArns:
    - String
  Path (p. 794): String
  Policies (p. 795):
    - Policies
  UserName: String
```

Properties

`Groups`

A name of a group to which you want to add the user.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`LoginProfile`

Creates a login profile so that the user can access the AWS Management Console.

Required: No

Type: IAM User LoginProfile (p. 1184)

Update requires: No interruption (p. 90)

`ManagedPolicyArns`

One or more managed policy ARNs to attach to this user.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`Path`

The path for the user name. For more information about paths, see IAM Identifiers in the *IAM User Guide*.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Policies`

The policies to associate with this user. For information about policies, see Overview of IAM Policies in the *IAM User Guide*.

Note
If you specify multiple polices, specify unique values for the policy name. If you don't, updates to the IAM user will fail.

Required: No

Type: List of IAM Policies (p. 1184)

Update requires: No interruption (p. 90)

`UserName`

A name for the IAM user. For valid values, see the `UserName` parameter for the `CreateUser` action in the *IAM API Reference*. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the user name.

Important
If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

If you specify a name, you must specify the `CAPABILITY_NAMED_IAM` value to acknowledge your template's capabilities. For more information, see Acknowledging IAM Resources in AWS CloudFormation Templates (p. 11).

Warning
Naming an IAM resource can cause an unrecoverable error if you reuse the same template in multiple regions. To prevent this, we recommend using `Fn::Join` and `AWS::Region` to create a region-specific name, as in the following example: `{"Fn::Join": ["", [{"Ref": "AWS::Region"}, {"Ref": "MyResourceName"}]]}`.

Required: No

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

Specifying this resource ID to the intrinsic Ref function will return the `UserName`. For example: `mystack-myuser-1CCXAFG2H2U4D`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`Arn`

Returns the Amazon Resource Name (ARN) for the specified AWS::IAM::User resource. For example: `arn:aws:iam::123456789012:user/mystack-myuser-1CCXAFG2H2U4D`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Template Examples

To view AWS::IAM::User snippets, see: Declaring an IAM User Resource (p. 318).

AWS::IAM::UserToGroupAddition

The `AWS::IAM::UserToGroupAddition` type adds AWS Identity and Access Management (IAM) users to a group.

This type supports updates. For more information about updating stacks, see AWS CloudFormation Stacks Updates (p. 89).

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type": "AWS::IAM::UserToGroupAddition",
   "Properties": {
      "GroupName (p. 796)": String,
      "Users (p. 796)": [ User1, ... ]
   }
}
```

YAML

```
Type: "AWS::IAM::UserToGroupAddition"
Properties:
  GroupName (p. 796): String
  Users (p. 796):
    - User1
```

Properties

`GroupName`

The name of group to add users to.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`Users`

Required: Yes

Type: List of users

Update requires: No interruption (p. 90)

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "MyUserToGroupAddition" }
```

For the `AWS::IAM::UserToGroupAddition` with the logical ID "MyUserToGroupAddition", `Ref` will return the AWS resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Template Examples

To view `AWS::IAM::UserToGroupAddition` snippets, see Adding Users to a Group (p. 322).

AWS::IoT::Certificate

Use the `AWS::IoT::Certificate` resource to declare an X.509 certificate.

For information about working with X.509 certificates, see Authentication in AWS IoT in the *AWS IoT Developer Guide*.

Syntax

JSON

```
{
   "Type": "AWS::IoT::Certificate",
   "Properties": {
      "CertificateSigningRequest": String,
      "Status": String
    }
}
```

YAML

```
Type: "AWS::IoT::Certificate"
  Properties:
    CertificateSigningRequest: String
    Status: String
```

Properties

`CertificateSigningRequest`

The certificate signing request (CSR).

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Status`

The status of the certificate.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

Return Values

Ref

When you provide the logical ID of this resource to the `Ref` intrinsic function, `Ref` returns the certificate ID. For example:

```
{ "Ref": "MyCertificate" }
```

A value similar to the following is returned:

```
a1234567b89c012d3e4fg567hij8k9l01mno1p23q45678901rs234567890t1u2
```

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`Arn`

Returns the Amazon Resource Name (ARN) for the instance profile. For example:

```
{ "Fn::GetAtt": ["MyCertificate", "Arn"] }
```

A value similar to the following is returned:

```
arn:aws:iot:ap-southeast-2:123456789012:cert/
a1234567b89c012d3e4fg567hij8k9l01mno1p23q45678901rs234567890t1u2
```

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Example

The following example declares an X.509 certificate and its status.

JSON

```
{
   "AWSTemplateFormatVersion": "2010-09-09",
   "Resources": {
      "MyCertificate": {
         "Type": "AWS::IoT::Certificate",
```

```
            "Properties": {
               "CertificateSigningRequest": {
                  "Ref": "CSRParameter"
               },
               "Status": {
                  "Ref": "StatusParameter"
               }
            }
         }
      },
      "Parameters": {
         "CSRParameter": {
            "Type": "String"
         },
         "StatusParameter": {
            "Type": "String"
         }
      }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  MyCertificate:
    Type: "AWS::IoT::Certificate"
    Properties:
      CertificateSigningRequest:
        Ref: "CSRParameter"
      Status:
        Ref: "StatusParameter"
Parameters:
  CSRParameter:
    Type: "String"
  StatusParameter:
    Type: "String"
```

AWS::IoT::Policy

Use the `AWS::IoT::Policy` resource to declare an AWS IoT policy.

For information about working with AWS IoT policies, see Authorization in the *AWS IoT Developer Guide*.

Syntax

JSON

```
{
   "Type": "AWS::IoT::Policy",
   "Properties": {
      "PolicyDocument": JSON object,
      "PolicyName": String
    }
}
```

YAML

```
Type: "AWS::IoT::Policy"
Properties:
```

```
  PolicyDocument: JSON object
  PolicyName: String
```

Properties

`PolicyDocument`

The JSON document that describes the policy.

Required: Yes

Type: JSON object

Update requires: Replacement (p. 90)

`PolicyName`

The name (the physical ID) of the AWS IoT policy.

Required: No

Type: String

Update requires: Replacement (p. 90)

Return Value

Ref

When you provide the logical ID of this resource to the `Ref` intrinsic function, `Ref` returns the policy name. For example:

```
{ "Ref": "MyPolicy" }
```

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example declares an AWS IoT policy.

JSON

```
{
   "AWSTemplateFormatVersion": "2010-09-09",
   "Resources": {
      "MyPolicy": {
         "Type": "AWS::IoT::Policy",
         "Properties": {
            "PolicyName": {
               "Ref": "NameParameter"
            },
            "PolicyDocument": {
               "Version": "2012-10-17",
               "Statement": [{
                  "Effect": "Allow",
                  "Action": [
                     "iot:Connect"
                  ],
                  "Resource": [
```

```
                      "*"
                   ]
                }]
             }
          }
       }
    },
    "Parameters": {
       "NameParameter": {
          "Type": "String"
       }
    }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  MyPolicy:
    Type: "AWS::IoT::Policy"
    Properties:
      PolicyName:
        Ref: "NameParameter"
      PolicyDocument:
        Version: "2012-10-17"
        Statement:
          -
            Effect: "Allow"
            Action:
              - "iot:Connect"
            Resource:
              - "*"
Parameters:
  NameParameter:
    Type: "String"
```

AWS::IoT::PolicyPrincipalAttachment

Use the `AWS::IoT::PolicyPrincipalAttachment` resource to attach an AWS IoT policy to a principal (an X.509 certificate or other credential).

For information about working with AWS IoT policies and principals, see Authorization in the *AWS IoT Developer Guide*.

Syntax

JSON

```
{
   "Type": "AWS::IoT::PolicyPrincipalAttachment",
   "Properties": {
      "PolicyName": String,
      "Principal": String
    }
}
```

YAML

```
Type: "AWS::IoT::PolicyPrincipalAttachment"
```

```
  Properties:
    PolicyName: String
    Principal: String
```

Properties

`PolicyName`

The name of the policy.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Principal`

The principal, which can be a certificate ARN (as returned from the `CreateCertificate` operation) or an Amazon Cognito ID.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Example

The following example attaches a policy to a principal.

JSON

```
{
   "AWSTemplateFormatVersion": "2010-09-09",
   "Resources": {
      "MyPolicyPrincipalAttachment": {
         "Type": "AWS::IoT::PolicyPrincipalAttachment",
         "Properties": {
            "PolicyName": {
               "Ref": "NameParameter"
            },
            "Principal": "arn:aws:iot:ap-southeast-2:123456789012:cert/
a1234567b89c012d3e4fg567hij8k9l01mno1p23q45678901rs234567890t1u2"
         }
      }
   },
   "Parameters": {
      "NameParameter": {
         "Type": "String"
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  MyPolicyPrincipalAttachment:
```

```
    Type: "AWS::IoT::PolicyPrincipalAttachment"
    Properties:
      PolicyName:
        Ref: "NameParameter"
      Principal: "arn:aws:iot:ap-southeast-2:123456789012:cert/
a1234567b89c012d3e4fg567hij8k9l01mno1p23q45678901rs234567890t1u2"
Parameters:
  NameParameter:
    Type: "String"
```

AWS::IoT::Thing

Use the `AWS::IoT::Thing` resource to declare an AWS IoT thing.

For information about working with things, see How AWS IoT Works and Device Registry for AWS IoT in the *AWS IoT Developer Guide*.

Syntax

JSON

```
{
   "Type": "AWS::IoT::Thing",
   "Properties": {
      "AttributePayload": { String:String, ... },
      "ThingName": String
    }
}
```

YAML

```
Type: "AWS::IoT::Thing"
Properties:
  AttributePayload:
    - String:String
  ThingName: String
```

Properties

`AttributePayload`

A JSON string that contains up to three key-value pairs, for example: `{ "attributes": { "string1":"string2" } }`.

Required: No

Type: String to string map

Update requires: No interruption (p. 90)

`ThingName`

The name (the physical ID) of the AWS IoT thing.

Required: No

Type: String

Update requires: Replacement (p. 90)

Return Value

Ref

When you provide the logical ID of this resource to the `Ref` intrinsic function, `Ref` returns the thing name. For example:

```
{ "Ref": "MyThing" }
```

For a stack named `MyStack`, a value similar to the following is returned:

```
MyStack-MyThing-AB1CDEFGHIJK
```

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example declares a thing and the values of its attributes.

JSON

```
{
   "AWSTemplateFormatVersion": "2010-09-09",
   "Resources": {
      "MyThing": {
         "Type": "AWS::IoT::Thing",
         "Properties": {
            "ThingName": {
               "Ref": "NameParameter"
            },
            "AttributePayload": {
               "attributes": {
                  "myAttributeA": {
                     "Ref": "MyAttributeValueA"
                  },
                  "myAttributeB": {
                     "Ref": "MyAttributeValueB"
                  },
                  "myAttributeC": {
                     "Ref": "MyAttributeValueC"
                  }
               }
            }
         }
      }
   },
   "Parameters": {
      "NameParameter": {
         "Type": "String"
      },
      "MyAttributeValueA": {
         "Type": "String",
         "Default": "myStringA123"
      },
      "MyAttributeValueB": {
         "Type": "String",
```

```
         "Default": "myStringB123"
      },
      "MyAttributeValueC": {
         "Type": "String",
         "Default": "myStringC123"
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  MyThing:
    Type: "AWS::IoT::Thing"
    Properties:
      ThingName:
        Ref: "NameParameter"
      AttributePayload:
        attributes:
          myAttributeA:
            Ref: "MyAttributeValueA"
          myAttributeB:
            Ref: "MyAttributeValueB"
          myAttributeC:
            Ref: "MyAttributeValueC"
Parameters:
  NameParameter:
    Type: "String"
  MyAttributeValueA:
    Type: "String"
    Default: "myStringA123"
  MyAttributeValueB:
    Type: "String"
    Default: "myStringB123"
  MyAttributeValueC:
    Type: "String"
    Default: "myStringC123"
```

AWS::IoT::ThingPrincipalAttachment

Use the `AWS::IoT::ThingPrincipalAttachment` resource to attach a principal (an X.509 certificate or another credential) to a thing.

For information about working with AWS IoT things and principals, see Authorization in the *AWS IoT Developer Guide*.

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type": "AWS::IoT::ThingPrincipalAttachment",
   "Properties": {
      "Principal": String,
      "ThingName": String
    }
```

```
}
```

YAML

```
Type: "AWS::IoT::ThingPrincipalAttachment"
Properties:
  Principal: String
  ThingName: String
```

Properties

`Principal`

The principal, which can be a certificate ARN (as returned from the `CreateCertificate` operation) or an Amazon Cognito ID.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`ThingName`

The name of the AWS IoT thing.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Example

The following example attaches a principal to a thing.

JSON

```
{
   "AWSTemplateFormatVersion": "2010-09-09",
   "Resources": {
      "MyThingPrincipalAttachment": {
         "Type": "AWS::IoT::ThingPrincipalAttachment",
         "Properties": {
            "ThingName": {
               "Ref": "NameParameter"
            },
            "Principal": "arn:aws:iot:ap-southeast-2:123456789012:cert/
a1234567b89c012d3e4fg567hij8k9l01mno1p23q45678901rs234567890t1u2"
         }
      }
   },
   "Parameters": {
      "NameParameter": {
         "Type": "String"
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  MyThingPrincipalAttachment:
    Type: "AWS::IoT::ThingPrincipalAttachment"
    Properties:
      ThingName:
        Ref: "NameParameter"
      Principal: "arn:aws:iot:ap-southeast-2:123456789012:cert/
a1234567b89c012d3e4fg567hij8k9l01mno1p23q45678901rs234567890t1u2"
Parameters:
  NameParameter:
    Type: "String"
```

AWS::IoT::TopicRule

Use the `AWS::IoT::TopicRule` resource to declare an AWS IoT rule.

For information about working with AWS IoT rules, see Rules for AWS IoT in the *AWS IoT Developer Guide*.

Syntax

JSON

```
{
   "Type": "AWS::IoT::TopicRule",
   "Properties": {
      "RuleName": String,
      "TopicRulePayload": TopicRulePayLoad
   }
}
```

YAML

```
Type: "AWS::IoT::TopicRule"
Properties:
  RuleName: String
  TopicRulePayload: TopicRulePayLoad
```

Properties

`RuleName`

The name (the physical ID) of the AWS IoT rule.

Required: No

Type: String

Update requires: Replacement (p. 90)

`TopicRulePayload`

The actions associated with the AWS IoT rule.

Required: Yes

Type: `TopicRulePayload` (p. 1197) object

Update requires: No interruption (p. 90)

Return Value

Ref

When you provide the logical ID of this resource to the `Ref` intrinsic function, `Ref` returns the topic rule name. For example:

```
{ "Ref": "MyTopicRule" }
```

For a stack named `My-Stack` (the `-` character is omitted), a value similar to the following is returned:

```
MyStackMyTopicRule12ABC3D456EFG
```

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example declares an AWS IoT rule.

JSON

```
{
   "AWSTemplateFormatVersion": "2010-09-09",
   "Resources": {
      "MyTopicRule": {
         "Type": "AWS::IoT::TopicRule",
         "Properties": {
            "RuleName": {
               "Ref": "NameParameter"
            },
            "TopicRulePayload": {
               "RuleDisabled": "true",
               "Sql": "SELECT temp FROM 'SomeTopic' WHERE temp > 60",
               "Actions": [{
                  "S3": {
                     "BucketName": {
                        "Ref": "MyBucket"
                     },
                     "RoleArn": {
                        "Fn::GetAtt": ["MyRole", "Arn"]
                     },
                     "Key": "MyKey.txt"
                  }
               }]
            }
         }
      },
      "MyBucket": {
         "Type": "AWS::S3::Bucket",
         "Properties": {}
      },
      "MyRole": {
         "Type": "AWS::IAM::Role",
         "Properties": {
```

```
            "AssumeRolePolicyDocument": {
               "Version": "2012-10-17",
               "Statement": [{
                  "Effect": "Allow",
                  "Principal": {
                     "Service": [
                        "iot.amazonaws.com"
                     ]
                  },
                  "Action": [
                     "sts:AssumeRole"
                  ]
               }]
            }
         }
      }
   },
   "Parameters": {
      "NameParameter": {
         "Type": "String"
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  MyTopicRule:
    Type: "AWS::IoT::TopicRule"
    Properties:
      RuleName:
        Ref: "NameParameter"
      TopicRulePayload:
        RuleDisabled: "true"
        Sql: >-
          Select temp FROM 'SomeTopic' WHERE temp > 60
        Actions:
          -
            S3:
              BucketName:
                Ref: "MyBucket"
              RoleArn:
                Fn::GetAtt:
                  - "MyRole"
                  - "Arn"
              Key: "MyKey.txt"
  MyBucket:
    Type: "AWS::S3::Bucket"
    Properties:
  MyRole:
    Type: "AWS::IAM::Role"
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          -
            Effect: "Allow"
            Principal:
              Service:
                - "iot.amazonaws.com"
            Action:
              - "sts:AssumeRole"
Parameters:
```

```
  NameParameter:
    Type: "String"
```

AWS::Kinesis::Stream

Creates an Amazon Kinesis stream that captures and transports data records that are emitted from data sources. For information about creating streams, see CreateStream in the *Amazon Kinesis API Reference*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::Kinesis::Stream",
   "Properties" : {
      "Name" : String,
      "ShardCount" : Integer,
      "Tags" : [ Resource Tag, ... ]
   }
}
```

YAML

```
Type: "AWS::Kinesis::Stream"
Properties:
  Name: String
  ShardCount: Integer
  Tags:
    - Resource Tag
```

Properties

`Name`

The name of the Amazon Kinesis stream. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the stream name. For more information, see Name Type (p. 1217).

> **Important**
> If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

`ShardCount`

The number of shards that the stream uses. For greater provisioned throughput, increase the number of shards.

Required: Yes

Type: Integer

Update requires: Replacement (p. 90)

`Tags`

An arbitrary set of tags (key–value pairs) to associate with the Amazon Kinesis stream.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90)

Return Values

Ref

When you specify an AWS::Kinesis::Stream resource as an argument to the `Ref` function, AWS CloudFormation returns the stream name (physical ID).

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for the `Arn` attribute.

`Arn`

The Amazon resource name (ARN) of the Amazon Kinesis stream, such as `arn:aws:kinesis:us-east-1:123456789012:stream/mystream`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

AWS::KinesisFirehose::DeliveryStream

The `AWS::KinesisFirehose::DeliveryStream` resource creates an Amazon Kinesis Firehose (Firehose) delivery stream that delivers real-time streaming data to an Amazon Simple Storage Service (Amazon S3), Amazon Redshift, or Amazon Elasticsearch Service (Amazon ES) destination. For more information, see Creating an Amazon Kinesis Firehose Delivery Stream in the *Amazon Kinesis Firehose Developer Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::KinesisFirehose::DeliveryStream",
  "Properties" : {
    "DeliveryStreamName" : String,

 "ElasticsearchDestinationConfiguration" : ElasticsearchDestinationConfiguration (p. 1200),
    "RedshiftDestinationConfiguration" : RedshiftDestinationConfiguration (p. 1203),
    "S3DestinationConfiguration" : S3DestinationConfiguration (p. 1206)
  }
}
```

YAML

```
Type: "AWS::KinesisFirehose::DeliveryStream"
Properties:
  DeliveryStreamName: String
  ElasticsearchDestinationConfiguration:
    ElasticsearchDestinationConfiguration (p. 1200)
  RedshiftDestinationConfiguration:
    RedshiftDestinationConfiguration (p. 1203)
  S3DestinationConfiguration:
    S3DestinationConfiguration (p. 1206)
```

Properties

`DeliveryStreamName`

A name for the delivery stream.

Required: No

Type: String

Update requires: Replacement (p. 90)

`ElasticsearchDestinationConfiguration`

An Amazon ES destination for the delivery stream.

Required: Conditional. You must specify only one destination configuration.

Type: Amazon Kinesis Firehose DeliveryStream ElasticsearchDestinationConfiguration (p. 1200)

Update requires: No interruption (p. 90). If you change the delivery stream destination from an Amazon ES destination to an Amazon S3 or Amazon Redshift destination, update requires some interruptions (p. 90).

`RedshiftDestinationConfiguration`

An Amazon Redshift destination for the delivery stream.

Required: Conditional. You must specify only one destination configuration.

Type: Amazon Kinesis Firehose DeliveryStream RedshiftDestinationConfiguration (p. 1203)

Update requires: No interruption (p. 90). If you change the delivery stream destination from an Amazon Redshift destination to an Amazon ES destination, update requires some interruptions (p. 90).

`S3DestinationConfiguration`

An Amazon S3 destination for the delivery stream.

Required: Conditional. You must specify only one destination configuration.

Type: Amazon Kinesis Firehose DeliveryStream S3DestinationConfiguration (p. 1206)

Update requires: No interruption (p. 90). If you change the delivery stream destination from an Amazon S3 destination to an Amazon ES destination, update requires some interruptions (p. 90).

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the delivery stream name, such as `mystack-deliverystream-1ABCD2EF3GHIJ`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a Firehose delivery stream that delivers data to an Amazon ES destination. Firehose backs up all data sent to the destination in an S3 bucket.

JSON

```
"ElasticSearchDeliveryStream": {
  "Type": "AWS::KinesisFirehose::DeliveryStream",
  "Properties": {
    "ElasticsearchDestinationConfiguration": {
      "BufferingHints": {
        "IntervalInSeconds": 60,
        "SizeInMBs": 50
      },
      "CloudWatchLoggingOptions": {
        "Enabled": true,
        "LogGroupName": "deliverystream",
        "LogStreamName": "elasticsearchDelivery"
      },
      "DomainARN": { "Ref" : "MyDomainARN" },
      "IndexName": { "Ref" : "MyIndexName" },
      "IndexRotationPeriod": "NoRotation",
      "TypeName" : "fromFirehose",
      "RetryOptions": {
         "DurationInSeconds": "60"
      },
      "RoleARN": { "Fn::GetAtt" : ["ESdeliveryRole", "Arn"] },
      "S3BackupMode": "AllDocuments",
      "S3Configuration": {
        "BucketARN": { "Ref" : "MyBackupBucketARN" },
        "BufferingHints": {
            "IntervalInSeconds": "60",
             "SizeInMBs": "50"
        },
        "CompressionFormat": "UNCOMPRESSED",
        "Prefix": "firehose/",
        "RoleARN": { "Fn::GetAtt" : ["S3deliveryRole", "Arn"] },
```

```
        "CloudWatchLoggingOptions" : {
          "Enabled" : true,
          "LogGroupName" : "deliverystream",
          "LogStreamName" : "s3Backup"
        }
      }
    }
  }
}
```

YAML

```
ElasticSearchDeliveryStream:
  Type: "AWS::KinesisFirehose::DeliveryStream"
  Properties:
    ElasticsearchDestinationConfiguration:
      BufferingHints:
        IntervalInSeconds: 60
        SizeInMBs: 50
      CloudWatchLoggingOptions:
        Enabled: true
        LogGroupName: "deliverystream"
        LogStreamName: "elasticsearchDelivery"
      DomainARN:
        Ref: "MyDomainARN"
      IndexName:
        Ref: "MyIndexName"
      IndexRotationPeriod: "NoRotation"
      TypeName: "fromFirehose"
      RetryOptions:
        DurationInSeconds: "60"
      RoleARN:
        Fn::GetAtt:
          - "ESdeliveryRole"
          - "Arn"
      S3BackupMode: "AllDocuments"
      S3Configuration:
        BucketARN:
          Ref: "MyBackupBucketARN"
        BufferingHints:
          IntervalInSeconds: "60"
          SizeInMBs: "50"
        CompressionFormat: "UNCOMPRESSED"
        Prefix: "firehose/"
        RoleARN:
          Fn::GetAtt:
            - "S3deliveryRole"
            - "Arn"
        CloudWatchLoggingOptions:
          Enabled: true
          LogGroupName: "deliverystream"
          LogStreamName: "s3Backup"
```

AWS::KMS::Alias

The `AWS::KMS::Alias` resource creates a display name for a customer master key (CMK) in AWS Key Management Service (AWS KMS). Using an alias to refer to a key can help you simplify key management. For example, when rotating keys, you can just update the alias mapping instead of tracking and changing key IDs. For more information, see Working with Aliases in the *AWS Key Management Service Developer Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::KMS::Alias",
  "Properties" : {
    "AliasName" : String,
    "TargetKeyId" : String
  }
}
```

YAML

```
Type: "AWS::KMS::Alias"
Properties:
  AliasName: String
  TargetKeyId: String
```

Properties

`AliasName`

The name of the alias. The name must start with `alias` followed by a forward slash, such as `alias/`. You can't specify aliases that begin with `alias/AWS`. These aliases are reserved.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`TargetKeyId`

The ID of the key for which you are creating the alias. Specify the key's globally unique identifier or Amazon Resource Name (ARN). You can't specify another alias.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the alias name, such as `alias/myKeyAlias`.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

The following examples create the `alias/myKeyAlias` alias for the `myKey` AWS KMS key.

JSON

```
"myKeyAlias" : {
  "Type" : "AWS::KMS::Alias",
  "Properties" : {
    "AliasName" : "alias/myKeyAlias",
    "TargetKeyId" : {"Ref":"myKey"}
  }
}
```

YAML

```
myKeyAlias:
  Type: AWS::KMS::Alias
  Properties:
    AliasName: alias/myKeyAlias
    TargetKeyId:
      Ref: myKey
```

AWS::KMS::Key

The `AWS::KMS::Key` resource creates a customer master key (CMK) in AWS Key Management Service (AWS KMS). Users (customers) can use the master key to encrypt their data stored in AWS services that are integrated with AWS KMS or within their applications. For more information, see What is the AWS Key Management Service? in the *AWS Key Management Service Developer Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::KMS::Key",
  "Properties" : {
    "Description" : String,
    "Enabled" : Boolean,
    "EnableKeyRotation" : Boolean,
    "KeyPolicy" : JSON object
  }
}
```

YAML

```
Type: "AWS::KMS::Key"
Properties:
  Description: String
  Enabled: Boolean
  EnableKeyRotation: Boolean
  KeyPolicy: JSON object
```

Properties

`Description`

A description of the key. Use a description that helps your users decide whether the key is appropriate for a particular task.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Enabled`

Indicates whether the key is available for use. AWS CloudFormation sets this value to `true` by default.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`EnableKeyRotation`

Indicates whether AWS KMS rotates the key. AWS CloudFormation sets this value to `false` by default.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`KeyPolicy`

An AWS KMS key policy to attach to the key. Use a policy to specify who has permission to use the key and which actions they can perform. For more information, see Key Policies in the *AWS Key Management Service Developer Guide*.

Required: Yes

Type: JSON object

Update requires: No interruption (p. 90)

Return Values

Ref

When you provide the logical ID of this resource to the `Ref` intrinsic function, it returns the key ID, such as `123ab456-a4c2-44cb-95fd-b781f32fbb37`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`Arn`

The ARN of the AWS KMS key, such as `arn:aws:kms:us-west-2:123456789012:key/12a34567-8c90-1defg-af84-0bf06c1747f3`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Example

The following example creates a custom CMK, which permits the IAM user `Alice` to administer the key and allows `Bob` to use the key for encrypting and decrypting data.

JSON

```
"myKey" : {
  "Type" : "AWS::KMS::Key",
  "Properties" : {
    "Description" : "A sample key",
    "KeyPolicy" : {
      "Version": "2012-10-17",
      "Id": "key-default-1",
      "Statement": [
        {
          "Sid": "Allow administration of the key",
          "Effect": "Allow",
          "Principal": { "AWS": "arn:aws:iam::123456789012:user/Alice" },
          "Action": [
            "kms:Create*",
            "kms:Describe*",
            "kms:Enable*",
            "kms:List*",
            "kms:Put*",
            "kms:Update*",
            "kms:Revoke*",
            "kms:Disable*",
            "kms:Get*",
            "kms:Delete*",
            "kms:ScheduleKeyDeletion",
            "kms:CancelKeyDeletion"
          ],
          "Resource": "*"
        },
        {
          "Sid": "Allow use of the key",
          "Effect": "Allow",
          "Principal": { "AWS": "arn:aws:iam::123456789012:user/Bob" },
          "Action": [
            "kms:Encrypt",
            "kms:Decrypt",
            "kms:ReEncrypt*",
            "kms:GenerateDataKey*",
            "kms:DescribeKey"
          ],
```

```
          "Resource": "*"
        }
      ]
    }
  }
}
```

YAML

```
myKey:
  Type: "AWS::KMS::Key"
  Properties:
    Description: "A sample key"
    KeyPolicy:
      Version: "2012-10-17"
      Id: "key-default-1"
      Statement:
        -
          Sid: "Allow administration of the key"
          Effect: "Allow"
          Principal:
            AWS: "arn:aws:iam::123456789012:user/Alice"
          Action:
            - "kms:Create*"
            - "kms:Describe*"
            - "kms:Enable*"
            - "kms:List*"
            - "kms:Put*"
            - "kms:Update*"
            - "kms:Revoke*"
            - "kms:Disable*"
            - "kms:Get*"
            - "kms:Delete*"
            - "kms:ScheduleKeyDeletion"
            - "kms:CancelKeyDeletion"
          Resource: "*"
        -
          Sid: "Allow use of the key"
          Effect: "Allow"
          Principal:
            AWS: "arn:aws:iam::123456789012:user/Bob"
          Action:
            - "kms:Encrypt"
            - "kms:Decrypt"
            - "kms:ReEncrypt*"
            - "kms:GenerateDataKey*"
            - "kms:DescribeKey"
          Resource: "*"
```

AWS::Lambda::EventSourceMapping

The `AWS::Lambda::EventSourceMapping` resource specifies a stream as an event source for an AWS Lambda (Lambda) function. The stream can be an Amazon Kinesis stream or an Amazon DynamoDB (DynamoDB) stream. Lambda invokes the associated function when records are posted to the stream. For more information, see CreateEventSourceMapping in the *AWS Lambda Developer Guide*.

Topics

• Example (p. 821)

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Lambda::EventSourceMapping",
  "Properties" : {
    "BatchSize" : Integer,
    "Enabled" : Boolean,
    "EventSourceArn" : String,
    "FunctionName" : String,
    "StartingPosition" : String
  }
}
```

YAML

```
Type: "AWS::Lambda::EventSourceMapping"
Properties:
  BatchSize: Integer
  Enabled: Boolean
  EventSourceArn: String
  FunctionName: String
  StartingPosition: String
```

Properties

`BatchSize`

The largest number of records that Lambda retrieves from your event source when invoking your function. Your function receives an event with all the retrieved records. For the default and valid values, see CreateEventSourceMapping in the *AWS Lambda Developer Guide*.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`Enabled`

Indicates whether Lambda begins polling the event source.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`EventSourceArn`

The Amazon Resource Name (ARN) of the Amazon Kinesis or DynamoDB stream that is the source of events. Any record added to this stream can invoke the Lambda function. For more information, see CreateEventSourceMapping in the *AWS Lambda Developer Guide*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`FunctionName`

The name or ARN of a Lambda function to invoke when Lambda detects an event on the stream.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`StartingPosition`

The position in the stream where Lambda starts reading. For valid values, see CreateEventSourceMapping in the *AWS Lambda Developer Guide*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example associates an Amazon Kinesis stream with a Lambda function.

JSON

```
"EventSourceMapping": {
  "Type": "AWS::Lambda::EventSourceMapping",
  "Properties": {
    "EventSourceArn" : { "Fn::Join" : [ "", [ "arn:aws:kinesis:", { "Ref" :
 "AWS::Region" }, ":", { "Ref" : "AWS::AccountId" }, ":stream/", { "Ref" :
 "KinesisStream" }] ] },
    "FunctionName" : { "Fn::GetAtt" : ["LambdaFunction", "Arn"] },
    "StartingPosition" : "TRIM_HORIZON"
  }
}
```

YAML

```
EventSourceMapping:
  Type: "AWS::Lambda::EventSourceMapping"
  Properties:
```

```
    EventSourceArn:
      Fn::Join:
        - ""
        -
          - "arn:aws:kinesis:"
          -
            Ref: "AWS::Region"
          - ":"
          -
            Ref: "AWS::AccountId"
          - ":stream/"
          -
            Ref: "KinesisStream"
    FunctionName:
      Fn::GetAtt:
        - "LambdaFunction"
        - "Arn"
    StartingPosition: "TRIM_HORIZON"
```

AWS::Lambda::Alias

The `AWS::Lambda::Alias` resource creates an alias that points to the version of an AWS Lambda (Lambda) function that you specify. Use aliases when you want to control which version of your function other services or applications invoke. Those services or applications can use your function's alias so that they don't need to be updated whenever you release a new version of your function. For more information, see Introduction to AWS Lambda Aliases in the *AWS Lambda Developer Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Lambda::Alias",
  "Properties" : {
    "Description" : String,
    "FunctionName" : String,
    "FunctionVersion" : String,
    "Name" : String
  }
}
```

YAML

```
Type: "AWS::Lambda::Alias"
Properties:
  Description: String
  FunctionName: String
  FunctionVersion: String
```

```
  Name: String
```

Properties

`Description`

Information about the alias, such as its purpose or the Lambda function that is associated with it.

Required: No

Type: String

Update requires: No interruption (p. 90)

`FunctionName`

The Lambda function that you want to associate with this alias. You can specify the function's name or its Amazon Resource Name (ARN).

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`FunctionVersion`

The version of the Lambda function that you want to associate with this alias.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`Name`

A name for the alias.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the ARN of the Lambda alias.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

The following examples create an alias named `TestingForMyApp`. The alias points to the `TestingNewFeature` version of the `MyFunction` Lambda function.

JSON

```
"AliasForMyApp" : {
  "Type" : "AWS::Lambda::Alias",
  "Properties" : {
    "FunctionName" : { "Ref" : "MyFunction" },
    "FunctionVersion" : { "Fn::GetAtt" : [ "TestingNewFeature", "Version" ] },
    "Name" : "TestingForMyApp"
  }
}
```

YAML

```
AliasForMyApp:
  Type: "AWS::Lambda::Alias"
  Properties:
    FunctionName:
      Ref: "MyFunction"
    FunctionVersion:
      Fn::GetAtt:
        - "TestingNewFeature"
        - "Version"
    Name: "TestingForMyApp"
```

AWS::Lambda::Function

The `AWS::Lambda::Function` resource creates an AWS Lambda (Lambda) function that can run code in response to events. For more information, see `CreateFunction` in the *AWS Lambda Developer Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Lambda::Function",
  "Properties" : {
    "Code" : Code,
    "DeadLetterConfig" : DeadLetterConfig (p. 1210),
    "Description" : String,
    "Environment" : Environment (p. 1210),
    "FunctionName" : String,
    "Handler" : String,
    "KmsKeyArn" : String,
    "MemorySize" : Integer,
    "Role" : String,
    "Runtime" : String,
```

```
    "Timeout" : Integer,
    "VpcConfig" : VPCConfig (p. 1216)
  }
}
```

YAML

```
Type: "AWS::Lambda::Function"
Properties:
  Code:
    Code
  DeadLetterConfig:
    DeadLetterConfig (p. 1210)
  Description: String
  Environment:
    Environment (p. 1210)
  FunctionName: String
  Handler: String
  KmsKeyArn: String
  MemorySize: Integer
  Role: String
  Runtime: String
  Timeout: Integer
  VpcConfig:
    VPCConfig (p. 1216)
```

Properties

`Code`

The source code of your Lambda function. You can point to a file in an Amazon Simple Storage Service (Amazon S3) bucket or specify your source code as inline text.

Required: Yes

Type: AWS Lambda Function Code (p. 1211)

Update requires: No interruption (p. 90)

`DeadLetterConfig`

Configures how Lambda handles events that it can't process. If you don't specify a Dead Letter Queue (DLQ) configuration, Lambda discards events after the maximum number of retries. For more information, see Dead Letter Queues in the *AWS Lambda Developer Guide*.

Required: No

Type: AWS Lambda Function DeadLetterConfig (p. 1210)

Update requires: No interruption (p. 90)

`Description`

A description of the function.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Environment`

Key-value pairs that Lambda caches and makes available for your Lambda functions. Use environment variables to apply configuration changes, such as test and production environment configurations, without changing your Lambda function source code.

Required: No

Type: AWS Lambda Function Environment (p. 1210)

Update requires: No interruption (p. 90)

`FunctionName`

A name for the function. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the function's name. For more information, see Name Type (p. 1217).

> **Important**
> If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Handler`

The name of the function (within your source code) that Lambda calls to start running your code. For more information, see the `Handler` property in the *AWS Lambda Developer Guide*.

> **Note**
> If you specify your source code as inline text by specifying the `ZipFile` property within the `Code` property, specify `index.`*`function_name`* as the handler.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`KmsKeyArn`

The Amazon Resource Name (ARN) of an AWS Key Management Service (AWS KMS) key that Lambda uses to encrypt and decrypt environment variable values.

Type: String

Required: No

Update requires: No interruption (p. 90)

`MemorySize`

The amount of memory, in MB, that is allocated to your Lambda function. Lambda uses this value to proportionally allocate the amount of CPU power. For more information, see Resource Model in the *AWS Lambda Developer Guide*.

Your function use case determines your CPU and memory requirements. For example, a database operation might need less memory than an image processing function. You must specify a value that is greater than or equal to `128`, and it must be a multiple of 64. You cannot specify a size larger than `1536`. The default value is 128 MB.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`Role`

The Amazon Resource Name (ARN) of the AWS Identity and Access Management (IAM) execution role that Lambda assumes when it runs your code to access AWS services.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`Runtime`

The runtime environment for the Lambda function that you are uploading. For valid values, see the Runtime property in the *AWS Lambda Developer Guide*.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

Note

Because Node.js 0.10.32 has been deprecated, you can no longer roll back a template that uses Node.js 0.10.32. If you update a stack to Node.js 0.10.32 and the update fails, AWS CloudFormation won't roll it back.

`Timeout`

The function execution time (in seconds) after which Lambda terminates the function. Because the execution time affects cost, set this value based on the function's expected execution time. By default, `Timeout` is set to `3` seconds.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`VpcConfig`

If the Lambda function requires access to resources in a VPC, specify a VPC configuration that Lambda uses to set up an elastic network interface (ENI). The ENI enables your function to connect to other resources in your VPC, but it doesn't provide public Internet access. If your function requires Internet access (for example, to access AWS services that don't have VPC endpoints), configure a Network Address Translation (NAT) instance inside your VPC or use an Amazon Virtual Private Cloud (Amazon VPC) NAT gateway. For more information, see NAT Gateways in the *Amazon VPC User Guide*.

Required: No

Type: AWS Lambda Function VPCConfig (p. 1216)

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

In the following sample, the `Ref` function returns the name of the `AMILookUp` function, such as `MyStack-AMILookUp-NT5EUXTNTXXD`.

```
{ "Ref": "AMILookUp" }
```

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`Arn`

The ARN of the Lambda function, such as `arn:aws:lambda:us-west-2:123456789012:MyStack-AMILookUp-NT5EUXTNTXXD`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Example

The following example uses a packaged file in an S3 bucket to create a Lambda function.

JSON

```
"AMIIDLookup": {
  "Type": "AWS::Lambda::Function",
  "Properties": {
    "Handler": "index.handler",
    "Role": { "Fn::GetAtt" : ["LambdaExecutionRole", "Arn"] },
    "Code": {
      "S3Bucket": "lambda-functions",
      "S3Key": "amilookup.zip"
    },
    "Runtime": "nodejs4.3",
    "Timeout": "25"
  }
}
```

YAML

```
AMIIDLookup:
  Type: "AWS::Lambda::Function"
  Properties:
    Handler: "index.handler"
    Role:
      Fn::GetAtt:
        - "LambdaExecutionRole"
        - "Arn"
    Code:
      S3Bucket: "lambda-functions"
      S3Key: "amilookup.zip"
```

```
    Runtime: "nodejs4.3"
    Timeout: "25"
```

Related Resources

For more information about how you can use a Lambda function with AWS CloudFormation custom resources, see AWS Lambda-backed Custom Resources (p. 368).

For a sample template, see AWS Lambda Template (p. 330).

AWS::Lambda::Permission

The `AWS::Lambda::Permission` resource associates a policy statement with a specific AWS Lambda (Lambda) function's access policy. The function policy grants a specific AWS service or application permission to invoke the function. For more information, see AddPermission in the *AWS Lambda Developer Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Lambda::Permission",
  "Properties" : {
    "Action" : String,
    "FunctionName" : String,
    "Principal" : String,
    "SourceAccount" : String,
    "SourceArn" : String
  }
}
```

YAML

```
Type: "AWS::Lambda::Permission"
Properties:
  Action: String
  FunctionName: String
  Principal: String
  SourceAccount: String
  SourceArn: String
```

Properties

`Action`

The Lambda actions that you want to allow in this statement. For example, you can specify `lambda:CreateFunction` to specify a certain action, or use a wildcard (`lambda:*`) to grant permission to all Lambda actions. For a list of actions, see Actions and Condition Context Keys for AWS Lambda in the *IAM User Guide*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`FunctionName`

The name (physical ID), Amazon Resource Name (ARN), or alias ARN of the Lambda function that you want to associate with this statement. Lambda adds this statement to the function's access policy.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Principal`

The entity for which you are granting permission to invoke the Lambda function. This entity can be any valid AWS service principal, such as `s3.amazonaws.com` or `sns.amazonaws.com`, or, if you are granting cross-account permission, an AWS account ID. For example, you might want to allow a custom application in another AWS account to push events to Lambda by invoking your function.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`SourceAccount`

The AWS account ID (without hyphens) of the source owner. For example, if you specify an S3 bucket in the `SourceArn` property, this value is the bucket owner's account ID. You can use this property to ensure that all source principals are owned by a specific account.

Important
This property is not supported by all event sources. For more information, see the `SourceAccount` parameter for the AddPermission action in the *AWS Lambda Developer Guide*.

Required: No

Type: String

Update requires: Replacement (p. 90)

`SourceArn`

The ARN of a resource that is invoking your function. When granting Amazon Simple Storage Service (Amazon S3) permission to invoke your function, specify this property with the bucket ARN as its value. This ensures that events generated only from the specified bucket, not just any bucket from any AWS account that creates a mapping to your function, can invoke the function.

Important
This property is not supported by all event sources. For more information, see the `SourceArn` parameter for the AddPermission action in the *AWS Lambda Developer Guide*.

Required: No

Type: String

Update requires: Replacement (p. 90)

Example

The following example grants an S3 bucket permission to invoke a Lambda function.

JSON

```
"LambdaInvokePermission": {
  "Type": "AWS::Lambda::Permission",
  "Properties": {
    "FunctionName" : { "Fn::GetAtt" : ["MyLambdaFunction", "Arn"] },
    "Action": "lambda:InvokeFunction",
    "Principal": "s3.amazonaws.com",
    "SourceAccount": { "Ref" : "AWS::AccountId" }
  }
}
```

YAML

```
LambdaInvokePermission:
  Type: "AWS::Lambda::Permission"
  Properties:
    FunctionName:
      Fn::GetAtt:
        - "MyLambdaFunction"
        - "Arn"
    Action: "lambda:InvokeFunction"
    Principal: "s3.amazonaws.com"
    SourceAccount:
      Ref: "AWS::AccountId"
```

AWS::Lambda::Version

The `AWS::Lambda::Version` resource publishes a specified version of an AWS Lambda (Lambda) function. When publishing a new version of your function, Lambda copies the latest version of your function. For more information, see Introduction to AWS Lambda Versioning in the *AWS Lambda Developer Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Lambda::Version",
  "Properties" : {
```

```
    "CodeSha256" : String,
    "Description" : String,
    "FunctionName" : String
  }
}
```

YAML

```
Type: "AWS::Lambda::Version"
Properties:
  CodeSha256" : String
  Description" : String
  FunctionName" : String
```

Properties

`CodeSha256`

The SHA-256 hash of the deployment package that you want to publish. This value must match the SHA-256 hash of the `$LATEST` version of the function. Specify this property to validate that you are publishing the correct package.

Required: No

Type: String

Update requires: Updates are not supported.

`Description`

A description of the version you are publishing. If you don't specify a value, Lambda copies the description from the `$LATEST` version of the function.

Required: No

Type: String

Update requires: Updates are not supported.

`FunctionName`

The Lambda function for which you want to publish a version. You can specify the function's name or its Amazon Resource Name (ARN).

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the ARN of the Lambda version, such as `arn:aws:lambda:us-west-2:123456789012:function:helloworld:1`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of the specified resource type.

`Version`

The published version of a Lambda version, such as `1`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Example

The following example publishes a new version of the `MyFunction` Lambda function.

JSON

```
"TestingNewFeature" : {
  "Type" : "AWS::Lambda::Version",
  "Properties" : {
    "FunctionName" : { "Ref" : "MyFunction" },
    "Description" : "A test version of MyFunction"
  }
}
```

YAML

```
TestingNewFeature:
  Type: "AWS::Lambda::Version"
  Properties:
    FunctionName:
      Ref: "MyFunction"
    Description: "A test version of MyFunction"
```

AWS::Logs::Destination

The `AWS::Logs::Destination` resource creates an Amazon CloudWatch Logs (CloudWatch Logs) destination, which enables you to specify a physical resource (such as an Amazon Kinesis stream) that subscribes to CloudWatch Logs log events from another AWS account. For more information, see Cross-Account Log Data Sharing with Subscriptions in the *Amazon CloudWatch User Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
```

```
  "Type" : "AWS::Logs::Destination",
  "Properties" : {
    "DestinationName" : String,
    "DestinationPolicy" : String,
    "RoleArn" : String,
    "TargetArn" : String
  }
}
```

YAML

```
Type: "AWS::Logs::Destination"
Properties:
  DestinationName: String
  DestinationPolicy: String
  RoleArn: String
  TargetArn: String
```

Properties

`DestinationName`

The name of the CloudWatch Logs destination.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`DestinationPolicy`

An AWS Identity and Access Management (IAM) policy that specifies who can write to your destination.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`RoleArn`

The Amazon Resource Name (ARN) of an IAM role that permits CloudWatch Logs to send data to the specified AWS resource (`TargetArn`).

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`TargetArn`

The ARN of the AWS resource that receives log events. Currently, you can specify only an Amazon Kinesis stream.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name, such as `TestDestination`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

In the following example, the target stream (`TestStream`) can receive log events from the `logger` IAM user that is in the `234567890123` AWS account. The user can call only the `PutSubscriptionFilter` action against the `TestDestination` destination.

JSON

```
"DestinationWithName" : {
  "Type" : "AWS::Logs::Destination",
  "Properties" : {
    "DestinationName": "TestDestination",
    "RoleArn": "arn:aws:iam::123456789012:role/LogKinesisRole",
    "TargetArn": "arn:aws:kinesis:us-east-1:123456789012:stream/TestStream",
    "DestinationPolicy": "{\"Version\" : \"2012-10-17\",\"Statement\" : [{\"Effect\" :
 \"Allow\", \"Principal\" : {\"AWS\" : \"arn:aws:iam::234567890123:user/logger\"},
\"Action\" : \"logs:PutSubscriptionFilter\", \"Resource\" : \"arn:aws:logs:us-
east-1:123456789012:destination:TestDestination\"}]}"
  }
}
```

YAML

```
DestinationWithName:
  Type: "AWS::Logs::Destination"
  Properties:
    DestinationName: "TestDestination"
    RoleArn: "arn:aws:iam::123456789012:role/LogKinesisRole"
    TargetArn: "arn:aws:kinesis:us-east-1:123456789012:stream/TestStream"
    DestinationPolicy: >
      {"Version" : "2012-10-17","Statement" : [{"Effect" : "Allow", "Principal" : {"AWS" :
 "arn:aws:iam::234567890123:user/logger"},"Action" : "logs:PutSubscriptionFilter",
 "Resource" : "arn:aws:logs:us-east-1:123456789012:destination:TestDestination"}]}
```

AWS::Logs::LogGroup

The `AWS::Logs::LogGroup` resource creates an Amazon CloudWatch Logs log group that defines common properties for log streams, such as their retention and access control rules. Each log stream must belong to one log group.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Logs::LogGroup",
  "Properties" : {
    "LogGroupName" : String,
    "RetentionInDays" : Integer
  }
}
```

YAML

```
Type: "AWS::Logs::LogGroup"
Properties:
  LogGroupName: String
  RetentionInDays: Integer
```

Properties

`LogGroupName`

A name for the log group. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the log group. For more information, see Name Type (p. 1217).

Important
If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

`RetentionInDays`

The number of days log events are kept in CloudWatch Logs. When a log event expires, CloudWatch Logs automatically deletes it. For valid values, see PutRetentionPolicy in the *Amazon CloudWatch Logs API Reference*.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`Arn`

The Amazon resource name (ARN) of the CloudWatch Logs log group, such as `arn:aws:logs:us-east-1:123456789012:log-group:/mystack-testgroup-12ABC1AB12A1:*`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Examples

The following example creates a CloudWatch Logs log group that retains events for 7 days.

JSON

```
"myLogGroup": {
    "Type": "AWS::Logs::LogGroup",
    "Properties": {
        "RetentionInDays": 7
    }
}
```

YAML

```
myLogGroup:
  Type: "AWS::Logs::LogGroup"
  Properties:
    RetentionInDays: 7
```

Additional Information

For an additional sample template, see Amazon CloudWatch Logs Template Snippets (p. 257).

AWS::Logs::LogStream

The `AWS::Logs::LogStream` resource creates an Amazon CloudWatch Logs log stream in a log group. A log stream represents the sequence of events coming from an application instance or resource that you are monitoring. For more information, see Monitoring Log Files in the *Amazon CloudWatch User Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Logs::LogStream",
  "Properties" : {
    "LogGroupName" : String,
    "LogStreamName" : String
  }
}
```

YAML

```
Type: "AWS::Logs::LogStream"
Properties:
  LogGroupName: String
  LogStreamName: String
```

Properties

`LogGroupName`

The name of the log group where the log stream is created.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`LogStreamName`

The name of the log stream to create. The name must be unique within the log group.

Required: No

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name, such as `MyAppLogStream`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a CloudWatch Logs log stream named `MyAppLogStream` in the `exampleLogGroup` log group.

JSON

```
"LogStream": {
  "Type": "AWS::Logs::LogStream",
  "Properties": {
```

```
    "LogGroupName" : "exampleLogGroup",
    "LogStreamName": "MyAppLogStream"
  }
}
```

YAML

```
LogStream:
  Type: "AWS::Logs::LogStream"
  Properties:
    LogGroupName: "exampleLogGroup"
    LogStreamName: "MyAppLogStream"
```

AWS::Logs::MetricFilter

The `AWS::Logs::MetricFilter` resource creates a metric filter that describes how Amazon CloudWatch Logs extracts information from logs that you specify and transforms it into Amazon CloudWatch metrics. If you have multiple metric filters that are associated with a log group, all the filters are applied to the log streams in that group.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type": "AWS::Logs::MetricFilter",
  "Properties": {
    "FilterPattern": String,
    "LogGroupName": String,
    "MetricTransformations": [ MetricTransformations, ... ]
  }
}
```

YAML

```
Type: "AWS::Logs::MetricFilter"
Properties:
  FilterPattern: String
  LogGroupName: String
  MetricTransformations:
    MetricTransformations
```

Properties

Note

For more information about constraints and values for each property, see PutMetricFilter in the *Amazon CloudWatch Logs API Reference*.

`FilterPattern`

Describes the pattern that CloudWatch Logs follows to interpret each entry in a log. For example, a log entry might contain fields such as timestamps, IP addresses, error codes, bytes transferred, and so on. You use the pattern to specify those fields and to specify what to look for in the log file. For example, if you're interested in error codes that begin with `1234`, your filter pattern might be `[timestamps, ip_addresses, error_codes = 1234*, size, ...]`.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`LogGroupName`

The name of an existing log group that you want to associate with this metric filter.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`MetricTransformations`

Describes how to transform data from a log into a CloudWatch metric.

Required: Yes

Type: A list of CloudWatch Logs MetricFilter MetricTransformation Property (p. 1061)

Important
Currently, you can specify only one metric transformation for each metric filter. If you want to specify multiple metric transformations, you must specify multiple metric filters.

Update requires: No interruption (p. 90)

Examples

The following example sends a value of `1` to the `404Count` metric whenever the status code field includes a `404` value.

JSON

```
"404MetricFilter": {
    "Type": "AWS::Logs::MetricFilter",
    "Properties": {
        "LogGroupName": { "Ref": "myLogGroup" },
        "FilterPattern": "[ip, identity, user_id, timestamp, request, status_code = 404,
 size]",
        "MetricTransformations": [
            {
                "MetricValue": "1",
                "MetricNamespace": "WebServer/404s",
                "MetricName": "404Count"
            }
        ]
    }
}
```

YAML

```
404MetricFilter:
  Type: "AWS::Logs::MetricFilter"
  Properties:
    LogGroupName:
      Ref: "myLogGroup"
    FilterPattern: "[ip, identity, user_id, timestamp, request, status_code = 404, size]"
    MetricTransformations:
      -
        MetricValue: "1"
        MetricNamespace: "WebServer/404s"
        MetricName: "404Count"
```

Additional Information

For an additional sample template, see Amazon CloudWatch Logs Template Snippets (p. 257).

AWS::Logs::SubscriptionFilter

The `AWS::Logs::SubscriptionFilter` resource creates an Amazon CloudWatch Logs (CloudWatch Logs) subscription filter that defines which log events are delivered to your Amazon Kinesis stream or AWS Lambda (Lambda) function and where to send them.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Logs::SubscriptionFilter",
  "Properties" : {
    "DestinationArn" : String,
    "FilterPattern" : String,
    "LogGroupName" : String,
    "RoleArn" : String
  }
}
```

YAML

```
Type: "AWS::Logs::SubscriptionFilter"
Properties:
  DestinationArn: String
  FilterPattern: String
  LogGroupName: String
  RoleArn: String
```

Properties

`DestinationArn`

The Amazon Resource Name (ARN) of the Amazon Kinesis stream or Lambda function that you want to use as the subscription feed destination.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`FilterPattern`

The filtering expressions that restrict what gets delivered to the destination AWS resource. For more information about the filter pattern syntax, see Filter and Pattern Syntax in the *Amazon CloudWatch User Guide*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`LogGroupName`

The log group to associate with the subscription filter. All log events that are uploaded to this log group are filtered and delivered to the specified AWS resource if the filter pattern matches the log events.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`RoleArn`

An IAM role that grants CloudWatch Logs permission to put data into the specified Amazon Kinesis stream. For Lambda and CloudWatch Logs destinations, don't specify this property because CloudWatch Logs gets the necessary permissions from the destination resource.

Required: No

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example sends log events that are associated with the `Root` user to an Amazon Kinesis stream.

JSON

```
"SubscriptionFilter" : {
  "Type" : "AWS::Logs::SubscriptionFilter",
  "Properties" : {
    "RoleArn" : { "Fn::GetAtt" : [ "CloudWatchIAMRole", "Arn" ] },
    "LogGroupName" : { "Ref" : "LogGroup" },
    "FilterPattern" : "{$.userIdentity.type = Root}",
    "DestinationArn" : { "Fn::GetAtt" : [ "KinesisStream", "Arn" ] }
  }
}
```

YAML

```
SubscriptionFilter:
  Type: "AWS::Logs::SubscriptionFilter"
  Properties:
    RoleArn:
      Fn::GetAtt:
        - "CloudWatchIAMRole"
        - "Arn"
    LogGroupName:
      Ref: "LogGroup"
    FilterPattern: "{$.userIdentity.type = Root}"
    DestinationArn:
      Fn::GetAtt:
        - "KinesisStream"
        - "Arn"
```

AWS::OpsWorks::App

Defines an AWS OpsWorks app for an AWS OpsWorks stack. The app specifies the code that you want to run on an application server.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type": "AWS::OpsWorks::App",
  "Properties": {
    "AppSource" : Source,
    "Attributes" : { String:String, ... },
    "DataSources" : [ DataSource (p. 1218), ... ],
    "Description" : String,
    "Domains" :  [ String, ... ],
    "EnableSsl" : Boolean,
```

```
    "Environment" : [ Environment, ... ],
    "Name" : String,
    "Shortname" : String,
    "SslConfiguration" : { SslConfiguration },
    "StackId" : String,
    "Type" : String
  }
}
```

YAML

```
Type: "AWS::OpsWorks::App"
Properties:
  AppSource:
    Source
  Attributes:
    String: String
  Description: String
  DataSources:
    - DataSource (p. 1218)
  Domains:
    - String
  EnableSsl: Boolean
  Environment:
    - Environment
  Name: String
  Shortname: String
  SslConfiguration:
    SslConfiguration
  StackId: String
  Type: String
```

Properties

`AppSource`

The information required to retrieve an app from a repository.

Required: No

Type: AWS OpsWorks Source Type (p. 1228)

Update requires: No interruption (p. 90)

`Attributes`

One or more user-defined key-value pairs to be added to the app attributes bag.

Required: No

Type: A list of key-value pairs

Update requires: No interruption (p. 90)

`Description`

A description of the app.

Required: No

Type: String

Update requires: No interruption (p. 90)

`DataSources`

A list of databases to associate with the AWS OpsWorks app.

Required: No

Type: List of DataSource (p. 1218)

Update requires: No interruption (p. 90)

`Domains`

The app virtual host settings, with multiple domains separated by commas. For example, `'www.example.com`, `example.com'`.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`EnableSsl`

Whether to enable SSL for this app.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`Environment`

The environment variables to associate with the AWS OpsWorks app.

Required: No

Type: List of AWS OpsWorks App Environment (p. 1219)

Update requires: No interruption (p. 90)

`Name`

The name of the AWS OpsWorks app.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`Shortname`

The app short name, which is used internally by AWS OpsWorks and by Chef recipes.

Required: No

Type: String

Update requires: Replacement (p. 90)

`SslConfiguration`

The SSL configuration

Required: No

Type: AWS OpsWorks SslConfiguration Type (p. 1230)

Update requires: No interruption (p. 90)

`StackId`

The ID of the AWS OpsWorks stack to associate this app with.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Type`

The app type. Each supported type is associated with a particular layer. For more information, see CreateApp in the *AWS OpsWorks Stacks API Reference*.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "myApp" }
```

For the AWS OpsWorks stack `myApp`, `Ref` returns the ID of the AWS OpsWorks app.

For more information about using the `Ref` function, see Ref (p. 1343).

Template Snippet

The following snippet creates an AWS OpsWorks app that uses a PHP application in a Git repository:

JSON

```
"myApp" : {
  "Type" : "AWS::OpsWorks::App",
  "Properties" : {
    "StackId" : {"Ref":"myStack"},
    "Type" : "php",
    "Name" : "myPHPapp",
    "AppSource" : {
      "Type" : "git",
      "Url" : "git://github.com/amazonwebservices/opsworks-demo-php-simple-app.git",
      "Revision" : "version1"
    }
  }
```

```
}
```

YAML

```
myApp:
  Type: "AWS::OpsWorks::App"
  Properties:
    StackId:
      Ref: "myStack"
    Type: "php"
    Name: "myPHPapp"
    AppSource:
      Type: "git"
      Url: "git://github.com/amazonwebservices/opsworks-demo-php-simple-app.git"
      Revision: "version1"
```

More Info

- AWS::OpsWorks::Stack (p. 862)
- AWS::OpsWorks::Layer (p. 856)
- AWS::OpsWorks::Instance (p. 848)

AWS::OpsWorks::ElasticLoadBalancerAttachment

Attaches an Elastic Load Balancing load balancer to an AWS OpsWorks layer that you specify.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type": "AWS::OpsWorks::ElasticLoadBalancerAttachment",
  "Properties": {
    "ElasticLoadBalancerName" : String,
    "LayerId" : String
  }
}
```

YAML

```
Type: "AWS::OpsWorks::ElasticLoadBalancerAttachment"
Properties:
  ElasticLoadBalancerName: String
  LayerId: String
```

Properties

`ElasticLoadBalancerName`

Elastic Load Balancing load balancer name.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`LayerId`

The AWS OpsWorks layer ID that the Elastic Load Balancing load balancer will be attached to.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

Template Snippet

The following snippet specifies a load balancer attachment to an AWS OpsWorks layer, both of which would be described elsewhere in the same template:

JSON

```
"ELBAttachment" : {
  "Type" : "AWS::OpsWorks::ElasticLoadBalancerAttachment",
    "Properties" : {
      "ElasticLoadBalancerName" : { "Ref" : "ELB" },
      "LayerId" : { "Ref" : "Layer" }
    }
}
```

YAML

```
ELBAttachment:
  Type: "AWS::OpsWorks::ElasticLoadBalancerAttachment"
  Properties:
    ElasticLoadBalancerName:
      Ref: "ELB"
    LayerId:
      Ref: "Layer"
```

See Also

- AWS::OpsWorks::Layer (p. 856)

AWS::OpsWorks::Instance

Creates an instance for an AWS OpsWorks stack. These instances are the Amazon Elastic Compute Cloud (Amazon EC2) instances that, for example, handle the work of serving applications and balancing traffic.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type": "AWS::OpsWorks::Instance",
  "Properties": {
    "AgentVersion" : String,
    "AmiId" : String,
    "Architecture" : String,
    "AutoScalingType" : String,
    "AvailabilityZone" : String,
    "BlockDeviceMappings" : [ BlockDeviceMapping (p. 1224), ... ],
    "EbsOptimized" : Boolean,
    "ElasticIps" : [ String, ... ],
    "Hostname" : String,
    "InstallUpdatesOnBoot" : Boolean,
    "InstanceType" : String,
    "LayerIds" :  [ String, ... ],
    "Os" : String,
    "RootDeviceType" : String,
    "SshKeyName" : String,
    "StackId" : String,
    "SubnetId" : String,
    "Tenancy" : String,
    "TimeBasedAutoScaling" : TimeBasedAutoScaling (p. 1233),
    "VirtualizationType" : String,
    "Volumes" : [ String, ... ]
  }
}
```

YAML

```
Type: "AWS::OpsWorks::Instance"
Properties:
AgentVersion: String
  AmiId: String
  Architecture: String
  AutoScalingType: String
  AvailabilityZone: String
  BlockDeviceMappings:
  - BlockDeviceMapping (p. 1224)
  EbsOptimized: Boolean
  ElasticIps:
  - String
  Hostname: String
  InstallUpdatesOnBoot: Boolean
  InstanceType: String
  LayerIds:
  - String
```

```
  Os: String
  RootDeviceType: String
  SshKeyName: String
  StackId: String
  SubnetId: String
  Tenancy: String
  TimeBasedAutoScaling:
  TimeBasedAutoScaling (p. 1233)
  VirtualizationType: String
  Volumes:
  - String
```

Properties

`AgentVersion`

The version of the AWS OpsWorks agent that AWS OpsWorks installs on each instance. AWS OpsWorks sends commands to the agent to performs tasks on your instances, such as starting Chef runs. For valid values, see the `AgentVersion` parameter for the CreateInstance action in the *AWS OpsWorks Stacks API Reference*.

Required: No

Type: String

Update requires: No interruption (p. 90)

`AmiId`

The ID of the custom Amazon Machine Image (AMI) to be used to create the instance. For more information about custom AMIs, see Using Custom AMIs in the *AWS OpsWorks User Guide*.

> **Note**
> If you specify this property, you must set the `Os` property to `Custom`.

Required: No

Type: String

Update requires: Updates are not supported.

`Architecture`

The instance architecture.

Required: No

Type: String

Update requires: Some interruptions (p. 90)

`AutoScalingType`

For scaling instances, the type of scaling. If you specify load-based scaling, do not specify a time-based scaling configuration. For valid values, see CreateInstance in the *AWS OpsWorks Stacks API Reference*.

Required: No

Type: String

Update requires: Replacement (p. 90)

`AvailabilityZone`

The instance Availability Zone.

Required: No

Type: String

Update requires: Replacement (p. 90)

`BlockDeviceMappings`

A list of block devices that are mapped to the AWS OpsWorks instance. For more information, see the `BlockDeviceMappings` parameter for the CreateInstance action in the *AWS OpsWorks Stacks API Reference*.

Required: No

Type: List of AWS OpsWorks Instance BlockDeviceMapping (p. 1224)

Update requires: Replacement (p. 90)

`EbsOptimized`

Whether the instance is optimized for Amazon Elastic Block Store (Amazon EBS) I/O. If you specify an Amazon EBS-optimized instance type, AWS OpsWorks enables EBS optimization by default. For more information, see Amazon EBS–Optimized Instances in the *Amazon EC2 User Guide for Linux Instances*.

Required: No

Type: Boolean

Update requires: Replacement (p. 90)

`ElasticIps`

A list of Elastic IP addresses to associate with the instance.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`Hostname`

The name of the instance host.

Required: No

Type: String

Update requires: No interruption (p. 90)

`InstallUpdatesOnBoot`

Whether to install operating system and package updates when the instance boots.

Required: No

Type: Boolean

Update requires: Some interruptions (p. 90)

`InstanceType`

The instance type, which must be supported by AWS OpsWorks. For more information, see CreateInstance in the *AWS OpsWorks Stacks API Reference*.

If you specify an Amazon EBS-optimized instance type, AWS OpsWorks enables EBS optimization by default. For more information about Amazon EBS-optimized instance types, see Amazon EBS–Optimized Instances in the *Amazon EC2 User Guide for Linux Instances*.

Required: Yes

Type: String

Update requires: Some interruptions (p. 90)

`LayerIds`

The IDs of the AWS OpsWorks layers to associate with this instance.

Required: Yes

Type: List of strings

Update requires: Some interruptions (p. 90)

`Os`

The instance operating system. For more information, see CreateInstance in the *AWS OpsWorks Stacks API Reference*.

Required: No

Type: String

Update requires: Replacement (p. 90)

`RootDeviceType`

The root device type of the instance.

Required: No

Type: String

Update requires: Replacement (p. 90)

`SshKeyName`

The SSH key name of the instance.

Required: No

Type: String

Update requires: Some interruptions (p. 90)

`StackId`

The ID of the AWS OpsWorks stack that this instance will be associated with.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`SubnetId`

The ID of the instance's subnet. If the stack is running in a VPC, you can use this parameter to override the stack's default subnet ID value and direct AWS OpsWorks to launch the instance in a different subnet.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Tenancy`

The tenancy of the instance. For more information, see the `Tenancy` parameter for the CreateInstance action in the *AWS OpsWorks Stacks API Reference*.

Required: No

Type: String

Update requires: Replacement (p. 90)

`TimeBasedAutoScaling`

The time-based scaling configuration for the instance.

Required: No

Type: AWS OpsWorks TimeBasedAutoScaling Type (p. 1233)

Update requires: Replacement (p. 90)

`VirtualizationType`

The instance's virtualization type, `paravirtual` or `hvm`.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Volumes`

A list of AWS OpsWorks volume IDs to associate with the instance. For more information, see AWS::OpsWorks::Volume (p. 870).

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "myInstance1" }
```

For the AWS OpsWorks instance `myInstance1`, `Ref` returns the AWS OpsWorks instance ID.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

- `PublicIp`

  The public IP address of the AWS OpsWorks instance, such as `192.0.2.0`.

  Note
  To use this attribute, the AWS OpsWorks instance must be in an AWS OpsWorks layer that auto-assigns public IP addresses.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Template Snippets

Basic AWS OpsWorks Instances

The following snippet creates two AWS OpsWorks instances that are associated with the `myStack` AWS OpsWorks stack and the `myLayer` AWS OpsWorks layer:

JSON

```
"myInstance1" : {
  "Type" : "AWS::OpsWorks::Instance",
  "Properties" : {
    "StackId" : {"Ref":"myStack"},
    "LayerIds" : [{"Ref":"myLayer"}],
    "InstanceType" : "m1.small"
  }
},

"myInstance2" : {
  "Type" : "AWS::OpsWorks::Instance",
  "Properties" : {
    "StackId" : {"Ref":"myStack"},
    "LayerIds" : [{"Ref":"myLayer"}],
    "InstanceType" : "m1.small"
  }
}
```

YAML

```
myInstance1:
  Type: "AWS::OpsWorks::Instance"
  Properties:
    StackId:
      Ref: "myStack"
    LayerIds:
      -
        Ref: "myLayer"
    InstanceType: "m1.small"
myInstance2:
  Type: "AWS::OpsWorks::Instance"
  Properties:
```

```
    StackId:
      Ref: "myStack"
    LayerIds:
      -
        Ref: "myLayer"
    InstanceType: "m1.small"
```

Time-based Auto Scaling Instance

In the following example, the `DBInstance` instance is online for four hours from UTC 1200-1600 on Friday, Saturday, and Sunday. The instance is offline for all other times and days.

JSON

```
"DBInstance" : {
  "Type" : "AWS::OpsWorks::Instance",
  "Properties" : {
    "AutoScalingType" : "timer",
    "StackId" : {"Ref":"Stack"},
    "LayerIds" : [{"Ref":"DBLayer"}],
    "InstanceType" : "m1.small",
    "TimeBasedAutoScaling" : {
      "Friday" : { "12" : "on", "13" : "on", "14" : "on", "15" : "on" },
      "Saturday" : { "12" : "on", "13" : "on", "14" : "on", "15" : "on" },
      "Sunday" : { "12" : "on", "13" : "on", "14" : "on", "15" : "on" }
    }
  }
}
```

YAML

```
DBInstance:
  Type: "AWS::OpsWorks::Instance"
  Properties:
    AutoScalingType: "timer"
    StackId:
      Ref: "Stack"
    LayerIds:
      - Ref: "DBLayer"
    InstanceType: "m1.small"
    TimeBasedAutoScaling:
      Friday:
        12: "on"
        13: "on"
        14: "on"
        15: "on"
      Saturday:
        12: "on"
        13: "on"
        14: "on"
        15: "on"
      Sunday:
        12: "on"
        13: "on"
        14: "on"
        15: "on"
```

More Info

- AWS::OpsWorks::Stack (p. 862)
- AWS::OpsWorks::Layer (p. 856)

• AWS::OpsWorks::App (p. 843)

AWS::OpsWorks::Layer

Creates an AWS OpsWorks layer. A layer defines, for example, which packages and applications are installed and how they are configured.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type": "AWS::OpsWorks::Layer",
  "Properties": {
    "Attributes" : { String:String },
    "AutoAssignElasticIps" : Boolean,
    "AutoAssignPublicIps" : Boolean,
    "CustomInstanceProfileArn" : String,
    "CustomJson" : JSON object,
    "CustomRecipes" : Recipes,
    "CustomSecurityGroupIds" :  [ String, ... ],
    "EnableAutoHealing" : Boolean,
    "InstallUpdatesOnBoot" : Boolean,
    "LifecycleEventConfiguration" : LifeCycleEventConfiguration,
    "LoadBasedAutoScaling" : LoadBasedAutoScaling,
    "Name" : String,
    "Packages" : [ String, ... ],
    "Shortname" : String,
    "StackId" : String,
    "Type" : String,
    "VolumeConfigurations" : [ VolumeConfiguration, ... ]
  }
}
```

YAML

```
Type: "AWS::OpsWorks::Layer"
Properties:
  Attributes:
    String:String
  AutoAssignElasticIps: Boolean
  AutoAssignPublicIps: Boolean
  CustomInstanceProfileArn: String
  CustomRecipes:
    Recipes
  CustomJson:
    JSON object
  CustomSecurityGroupIds:
```

```
    - String
  EnableAutoHealing: Boolean
  InstallUpdatesOnBoot: Boolean
  LifecycleEventConfiguration:
    LifeCycleEventConfiguration
  LoadBasedAutoScaling:
    LoadBasedAutoScaling
  Name: String
  Packages:
    - String
  Shortname: String
  StackId: String
  Type: String
  VolumeConfigurations:
    - VolumeConfiguration
```

Properties

`Attributes`

One or more user-defined key-value pairs to be added to the stack attributes bag.

Required: No

Type: A list of key-value pairs

Update requires: No interruption (p. 90)

`AutoAssignElasticIps`

Whether to automatically assign an Elastic IP address to Amazon EC2 instances in this layer.

Required: Yes

Type: Boolean

Update requires: No interruption (p. 90)

`AutoAssignPublicIps`

For AWS OpsWorks stacks that are running in a VPC, whether to automatically assign a public IP address to Amazon EC2 instances in this layer.

Required: Yes

Type: Boolean

Update requires: No interruption (p. 90)

`CustomInstanceProfileArn`

The Amazon Resource Name (ARN) of an IAM instance profile that is to be used for the Amazon EC2 instances in this layer.

Required: No

Type: String

Update requires: No interruption (p. 90)

`CustomJson`

A custom stack configuration and deployment attributes that AWS OpsWorks installs on the layer's instances. For more information, see the `CustomJson` parameter for the CreateLayer action in the *AWS OpsWorks Stacks API Reference*.

Required: No

Type: JSON object

`CustomRecipes`

Custom event recipes for this layer.

Required: No

Type: AWS OpsWorks Recipes Type (p. 1227)

Update requires: No interruption (p. 90)

`CustomSecurityGroupIds`

Custom security group IDs for this layer.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`EnableAutoHealing`

Whether to automatically heal Amazon EC2 instances that have become disconnected or timed out.

Required: Yes

Type: Boolean

Update requires: No interruption (p. 90)

`InstallUpdatesOnBoot`

Whether to install operating system and package updates when the instance boots.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`LifecycleEventConfiguration`

The lifecycle events for the AWS OpsWorks layer.

Required: No

Type: AWS OpsWorks Layer LifeCycleConfiguration (p. 1222)

Update requires: No interruption (p. 90)

`LoadBasedAutoScaling`

The load-based scaling configuration for the AWS OpsWorks layer.

Required: No

Type: AWS OpsWorks LoadBasedAutoScaling Type (p. 1223)

Update requires: No interruption (p. 90)

`Name`

The AWS OpsWorks layer name.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`Packages`

The packages for this layer.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`Shortname`

The layer short name, which is used internally by AWS OpsWorks and by Chef recipes. The short name is also used as the name for the directory where your app files are installed.

The name can have a maximum of 200 characters, which are limited to the alphanumeric characters, '-', '_', and '.'.

> **Important**
> If you update a property that requires the layer to be replaced, you must specify a new short name. You cannot have multiple layers with the same short name.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`StackId`

The ID of the AWS OpsWorks stack that this layer will be associated with.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Type`

The layer type. A stack cannot have more than one layer of the same type, except for the `custom` type. You can have any number of `custom` types. For more information, see CreateLayer in the *AWS OpsWorks Stacks API Reference*.

> **Important**
> If you update a property that requires the layer to be replaced, you must specify a new type unless you have a `custom` type. You can have any number of `custom` types.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`VolumeConfigurations`

Describes the Amazon EBS volumes for this layer.

Required: No

Type: A list of AWS OpsWorks VolumeConfiguration Type (p. 1234)

Update requires: Some interruptions (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "myLayer" }
```

For the AWS OpsWorks layer `myLayer`, `Ref` returns the AWS OpsWorks layer ID.

For more information about using the `Ref` function, see Ref (p. 1343).

Template Snippet

AWS OpsWorks PHP Layer

The following snippet creates an AWS OpsWorks PHP layer that is associated with the `myStack` AWS OpsWorks stack. The layer is dependent on the `myApp` AWS OpsWorks application.

JSON

```
"myLayer": {
  "Type": "AWS::OpsWorks::Layer",
  "DependsOn": "myApp",
  "Properties": {
    "StackId": {"Ref": "myStack"},
    "Type": "php-app",
    "Shortname" : "php-app",
    "EnableAutoHealing" : "true",
    "AutoAssignElasticIps" : "false",
    "AutoAssignPublicIps" : "true",
    "Name": "MyPHPApp"
  }
}
```

YAML

```
myLayer:
  Type: "AWS::OpsWorks::Layer"
  DependsOn: "myApp"
  Properties:
    StackId:
      Ref: "myStack"
    Type: "php-app"
    Shortname: "php-app"
    EnableAutoHealing: "true"
    AutoAssignElasticIps: "false"
    AutoAssignPublicIps: "true"
    Name: "MyPHPApp"
```

Load-based Auto Scaling Layer

The following snippet creates a load-based automatic scaling AWS OpsWorks PHP layer that is associated with the `myStack` AWS OpsWorks stack.

JSON

```
"myLayer": {
  "Type": "AWS::OpsWorks::Layer",
  "DependsOn": "myApp",
  "Properties": {
    "StackId": {"Ref": "myStack"},
    "Type": "php-app",
    "Shortname" : "php-app",
    "EnableAutoHealing" : "true",
    "AutoAssignElasticIps" : "false",
    "AutoAssignPublicIps" : "true",
    "Name": "MyPHPApp",
    "LoadBasedAutoScaling" : {
      "Enable" : "true",
      "UpScaling" : {
        "InstanceCount" : 1,
        "ThresholdsWaitTime" : 1,
        "IgnoreMetricsTime" : 1,
        "CpuThreshold" : 70.0,
        "MemoryThreshold" : 30.0,
        "LoadThreshold" : 0.7
      },
      "DownScaling" : {
        "InstanceCount" : 1,
        "ThresholdsWaitTime" : 1,
        "IgnoreMetricsTime" : 1,
        "CpuThreshold" : 30.0,
        "MemoryThreshold" : 70.0,
        "LoadThreshold" : 0.3
      }
    }
  }
}
```

YAML

```
myLayer:
  Type: "AWS::OpsWorks::Layer"
  DependsOn: "myApp"
  Properties:
    StackId:
      Ref: "myStack"
    Type: "php-app"
    Shortname: "php-app"
    EnableAutoHealing: "true"
    AutoAssignElasticIps: "false"
    AutoAssignPublicIps: "true"
    Name: "MyPHPApp"
    LoadBasedAutoScaling:
      Enable: "true"
      UpScaling:
        InstanceCount: 1
        ThresholdsWaitTime: 1
        IgnoreMetricsTime: 1
        CpuThreshold: 70
        MemoryThreshold: 30
        LoadThreshold: 0.7
      DownScaling:
        InstanceCount: 1
        ThresholdsWaitTime: 1
        IgnoreMetricsTime: 1
        CpuThreshold: 30
        MemoryThreshold: 70
```

```
        LoadThreshold: 0.3
```

See Also

- AWS::OpsWorks::Stack (p. 862)
- AWS::OpsWorks::App (p. 843)
- AWS::OpsWorks::Instance (p. 848)

AWS::OpsWorks::Stack

Creates an AWS OpsWorks stack. An AWS OpsWorks stack represents a set of instances that you want to manage collectively, typically because they have a common purpose such as serving PHP applications.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::OpsWorks::Stack",
  "Properties" : {
    "AgentVersion" : String,
    "Attributes" : { String:String, ... },
    "ChefConfiguration" : { ChefConfiguration },
    "CloneAppIds" : [ String, ... ],
    "ClonePermissions" : Boolean,
    "ConfigurationManager" : { StackConfigurationManager },
    "CustomCookbooksSource" : { Source },
    "CustomJson" : JSON,
    "DefaultAvailabilityZone" : String,
    "DefaultInstanceProfileArn" : String,
    "DefaultOs" : String,
    "DefaultRootDeviceType" : String,
    "DefaultSshKeyName" : String,
    "DefaultSubnetId" : String,
    "EcsClusterArn" : String,
    "ElasticIps" : [ ElasticIp (p. 1231), ... ],
    "HostnameTheme" : String,
    "Name" : String,
    "RdsDbInstances" : [ RdsDbInstance (p. 1231), ... ],
    "ServiceRoleArn" : String,
    "SourceStackId" : String,
    "UseCustomCookbooks" : Boolean,
    "UseOpsworksSecurityGroups" : Boolean,
    "VpcId" : String
  }
}
```

YAML

```
Type: "AWS::OpsWorks::Stack"
Properties:
  AgentVersion: String
  Attributes:
    String:String
  ChefConfiguration:
    ChefConfiguration
  CloneAppIds:
  - String
  ClonePermissions: Boolean
  ConfigurationManager:
    StackConfigurationManager
  CustomCookbooksSource:
    Source
  CustomJson: JSON
  DefaultAvailabilityZone: String
  DefaultInstanceProfileArn: String
  DefaultOs: String
  DefaultRootDeviceType: String
  DefaultSshKeyName: String
  DefaultSubnetId: String
  EcsClusterArn: String
  ElasticIps:
  - ElasticIp (p. 1231)
  HostnameTheme: String
  Name: String
  RdsDbInstances:
  - RdsDbInstance (p. 1231)
  ServiceRoleArn: String
  SourceStackId: String
  UseCustomCookbooks: Boolean
  UseOpsworksSecurityGroups: Boolean
  VpcId: String
```

Properties

`AgentVersion`

The AWS OpsWorks agent version that you want to use. The agent communicates with the service and handles tasks such as initiating Chef runs in response to lifecycle events. For valid values, see the AgentVersion parameter for the `CreateStack` action in the *AWS OpsWorks Stacks API Reference*.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Attributes`

One or more user-defined key-value pairs to be added to the stack attributes bag.

Required: No

Type: A list of key-value pairs

Update requires: No interruption (p. 90)

`ChefConfiguration`

Describes the Chef configuration. For more information, see the CreateStack ChefConfiguration parameter in the *AWS OpsWorks Stacks API Reference*.

Note
To enable Berkshelf, you must select a Chef version in the `ConfigurationManager` property that supports Berkshelf.

Required: No

Type: AWS OpsWorks ChefConfiguration Type (p. 1222)

Update requires: No interruption (p. 90)

`CloneAppIds`

If you're cloning an AWS OpsWorks stack, a list of AWS OpsWorks application stack IDs from the source stack to include in the cloned stack.

Required: No

Type: List of strings

Update requires: Replacement (p. 90)

`ClonePermissions`

If you're cloning an AWS OpsWorks stack, indicates whether to clone the source stack's permissions.

Required: No

Type: Boolean

Update requires: Replacement (p. 90)

`ConfigurationManager`

Describes the configuration manager. When you create a stack, you use the configuration manager to specify the Chef version. For supported Chef versions, see the CreateStack ConfigurationManager parameter in the *AWS OpsWorks Stacks API Reference*.

Required: No

Type: AWS OpsWorks StackConfigurationManager Type (p. 1232)

Update requires: No interruption (p. 90)

`CustomCookbooksSource`

Contains the information required to retrieve a cookbook from a repository.

Required: No

Type: AWS OpsWorks Source Type (p. 1228)

Update requires: No interruption (p. 90)

`CustomJson`

A user-defined custom JSON object. The custom JSON is used to override the corresponding default stack configuration JSON values. For more information, see CreateStack in the *AWS OpsWorks Stacks API Reference*.

Important
AWS CloudFormation submits all JSON attributes as strings, including any Boolean or number attributes. If you have recipes that expect booleans or numbers, you must modify the recipes to accept strings and to interpret those strings as booleans or numbers.

Required: No

Type: JSON object

Update requires: No interruption (p. 90)

`DefaultAvailabilityZone`

The stack's default Availability Zone, which must be in the specified region.

Required: No

Type: String

Update requires: No interruption (p. 90)

`DefaultInstanceProfileArn`

The Amazon Resource Name (ARN) of an IAM instance profile that is the default profile for all of the stack's Amazon EC2 instances.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`DefaultOs`

The stack's default operating system. For more information, see CreateStack in the *AWS OpsWorks Stacks API Reference*.

Required: No

Type: String

Update requires: No interruption (p. 90)

`DefaultRootDeviceType`

The default root device type. This value is used by default for all instances in the stack, but you can override it when you create an instance. For more information, see CreateStack in the *AWS OpsWorks Stacks API Reference*.

Required: No

Type: String

Update requires: No interruption (p. 90)

`DefaultSshKeyName`

A default SSH key for the stack instances. You can override this value when you create or update an instance.

Required: No

Type: String

Update requires: No interruption (p. 90)

`DefaultSubnetId`

The stack's default subnet ID. All instances are launched into this subnet unless you specify another subnet ID when you create the instance.

Required: Conditional. If you specify the `VpcId` property, you must specify this property.

Type: String

Update requires: No interruption (p. 90)

`EcsClusterArn`

The Amazon Resource Name (ARN) of the Amazon EC2 Container Service (Amazon ECS) cluster to register with the AWS OpsWorks stack.

> **Note**
> If you specify a cluster that's registered with another AWS OpsWorks stack, AWS CloudFormation deregisters the existing association before registering the cluster.

Required: No

Type: String

Update requires: No interruption (p. 90)

`ElasticIps`

A list of Elastic IP addresses to register with the AWS OpsWorks stack.

> **Note**
> If you specify an IP address that's registered with another AWS OpsWorks stack, AWS CloudFormation deregisters the existing association before registering the IP address.

Required: No

Type: List of AWS OpsWorks Stack ElasticIp (p. 1231)

Update requires: No interruption (p. 90)

`HostnameTheme`

The stack's host name theme, with spaces replaced by underscores. The theme is used to generate host names for the stack's instances. For more information, see CreateStack in the *AWS OpsWorks Stacks API Reference*.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Name`

The name of the AWS OpsWorks stack.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`RdsDbInstances`

The Amazon Relational Database Service (Amazon RDS) DB instance to register with the AWS OpsWorks stack.

> **Note**
> If you specify a DB instance that's registered with another AWS OpsWorks stack, AWS CloudFormation deregisters the existing association before registering the DB instance.

Required: No

Type: List of AWS OpsWorks Stack RdsDbInstance (p. 1231)

Update requires: No interruption (p. 90)

`ServiceRoleArn`

The AWS Identity and Access Management (IAM) role that AWS OpsWorks uses to work with AWS resources on your behalf. You must specify an Amazon Resource Name (ARN) for an existing IAM role.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`SourceStackId`

If you're cloning an AWS OpsWorks stack, the stack ID of the source AWS OpsWorks stack to clone.

Required: No

Type: String

Update requires: Replacement (p. 90)

`UseCustomCookbooks`

Whether the stack uses custom cookbooks.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`UseOpsworksSecurityGroups`

Whether to associate the AWS OpsWorks built-in security groups with the stack's layers.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`VpcId`

The ID of the VPC that the stack is to be launched into, which must be in the specified region. All instances are launched into this VPC. If you specify this property, you must specify the `DefaultSubnetId` property.

Required: No

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "myStack" }
```

For the AWS OpsWorks stack `myStack`, `Ref` returns the AWS OpsWorks stack ID.

For more information about using the `Ref` function, see Ref (p. 1343).

Template Snippet

The following snippet creates an AWS OpsWorks stack that uses the default service role and Amazon EC2 role, which are created after you use AWS OpsWorks for the first time:

JSON

```
"myStack" : {
  "Type" : "AWS::OpsWorks::Stack",
  "Properties" : {
    "Name" : {"Ref":"OpsWorksStackName"},
    "ServiceRoleArn" : { "Fn::Join": ["", ["arn:aws:iam::", {"Ref":"AWS::AccountId"},
 ":role/aws-opsworks-service-role"]] },
    "DefaultInstanceProfileArn" : { "Fn::Join": ["", ["arn:aws:iam::",
 {"Ref":"AWS::AccountId"}, ":instance-profile/aws-opsworks-ec2-role"]] },
    "DefaultSshKeyName" : {"Ref":"KeyName"}
  }
}
```

YAML

```
myStack:
  Type: "AWS::OpsWorks::Stack"
  Properties:
    Name:
      Ref: "OpsWorksStackName"
    ServiceRoleArn:
      Fn::Join:
        - ""
        -
          - "arn:aws:iam::"
          -
            Ref: "AWS::AccountId"
          - ":role/aws-opsworks-service-role"
    DefaultInstanceProfileArn:
      Fn::Join:
        - ""
        -
          - "arn:aws:iam::"
          -
            Ref: "AWS::AccountId"
          - ":instance-profile/aws-opsworks-ec2-role"
    DefaultSshKeyName:
      Ref: "KeyName"
```

Additional Information

- For a complete sample AWS OpsWorks template, see AWS OpsWorks Template Snippets (p. 334).
- AWS::OpsWorks::Layer (p. 856)
- AWS::OpsWorks::App (p. 843)
- AWS::OpsWorks::Instance (p. 848)

AWS::OpsWorks::UserProfile

The `AWS::OpsWorks::UserProfile` configures SSH access for users who require access to instances in an AWS OpsWorks stack.

Topics

Syntax

JSON

```
{
  "Type" : "AWS::OpsWorks::UserProfile",
  "Properties" : {
    "AllowSelfManagement" : Boolean,
    "IamUserArn" : String,
    "SshPublicKey" : String
  }
}
```

YAML

```
Type: "AWS::OpsWorks::UserProfile"
Properties:
  AllowSelfManagement: Boolean
  IamUserArn: String
  SshPublicKey: String
```

Properties

`AllowSelfManagement`

Indicates whether users can use the AWS OpsWorks **My Settings** page to specify their own SSH public key. For more information, see Setting an IAM User's Public SSH Key in the *AWS OpsWorks User Guide*.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`IamUserArn`

The Amazon Resource Name (ARN) of the AWS Identity and Access Management (IAM) user to associate with this configuration.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`SshPublicKey`

The public SSH key that is associated with the IAM user. The IAM user must have or be given the corresponding private key to access instances.

Required: No

Type: String

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the IAM user ARN, such as `arn:aws:iam::123456789012:user/opsworksuser`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example registers a public key to the `testUser` IAM user. The user can also use self-management to specify his or her own public key.

JSON

```
"userProfile": {
  "Type": "AWS::OpsWorks::UserProfile",
  "Properties": {
    "IamUserArn": {
      "Fn::GetAtt": ["testUser", "Arn"]
    },
    "AllowSelfManagement": "true",
    "SshPublicKey": "xyz1234567890"
  }
}
```

YAML

```
userProfile:
  Type: AWS::OpsWorks::UserProfile
  Properties:
    IamUserArn: !GetAtt [testUser, Arn]
    AllowSelfManagement: 'true'
    SshPublicKey: xyz1234567890
```

AWS::OpsWorks::Volume

The `AWS::OpsWorks::Volume` resource registers an Amazon Elastic Block Store (Amazon EBS) volume with an AWS OpsWorks stack.

Topics

- Example (p. 872)

Syntax

JSON

```
{
  "Type" : "AWS::OpsWorks::Volume",
  "Properties" : {
    "Ec2VolumeId" : String,
    "MountPoint" : String,
    "Name" : String,
    "StackId" : String
  }
}
```

YAML

```
Type: "AWS::OpsWorks::Volume"
Properties:
  Ec2VolumeId: String
  MountPoint: String
  Name: String
  StackId: String
```

Properties

`Ec2VolumeId`

The ID of the Amazon EBS volume to register with the AWS OpsWorks stack.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`MountPoint`

The mount point for the Amazon EBS volume, such as `/mnt/disk1`.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Name`

A name for the Amazon EBS volume.

Required: No

Type: String

Update requires: No interruption (p. 90)

`StackId`

The ID of the AWS OpsWorks stack that AWS OpsWorks registers the volume to.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the AWS OpsWorks volume ID, such as `1ab23cd4-92ff-4501-b37c-example`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example registers the `ec2volume` volume with the `opsworksstack` stack, both of which are declared elsewhere in the same template.

JSON

```
"opsworksVolume": {
  "Type": "AWS::OpsWorks::Volume",
  "Properties": {
    "Ec2VolumeId": { "Ref": "ec2volume" },
    "MountPoint": "/dev/sdb",
    "Name": "testOpsWorksVolume",
    "StackId": { "Ref": "opsworksstack" }
  }
}
```

YAML

```
opsworksVolume:
  Type: AWS::OpsWorks::Volume
  Properties:
    Ec2VolumeId: !Ref 'ec2volume'
    MountPoint: /dev/sdb
    Name: testOpsWorksVolume
    StackId: !Ref 'opsworksstack'
```

AWS::RDS::DBCluster

The `AWS::RDS::DBCluster` resource creates a cluster, such as an Aurora for Amazon RDS (Amazon Aurora) DB cluster. Amazon Aurora is a fully managed, MySQL-compatible, relational database engine. For more information, see Aurora on Amazon RDS in the *Amazon Relational Database Service User Guide*.

> **Note**
> Currently, you can create this resource only in regions in which Amazon Aurora is supported.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::RDS::DBCluster",
  "Properties" :
  {
    "AvailabilityZones" : [ String, ... ],
    "BackupRetentionPeriod" : Integer,
    "DatabaseName" : String,
    "DBClusterParameterGroupName" : String,
    "DBSubnetGroupName" : String,
    "Engine" : String,
    "EngineVersion" : String,
    "KmsKeyId" : String,
    "MasterUsername" : String,
    "MasterUserPassword" : String,
    "Port" : Integer,
    "PreferredBackupWindow" : String,
    "PreferredMaintenanceWindow" : String,
    "SnapshotIdentifier" : String,
    "StorageEncrypted" : Boolean,
    "Tags" : [ Resource Tag, ... ],
    "VpcSecurityGroupIds" : [ String, ... ]
  }
}
```

YAML

```
Type: "AWS::RDS::DBCluster"
Properties:
  AvailabilityZones:
    - String
  BackupRetentionPeriod: Integer
  DatabaseName: String
  DBClusterParameterGroupName: String
  DBSubnetGroupName: String
  Engine: String
  EngineVersion: String
  KmsKeyId: String
  MasterUsername: String
  MasterUserPassword: String
  Port: Integer
  PreferredBackupWindow: String
  PreferredMaintenanceWindow: String
  SnapshotIdentifier: String
  StorageEncrypted: Boolean
  Tags:
    Resource Tag
  VpcSecurityGroupIds:
    - String
```

Properties

`AvailabilityZones`

A list of Availability Zones (AZs) in which DB instances in the cluster can be created.

Required: No

Type: String

Update requires: Replacement (p. 90)

`BackupRetentionPeriod`

The number of days for which automatic backups are retained. For more information, see CreateDBCluster in the *Amazon Relational Database Service API Reference*.

Required: No

Type: Integer

Update requires: No interruption (p. 90) or some interruptions (p. 90). For more information, see ModifyDBInstance in the *Amazon Relational Database Service API Reference*.

`DatabaseName`

The name of your database. You can specify a name of up to eight alpha-numeric characters. If you do not provide a name, Amazon Relational Database Service (Amazon RDS) won't create a database in this DB cluster.

Required: No

Type: String

Update requires: Replacement (p. 90)

`DBClusterParameterGroupName`

The name of the DB cluster parameter group to associate with this DB cluster. For the default value, see the `DBClusterParameterGroupName` parameter of the CreateDBCluster action in the *Amazon Relational Database Service API Reference*.

Required: No

Type: String

Update requires: Some interruptions (p. 90)

`DBSubnetGroupName`

A DB subnet group that you want to associate with this DB cluster.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Engine`

The name of the database engine that you want to use for this DB cluster.

For valid values, see the `Engine` parameter of the CreateDBCluster action in the *Amazon Relational Database Service API Reference*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`EngineVersion`

The version number of the database engine that you want to use.

Required: No

Type: String

Update requires: Replacement (p. 90)

`KmsKeyId`

The Amazon Resource Name (ARN) of the AWS Key Management Service master key that is used to encrypt the database instances in the DB cluster, such as `arn:aws:kms:us-east-1:012345678910:key/abcd1234-a123-456a-a12b-a123b4cd56ef`. If you enable the `StorageEncrypted` property but don't specify this property, the default master key is used. If you specify this property, you must set the `StorageEncrypted` property to `true`.

If you specify the `SnapshotIdentifier`, do not specify this property. The value is inherited from the snapshot DB cluster.

Required: No

Type: String

Update requires: Replacement (p. 90).

`MasterUsername`

The master user name for the DB instance.

Required: Conditional. You must specify this property unless you specify the `SnapshotIdentifier` property. In that case, do not specify this property.

Type: String

Update requires: Replacement (p. 90).

`MasterUserPassword`

The password for the master database user.

Required: Conditional. You must specify this property unless you specify the `SnapshotIdentifier` property. In that case, do not specify this property.

Type: String

Update requires: No interruption (p. 90)

`Port`

The port number on which the DB instances in the cluster can accept connections.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`PreferredBackupWindow`

if automated backups are enabled (see the `BackupRetentionPeriod` property), the daily time range in UTC during which you want to create automated backups.

For valid values, see the `PreferredBackupWindow` parameter of the CreateDBInstance action in the *Amazon Relational Database Service API Reference*.

Required: No

Type: String

Update requires: No interruption (p. 90)

`PreferredMaintenanceWindow`

The weekly time range (in UTC) during which system maintenance can occur.

For valid values, see the `PreferredMaintenanceWindow` parameter of the CreateDBInstance action in the *Amazon Relational Database Service API Reference*.

Required: No

Type: String

Update requires: No interruption (p. 90) or some interruptions (p. 90). For more information, see ModifyDBInstance in the *Amazon Relational Database Service API Reference*.

`SnapshotIdentifier`

The identifier for the DB cluster snapshot from which you want to restore.

Required: No

Type: String

Update requires: Replacement (p. 90)

`StorageEncrypted`

Indicates whether the DB instances in the cluster are encrypted.

If you specify the `SnapshotIdentifier` property, do not specify this property. The value is inherited from the snapshot DB cluster.

Required: Conditional. If you specify the `KmsKeyId` property, you must enable encryption.

Type: Boolean

Update requires: Replacement (p. 90).

`Tags`

The tags that you want to attach to this DB cluster.

Required: No

Type: A list of resource tags (p. 1236)

Update requires: No interruption (p. 90)

`VpcSecurityGroupIds`

A list of VPC security groups to associate with this DB cluster.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

- **Endpoint.Address**

  The connection endpoint for the DB cluster. For example: `mystack-mydbcluster-1apw1j4phylrk.cg034hpkmmjt.us-east-1.rds.amazonaws.com`.
- **Endpoint.Port**

  The number of the port on which the DB cluster accepts connections, such as `3306`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Example

The following snippet creates an Amazon Aurora DB cluster and adds two DB instances to it. Because Amazon RDS automatically assigns a writer and reader DB instances in the cluster, use the cluster endpoint to read and write data, not the individual DB instance endpoints.

JSON

```
"RDSCluster" : {
  "Type" : "AWS::RDS::DBCluster",
  "Properties" : {
    "MasterUsername" : { "Ref" : "username" },
    "MasterUserPassword" : { "Ref" : "password" },
    "Engine" : "aurora",
    "DBSubnetGroupName" : { "Ref" : "DBSubnetGroup" },
    "DBClusterParameterGroupName" : { "Ref" : "RDSDBClusterParameterGroup" }
  }
},
"RDSDBInstance1" : {
  "Type" : "AWS::RDS::DBInstance",
  "Properties" : {
    "DBSubnetGroupName" : {
      "Ref" : "DBSubnetGroup"
    },
    "DBParameterGroupName" :{"Ref": "RDSDBParameterGroup"},
    "Engine" : "aurora",
    "DBClusterIdentifier" : {
      "Ref" : "RDSCluster"
    },
    "PubliclyAccessible" : "true",
    "AvailabilityZone" : { "Fn::GetAtt" : [ "Subnet1", "AvailabilityZone" ] },
    "DBInstanceClass" : "db.r3.xlarge"
  }
},
"RDSDBInstance2" : {
  "Type" : "AWS::RDS::DBInstance",
  "Properties" : {
```

```
    "DBSubnetGroupName" : {
      "Ref" : "DBSubnetGroup"
    },
    "DBParameterGroupName" :{"Ref": "RDSDBParameterGroup"},
    "Engine" : "aurora",
    "DBClusterIdentifier" : {
      "Ref" : "RDSCluster"
    },
    "PubliclyAccessible" : "true",
    "AvailabilityZone" : { "Fn::GetAtt" : [ "Subnet2", "AvailabilityZone" ] },
    "DBInstanceClass" : "db.r3.xlarge"
  }
},
"RDSDBClusterParameterGroup" : {
  "Type": "AWS::RDS::DBClusterParameterGroup",
  "Properties" : {
    "Description" : "CloudFormation Sample Aurora Cluster Parameter Group",
    "Family" : "aurora5.6",
    "Parameters" : {
      "time_zone" : "US/Eastern"
    }
  }
},
"RDSDBParameterGroup": {
  "Type": "AWS::RDS::DBParameterGroup",
  "Properties" : {
    "Description" : "CloudFormation Sample Aurora Parameter Group",
    "Family" : "aurora5.6",
    "Parameters" : {
      "sql_mode": "IGNORE_SPACE"
    }
  }
}
```

YAML

```
RDSCluster:
  Type: AWS::RDS::DBCluster
  Properties:
    MasterUsername:
      Ref: username
    MasterUserPassword:
      Ref: password
    Engine: aurora
    DBSubnetGroupName:
      Ref: DBSubnetGroup
    DBClusterParameterGroupName:
      Ref: RDSDBClusterParameterGroup
RDSDBInstance1:
  Type: AWS::RDS::DBInstance
  Properties:
    DBSubnetGroupName:
      Ref: DBSubnetGroup
    DBParameterGroupName:
      Ref: RDSDBParameterGroup
    Engine: aurora
    DBClusterIdentifier:
      Ref: RDSCluster
    PubliclyAccessible: 'true'
    AvailabilityZone:
      Fn::GetAtt:
      - Subnet1
      - AvailabilityZone
    DBInstanceClass: db.r3.xlarge
```

```
RDSDBInstance2:
  Type: AWS::RDS::DBInstance
  Properties:
    DBSubnetGroupName:
      Ref: DBSubnetGroup
    DBParameterGroupName:
      Ref: RDSDBParameterGroup
    Engine: aurora
    DBClusterIdentifier:
      Ref: RDSCluster
    PubliclyAccessible: 'true'
    AvailabilityZone:
      Fn::GetAtt:
      - Subnet2
      - AvailabilityZone
    DBInstanceClass: db.r3.xlarge
RDSDBClusterParameterGroup:
  Type: AWS::RDS::DBClusterParameterGroup
  Properties:
    Description: CloudFormation Sample Aurora Cluster Parameter Group
    Family: aurora5.6
    Parameters:
      time_zone: US/Eastern
RDSDBParameterGroup:
  Type: AWS::RDS::DBParameterGroup
  Properties:
    Description: CloudFormation Sample Aurora Parameter Group
    Family: aurora5.6
    Parameters:
      sql_mode: IGNORE_SPACE
```

AWS::RDS::DBClusterParameterGroup

The `AWS::RDS::DBClusterParameterGroup` resource creates a new Amazon Relational Database Service (Amazon RDS) database (DB) cluster parameter group. For more information about DB cluster parameter groups, see Appendix: DB Cluster and DB Instance Parameters in the *Amazon Relational Database Service User Guide*.

> **Note**
> Applying a parameter group to a DB cluster might require instances to reboot, resulting in a database outage while the instances reboot.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::RDS::DBClusterParameterGroup",
  "Properties" : {
    "Description" : String,
    "Family" : String,
```

```
    "Parameters" : DBParameters,
    "Tags" : [ Resource Tag, ... ]
  }
}
```

YAML

```
Type: "AWS::RDS::DBClusterParameterGroup"
Properties:
  Description: String
  Family: String
  Parameters: DBParameters
  Tags:
    Resource Tag
```

Properties

`Description`

A friendly description for this DB cluster parameter group.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Family`

The database family of this DB cluster parameter group, such as `aurora5.6`.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Parameters`

The parameters to set for this DB cluster parameter group. For a list of parameter keys, see Appendix: DB Cluster and DB Instance Parameters in the *Amazon Relational Database Service User Guide*.

Changes to dynamic parameters are applied immediately. Changes to static parameters require a reboot without failover to the DB instance that is associated with the parameter group before the change can take effect.

Required: Yes

Type: A JSON object consisting of string key-value pairs, as shown in the following example:

```
"Parameters" : {
   "Key1" : "Value1",
   "Key2" : "Value2",
   "Key3" : "Value3"
}
```

Update requires: No interruption (p. 90) or some interruptions (p. 90), depending on the parameters that you update.

`Tags`

The tags that you want to attach to this parameter group.

Required: No

Type: A list of resource tags (p. 1236)

Update requires: Updates are not supported.

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name..

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following snippet creates a parameter group that sets the character set database to UTF32:

JSON

```
"RDSDBClusterParameterGroup" : {
  "Type" : "AWS::RDS::DBClusterParameterGroup",
  "Properties" : {
    "Parameters" : {
      "character_set_database" : "utf32"
    },
    "Family" : "aurora5.6",
    "Description" : "A sample parameter group"
  }
}
```

YAML

```
RDSDBClusterParameterGroup:
  Type: "AWS::RDS::DBClusterParameterGroup"
  Properties:
    Parameters:
      character_set_database: "utf32"
    Family: "aurora5.6"
    Description: "A sample parameter group"
```

AWS::RDS::DBInstance

The `AWS::RDS::DBInstance` type creates an Amazon Relational Database Service (Amazon RDS) DB instance. For detailed information about configuring RDS DB instances, see CreateDBInstance.

> **Important**
> If a DB instance is deleted or replaced during an update, AWS CloudFormation deletes all automated snapshots. However, it retains manual DB snapshots. During an update that requires replacement, you can apply a stack policy to prevent DB instances from being replaced. For more information, see Prevent Updates to Stack Resources (p. 110).

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::RDS::DBInstance",
  "Properties" :
  {
    "AllocatedStorage (p. 883)" : String,
    "AllowMajorVersionUpgrade" : Boolean,
    "AutoMinorVersionUpgrade (p. 884)" : Boolean,
    "AvailabilityZone (p. 884)" : String,
    "BackupRetentionPeriod (p. 884)" : String,
    "CharacterSetName" : String,
    "CopyTagsToSnapshot" : Boolean,
    "DBClusterIdentifier" : String,
    "DBInstanceClass (p. 885)" : String,
    "DBInstanceIdentifier" : String,
    "DBName (p. 885)" : String,
    "DBParameterGroupName (p. 886)" : String,
    "DBSecurityGroups (p. 886)" : [ String, ... ],
    "DBSnapshotIdentifier (p. 887)" : String,
    "DBSubnetGroupName (p. 887)" : String,
    "Domain" : String,
    "DomainIAMRoleName" : String,
    "Engine (p. 888)" : String,
    "EngineVersion (p. 888)" : String,
    "Iops (p. 888)" : Number,
    "KmsKeyId" : String,
    "LicenseModel (p. 889)" : String,
    "MasterUsername (p. 889)" : String,
    "MasterUserPassword (p. 889)" : String,
    "MonitoringInterval (p. 889)" : Integer,
    "MonitoringRoleArn (p. 889)" : String,
    "MultiAZ (p. 890)" : Boolean,
    "OptionGroupName" : String,
    "Port (p. 890)" : String,
    "PreferredBackupWindow (p. 890)" : String,
    "PreferredMaintenanceWindow (p. 890)" : String,
    "PubliclyAccessible" : Boolean,
    "SourceDBInstanceIdentifier" : String,
    "StorageEncrypted" : Boolean,
    "StorageType" : String,
    "Tags (p. 892)" : [ Resource Tag, ... ],
    "Timezone" : String,
    "VPCSecurityGroups (p. 892)" : [ String, ... ]
  }
}
```

YAML

```
Type: "AWS::RDS::DBInstance"
Properties:
```

```
  AllocatedStorage (p. 883): String
  AllowMajorVersionUpgrade: Boolean
  AutoMinorVersionUpgrade (p. 884): Boolean
  AvailabilityZone (p. 884): String
  BackupRetentionPeriod (p. 884): String
  CharacterSetName: String
  CopyTagsToSnapshot: Boolean
  DBClusterIdentifier: String
  DBInstanceClass (p. 885): String
  DBInstanceIdentifier: String
  DBName (p. 885): String
  DBParameterGroupName (p. 886): String
  DBSecurityGroups (p. 886):
    - String
  DBSnapshotIdentifier (p. 887): String
  DBSubnetGroupName (p. 887): String
  Domain: String
  DomainIAMRoleName: String
  Engine (p. 888): String
  EngineVersion (p. 888): String
  Iops (p. 888): Number
  KmsKeyId: String
  LicenseModel (p. 889): String
  MasterUsername (p. 889): String
  MasterUserPassword (p. 889): String
  MonitoringInterval (p. 889): Integer
  MonitoringRoleArn (p. 889): String
  MultiAZ (p. 890): Boolean
  OptionGroupName: String
  Port (p. 890): String
  PreferredBackupWindow (p. 890): String
  PreferredMaintenanceWindow (p. 890): String
  PubliclyAccessible: Boolean
  SourceDBInstanceIdentifier: String
  StorageEncrypted: Boolean
  StorageType: String
  Tags (p. 892):
    Resource Tag
  Timezone: String
  VPCSecurityGroups (p. 892):
    - String
```

Properties

`AllocatedStorage`

The allocated storage size, specified in gigabytes (GB).

If any value is set in the `Iops` parameter, `AllocatedStorage` must be at least 100 GB, which corresponds to the minimum `Iops` value of 1,000. If you increase the `Iops` value (in 1,000 IOPS increments), then you must also increase the `AllocatedStorage` value (in 100 GB increments).

Required: Conditional. This property is required unless you specify the `DBClusterIdentifier` property. In that case, do not specify this property.

Type: String

Update requires: No interruption (p. 90)

`AllowMajorVersionUpgrade`

If you update the `EngineVersion` property to a version that's different from the DB instance's current major version, set this property to `true`. For more information, see ModifyDBInstance in the *Amazon Relational Database Service API Reference*.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`AutoMinorVersionUpgrade`

Indicates that minor engine upgrades are applied automatically to the DB instance during the maintenance window. The default value is `true`.

Required: No

Type: Boolean

Update requires: No interruption (p. 90) or some interruptions (p. 90). For more information, see ModifyDBInstance in the *Amazon Relational Database Service API Reference*.

`AvailabilityZone`

The name of the Availability Zone where the DB instance is located. You cannot set the `AvailabilityZone` parameter if the `MultiAZ` parameter is set to `true`.

Required: No

Type: String

Update requires: Replacement (p. 90)

`BackupRetentionPeriod`

The number of days during which automatic DB snapshots are retained.

Important
If this DB instance is deleted or replaced during an update, AWS CloudFormation deletes all automated snapshots. However, it retains manual DB snapshots.

Required: No

Type: String

Update requires: No interruption (p. 90) or some interruptions (p. 90). For more information, see ModifyDBInstance in the *Amazon Relational Database Service API Reference*.

`CharacterSetName`

For supported engines, specifies the character set to associate with the DB instance. For more information, see Appendix: Oracle Character Sets Supported in Amazon RDS in the *Amazon Relational Database Service User Guide*.

If you specify the `DBSnapshotIdentifier` or `SourceDBInstanceIdentifier` property, do not specify this property. The value is inherited from the snapshot or source DB instance.

Required: No

Type: String

Update requires: Replacement (p. 90)

`CopyTagsToSnapshot`

Indicates whether to copy all of the user-defined tags from the DB instance to snapshots of the DB instance. By default, Amazon RDS doesn't copy tags to snapshots. Amazon RDS doesn't copy tags with the `aws::` prefix unless it's the DB instance's final snapshot (the snapshot when you delete the DB instance).

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`DBClusterIdentifier`

The name of an existing DB cluster that this instance will be associated with. If you specify this property, specify `aurora` for the `Engine` property and do not specify any of the following properties: `AllocatedStorage`, `BackupRetentionPeriod`, `CharacterSetName`, `DBSecurityGroups`, `PreferredBackupWindow`, `PreferredMaintenanceWindow`, `Port`, `SourceDBInstanceIdentifier`, or `StorageType`.

Amazon RDS assigns the first DB instance in the cluster as the primary, and additional DB instances as replicas.

Required: No

Type: String

Update requires: Replacement (p. 90)

`DBInstanceClass`

The name of the compute and memory capacity classes of the DB instance.

Required: Yes

Type: String

Update requires: Some interruptions (p. 90)

`DBInstanceIdentifier`

A name for the DB instance. If you specify a name, AWS CloudFormation converts it to lower case. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the DB instance. For more information, see Name Type (p. 1217).

Important
If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

`DBName`

The name of the DB instance that was provided at the time of creation, if one was specified. This same name is returned for the life of the DB instance.

Important
If you specify the `DBSnapshotIdentifier (p. 887)` property, AWS CloudFormation ignores this property.
If you restore DB instances from snapshots, this property doesn't apply to the MySQL, PostgreSQL, or MariaDB engines.

Required: No

Type: String

Update requires: Replacement (p. 90)

`DBParameterGroupName`

The name of an existing DB parameter group or a reference to an AWS::RDS::DBParameterGroup (p. 895) resource created in the template.

Required: No

Type: String

Update requires: No interruption (p. 90) or some interruptions (p. 90). If any of the data members of the referenced parameter group are changed during an update, the DB instance might need to be restarted, causing some interruption. If the parameter group contains static parameters, whether they were changed or not, an update triggers a reboot.

`DBSecurityGroups`

A list of the DB security groups to assign to the DB instance. The list can include both the name of existing DB security groups or references to AWS::RDS::DBSecurityGroup (p. 898) resources created in the template.

If you set `DBSecurityGroups`, you must not set VPCSecurityGroups (p. 892), and vice-versa.

Important

If you specify this property, AWS CloudFormation sends only the following properties (if specified) to Amazon RDS:

- `AllocatedStorage`
- `AutoMinorVersionUpgrade`
- `AvailabilityZone`
- `BackupRetentionPeriod`
- `CharacterSetName`
- `DbInstanceClass`
- `DbName`
- `DbParameterGroupName`
- `DbSecurityGroups`
- `DbSubnetGroupName`
- `Engine`
- `EngineVersion`
- `Iops`
- `LicenseModel`
- `MasterUsername`
- `MasterUserPassword`
- `MultiAZ`
- `OptionGroupName`
- `PreferredBackupWindow`
- `PreferredMaintenanceWindow`

All other properties are ignored. Specify a VPC security group if you want to submit other properties, such as `StorageType`, `StorageEncrypted`, or `KmsKeyId`. If you're already using the `DBSecurityGroups` property, you can't use these other properties by updating your DB instance to use a VPC security group. You must recreate the DB instance.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`DBSnapshotIdentifier`

The name or ARN of the DB snapshot used to restore the DB instance. If you are restoring from a shared manual DB snapshot, you must specify the Amazon Resource Name (ARN) of the snapshot.

By specifying this property, you can create a DB instance from the specified DB snapshot. If the `DBSnapshotIdentifier` property is an empty string or the `AWS::RDS::DBInstance` declaration has no `DBSnapshotIdentifier` property, AWS CloudFormation creates a new database. If the property contains a value (other than an empty string), AWS CloudFormation creates a database from the specified snapshot. If a snapshot with the specified name does not exist, AWS CloudFormation can't create the database and it rolls the back the stack.

Some DB instance properties are not valid when you restore from a snapshot, such as the `MasterUsername` and `MasterUserPassword` properties. For information about the properties that you can specify, see the RestoreDBInstanceFromDBSnapshot action in the *Amazon Relational Database Service API Reference*.

Important
If you specify this property, AWS CloudFormation ignores the `DBName` (p. 885) property.

Required: No

Type: String

Update requires: Replacement (p. 90)

`DBSubnetGroupName`

A DB subnet group to associate with the DB instance.

If there is no DB subnet group, then the instance is not a VPC DB instance.

For more information about using Amazon RDS in a VPC, see Using Amazon RDS with Amazon Virtual Private Cloud (VPC) in the *Amazon Relational Database Service Developer Guide*.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Domain`

For an Amazon RDS DB instance that is running Microsoft SQL Server, the Active Directory directory ID to create the instance in. Amazon RDS uses Windows Authentication to authenticate users that connect to the DB instance. For more information, see Using Windows Authentication with an Amazon RDS DB Instance Running Microsoft SQL Server in the *Amazon Relational Database Service User Guide*.

If you specify this property, you must specify a SQL Server engine for the `Engine` property.

Required: No

Type: String

Update requires: No interruption (p. 90)

`DomainIAMRoleName`

The name of an IAM role that Amazon RDS uses when calling the Directory Service APIs.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Engine`

The database engine that the DB instance uses. This property is optional when you specify the `DBSnapshotIdentifier` property to create DB instances.

For valid values, see the `Engine` parameter of the CreateDBInstance action in the *Amazon Relational Database Service API Reference*.

Required: Conditional

Type: String

Update requires: Replacement (p. 90)

`EngineVersion`

The version number of the database engine that the DB instance uses.

Required: No

Type: String

Update requires: Some interruptions (p. 90)

`Iops`

The number of I/O operations per second (IOPS) that the database provisions. The value must be equal to or greater than `1000`.

If you specify this property, you must follow the range of allowed ratios of your requested IOPS rate to the amount of storage that you allocate (IOPS to allocated storage). For example, you can provision an Oracle database instance with `1000` IOPS and `200` GB of storage (a ratio of 5:1) or specify 2000 IOPS with 200 GB of storage (a ratio of 10:1). For more information, see Amazon RDS Provisioned IOPS Storage to Improve Performance in the *Amazon Relational Database Service User Guide*.

Required: Conditional. If you specify `io1` for the `StorageType` property, you must specify this property.

Type: Number

Update requires: No interruption (p. 90)

`KmsKeyId`

The ARN of the AWS Key Management Service (AWS KMS) master key that is used to encrypt the DB instance, such as `arn:aws:kms:us-east-1:012345678910:key/abcd1234-a123-456a-a12b-a123b4cd56ef`. If you enable the `StorageEncrypted` property but don't specify this property, AWS CloudFormation uses the default master key. If you specify this property, you must set the `StorageEncrypted` property to `true`.

If you specify the `DBSnapshotIdentifier` or `SourceDBInstanceIdentifier` property, do not specify this property. The value is inherited from the snapshot or source database instance.

Note

If you specify `DBSecurityGroups`, AWS CloudFormation ignores this property. To specify both a security group and this property, you must use a VPC security group. For more information about Amazon RDS and VPC, see Using Amazon RDS with Amazon VPC in the *Amazon Relational Database Service User Guide*.

Required: No

Type: String

Update requires: Replacement (p. 90)

`LicenseModel`

The license model of the DB instance.

Required: No

Type: String

Update requires: Replacement (p. 90)

`MasterUsername`

The master user name for the DB instance.

Note
If you specify the `SourceDBInstanceIdentifier` or `DBSnapshotIdentifier` property, do not specify this property. The value is inherited from the source DB instance or snapshot.

Required: Conditional

Type: String

Update requires: Replacement (p. 90)

`MasterUserPassword`

The master password for the DB instance.

Note
If you specify the `SourceDBInstanceIdentifier` or `DBSnapshotIdentifier` property, do not specify this property. The value is inherited from the source DB instance or snapshot.

Required: Conditional

Type: String

Update requires: No interruption (p. 90)

`MonitoringInterval`

The interval, in seconds, between points when Amazon RDS collects enhanced monitoring metrics for the DB instance. To disable metrics collection, specify `0`.

For default and valid values, see the `MonitoringInterval` parameter for the CreateDBInstance action in the *Amazon Relational Database Service API Reference*.

Required: Conditional. If you specify the `MonitoringRoleArn` property, specify a value other than `0` for `MonitoringInterval`.

Type: Integer

Update requires: No interruption (p. 90) or some interruptions (p. 90). For more information, see ModifyDBInstance in the *Amazon Relational Database Service API Reference*.

`MonitoringRoleArn`

The ARN of the AWS Identity and Access Management (IAM) role that permits Amazon RDS to send enhanced monitoring metrics to Amazon CloudWatch, for example, `arn:aws:iam:123456789012:role/`

`emaccess`. For information on creating a monitoring role, see To create an IAM role for Amazon RDS Enhanced Monitoring in the *Amazon Relational Database Service User Guide*.

Required: Conditional. If you specify a value other than `0` for the `MonitoringInterval` property, specify a value for `MonitoringRoleArn`.

Type: String

Update requires: No interruption (p. 90)

`MultiAZ`

Specifies if the database instance is a multiple Availability Zone deployment. You cannot set the `AvailabilityZone` parameter if the `MultiAZ` parameter is set to true.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`OptionGroupName`

The option group that this DB instance is associated with.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Port`

The port for the instance.

Required: No

Type: String

Update requires: Replacement (p. 90)

`PreferredBackupWindow`

The daily time range during which automated backups are performed if automated backups are enabled, as determined by the `BackupRetentionPeriod` property. For valid values, see the `PreferredBackupWindow` parameter for the CreateDBInstance action in the *Amazon Relational Database Service API Reference*.

Required: No

Type: String

Update requires: No interruption (p. 90)

`PreferredMaintenanceWindow`

The weekly time range (in UTC) during which system maintenance can occur. For valid values, see the `PreferredMaintenanceWindow` parameter for the CreateDBInstance action in the *Amazon Relational Database Service API Reference*.

Note
This property applies when AWS CloudFormation initially creates the DB instance. If you use AWS CloudFormation to update the DB instance, those updates are applied immediately.

Required: No

Type: String

Update requires: No interruption (p. 90) or some interruptions (p. 90). For more information, see ModifyDBInstance in the *Amazon Relational Database Service API Reference*.

`PubliclyAccessible`

Indicates whether the DB instance is an Internet-facing instance. If you specify `true`, AWS CloudFormation creates an instance with a publicly resolvable DNS name, which resolves to a public IP address. If you specify `false`, AWS CloudFormation creates an internal instance with a DNS name that resolves to a private IP address.

The default behavior value depends on your VPC setup and the database subnet group. For more information, see the `PubliclyAccessible` parameter in CreateDBInstance in the *Amazon Relational Database Service API Reference*.

If this resource has a public IP address and is also in a VPC that is defined in the same template, you must use the `DependsOn` attribute to declare a dependency on the VPC-gateway attachment. For more information, see DependsOn Attribute (p. 1298).

Note
If you specify `DBSecurityGroups`, AWS CloudFormation ignores this property. To specify a security group and this property, you must use a VPC security group. For more information about Amazon RDS and VPC, see Using Amazon RDS with Amazon VPC in the *Amazon Relational Database Service User Guide*.

Required: No

Type: Boolean

Update requires: Replacement (p. 90)

`SourceDBInstanceIdentifier`

If you want to create a read replica DB instance, specify the ID of the source DB instance. Each DB instance can have a limited number of read replicas. For more information, see Working with Read Replicas in the *Amazon Relational Database Service Developer Guide*.

The `SourceDBInstanceIdentifier` property determines whether a DB instance is a read replica. If you remove the `SourceDBInstanceIdentifier` property from your template and then update your stack, AWS CloudFormation deletes the read replica and creates a new DB instance (not a read replica).

Important
- If you specify a source DB instance that uses VPC security groups, we recommend that you specify the `VPCSecurityGroups` property. If you don't specify the property, the read replica inherits the value of the `VPCSecurityGroups` property from the source DB when you create the replica. However, if you update the stack, AWS CloudFormation reverts the replica's `VPCSecurityGroups` property to the default value because it's not defined in the stack's template. This change might cause unexpected issues.
- Read replicas do not support deletion policies. AWS CloudFormation ignores any deletion policy that's associated with a read replica.
- If you specify `SourceDBInstanceIdentifier`, do not set the `MultiAZ` property to `true` and do not specify the `DBSnapshotIdentifier` property. You cannot deploy read replicas in multiple Availability Zones, and you cannot create a read replica from a snapshot.
- Do not set the `BackupRetentionPeriod`, `DBName`, `MasterUsername`, `MasterUserPassword`, and `PreferredBackupWindow` properties. The database attributes are inherited from the source DB instance, and backups are disabled for read replicas.
- If the source DB instance is in a different region than the read replica, specify an ARN for a valid DB instance. For more information, see Constructing a Amazon RDS Amazon Resource Name (ARN) in the *Amazon Relational Database Service User Guide*.

- For DB instances in Amazon Aurora clusters, do not specify this property. Amazon RDS automatically assigns writer and reader DB instances.

Required: No

Type: String

Update requires: Replacement (p. 90)

`StorageEncrypted`

Indicates whether the DB instance is encrypted.

If you specify the `DBClusterIdentifier`, `DBSnapshotIdentifier`, or `SourceDBInstanceIdentifier` property, do not specify this property. The value is inherited from the cluster, snapshot, or source DB instance.

Required: Conditional. If you specify the `KmsKeyId` property, you must enable encryption.

Type: Boolean

Update requires: Replacement (p. 90)

`StorageType`

The storage type associated with this DB instance.

For the default and valid values, see the `StorageType` parameter of the CreateDBInstance action in the *Amazon Relational Database Service API Reference*.

Required: No

Type: String

Update requires: Some interruptions (p. 90)

`Tags`

An arbitrary set of tags (key–value pairs) for this DB instance.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90)

`Timezone`

The time zone of the DB instance, which you can specify to match the time zone of your applications. To see which engines supports time zones, see the `Timezone` parameter for the CreateDBInstance action in the *Amazon Relational Database Service API Reference*.

Required: No

Type: String

Update requires: Replacement (p. 90)

`VPCSecurityGroups`

A list of the VPC security group IDs to assign to the DB instance. The list can include both the physical IDs of existing VPC security groups and references to AWS::EC2::SecurityGroup (p. 608) resources created in the template.

If you set `VPCSecurityGroups`, you must not set DBSecurityGroups (p. 886), and vice versa.

Important

You can migrate a DB instance in your stack from an RDS DB security group to a VPC security group, but keep the following in mind:

- You cannot revert to using an RDS security group after you establish a VPC security group membership.
- When you migrate your DB instance to VPC security groups, if your stack update rolls back because the DB instance update fails, or because an update fails in another AWS CloudFormation resource, the rollback will fail because it cannot revert to an RDS security group.
- To use the properties that are available when you use a VPC security group, you must recreate the DB instance. If you don't, AWS CloudFormation submits only the property values that are listed in the DBSecurityGroups (p. 886) property.

To avoid this situation, migrate your DB instance to using VPC security groups only when that is the only change in your stack template.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

Updating and Deleting AWS::RDS::DBInstance Resources

When properties labeled "*Update requires*: Replacement (p. 90)" are updated, AWS CloudFormation first creates a replacement DB instance, then changes references from other dependent resources to point to the replacement DB instance, and finally deletes the old DB instance.

Important

We highly recommend that you take a snapshot of the database before updating the stack. If you don't, you will lose the data when AWS CloudFormation replaces your DB instance. To preserve your data, perform the following procedure:

1. Deactivate any applications that are using the DB instance so that there is no activity on the DB instance.
2. Create a snapshot of the DB instance. For more information about creating DB snapshots, see Creating a DB snapshot.
3. If you want to restore your instance using a DB snapshot, modify the updated template with your DB instance changes and add the `DBSnapshotIdentifier` property with the ID of the DB snapshot that you want to use.
4. Update the stack.

For more information about updating other properties of this resource, see ModifyDBInstance. For more information about updating stacks, see AWS CloudFormation Stacks Updates (p. 89).

You can set a deletion policy for your DB instance to control how AWS CloudFormation handles the instance when the stack is deleted. For Amazon RDS DB instances, you can choose to *retain* the instance, to *delete* the instance, or to *create a snapshot* of the instance. For more information, see DeletionPolicy Attribute (p. 1297).

Return Values

Ref

When you provide the RDS DB instance's logical name to the `Ref` intrinsic function, `Ref` will return the `DBInstanceIdentifier`. For example: `mystack-mydb-ea5ugmfvuaxg`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

- **Endpoint.Address**

  The connection endpoint for the database. For example: `mystack-mydb-1apw1j4phylrk.cg034hpkmmjt.us-east-1.rds.amazonaws.com`.
- **Endpoint.Port**

  The port number on which the database accepts connections. For example: `3306`.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Examples

DBInstance with a set MySQL version, Tags and DeletionPolicy

This example shows how to set the MySQL version that has a `DeletionPolicy Attribute (p. 1297)` set. With the `DeletionPolicy` set to `Snapshot`, AWS CloudFormation will take a snapshot of this DB instance before deleting it during stack deletion. A tag that contains a friendly name for the database is also set.

JSON

```
"MyDB" : {
   "Type" : "AWS::RDS::DBInstance",
   "Properties" : {
      "DBName" : { "Ref" : "DBName" },
      "AllocatedStorage" : { "Ref" : "DBAllocatedStorage" },
      "DBInstanceClass" : { "Ref" : "DBInstanceClass" },
      "Engine" : "MySQL",
      "EngineVersion" : "5.5",
      "MasterUsername" : { "Ref" : "DBUser" },
      "MasterUserPassword" : { "Ref" : "DBPassword" },
      "Tags" : [ { "Key" : "Name", "Value" : "My SQL Database" } ]
   },
   "DeletionPolicy" : "Snapshot"
}
```

YAML

```
MyDB:
  Type: "AWS::RDS::DBInstance"
  Properties:
    DBName:
      Ref: "DBName"
    AllocatedStorage:
      Ref: "DBAllocatedStorage"
    DBInstanceClass:
      Ref: "DBInstanceClass"
    Engine: "MySQL"
    EngineVersion: "5.5"
    MasterUsername:
      Ref: "DBUser"
    MasterUserPassword:
```

```
      Ref: "DBPassword"
    Tags:
      -
        Key: "Name"
        Value: "My SQL Database"
  DeletionPolicy: "Snapshot"
```

DBInstance with provisioned IOPS

This example sets a provisioned IOPS value in the Iops (p. 888) property. Note that the AllocatedStorage (p. 883) property is set according to the 10:1 ratio between IOPS and GiBs of storage.

JSON

```
"MyDB" : {
   "Type" : "AWS::RDS::DBInstance",
   "Properties" : {
      "AllocatedStorage" : "100",
      "DBInstanceClass" : "db.m1.small",
      "Engine" : "MySQL",
      "EngineVersion" : "5.5",
      "Iops" : "1000",
      "MasterUsername" : { "Ref" : "DBUser" },
      "MasterUserPassword" : { "Ref" : "DBPassword" }
   }
}
```

YAML

```
MyDB:
  Type: "AWS::RDS::DBInstance"
  Properties:
    AllocatedStorage: "100"
    DBInstanceClass: "db.m1.small"
    Engine: "MySQL"
    EngineVersion: "5.5"
    Iops: "1000"
    MasterUsername:
      Ref: "DBUser"
    MasterUserPassword:
      Ref: "DBPassword"
```

AWS::RDS::DBParameterGroup

Creates a custom parameter group for an RDS database family. For more information about RDS parameter groups, see Working with DB Parameter Groups in the *Amazon Relational Database Service User Guide*.

This type can be declared in a template and referenced in the `DBParameterGroupName` parameter of AWS::RDS::DBInstance (p. 881).

> **Note**
> Applying a ParameterGroup to a DBInstance may require the instance to reboot, resulting in a database outage for the duration of the reboot.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::RDS::DBParameterGroup",
   "Properties" : {
      "Description (p. 896)" : String,
      "Family (p. 896)" : String,
      "Parameters (p. 896)" : DBParameters,
      "Tags" : [ Resource Tag, ... ]
   }
}
```

YAML

```
Type: "AWS::RDS::DBParameterGroup"
Properties:
  Description (p. 896): String
  Family (p. 896): String
  Parameters (p. 896):
    DBParameters
  Tags:
    - Resource Tag
```

Properties

`Description`

A friendly description of the RDS parameter group. For example, `"My Parameter Group"`.

Required: Yes

Type: String

Update requires: Updates are not supported.

`Family`

The database family of this RDS parameter group. For example, `"MySQL5.1"`.

Required: Yes

Type: String

Update requires: Updates are not supported.

`Parameters`

The parameters to set for this RDS parameter group.

Required: No

Type: A JSON object consisting of string key-value pairs, as shown in the following example:

```
"Parameters" : {
   "Key1" : "Value1",
   "Key2" : "Value2",
   "Key3" : "Value3"
}
```

Update requires: No interruption (p. 90) or Some interruptions (p. 90). Changes to dynamic parameters are applied immediately. During an update, if you have static parameters (whether they were changed or not), triggers AWS CloudFormation to reboot the associated DB instance without failover.

`Tags`

The tags that you want to attach to the RDS parameter group.

Required: No

Type: A list of resource tags (p. 1236).

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "MyDBParameterGroup" }
```

For the RDS::DBParameterGroup with the logical ID "MyDBParameterGroup", `Ref` will return the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following snippet creates a parameter group for an Aurora DB cluster that applies the `IGNORE_SPACE` SQL mode.

JSON

```
"RDSDBParameterGroup": {
  "Type": "AWS::RDS::DBParameterGroup",
  "Properties" : {
    "Description" : "CloudFormation Sample Parameter Group",
    "Family" : "aurora5.6",
    "Parameters" : {
      "sql_mode": "IGNORE_SPACE"
    }
  }
}
```

YAML

```
RDSDBParameterGroup:
```

```
  Type: AWS::RDS::DBParameterGroup
  Properties:
    Description: CloudFormation Sample Parameter Group
    Family: aurora5.6
    Parameters:
      sql_mode: IGNORE_SPACE
```

AWS::RDS::DBSecurityGroup

The AWS::RDS::DBSecurityGroup type is used to create or update an Amazon RDS DB Security Group. For more information about DB security groups, see Working with DB Security Groups in the *Amazon Relational Database Service Developer Guide*. For details on the settings for DB security groups, see CreateDBSecurityGroup.

> **Note**
> If you use DB security groups, the settings that you can specify for your DB instances are limited. For more information, see the DBSecurityGroups (p. 886) property.

When you specify an AWS::RDS::DBSecurityGroup as an argument to the `Ref` function, AWS CloudFormation returns the value of the `DBSecurityGroupName`.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::RDS::DBSecurityGroup",
   "Properties" :
   {
      "EC2VpcId (p. 899)" : { "Ref" : "myVPC" },
      "DBSecurityGroupIngress (p. 899)" : [ RDS Security Group Rule (p. 1239) object 1, ...
 ],
      "GroupDescription (p. 899)" : String,
      "Tags" : [ Resource Tag, ... ]
   }
}
```

YAML

```
Type: "AWS::RDS::DBSecurityGroup"
Properties:
  EC2VpcId (p. 899): String
  DBSecurityGroupIngress (p. 899):
    - RDS Security Group Rule (p. 1239)
  GroupDescription (p. 899): String
  Tags:
    - Resource Tag
```

Properties

`EC2VpcId`

The Id of VPC. Indicates which VPC this DB Security Group should belong to.
Type: String

Required: Conditional. Must be specified to create a DB Security Group for a VPC; may not be specified otherwise.

Update requires: Replacement (p. 90)

`DBSecurityGroupIngress`

Network ingress authorization for an Amazon EC2 security group or an IP address range.
Type: List of RDS Security Group Rules (p. 1239).

Required: Yes

Update requires: No interruption (p. 90)

`GroupDescription`

Description of the security group.
Type: String

Required: Yes

Update requires: Replacement (p. 90)

`Tags`

The tags that you want to attach to the Amazon RDS DB security group.

Required: No

Type: A list of resource tags (p. 1236).

Update requires: No interruption (p. 90)

Template Examples

Tip
For more RDS template examples, see Amazon RDS Template Snippets (p. 346).

Single VPC security group

This template snippet creates/updates a single VPC security group, referred to by EC2SecurityGroupName.

JSON

```
"DBSecurityGroup": {
   "Type": "AWS::RDS::DBSecurityGroup",
   "Properties": {
      "EC2VpcId" : { "Ref" : "VpcId" },
      "DBSecurityGroupIngress": [
         {"EC2SecurityGroupName": { "Ref": "WebServerSecurityGroup"}}
      ],
      "GroupDescription": "Frontend Access"
   }
}
```

YAML

```
DBSecurityGroup:
```

```
  Type: "AWS::RDS::DBSecurityGroup"
  Properties:
    EC2VpcId:
      Ref: "VpcId"
    DBSecurityGroupIngress:
      -
        EC2SecurityGroupName:
          Ref: "WebServerSecurityGroup"
    GroupDescription: "Frontend Access"
```

Multiple VPC security groups

This template snippet creates/updates multiple VPC security groups.

JSON

```
{
   "Resources" : {
      "DBinstance" : {
         "Type" : "AWS::RDS::DBInstance",
         "Properties" : {
            "DBSecurityGroups" : [ {"Ref" : "DbSecurityByEC2SecurityGroup"} ],
            "AllocatedStorage" : "5",
            "DBInstanceClass" : "db.m1.small",
            "Engine" : "MySQL",
            "MasterUsername" : "YourName",
            "MasterUserPassword" : "YourPassword"
         },
         "DeletionPolicy" : "Snapshot"
      },
      "DbSecurityByEC2SecurityGroup" : {
         "Type" : "AWS::RDS::DBSecurityGroup",
         "Properties" : {
            "GroupDescription" : "Ingress for Amazon EC2 security group",
            "DBSecurityGroupIngress" : [ {
                  "EC2SecurityGroupId" : "sg-b0ff1111",
                  "EC2SecurityGroupOwnerId" : "111122223333"
               }, {
                  "EC2SecurityGroupId" : "sg-ffd722222",
                  "EC2SecurityGroupOwnerId" : "111122223333"
               } ]
         }
      }
   }
}
```

YAML

```
Resources:
  DBinstance:
    Type: "AWS::RDS::DBInstance"
    Properties:
      DBSecurityGroups:
        -
          Ref: "DbSecurityByEC2SecurityGroup"
      AllocatedStorage: "5"
      DBInstanceClass: "db.m1.small"
      Engine: "MySQL"
      MasterUsername: "YourName"
      MasterUserPassword: "YourPassword"
    DeletionPolicy: "Snapshot"
  DbSecurityByEC2SecurityGroup:
```

```
    Type: "AWS::RDS::DBSecurityGroup"
    Properties:
      GroupDescription: "Ingress for Amazon EC2 security group"
      DBSecurityGroupIngress:
        -
          EC2SecurityGroupId: "sg-b0ff1111"
          EC2SecurityGroupOwnerId: "111122223333"
        -
          EC2SecurityGroupId: "sg-ffd722222"
          EC2SecurityGroupOwnerId: "111122223333"
```

AWS::RDS::DBSecurityGroupIngress

The AWS::RDS::DBSecurityGroupIngress type enables ingress to a DBSecurityGroup using one of two forms of authorization. First, EC2 or VPC security groups can be added to the DBSecurityGroup if the application using the database is running on EC2 or VPC instances. Second, IP ranges are available if the application accessing your database is running on the Internet. For more information about DB security groups, see Working with DB security groups

This type supports updates. For more information about updating stacks, see AWS CloudFormation Stacks Updates (p. 89).

For details about the settings for DB security group ingress, see AuthorizeDBSecurityGroupIngress.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::RDS::DBSecurityGroupIngress",
  "Properties" : {
   "CIDRIP (p. 902)": String,
   "DBSecurityGroupName (p. 902)": String,
   "EC2SecurityGroupId (p. 902)": String,
   "EC2SecurityGroupName (p. 902)": String,
   "EC2SecurityGroupOwnerId (p. 902)": String
}
```

YAML

```
Type: "AWS::RDS::DBSecurityGroupIngress"
Properties:
  CIDRIP (p. 902): String
  DBSecurityGroupName (p. 902): String
  EC2SecurityGroupId (p. 902): String
  EC2SecurityGroupName (p. 902): String
  EC2SecurityGroupOwnerId (p. 902): String
```

Properties

`CIDRIP`

The IP range to authorize.

For an overview of CIDR ranges, go to the Wikipedia Tutorial.
Type: String

Update requires: No interruption (p. 90)

`DBSecurityGroupName`

The name (ARN) of the AWS::RDS::DBSecurityGroup (p. 898) to which this ingress will be added.
Type: String
Required: Yes

Update requires: No interruption (p. 90)

`EC2SecurityGroupId`

The ID of the VPC or EC2 security group to authorize.

For VPC DB security groups, use EC2SecurityGroupId. For EC2 security groups, use EC2SecurityGroupOwnerId and either EC2SecurityGroupName or EC2SecurityGroupId.
Type: String
Required: No

Update requires: No interruption (p. 90)

`EC2SecurityGroupName`

The name of the EC2 security group to authorize.

For VPC DB security groups, use EC2SecurityGroupId. For EC2 security groups, use EC2SecurityGroupOwnerId and either EC2SecurityGroupName or EC2SecurityGroupId.
Type: String
Required: No

Update requires: No interruption (p. 90)

`EC2SecurityGroupOwnerId`

The AWS Account Number of the owner of the EC2 security group specified in the EC2SecurityGroupName parameter. The AWS Access Key ID is not an acceptable value.

For VPC DB security groups, use EC2SecurityGroupId. For EC2 security groups, use EC2SecurityGroupOwnerId and either EC2SecurityGroupName or EC2SecurityGroupId.
Type: String
Required: No

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

See Also

- AuthorizeDBSecurityGroupIngress in the *Amazon Relational Database Service API Reference*

AWS::RDS::DBSubnetGroup

The AWS::RDS::DBSubnetGroup type creates an RDS database subnet group. Subnet groups must contain at least two subnet in two different Availability Zones in the same region.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::RDS::DBSubnetGroup",
   "Properties" : {
      "DBSubnetGroupDescription (p. 903)" : String,
      "SubnetIds (p. 904)" : [ String, ... ],
      "Tags" : [ Resource Tag, ... ]
   }
}
```

YAML

```
Type: "AWS::RDS::DBSubnetGroup"
Properties:
  DBSubnetGroupDescription (p. 903): String
  SubnetIds (p. 904):
    - String
  Tags:
    - Resource Tag
```

Properties

`DBSubnetGroupDescription`

The description for the DB Subnet Group.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`SubnetIds`

The EC2 Subnet IDs for the DB Subnet Group.

Required: Yes

Type: List of strings

Update requires: No interruption (p. 90)

`Tags`

The tags that you want to attach to the RDS database subnet group.

Required: No

Type: A list of resource tags (p. 1236) in key-value format.

Update requires: No interruption (p. 90)

Return Value

Ref

When you pass the logical ID of an `AWS::RDS::DBSubnetGroup` resource to the intrinsic `Ref` function, the function returns the name of the DB subnet group, such as `mystack-mydbsubnetgroup-0a12bc456789de0fg`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

JSON

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "myDBSubnetGroup" : {
         "Type" : "AWS::RDS::DBSubnetGroup",
         "Properties" : {
            "DBSubnetGroupDescription" : "description",
            "SubnetIds" : [ "subnet-7b5b4112", "subnet-7b5b4115" ],
            "Tags" : [ {"Key" : "String", "Value" : "String"} ]
         }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  myDBSubnetGroup:
    Type: "AWS::RDS::DBSubnetGroup"
    Properties:
      DBSubnetGroupDescription: "description"
      SubnetIds:
```

```
        - "subnet-7b5b4112"
        - "subnet-7b5b4115"
      Tags:
        -
          Key: "String"
          Value: "String"
```

See Also

- CreateDBSubnetGroup in the *Amazon Relational Database Service API Reference*
- ModifyDBSubnetGroup in the *Amazon Relational Database Service API Reference*
- AWS CloudFormation Stacks Updates (p. 89)

AWS::RDS::EventSubscription

Use the `AWS::RDS::EventSubscription` resource to get notifications for Amazon Relational Database Service events through the Amazon Simple Notification Service. For more information, see Using Amazon RDS Event Notification in the *Amazon Relational Database Service User Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::RDS::EventSubscription",
  "Properties" : {
    "Enabled" : Boolean,
    "EventCategories" : [ String, ... ],
    "SnsTopicArn" : String,
    "SourceIds" : [ String, ... ],
    "SourceType" : String
  }
}
```

YAML

```
Type: "AWS::RDS::EventSubscription"
Properties:
  Enabled: Boolean
  EventCategories:
    - String
  SnsTopicArn: String
  SourceIds:
    - String
  SourceType: String
```

Properties

`Enabled`

Indicates whether to activate the subscription. If you don't specify this property, AWS CloudFormation activates the subscription.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`EventCategories`

A list of event categories that you want to subscribe to for a given source type. If you don't specify this property, you are notified about all event categories. For more information, see Using Amazon RDS Event Notification in the *Amazon Relational Database Service User Guide*.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`SnsTopicArn`

The Amazon Resource Name (ARN) of an Amazon SNS topic that you want to send event notifications to.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`SourceIds`

A list of identifiers for which Amazon RDS provides notification events.

If you don't specify a value, notifications are provided for all sources. If you specify multiple values, they must be of the same type. For example, if you specify a database instance ID, all other values must be database instance IDs.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`SourceType`

The type of source for which Amazon RDS provides notification events. For example, if you want to be notified of events generated by a database instance, set this parameter to `db-instance`. If you don't specify a value, notifications are provided for all source types. For valid values, see the `SourceType` parameter for the CreateEventSubscription action in the *Amazon Relational Database Service API Reference*.

Required: Conditional. If you specify the `SourceIds` or `EventCategories` property, you must specify this property.

Type: String

Update requires: Replacement (p. 90) if you're removing this property after it was previously specified. All other updates require no interruption (p. 90).

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "myEventSubscription" }
```

For the resource with the logical ID `myEventSubscription`, `Ref` returns the Amazon RDS event subscription name, such as: `mystack-myEventSubscription-1DDYF1E3B3I`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following snippet creates an event subscription for an existing database instance `db-instance-1` and a database with the logical ID `myDBInstance`, which is declared elsewhere in the same template.

JSON

```
"myEventSubscription": {
  "Type": "AWS::RDS::EventSubscription",
  "Properties": {
    "EventCategories": ["configuration change", "failure", "deletion"],
    "SnsTopicArn": "arn:aws:sns:us-west-2:123456789012:example-topic",
    "SourceIds": ["db-instance-1", { "Ref" : "myDBInstance" }],
    "SourceType":"db-instance",
    "Enabled" : false
  }
}
```

YAML

```
myEventSubscription:
  Type: "AWS::RDS::EventSubscription"
  Properties:
    EventCategories:
      - "configuration change"
      - "failure"
      - "deletion"
    SnsTopicArn: "arn:aws:sns:us-west-2:123456789012:example-topic"
    SourceIds:
      - "db-instance-1"
      -
        Ref: "myDBInstance"
    SourceType: "db-instance"
    Enabled: false
```

AWS::RDS::OptionGroup

Use the `AWS::RDS::OptionGroup` resource to create an option group that can make managing data and databases easier. For more information about option groups, see Working with Option Groups in the *Amazon Relational Database Service User Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::RDS::OptionGroup",
   "Properties" : {
      "EngineName" : String,
      "MajorEngineVersion" : String,
      "OptionGroupDescription" : String,
      "OptionConfigurations" : [ OptionConfigurations, ... ],
      "Tags" : [ Resource Tag, ... ]
   }
}
```

YAML

```
Type: "AWS::RDS::OptionGroup"
Properties:
  EngineName: String
  MajorEngineVersion: String
  OptionGroupDescription: String
  OptionConfigurations:
    - OptionConfigurations
  Tags:
    Resource Tag
```

Properties

`EngineName`

The name of the database engine that this option group is associated with.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`MajorEngineVersion`

The major version number of the database engine that this option group is associated with.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`OptionGroupDescription`

A description of the option group.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`OptionConfigurations`

The configurations for this option group.

Required: Yes

Type: Amazon RDS OptionGroup OptionConfigurations (p. 1237)

Update requires: Replacement (p. 90)

`Tags`

An arbitrary set of tags (key–value pairs) for this option group.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "myOptionGroup" }
```

For the `myOptionGroup` resource, `Ref` returns the name of the option group.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

Multiple Option Configurations

The following snippet creates an option group with two option configurations (`OEM` and `APEX`):

JSON

```
"OracleOptionGroup": {
  "Type": "AWS::RDS::OptionGroup",
  "Properties": {
    "EngineName": "oracle-ee",
    "MajorEngineVersion": "12.1",
    "OptionGroupDescription": "A test option group",
    "OptionConfigurations":[
```

```
      {
        "OptionName": "OEM",
        "DBSecurityGroupMemberships": ["default"],
        "Port": "5500"
      },
      {
        "OptionName": "APEX"
      }
    ]
  }
}
```

YAML

```
OracleOptionGroup:
  Type: "AWS::RDS::OptionGroup"
  Properties:
    EngineName: "oracle-ee"
    MajorEngineVersion: "12.1"
    OptionGroupDescription: "A test option group"
    OptionConfigurations:
      -
        OptionName: "OEM"
        DBSecurityGroupMemberships:
          - "default"
        Port: "5500"
      -
        OptionName: "APEX"
```

Multiple Settings

The following snippet creates an option group that specifies two option settings for the `MEMCACHED` option:

JSON

```
"SQLOptionGroup": {
  "Type": "AWS::RDS::OptionGroup",
  "Properties": {
    "EngineName": "mysql",
    "MajorEngineVersion": "5.6",
    "OptionGroupDescription": "A test option group",
    "OptionConfigurations":[
      {
        "OptionName": "MEMCACHED",
        "VpcSecurityGroupMemberships": ["sg-a1238db7"],
        "Port": "1234",
        "OptionSettings": [
          {"Name": "CHUNK_SIZE", "Value": "32"},
          {"Name": "BINDING_PROTOCOL", "Value": "ascii"}
        ]
      }
    ]
  }
}
```

YAML

```
SQLOptionGroup:
  Type: "AWS::RDS::OptionGroup"
  Properties:
    EngineName: "mysql"
```

```
    MajorEngineVersion: "5.6"
    OptionGroupDescription: "A test option group"
    OptionConfigurations:
      -
        OptionName: "MEMCACHED"
        VpcSecurityGroupMemberships:
          - "sg-a1238db7"
        Port: "1234"
        OptionSettings:
          -
            Name: "CHUNK_SIZE"
            Value: "32"
          -
            Name: "BINDING_PROTOCOL"
            Value: "ascii"
```

AWS::Redshift::Cluster

The `AWS::Redshift::Cluster` resource creates an Amazon Redshift cluster. A cluster is a fully managed data warehouse that consists of a set of compute nodes. For more information about default values and valid values, see CreateCluster in the *Amazon Redshift API Reference*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Redshift::Cluster",
  "Properties" : {
    "AllowVersionUpgrade" : Boolean,
    "AutomatedSnapshotRetentionPeriod" : Integer,
    "AvailabilityZone" : String,
    "ClusterParameterGroupName" : String,
    "ClusterSecurityGroups" : [ String, ... ],
    "ClusterSubnetGroupName" : String,
    "ClusterType" : String,
    "ClusterVersion" : String,
    "DBName" : String,
    "ElasticIp" : String,
    "Encrypted" : Boolean,
    "HsmClientCertificateIdentifier" : String,
    "HsmConfigurationIdentifier" : String,
    "KmsKeyId" : String,
    "MasterUsername" : String,
    "MasterUserPassword" : String,
    "NodeType" : String,
    "NumberOfNodes" : Integer,
    "OwnerAccount" : String,
```

```
    "Port" : Integer,
    "PreferredMaintenanceWindow" : String,
    "PubliclyAccessible" : Boolean,
    "SnapshotClusterIdentifier" : String,
    "SnapshotIdentifier" : String,
    "VpcSecurityGroupIds" : [ String, ... ]
  }
}
```

YAML

```
Type: "AWS::Redshift::Cluster"
Properties:
  AllowVersionUpgrade: Boolean
  AutomatedSnapshotRetentionPeriod: Integer
  AvailabilityZone: String
  ClusterParameterGroupName: String
  ClusterSecurityGroups: - String
  ClusterSubnetGroupName: String
  ClusterType: String
  ClusterVersion: String
  DBName: String
  ElasticIp: String
  Encrypted: Boolean
  HsmClientCertificateIdentifier: String
  HsmConfigurationIdentifier: String
  KmsKeyId: String
  MasterUsername: String
  MasterUserPassword: String
  NodeType: String
  NumberOfNodes: Integer
  OwnerAccount: String
  Port: Integer
  PreferredMaintenanceWindow: String
  PubliclyAccessible: Boolean
  SnapshotClusterIdentifier: String
  SnapshotIdentifier: String
  VpcSecurityGroupIds: - String
```

Properties

`AllowVersionUpgrade`

When a new version of the Amazon Redshift is released, tells whether upgrades can be applied to the engine that is running on the cluster. The upgrades are applied during the maintenance window.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`AutomatedSnapshotRetentionPeriod`

The number of days that automated snapshots are retained. If you set the value to `0`, automated snapshots are disabled.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`AvailabilityZone`

The Amazon Elastic Compute Cloud (Amazon EC2) Availability Zone in which you want to provision your Amazon Redshift cluster. For example, if you have several EC2 instances running in a specific Availability Zone, you might want the cluster to be provisioned in the same zone in order to decrease network latency.

Required: No

Type: String

Update requires: Replacement (p. 90)

`ClusterParameterGroupName`

The name of the parameter group that you want to associate with this cluster.

Required: No

Type: String

Update requires: Some interruptions (p. 90)

`ClusterSecurityGroups`

A list of security groups that you want to associate with this cluster.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

`ClusterSubnetGroupName`

The name of a cluster subnet group that you want to associate with this cluster.

Required: No

Type: String

Update requires: Replacement (p. 90)

`ClusterType`

The type of cluster. You can specify `single-node` or `multi-node`.

Required: Yes

Type: String

Update requires: Some interruptions (p. 90)

`ClusterVersion`

The Amazon Redshift engine version that you want to deploy on the cluster.

Required: No

Type: String

Update requires: No interruption (p. 90)

`DBName`

The name of the first database that is created when the cluster is created.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`ElasticIp`

The Elastic IP (EIP) address for the cluster.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Encrypted`

Indicates whether the data in the cluster is encrypted at rest.

Required: No

Type: Boolean

Update requires: Replacement (p. 90)

`HsmClientCertificateIdentifier`

Specifies the name of the HSM client certificate that the Amazon Redshift cluster uses to retrieve the data encryption keys stored in an HSM.

Required: No

Type: String

Update requires: No interruption (p. 90)

`HsmConfigurationIdentifier`

Specifies the name of the hardware security module (HSM) configuration that contains the information that the Amazon Redshift cluster can use to retrieve and store keys in an HSM.

Required: No

Type: String

Update requires: No interruption (p. 90)

`KmsKeyId`

The AWS Key Management Service (AWS KMS) key ID that you want to use to encrypt data in the cluster.

Required: No

Type: String

Update requires: Replacement (p. 90)

`MasterUsername`

The user name that is associated with the master user account for this cluster.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`MasterUserPassword`

The password associated with the master user account for this cluster.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`NodeType`

The node type that is provisioned for this cluster.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`NumberOfNodes`

The number of compute nodes in the cluster. If you specify `multi-node` for the `ClusterType` parameter, you must specify a number greater than 1.

Important
You can't specify this parameter for a single-node cluster.

Required: Conditional

Type: Integer

Update requires: Some interruptions (p. 90)

`OwnerAccount`

When you restore from a snapshot from another AWS account, the 12-digit AWS account ID that contains that snapshot.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Port`

The port number on which the cluster accepts incoming connections.

Required: No

Type: Integer

Update requires: Replacement (p. 90)

`PreferredMaintenanceWindow`

The weekly time range (in UTC) during which automated cluster maintenance can occur. The format of the time range is `ddd:hh24:mi-ddd:hh24:mi`.

Required: No

Type: String

Update requires: No interruption (p. 90)

`PubliclyAccessible`

Indicates whether the cluster can be accessed from a public network.

Required: No

Type: Boolean

Update requires: Replacement (p. 90)

`SnapshotClusterIdentifier`

The name of the cluster the source snapshot was created from. For more information about restoring from a snapshot, see the RestoreFromClusterSnapshot action in the *Amazon Redshift API Reference*.

Required: No

Required: Conditional. This property is required if your IAM policy includes a restriction on the cluster name, where the resource element specifies anything other than the wildcard character (`*`) for the cluster name.

Update requires: Replacement (p. 90)

`SnapshotIdentifier`

The name of the snapshot from which to create a new cluster.

Required: Conditional. If you specified the `SnapshotClusterIdentifier` property, you must specify this property.

Type: String

Update requires: Replacement (p. 90)

`VpcSecurityGroupIds`

A list of VPC security groups that are associated with this cluster.

Required: No

Type: List of strings

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "myCluster" }
```

For the Amazon Redshift cluster `myCluster`, `Ref` returns the name of the cluster.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`Endpoint.Address`

The connection endpoint for the Amazon Redshift cluster. For example: `examplecluster.cg034hpkmmjt.us-east-1.redshift.amazonaws.com`.

`Endpoint.Port`

The port number on which the Amazon Redshift cluster accepts connections. For example: `5439`.

Examples

The following examples describes a single-node Amazon Redshift cluster. The master user password is referenced from an input parameter that is in the same template.

JSON

```
"myCluster" : {
  "Type": "AWS::Redshift::Cluster",
  "Properties": {
    "DBName" : "mydb",
    "MasterUsername" : "master",
    "MasterUserPassword" : { "Ref" : "MasterUserPassword" },
    "NodeType" : "dw.hs1.xlarge",
    "ClusterType" : "single-node"
  }
}
```

YAML

```
myCluster:
  Type: "AWS::Redshift::Cluster"
  Properties:
    DBName: "mydb"
    MasterUsername: "master"
    MasterUserPassword:
      Ref: "MasterUserPassword"
    NodeType: "dw.hs1.xlarge"
    ClusterType: "single-node"
```

More Info

For a complete example template, see Amazon Redshift Template Snippets (p. 340).

AWS::Redshift::ClusterParameterGroup

Creates an Amazon Redshift parameter group that you can associate with an Amazon Redshift cluster. The parameters in the group apply to all the databases that you create in the cluster.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Redshift::ClusterParameterGroup",
  "Properties" : {
    "Description" : String,
    "ParameterGroupFamily" : String,
    "Parameters" : [ Parameter, ... ]
  }
}
```

YAML

```
Type: "AWS::Redshift::ClusterParameterGroup"
Properties:
  Description: String
  ParameterGroupFamily: String
  Parameters:
    - Parameter
```

Properties

`Description`

A description of the parameter group.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`ParameterGroupFamily`

The Amazon Redshift engine version that applies to this cluster parameter group. The cluster engine version determines the set of parameters that you can specify in the `Parameters` property.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Parameters`

A list of parameter names and values that are allowed by the Amazon Redshift engine version that you specified in the `ParameterGroupFamily` property. For more information, see Amazon Redshift Parameter Groups in the *Amazon Redshift Cluster Management Guide*.

Required: No

Type: Amazon Redshift Parameter Type (p. 1236)

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "myClusterParameterGroup" }
```

For the Amazon Redshift cluster parameter group `myClusterParameterGroup`, `Ref` returns the name of the cluster parameter group.

For more information about using the `Ref` function, see Ref (p. 1343).

Template Snippets

Single Parameter

The following snippet describes a parameter group with one parameter that is specified:

JSON

```
"myClusterParameterGroup" : {
  "Type" : "AWS::Redshift::ClusterParameterGroup",
  "Properties" : {
    "Description" : "My parameter group",
    "ParameterGroupFamily" : "redshift-1.0",
    "Parameters" : [ {
      "ParameterName" : "enable_user_activity_logging",
      "ParameterValue" : "true"
    }]
  }
}
```

YAML

```
myClusterParameterGroup:
  Type: "AWS::Redshift::ClusterParameterGroup"
  Properties:
    Description: "My parameter group"
    ParameterGroupFamily: "redshift-1.0"
    Parameters:
      -
        ParameterName: "enable_user_activity_logging"
        ParameterValue: "true"
```

Workload Management Configuration

The following snippet modifies the workload management configuration using the `wlm_json_configuration` parameter. The parameter value is a JSON object that must be passed as a string enclosed in quotation marks (").

JSON

```
"RedshiftClusterParameterGroup" : {
```

```
  "Type" : "AWS::Redshift::ClusterParameterGroup",
  "Properties" : {
    "Description" : "Cluster parameter group",
    "ParameterGroupFamily" : "redshift-1.0",
    "Parameters" : [{
      "ParameterName" : "wlm_json_configuration",
      "ParameterValue" : "[{\"user_group\":[\"example_user_group1\"],\"query_group\":
[\"example_query_group1\"],\"query_concurrency\":7},{\"query_concurrency\":5}]"
    }]
  }
}
```

YAML

```
RedshiftClusterParameterGroup:
  Type: "AWS::Redshift::ClusterParameterGroup"
  Properties:
    Description: "Cluster parameter group"
    ParameterGroupFamily: "redshift-1.0"
    Parameters:
      -
        ParameterName: "wlm_json_configuration"
        ParameterValue: "[{\"user_group\":[\"example_user_group1\"],\"query_group\":
[\"example_query_group1\"],\"query_concurrency\":7},{\"query_concurrency\":5}]"
```

AWS::Redshift::ClusterSecurityGroup

Creates an Amazon Redshift security group. You use security groups to control access to Amazon Redshift clusters that are not in a VPC.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Redshift::ClusterSecurityGroup",
  "Properties" : {
    "Description" : String
  }
}
```

YAML

```
Type: "AWS::Redshift::ClusterSecurityGroup"
Properties:
```

```
  Description: String
```

Properties

`Description`

A description of the security group.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "myClusterSecurityGroup" }
```

For the Amazon Redshift cluster security group `myClusterSecurityGroup`, `Ref` returns the name of the cluster security group.

For more information about using the `Ref` function, see Ref (p. 1343).

Template Snippet

The following snippet creates an Amazon Redshift cluster security group that you can associate cluster security group ingress rules with:

JSON

```
"myClusterSecurityGroup" : {
  "Type": "AWS::Redshift::ClusterSecurityGroup",
  "Properties": {
    "Description" : "Security group to determine where connections to the Amazon Redshift
 cluster can come from"
  }
}
```

YAML

```
myClusterSecurityGroup:
  Type: "AWS::Redshift::ClusterSecurityGroup"
  Properties:
    Description: "Security group to determine where connections to the Amazon Redshift
 cluster can come from"
```

See Also

- AWS::Redshift::ClusterSecurityGroupIngress (p. 922)

AWS::Redshift::ClusterSecurityGroupIngress

Specifies inbound (ingress) rules for an Amazon Redshift security group.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Redshift::ClusterSecurityGroupIngress",
  "Properties" : {
    "ClusterSecurityGroupName" : String,
    "CIDRIP" : String,
    "EC2SecurityGroupName" : String,
    "EC2SecurityGroupOwnerId" : String
  }
}
```

YAML

```
Type: "AWS::Redshift::ClusterSecurityGroupIngress"
Properties:
  ClusterSecurityGroupName: String
  CIDRIP: String
  EC2SecurityGroupName: String
  EC2SecurityGroupOwnerId: String
```

Properties

`ClusterSecurityGroupName`

The name of the Amazon Redshift security group that will be associated with the ingress rule.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`CIDRIP`

The IP address range that has inbound access to the Amazon Redshift security group.

Required: No

Type: String

Update requires: Replacement (p. 90)

`EC2SecurityGroupName`

The Amazon EC2 security group that will be added the Amazon Redshift security group.

Required: No

Type: String

Update requires: Replacement (p. 90)

`EC2SecurityGroupOwnerId`

The 12-digit AWS account number of the owner of the Amazon EC2 security group that is specified by the `EC2SecurityGroupName` parameter.

Required: Conditional. If you specify the `EC2SecurityGroupName` property, you must specify this property.

Type: String

Update requires: Replacement (p. 90)

Template Snippet

The following snippet describes a ingress rules for an Amazon Redshift cluster security group:

JSON

```
"myClusterSecurityGroupIngressIP" : {
  "Type": "AWS::Redshift::ClusterSecurityGroupIngress",
  "Properties": {
    "ClusterSecurityGroupName" : {"Ref":"myClusterSecurityGroup"},
    "CIDRIP" : "10.0.0.0/16"
  }
}
```

YAML

```
myClusterSecurityGroupIngressIP:
  Type: "AWS::Redshift::ClusterSecurityGroupIngress"
  Properties:
    ClusterSecurityGroupName:
      Ref: "myClusterSecurityGroup"
    CIDRIP: "10.0.0.0/16"
```

See Also

- AWS::Redshift::ClusterSecurityGroup (p. 920)

AWS::Redshift::ClusterSubnetGroup

Creates an Amazon Redshift subnet group. You must provide a list of one or more subnets in your existing Amazon VPC when creating an Amazon Redshift subnet group.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Redshift::ClusterSubnetGroup",
  "Properties" : {
    "Description" : String,
    "SubnetIds" : [ String, ... ]
  }
}
```

YAML

```
Type: "AWS::Redshift::ClusterSubnetGroup"
Properties:
  Description: String
  SubnetIds:
    - String
```

Properties

`Description`

A description of the subnet group.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`SubnetIds`

A list of VPC subnet IDs. You can modify a maximum of 20 subnets.

Required: Yes

Type: List of strings

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "myClusterSubnetGroup" }
```

For the Amazon Redshift cluster subnet group `myClusterSubnetGroup`, `Ref` returns the name of the cluster subnet group.

For more information about using the `Ref` function, see Ref (p. 1343).

Template Snippet

The following snippet specifies one subnet for an Amazon Redshift cluster subnet group.

```
"myClusterSubnetGroup" : {
  "Type": "AWS::Redshift::ClusterSubnetGroup",
  "Properties": {
    "Description" : "My ClusterSubnetGroup",
    "SubnetIds" : ["subnet-7fbc2813"]
  }
}
```

AWS::Route53::HealthCheck

Use the `AWS::Route53::HealthCheck` resource to check the health of your resources before Amazon Route 53 responds to a DNS query. For more information, see How Health Checks Work in Simple Amazon Route 53 Configurations in the *Amazon Route 53 Developer Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Route53::HealthCheck",
  "Properties" : {
    "HealthCheckConfig" : HealthCheckConfig,
    "HealthCheckTags" : [ HealthCheckTags, ... ]
  }
}
```

YAML

```
Type: "AWS::Route53::HealthCheck"
Properties:
  HealthCheckConfig:
    HealthCheckConfig
  HealthCheckTags:
    - HealthCheckTags
```

Properties

`HealthCheckConfig`

An Amazon Route 53 health check.

Required: Yes

Type: Amazon Route 53 HealthCheckConfig (p. 1242)

Update requires: No interruption (p. 90)

`HealthCheckTags`

An arbitrary set of tags (key–value pairs) for this health check.

Required: No

Type: A list of Amazon Route 53 HealthCheckTags (p. 1246)

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the health check ID, such as `e0a123b4-4dba-4650-935e-example`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates an Amazon Route 53 health check that sends request to the specified endpoint.

JSON

```
"myHealthCheck": {
  "Type": "AWS::Route53::HealthCheck",
  "Properties": {
    "HealthCheckConfig": {
      "IPAddress": "000.000.000.000",
      "Port": "80",
      "Type": "HTTP",
      "ResourcePath": "/example/index.html",
      "FullyQualifiedDomainName": "example.com",
      "RequestInterval": "30",
      "FailureThreshold": "3"
    },
    "HealthCheckTags" : [{
      "Key": "SampleKey1",
      "Value": "SampleValue1"
    },
    {
      "Key": "SampleKey2",
      "Value": "SampleValue2"
    }]
  }
}
```

YAML

```
myHealthCheck:
  Type: "AWS::Route53::HealthCheck"
  Properties:
    HealthCheckConfig:
      IPAddress: "000.000.000.000"
      Port: "80"
      Type: "HTTP"
      ResourcePath: "/example/index.html"
      FullyQualifiedDomainName: "example.com"
      RequestInterval: "30"
      FailureThreshold: "3"
    HealthCheckTags:
      -
        Key: "SampleKey1"
        Value: "SampleValue1"
      -
        Key: "SampleKey2"
        Value: "SampleValue2"
```

AWS::Route53::HostedZone

The `AWS::Route53::HostedZone` resource creates a hosted zone, which can contain a collection of record sets for a domain. You cannot create a hosted zone for a top-level domain (TLD). For more information, see POST CreateHostedZone or POST CreateHostedZone (Private) in the *Amazon Route 53 API Reference*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Route53::HostedZone",
  "Properties" : {
    "HostedZoneConfig" : HostedZoneConfig,
    "HostedZoneTags" : [  HostedZoneTags, ... ],
    "Name" : String,
    "VPCs" : [ HostedZoneVPCs, ... ]
  }
}
```

YAML

```
Type: "AWS::Route53::HostedZone"
Properties:
  HostedZoneConfig:
    HostedZoneConfig
```

```
  HostedZoneTags:
    -  HostedZoneTags
  Name: String
  VPCs:
    - HostedZoneVPCs
```

Properties

`HostedZoneConfig`

A complex type that contains an optional comment about your hosted zone.

Required: No

Type: Amazon Route 53 HostedZoneConfig Property (p. 1247)

Update requires: No interruption (p. 90)

`HostedZoneTags`

An arbitrary set of tags (key–value pairs) for this hosted zone.

Required: No

Type: List of Amazon Route 53 HostedZoneTags (p. 1248)

Update requires: No interruption (p. 90)

`Name`

The name of the domain. For resource record types that include a domain name, specify a fully qualified domain name.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`VPCs`

One or more VPCs that you want to associate with this hosted zone. When you specify this property, AWS CloudFormation creates a private hosted zone.

Required: No

Type: List of Amazon Route 53 HostedZoneVPCs (p. 1248)

If this property was specified previously and you're modifying values, updates require no interruption (p. 90). If this property wasn't specified and you add values, updates require replacement (p. 90). Also, if this property was specified and you remove all values, updates require replacement (p. 90).

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "myHostedZone" }
```

`Ref` returns the hosted zone ID, such as `Z23ABC4XYZL05B`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`NameServers`

Returns the set of name servers for the specific hosted zone. For example: `ns1.example.com`.

Example

The following template snippet creates a private hosted zone for the `example.com` domain.

JSON

```
"DNS": {
  "Type": "AWS::Route53::HostedZone",
  "Properties": {
    "HostedZoneConfig": {
      "Comment": "My hosted zone for example.com"
    },
    "Name": "example.com",
    "VPCs": [{
      "VPCId": "vpc-abcd1234",
      "VPCRegion": "ap-northeast-1"
    },
    {
      "VPCId": "vpc-efgh5678",
      "VPCRegion": "us-west-2"
    }],
    "HostedZoneTags" : [{
      "Key": "SampleKey1",
      "Value": "SampleValue1"
    },
    {
      "Key": "SampleKey2",
      "Value": "SampleValue2"
    }]
  }
}
```

YAML

```
DNS:
  Type: "AWS::Route53::HostedZone"
  Properties:
    HostedZoneConfig:
      Comment: "My hosted zone for example.com"
    Name: "example.com"
    VPCs:
      -
        VPCId: "vpc-abcd1234"
        VPCRegion: "ap-northeast-1"
      -
        VPCId: "vpc-efgh5678"
        VPCRegion: "us-west-2"
```

```
    HostedZoneTags:
      -
        Key: "SampleKey1"
        Value: "SampleValue1"
      -
        Key: "SampleKey2"
        Value: "SampleValue2"
```

AWS::Route53::RecordSet

The `AWS::Route53::RecordSet` type can be used as a standalone resource or as an embedded property in the AWS::Route53::RecordSetGroup (p. 935) type. Note that some `AWS::Route53::RecordSet` properties are valid only when used within `AWS::Route53::RecordSetGroup`.

For more information about constraints and values for each property, see POST CreateHostedZone for hosted zones and POST ChangeResourceRecordSet for resource record sets.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::Route53::RecordSet",
  "Properties" : {
    "AliasTarget (p. 931)" : AliasTarget (p. 1240),
    "Comment" : String,
    "Failover" : String,
    "GeoLocation" : GeoLocation,
    "HealthCheckId" : String,
    "HostedZoneId (p. 932)" : String,
    "HostedZoneName (p. 932)" : String,
    "Name (p. 932)" : String,
    "Region (p. 932)" : String,
    "ResourceRecords (p. 933)" : [ String ],
    "SetIdentifier (p. 933)" : String,
    "TTL (p. 933)" : String,
    "Type (p. 933)" : String,
    "Weight (p. 934)" : Integer
  }
}
```

YAML

```
Type: "AWS::Route53::RecordSet"
Properties:
  AliasTarget (p. 931): AliasTarget (p. 1240)
  Comment: String
  Failover: String
  GeoLocation:
```

```
    GeoLocation
  HealthCheckId: String
  HostedZoneId (p. 932): String
  HostedZoneName (p. 932): String
  Name (p. 932): String
  Region (p. 932): String
  ResourceRecords (p. 933):
    - String
  SetIdentifier (p. 933): String
  TTL (p. 933): String
  Type (p. 933): String
  Weight (p. 934): Integer
```

Properties

`AliasTarget`

Alias resource record sets only: Information about the domain to which you are redirecting traffic.

If you specify this property, do not specify the `TTL` property. The alias uses a TTL value from the alias target record.

For more information about alias resource record sets, see Creating Alias Resource Record Sets in the *Amazon Route 53 Developer Guide* and POST ChangeResourceRecordSets in the Amazon Route 53 API reference.

Required: Conditional. Required if you are creating an alias resource record set.

Type: AliasTarget (p. 1240)

Update requires: No interruption (p. 90)

`Comment`

Any comments that you want to include about the hosted zone.

> **Important**
> If the record set is part of a record set group, this property isn't valid. Don't specify this property.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Failover`

Designates the record set as a `PRIMARY` or `SECONDARY` failover record set. When you have more than one resource performing the same function, you can configure Amazon Route 53 to check the health of your resources and use only health resources to respond to DNS queries. You cannot create nonfailover resource record sets that have the same `Name` and `Type` property values as failover resource record sets. For more information, see the Failover content in the *Amazon Route 53 API Reference*.

If you specify this property, you must specify the `SetIdentifier` property.

Required: No

Type: String

Update requires: No interruption (p. 90)

`GeoLocation`

Describes how Amazon Route 53 responds to DNS queries based on the geographic origin of the query.

Required: No

Type: Amazon Route 53 Record Set GeoLocation Property (p. 1241)

Update requires: No interruption (p. 90)

`HealthCheckId`

The health check ID that you want to apply to this record set. Amazon Route 53 returns this resource record set in response to a DNS query only while record set is healthy.

Required: No

Type: String

Update requires: No interruption (p. 90)

`HostedZoneId`

The ID of the hosted zone.

Required: Conditional. You must specify either the `HostedZoneName` or `HostedZoneId`, but you cannot specify both. If this record set is part of a record set group, do not specify this property.

Type: String

Update requires: Replacement (p. 90)

`HostedZoneName`

The name of the domain for the hosted zone where you want to add the record set.

When you create a stack using an `AWS::Route53::RecordSet` that specifies `HostedZoneName`, AWS CloudFormation attempts to find a hosted zone whose name matches the `HostedZoneName`. If AWS CloudFormation cannot find a hosted zone with a matching domain name, or if there is more than one hosted zone with the specified domain name, AWS CloudFormation will not create the stack.

If you have multiple hosted zones with the same domain name, you must explicitly specify the hosted zone using `HostedZoneId`.

Required: Conditional. You must specify either the `HostedZoneName` or `HostedZoneId`, but you cannot specify both. If this record set is part of a record set group, do not specify this property.

Type: String

Update requires: Replacement (p. 90)

`Name`

The name of the domain. You must specify a fully qualified domain name that ends with a period as the last label indication. If you omit the final period, Amazon Route 53 adds it.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Region`

Latency resource record sets only: The Amazon EC2 region where the resource that is specified in this resource record set resides. The resource typically is an AWS resource, for example, Amazon

EC2 instance or an Elastic Load Balancing load balancer, and is referred to by an IP address or a DNS domain name, depending on the record type.

When Amazon Route 53 receives a DNS query for a domain name and type for which you have created latency resource record sets, Amazon Route 53 selects the latency resource record set that has the lowest latency between the end user and the associated Amazon EC2 region. Amazon Route 53 then returns the value that is associated with the selected resource record set.

The following restrictions must be followed:

- You can only specify one resource record per latency resource record set.
- You can only create one latency resource record set for each Amazon EC2 region.
- You are not required to create latency resource record sets for all Amazon EC2 regions. Amazon Route 53 will choose the region with the best latency from among the regions for which you create latency resource record sets.
- You cannot create both weighted and latency resource record sets that have the same values for the Name and Type elements.

To see a list of regions by service, see Regions and Endpoints in the *AWS General Reference*.

`ResourceRecords`

List of resource records to add. Each record should be in the format appropriate for the record type specified by the `Type` property. For information about different record types and their record formats, see Appendix: Domain Name Format in the *Amazon Route 53 Developer Guide*.

Required: Conditional. If you don't specify the `AliasTarget` property, you must specify this property. If you are creating an alias resource record set, do not specify this property.

Type: List of strings

Update requires: No interruption (p. 90)

`SetIdentifier`

A unique identifier that differentiates among multiple resource record sets that have the same combination of DNS name and type.

Required: Conditional. Required if you are creating a weighted, latency, failover, or geolocation resource record set.

For more information, see the SetIdentifier content in the *Amazon Route 53 Developer Guide*.

Type: String

Update requires: No interruption (p. 90)

`TTL`

The resource record cache time to live (TTL), in seconds. If you specify this property, do not specify the `AliasTarget` property. For alias target records, the alias uses a TTL value from the target.

If you specify this property, you must specify the `ResourceRecords` property.

Required: Conditional. If you don't specify the `AliasTarget` property, you must specify this property. If you are creating an alias resource record set, do not specify this property.

Type: String

Update requires: No interruption (p. 90)

`Type`

The type of records to add. For valid values, see the Type content in the *Amazon Route 53 API Reference*. In AWS CloudFormation, you can't create records of type `NS` or `SOA`.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`Weight`

Weighted resource record sets only: Among resource record sets that have the same combination of DNS name and type, a value that determines what portion of traffic for the current resource record set is routed to the associated location.

For more information about weighted resource record sets, see Setting Up Weighted Resource Record Sets in the *Amazon Route 53 Developer Guide*.

Required: Conditional. Required if you are creating a weighted resource record set.

Type: Number. Weight expects integer values.

Update requires: No interruption (p. 90)

Return Value

When you specify an `AWS::Route53::RecordSet` type as an argument to the `Ref` function, AWS CloudFormation returns the value of the domain name of the record set.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

Mapping an Amazon Route 53 A record to the public IP of an Amazon EC2 instance

JSON

```
"Resources" : {
   "Ec2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
         "ImageId" : { "Fn::FindInMap" : [
            "RegionMap", { "Ref" : "AWS::Region" }, "AMI"
         ] }
      }
   },
   "myDNSRecord" : {
      "Type" : "AWS::Route53::RecordSet",
      "Properties" : {
         "HostedZoneName" : { "Ref" : "HostedZoneResource" },
         "Comment" : "DNS name for my instance.",
         "Name" : {
            "Fn::Join" : [ "", [
               {"Ref" : "Ec2Instance"}, ".",
               {"Ref" : "AWS::Region"}, ".",
               {"Ref" : "HostedZone"} ,"."
            ] ]
         },
         "Type" : "A",
         "TTL" : "900",
         "ResourceRecords" : [
```

```
            { "Fn::GetAtt" : [ "Ec2Instance", "PublicIp" ] }
         ]
      }
   }
}
```

YAML

```
Resources:
  Ec2Instance:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !FindInMap [RegionMap, !Ref 'AWS::Region', AMI]
  myDNSRecord:
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneName: !Ref 'HostedZoneResource'
      Comment: DNS name for my instance.
      Name: !Join ['', [!Ref 'Ec2Instance', ., !Ref 'AWS::Region', ., !Ref
 'HostedZone', .]]
      Type: A
      TTL: '900'
      ResourceRecords:
      - !GetAtt Ec2Instance.PublicIp
```

Additional Information

For additional `AWS::Route53::RecordSet` snippets, see Amazon Route 53 Template Snippets (p. 351) .

AWS::Route53::RecordSetGroup

The AWS::Route53::RecordSetGroup resource creates record sets for a hosted zone. For more information about constraints and values for each property, see POST CreateHostedZone for hosted zones and POST ChangeResourceRecordSet for resource record sets.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::Route53::RecordSetGroup",
   "Properties" : {
      "Comment (p. 936)" : String,
      "HostedZoneId (p. 936)" : String,
      "HostedZoneName (p. 936)" : String,
      "RecordSets (p. 936)" : [ RecordSet1, ... ]
   }
}
```

YAML

```
Type: "AWS::Route53::RecordSetGroup"
Properties:
  Comment (p. 936): String
  HostedZoneId (p. 936): String
  HostedZoneName (p. 936): String
  RecordSets (p. 936):
    - RecordSet1
```

Properties

`Comment`

Any comments you want to include about the hosted zone.

Required: No

Type: String

Update requires: No interruption (p. 90)

`HostedZoneId`

The ID of the hosted zone.

Required: Conditional: You must specify either the `HostedZoneName` or `HostedZoneId`, but you cannot specify both.

Type: String

Update requires: Replacement (p. 90)

`HostedZoneName`

The name of the domain for the hosted zone where you want to add the record set.

When you create a stack using an `AWS::Route53::RecordSet` that specifies `HostedZoneName`, AWS CloudFormation attempts to find a hosted zone whose name matches the `HostedZoneName`. If AWS CloudFormation cannot find a hosted zone with a matching domain name, or if there is more than one hosted zone with the specified domain name, AWS CloudFormation will not create the stack.

If you have multiple hosted zones with the same domain name, you must explicitly specify the hosted zone using `HostedZoneId`.

Required: Conditional. You must specify either the `HostedZoneName` or `HostedZoneId`, but you cannot specify both.

Type: String

Update requires: Replacement (p. 90)

`RecordSets`

List of resource record sets to add.

Required: Yes

Type:: List of AWS::Route53::RecordSet (p. 930) objects, as shown in the following example:

```
"RecordSets" : [
  {
    "Name" : "mysite.example.com.",
    "Type" : "CNAME",
    "TTL" : "900",
    "SetIdentifier" : "Frontend One",
    "Weight" : "4",
    "ResourceRecords" : ["example-ec2.amazonaws.com"]
  },
  {
    "Name" : "mysite.example.com.",
    "Type" : "CNAME",
```

```
    "TTL" : "900",
    "SetIdentifier" : "Frontend Two",
    "Weight" : "6",
    "ResourceRecords" : ["example-ec2-larger.amazonaws.com"]
  }
]
```

Update requires: No interruption (p. 90)

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name. For example:

```
{ "Ref": "MyRecordSetGroup" }
```

For the resource with the logical ID "MyRecordSetGroup", `Ref` will return the AWS resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Template Examples

For AWS::Route53::RecordSetGroup snippets, see Amazon Route 53 Template Snippets (p. 351).

AWS::S3::Bucket

The `AWS::S3::Bucket` resource creates an Amazon Simple Storage Service (Amazon S3) bucket in the same AWS Region in which you create the AWS CloudFormation stack.

To control how AWS CloudFormation handles the bucket when the stack is deleted, you can set a deletion policy for your bucket. For Amazon S3 buckets, you can choose to *retain* the bucket or to *delete* the bucket. For more information, see DeletionPolicy Attribute (p. 1297).

Important
You can delete only empty buckets. Deletion will fail for buckets that have contents.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::S3::Bucket",
  "Properties" : {
    "AccessControl" : String,
```

```
    "BucketName" : String,
    "CorsConfiguration" : CORS Configuration,
    "LifecycleConfiguration" : Lifecycle Configuration,
    "LoggingConfiguration" : Logging Configuration,
    "NotificationConfiguration" : Notification Configuration,
    "ReplicationConfiguration" : Replication Configuration,
    "Tags" : [ Resource Tag, ... ],
    "VersioningConfiguration" : Versioning Configuration,
    "WebsiteConfiguration" : Website Configuration Type
  }
}
```

YAML

```
Type: "AWS::S3::Bucket"
Properties:
  AccessControl: String
  BucketName: String
  CorsConfiguration:
    CORS Configuration
  LifecycleConfiguration:
    Lifecycle Configuration
  LoggingConfiguration:
    Logging Configuration
  NotificationConfiguration:
    Notification Configuration
  ReplicationConfiguration:
    Replication Configuration
  Tags:
    - Resource Tag
  VersioningConfiguration:
    Versioning Configuration
  WebsiteConfiguration:
    Website Configuration Type
```

Properties

`AccessControl`

A canned access control list (ACL) that grants predefined permissions to the bucket. For more information about canned ACLs, see Canned ACLs in the Amazon S3 documentation in the *Amazon Simple Storage Service Developer Guide*..

Required: No

Type: String

Valid values: `AuthenticatedRead` | `AwsExecRead` | `BucketOwnerRead` | `BucketOwnerFullControl` | `LogDeliveryWrite` | `Private` | `PublicRead` | `PublicReadWrite`

Update requires: No interruption (p. 90)

`BucketName`

A name for the bucket. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the bucket name. For more information, see Name Type (p. 1217). The bucket name must contain only lowercase letters, numbers, periods (.), and dashes (-).

Important
If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

`CorsConfiguration`

Rules that define cross-origin resource sharing of objects in this bucket. For more information, see Enabling Cross-Origin Resource Sharing in the *Amazon Simple Storage Service Developer Guide*.

Required: No

Type: Amazon S3 Cors Configuration (p. 1249)

Update requires: No interruption (p. 90)

`LifecycleConfiguration`

Rules that define how Amazon S3 manages objects during their lifetime. For more information, see Object Lifecycle Management in the *Amazon Simple Storage Service Developer Guide*.

Required: No

Type: Amazon S3 Lifecycle Configuration (p. 1251)

Update requires: No interruption (p. 90)

`LoggingConfiguration`

Settings that define where logs are stored.

Required: No

Type: Amazon S3 Logging Configuration (p. 1255)

Update requires: No interruption (p. 90)

`NotificationConfiguration`

Configuration that defines how Amazon S3 handles bucket notifications.

Required: No

Type: Amazon S3 NotificationConfiguration (p. 1256)

Update requires: No interruption (p. 90)

`ReplicationConfiguration`

Configuration for replicating objects in an S3 bucket. To enable replication, you must also enable versioning by using the `VersioningConfiguration` property.

Amazon S3 can store replicated objects in only one destination (S3 bucket). The destination bucket must already exist and be in a different AWS Region than your source bucket.

Required: No

Type: Amazon S3 ReplicationConfiguration (p. 1263)

Update requires: No interruption (p. 90)

`Tags`

An arbitrary set of tags (key-value pairs) for this S3 bucket.

Important
We recommend limiting the number of tags to seven. Applying more than seven tags prevents the AWS CLI and the AWS CloudFormation console and API actions from listing the tags for the S3 bucket.

Required: No

Type: AWS CloudFormation Resource Tags (p. 1236)

Update requires: No interruption (p. 90)

`VersioningConfiguration`

Enables multiple variants of all objects in this bucket. You might enable versioning to prevent objects from being deleted or overwritten by mistake or to archive objects so that you can retrieve previous versions of them.

Required: No

Type: Amazon S3 Versioning Configuration (p. 1265)

Update requires: No interruption (p. 90)

`WebsiteConfiguration`

Information used to configure the bucket as a static website. For more information, see Hosting Websites on Amazon S3.

Required: No

Type: Website Configuration Type (p. 1266)

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

Example: `mystack-mybucket-kdwwxmddtr2g`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`DomainName`

Returns the IPv4 DNS name of the specified bucket.

Example: `mystack-mybucket-kdwwxmddtr2g.s3.amazonaws.com`

`DualStackDomainName`

Returns the IPv6 DNS name of the specified bucket.

Example: `mystack-mybucket-kdwwxmddtr2g.s3.dualstack.us-east-1.amazonaws.com/`

For more information about dual-stack endpoints, see Using Amazon S3 Dual-Stack Endpoints.

`WebsiteURL`

The Amazon S3 website endpoint for the specified bucket.

Example (IPv4): `http://mystack-mybucket-kdwwxmddtr2g.s3-website-us-east-1.amazonaws.com/`

Example (IPv6): `http://mystack-mybucket-kdwwxmddtr2g.s3.dualstack.us-east-1.amazonaws.com/`

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Examples

Associate a Replication Configuration IAM Role with an S3 Bucket

The following example creates an S3 bucket and grants it permission to write to a replication bucket by using an AWS Identity and Access Management (IAM) role. To avoid a circular dependency,the role's policy is declared as a separate resource. The bucket depends on the `WorkItemBucketBackupRole` role. If the policy is included in the role, the role also depends on the bucket.

JSON

```
"RecordServiceS3Bucket": {
  "Type": "AWS::S3::Bucket",
  "DeletionPolicy": "Retain",
  "Properties": {
    "ReplicationConfiguration": {
      "Role": {
        "Fn::GetAtt": [
          "WorkItemBucketBackupRole",
          "Arn"
        ]
      },
      "Rules": [{
        "Destination": {
          "Bucket": {
            "Fn::Join": [ "", [
              "arn:aws:s3:::", {
                "Fn::Join": [ "-", [
                  { "Ref": "AWS::Region" },
                  { "Ref": "AWS::StackName" },
                  "replicationbucket"
                ]]
              }
            ]]
          },
          "StorageClass": "STANDARD"
        },
        "Id": "Backup",
        "Prefix": "",
        "Status": "Enabled"
      }]
    },
    "VersioningConfiguration": {
      "Status": "Enabled"
    }
  }
},
"WorkItemBucketBackupRole": {
  "Type": "AWS::IAM::Role",
```

```
    "Properties": {
      "AssumeRolePolicyDocument": {
        "Statement": [{
          "Action": [ "sts:AssumeRole" ],
          "Effect": "Allow",
          "Principal": {
            "Service": [ "s3.amazonaws.com" ]
          }
        }]
      }
    }
  },
  "BucketBackupPolicy": {
    "Type": "AWS::IAM::Policy",
    "Properties": {
      "PolicyDocument": {
        "Statement": [{
          "Action": [
            "s3:GetReplicationConfiguration",
            "s3:ListBucket"
          ],
          "Effect": "Allow",
          "Resource": [{
            "Fn::Join": [ "", [
                "arn:aws:s3:::", {
                  "Ref": "RecordServiceS3Bucket"
                }
              ]
            ]
          }]
        },{
          "Action": [
            "s3:GetObjectVersion",
            "s3:GetObjectVersionAcl"
          ],
          "Effect": "Allow",
          "Resource": [{
            "Fn::Join": [ "", [
                "arn:aws:s3:::", {
                  "Ref": "RecordServiceS3Bucket"
                },
                "/*"
              ]
            ]
          }]
        }, {
          "Action": [
            "s3:ReplicateObject",
            "s3:ReplicateDelete"
          ],
          "Effect": "Allow",
          "Resource": [{
            "Fn::Join": [ "", [
               "arn:aws:s3:::", {
                 "Fn::Join": [ "-", [
                   { "Ref": "AWS::Region" },
                   { "Ref": "AWS::StackName" },
                   "replicationbucket"
                 ]]
               },
               "/*"
            ]]
          }]
        }]
      },
      "PolicyName": "BucketBackupPolicy",
```

```
    "Roles": [{
      "Ref": "WorkItemBucketBackupRole"
    }]
  }
}
```

YAML

```
RecordServiceS3Bucket:
  Type: AWS::S3::Bucket
  DeletionPolicy: Retain
  Properties:
    ReplicationConfiguration:
      Role: !GetAtt [WorkItemBucketBackupRole, Arn]
      Rules:
      - Destination:
          Bucket: !Join ['', ['arn:aws:s3:::', !Join ['-', [!Ref 'AWS::Region', !Ref
 'AWS::StackName',
                  replicationbucket]]]]
          StorageClass: STANDARD
        Id: Backup
        Prefix: ''
        Status: Enabled
    VersioningConfiguration:
      Status: Enabled
WorkItemBucketBackupRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Statement:
      - Action: ['sts:AssumeRole']
        Effect: Allow
        Principal:
          Service: [s3.amazonaws.com]
BucketBackupPolicy:
  Type: AWS::IAM::Policy
  Properties:
    PolicyDocument:
      Statement:
      - Action: ['s3:GetReplicationConfiguration', 's3:ListBucket']
        Effect: Allow
        Resource:
        - !Join ['', ['arn:aws:s3:::', !Ref 'RecordServiceS3Bucket']]
      - Action: ['s3:GetObjectVersion', 's3:GetObjectVersionAcl']
        Effect: Allow
        Resource:
        - !Join ['', ['arn:aws:s3:::', !Ref 'RecordServiceS3Bucket', /*]]
      - Action: ['s3:ReplicateObject', 's3:ReplicateDelete']
        Effect: Allow
        Resource:
        - !Join ['', ['arn:aws:s3:::', !Join ['-', [!Ref 'AWS::Region', !Ref
 'AWS::StackName',
                replicationbucket]], /*]]
    PolicyName: BucketBackupPolicy
    Roles: [!Ref 'WorkItemBucketBackupRole']
```

Configure a Static Website with a Routing Rule

In this example, `AWS::S3::Bucket's Fn::GetAtt` values are used to provide outputs. If an HTTP 404 error occurs, the routing rule redirects requests to an EC2 instance and inserts the object key prefix `report-404/` in the redirect. For example, if you request a page called `ExamplePage.html` and it results in a HTTP 404 error, the request is routed to a page called `report-404/ExamplePage.html` on the specified instance. For all other HTTP error codes, `error.html` is returned.

JSON

```
"Resources" : {
   "S3Bucket" : {
      "Type" : "AWS::S3::Bucket",
      "Properties" : {
         "AccessControl" : "PublicRead",
         "BucketName" : "PublicBucket",
         "WebsiteConfiguration" : {
            "IndexDocument" : "index.html",
            "ErrorDocument" : "error.html",
            "RoutingRules": [
                {
                    "RoutingRuleCondition": {
                        "HttpErrorCodeReturnedEquals": "404",
                        "KeyPrefixEquals": "out1/"
                    },
                    "RedirectRule": {
                        "HostName": "ec2-11-22-333-44.compute-1.amazonaws.com",
                        "ReplaceKeyPrefixWith": "report-404/"
                    }
                }
            ]
         }
      },
      "DeletionPolicy" : "Retain"
   }
},

"Outputs" : {
   "WebsiteURL" : {
      "Value" : { "Fn::GetAtt" : [ "S3Bucket", "WebsiteURL" ] },
      "Description" : "URL for website hosted on S3"
   },
   "S3BucketSecureURL" : {
      "Value" : { "Fn::Join" : [
         "", [ "https://", { "Fn::GetAtt" : [ "S3Bucket", "DomainName" ] } ]
      ] },
      "Description" : "Name of S3 bucket to hold website content"
   }
}
```

YAML

```
Resources:
  S3Bucket:
    Type: AWS::S3::Bucket
    Properties:
      AccessControl: PublicRead
      BucketName: PublicBucket
      WebsiteConfiguration:
        IndexDocument: index.html
        ErrorDocument: error.html
        RoutingRules:
        - RoutingRuleCondition:
            HttpErrorCodeReturnedEquals: '404'
            KeyPrefixEquals: out1/
          RedirectRule:
            HostName: ec2-11-22-333-44.compute-1.amazonaws.com
            ReplaceKeyPrefixWith: report-404/
    DeletionPolicy: Retain
Outputs:
  WebsiteURL:
    Value: !GetAtt [S3Bucket, WebsiteURL]
```

```
    Description: URL for website hosted on S3
  S3BucketSecureURL:
    Value: !Join ['', ['https://', !GetAtt [S3Bucket, DomainName]]]
    Description: Name of S3 bucket to hold website content
```

Enable Cross-Origin Resource Sharing

The following example template shows an S3 bucket with two cross-origin resource sharing rules.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Resources": {
        "S3Bucket": {
            "Type": "AWS::S3::Bucket",
            "Properties": {
                "AccessControl": "PublicReadWrite",
                "CorsConfiguration": {
                    "CorsRules": [
                        {
                            "AllowedHeaders": [
                                "*"
                            ],
                            "AllowedMethods": [
                                "GET"
                            ],
                            "AllowedOrigins": [
                                "*"
                            ],
                            "ExposedHeaders": [
                                "Date"
                            ],
                            "Id": "myCORSRuleId1",
                            "MaxAge": "3600"
                        },
                        {
                            "AllowedHeaders": [
                                "x-amz-*"
                            ],
                            "AllowedMethods": [
                                "DELETE"
                            ],
                            "AllowedOrigins": [
                                "http://www.example1.com",
                                "http://www.example2.com"
                            ],
                            "ExposedHeaders": [
                                "Connection",
                                "Server",
                                "Date"
                            ],
                            "Id": "myCORSRuleId2",
                            "MaxAge": "1800"
                        }
                    ]
                }
            }
        }
    },
    "Outputs": {
        "BucketName": {
            "Value": {
                "Ref": "S3Bucket"
```

```
            },
            "Description": "Name of the sample Amazon S3 bucket with CORS enabled."
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  S3Bucket:
    Type: AWS::S3::Bucket
    Properties:
      AccessControl: PublicReadWrite
      CorsConfiguration:
        CorsRules:
        - AllowedHeaders: ['*']
          AllowedMethods: [GET]
          AllowedOrigins: ['*']
          ExposedHeaders: [Date]
          Id: myCORSRuleId1
          MaxAge: '3600'
        - AllowedHeaders: [x-amz-*]
          AllowedMethods: [DELETE]
          AllowedOrigins: ['http://www.example1.com', 'http://www.example2.com']
          ExposedHeaders: [Connection, Server, Date]
          Id: myCORSRuleId2
          MaxAge: '1800'
Outputs:
  BucketName:
    Value: !Ref 'S3Bucket'
    Description: Name of the sample Amazon S3 bucket with CORS enabled.
```

Manage the Lifecycle for Amazon S3 Objects

The following example template shows an S3 bucket with a lifecycle configuration rule. The rule applies to all objects with the `glacier` key prefix. The objects are transitioned to Amazon Glacier after one day, and deleted after one year.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Resources": {
        "S3Bucket": {
            "Type": "AWS::S3::Bucket",
            "Properties": {
                "AccessControl": "PublicReadWrite",
                "LifecycleConfiguration": {
                    "Rules": [
                        {
                            "Id": "GlacierRule",
                            "Prefix": "glacier",
                            "Status": "Enabled",
                            "ExpirationInDays": "365",
                            "Transition": {
                                "TransitionInDays": "1",
                                "StorageClass": "Glacier"
                            }
                        }
                    ]
                }
            }
```

```
        }
    },
    "Outputs": {
        "BucketName": {
            "Value": {
                "Ref": "S3Bucket"
            },
            "Description": "Name of the sample Amazon S3 bucket with a lifecycle
 configuration."
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  S3Bucket:
    Type: AWS::S3::Bucket
    Properties:
      AccessControl: PublicReadWrite
      LifecycleConfiguration:
        Rules:
        - Id: GlacierRule
          Prefix: glacier
          Status: Enabled
          ExpirationInDays: '365'
          Transition:
            TransitionInDays: '1'
            StorageClass: Glacier
Outputs:
  BucketName:
    Value: !Ref 'S3Bucket'
    Description: Name of the sample Amazon S3 bucket with a lifecycle configuration.
```

Log Access Requests for a Specific S3 Bucket

The following example template creates two S3 buckets. The `LoggingBucket` bucket store the logs from the `S3Bucket` bucket. To receive logs from the `S3Bucket` bucket, the logging bucket requires log delivery write permissions.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Resources": {
        "S3Bucket": {
            "Type": "AWS::S3::Bucket",
            "Properties": {
                "AccessControl": "PublicRead",
                "LoggingConfiguration": {
                    "DestinationBucketName": {"Ref" : "LoggingBucket"},
                    "LogFilePrefix": "testing-logs"
                }
            }
        },
        "LoggingBucket": {
            "Type": "AWS::S3::Bucket",
            "Properties": {
                "AccessControl": "LogDeliveryWrite"
            }
        }
    },
```

```
    "Outputs": {
        "BucketName": {
            "Value": {
                "Ref": "S3Bucket"
            },
            "Description": "Name of the sample Amazon S3 bucket with a logging
 configuration."
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  S3Bucket:
    Type: AWS::S3::Bucket
    Properties:
      AccessControl: PublicRead
      LoggingConfiguration:
        DestinationBucketName: !Ref 'LoggingBucket'
        LogFilePrefix: testing-logs
  LoggingBucket:
    Type: AWS::S3::Bucket
    Properties:
      AccessControl: LogDeliveryWrite
Outputs:
  BucketName:
    Value: !Ref 'S3Bucket'
    Description: Name of the sample Amazon S3 bucket with a logging configuration.
```

Receive S3 Bucket Notifications to an Amazon SNS Topic

The following example template shows an S3 bucket with a notification configuration that sends an event to the specified Amazon SNS topic when Amazon S3 has lost all replicas of an object.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Resources": {
        "S3Bucket": {
            "Type": "AWS::S3::Bucket",
            "Properties": {
                "AccessControl": "PublicReadWrite",
                "NotificationConfiguration": {
                    "TopicConfigurations": [
                        {
                            "Topic": "arn:aws:sns:us-east-1:123456789012:TestTopic",
                            "Event": "s3:ReducedRedundancyLostObject"
                        }
                    ]
                }
            }
        }
    },
    "Outputs": {
        "BucketName": {
            "Value": {
                "Ref": "S3Bucket"
            },
            "Description": "Name of the sample Amazon S3 bucket with a notification
 configuration."
```

```
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  S3Bucket:
    Type: AWS::S3::Bucket
    Properties:
      AccessControl: PublicReadWrite
      NotificationConfiguration:
        TopicConfigurations:
        - Topic: arn:aws:sns:us-east-1:123456789012:TestTopic
          Event: s3:ReducedRedundancyLostObject
Outputs:
  BucketName:
    Value: !Ref 'S3Bucket'
    Description: Name of the sample Amazon S3 bucket with a notification configuration.
```

Replicate Objects and Store Them in Another S3 Bucket

The following example includes two replication rules. Amazon S3 replicates objects with the `MyPrefix` or `MyOtherPrefix` prefixes and stores them in the `my-replication-bucket` bucket, which must be in a different AWS Region than the `S3Bucket` bucket.

JSON

```
"S3Bucket": {
  "Type": "AWS::S3::Bucket",
  "Properties": {
    "VersioningConfiguration":{
      "Status":"Enabled"
    },
    "ReplicationConfiguration": {
      "Role": "arn:aws:iam::123456789012:role/replication_role",
      "Rules": [
        {
          "Id": "MyRule1",
          "Status": "Enabled",
          "Prefix": "MyPrefix",
          "Destination": {
            "Bucket": "arn:aws:s3:::my-replication-bucket",
            "StorageClass": "STANDARD"
          }
        },
        {
          "Status": "Enabled",
          "Prefix": "MyOtherPrefix",
          "Destination": {
            "Bucket": "arn:aws:s3:::my-replication-bucket"
          }
        }
      ]
    }
  }
}
```

YAML

```
S3Bucket:
```

```
  Type: AWS::S3::Bucket
  Properties:
    VersioningConfiguration:
      Status: Enabled
    ReplicationConfiguration:
      Role: arn:aws:iam::123456789012:role/replication_role
      Rules:
      - Id: MyRule1
        Status: Enabled
        Prefix: MyPrefix
        Destination:
          Bucket: arn:aws:s3:::my-replication-bucket
          StorageClass: STANDARD
      - Status: Enabled
        Prefix: MyOtherPrefix
        Destination:
          Bucket: arn:aws:s3:::my-replication-bucket
```

More Info

- For more examples, see Amazon S3 Template Snippets (p. 356).
- DeletionPolicy Attribute (p. 1297)
- Access Control List (ACL) Overview in the *Amazon Simple Storage Service Developer Guide*
- Hosting a Static Website on Amazon S3 in the *Amazon Simple Storage Service Developer Guide*

AWS::S3::BucketPolicy

The AWS::S3::BucketPolicy type applies an Amazon S3 bucket policy to an Amazon S3 bucket.

AWS::S3::BucketPolicy Snippet: Declaring an Amazon S3 Bucket Policy (p. 323)

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::S3::BucketPolicy",
   "Properties" : {
      "Bucket" : String,
      "PolicyDocument" : JSON
   }
}
```

YAML

```
Type: "AWS::S3::BucketPolicy"
```

```
Properties:
  Bucket: String
  PolicyDocument: JSON
```

Properties

`Bucket`

The Amazon S3 bucket that the policy applies to.

Required: Yes

Type: String

You cannot update this property. If you want to add or remove a bucket from a bucket policy, you must modify your AWS CloudFormation template by creating a new bucket policy resource and removing the old one. Then use the modified template to update your AWS CloudFormation stack.

`PolicyDocument`

A policy document containing permissions to add to the specified bucket. For more information, see Access Policy Language Overview in the *Amazon Simple Storage Service Developer Guide*.

Required: Yes

Type: JSON object

Update requires: No interruption (p. 90)

Examples

Bucket policy that allows GET requests from specific referers

The following sample is a bucket policy that is attached to the `myExampleBucket` bucket and allows GET requests that originate from `www.example.com` and `example.com`:

JSON

```
"SampleBucketPolicy" : {
  "Type" : "AWS::S3::BucketPolicy",
  "Properties" : {
    "Bucket" : {"Ref" : "myExampleBucket"},
    "PolicyDocument": {
      "Statement":[{
     "Action":["s3:GetObject"],
     "Effect":"Allow",
     "Resource": { "Fn::Join" : ["", ["arn:aws:s3:::", { "Ref" : "myExampleBucket" } , "/
*" ]]},
     "Principal":"*",
        "Condition":{
          "StringLike":{
            "aws:Referer":[
              "http://www.example.com/*",
              "http://example.com/*"
            ]
          }
        }
      }]
    }
```

```
  }
}
```

YAML

```
SampleBucketPolicy:
  Type: "AWS::S3::BucketPolicy"
  Properties:
    Bucket:
      Ref: "myExampleBucket"
    PolicyDocument:
      Statement:
        -
          Action:
            - "s3:GetObject"
          Effect: "Allow"
          Resource:
            Fn::Join:
              - ""
              -
                - "arn:aws:s3:::"
                -
                  Ref: "myExampleBucket"
                - "/*"
          Principal: "*"
          Condition:
            StringLike:
              aws:Referer:
                - "http://www.example.com/*"
                - "http://example.com/*"
```

AWS::SDB::Domain

Use the `AWS::SDB::Domain` resource to declare an Amazon SimpleDB domain. When you specify `AWS::SDB::Domain` as an argument in a `Ref` function, AWS CloudFormation returns the value of the `DomainName`.

> **Important**
> The `AWS::SDB::Domain` resource does not allow any updates, including metadata updates.

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::SDB::Domain",
  "Properties" : {
    "Description" : String
  }
}
```

YAML

```
Type: "AWS::SDB::Domain"
Properties:
  Description: String
```

Properties

`Description`

Information about the Amazon SimpleDB domain.

Required: No

Type: String

Update requires: Updates are not supported.

AWS::SNS::Subscription

The `AWS::SNS::Subscription` resource subscribes an endpoint to an Amazon Simple Notification Service (Amazon SNS) topic. The owner of the endpoint must confirm the subscription before Amazon SNS creates the subscription.

Topics

Syntax

JSON

```
{
  "Type" : "AWS::SNS::Subscription",
  "Properties" : {
    "Endpoint" : String,
    "Protocol" : String,
    "TopicArn" : String
  }
}
```

YAML

```
Type: "AWS::SNS::Subscription"
Properties:
  Endpoint: String
  Protocol: String
  TopicArn: String
```

Properties

`Endpoint`

The endpoint that receives notifications from the Amazon SNS topic. The endpoint value depends on the protocol that you specify. For more information, see the Subscribe Endpoint parameter in the *Amazon Simple Notification Service API Reference*.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Protocol`

The subscription's protocol. For more information, see the Subscribe Protocol parameter in the *Amazon Simple Notification Service API Reference*.

Required: No

Type: String

Update requires: Replacement (p. 90)

`TopicArn`

The Amazon Resource Name (ARN) of the topic to subscribe to.

Required: No

Type: String

Update requires: Replacement (p. 90)

Example

The following

JSON

```
"MySubscription" : {
  "Type" : "AWS::SNS::Subscription",
  "Properties" : {
    "Endpoint" : "test@email.com",
    "Protocol" : "email",
    "TopicArn" : {"Ref" : "MySNSTopic"}
  }
}
```

YAML

```
MySubscription:
  Type: AWS::SNS::Subscription
  Properties:
    Endpoint: test@email.com
    Protocol: email
    TopicArn: !Ref 'MySNSTopic'
```

AWS::SNS::Topic

The `AWS::SNS::Topic` type creates an Amazon Simple Notification Service (Amazon SNS) topic.

Topics

- Examples (p. 956)
- See Also (p. 957)

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::SNS::Topic",
  "Properties" : {
    "DisplayName" : String,
    "Subscription" : [ SNS Subscription, ... ],
    "TopicName" : String
  }
}
```

YAML

```
Type: "AWS::SNS::Topic"
Properties:
  DisplayName: String
  Subscription:
    SNS Subscription
  TopicName: String
```

Properties

`DisplayName`

A developer-defined string that can be used to identify this SNS topic.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Subscription`

The SNS subscriptions (endpoints) for this topic.

Required: No

Type: List of SNS Subscriptions (p. 1271)

Update requires: No interruption (p. 90)

`TopicName`

A name for the topic. If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the topic name. For more information, see Name Type (p. 1217).

Important
If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

For the `AWS::SNS::Topic` resource, the `Ref` intrinsic function returns the topic ARN, for example: `arn:aws:sns:us-east-1:123456789012:mystack-mytopic-NZJ5JSMVGFIE`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`TopicName`

Returns the name for an Amazon SNS topic.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Examples

An example of an SNS topic subscribed to by two SQS queues:

JSON

```
"MySNSTopic" : {
   "Type" : "AWS::SNS::Topic",
   "Properties" : {
      "Subscription" : [
         { "Endpoint" : { "Fn::GetAtt" : [ "MyQueue1", "Arn" ] }, "Protocol" : "sqs" },
         { "Endpoint" : { "Fn::GetAtt" : [ "MyQueue2", "Arn" ] }, "Protocol" : "sqs" }
      ],
      "TopicName" : "SampleTopic"
   }
}
```

YAML

```
MySNSTopic:
  Type: "AWS::SNS::Topic"
  Properties:
    Subscription:
      -
        Endpoint:
          Fn::GetAtt:
            - "MyQueue1"
            - "Arn"
        Protocol: "sqs"
      -
```

```
        Endpoint:
          Fn::GetAtt:
            - "MyQueue2"
            - "Arn"
        Protocol: "sqs"
    TopicName: "SampleTopic"
```

See Also

- Using an AWS CloudFormation Template to Create a Topic that Sends Messages to Amazon SQS Queues in the *Amazon Simple Notification Service Developer Guide*

AWS::SNS::TopicPolicy

The `AWS::SNS::TopicPolicy` resource associates Amazon SNS topics with a policy.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::SNS::TopicPolicy",
  "Properties" :
    {
      "PolicyDocument" : JSON,
      "Topics" : [ List of SNS topic ARNs, ... ]
    }
}
```

YAML

```
Type: "AWS::SNS::TopicPolicy"
Properties:
  PolicyDocument: JSON
  Topics:
    - List of SNS topic ARNs
```

Properties

`PolicyDocument`

A policy document that contains permissions to add to the specified SNS topics.

Required: Yes

Type: JSON object

Update requires: No interruption (p. 90)

`Topics`

The Amazon Resource Names (ARN) of the topics to which you want to add the policy. You can use the Ref function (p. 1343) to specify an AWS::SNS::Topic (p. 954) resource.

Required: Yes

Type: A list of Amazon SNS topics ARNs

Update requires: No interruption (p. 90)

For sample `AWS::SNS::TopicPolicy` snippets, see Declaring an Amazon SNS Topic Policy (p. 324).

AWS::SQS::Queue

The `AWS::SQS::Queue` resource creates an Amazon Simple Queue Service (Amazon SQS) queue.

For more information about creating FIFO (first-in-first-out) queues, see the tutorial Create a queue using AWS CloudFormation in the *Amazon Simple Queue Service Developer Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::SQS::Queue",
   "Properties" : {
      "ContentBasedDeduplication" : Boolean,
      "DelaySeconds": Integer,
      "FifoQueue" : Boolean,
      "MaximumMessageSize": Integer,
      "MessageRetentionPeriod": Integer,
      "QueueName": String,
      "ReceiveMessageWaitTimeSeconds": Integer,
      "RedrivePolicy": RedrivePolicy,
      "VisibilityTimeout": Integer
   }
}
```

YAML

```
Type: "AWS::SQS::Queue"
Properties:
  ContentBasedDeduplication: Boolean
  DelaySeconds: Integer
  FifoQueue: Boolean
```

```
  MaximumMessageSize: Integer
  MessageRetentionPeriod: Integer
  QueueName: String
  ReceiveMessageWaitTimeSeconds: Integer
  RedrivePolicy:
    RedrivePolicy
  VisibilityTimeout: Integer
```

Properties

`ContentBasedDeduplication`

For First-In-First-Out (FIFO) queues, specifies whether to enable content-based deduplication. During the deduplication interval, Amazon SQS treats messages that are sent with identical content as duplicates and delivers only one copy of the message. For more information, see the `ContentBasedDeduplication` attribute for the CreateQueue action in the *Amazon Simple Queue Service API Reference*.

Required: No

Type: Boolean

Update requires: No interruption (p. 90)

`DelaySeconds`

The time in seconds that the delivery of all messages in the queue is delayed. You can specify an integer value of `0` to `900` (15 minutes). The default value is `0`.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`FifoQueue`

Indicates whether this queue is a FIFO queue. For more information, see FIFO (First-In-First-Out) Queues in the *Amazon Simple Queue Service Developer Guide*.

Required: No

Type: Boolean

Update requires: Replacement (p. 90)

`MaximumMessageSize`

The limit of how many bytes that a message can contain before Amazon SQS rejects it. You can specify an integer value from `1024` bytes (1 KiB) to `262144` bytes (256 KiB). The default value is `262144` (256 KiB).

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`MessageRetentionPeriod`

The number of seconds that Amazon SQS retains a message. You can specify an integer value from `60` seconds (1 minute) to `1209600` seconds (14 days). The default value is `345600` seconds (4 days).

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`QueueName`

A name for the queue. To create a FIFO queue, the name of your FIFO queue must end with the `.fifo` suffix. For more information, see FIFO (First-In-First-Out) Queues in the *Amazon Simple Queue Service Developer Guide*.

If you don't specify a name, AWS CloudFormation generates a unique physical ID and uses that ID for the queue name. For more information, see Name Type (p. 1217).

Important
If you specify a name, you cannot perform updates that require replacement of this resource. You can perform updates that require no or some interruption. If you must replace the resource, specify a new name.

Required: No

Type: String

Update requires: Replacement (p. 90)

`ReceiveMessageWaitTimeSeconds`

Specifies the duration, in seconds, that the `ReceiveMessage` action call waits until a message is in the queue in order to include it in the response, as opposed to returning an empty response if a message is not yet available. You can specify an integer from `1` to `20`. The short polling is used as the default or when you specify `0` for this property. For more information, see Amazon SQS Long Poll.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

`RedrivePolicy`

Specifies an existing dead letter queue to receive messages after the source queue (this queue) fails to process a message a specified number of times.

Required: No

Type: Amazon SQS RedrivePolicy (p. 1272)

Update requires: No interruption (p. 90)

`VisibilityTimeout`

The length of time during which a message will be unavailable after a message is delivered from the queue. This blocks other components from receiving the same message and gives the initial component time to process and delete the message from the queue.

Values must be from 0 to 43200 seconds (12 hours). If you don't specify a value, AWS CloudFormation uses the default value of 30 seconds.

For more information about Amazon SQS queue visibility timeouts, see Visibility Timeout in the *Amazon Simple Queue Service Developer Guide*.

Required: No

Type: Integer

Update requires: No interruption (p. 90)

Return Values

Ref

The `AWS::SQS::Queue` type returns the queue URL, for example: `https://sqs.us-east-1.amazonaws.com/123456789012/aa4-MyQueue-Z5NOSZO2PZE9`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`Arn`

Returns the Amazon Resource Name (ARN) of the queue. For example: `arn:aws:sqs:us-east-1:123456789012:mystack-myqueue-15PG5C2FC1CW8`.

`QueueName`

Returns the queue name. For example:

`mystack-myqueue-1VF9BKQH5BJVI`

Examples

SQS Queue with Cloudwatch Alarms

JSON

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",

  "Description" : "AWS CloudFormation Sample Template SQS_With_CloudWatch_Alarms: Sample
 template showing how to create an SQS queue with Amazon CloudWatch alarms on queue depth.
 **WARNING** This template creates an Amazon SQS queue and one or more Amazon CloudWatch
 alarms. You will be billed for the AWS resources used if you create a stack from this
 template.",

  "Parameters" : {
    "AlarmEmail": {
      "Default": "nobody@amazon.com",
      "Description": "Email address to notify if operational problems arise",
      "Type": "String"
    }
  },

  "Resources" : {
    "MyQueue" : {
      "Type" : "AWS::SQS::Queue",
      "Properties" : {
         "QueueName" : "SampleQueue"
      }
    },
    "AlarmTopic": {
      "Type": "AWS::SNS::Topic",
      "Properties": {
        "Subscription": [{
          "Endpoint": { "Ref": "AlarmEmail" },
          "Protocol": "email"
```

```
        }]
      }
    },
    "QueueDepthAlarm": {
      "Type": "AWS::CloudWatch::Alarm",
      "Properties": {
        "AlarmDescription": "Alarm if queue depth grows beyond 10 messages",
        "Namespace": "AWS/SQS",
        "MetricName": "ApproximateNumberOfMessagesVisible",
        "Dimensions": [{
          "Name": "QueueName",
          "Value" : { "Fn::GetAtt" : ["MyQueue", "QueueName"] }
        }],
        "Statistic": "Sum",
        "Period": "300",
        "EvaluationPeriods": "1",
        "Threshold": "10",
        "ComparisonOperator": "GreaterThanThreshold",
        "AlarmActions": [{
          "Ref": "AlarmTopic"
        }],
        "InsufficientDataActions": [{
          "Ref": "AlarmTopic"
        }]
      }
    }
  },
  "Outputs" : {
    "QueueURL" : {
      "Description" : "URL of newly created SQS Queue",
      "Value" : { "Ref" : "MyQueue" }
    },
    "QueueARN" : {
      "Description" : "ARN of newly created SQS Queue",
      "Value" : { "Fn::GetAtt" : ["MyQueue", "Arn"]}
    },
    "QueueName" : {
      "Description" : "Name newly created SQS Queue",
      "Value" : { "Fn::GetAtt" : ["MyQueue", "QueueName"]}
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Description: "AWS CloudFormation Sample Template SQS_With_CloudWatch_Alarms: Sample
 template showing how to create an SQS queue with Amazon CloudWatch alarms on queue depth.
 **WARNING** This template creates an Amazon SQS queue and one or more Amazon CloudWatch
 alarms. You will be billed for the AWS resources used if you create a stack from this
 template."
Parameters:
  AlarmEmail:
    Default: "nobody@amazon.com"
    Description: "Email address to notify if operational problems arise"
    Type: "String"
Resources:
  MyQueue:
    Type: "AWS::SQS::Queue"
    Properties:
      QueueName: "SampleQueue"
  AlarmTopic:
    Type: "AWS::SNS::Topic"
    Properties:
```

```
      Subscription:
        -
          Endpoint:
            Ref: "AlarmEmail"
          Protocol: "email"
  QueueDepthAlarm:
    Type: "AWS::CloudWatch::Alarm"
    Properties:
      AlarmDescription: "Alarm if queue depth grows beyond 10 messages"
      Namespace: "AWS/SQS"
      MetricName: "ApproximateNumberOfMessagesVisible"
      Dimensions:
        -
          Name: "QueueName"
          Value:
            Fn::GetAtt:
              - "MyQueue"
              - "QueueName"
      Statistic: "Sum"
      Period: "300"
      EvaluationPeriods: "1"
      Threshold: "10"
      ComparisonOperator: "GreaterThanThreshold"
      AlarmActions:
        -
          Ref: "AlarmTopic"
      InsufficientDataActions:
        -
          Ref: "AlarmTopic"
Outputs:
  QueueURL:
    Description: "URL of newly created SQS Queue"
    Value:
      Ref: "MyQueue"
  QueueARN:
    Description: "ARN of newly created SQS Queue"
    Value:
      Fn::GetAtt:
        - "MyQueue"
        - "Arn"
  QueueName:
    Description: "Name newly created SQS Queue"
    Value:
      Fn::GetAtt:
        - "MyQueue"
        - "QueueName"
```

SQS Queue with a Dead Letter Queue

The following sample creates a source queue and a dead letter queue. Because the source queue specifies the dead letter queue in its redrive policy, the source queue is dependent on the creation of the dead letter queue.

JSON

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",

  "Resources" : {
    "MySourceQueue" : {
      "Type" : "AWS::SQS::Queue",
      "Properties" : {
        "RedrivePolicy": {
          "deadLetterTargetArn" : {"Fn::GetAtt" : [ "MyDeadLetterQueue" , "Arn" ]},
```

```
          "maxReceiveCount" : 5
        }
      }
    },
    "MyDeadLetterQueue" : {
      "Type" : "AWS::SQS::Queue"
    }
  },

  "Outputs" : {
    "SourceQueueURL" : {
      "Description" : "URL of the source queue",
      "Value" : { "Ref" : "MySourceQueue" }
    },
    "SourceQueueARN" : {
      "Description" : "ARN of the source queue",
      "Value" : { "Fn::GetAtt" : ["MySourceQueue", "Arn"]}
    },
    "DeadLetterQueueURL" : {
      "Description" : "URL of the dead letter queue",
      "Value" : { "Ref" : "MyDeadLetterQueue" }
    },
    "DeadLetterQueueARN" : {
      "Description" : "ARN of the dead letter queue",
      "Value" : { "Fn::GetAtt" : ["MyDeadLetterQueue", "Arn"]}
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  MySourceQueue:
    Type: "AWS::SQS::Queue"
    Properties:
      RedrivePolicy:
        deadLetterTargetArn:
          Fn::GetAtt:
            - "MyDeadLetterQueue"
            - "Arn"
        maxReceiveCount: 5
  MyDeadLetterQueue:
    Type: "AWS::SQS::Queue"
Outputs:
  SourceQueueURL:
    Description: "URL of the source queue"
    Value:
      Ref: "MySourceQueue"
  SourceQueueARN:
    Description: "ARN of the source queue"
    Value:
      Fn::GetAtt:
        - "MySourceQueue"
        - "Arn"
  DeadLetterQueueURL:
    Description: "URL of the dead letter queue"
    Value:
      Ref: "MyDeadLetterQueue"
  DeadLetterQueueARN:
    Description: "ARN of the dead letter queue"
    Value:
      Fn::GetAtt:
        - "MyDeadLetterQueue"
```

```
        - "Arn"
```

More Info

- CreateQueue in the *Amazon Simple Queue Service API Reference*
- What is Amazon Simple Queue Service? in the *Amazon Simple Queue Service Developer Guide*

AWS::SQS::QueuePolicy

The AWS::SQS::QueuePolicy type applies a policy to Amazon SQS queues.

AWS::SQS::QueuePolicy Snippet: Declaring an Amazon SQS Policy (p. 325)

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "Type" : "AWS::SQS::QueuePolicy",
   "Properties" : {
      "PolicyDocument" : JSON,
      "Queues" : [ String, ... ]
   }
}
```

YAML

```
Type: "AWS::SQS::QueuePolicy"
Properties:
  PolicyDocument: JSON
  Queues:
    - String
```

Properties

`PolicyDocument`

A policy document that contains the permissions for the specified Amazon SQS queues. For more information about Amazon SQS policies, see Creating Custom Policies Using the Access Policy Language in the *Amazon Simple Queue Service Developer Guide*.

Required: Yes

Type: JSON object

Update requires: No interruption (p. 90)

`Queues`

The URLs of the queues to which you want to add the policy. You can use the Ref function (p. 1343) to specify an AWS::SQS::Queue (p. 958) resource.

Required: Yes

Type: List of strings

Update requires: No interruption (p. 90)

AWS::SSM::Association

The `AWS::SSM::Association` resource associates an Amazon EC2 Systems Manager (SSM) document with EC2 instances that contain a configuration agent to process the document.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::SSM::Association",
  "Properties" : {
    "DocumentVersion" : String,
    "InstanceId" : String,
    "Name" : String,
    "Parameters" : { String: [String, ...] },
    "ScheduleExpression" : String,
    "Targets" : [ Targets (p. 1270) ]
  }
}
```

YAML

```
Type: "AWS::SSM::Association"
Properties:
  DocumentVersion: String
  InstanceId: String
  Name: String
  Parameters:
    String:
      - String
  ScheduleExpression: String
  Targets:
    - Targets (p. 1270)
```

Properties

`DocumentVersion`

The version of the SSM document to associate with the target.

Required: No

Type: String

Update requires: No interruption (p. 90)

`InstanceId`

The ID of the instance that the SSM document is associated with.

Required: Conditional. You must specify the `InstanceId` or `Targets` property.

Type: String

Update requires: Replacement (p. 90)

`Name`

The name of the SSM document.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Parameters`

Parameter values that the SSM document uses at runtime.

Required: No

Type: String to list-of-strings map

Update requires: No interruption (p. 90)

`ScheduleExpression`

A Cron expression that specifies when the association is applied to the target. For supported expressions, see the `ScheduleExpression` parameter for the CreateAssociation action in the *Amazon EC2 Systems Manager API Reference*.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Targets`

The targets that the SSM document sends commands to.

Required: Conditional. You must specify the `InstanceId` or `Targets` property.

Type: List of Amazon EC2 Systems Manager Association Targets (p. 1270)

Update requires: Replacement (p. 90)

Example

The following example associates an SSM document with a specific instance. The ID of the instance is specified by the `myInstanceId` parameter.

JSON

```
"association": {
  "Type": "AWS::SSM::Association",
  "Properties": {
    "Name": {
      "Ref": "document"
    },
    "Parameters": {
      "Directory": ["myWorkSpace"]
    },
    "Targets": [{
      "Key": "InstanceIds",
      "Values": [{
        "Ref": "myInstanceId"
      }]
    }]
  }
}
```

YAML

```
association:
  Type: AWS::SSM::Association
  Properties:
    Name: !Ref 'document'
    Parameters:
      Directory: [FakeDirectory]
    Targets:
    - Key: InstanceIds
      Values: [!Ref 'myInstanceId']
```

AWS::SSM::Document

The `AWS::SSM::Document` resource creates an Amazon EC2 Systems Manager (SSM) document that describes an instance configuration, which you can use to set up and run commands on your instances.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::SSM::Document",
  "Properties" : {
    "Content" : JSON object,
    "DocumentType" : String
  }
}
```

YAML

```
Type: "AWS::SSM::Document"
Properties:
  Content: JSON object
  DocumentType: String
```

Properties

`Content`

A JSON object that describes an instance configuration. For more information, see Creating SSM Documents in the *Amazon EC2 User Guide for Linux Instances*.

Required: Yes

Type: JSON object

Update requires: Replacement (p. 90)

`DocumentType`

The type of document to create that relates to the purpose of your document, such as running commands, bootstrapping software, or automating tasks. For valid values, see the CreateDocument action in the *Amazon EC2 Systems Manager API Reference*.

Required: No

Type: String

Update requires: Replacement (p. 90)

Return Value

Ref

When you pass the logical ID of an `AWS::SSM::Document` resource to the intrinsic `Ref` function, the function returns the SSM document name, such as `ssm-myinstanceconfig-ABCNPH3XCAO6`.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

The following SSM document joins instances to a directory in AWS Directory Service. The three runtime configuration parameters specify which directory the instance joins. You specify these parameter values when you associate the document with an instance.

JSON

```
"document" : {
  "Type" : "AWS::SSM::Document",
  "Properties" : {
    "Content" : {
      "schemaVersion":"1.2",
      "description":"Join instances to an AWS Directory Service domain.",
      "parameters":{
        "directoryId":{
          "type":"String",
          "description":"(Required) The ID of the AWS Directory Service directory."
        },
        "directoryName":{
          "type":"String",
          "description":"(Required) The name of the directory; for example,
 test.example.com"
        },
        "dnsIpAddresses":{
          "type":"StringList",
          "default":[
          ],
          "description":"(Optional) The IP addresses of the DNS servers in the directory.
 Required when DHCP is not configured. Learn more at http://docs.aws.amazon.com/
directoryservice/latest/simple-ad/join_get_dns_addresses.html",
          "allowedPattern":"((25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\\.){3}(25[0-5]|2[0-4]
[0-9]|[01]?[0-9][0-9]?)"
        }
      },
      "runtimeConfig":{
        "aws:domainJoin":{
          "properties":{
            "directoryId":"{{ directoryId }}",
            "directoryName":"{{ directoryName }}",
            "dnsIpAddresses":"{{ dnsIpAddresses }}"
          }
        }
      }
    }
  }
}
```

YAML

```
document:
  Type: "AWS::SSM::Document"
  Properties:
    Content:
      schemaVersion: "1.2"
      description: "Join instances to an AWS Directory Service domain."
      parameters:
        directoryId:
          type: "String"
          description: "(Required) The ID of the AWS Directory Service directory."
        directoryName:
          type: "String"
          description: "(Required) The name of the directory; for example,
 test.example.com"
        dnsIpAddresses:
          type: "StringList"
          default: []
          description: "(Optional) The IP addresses of the DNS servers in the directory.
 Required when DHCP is not configured. Learn more at http://docs.aws.amazon.com/
directoryservice/latest/simple-ad/join_get_dns_addresses.html"
```

```
          allowedPattern: "((25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\\.){3}(25[0-5]|2[0-4]
[0-9]|[01]?[0-9][0-9]?)"
      runtimeConfig:
        aws:domainJoin:
          properties:
            directoryId: "{{ directoryId }}"
            directoryName: "{{ directoryName }}"
            dnsIpAddresses: "{{ dnsIpAddresses }}"
```

The following example shows how to associate the SSM document with an instance. The `DocumentName` property specifies the SSM document and the `AssociationParameters` property specifies values for the runtime configuration parameters.

JSON

```
"myEC2" : {
  "Type" : "AWS::EC2::Instance",
  "Properties" : {
    "ImageId" : {"Ref" : "myImageId"},
    "InstanceType" : "t2.micro",
    "SsmAssociations" : [ {
      "DocumentName" : {"Ref" : "document"},
      "AssociationParameters" : [
        { "Key" : "directoryId", "Value" : [ { "Ref" : "myDirectory" } ] },
        { "Key" : "directoryName", "Value" : ["testDirectory.example.com"] },
        { "Key" : "dnsIpAddresses", "Value" : { "Fn::GetAtt" : ["myDirectory",
 "DnsIpAddresses"] } }
      ]
    } ],
    "IamInstanceProfile" : {"Ref" : "myInstanceProfile"},
    "NetworkInterfaces" : [ {
      "DeviceIndex" : "0",
      "AssociatePublicIpAddress" : "true",
      "SubnetId" : {"Ref" : "mySubnet"}
    } ],
    "KeyName" : {"Ref" : "myKeyName"}
  }
}
```

YAML

```
myEC2:
  Type: "AWS::EC2::Instance"
  Properties:
    ImageId:
      Ref: "myImageId"
    InstanceType: "t2.micro"
    SsmAssociations:
      -
        DocumentName:
          Ref: "document"
        AssociationParameters:
          -
            Key: "directoryId"
            Value:
              -
                Ref: "myDirectory"
          -
            Key: "directoryName"
            Value:
              - "testDirectory.example.com"
          -
```

```
          Key: "dnsIpAddresses"
          Value:
            Fn::GetAtt:
              - "myDirectory"
              - "DnsIpAddresses"
  IamInstanceProfile:
    Ref: "myInstanceProfile"
  NetworkInterfaces:
    -
      DeviceIndex: "0"
      AssociatePublicIpAddress: "true"
      SubnetId:
        Ref: "mySubnet"
  KeyName:
    Ref: "myKeyName"
```

AWS::SSM::Parameter

The `AWS::SSM::Parameter` resource creates an Amazon EC2 Systems Manager (SSM) parameter in Parameter Store.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::SSM::Parameter",
  "Properties" : {
    "Name" : String,
    "Description" : String,
    "Type" : String,
    "Value" : String
  }
}
```

YAML

```
Type: "AWS::SSM::Parameter"
Properties:
  Name: String
  Description: String
  Type: String
  Value: String
```

Properties

`Name`

The name of the parameter. Names must not be prefixed with `aws` or `ssm`.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Description`

Information about the parameter that you want to add to the system.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Type`

The type of parameter. Valid values include the following: `String` or `StringList`.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

`Value`

The parameter value. Value must not nest another parameter. Do not use `{{}}` in the value.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

Return Value

Ref

When you pass the logical ID of an `AWS::SSM::Parameter` resource to the intrinsic `Ref` function, the function returns the Name of the SSM parameter. For example, `ssm-myparameter-ABCNPH3XCAO6`.

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`Type`

Returns the type of the parameter. Valid values include: `String` or `StringList`.

`Value`

Returns the value of the parameter.

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Examples

SSM Parameter (String) Example

The following example snippet creates an SSM parameter in the Parameter Store.

JSON

```
{
   "Description": "Create SSM Parameter",
   "Resources": {
      "BasicParameter": {
         "Type": "AWS::SSM::Parameter",
         "Properties": {
            "Name": "command",
            "Type": "String",
            "Value": "date",
            "Description": "SSM Parameter for running date command."
         }
      }
   }
}
```

YAML

```
Description: "Create SSM Parameter"
Resources:
  BasicParameter:
    Type: "AWS::SSM::Parameter"
    Properties:
      Name: "command"
      Type: "String"
      Value: "date"
      Description: "SSM Parameter for running date command."
```

SSM Parameter (StringList) Example

The following example creates an SSM parameter with a StringList type.

JSON

```
{
   "Description": "Create SSM Parameter",
   "Resources": {
      "BasicParameter": {
         "Type": "AWS::SSM::Parameter",
         "Properties": {
            "Name": "commands",
            "Type": "StringList",
            "Value": "date,ls",
            "Description": "SSM Parameter of type StringList."
         }
      }
   }
}
```

YAML

```
Description: "Create SSM Parameter"
```

```
Resources:
  BasicParameter:
    Type: "AWS::SSM::Parameter"
    Properties:
      Name: "commands"
      Type: "StringList"
      Value: "date,ls"
      Description: "SSM Parameter of type StringList."
```

AWS::StepFunctions::Activity

Use the `AWS::StepFunctions::Activity` resource to create an AWS Step Functions activity.

For information about creating an activity and creating a state machine with an activity, see Tutorial: An Activity State Machine in the *AWS Step Functions Developer Guide* and `CreateActivity` in the *AWS Step Functions API Reference*.

Syntax

JSON

```
{
   "Type": "AWS::StepFunctions::Activity",
   "Properties": {
      "Name": String
    }
}
```

YAML

```
Type: "AWS::StepFunctions::Activity"
Properties:
  Name: String
```

Properties

`Name`

The name of the activity to create. This name must be unique for your AWS account and region.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When you provide the logical ID of this resource to the `Ref` intrinsic function, `Ref` returns the ARN of the created activity. For example:

```
{ "Ref": "MyActivity" }
```

Returns a value similar to the following:

```
arn:aws:states:us-east-1:111122223333:activity:myActivity
```

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`Name`

Returns the name of the activity. For example:

```
{ "Fn::GetAtt": ["MyActivity", "Name"] }
```

Returns a value similar to the following:

```
myActivity
```

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Example

The following example creates a Step Functions activity.

JSON

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Description" : "An example template for a Step Functions activity.",
   "Resources" : {
      "MyActivity" : {
         "Type" : "AWS::StepFunctions::Activity",
         "Properties" : {
            "Name" : "myActivity"
         }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Description: "A sample template for a Step Functions activity"
Resources:
  MyActivity:
    Type: "AWS::StepFunctions::Activity"
    Properties:
      Name: myActivity
```

AWS::StepFunctions::StateMachine

Use the `AWS::StepFunctions::StateMachine` resource to create an AWS Step Functions state machine.

Note
AWS CloudFormation automatically generates a name for your state machine. If you change the contents of the `DefinitionString` or `RoleArn` properties and update the stack, AWS CloudFormation generates a new state machine name and deletes the old state machine.

For information about creating state machines, see Tutorial: A Lambda State Machine in the *AWS Step Functions Developer Guide* and `CreateStateMachine` in the *AWS Step Functions API Reference*.

Syntax

JSON

```
{
   "Type": "AWS::StepFunctions::StateMachine",
   "Properties": {
      "DefinitionString": String,
      "RoleArn": String
    }
}
```

YAML

```
Type: "AWS::StepFunctions::StateMachine"
Properties:
  DefinitionString: String
  RoleArn: String
```

Properties

`DefinitionString`

The Amazon States Language definition of the state machine. For more information, see Amazon States Language in the *AWS Step Functions Developer Guide*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`RoleArn`

The Amazon Resource Name (ARN) of the IAM role to use for this state machine.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When you provide the logical ID of this resource to the `Ref` intrinsic function, `Ref` returns the ARN of the created state machine. For example:

```
{ "Ref": "MyStateMachine" }
```

Returns a value similar to the following:

```
arn:aws:states:us-east-1:111122223333:stateMachine:MyStateMachine-ABCDEFGHIJ1K
```

For more information about using the `Ref` function, see Ref (p. 1343).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

`Name`

Returns the name of the state machine. For example:

```
{ "Fn::GetAtt": ["MyStateMachine", "Name"] }
```

Returns a value similar to the following:

```
MyStateMachine-ABCDEFGHIJ1K
```

For more information about using `Fn::GetAtt`, see Fn::GetAtt (p. 1324).

Examples

The following examples create a Step Functions state machine.

JSON

Using a Single-Line Property

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Description" : "An example template for a Step Functions state machine.",
   "Resources" : {
      "MyStateMachine" : {
         "Type" : "AWS::StepFunctions::StateMachine",
         "Properties" : {
            "DefinitionString" : "{\"StartAt\": \"HelloWorld\", \"States\":
 {\"HelloWorld\": {\"Type\": \"Task\", \"Resource\": \"arn:aws:lambda:us-
east-1:111122223333:function:HelloFunction\", \"End\": true}}}",
            "RoleArn" : "arn:aws:iam::111122223333:role/service-role/StatesExecutionRole-
us-east-1"
         }
      }
   }
}
```

Using the Fn::Join Intrinsic Function

```
{
    "AWSTemplateFormatVersion" : "2010-09-09",
    "Description" : "An example template for a Step Functions state machine.",
```

```
    "Resources": {
       "MyStateMachine": {
          "Type": "AWS::StepFunctions::StateMachine",
             "Properties": {
                "DefinitionString": {
                   "Fn::Join": [
                      "\n",
                      [
                         "{",
                         "    \"StartAt\": \"HelloWorld\",",
                         "    \"States\" : {",
                         "        \"HelloWorld\" : {",
                         "            \"Type\" : \"Task\", ",
                         "            \"Resource\" : \"arn:aws:lambda:us-
east-1:111122223333:function:HelloFunction\",",
                         "            \"End\" : true",
                         "        }",
                         "    }",
                         "}"
                      ]
                   ]
                },
          "RoleArn" : "arn:aws:iam::111122223333:role/service-role/StatesExecutionRole-us-
east-1"
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: An example template for a Step Functions state machine.
Resources:
  MyStateMachine:
    Type: AWS::StepFunctions::StateMachine
    Properties:
      DefinitionString: |-
        {
          "StartAt": "HelloWorld",
          "States": {
            "HelloWorld": {
              "Type": "Task",
              "Resource": "arn:aws:lambda:us-east-1:111122223333:function:HelloFunction",
              "End": true
            }
          }
        }
      RoleArn: arn:aws:iam::111122223333:role/service-role/StatesExecutionRole-us-east-1
```

AWS::WAF::ByteMatchSet

The `AWS::WAF::ByteMatchSet` resource creates an AWS WAF `ByteMatchSet` that identifies a part of a web request that you want to inspect. For more information, see CreateByteMatchSet in the *AWS WAF API Reference*.

Topics

- Examples (p. 981)

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::WAF::ByteMatchSet",
  "Properties" : {
    "ByteMatchTuples" : [ Byte match tuple, ... ],
    "Name" : String
  }
}
```

YAML

```
Type: "AWS::WAF::ByteMatchSet"
Properties:
  ByteMatchTuples:
    - Byte match tuple
  Name: String
```

Properties

`ByteMatchTuples`

Settings for the `ByteMatchSet`, such as the bytes (typically a string that corresponds with ASCII characters) that you want AWS WAF to search for in web requests.

Required: No

Type: List of AWS WAF ByteMatchSet ByteMatchTuples (p. 1273)

Update requires: No interruption (p. 90)

`Name`

A friendly name or description of the `ByteMatchSet`.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource physical ID, such as `1234a1a-a1b1-12a1-abcd-a123b123456`.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

HTTP Referers

The following example defines a set of HTTP referers to match.

JSON

```
"BadReferers": {
  "Type": "AWS::WAF::ByteMatchSet",
  "Properties": {
    "Name": "ByteMatch for matching bad HTTP referers",
    "ByteMatchTuples": [
      {
        "FieldToMatch" : {
          "Type": "HEADER",
          "Data": "referer"
        },
        "TargetString" : "badrefer1",
        "TextTransformation" : "NONE",
        "PositionalConstraint" : "CONTAINS"
      },
      {
        "FieldToMatch" : {
          "Type": "HEADER",
          "Data": "referer"
        },
        "TargetString" : "badrefer2",
        "TextTransformation" : "NONE",
        "PositionalConstraint" : "CONTAINS"
      }
    ]
  }
}
```

YAML

```
BadReferers:
  Type: "AWS::WAF::ByteMatchSet"
  Properties:
    Name: "ByteMatch for matching bad HTTP referers"
    ByteMatchTuples:
      -
        FieldToMatch:
          Type: "HEADER"
          Data: "referer"
        TargetString: "badrefer1"
        TextTransformation: "NONE"
        PositionalConstraint: "CONTAINS"
      -
        FieldToMatch:
          Type: "HEADER"
          Data: "referer"
        TargetString: "badrefer2"
        TextTransformation: "NONE"
        PositionalConstraint: "CONTAINS"
```

Associate a ByteMatchSet with a Web ACL Rule

The following example associates the `BadReferers` byte match set with a web access control list (ACL) rule.

JSON

```
"BadReferersRule" : {
  "Type": "AWS::WAF::Rule",
  "Properties": {
    "Name": "BadReferersRule",
    "MetricName" : "BadReferersRule",
    "Predicates": [
      {
        "DataId" : {  "Ref" : "BadReferers" },
        "Negated" : false,
        "Type" : "ByteMatch"
      }
    ]
  }
}
```

YAML

```
BadReferersRule:
  Type: "AWS::WAF::Rule"
  Properties:
    Name: "BadReferersRule"
    MetricName: "BadReferersRule"
    Predicates:
      -
        DataId:
          Ref: "BadReferers"
        Negated: false
        Type: "ByteMatch"
```

Create a Web ACL

The following example associates the `BadReferersRule` rule with a web ACL. The web ACL allows all requests except for ones with referers that match the `BadReferersRule` rule.

JSON

```
"MyWebACL": {
  "Type": "AWS::WAF::WebACL",
  "Properties": {
    "Name": "WebACL to block blacklisted IP addresses",
    "DefaultAction": {
      "Type": "ALLOW"
    },
    "MetricName" : "MyWebACL",
    "Rules": [
      {
        "Action" : {
          "Type" : "BLOCK"
        },
        "Priority" : 1,
        "RuleId" : { "Ref" : "BadReferersRule" }
      }
    ]
  }
}
```

YAML

```
MyWebACL:
```

```
  Type: "AWS::WAF::WebACL"
  Properties:
    Name: "WebACL to block blacklisted IP addresses"
    DefaultAction:
      Type: "ALLOW"
    MetricName: "MyWebACL"
    Rules:
      -
        Action:
          Type: "BLOCK"
        Priority: 1
        RuleId:
          Ref: "BadReferersRule"
```

AWS::WAF::IPSet

The `AWS::WAF::IPSet` resource creates an AWS WAF `IPSet` that specifies which web requests to permit or block based on the IP addresses from which the requests originate. For more information, see CreateIPSet in the *AWS WAF API Reference*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::WAF::IPSet",
  "Properties" : {
    "IPSetDescriptors" : [ IPSet descriptor, ... ],
    "Name" : String
  }
}
```

YAML

```
Type: "AWS::WAF::IPSet"
Properties:
  IPSetDescriptors:
    - IPSet descriptor
  Name: String
```

Properties

`IPSetDescriptors`

The IP address type and IP address range (in CIDR notation) from which web requests originate. If you associate the `IPSet` with a web ACL (p. 994) that is associated with a Amazon CloudFront

(CloudFront) distribution, this descriptor is the value of one of the following fields in the CloudFront access logs:

`c-ip`

If the viewer did not use an HTTP proxy or a load balancer to send the request

`x-forwarded-for`

If the viewer did use an HTTP proxy or a load balancer to send the request

Required: No

Type: List of AWS WAF IPSet IPSetDescriptors (p. 1275)

Update requires: No interruption (p. 90)

`Name`

A friendly name or description of the `IPSet`.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource physical ID, such as `1234a1a-a1b1-12a1-abcd-a123b123456`.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

Define IP Addresses

The following example defines a set of IP addresses for a web access control list (ACL) rule.

JSON

```
"MyIPSetBlacklist": {
  "Type": "AWS::WAF::IPSet",
  "Properties": {
    "Name": "IPSet for blacklisted IP adresses",
    "IPSetDescriptors": [
      {
        "Type" : "IPV4",
        "Value" : "192.0.2.44/32"
      },
      {
        "Type" : "IPV4",
        "Value" : "192.0.7.0/24"
      }
    ]
  }
}
```

YAML

```
MyIPSetBlacklist:
  Type: "AWS::WAF::IPSet"
  Properties:
    Name: "IPSet for blacklisted IP adresses"
    IPSetDescriptors:
      -
        Type: "IPV4"
        Value: "192.0.2.44/32"
      -
        Type: "IPV4"
        Value: "192.0.7.0/24"
```

Associate an IPSet with a Web ACL Rule

The following example associates the `MyIPSetBlacklist` IP Set with a web ACL rule.

JSON

```
"MyIPSetRule" : {
  "Type": "AWS::WAF::Rule",
  "Properties": {
    "Name": "MyIPSetRule",
    "MetricName" : "MyIPSetRule",
    "Predicates": [
      {
        "DataId" : {  "Ref" : "MyIPSetBlacklist" },
        "Negated" : false,
        "Type" : "IPMatch"
      }
    ]
  }
}
```

YAML

```
MyIPSetRule:
  Type: "AWS::WAF::Rule"
  Properties:
    Name: "MyIPSetRule"
    MetricName: "MyIPSetRule"
    Predicates:
      -
        DataId:
          Ref: "MyIPSetBlacklist"
        Negated: false
        Type: "IPMatch"
```

Create a Web ACL

The following example associates the `MyIPSetRule` rule with a web ACL. The web ACL allows requests that originate from all IP addresses except for addresses that are defined in the `MyIPSetRule`.

JSON

```
"MyWebACL": {
  "Type": "AWS::WAF::WebACL",
  "Properties": {
```

```
    "Name": "WebACL to block blacklisted IP addresses",
    "DefaultAction": {
      "Type": "ALLOW"
    },
    "MetricName" : "MyWebACL",
    "Rules": [
      {
        "Action" : {
          "Type" : "BLOCK"
        },
        "Priority" : 1,
        "RuleId" : { "Ref" : "MyIPSetRule" }
      }
    ]
  }
}
```

YAML

```
MyWebACL:
  Type: "AWS::WAF::WebACL"
  Properties:
    Name: "WebACL to block blacklisted IP addresses"
    DefaultAction:
      Type: "ALLOW"
    MetricName: "MyWebACL"
    Rules:
      -
        Action:
          Type: "BLOCK"
        Priority: 1
        RuleId:
          Ref: "MyIPSetRule"
```

AWS::WAF::Rule

The `AWS::WAF::Rule` resource creates an AWS WAF rule that specifies a combination of `IPSet`, `ByteMatchSet`, and `SqlInjectionMatchSet` objects that identify the web requests to allow, block, or count. To implement rules, you must associate them with a web ACL (p. 994).

For more information, see CreateRule in the *AWS WAF API Reference*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::WAF::Rule",
  "Properties" : {
```

```
    "MetricName" : String,
    "Name" : String,
    "Predicates" : [ Predicate, ... ]
  }
}
```

YAML

```
Type: "AWS::WAF::Rule"
Properties:
  MetricName: String
  Name: String
  Predicates:
    - Predicate
```

Properties

`MetricName`

A friendly name or description for the metrics of the rule. For valid values, see the `MetricName` parameter for the `CreateRule` action in the *AWS WAF API Reference*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Name`

A friendly name or description of the rule.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Predicates`

The `ByteMatchSet`, `IPSet`, `SizeConstraintSet`, `SqlInjectionMatchSet`, or `XssMatchSet` objects to include in a rule. If you add more than one predicate to a rule, a request must match all conditions in order to be allowed or blocked.

Required: No

Type: List of AWS WAF Rule Predicates (p. 1276)

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource physical ID, such as `1234a1a-a1b1-12a1-abcd-a123b123456`.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

Associate an IPSet with a Web ACL Rule

The following example associates the `MyIPSetBlacklist` `IPSet` object with a web ACL rule.

JSON

```
"MyIPSetRule" : {
  "Type": "AWS::WAF::Rule",
  "Properties": {
    "Name": "MyIPSetRule",
    "MetricName" : "MyIPSetRule",
    "Predicates": [
      {
        "DataId" : {  "Ref" : "MyIPSetBlacklist" },
        "Negated" : false,
        "Type" : "IPMatch"
      }
    ]
  }
}
```

YAML

```
MyIPSetRule:
  Type: "AWS::WAF::Rule"
  Properties:
    Name: "MyIPSetRule"
    MetricName: "MyIPSetRule"
    Predicates:
      -
        DataId:
          Ref: "MyIPSetBlacklist"
        Negated: false
        Type: "IPMatch"
```

AWS::WAF::SizeConstraintSet

The `AWS::WAF::SizeConstraintSet` resource specifies a size constraint that AWS WAF uses to check the size of a web request and which parts of the request to check. For more information, see CreateSizeConstraintSet in the *AWS WAF API Reference*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
```

```
  "Type" : "AWS::WAF::SizeConstraintSet",
  "Properties" : {
    "Name" : String,
    "SizeConstraints" : [ SizeConstraint, ... ]
  }
}
```

YAML

```
Type: "AWS::WAF::SizeConstraintSet"
Properties:
  Name: String
  SizeConstraints:
    - SizeConstraint
```

Properties

`Name`

A friendly name or description for the `SizeConstraintSet`.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`SizeConstraints`

The size constraint and the part of the web request to check.

Required: Yes

Type: List of AWS WAF SizeConstraintSet SizeConstraint (p. 1277)

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource physical ID, such as `1234a1a-a1b1-12a1-abcd-a123b123456`.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

The following examples show you how to define a size constraint, add it to a rule, and add the rule to a web access control list (ACL).

Define a Size Constraint

The following example checks that the body of an HTTP request equals `4096` bytes.

JSON

```
"MySizeConstraint": {
```

```
  "Type": "AWS::WAF::SizeConstraintSet",
  "Properties": {
    "Name": "SizeConstraints",
    "SizeConstraints": [
      {
        "ComparisonOperator": "EQ",
        "FieldToMatch": {
          "Type": "BODY"
        },
        "Size": "4096",
        "TextTransformation": "NONE"
      }
    ]
  }
}
```

YAML

```
  MySizeConstraint:
    Type: "AWS::WAF::SizeConstraintSet"
    Properties:
      Name: "SizeConstraints"
      SizeConstraints:
        -
          ComparisonOperator: "EQ"
          FieldToMatch:
            Type: "BODY"
          Size: "4096"
          TextTransformation: "NONE"
```

Associate a `SizeConstraintSet` with a Web ACL Rule

The following example associates the `MySizeConstraint` object with a web ACL rule.

JSON

```
"SizeConstraintRule" : {
  "Type": "AWS::WAF::Rule",
  "Properties": {
    "Name": "SizeConstraintRule",
    "MetricName" : "SizeConstraintRule",
    "Predicates": [
      {
        "DataId" : {  "Ref" : "MySizeConstraint" },
        "Negated" : false,
        "Type" : "SizeConstraint"
      }
    ]
  }
}
```

YAML

```
SizeConstraintRule:
  Type: "AWS::WAF::Rule"
  Properties:
    Name: "SizeConstraintRule"
    MetricName: "SizeConstraintRule"
    Predicates:
      -
        DataId:
```

```
          Ref: "MySizeConstraint"
        Negated: false
        Type: "SizeConstraint"
```

Create a Web ACL

The following example associates the `SizeConstraintRule` rule with a web ACL. The web ACL blocks all requests except for requests with a body size equal to `4096` bytes.

JSON

```
"MyWebACL": {
  "Type": "AWS::WAF::WebACL",
  "Properties": {
    "Name": "Web ACL to allow requests with a specific size",
    "DefaultAction": {
      "Type": "BLOCK"
    },
    "MetricName" : "SizeConstraintWebACL",
    "Rules": [
      {
        "Action" : {
          "Type" : "ALLOW"
        },
        "Priority" : 1,
        "RuleId" : { "Ref" : "SizeConstraintRule" }
      }
    ]
  }
}
```

YAML

```
MyWebACL:
  Type: "AWS::WAF::WebACL"
  Properties:
    Name: "Web ACL to allow requests with a specific size"
    DefaultAction:
      Type: "BLOCK"
    MetricName: "SizeConstraintWebACL"
    Rules:
      -
        Action:
          Type: "ALLOW"
        Priority: 1
        RuleId:
          Ref: "SizeConstraintRule"
```

AWS::WAF::SqlInjectionMatchSet

The `AWS::WAF::SqlInjectionMatchSet` resource creates an AWS WAF `SqlInjectionMatchSet`, which you use to allow, block, or count requests that contain malicious SQL code in a specific part of web requests. For more information, see CreateSqlInjectionMatchSet in the *AWS WAF API Reference*.

Topics

• Examples (p. 993)

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::WAF::SqlInjectionMatchSet",
  "Properties" : {
    "Name" : String,
    "SqlInjectionMatchTuples" : [ SqlInjectionMatchTuple, ... ]
  }
}
```

YAML

```
Type: "AWS::WAF::SqlInjectionMatchSet"
Properties:
  Name: String
  SqlInjectionMatchTuples:
    - SqlInjectionMatchTuple
```

Properties

`Name`

A friendly name or description of the `SqlInjectionMatchSet`.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`SqlInjectionMatchTuples`

The parts of web requests that you want AWS WAF to inspect for malicious SQL code and, if you want AWS WAF to inspect a header, the name of the header.

Required: No

Type: List of AWS WAF SqlInjectionMatchSet SqlInjectionMatchTuples (p. 1279)

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource physical ID, such as `1234a1a-a1b1-12a1-abcd-a123b123456`.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

Find SQL Injections

The following example looks for snippets of SQL code in the query string of an HTTP request.

JSON

```
"SqlInjDetection": {
  "Type": "AWS::WAF::SqlInjectionMatchSet",
  "Properties": {
    "Name": "Find SQL injections in the query string",
    "SqlInjectionMatchTuples": [
      {
        "FieldToMatch" : {
          "Type": "QUERY_STRING"
        },
        "TextTransformation" : "URL_DECODE"
      }
    ]
  }
}
```

YAML

```
SqlInjDetection:
  Type: "AWS::WAF::SqlInjectionMatchSet"
  Properties:
    Name: "Find SQL injections in the query string"
    SqlInjectionMatchTuples:
      -
        FieldToMatch:
          Type: "QUERY_STRING"
        TextTransformation: "URL_DECODE"
```

Associate a SQL Injection Match Set with a Web ACL Rule

The following example associates the `SqlInjDetection` match set with a web access control list (ACL) rule.

JSON

```
"SqlInjRule" : {
  "Type": "AWS::WAF::Rule",
  "Properties": {
    "Name": "SqlInjRule",
    "MetricName" : "SqlInjRule",
    "Predicates": [
      {
        "DataId" : {  "Ref" : "SqlInjDetection" },
        "Negated" : false,
        "Type" : "SqlInjectionMatch"
      }
    ]
  }
}
```

YAML

```
SqlInjRule:
```

```
  Type: "AWS::WAF::Rule"
  Properties:
    Name: "SqlInjRule"
    MetricName: "SqlInjRule"
    Predicates:
      -
        DataId:
          Ref: "SqlInjDetection"
        Negated: false
        Type: "SqlInjectionMatch"
```

Create a Web ACL

The following example associates the `SqlInjRule` rule with a web ACL. The web ACL allows all requests except for ones with SQL code in the query string of a request.

JSON

```
"MyWebACL": {
  "Type": "AWS::WAF::WebACL",
  "Properties": {
    "Name": "Web ACL to block SQL injection in the query string",
    "DefaultAction": {
      "Type": "ALLOW"
    },
    "MetricName" : "SqlInjWebACL",
    "Rules": [
      {
        "Action" : {
          "Type" : "BLOCK"
        },
        "Priority" : 1,
        "RuleId" : { "Ref" : "SqlInjRule" }
      }
    ]
  }
}
```

YAML

```
MyWebACL:
  Type: "AWS::WAF::WebACL"
  Properties:
    Name: "Web ACL to block SQL injection in the query string"
    DefaultAction:
      Type: "ALLOW"
    MetricName: "SqlInjWebACL"
    Rules:
      -
        Action:
          Type: "BLOCK"
        Priority: 1
        RuleId:
          Ref: "SqlInjRule"
```

AWS::WAF::WebACL

The `AWS::WAF::WebACL` resource creates an AWS WAF web access control group (ACL) containing the rules that identify the Amazon CloudFront (CloudFront) web requests that you want to allow, block, or count. For more information, see CreateWebACL in the *AWS WAF API Reference*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::WAF::WebACL",
  "Properties" : {
    "DefaultAction" : Action,
    "MetricName" : String,
    "Name" : String,
    "Rules" : [ Rule, ... ]
  }
}
```

YAML

```
Type: "AWS::WAF::WebACL"
Properties:
  DefaultAction:
    Action
  MetricName: String
  Name: String
  Rules:
    - Rule
```

Properties

`DefaultAction`

The action that you want AWS WAF to take when a request doesn't match the criteria in any of the rules that are associated with the web ACL.

Required: Yes

Type: AWS WAF WebACL Action (p. 1282)

Update requires: No interruption (p. 90)

`MetricName`

A friendly name or description for the Amazon CloudWatch metric of this web ACL. For valid values, see the `MetricName` parameter of the CreateWebACL action in the *AWS WAF API Reference*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Name`

A friendly name or description of the web ACL.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`Rules`

The rules to associate with the web ACL and the settings for each rule.

Required: No

Type: List of AWS WAF WebACL Rules (p. 1283)

Update requires: No interruption (p. 90)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name, such as `1234a1a-a1b1-12a1-abcd-a123b123456`.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

Create a Web ACL

The following example defines a web ACL that allows, by default, any web request. However, if the request matches any rule, AWS WAF blocks the request. AWS WAF evaluates each rule in priority order, starting with the lowest value.

JSON

```
"MyWebACL": {
  "Type": "AWS::WAF::WebACL",
  "Properties": {
    "Name": "WebACL to with three rules",
    "DefaultAction": {
      "Type": "ALLOW"
    },
    "MetricName" : "MyWebACL",
    "Rules": [
      {
        "Action" : {
          "Type" : "BLOCK"
        },
        "Priority" : 1,
        "RuleId" : { "Ref" : "MyRule" }
      },
      {
        "Action" : {
          "Type" : "BLOCK"
        },
        "Priority" : 2,
        "RuleId" : { "Ref" : "BadReferersRule" }
```

```
      },
      {
        "Action" : {
          "Type" : "BLOCK"
        },
        "Priority" : 3,
        "RuleId" : { "Ref" : "SqlInjRule" }
      }
    ]
  }
}
```

YAML

```
MyWebACL:
  Type: "AWS::WAF::WebACL"
  Properties:
    Name: "WebACL to with three rules"
    DefaultAction:
      Type: "ALLOW"
    MetricName: "MyWebACL"
    Rules:
      -
        Action:
          Type: "BLOCK"
        Priority: 1
        RuleId:
          Ref: "MyRule"
      -
        Action:
          Type: "BLOCK"
        Priority: 2
        RuleId:
          Ref: "BadReferersRule"
      -
        Action:
          Type: "BLOCK"
        Priority: 3
        RuleId:
          Ref: "SqlInjRule"
```

Associate a Web ACL with a CloudFront Distribution

The follow example associates the `MyWebACL` web ACL with a CloudFront distribution. The web ACL restricts which requests can access content served by CloudFront.

JSON

```
"myDistribution": {
  "Type": "AWS::CloudFront::Distribution",
  "Properties": {
    "DistributionConfig": {
      "WebACLId": { "Ref" : "MyWebACL" },
      "Origins": [
        {
          "DomainName": "test.example.com",
          "Id": "myCustomOrigin",
          "CustomOriginConfig": {
            "HTTPPort": "80",
            "HTTPSPort": "443",
            "OriginProtocolPolicy": "http-only"
          }
```

```
        }
      ],
      "Enabled": "true",
      "Comment": "TestDistribution",
      "DefaultRootObject": "index.html",
      "DefaultCacheBehavior": {
        "TargetOriginId": "myCustomOrigin",
        "SmoothStreaming" : "false",
        "ForwardedValues": {
          "QueryString": "false",
          "Cookies" : { "Forward" : "all" }
        },
        "ViewerProtocolPolicy": "allow-all"
      },
      "CustomErrorResponses" : [
        {
          "ErrorCode" : "404",
          "ResponsePagePath" : "/error-pages/404.html",
          "ResponseCode" : "200",
          "ErrorCachingMinTTL" : "30"
        }
      ],
      "PriceClass" : "PriceClass_200",
      "Restrictions" : {
        "GeoRestriction" : {
          "RestrictionType" : "whitelist",
          "Locations" : [ "AQ", "CV" ]
        }
      },
      "ViewerCertificate" : { "CloudFrontDefaultCertificate" : "true" }
    }
  }
}
```

YAML

```
myDistribution:
  Type: "AWS::CloudFront::Distribution"
  Properties:
    DistributionConfig:
      WebACLId:
        Ref: "MyWebACL"
      Origins:
        -
          DomainName: "test.example.com"
          Id: "myCustomOrigin"
          CustomOriginConfig:
            HTTPPort: "80"
            HTTPSPort: "443"
            OriginProtocolPolicy: "http-only"
      Enabled: "true"
      Comment: "TestDistribution"
      DefaultRootObject: "index.html"
      DefaultCacheBehavior:
        TargetOriginId: "myCustomOrigin"
        SmoothStreaming: "false"
        ForwardedValues:
          QueryString: "false"
          Cookies:
            Forward: "all"
        ViewerProtocolPolicy: "allow-all"
      CustomErrorResponses:
        -
          ErrorCode: "404"
```

```
          ResponsePagePath: "/error-pages/404.html"
          ResponseCode: "200"
          ErrorCachingMinTTL: "30"
      PriceClass: "PriceClass_200"
      Restrictions:
        GeoRestriction:
          RestrictionType: "whitelist"
          Locations:
            - "AQ"
            - "CV"
      ViewerCertificate:
        CloudFrontDefaultCertificate: "true"
```

AWS::WAF::XssMatchSet

The `AWS::WAF::XssMatchSet` resource specifies the parts of web requests that you want AWS WAF to inspect for cross-site scripting attacks and the name of the header to inspect. For more information, see XssMatchSet in the *AWS WAF API Reference*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::WAF::XssMatchSet",
  "Properties" : {
    "Name" : String,
    "XssMatchTuples" : [ XssMatchTuple, ... ]
  }
}
```

YAML

```
Type: "AWS::WAF::XssMatchSet"
Properties:
  Name: String
  XssMatchTuples:
    - XssMatchTuple
```

Properties

`Name`

A friendly name or description for the `XssMatchSet`.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`XssMatchTuples`

The parts of web requests that you want to inspect for cross-site scripting attacks.

Required: No

Type: List of AWS WAF XssMatchSet XssMatchTuple (p. 1280)

Update requires: No interruption (p. 90)

Return Value

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource physical ID, such as `1234a1a-a1b1-12a1-abcd-a123b123456`.

For more information about using the `Ref` function, see Ref (p. 1343).

Examples

Define Which Part of a Request to Check for Cross-site Scripting

The following example looks for cross-site scripting in the URI or query string of an HTTP request.

JSON

```
"DetectXSS": {
  "Type": "AWS::WAF::XssMatchSet",
  "Properties": {
    "Name": "XssMatchSet",
    "XssMatchTuples": [
      {
        "FieldToMatch": {
          "Type": "URI"
        },
        "TextTransformation": "NONE"
      },
      {
        "FieldToMatch": {
          "Type": "QUERY_STRING"
        },
        "TextTransformation": "NONE"
      }
    ]
  }
}
```

YAML

```
DetectXSS:
  Type: "AWS::WAF::XssMatchSet"
  Properties:
    Name: "XssMatchSet"
    XssMatchTuples:
      -
        FieldToMatch:
          Type: "URI"
        TextTransformation: "NONE"
```

```
      -
        FieldToMatch:
          Type: "QUERY_STRING"
        TextTransformation: "NONE"
```

Associate an XssMatchSet with a Web ACL Rule

The following example associates the `DetectXSS` match set with a web access control list (ACL) rule.

JSON

```
"XSSRule" : {
  "Type": "AWS::WAF::Rule",
  "Properties": {
    "Name": "XSSRule",
    "MetricName" : "XSSRule",
    "Predicates": [
      {
        "DataId" : {  "Ref" : "DetectXSS" },
        "Negated" : false,
        "Type" : "XssMatch"
      }
    ]
  }
}
```

YAML

```
XSSRule:
  Type: "AWS::WAF::Rule"
  Properties:
    Name: "XSSRule"
    MetricName: "XSSRule"
    Predicates:
      -
        DataId:
          Ref: "DetectXSS"
        Negated: false
        Type: "XssMatch"
```

Create a Web ACL

The following example associates the `XSSRule` rule with a web ACL. The web ACL allows all requests except for ones that contain cross-site scripting in the URI or query string of an HTTP request.

JSON

```
"MyWebACL": {
  "Type": "AWS::WAF::WebACL",
  "Properties": {
    "Name": "Web ACL to block cross-site scripting",
    "DefaultAction": {
      "Type": "ALLOW"
    },
    "MetricName" : "DetectXSSWebACL",
    "Rules": [
      {
        "Action" : {
          "Type" : "BLOCK"
        },
        "Priority" : 1,
```

```
        "RuleId" : { "Ref" : "XSSRule" }
      }
    ]
  }
}
```

YAML

```
MyWebACL:
  Type: "AWS::WAF::WebACL"
  Properties:
    Name: "Web ACL to block cross-site scripting"
    DefaultAction:
      Type: "ALLOW"
    MetricName: "DetectXSSWebACL"
    Rules:
      -
        Action:
          Type: "BLOCK"
        Priority: 1
        RuleId:
          Ref: "XSSRule"
```

AWS::WorkSpaces::Workspace

The `AWS::WorkSpaces::Workspace` resource creates an Amazon WorkSpaces workspace, which is a cloud-based desktop experience for end users. For more information, see the *Amazon WorkSpaces Administration Guide*.

Topics

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Type" : "AWS::WorkSpaces::Workspace",
  "Properties" : {
    "BundleId" : String,
    "DirectoryId" : String,
    "UserName" : String,
    "RootVolumeEncryptionEnabled" : Boolean,
    "UserVolumeEncryptionEnabled" : Boolean,
    "VolumeEncryptionKey" : String
  }
}
```

YAML

```
Type: "AWS::WorkSpaces::Workspace"
```

```
Properties:
  BundleId: String
  DirectoryId: String
  UserName: String
  RootVolumeEncryptionEnabled: Boolean
  UserVolumeEncryptionEnabled: Boolean
  VolumeEncryptionKey: String
```

Properties

`BundleId`

The identifier of the bundle from which you want to create the workspace. A bundle specifies the details of the workspace, such as the installed applications and the size of CPU, memory, and storage. Use the DescribeWorkspaceBundles action to list the bundles that AWS offers.

Required: Yes

Type: String

Update requires: Updates are not supported.. To update this property, you must also update another property that triggers a replacement, such as the `UserName` property.

`DirectoryId`

The identifier of the AWS Directory Service directory in which you want to create the workspace. The directory must already be registered with Amazon WorkSpaces. Use the DescribeWorkspaceDirectories action to list the directories that are available.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`UserName`

The name of the user to which the workspace is assigned. This user name must exist in the specified AWS Directory Service directory.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

`RootVolumeEncryptionEnabled`

Indicates whether Amazon WorkSpaces encrypts data stored on the root volume (`C:` drive).

Required: No

Type: Boolean

Update requires: Updates are not supported.. To update this property, you must also update another property that triggers a replacement, such as the `UserName` property.

`UserVolumeEncryptionEnabled`

Indicates whether Amazon WorkSpaces encrypts data stored on the user volume (`D:` drive).

Required: No

Type: Boolean

Update requires: Updates are not supported.. To update this property, you must also update another property that triggers a replacement, such as the `UserName` property.

`VolumeEncryptionKey`

The AWS Key Management Service (AWS KMS) key ID that Amazon WorkSpaces uses to encrypt data stored on your workspace.

Required: No

Type: String

Update requires: Updates are not supported.. To update this property, you must also update another property that triggers a replacement, such as the `UserName` property.

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, `Ref` returns the resource name.

For more information about using the `Ref` function, see Ref (p. 1343).

Example

The following example creates a workspace for user `test`. The bundle and directory IDs are specified as parameters in the same template.

JSON

```
"workspace1" : {
  "Type" : "AWS::WorkSpaces::Workspace",
  "Properties" : {
    "BundleId" : {"Ref" : "BundleId"},
    "DirectoryId" : {"Ref" : "DirectoryId"},
    "UserName" : "test"
  }
}
```

YAML

```
workspace1:
  Type: "AWS::WorkSpaces::Workspace"
  Properties:
    BundleId:
      Ref: "BundleId"
    DirectoryId:
      Ref: "DirectoryId"
    UserName: "test"
```

Resource Property Types Reference

This section details the resource-specific properties for the resources supported by AWS CloudFormation.

Topics

Amazon API Gateway ApiKey StageKey

`StageKey` is a property of the AWS::ApiGateway::ApiKey (p. 397) resource that specifies the Amazon API Gateway (API Gateway) stage to associate with the API key. This association allows only clients with the key to make requests to methods in that stage.

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "RestApiId" : String,
  "StageName" : String
}
```

YAML

```
RestApiId: String
StageName: String
```

Properties

`RestApiId`

The ID of a `RestApi` resource that includes the stage with which you want to associate the API key.

Required: No

Type: String

`StageName`

The name of the stage with which to associate the API key. The stage must be included in the `RestApi` resource that you specified in the `RestApiId` property.

Required: No

Type: String

Amazon API Gateway Deployment StageDescription

`StageDescription` is a property of the AWS::ApiGateway::Deployment (p. 405) resource that configures an Amazon API Gateway (API Gateway) deployment stage.

Syntax

JSON

```
{
  "CacheClusterEnabled" : Boolean,
  "CacheClusterSize" : String,
  "CacheDataEncrypted" : Boolean,
  "CacheTtlInSeconds" : Integer,
  "CachingEnabled" : Boolean,
  "ClientCertificateId" : String,
  "DataTraceEnabled" : Boolean,
  "Description" : String,
  "LoggingLevel" : String,
  "MethodSettings" : [ MethodSetting (p. 1014) ],
  "MetricsEnabled" : Boolean,
  "StageName" : String,
```

```
  "ThrottlingBurstLimit" : Integer,
  "ThrottlingRateLimit" : Number,
  "Variables" : { String:String, ... }
}
```

YAML

```
CacheClusterEnabled: Boolean
CacheClusterSize: String
CacheDataEncrypted: Boolean
CacheTtlInSeconds: Integer
CachingEnabled: Boolean
ClientCertificateId: String
DataTraceEnabled: Boolean
Description: String
LoggingLevel: String
MethodSettings:
  - MethodSetting (p. 1014)
MetricsEnabled: Boolean
StageName: String
ThrottlingBurstLimit: Integer
ThrottlingRateLimit: Number
Variables:
  String: String
```

Properties

`CacheClusterEnabled`

Indicates whether cache clustering is enabled for the stage.

Required: No

Type: Boolean

`CacheClusterSize`

The size of the stage's cache cluster.

Required: No

Type: String

`CacheDataEncrypted`

Indicates whether the cached responses are encrypted.

Required: No

Type: Boolean

`CacheTtlInSeconds`

The time-to-live (TTL) period, in seconds, that specifies how long API Gateway caches responses.

Required: No

Type: Integer

`CachingEnabled`

Indicates whether responses are cached and returned for requests. You must enable a cache cluster on the stage to cache responses. For more information, see Enable API Gateway Caching in a Stage to Enhance API Performance in the *API Gateway Developer Guide*.

Required: No

Type: Boolean

`ClientCertificateId`

The identifier of the client certificate that API Gateway uses to call your integration endpoints in the stage.

Required: No

Type: String

`DataTraceEnabled`

Indicates whether data trace logging is enabled for methods in the stage. API Gateway pushes these logs to Amazon CloudWatch Logs.

Required: No

Type: Boolean

`Description`

A description of the purpose of the stage.

Required: No

Type: String

`LoggingLevel`

The logging level for this method. For valid values, see the `loggingLevel` property of the Stage resource in the *Amazon API Gateway API Reference*.

Required: No

Type: String

`MethodSettings`

Configures settings for all of the stage's methods.

Required: No

Type: Amazon API Gateway Deployment StageDescription MethodSetting (p. 1014)

`MetricsEnabled`

Indicates whether Amazon CloudWatch metrics are enabled for methods in the stage.

Required: No

Type: Boolean

`StageName`

The name of the stage, which API Gateway uses as the first path segment in the invoke Uniform Resource Identifier (URI).

Required: No

Type: String

`ThrottlingBurstLimit`

The number of burst requests per second that API Gateway permits across all APIs, stages, and methods in your AWS account. For more information, see Manage API Request Throttling in the *API Gateway Developer Guide*.

Required: No

Type: Integer

`ThrottlingRateLimit`

The number of steady-state requests per second that API Gateway permits across all APIs, stages, and methods in your AWS account. For more information, see Manage API Request Throttling in the *API Gateway Developer Guide*.

Required: No

Type: Number

`Variables`

A map that defines the stage variables. Variable names must consist of alphanumeric characters, and the values must match the following regular expression: `[A-Za-z0-9-._~:/?#&=,]+`.

Required: No

Type: Mapping of key-value pairs

Amazon API Gateway Deployment StageDescription MethodSetting

`MethodSetting` is a property of the Amazon API Gateway Deployment StageDescription (p. 1011) property that configures settings for all methods in an Amazon API Gateway (API Gateway) stage.

Syntax

JSON

```
{
  "CacheDataEncrypted" : Boolean,
  "CacheTtlInSeconds" : Integer,
  "CachingEnabled" : Boolean,
  "DataTraceEnabled" : Boolean,
  "HttpMethod" : String,
  "LoggingLevel" : String,
  "MetricsEnabled" : Boolean,
  "ResourcePath" : String,
  "ThrottlingBurstLimit" : Integer,
  "ThrottlingRateLimit" : Number
}
```

YAML

```
CacheDataEncrypted: Boolean
CacheTtlInSeconds: Integer
CachingEnabled: Boolean
DataTraceEnabled: Boolean
HttpMethod: String
LoggingLevel: String
MetricsEnabled: Boolean
ResourcePath: String
ThrottlingBurstLimit: Integer
ThrottlingRateLimit: Number
```

Properties

`CacheDataEncrypted`

Indicates whether the cached responses are encrypted.

Required: No

Type: Boolean

`CacheTtlInSeconds`

The time-to-live (TTL) period, in seconds, that specifies how long API Gateway caches responses.

Required: No

Type: Integer

`CachingEnabled`

Indicates whether responses are cached and returned for requests. You must enable a cache cluster on the stage to cache responses. For more information, see Enable API Gateway Caching in a Stage to Enhance API Performance in the *API Gateway Developer Guide*.

Required: No

Type: Boolean

`DataTraceEnabled`

Indicates whether data trace logging is enabled for methods in the stage. API Gateway pushes these logs to Amazon CloudWatch Logs.

Required: No

Type: Boolean

`HttpMethod`

The HTTP method.

Required: No

Type: String

`LoggingLevel`

The logging level for this method. For valid values, see the `loggingLevel` property of the Stage resource in the *Amazon API Gateway API Reference*.

Required: No

Type: String

`MetricsEnabled`

Indicates whether Amazon CloudWatch metrics are enabled for methods in the stage.

Required: No

Type: Boolean

`ResourcePath`

The resource path for this method. Forward slashes (`/`) are encoded as `~1` and the initial slash must include a forward slash. For example, the path value `/resource/subresource` must be encoded as `/~1resource~1subresource`. To specify the root path, use only a slash (`/`).

Required: No

Type: String

`ThrottlingBurstLimit`

The number of burst requests per second that API Gateway permits across all APIs, stages, and methods in your AWS account. For more information, see Manage API Request Throttling in the *API Gateway Developer Guide*.

Required: No

Type: Integer

`ThrottlingRateLimit`

The number of steady-state requests per second that API Gateway permits across all APIs, stages, and methods in your AWS account. For more information, see Manage API Request Throttling in the *API Gateway Developer Guide*.

Required: No

Type: Number

Amazon API Gateway Method Integration

`Integration` is a property of the AWS::ApiGateway::Method (p. 408) resource that specifies information about the target back end that an Amazon API Gateway (API Gateway) method calls.

Syntax

JSON

```
{
  "CacheKeyParameters" : [ String, ... ],
  "CacheNamespace" : String,
  "Credentials" : String,
  "IntegrationHttpMethod" : String,
  "IntegrationResponses" : [ IntegrationResponse (p. 1019), ... ],
  "PassthroughBehavior" : String,
  "RequestParameters" : { String:String, ... },
  "RequestTemplates" : { String:String, ... },
  "Type" : String,
  "Uri" : String
}
```

YAML

```
CacheKeyParameters:
  - String
CacheNamespace: String
Credentials: String
IntegrationHttpMethod: String
IntegrationResponses:
  IntegrationResponse (p. 1019)
PassthroughBehavior: String
RequestParameters:
  String: String
RequestTemplates:
```

```
  String: String
Type: String
Uri: String
```

Properties

`CacheKeyParameters`

A list of request parameters whose values API Gateway will cache.

Required: No

Type: List of strings

`CacheNamespace`

An API-specific tag group of related cached parameters.

Required: No

Type: String

`Credentials`

The credentials required for the integration. To specify an AWS Identity and Access Management (IAM) role that API Gateway assumes, specify the role's Amazon Resource Name (ARN). To require that the caller's identity be passed through from the request, specify `arn:aws:iam::*:user/*`.

To use resource-based permissions on the AWS Lambda (Lambda) function, don't specify this property. Use the AWS::Lambda::Permission (p. 829) resource to permit API Gateway to call the function. For more information, see Allow Amazon API Gateway to Invoke a Lambda Function in the *AWS Lambda Developer Guide*.

Required: No

Type: String

`IntegrationHttpMethod`

The integration's HTTP method type.

Required: Conditional. For the `Type` property, if you specify `MOCK`, this property is optional. For all other types, you must specify this property.

Type: String

`IntegrationResponses`

The response that API Gateway provides after a method's back end completes processing a request. API Gateway intercepts the back end's response so that you can control how API Gateway surfaces back-end responses. For example, you can map the back-end status codes to codes that you define.

Required: No

Type: List of Amazon API Gateway Method Integration IntegrationResponse (p. 1019)

`PassthroughBehavior`

Indicates when API Gateway passes requests to the targeted back end. This behavior depends on the request's `Content-Type` header and whether you defined a mapping template for it.

For more information and valid values, see the `passthroughBehavior` field in the *API Gateway API Reference*.

Required: No

Type: String

`RequestParameters`

The request parameters that API Gateway sends with the back-end request. Specify request parameters as key-value pairs (string-to-string maps), with a destination as the key and a source as the value.

Specify the destination using the following pattern `integration.request.location.name`, where `location` is `querystring`, `path`, or `header`, and `name` is a valid, unique parameter name.

The source must be an existing method request parameter or a static value. Static values must be enclosed in single quotation marks and pre-encoded based on their destination in the request.

Required: No

Type: Mapping of key-value pairs

`RequestTemplates`

A map of Apache Velocity templates that are applied on the request payload. The template that API Gateway uses is based on the value of the Content-Type header sent by the client. The content type value is the key, and the template is the value (specified as a string), such as the following snippet:

```
"application/json": "{\n    \"statusCode\": \"200\"\n}"
```

For more information about templates, see API Gateway API Request and Response Payload-Mapping Template Reference in the *API Gateway Developer Guide*.

Required: No

Type: Mapping of key-value pairs

`Type`

The type of back end your method is running, such as `HTTP` or `MOCK`. For all of the valid values, see the `type` property for the `Integration` resource in the *Amazon API Gateway REST API Reference*.

Required: Yes

Type: String

`Uri`

The integration's Uniform Resource Identifier (URI).

If you specify `HTTP` for the `Type` property, specify the API endpoint URL.

If you specify `MOCK` for the `Type` property, don't specify this property.

If you specify `AWS` for the `Type` property, specify an AWS service that follows the form: `arn:aws:apigateway:region:subdomain.service|service:path|action/service_api`. For example, a Lambda function URI follows the form: `arn:aws:apigateway:region:lambda:path/path`. The path is usually in the form `/2015-03-31/functions/LambdaFunctionARN/invocations`. For more information, see the `uri` property of the Integration resource in the *Amazon API Gateway REST API Reference*.

Required: Conditional. If you specified `HTTP` or `AWS` for the `Type` property, you must specify this property.

Type: String

Amazon API Gateway Method Integration IntegrationResponse

`IntegrationResponse` is a property of the Amazon API Gateway Method Integration IntegrationResponse (p. 1019) property that specifies the response that Amazon API Gateway (API Gateway) sends after a method's back end finishes processing a request.

Syntax

JSON

```
{
  "ResponseParameters" : { String:String, ... },
  "ResponseTemplates" : { String:String, ... },
  "SelectionPattern" : String,
  "StatusCode" : String
}
```

YAML

```
ResponseParameters:
  String: String
ResponseTemplates:
  String: String
SelectionPattern: String
StatusCode: String
```

Properties

`ResponseParameters`

The response parameters from the back-end response that API Gateway sends to the method response. Specify response parameters as key-value pairs (string-to-string mappings (p. 140)).

Use the destination as the key and the source as the value:

- The destination must be an existing response parameter in the `MethodResponse` (p. 1020) property.
- The source must be an existing method request parameter or a static value. You must enclose static values in single quotation marks and pre-encode these values based on the destination specified in the request.

For more information, see API Gateway API Request and Response Parameter-Mapping Reference in the *API Gateway Developer Guide*.

Required: No

Type: Mapping of key-value pairs

`ResponseTemplates`

The templates used to transform the integration response body. Specify templates as key-value pairs (string-to-string maps), with a content type as the key and a template as the value. For more information, see API Gateway API Request and Response Payload-Mapping Template Reference in the *API Gateway Developer Guide*.

Required: No

Type: Mapping of key-value pairs

`SelectionPattern`

A regular expression (p. 383) that specifies which error strings or status codes from the back end map to the integration response.

Required: No

Type: String

`StatusCode`

The status code that API Gateway uses to map the integration response to a `MethodResponse` (p. 1020) status code.

Required: No

Type: String

Amazon API Gateway Method MethodResponse

`MethodResponse` is a property of the AWS::ApiGateway::Method (p. 408) resource that defines the responses that can be sent to the client who calls an Amazon API Gateway (API Gateway) method.

Syntax

JSON

```
{
  "ResponseModels" : { String:String, ... },
  "ResponseParameters" : { String:Boolean, ... },
  "StatusCode" : String
}
```

YAML

```
ResponseModels:
  String: String
ResponseParameters:
  String: Boolean
StatusCode: String
```

Properties

`ResponseModels`

The resources used for the response's content type. Specify response models as key-value pairs (string-to-string maps), with a content type as the key and a `Model` (p. 412) resource name as the value.

Required: No

Type: Mapping of key-value pairs

`ResponseParameters`

Response parameters that API Gateway sends to the client that called a method. Specify response parameters as key-value pairs (string-to-Boolean maps), with a destination as the key and a Boolean as the value. Specify the destination using the following pattern: `method.response.header.name`, where the `name` is a valid, unique header name. The Boolean specifies whether a parameter is required.

Required: No

Type: Mapping of key-value pairs

`StatusCode`

The method response's status code, which you map to an `IntegrationResponse (p. 1019)`.

Required: Yes

Type: String

Amazon API Gateway RestApi S3Location

`S3Location` is a property of the AWS::ApiGateway::RestApi (p. 417) resource that specifies the Amazon Simple Storage Service (Amazon S3) location of a OpenAPI (formerly Swagger) file that defines a set of RESTful APIs in JSON or YAML for an Amazon API Gateway (API Gateway) RestApi.

Note
On January 1, 2016, the Swagger Specification was donated to the OpenAPI initiative, becoming the foundation of the OpenAPI Specification.

Syntax

JSON

```
{
  "Bucket" : String,
  "ETag" : String,
  "Key" : String,
  "Version" : String
}
```

YAML

```
Bucket: String
ETag: String
Key: String
Version: String
```

Properties

`Bucket`

The name of the S3 bucket where the OpenAPI file is stored.

Required: No

Type: String

`ETag`

The Amazon S3 ETag (a file checksum) of the OpenAPI file. If you don't specify a value, API Gateway skips ETag validation of your OpenAPI file.

Required: No

Type: String

`Key`

The file name of the OpenAPI file (Amazon S3 object name).

Required: No

Type: String

`Version`

For versioning-enabled buckets, a specific version of the OpenAPI file.

Required: No

Type: String

Amazon API Gateway Stage MethodSetting

`MethodSetting` is a property of the AWS::ApiGateway::Stage (p. 420) resource that configures settings for all methods in an Amazon API Gateway (API Gateway) stage.

Syntax

JSON

```
{
  "CacheDataEncrypted" : Boolean,
  "CacheTtlInSeconds" : Integer,
  "CachingEnabled" : Boolean,
  "DataTraceEnabled" : Boolean,
  "HttpMethod" : String,
  "LoggingLevel" : String,
  "MetricsEnabled" : Boolean,
  "ResourcePath" : String,
  "ThrottlingBurstLimit" : Integer,
  "ThrottlingRateLimit" : Number
}
```

YAML

```
CacheDataEncrypted: Boolean
CacheTtlInSeconds: Integer
CachingEnabled: Boolean
DataTraceEnabled: Boolean
HttpMethod: String
LoggingLevel: String
MetricsEnabled: Boolean
ResourcePath: String
ThrottlingBurstLimit: Integer
ThrottlingRateLimit: Number
```

Properties

`CacheDataEncrypted`

Indicates whether the cached responses are encrypted.

Required: No

Type: Boolean

`CacheTtlInSeconds`

The time-to-live (TTL) period, in seconds, that specifies how long API Gateway caches responses.

Required: No

Type: Integer

`CachingEnabled`

Indicates whether responses are cached and returned for requests. You must enable a cache cluster on the stage to cache responses.

Required: No

Type: Boolean

`DataTraceEnabled`

Indicates whether data trace logging is enabled for methods in the stage. API Gateway pushes these logs to Amazon CloudWatch Logs.

Required: No

Type: Boolean

`HttpMethod`

The HTTP method.

Required: Yes

Type: String

`LoggingLevel`

The logging level for this method. For valid values, see the `loggingLevel` property of the Stage resource in the *Amazon API Gateway API Reference*.

Required: No

Type: String

`MetricsEnabled`

Indicates whether Amazon CloudWatch metrics are enabled for methods in the stage.

Required: No

Type: Boolean

`ResourcePath`

The resource path for this method. Forward slashes (`/`) are encoded as `~1` and the initial slash must include a forward slash. For example, the path value `/resource/subresource` must be encoded as `/~1resource~1subresource`. To specify the root path, use only a slash (`/`).

Required: Yes

Type: String

`ThrottlingBurstLimit`

The number of burst requests per second that API Gateway permits across all APIs, stages, and methods in your AWS account. For more information, see Manage API Request Throttling in the *API Gateway Developer Guide*.

Required: No

Type: Integer

`ThrottlingRateLimit`

The number of steady-state requests per second that API Gateway permits across all APIs, stages, and methods in your AWS account. For more information, see Manage API Request Throttling in the *API Gateway Developer Guide*.

Required: No

Type: Number

Amazon API Gateway UsagePlan ApiStage

`ApiStage` is a property of the AWS::ApiGateway::UsagePlan (p. 423) resource that specifies which Amazon API Gateway (API Gateway) stages and APIs to associate with a usage plan.

Syntax

JSON

```
{
  "ApiId" : String,
  "Stage" : String
}
```

YAML

```
ApiId: String
Stage: String
```

Properties

`ApiId`

The ID of an API that is in the specified `Stage` property that you want to associate with the usage plan.

Required: No

Type: String

`Stage`

The name of an API Gateway stage to associate with the usage plan.

Required: No

Type: String

Amazon API Gateway UsagePlan QuotaSettings

`QuotaSettings` is a property of the AWS::ApiGateway::UsagePlan (p. 423) resource that specifies the maximum number of requests users can make to your Amazon API Gateway (API Gateway) APIs.

Syntax

JSON

```
{
  "Limit" : Integer,
  "Offset" : Integer,
  "Period" : String
}
```

YAML

```
Limit: Integer
Offset: Integer
Period: String
```

Properties

`Limit`

The maximum number of requests that users can make within the specified time period.

Required: No

Type: Integer

`Offset`

For the initial time period, the number of requests to subtract from the specified limit. When you first implement a usage plan, the plan might start in the middle of the week or month. With this property, you can decrease the limit for this initial time period.

Required: No

Type: Integer

`Period`

The time period for which the maximum limit of requests applies, such as `DAY` or `WEEK`. For valid values, see the `period` property for the UsagePlan resource in the *Amazon API Gateway REST API Reference*.

Required: No

Type: String

Amazon API Gateway UsagePlan ThrottleSettings

`ThrottleSettings` is a property of the AWS::ApiGateway::UsagePlan (p. 423) resource that specifies the overall request rate (average requests per second) and burst capacity when users call your Amazon API Gateway (API Gateway) APIs.

Syntax

JSON

```
{
```

```
  "BurstLimit" : Integer,
  "RateLimit" : Number
}
```

YAML

```
BurstLimit: Integer
RateLimit: Number
```

Properties

`BurstLimit`

The maximum API request rate limit over a time ranging from one to a few seconds. The maximum API request rate limit depends on whether the underlying token bucket is at its full capacity. For more information about request throttling, see Manage API Request Throttling in the *API Gateway Developer Guide*.

Required: No

Type: Integer

`RateLimit`

The API request steady-state rate limit (average requests per second over an extended period of time). For more information about request throttling, see Manage API Request Throttling in the *API Gateway Developer Guide*.

Required: No

Type: Number

Application Auto Scaling ScalingPolicy StepScalingPolicyConfiguration

`StepScalingPolicyConfiguration` is a property of the AWS::ApplicationAutoScaling::ScalingPolicy (p. 430) resource that configures when Application Auto Scaling scales resources up or down, and by how much.

Syntax

JSON

```
{
  "AdjustmentType" : String,
  "Cooldown" : Integer,
  "MetricAggregationType" : String,
  "MinAdjustmentMagnitude" : Integer,
  "StepAdjustments" : [ StepAdjustment (p. 1027), ... ]
}
```

YAML

```
AdjustmentType: String
Cooldown: Integer
```

```
MetricAggregationType: String
MinAdjustmentMagnitude: Integer
StepAdjustments:
  StepAdjustment
```

Properties

`AdjustmentType`

Specifies whether the `ScalingAdjustment` value in the `StepAdjustment` property is an absolute number or a percentage of the current capacity. For valid values, see the `AdjustmentType` content for the StepScalingPolicyConfiguration data type in the *Application Auto Scaling API Reference*.

Required: No

Type: String

`Cooldown`

The amount of time, in seconds, after a scaling activity completes before any further trigger-related scaling activities can start. For more information, see the `Cooldown` content for the StepScalingPolicyConfiguration data type in the *Application Auto Scaling API Reference*.

Required: No

Type: Integer

`MetricAggregationType`

The aggregation type for the CloudWatch metrics. You can specify `Minimum`, `Maximum`, or `Average`. By default, AWS CloudFormation specifies `Average`. For more information, see Aggregation in the *Amazon CloudWatch User Guide*.

Required: No

Type: String

`MinAdjustmentMagnitude`

The minimum number of resources to adjust when a scaling activity is triggered. If you specify `PercentChangeInCapacity` for the adjustment type, the scaling policy scales the target by this amount.

Required: No

Type: Integer

`StepAdjustments`

A set of adjustments that enable you to scale based on the size of the alarm breach.

Required: No

Type: List of Application Auto Scaling ScalingPolicy StepScalingPolicyConfiguration StepAdjustment (p. 1027)

Application Auto Scaling ScalingPolicy StepScalingPolicyConfiguration StepAdjustment

`StepAdjustment` is a property of the Application Auto Scaling ScalingPolicy StepScalingPolicyConfiguration (p. 1026) property that configures a scaling adjustment based on the difference between the value of the aggregated CloudWatch metric and the breach threshold that you've defined for the alarm (the size of the breach). For more information, see Step Adjustments in the *Auto Scaling User Guide*.

Syntax

JSON

```
{
  "MetricIntervalLowerBound" : Number,
  "MetricIntervalUpperBound" : Number,
  "ScalingAdjustment" : Integer
}
```

YAML

```
MetricIntervalLowerBound: Number
MetricIntervalUpperBound: Number
ScalingAdjustment: Integer
```

Properties

`MetricIntervalLowerBound`

The lower bound of the breach size. The lower bound is the difference between the breach threshold and the aggregated CloudWatch metric value. If the metric value is within the lower and upper bounds, Application Auto Scaling triggers this step adjustment.

If the metric value is above the breach threshold, the metric must be greater than or equal to the threshold plus the lower bound to trigger this step adjustment (the metric value is inclusive). If the metric value is below the breach threshold, the metric must be greater than the threshold plus the lower bound to trigger this step adjustment (the metric value is exclusive). A null value indicates negative infinity.

Required: Conditional. You must specify at least one upper or lower bound.

Type: Number

`MetricIntervalUpperBound`

The upper bound of the breach size. The upper bound is the difference between the breach threshold and the CloudWatch metric value. If the metric value is within the lower and upper bounds, Application Auto Scaling triggers this step adjustment.

If the metric value is above the breach threshold, the metric must be less than the threshold plus the upper bound to trigger this step adjustment (the metric value is exclusive). If the metric value is below the breach threshold, the metric must be less than or equal to the threshold plus the upper bound to trigger this step adjustment (the metric value is inclusive). A null value indicates positive infinity.

Required: Conditional. You must specify at least one upper or lower bound.

Type: Number

`ScalingAdjustment`

The amount by which to scale. The adjustment is based on the value that you specified in the `AdjustmentType` property (either an absolute number or a percentage). A positive value adds to the current capacity and a negative number subtracts from the current capacity.

Required: Yes

Type: Integer

AWS CloudFormation AutoScaling Block Device Mapping Property Type

The `AutoScaling Block Device Mapping` type is an embedded property of the AWS::AutoScaling::LaunchConfiguration (p. 440) type.

Syntax

JSON

```
{
   "DeviceName (p. 1029)" : String,
   "Ebs (p. 1029)" : AutoScaling EBS Block Device,
   "NoDevice" : Boolean,
   "VirtualName (p. 1030)" : String
}
```

YAML

```
DeviceName (p. 1029): String
Ebs (p. 1029):
  AutoScaling EBS Block Device
NoDevice: Boolean
VirtualName (p. 1030): String
```

Properties

Note
For more information about the constraints and valid values of each property, see Ebs in the *Auto Scaling API Reference*.

`DeviceName`

The name of the device within Amazon EC2.

Required: Yes

Type: String

`Ebs`

The Amazon Elastic Block Store volume information.

Required: Conditional You can specify either `VirtualName` or `Ebs`, but not both.

Type: AutoScaling EBS Block Device (p. 1030).

`NoDevice`

Suppresses the device mapping. If `NoDevice` is set to true for the root device, the instance might fail the Amazon EC2 health check. Auto Scaling launches a replacement instance if the instance fails the health check.

Required: No

Type: Boolean

`VirtualName`

The name of the virtual device. The name must be in the form `ephemeralX` where *X* is a number starting from zero (0), for example, `ephemeral0`.

Required: Conditional You can specify either `VirtualName` or `Ebs`, but not both.

Type: String

AWS CloudFormation AutoScaling EBS Block Device Property Type

The AutoScaling EBS Block Device type is an embedded property of the AutoScaling Block Device Mapping (p. 1029) type.

Syntax

JSON

```
{
  "DeleteOnTermination" : Boolean,
  "Encrypted" : Boolean,
  "Iops" : Integer,
  "SnapshotId (p. 1031)" : String,
  "VolumeSize (p. 1031)" : Integer,
  "VolumeType" : String
}
```

YAML

```
DeleteOnTermination: Boolean
Encrypted: Boolean
Iops: Integer
SnapshotId (p. 1031): String
VolumeSize (p. 1031): Integer
VolumeType: String
```

Properties

`DeleteOnTermination`

Indicates whether to delete the volume when the instance is terminated. By default, Auto Scaling uses `true`.

Required: No

Type: Boolean

`Encrypted`

Indicates whether the volume is encrypted. Encrypted EBS volumes must be attached to instances that support Amazon EBS encryption. Volumes that you create from encrypted snapshots are automatically encrypted. You cannot create an encrypted volume from an unencrypted snapshot or an unencrypted volume from an encrypted snapshot.

Required: No

Type: Boolean

`Iops`

The number of I/O operations per second (IOPS) that the volume supports. The maximum ratio of IOPS to volume size is 30.

Required: No

Type: Integer.

`SnapshotId`

The snapshot ID of the volume to use.

Required: Conditional If you specify both `SnapshotId` and `VolumeSize`, `VolumeSize` must be equal or greater than the size of the snapshot.

Type: String

`VolumeSize`

The volume size, in Gibibytes (GiB). This can be a number from 1 – 1024. If the volume type is EBS optimized, the minimum value is 10. For more information about specifying the volume type, see EbsOptimized in AWS::AutoScaling::LaunchConfiguration (p. 440).

Required: Conditional If you specify both `SnapshotId` and `VolumeSize`, `VolumeSize` must be equal or greater than the size of the snapshot.

Type: Integer.

Update requires: Some interruptions (p. 90)

`VolumeType`

The volume type. By default, Auto Scaling uses the `standard` volume type. For more information, see Ebs in the *Auto Scaling API Reference*.

Required: No

Type: String

Examples

For AutoScaling EBS Block Device snippets, see Auto Scaling Launch Configuration Resource (p. 240).

Auto Scaling MetricsCollection

The `MetricsCollection` is a property of the AWS::AutoScaling::AutoScalingGroup (p. 433) resource that describes the group metrics that an Auto Scaling group sends to CloudWatch. These metrics describe the group rather than any of its instances. For more information, see EnableMetricsCollection in the *Auto Scaling API Reference*.

Syntax

JSON

```
{
  "Granularity" : String,
```

```
  "Metrics" : [ String, ... ]
}
```

YAML

```
Granularity: String
Metrics:
  - String
```

Properties

`Granularity`

The frequency at which Auto Scaling sends aggregated data to CloudWatch. For example, you can specify `1Minute` to send aggregated data to CloudWatch every minute.

Required: Yes

Type: String

`Metrics`

The list of metrics to collect. If you don't specify any metrics, all metrics are enabled.

Required: No

Type: List of strings

Auto Scaling NotificationConfigurations

The `NotificationConfigurations` property is an embedded property of the AWS::AutoScaling::AutoScalingGroup (p. 433) resource that specifies the events for which the Auto Scaling group sends notifications.

Syntax

JSON

```
{
   "NotificationTypes" : [ String, ... ],
   "TopicARN" : String
}
```

YAML

```
NotificationTypes:
  - String
TopicARN: String
```

Properties

`NotificationTypes`

A list of event types that trigger a notification. Event types can include any of the following types: `autoscaling:EC2_INSTANCE_LAUNCH`, `autoscaling:EC2_INSTANCE_LAUNCH_ERROR`, `autoscaling:EC2_INSTANCE_TERMINATE`, `autoscaling:EC2_INSTANCE_TERMINATE_ERROR`, and `autoscaling:TEST_NOTIFICATION`. For more information about event types, see DescribeAutoScalingNotificationTypes in the *Auto Scaling API Reference*.

Required: Yes

Type: List of strings

`TopicARN`

The Amazon Resource Name (ARN) of the Amazon Simple Notification Service (SNS) topic.

Required: Yes

Type: String

Examples

For NotificationConfigurations snippets, see Auto Scaling Group with Notifications (p. 242).

Auto Scaling ScalingPolicy StepAdjustments

`StepAdjustments` is a property of the AWS::AutoScaling::ScalingPolicy (p. 452) resource that describes a scaling adjustment based on the difference between the value of the aggregated CloudWatch metric and the breach threshold that you've defined for the alarm. For more information, see StepAdjustment in the *Auto Scaling API Reference*.

Syntax

JSON

```
{
  "MetricIntervalLowerBound" : Number,
  "MetricIntervalUpperBound" : Number,
  "ScalingAdjustment" : Integer
}
```

YAML

```
MetricIntervalLowerBound: Number
MetricIntervalUpperBound: Number
ScalingAdjustment: Integer
```

Properties

`MetricIntervalLowerBound`

The lower bound of the breach size. The lower bound is the difference between the breach threshold and the aggregated CloudWatch metric value. If the metric value is within the lower and upper bounds, Auto Scaling triggers this step adjustment.

If the metric value is above the breach threshold, the metric must be greater than or equal to the threshold plus the lower bound to trigger this step adjustment (the metric value is inclusive). If the metric value is below the breach threshold, the metric must be greater than the threshold plus the lower bound to trigger this step adjustment (the metric value is exclusive). A null value indicates negative infinity.

Required: Conditional. You must specify at least one upper or lower bound.

Type: Number

`MetricIntervalUpperBound`

The upper bound of the breach size. The upper bound is the difference between the breach threshold and the CloudWatch metric value. If the metric value is within the lower and upper bounds, Auto Scaling triggers this step adjustment.

If the metric value is above the breach threshold, the metric must be less than the threshold plus the upper bound to trigger this step adjustment (the metric value is exclusive). If the metric value is below the breach threshold, the metric must be less than or equal to the threshold plus the upper bound to trigger this step adjustment (the metric value is inclusive). A null value indicates positive infinity.

Required: Conditional. You must specify at least one upper or lower bound.

Type: Number

`ScalingAdjustment`

The amount by which to scale. The adjustment is based on the value that you specified in the `AdjustmentType` property (either an absolute number or a percentage). A positive value adds to the current capacity and a negative number subtracts from the current capacity.

Required: Yes

Type: Integer

Auto Scaling Tags Property Type

The Auto Scaling Tags property is an embedded property of the AWS::AutoScaling::AutoScalingGroup (p. 433) type. For more information about tags, go to Tagging Auto Scaling Groups and Amazon EC2 Instances in the *Auto Scaling User Guide*.

AWS CloudFormation adds the following tags to all Auto Scaling groups and associated instances:

- aws:cloudformation:stack-name
- aws:cloudformation:stack-id
- aws:cloudformation:logical-id

Syntax

JSON

```
{
   "Key (p. 1035)" : String,
   "Value (p. 1035)" : String,
   "PropagateAtLaunch (p. 1035)" : Boolean
}
```

YAML

```
Key (p. 1035): String
Value (p. 1035): String
```

```
PropagateAtLaunch (p. 1035): Boolean
```

Properties

`Key`
: The key name of the tag.

  Required: Yes

  Type: String

`Value`
: The value for the tag.

  Required: Yes

  Type: String

`PropagateAtLaunch`
: Set to `true` if you want AWS CloudFormation to copy the tag to EC2 instances that are launched as part of the auto scaling group. Set to `false` if you want the tag attached only to the auto scaling group and not copied to any instances launched as part of the auto scaling group.

  Required: Yes

  Type: Boolean

Example

The following example template snippet creates two Auto Scaling tags. The first tag, `MyTag1`, is attached to an Auto Scaling group named `WebServerGroup` and is copied to any EC2 instances launched as part of the Auto Scaling group. The second tag, `MyTag2`, is attached only to the Auto Scaling group named `WebServerGroup`.

```
"WebServerGroup" : {
   "Type" : "AWS::AutoScaling::AutoScalingGroup",
   "Properties" : {
      "AvailabilityZones" : { "Fn::GetAZs" : "" },
      "LaunchConfigurationName" : { "Ref" : "LaunchConfig" },
      "MinSize" : "1",
      "MaxSize" : "2",
      "LoadBalancerNames" : [ { "Ref" : "ElasticLoadBalancer" } ],
      "Tags" : [ {
         "Key" : "MyTag1",
         "Value" : "Hello World 1",
         "PropagateAtLaunch" : "true"
      }, {
         "Key" : "MyTag2",
         "Value" : "Hello World 2",
         "PropagateAtLaunch" : "false"
      } ]
   }
}
```

AWS Certificate Manager Certificate DomainValidationOption

`DomainValidationOption` is a property of the AWS::CertificateManager::Certificate (p. 459) resource that specifies the AWS Certificate Manager (ACM) Certificate domain that registrars use to send validation emails.

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "DomainName" : String,
  "ValidationDomain" : String
}
```

YAML

```
DomainName: String
ValidationDomain: String
```

Properties

`DomainName`

Fully Qualified Domain Name (FQDN) of the Certificate that you are requesting.

Required: Yes

Type: String

`ValidationDomain`

The domain that domain name registrars use to send validation emails. Registrars use this value as the email address suffix when sending emails to verify your identity. This value must be the same as the domain name or a superdomain of the domain name. For more information, see the `ValidationDomain` content for the DomainValidationOption data type in the *AWS Certificate Manager API Reference*.

Required: Yes

Type: String

CloudFormation Stack Parameters Property Type

The `Parameters` type is an embedded property of the AWS::CloudFormation::Stack (p. 485) type.

The Parameters type contains a set of value pairs that represent the parameters that will be passed to the template used to create an `AWS::CloudFormation::Stack` resource. Each parameter has a name corresponding to a parameter defined in the embedded template and a value representing the value that you want to set for the parameter. For example, the sample template EC2ChooseAMI.template contains the following Parameters section:

JSON

```
"Parameters" : {
   "InstanceType" : {
      "Type" : "String",
      "Default" : "m1.small",
      "Description" : "EC2 instance type, e.g. m1.small, m1.large, etc."
   },
```

```
   "WebServerPort" : {
      "Type" : "String",
      "Default" : "80",
      "Description" : "TCP/IP port of the web server"
   },
   "KeyName" : {
      "Type" : "String",
      "Description" : "Name of an existing EC2 KeyPair to enable SSH access to the web
 server"
   }
}
```

YAML

```
Parameters:
  InstanceType:
    Type: "String"
    Default: "m1.small"
    Description: "EC2 instance type, e.g. m1.small, m1.large, etc."
  WebServerPort:
    Type: "String"
    Default: "80"
    Description: "TCP/IP port of the web server"
  KeyName:
    Type: "String"
    Description: "Name of an existing EC2 KeyPair to enable SSH access to the web server"
```

Nested Stack

You could use the following template to embed a stack (myStackWithParams) using the EC2ChooseAMI.template and use the Parameters property in the AWS::CloudFormation::Stack resource to specify an InstanceType and KeyName:

JSON

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "myStackWithParams" : {
         "Type" : "AWS::CloudFormation::Stack",
         "Properties" : {
            "TemplateURL" : "https://s3.amazonaws.com/cloudformation-templates-us-east-1/
EC2ChooseAMI.template",
            "Parameters" : {
               "InstanceType" : "t1.micro",
               "KeyName" : "mykey"
            }
         }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  myStackWithParams:
```

```
    Type: "AWS::CloudFormation::Stack"
    Properties:
      TemplateURL: "https://s3.amazonaws.com/cloudformation-templates-us-east-1/
EC2ChooseAMI.template"
      Parameters:
        InstanceType: "t1.micro"
        KeyName: "mykey"
```

AWS CloudFormation Interface Label

`Label` is a property of the ParameterGroup (p. 1038) and ParameterLabel (p. 1039) properties that defines name for a parameter group or parameter.

Syntax

JSON

```
{
  "default" : String
}
```

YAML

```
default: String
```

Properties

`default`

The default label that the AWS CloudFormation console uses to name a parameter group or parameter.

Required: No

Type: String

AWS CloudFormation Interface ParameterGroup

`ParameterGroup` is a property of the AWS::CloudFormation::Interface (p. 483) resource that defines a parameter group and the parameters to include in the group.

Syntax

JSON

```
{
  "Label" : Label,
  "Parameters" : [ String, ... ]
}
```

YAML

```
Label: Label
```

```
Parameters:
  - String
```

Properties

`Label`

A name for the parameter group.

Required: No

Type: AWS CloudFormation Interface Label (p. 1038)

`Parameters`

A list of case-sensitive parameter logical IDs to include in the group. Parameters must already be defined in the `Parameters` section of the template. A parameter can be included in only one parameter group.

The console lists the parameters that you don't associate with a parameter group in alphabetical order in the `Other parameters` group.

Required: No

Type: List of strings

AWS CloudFormation Interface ParameterLabel

`ParameterLabel` is a property of the AWS::CloudFormation::Interface (p. 483) resource that specifies a friendly name or description for a parameter that the AWS CloudFormation console shows instead of the parameter's logical ID.

Syntax

JSON

```
{
  "ParameterLogicalID" : Label
}
```

YAML

```
ParameterLogicalID: Label
```

Properties

`ParameterLogicalID`

A label for a parameter. The label defines a friendly name or description that the AWS CloudFormation console shows on the **Specify Parameters** page when a stack is created or updated. The *`ParameterLogicalID`* key must be the case-sensitive logical ID of a valid parameter that has been declared in the `Parameters` section of the template.

Required: No

Type: AWS CloudFormation Interface Label (p. 1038)

CloudFront DistributionConfig

`DistributionConfig` is a property of the AWS::CloudFront::Distribution (p. 492) property that describes which Amazon CloudFront origin servers to get your files from when users request the files through your website or application.

Syntax

JSON

```
{
   "Aliases (p. 1040)" : [ String, ... ],
   "CacheBehaviors (p. 1041)" : [ CacheBehavior, ... ],
   "Comment (p. 1041)" : String,
   "CustomErrorResponses" : [ CustomErrorResponse, ... ],
   "DefaultCacheBehavior (p. 1041)" : DefaultCacheBehavior,
   "DefaultRootObject (p. 1041)" : String,
   "Enabled (p. 1041)" : Boolean,
   "HttpVersion" : String,
   "Logging (p. 1042)" : Logging,
   "Origins (p. 1042)" : [ Origin, ... ],
   "PriceClass" : String,
   "Restrictions" : Restriction,
   "ViewerCertificate" : ViewerCertificate,
   "WebACLId" : String
}
```

YAML

```
Aliases (p. 1040):
  - String
CacheBehaviors (p. 1041):
  CacheBehavior
Comment (p. 1041): String
CustomErrorResponses:
  CustomErrorResponse
DefaultCacheBehavior (p. 1041):
  DefaultCacheBehavior
DefaultRootObject (p. 1041): String
Enabled (p. 1041): Boolean
HttpVersion: String
Logging (p. 1042):
  Logging
Origins (p. 1042):
  Origin
PriceClass: String
Restrictions:
  Restriction
ViewerCertificate:
  ViewerCertificate
WebACLId: String
```

Properties

`Aliases`

CNAMEs (alternate domain names), if any, for the distribution.

Required: No

Type: List of strings

`CacheBehaviors`

A list of CacheBehavior types for the distribution.

Required: No

Type: List of CacheBehavior (p. 1042)

`Comment`

Any comments that you want to include about the distribution.

Required: No

Type: String

`CustomErrorResponses`

Whether CloudFront replaces HTTP status codes in the `4xx` and `5xx` range with custom error messages before returning the response to the viewer.

Required: No

Type List of CloudFront DistributionConfig CustomErrorResponse (p. 1045)

`DefaultCacheBehavior`

The default cache behavior that is triggered if you do not specify the `CacheBehavior` property or if files don't match any of the values of `PathPattern` in the `CacheBehavior` property.

Required: Yes

Type: DefaultCacheBehavior type (p. 1046)

`DefaultRootObject`

The object (such as `index.html`) that you want CloudFront to request from your origin when the root URL for your distribution (such as `http://example.com/`) is requested.

Note
Specifying a default root object avoids exposing the contents of your distribution.

Required: No

Type: String

`Enabled`

Controls whether the distribution is enabled to accept end user requests for content.

Required: Yes

Type: Boolean

`HttpVersion`

The latest HTTP version that viewers can use to communicate with CloudFront. Viewers that don't support the latest version automatically use an earlier HTTP version. By default, AWS CloudFormation specifies `http1.1`.

For valid values, see the `HttpVersion` content for the DistributionConfig data type in the *Amazon CloudFront API Reference*.

Required: No

Type: String

`Logging`

Controls whether access logs are written for the distribution. To turn on access logs, specify this property.

Required: No

Type: Logging (p. 1049) type

`Origins`

A list of origins for this CloudFront distribution. For each origin, you can specify whether it is an Amazon S3 or custom origin.

Required: Yes

Type: List of Origins (p. 1050).

`PriceClass`

The price class that corresponds with the maximum price that you want to pay for the CloudFront service. For more information, see Choosing the Price Class in the *Amazon CloudFront Developer Guide*.

For more information about the valid values, see the `PriceClass` content for the DistributionConfig data type in the *Amazon CloudFront API Reference*.

Required: No

Type: String

`Restrictions`

Specifies restrictions on who or how viewers can access your content.

Required: No

Type: CloudFront DistributionConfiguration Restrictions (p. 1054)

`ViewerCertificate`

The certificate to use when viewers use HTTPS to request objects.

Required: No

Type: CloudFront DistributionConfiguration ViewerCertificate (p. 1055)

`WebACLId`

The AWS WAF web ACL (p. 994) to associate with this distribution. AWS WAF is a web application firewall that enables you to monitor the HTTP and HTTPS requests that are forwarded to CloudFront and to control who can access your content. CloudFront permits or forbids requests based on conditions that you specify, such as the IP addresses from which requests originate or the values of query strings.

Required: No

Type: String

CloudFront DistributionConfig CacheBehavior

`CacheBehavior` is a property of the DistributionConfig (p. 1040) property that describes the Amazon CloudFront (CloudFront) cache behavior when the requested URL matches a pattern.

Syntax

JSON

```
{
  "AllowedMethods" : [ String, ... ],
  "CachedMethods" : [ String, ... ],
  "Compress" : Boolean,
  "DefaultTTL" : Number,
  "ForwardedValues" : ForwardedValues,
  "MaxTTL" : Number,
  "MinTTL" : Number,
  "PathPattern" : String,
  "SmoothStreaming" : Boolean,
  "TargetOriginId" : String,
  "TrustedSigners" : [ String, ... ],
  "ViewerProtocolPolicy" : String
}
```

YAML

```
AllowedMethods:
  - String
CachedMethods:
  - String
Compress: Boolean
DefaultTTL: Number
ForwardedValues:
  ForwardedValues
MaxTTL: Number
MinTTL: Number
PathPattern: String
SmoothStreaming: Boolean
TargetOriginId: String
TrustedSigners:
  - String
ViewerProtocolPolicy: String
```

Properties

> **Note**
> For more information about the constraints and valid values of each property, see the CacheBehavior data type in the *Amazon CloudFront API Reference*.

`AllowedMethods`

HTTP methods that CloudFront processes and forwards to your Amazon S3 bucket or your custom origin. You can specify `["HEAD", "GET"]`, `["GET", "HEAD", "OPTIONS"]`, or `["DELETE", "GET", "HEAD", "OPTIONS", "PATCH", "POST", "PUT"]`. If you don't specify a value, AWS CloudFormation specifies `["HEAD", "GET"]`.

Required: No

Type: List of strings

`CachedMethods`

HTTP methods for which CloudFront caches responses. You can specify `["HEAD", "GET"]` or `["GET", "HEAD", "OPTIONS"]`. If you don't specify a value, AWS CloudFormation specifies `["HEAD", "GET"]`.

Required: No

Type: List of strings

`Compress`

Indicates whether CloudFront automatically compresses certain files for this cache behavior. For more information, see Serving Compressed Files in the *Amazon CloudFront Developer Guide*.

Required: No

Type: Boolean

`DefaultTTL`

The default time in seconds that objects stay in CloudFront caches before CloudFront forwards another request to your custom origin to determine whether the object has been updated. This value applies only when your custom origin does not add HTTP headers, such as `Cache-Control max-age`, `Cache-Control s-maxage`, and `Expires` to objects.

By default, AWS CloudFormation specifies `86400` seconds (one day). If the value of the `MinTTL` property is greater than the default value, CloudFront uses the minimum Time to Live (TTL) value.

Required: No

Type: Number

`ForwardedValues`

Specifies how CloudFront handles query strings or cookies.

Required: Yes

Type: ForwardedValues (p. 1057) type

`MaxTTL`

The maximum time in seconds that objects stay in CloudFront caches before CloudFront forwards another request to your custom origin to determine whether the object has been updated. This value applies only when your custom origin does not add HTTP headers, such as `Cache-Control max-age`, `Cache-Control s-maxage`, and `Expires` to objects.

By default, AWS CloudFormation specifies `31536000` seconds (one year). If the value of the `MinTTL` or `DefaultTTL` property is greater than the maximum value, CloudFront uses the default TTL value.

Required: No

Type: Number

`MinTTL`

The minimum amount of time that you want objects to stay in the cache before CloudFront queries your origin to see whether the object has been updated.

Required: No

Type: Number

`PathPattern`

The pattern to which this cache behavior applies. For example, you can specify `images/*.jpg`.

When CloudFront receives an end-user request, CloudFront compares the requested path with path patterns in the order in which cache behaviors are listed in the template.

Required: Yes

Type: String

`SmoothStreaming`

Indicates whether to use the origin that is associated with this cache behavior to distribute media files in the Microsoft Smooth Streaming format. If you specify `true`, you can still use this cache behavior to distribute other content if the content matches the `PathPattern` value.

Required: No

Type: Boolean

`TargetOriginId`

The ID value of the origin to which you want CloudFront to route requests when a request matches the value of the `PathPattern` property.

Required: Yes

Type: String

`TrustedSigners`

A list of AWS accounts that can create signed URLs in order to access private content.

Required: No

Type: List of strings

`ViewerProtocolPolicy`

The protocol that users can use to access the files in the origin that you specified in the `TargetOriginId` property when a request matches the value of the `PathPattern` property. For more information about the valid values, see the `ViewerProtocolPolicy` content for the CacheBehavior data type in the *Amazon CloudFront API Reference*.

Required: Yes

Type: String

CloudFront DistributionConfig CustomErrorResponse

`CustomErrorResponse` is a property of the CloudFront DistributionConfig (p. 1040) resource that defines custom error messages for certain HTTP status codes.

Syntax

JSON

```
{
  "ErrorCachingMinTTL" : Integer,
  "ErrorCode" : Integer,
  "ResponseCode" : Integer,
  "ResponsePagePath" : String
}
```

YAML

```
ErrorCachingMinTTL: Integer
ErrorCode: Integer
ResponseCode: Integer
```

```
ResponsePagePath: String
```

Properties

Note
For more information about the constraints and valid values of each property, see the CustomErrorResponse data type in the *Amazon CloudFront API Reference*.

`ErrorCachingMinTTL`

The minimum amount of time, in seconds, that Amazon CloudFront caches the HTTP status code that you specified in the `ErrorCode` property. The default value is `300`.

Required: No

Type: Integer

`ErrorCode`

An HTTP status code for which you want to specify a custom error page. You can specify `400`, `403`, `404`, `405`, `414`, `500`, `501`, `502`, `503`, or `504`.

Required: Yes

Type: Integer

`ResponseCode`

The HTTP status code that CloudFront returns to viewer along with the custom error page. You can specify `200`, `400`, `403`, `404`, `405`, `414`, `500`, `501`, `502`, `503`, or `504`.

Required: Conditional. Required if you specified the `ResponsePagePath` property.

Type: Integer

`ResponsePagePath`

The path to the custom error page that CloudFront returns to a viewer when your origin returns the HTTP status code that you specified in the `ErrorCode` property. For example, you can specify `/404-errors/403-forbidden.html`.

Required: Conditional. Required if you specified the `ResponseCode` property.

Type: String

CloudFront DefaultCacheBehavior

`DefaultCacheBehavior` is a property of the DistributionConfig (p. 1040) property that describes the default cache behavior for an Amazon CloudFront distribution.

Syntax

JSON

```
{
  "AllowedMethods" : [ String, ... ],
  "CachedMethods" : [ String, ... ],
  "Compress" : Boolean,
  "DefaultTTL" : Number,
```

```
  "ForwardedValues" : ForwardedValues,
  "MaxTTL" : Number,
  "MinTTL" : Number,
  "SmoothStreaming" : Boolean,
  "TargetOriginId" : String,
  "TrustedSigners" : [ String, ... ],
  "ViewerProtocolPolicy" : String
}
```

YAML

```
AllowedMethods:
  - String
CachedMethods:
  - String
Compress: Boolean
DefaultTTL: Number
ForwardedValues:
  ForwardedValues
MaxTTL: Number
MinTTL: Number
SmoothStreaming: Boolean
TargetOriginId: String
TrustedSigners:
  - String
ViewerProtocolPolicy : String
```

Properties

Note

For more information about the constraints and valid values of each property, see the DefaultCacheBehavior data type in the *Amazon CloudFront API Reference*.

`AllowedMethods`

HTTP methods that CloudFront processes and forwards to your Amazon S3 bucket or your custom origin. In AWS CloudFormation templates, you can specify `["HEAD", "GET"]`, `["GET", "HEAD", "OPTIONS"]`, or `["DELETE", "GET", "HEAD", "OPTIONS", "PATCH", "POST", "PUT"]`. If you don't specify a value, AWS CloudFormation specifies `["HEAD", "GET"]`.

Required: No

Type: List of strings

`CachedMethods`

HTTP methods for which CloudFront caches responses. In AWS CloudFormation templates, you can specify `["HEAD", "GET"]` or `["GET", "HEAD", "OPTIONS"]`. If you don't specify a value, AWS CloudFormation specifies `["HEAD", "GET"]`.

Required: No

Type: List of strings

`Compress`

Indicates whether CloudFront automatically compresses certain files for this cache behavior. For more information, see Serving Compressed Files in the *Amazon CloudFront Developer Guide*.

Required: No

Type: Boolean

`DefaultTTL`

The default time in seconds that objects stay in CloudFront caches before CloudFront forwards another request to your custom origin to determine whether the object has been updated. This value applies only when your custom origin does not add HTTP headers, such as `Cache-Control max-age`, `Cache-Control s-maxage`, and `Expires` to objects.

By default, AWS CloudFormation specifies `86400` seconds (one day). If the value of the `MinTTL` property is greater than the default value, CloudFront uses the minimum Time To Live (TTL) value.

Required: No

Type: Number

`ForwardedValues`

Specifies how CloudFront handles query strings or cookies.

Required: Yes

Type: ForwardedValues (p. 1057) type

`MaxTTL`

The maximum time in seconds that objects stay in CloudFront caches before CloudFront forwards another request to your custom origin to determine whether the object has been updated. This value applies only when your custom origin does not add HTTP headers, such as `Cache-Control max-age`, `Cache-Control s-maxage`, and `Expires` to objects.

By default, AWS CloudFormation specifies `31536000` seconds (one year). If the value of the `MinTTL` or `DefaultTTL` property is greater than the maximum value, CloudFront uses the default TTL value.

Required: No

Type: Number

`MinTTL`

The minimum amount of time that you want objects to stay in the cache before CloudFront queries your origin to see whether the object has been updated.

Required: No

Type: String

`SmoothStreaming`

Indicates whether to use the origin that is associated with this cache behavior to distribute media files in the Microsoft Smooth Streaming format.

Required: No

Type: Boolean

`TargetOriginId`

The value of ID for the origin that CloudFront routes requests to when the default cache behavior is applied to a request.

Required: Yes

Type: String

`TrustedSigners`

A list of AWS accounts that can create signed URLs in order to access private content.

Required: No

Type: List of strings

`ViewerProtocolPolicy`

The protocol that users can use to access the files in the origin that you specified in the `TargetOriginId` property when the default cache behavior is applied to a request. For more information about the valid values, see the `ViewerProtocolPolicy` content for the DefaultCacheBehavior data type in the *Amazon CloudFront API Reference*.

Required: Yes

Type: String

CloudFront Logging

`Logging` is a property of the DistributionConfig (p. 1040) property that enables Amazon CloudFront to deliver access logs for each distribution to an Amazon Simple Storage Service (S3) bucket.

Syntax

JSON

```
{
  "Bucket" : String,
  "IncludeCookies" : Boolean,
  "Prefix" : String
}
```

YAML

```
Bucket: String
IncludeCookies: Boolean
Prefix: String
```

Properties

Note
For more information about the constraints and valid values of each property, see the LoggingConfig data type in the *Amazon CloudFront API Reference*.

`Bucket`

The Amazon S3 bucket address where access logs are stored, for example, `mybucket.s3.amazonaws.com`.

Required: Yes

Type: String

`IncludeCookies`

Indicates whether CloudFront includes cookies in access logs.

Required: No

Type: Boolean

`Prefix`

A prefix for the access log file names for this distribution.

Required: No

Type: String

CloudFront DistributionConfig Origin

`Origin` is a property of the DistributionConfig (p. 1040) property that describes an Amazon CloudFront distribution origin.

Syntax

JSON

```
{
  "CustomOriginConfig" : Custom Origin,
  "DomainName" : String,
  "Id" : String,
  "OriginCustomHeaders" : [ OriginCustomHeader, ... ]
  "OriginPath" : String,
  "S3OriginConfig" : S3 Origin
}
```

YAML

```
CustomOriginConfig:
  Custom Origin
DomainName: String
Id: String
OriginCustomHeaders:
  - OriginCustomHeader
OriginPath: String
S3OriginConfig:
  S3 Origin
```

Properties

Note
For more information about the constraints and valid values of each property, see the Origin data type in the *Amazon CloudFront API Reference*.

`CustomOriginConfig`

Origin information to specify a custom origin.

Required: Conditional. You cannot use `CustomOriginConfig` and `S3OriginConfig` in the same distribution, but you *must* specify one or the other.

Type: CustomOrigin (p. 1051) type

`DomainName`

The DNS name of the Amazon Simple Storage Service (S3) bucket or the HTTP server from which you want CloudFront to get objects for this origin.

Required: Yes

Type: String

`Id`

An identifier for the origin. The value of `Id` must be unique within the distribution.

Required: Yes

Type: String

`OriginCustomHeaders`

Custom headers that CloudFront includes when it forwards a request to your origin.

Required: No

Type: List of OriginCustomHeader (p. 1052) type

`OriginPath`

The path that CloudFront uses to request content from an S3 bucket or custom origin. The combination of the `DomainName` and `OriginPath` properties must resolve to a valid path. The value must start with a slash mark (`/`) and cannot end with a slash mark.

Required: No

Type: String

`S3OriginConfig`

Origin information to specify an S3 origin.

Required: Conditional. You cannot use `S3OriginConfig` and `CustomOriginConfig` in the same distribution, but you *must* specify one or the other.

Type: S3Origin (p. 1053) type

CloudFront DistributionConfig Origin CustomOrigin

`CustomOrigin` is a property of the Amazon CloudFront Origin (p. 1050) property that describes an HTTP server.

Syntax

JSON

```
{
  "HTTPPort" : Integer,
  "HTTPSPort" : Integer,
  "OriginProtocolPolicy" : String,
  "OriginSSLProtocols" : [ String, ... ]
}
```

YAML

```
HTTPPort: Integer
HTTPSPort: Integer
OriginProtocolPolicy: String
```

```
OriginSSLProtocols:
  - String
```

Properties

Note
For more information about the constraints and valid values of each property, see the CustomOriginConfig data type in the *Amazon CloudFront API Reference*.

`HTTPPort`

The HTTP port the custom origin listens on.

Required: No

Type: Integer

`HTTPSPort`

The HTTPS port the custom origin listens on.

Required: No

Type: Integer

`OriginProtocolPolicy`

The origin protocol policy to apply to your origin.

Required: Yes

Type: String

`OriginSSLProtocols`

The SSL protocols that CloudFront can use when establishing an HTTPS connection with your origin. By default, AWS CloudFormation specifies the `TLSv1` and `SSLv3` protocols.

Required: No

Type: List of strings

CloudFront DistributionConfig Origin OriginCustomHeader

`OriginCustomHeader` is a property of the Amazon CloudFront Origin (p. 1050) property that specifies the custom headers CloudFront includes when it forwards requests to your origin.

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "HeaderName" : String,
  "HeaderValue" : String
}
```

YAML

```
HeaderName: String
HeaderValue: String
```

Properties

`HeaderName`

The name of a header that CloudFront forwards to your origin. For more information, see Forwarding Custom Headers to Your Origin (Web Distributions Only) in the *Amazon CloudFront Developer Guide*.

Required: Yes

Type: String

`HeaderValue`

The value for the header that you specified in the `HeaderName` property.

Required: Yes

Type: String

CloudFront DistributionConfig Origin S3Origin

`S3Origin` is a property of the Origin (p. 1050) property that describes the Amazon Simple Storage Service (S3) origin to associate with an Amazon CloudFront origin.

Syntax

JSON

```
{
   "OriginAccessIdentity" : String
}
```

YAML

```
OriginAccessIdentity: String
```

Properties

Note
For more information about the constraints and valid values of each property, see the S3Origin data type in the *Amazon CloudFront API Reference*.

`OriginAccessIdentity`

The CloudFront origin access identity to associate with the origin. This is used to configure the origin so that end users can access objects in an Amazon S3 bucket through CloudFront only.

Required: No

Type: String

CloudFront DistributionConfiguration Restrictions

`Restrictions` is a property of the CloudFront DistributionConfig (p. 1040) property that lets you limit which viewers can access your content.

Syntax

JSON

```
{
  "GeoRestriction" : GeoRestriction
}
```

YAML

```
GeoRestriction: GeoRestriction
```

Properties

Note
For more information about the constraints and valid values of each property, see the Restrictions data type in the *Amazon CloudFront API Reference*.

`GeoRestriction`

The countries in which viewers are able to access your content.

Required: Yes

Type: CloudFront DistributionConfig Restrictions GeoRestriction (p. 1054)

CloudFront DistributionConfig Restrictions GeoRestriction

`GeoRestriction` is a property of the CloudFront DistributionConfiguration Restrictions (p. 1054) property that describes the countries in which Amazon CloudFront allows viewers to access your content.

Syntax

JSON

```
{
  "Locations" : [ String, ... ],
  "RestrictionType" : String
}
```

YAML

```
Locations:
  - String
RestrictionType: String
```

Properties

Note
For more information about the constraints and valid values of each property, see the GeoRestriction data type in the *Amazon CloudFront API Reference*.

`Locations`

The two-letter, uppercase country code for a country that you want to include in your blacklist or whitelist.

Required: Conditional. Required if you specified `blacklist` or `whitelist` for the `RestrictionType` property.

Type: List of strings

`RestrictionType`

The method to restrict distribution of your content:

`blacklist`

Prevents viewers in the countries that you specified from accessing your content.

`whitelist`

Allows viewers in the countries that you specified to access your content.

`none`

No distribution restrictions by country.

Required: Yes

Type: String

CloudFront DistributionConfiguration ViewerCertificate

`ViewerCertificate` is a property of the CloudFront DistributionConfig (p. 1040) property that specifies which certificate to use when viewers use HTTPS to request objects.

Syntax

JSON

```
{
  "AcmCertificateArn" : String,
  "CloudFrontDefaultCertificate" : Boolean,
  "IamCertificateId" : String,
  "MinimumProtocolVersion" : String,
  "SslSupportMethod" : String
}
```

YAML

```
AcmCertificateArn: String
CloudFrontDefaultCertificate: Boolean
IamCertificateId: String
MinimumProtocolVersion: String
```

```
SslSupportMethod: String
```

Properties

`AcmCertificateArn`

If you're using an alternate domain name, the Amazon Resource Name (ARN) of an AWS Certificate Manager (ACM) certificate. Use the ACM service to provision and manage your certificates. For more information, see the AWS Certificate Manager User Guide.

Note
Currently, you can specify only certificates that are in the US East (N. Virginia) region.

Required: Conditional. You must specify one of the following properties: `AcmCertificateArn`, `CloudFrontDefaultCertificate`, or `IamCertificateId`.

Type: String

`CloudFrontDefaultCertificate`

Indicates whether to use the default certificate for your CloudFront domain name when viewers use HTTPS to request your content.

Required: Conditional. You must specify one of the following properties: `AcmCertificateArn`, `CloudFrontDefaultCertificate`, or `IamCertificateId`.

Type: Boolean

`IamCertificateId`

If you're using an alternate domain name, the ID of a server certificate that was purchased from a certificate authority. This ID is the `ServerCertificateId` value, which AWS Identity and Access Management (IAM) returns when the certificate is added to the IAM certificate store, such as `ASCACKCEVSQ6CEXAMPLE1`.

Required: Conditional. You must specify one of the following properties: `AcmCertificateArn`, `CloudFrontDefaultCertificate`, or `IamCertificateId`.

Type: String

`MinimumProtocolVersion`

The minimum version of the SSL protocol that you want CloudFront to use for HTTPS connections. CloudFront serves your objects only to browsers or devices that support at least the SSL version that you specify. For valid values, see the `MinimumProtocolVersion` content for the ViewerCertificate data type in the *Amazon CloudFront API Reference*.

AWS CloudFormation specifies `SSLv3` by default. However, if you specify the `IamCertificateId` or `AcmCertificateArn` property and specify SNI only for the `SslSupportMethod` property, AWS CloudFormation specifies `TLSv1` for the minimum protocol version.

Required: No

Type: String

`SslSupportMethod`

Specifies how CloudFront serves HTTPS requests. For valid values, see the `SslSupportMethod` content for the ViewerCertificate data type in the *Amazon CloudFront API Reference*.

Required: Conditional. Required if you specified the `IamCertificateId` or `AcmCertificateArn` property.

Type: String

CloudFront ForwardedValues

`ForwardedValues` is a property of the DefaultCacheBehavior (p. 1046) and CacheBehavior (p. 1042) properties that indicates whether Amazon CloudFront forwards query strings or cookies.

Syntax

JSON

```
{
  "Cookies" : Cookies,
  "Headers" : [ String, ... ],
  "QueryString" : Boolean,
  "QueryStringCacheKeys" : [ String, ... ]
}
```

YAML

```
Cookies:
  Cookies
Headers:
  - String
QueryString: Boolean
QueryStringCacheKeys:
  - String
```

Properties

Note
For more information about the constraints and valid values of each property, see the ForwardedValues data type in the *Amazon CloudFront API Reference*.

`Cookies`

Forwards specified cookies to the origin of the cache behavior. For more information, see Configuring CloudFront to Cache Based on Cookies in the *Amazon CloudFront Developer Guide*.

Required: No

Type: CloudFront ForwardedValues Cookies (p. 1058)

`Headers`

Specifies the headers that you want Amazon CloudFront to forward to the origin for this cache behavior (whitelisted headers). For the headers that you specify, Amazon CloudFront also caches separate versions of a specified object that is based on the header values in viewer requests.

For custom origins, if you specify a single asterisk (["*"]), all headers are forwarded. If you don't specify a value, only the default headers are forwarded. For Amazon S3 origins, you can forward only selected headers; specifying * is not supported. For more information, see Configuring CloudFront to Cache Objects Based on Request Headers in the *Amazon CloudFront Developer Guide*.

Required: No

Type: List of strings

`QueryString`

Indicates whether you want CloudFront to forward query strings to the origin that is associated with this cache behavior. If so, specify `true`; if not, specify `false`. For more information about forwarding query strings, see the `QueryString` parameter for the ForwardedValues type in the *Amazon CloudFront API Reference*.

Required: Yes

Type: Boolean

`QueryStringCacheKeys`

If you forward query strings to the origin, specifies the query string parameters that CloudFront uses to determine which content to cache. For more information, see Configuring CloudFront to Cache Based on Query String Parameters in the *Amazon CloudFront Developer Guide*.

Required: No

Type: List of strings

CloudFront ForwardedValues Cookies

`Cookies` is a property of the CloudFront ForwardedValues (p. 1057) property that describes which cookies are forwarded to the Amazon CloudFront origin.

Syntax

JSON

```
{
  "Forward" : String,
  "WhitelistedNames" : [ String, ... ]
}
```

YAML

```
Forward: String
WhitelistedNames:
  - String
```

Properties

Note
For more information about the constraints and valid values of each property, see the CookiePreference data type in the *Amazon CloudFront API Reference*.

`Forward`

The cookies to forward to the origin of the cache behavior. You can specify `none`, `all`, or `whitelist`.

Required: Yes

Type: String

`WhitelistedNames`

The names of cookies to forward to the origin for the cache behavior.

Required: Conditional. Required if you specified `whitelist` for the `Forward` property.

Type: List of strings

CloudWatch Metric Dimension Property Type

The Metric Dimension is an embedded property of the AWS::CloudWatch::Alarm (p. 498) type. Dimensions are arbitrary name/value pairs that can be associated with a CloudWatch metric. You can specify a maximum of 10 dimensions for a given metric.

Syntax

JSON

```
{
  "Name" : String,
  "Value" : String
}
```

YAML

```
Name: String
Value: String
```

Properties

`Name`

The name of the dimension, from 1–255 characters in length.

Required: Yes

Type: String

`Value`

The value representing the dimension measurement, from 1–255 characters in length.

Required: Yes

Type: String

Examples

Two CloudWatch alarms with dimension values supplied by the Ref function

The Ref (p. 1343) and Fn::GetAtt (p. 1324) intrinsic functions are often used to supply values for CloudWatch metric dimensions. Here is an example using the `Ref` function.

```

"CPUAlarmHigh": {
   "Type": "AWS::CloudWatch::Alarm",
   "Properties": {
      "AlarmDescription": "Scale-up if CPU is greater than 90% for 10 minutes",
```

```
      "MetricName": "CPUUtilization",
      "Namespace": "AWS/EC2",
      "Statistic": "Average",
      "Period": "300",
      "EvaluationPeriods": "2",
      "Threshold": "90",
      "AlarmActions": [ { "Ref": "WebServerScaleUpPolicy" } ],
      "Dimensions": [
         {
            "Name": "AutoScalingGroupName",
            "Value": { "Ref": "WebServerGroup" }
         }
      ],
      "ComparisonOperator": "GreaterThanThreshold"
   }
},
"CPUAlarmLow": {
   "Type": "AWS::CloudWatch::Alarm",
   "Properties": {
      "AlarmDescription": "Scale-down if CPU is less than 70% for 10 minutes",
      "MetricName": "CPUUtilization",
      "Namespace": "AWS/EC2",
      "Statistic": "Average",
      "Period": "300",
      "EvaluationPeriods": "2",
      "Threshold": "70",
      "AlarmActions": [ { "Ref": "WebServerScaleDownPolicy" } ],
      "Dimensions": [
         {
            "Name": "AutoScalingGroupName",
            "Value": { "Ref": "WebServerGroup" }
         }
      ],
      "ComparisonOperator": "LessThanThreshold"
   }
}
```

See Also

- Dimension in the *Amazon CloudWatch API Reference*
- Amazon CloudWatch Metrics, Namespaces, and Dimensions Reference in the *Amazon CloudWatch Developer Guide*

Amazon CloudWatch Events Rule Target

`Target` is a property of the AWS::Events::Rule (p. 761) resource that specifies the targets that CloudWatch Events invokes when a rule is triggered, such as AWS Lambda (Lambda) functions or Amazon Kinesis streams.

Syntax

JSON

```
{
  "Arn" : String,
  "Id" : String,
  "Input" : String,
  "InputPath" : String
}
```

YAML

```
Arn: String
Id: String
Input: String
InputPath: String
```

Properties

`Arn`

The Amazon Resource Name (ARN) of the target.

Required: Yes

Type: String

`Id`

A unique, user-defined identifier for the target. Acceptable values include alphanumeric characters, periods (.), hypens (-), and underscores (_).

Required: Yes

Type: String

`Input`

A JSON-formatted text string that is passed to the target. This value overrides the matched event.

Required: No. If you don't specify both this property and the `InputPath`, CloudWatch Events passes the entire matched event to the target.

Type: String

`InputPath`

When you don't want to pass the entire matched event, the JSONPath that describes which part of the event to pass to the target.

Required: No. If you don't specify both this property and the `Input`, CloudWatch Events passes the entire matched event to the target.

Type: String

CloudWatch Logs MetricFilter MetricTransformation Property

`MetricTransformation` is a property of the AWS::Logs::MetricFilter (p. 839) resource that describes how to transform log streams into a CloudWatch metric.

Syntax

JSON

```
{
  "MetricName": String,
```

```
  "MetricNamespace": String,
  "MetricValue": String
}
```

YAML

```
MetricName: String
MetricNamespace: String
MetricValue: String
```

Properties

> **Note**
> For more information about constraints and values for each property, see MetricTransformation in the *Amazon CloudWatch Logs API Reference*.

`MetricName`

The name of the CloudWatch metric to which the log information will be published.

Required: Yes

Type: String

`MetricNamespace`

The destination namespace of the CloudWatch metric. Namespaces are containers for metrics. For example, you can add related metrics in the same namespace.

Required: Yes

Type: String

`MetricValue`

The value that is published to the CloudWatch metric. For example, if you're counting the occurrences of a particular term like `Error`, specify `1` for the metric value. If you're counting the number of bytes transferred, reference the value that is in the log event by using `$` followed by the name of the field that you specified in the filter pattern, such as `$size`.

Required: Yes

Type: String

Examples

For samples of the `MetricTransformation` property, see AWS::Logs::MetricFilter (p. 839) or Amazon CloudWatch Logs Template Snippets (p. 257).

AWS CodeBuild Project Artifacts

`Artifacts` is a property of the AWS::CodeBuild::Project (p. 503) resource that specifies output settings for artifacts generated by an AWS CodeBuild build.

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Location" : String,
  "Name" : String,
  "NamespaceType" : String,
  "Packaging" : String,
  "Path" : String,
  "Type" : String
}
```

YAML

```
Location: String
Name: String
NamespaceType: String
Packaging: String
Path: String
Type: String
```

Properties

`Location`

The location where AWS CodeBuild saves the build output artifacts. For valid values, see the `artifacts-location` field in the *AWS CodeBuild User Guide*.

Required: Conditional. If you specify `CODEPIPELINE` or `NO_ARTIFACTS` for the `Type` property, don't specify this property. For all of the other types, you must specify this property.

Type: String

`Name`

The name of the build output folder where AWS CodeBuild saves the build output artifacts. For .zip packages, the name of the build output .zip file that contains the build output artifacts.

Required: Conditional. If you specify `CODEPIPELINE` or `NO_ARTIFACTS` for the `Type` property, don't specify this property. For all of the other types, you must specify this property.

Type: String

`NamespaceType`

The information AWS CodeBuild adds to the build output path, such as a build ID. For more information, see the `namespaceType` field in the *AWS CodeBuild User Guide*.

Required: No

Type: String

`Packaging`

Indicates how AWS CodeBuild packages the build output artifacts. For valid values, see the `packaging` field in the *AWS CodeBuild User Guide*.

Required: No

Type: String

`Path`

The path to the build output folder where AWS CodeBuild saves the build output artifacts.

Required: No

Type: String

`Type`

The type of build output artifact. For valid values, see the `artifacts-type` field in the *AWS CodeBuild User Guide*.

Required: Yes

Type: String

AWS CodeBuild Project Environment

`Environment` is a property of the AWS::CodeBuild::Project (p. 503) resource that specifies the environment for an AWS CodeBuild project.

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "ComputeType" : String,
  "EnvironmentVariables" : [ EnvironmentVariables (p. 1065) ],
  "Image" : String,
  "Type" : String
}
```

YAML

```
ComputeType: String
EnvironmentVariables:
  - EnvironmentVariables (p. 1065)
Image: String
Type: String
```

Properties

`ComputeType`

The type of compute environment, such as `BUILD_GENERAL1_SMALL`. The compute type determines the number of CPU cores and memory the build environment uses. For valid values, see the `computeType` field in the *AWS CodeBuild User Guide*.

Required: Yes

Type: String

`EnvironmentVariables`

The environment variables that your builds can use. For more information, see the `environmentVariables` field in the *AWS CodeBuild User Guide*.

Required: No

Type: List of AWS CodeBuild Project Environment EnvironmentVariables (p. 1065)

`Image`

The Docker image identifier that the build environment uses. For more information, see the `image` field in the *AWS CodeBuild User Guide*.

Required: Yes

Type: String

`Type`

The type of build environment. For valid values, see the `environment-type` field in the *AWS CodeBuild User Guide*.

Required: Yes

Type: String

AWS CodeBuild Project Environment EnvironmentVariables

`EnvironmentVariables` is a property of the AWS CodeBuild Project Environment (p. 1064) property that specifies the name and value of an environment variable for an AWS CodeBuild project environment. When you use the environment to run a build, these variables are available for your builds to use.

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Name" : String,
  "Value" : String
}
```

YAML

```
Name: String
Value: String
```

Properties

`Name`

The name of an environment variable.

Required: Yes

Type: String

`Value`

The value of the environment variable.

Required: Yes

Type: String

AWS CodeBuild Project Source

`Source` is a property of the AWS::CodeBuild::Project (p. 503) resource that specifies the source code settings for an AWS CodeBuild project.

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "BuildSpec" : String,
  "Location" : String,
  "Type" : String
}
```

YAML

```
BuildSpec: String
Location: String
Type: String
```

Properties

`BuildSpec`

The build specification, specified as a single string. For more information, see the `Build Spec Reference` in the *AWS CodeBuild User Guide*.

Required: No

Type: String

`Location`

The location of the source code in the specified repository type. For more information, see the `source-location` field in the *AWS CodeBuild User Guide*.

Required: Conditional. If you specify `CODEPIPELINE` for the `Type` property, don't specify this property. For all of the other types, you must specify this property.

Type: String

`Type`

The type of repository that contains your source code. For valid values, see the `source-type` field in the *AWS CodeBuild User Guide*.

Required: Yes

Type: String

AWS CodeCommit Repository Trigger

`Trigger` is a property of the AWS::CodeCommit::Repository (p. 507) resource that defines the actions to take in response to events that occur in the AWS CodeCommit repository.

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Branches" : [ String, ... ],
  "CustomData" : String,
  "DestinationArn" : String,
  "Events" : [ String, ... ],
  "Name" : String
}
```

YAML

```
Branches:
- String
CustomData: String
DestinationArn: String
Events:
- String
Name: String
```

Properties

`Branches`

The names of the branches in the AWS CodeCommit repository that contain events that you want to include in the trigger. If you don't specify at least one branch, the trigger applies to all branches.

Required: No

Type: List of strings

`CustomData`

When an event is triggered, additional information that AWS CodeCommit includes when it sends information to the target.

Required: No

Type: String

`DestinationArn`

The Amazon Resource Name (ARN) of the resource that is the target for this trigger. For valid targets, see Manage Triggers for an AWS CodeCommit Repository in the *AWS CodeCommit User Guide*.

Required: No

Type: String

`Events`

The repository events for which AWS CodeCommit sends information to the target, which you specified in the `DestinationArn` property. If you don't specify events, the trigger runs for all repository events. For valid values, see the RepositoryTrigger data type in the *AWS CodeCommit API Reference*.

Required: No

Type: List of strings

`Name`

A name for the trigger.

Required: Yes

Type: String

AWS CodeDeploy DeploymentConfig MinimumHealthyHosts

`MinimumHealthyHosts` is a property of the AWS::CodeDeploy::DeploymentConfig (p. 510) resource that defines how many instances must be healthy during an AWS CodeDeploy deployment.

Syntax

JSON

```
{
  "Type" : String,
  "Value" : Integer
}
```

YAML

```
Type: String
Value: Integer
```

Properties

`Type`

The type of count to use, such as an absolute value or a percentage of the total number of instances in the deployment. For valid values, see MinimumHealthyHosts in the *AWS CodeDeploy API Reference*.

Required: Yes

Type: String

`Value`

The minimum number of healthy instances.

Required: Yes

Type: Integer

AWS CodeDeploy DeploymentGroup Deployment

`Deployment` is a property of the AWS::CodeDeploy::DeploymentGroup (p. 512) resource that specifies the AWS CodeDeploy application revision that will be deployed to the deployment group.

Syntax

JSON

```
{
  "Description" : String,
  "IgnoreApplicationStopFailures" : Boolean,
  "Revision" : Revision
}
```

YAML

```
Description: String
IgnoreApplicationStopFailures: Boolean
Revision:
  Revision
```

Properties

`Description`

A description about this deployment.

Required: No

Type: String

`IgnoreApplicationStopFailures`

Whether to continue the deployment if the `ApplicationStop` deployment lifecycle event fails. If you want AWS CodeDeploy to continue the deployment lifecycle even if the `ApplicationStop` event fails on an instance, specify `true`. The deployment continues to the `BeforeInstall` deployment lifecycle event. If you want AWS CodeDeploy to stop deployment on the instance if the `ApplicationStop` event fails, specify `false` or do not specify a value.

Required: No

Type: Boolean

`Revision`

The location of the application revision to deploy.

Required: Yes

Type: AWS CodeDeploy DeploymentGroup Deployment Revision (p. 1069)

AWS CodeDeploy DeploymentGroup Deployment Revision

`Revision` is a property of the AWS::CodeDeploy::DeploymentGroup (p. 512) property that defines the location of the AWS CodeDeploy application revision to deploy.

Syntax

JSON

```
{
  "GitHubLocation" : GitHubLocation,
  "RevisionType" : String,
  "S3Location" : S3Location
}
```

YAML

```
GitHubLocation:
  GitHubLocation
RevisionType: String
S3Location:
  S3Location
```

Properties

`GitHubLocation`

If your application revision is stored in GitHub, information about the location where it is stored.

Required: No

Type: AWS CodeDeploy DeploymentGroup Deployment Revision GitHubLocation (p. 1070)

`RevisionType`

The application revision's location, such as in an S3 bucket or GitHub repository. For valid values, see RevisionLocation in the *AWS CodeDeploy API Reference*.

Required: No

Type: String

`S3Location`

If the application revision is stored in an S3 bucket, information about the location.

Required: No

Type: AWS CodeDeploy DeploymentGroup Deployment Revision S3Location (p. 1071)

AWS CodeDeploy DeploymentGroup Deployment Revision GitHubLocation

`GitHubLocation` is a property of the AWS CodeDeploy DeploymentGroup Deployment Revision (p. 1069) property that specifies the location of an application revision that is stored in GitHub.

Syntax

JSON

```
{
```

```
  "CommitId" : String,
  "Repository" : String
}
```

YAML

```
CommitId: String
Repository: String
```

Properties

`CommitId`

The SHA1 commit ID of the GitHub commit to use as your application revision.

Required: Yes

Type: String

`Repository`

The GitHub account and repository name that includes the application revision. Specify the value as *account*/*repository_name*.

Required: Yes

Type: String

AWS CodeDeploy DeploymentGroup Deployment Revision S3Location

`S3Location` is a property of the AWS CodeDeploy DeploymentGroup Deployment Revision (p. 1069) property that specifies the location of an application revision that is stored in Amazon Simple Storage Service (Amazon S3).

Syntax

JSON

```
{
  "Bucket" : String,
  "BundleType" : String,
  "ETag" : String,
  "Key" : String,
  "Version" : String
}
```

YAML

```
Bucket: String
BundleType: String
ETag: String
Key: String
Version: String
```

Properties

`Bucket`

The name of the S3 bucket where the application revision is stored.

Required: Yes

Type: String

`BundleType`

The file type of the application revision, such as `tar`, `tgz`, or `zip`. For valid values, see S3Location in the *AWS CodeDeploy API Reference*.

Required: Yes

Type: String

`ETag`

The Amazon S3 ETag (a file checksum) of the application revision. If you don't specify a value, AWS CodeDeploy skips the ETag validation of your application revision.

Required: No

Type: String

`Key`

The file name of the application revision (Amazon S3 object name).

Required: Yes

Type: String

`Version`

For versioning-enabled buckets, a specific version of the application revision.

Required: No

Type: String

AWS CodeDeploy DeploymentGroup Ec2TagFilters

`Ec2TagFilters` is a property of the AWS::CodeDeploy::DeploymentGroup (p. 512) resource that specifies which EC2 instances to associate with the deployment group.

Syntax

JSON

```
{
  "Key" : String,
  "Type" : String,
  "Value" : String
}
```

YAML

```
Key: String
Type: String
Value: String
```

Properties

`Key`

Filter instances with this key.

Required: No

Type: String

`Type`

The filter type. For example, you can filter instances by the key, tag value, or both. For valid values, see EC2TagFilter in the *AWS CodeDeploy API Reference*.

Required: Yes

Type: String

`Value`

Filter instances with this tag value.

Required: No

Type: String

AWS CodeDeploy DeploymentGroup OnPremisesInstanceTagFilters

`OnPremisesInstanceTagFilters` is a property of the AWS::CodeDeploy::DeploymentGroup (p. 512) resource that specifies which on-premises instances to associate with the deployment group. To register on-premise instances with AWS CodeDeploy, see Configure Existing On-Premises Instances by Using AWS CodeDeploy in the *AWS CodeDeploy User Guide*.

Syntax

JSON

```
{
  "Key" : String,
  "Type" : String,
  "Value" : String
}
```

YAML

```
Key: String
Type: String
```

```
Value: String
```

Properties

`Key`

Filter on-premises instances with this key.

Required: No

Type: String

`Type`

The filter type. For example, you can filter on-premises instances by the key, tag value, or both. For valid values, see EC2TagFilter in the *AWS CodeDeploy API Reference*.

Required: No

Type: String

`Value`

Filter on-premises instances with this tag value.

Required: No

Type: String

AWS CodePipeline CustomActionType ArtifactDetails

`ArtifactDetails` is a property of the AWS::CodePipeline::CustomActionType (p. 517) resource that specifies the details of an artifact for an AWS CodePipeline custom action. For valid values, see ArtifactDetails in the *AWS CodePipeline API Reference*.

Syntax

JSON

```
{
  "MaximumCount" : Integer,
  "MinimumCount" : Integer
}
```

Yaml

```
MaximumCount: Integer
MinimumCount: Integer
```

Properties

`MaximumCount`

The maximum number of artifacts allowed for the action type.

Required: Yes

Type: Integer

`MinimumCount`

The minimum number of artifacts allowed for the action type.

Required: Yes

Type: Integer

AWS CodePipeline CustomActionType ConfigurationProperties

`ConfigurationProperties` is a property of the AWS::CodePipeline::CustomActionType (p. 517) resource that defines a configuration for an AWS CodePipeline custom action.

Syntax

JSON

```
{
  "Description" : String,
  "Key" : Boolean,
  "Name" : String,
  "Queryable" : Boolean,
  "Required" : Boolean,
  "Secret" : Boolean,
  "Type" : String
}
```

YAML

```
Description: String
Key: Boolean
Name: String
Queryable: Boolean
Required: Boolean
Secret: Boolean
Type: String
```

Properties

`Description`

A description of this configuration property that will be displayed to users.

Required: No

Type: String

`Key`

Indicates whether the configuration property is a key.

Required: Yes

Type: Boolean

`Name`

A name for this configuration property.

Required: Yes

Type: String

`Queryable`

Indicates whether the configuration property will be used with the `PollForJobs` call. A custom action can have one queryable property. The queryable property must be required (see the `Required` property) and must not be secret (see the `Secret` property). For more information, see the `queryable` contents for the ActionConfigurationProperty data type in the *AWS CodePipeline API Reference*.

Required: No

Type: Boolean

`Required`

Indicates whether the configuration property is a required value.

Required: Yes

Type: Boolean

`Secret`

Indicates whether the configuration property is secret. Secret configuration properties are hidden from all AWS CodePipeline calls except for `GetJobDetails`, `GetThirdPartyJobDetails`, `PollForJobs`, and `PollForThirdPartyJobs`.

Required: Yes

Type: Boolean

`Type`

The type of the configuration property, such as `String`, `Number`, or `Boolean`.

Required: No

Type: String

AWS CodePipeline CustomActionType Settings

`Settings` is a property of the AWS::CodePipeline::CustomActionType (p. 517) resource that provides URLs that users can access to view information about the AWS CodePipeline custom action.

Syntax

JSON

```
{
  "EntityUrlTemplate" : String,
  "ExecutionUrlTemplate" : String,
  "RevisionUrlTemplate" : String,
  "ThirdPartyConfigurationUrl" : String
}
```

YAML

```
EntityUrlTemplate: String
ExecutionUrlTemplate: String
RevisionUrlTemplate: String
ThirdPartyConfigurationUrl: String
```

Properties

`EntityUrlTemplate`

The URL that is returned to the AWS CodePipeline console that links to the resources of the external system, such as the configuration page for an AWS CodeDeploy deployment group.

Required: No

Type: String

`ExecutionUrlTemplate`

The URL that is returned to the AWS CodePipeline console that links to the top-level landing page for the external system, such as the console page for AWS CodeDeploy.

Required: No

Type: String

`RevisionUrlTemplate`

The URL that is returned to the AWS CodePipeline console that links to the page where customers can update or change the configuration of the external action.

Required: No

Type: String

`ThirdPartyConfigurationUrl`

The URL of a sign-up page where users can sign up for an external service and specify the initial configurations for the service's action.

Required: No

Type: String

AWS CodePipeline Pipeline ArtifactStore

`ArtifactStore` is a property of the AWS::CodePipeline::Pipeline (p. 520) resource that defines the S3 location where AWS CodePipeline stores pipeline artifacts.

Syntax

JSON

```
{
  "EncryptionKey" : EncryptionKey,
  "Location" : String,
  "Type" : String
```

```
}
```

YAML

```
EncryptionKey: EncryptionKey
Location: String
Type: String
```

Properties

`EncryptionKey`

The encryption key AWS CodePipeline uses to encrypt the data in the artifact store, such as an AWS Key Management Service (AWS KMS) key. If you don't specify a key, AWS CodePipeline uses the default key for Amazon Simple Storage Service (Amazon S3).

Required: No

Type: AWS CodePipeline Pipeline ArtifactStore EncryptionKey (p. 1078)

`Location`

The location where AWS CodePipeline stores artifacts for a pipeline, such as an S3 bucket.

Required: Yes

Type: String

`Type`

The type of the artifact store, such as Amazon S3. For valid values, see ArtifactStore in the *AWS CodePipeline API Reference*.

Required: Yes

Type: String

AWS CodePipeline Pipeline ArtifactStore EncryptionKey

`EncryptionKey` is a property of the AWS CodePipeline Pipeline ArtifactStore (p. 1077) property that specifies which key AWS CodePipeline uses to encrypt data in the artifact store, such as an AWS Key Management Service (AWS KMS) key.

Syntax

JSON

```
{
  "Id" : String,
  "Type" : String
}
```

YAML

```
Id: String
```

```
Type: String
```

Properties

`Id`

The ID of the key. For an AWS KMS key, specify the key ID or key Amazon Resource Number (ARN).

Required: Yes

Type: String

`Type`

The type of encryption key, such as `KMS`. For valid values, see EncryptionKey in the *AWS CodePipeline API Reference*.

Required: Yes

Type: String

AWS CodePipeline Pipeline DisableInboundStageTransitions

`DisableInboundStageTransitions` is a property of the AWS::CodePipeline::Pipeline (p. 520) resource that specifies which AWS CodePipeline stage to disable transitions to.

Syntax

JSON

```
{
  "Reason" : String,
  "StageName" : String
}
```

YAML

```
Reason: String
StageName: String
```

Properties

`Reason`

An explanation of why the transition between two stages of a pipeline was disabled.

Required: Yes

Type: String

`StageName`

The name of the stage to which transitions are disabled.

Required: Yes

Type: String

AWS CodePipeline Pipeline Stages

`Stages` is a property of the AWS::CodePipeline::Pipeline (p. 520) resource that specifies a sequence of tasks for AWS CodePipeline to complete on an artifact.

Syntax

JSON

```
{
  "Actions" : [ Actions, ... ],
  "Blockers" : [ Blockers, ... ],
  "Name" : String
}
```

YAML

```
Actions:
  - Actions
Blockers:
  - Blockers
Name: String
```

Properties

`Actions`

The actions to include in this stage.

Required: Yes

Type: List of AWS CodePipeline Pipeline Stages Actions (p. 1080)

`Blockers`

The gates included in a stage.

Required: No

Type: List of AWS CodePipeline Pipeline Stages Blockers (p. 1084)

`Name`

A name for this stage.

Required: Yes

Type: String

AWS CodePipeline Pipeline Stages Actions

`Actions` is a property of the AWS CodePipeline Pipeline Stages (p. 1080) property that specifies an action for an AWS CodePipeline stage.

Syntax

JSON

```
{
  "ActionTypeId" : ActionTypeID,
  "Configuration" : { Key : Value },
  "InputArtifacts" : [ InputArtifacts, ... ],
  "Name" : String,
  "OutputArtifacts" : [ OutputArtifacts, ... ],
  "RoleArn" : String,
  "RunOrder" : Integer
}
```

YAML

```
ActionTypeId:
  ActionTypeID
Configuration:
  Key : Value
InputArtifacts:
  - InputArtifacts
Name: String
OutputArtifacts:
  - OutputArtifacts
RoleArn: String
RunOrder: Integer
```

Properties

`ActionTypeId`

Specifies the action type and the provider of the action.

Required: Yes

Type: AWS CodePipeline Pipeline Stages Actions ActionTypeId (p. 1082)

`Configuration`

The action's configuration. These are key-value pairs that specify input values for an action.

Required: No

Type: JSON object

`InputArtifacts`

The name or ID of the artifact that the action consumes, such as a test or build artifact.

Required: No

Type: List of AWS CodePipeline Pipeline Stages Actions InputArtifacts (p. 1083)

`Name`

The action name.

Required: Yes

Type: String

`OutputArtifacts`

The artifact name or ID that is a result of the action, such as a test or build artifact.

Required: No

Type: List of AWS CodePipeline Pipeline Stages Actions OutputArtifacts (p. 1084)

`RoleArn`

The Amazon Resource Name (ARN) of a service role that the action uses. The pipeline's role assumes this role.

Required: No

Type: String

`RunOrder`

The order in which AWS CodePipeline runs this action.

Required: No

Type: Integer

AWS CodePipeline Pipeline Stages Actions ActionTypeId

`ActionTypeId` is a property of the AWS CodePipeline Pipeline Stages Actions (p. 1080) property that specifies the action type and provider for an AWS CodePipeline action.

Syntax

JSON

```
{
  "Category" : String,
  "Owner" : String,
  "Provider" : String,
  "Version" : String
}
```

YAML

```
Category: String
Owner: String
Provider: String
Version: String
```

Properties

`Category`

A category that defines which action type the owner (the entitiy that performs the action) performs. The category that you select determine the providers that you can specify for the `Provider` property. For valid values, see ActionTypeId in the *AWS CodePipeline API Reference*.

Required: Yes

Type: String

`Owner`

The entity that performs the action. For valid values, see ActionTypeId in the *AWS CodePipeline API Reference*.

Required: Yes

Type: String

`Provider`

The service provider that the action calls. The providers that you can specify are determined by the category that you select. For example, a valid provider for the `Deploy` category is AWS CodeDeploy, which you would specify as `CodeDeploy`.

Required: Yes

Type: String

`Version`

A version identifier for this action.

Required: Yes

Type: String

AWS CodePipeline Pipeline Stages Actions InputArtifacts

`InputArtifacts` is a property of the AWS CodePipeline Pipeline Stages Actions (p. 1080) property that specifies an artifact that the AWS CodePipeline action works on, such as a test or build artifact.

Syntax

JSON

```
{
  "Name" : String
}
```

YAML

```
Name: String
```

Properties

`Name`

The name of the artifact that the AWS CodePipeline action works on, such as `My App`.The input artifact of an action must match the output artifact from any preceding action.

Required: Yes

Type: String

AWS CodePipeline Pipeline Stages Actions OutputArtifacts

`OutputArtifacts` is a property of the AWS CodePipeline Pipeline Stages Actions (p. 1080) property that specifies an artifact that is the result of an AWS CodePipeline action, such as a test or build artifact.

Syntax

JSON

```
{
  "Name" : String
}
```

YAML

```
Name: String
```

Properties

`Name`

The name of the artifact that is the result of an AWS CodePipeline action, such as `My App`. Output artifact names must be unique within a pipeline.

Required: Yes

Type: String

AWS CodePipeline Pipeline Stages Blockers

`Blockers` is a property of the AWS CodePipeline Pipeline Stages (p. 1080) property that specifies an AWS CodePipeline gate declaration.

Syntax

JSON

```
{
  "Name" : String,
  "Type" : String
}
```

YAML

```
Name: String
Type: String
```

Properties

`Name`

The name of the gate declaration.

Required: Yes

Type: String

`Type`

The type of gate declaration. For valid values, see BlockerDeclaration in the *AWS CodePipeline API Reference*.

Required: Yes

Type: String

AWS Config ConfigRule Scope

`Scope` is a property of the AWS::Config::ConfigRule (p. 525) resource that specifies which AWS resources will trigger AWS Config to run an evaluation when their configurations change. The scope can include one or more resource types, a tag key and value, or one resource type and one resource ID. You cannot specify a tag-key value and a resource ID or type.

Syntax

JSON

```
{
  "ComplianceResourceId" : String,
  "ComplianceResourceTypes" : [ String, ... ],
  "TagKey" : String,
  "TagValue" : String
}
```

YAML

```
ComplianceResourceId: String
ComplianceResourceTypes:
  - String
TagKey: String
TagValue: String
```

Properties

`ComplianceResourceId`

The ID of an AWS resource that you want AWS Config to evaluate against a rule. If you specify an ID, you must also specify a resource type for the `ComplianceResourceTypes` property.

Required: No

Type: String

`ComplianceResourceTypes`

The types of AWS resources that you want AWS Config to evaluate against the rule. If you specify the `ComplianceResourceId` property, specify only one resource type.

Required: Conditional. If you specify a value for the `ComplianceResourceId` property, you must also specify this property.

Type: List of strings

`TagKey`

The tag key that is applied to the AWS resources that you want AWS Config to evaluate against the rule.

Required: Conditional. If you specify a tag value, you must specify this property.

Type: String

`TagValue`

The tag value that is applied to the AWS resources that you want AWS Config to evaluate against the rule.

Required: Conditional. If you specify a tag key, you must specify this property.

Type: String

AWS Config ConfigRule Source

`Source` is a property of the AWS::Config::ConfigRule (p. 525) resource that specifies the rule owner, the rule identifier, and the events that trigger an AWS Config evaluation of your AWS resources.

Syntax

JSON

```
{
  "Owner" : String,
  "SourceDetails" : [ SourceDetail, ... ],
  "SourceIdentifier" : String
}
```

YAML

```
Owner: String
SourceDetails:
  - SourceDetail
SourceIdentifier: String
```

Properties

`Owner`

Indicates who owns and manages the AWS Config rule. For valid values, see the Source data type in the *AWS Config API Reference*.

Required: Yes

Type: String

`SourceDetails`

Provides the source and type of event that triggers AWS Config to evaluate your AWS resources.

Required: No

Type: List of AWS Config ConfigRule Source SourceDetails (p. 1087)

`SourceIdentifier`

For AWS managed rules, the identifier of the rule. For a list of identifiers, see AWS Managed Rules in the *AWS Config Developer Guide*.

For customer managed rules, the Amazon Resource Name (ARN) of the rule's Lambda function.

Required: Yes

Type: String

AWS Config ConfigRule Source SourceDetails

`SourceDetails` is a property of the AWS Config ConfigRule Source (p. 1086) property that specifies the source and type of event that triggers AWS Config to evaluate your AWS resources.

Syntax

JSON

```
{
  "EventSource" : String,
  "MessageType" : String
}
```

YAML

```
EventSource: String
MessageType: String
```

Properties

`EventSource`

The source, such as an AWS service, that generate events, triggering AWS Config to evaluate your AWS resources. For valid values, see the SourceDetail data type in the *AWS Config API Reference*.

Required: Yes

Type: String

`MessageType`

The type of Amazon Simple Notification Service (Amazon SNS) message that triggers AWS Config to run an evaluation.

To run an evaluation when AWS Config delivers a configuration item change notification, specify `ConfigurationItemChangeNotification`.

To run an evaluation when AWS Config delivers a configuration snapshot, specify `ConfigurationSnapshotDeliveryCompleted`.

Required: Yes

Type: String

AWS Config ConfigurationRecorder RecordingGroup

`RecordingGroup` is property of the AWS::Config::ConfigurationRecorder (p. 531) resource that defines which AWS resource types to include in a recording group.

Syntax

JSON

```
{
  "AllSupported" : Boolean,
  "IncludeGlobalResourceTypes" : Boolean,
  "ResourceTypes" : [ String, ... ]
}
```

YAML

```
AllSupported: Boolean
IncludeGlobalResourceTypes: Boolean
ResourceTypes:
  - String
```

Properties

`AllSupported`

Indicates whether to record all supported resource types. If you specify this property, do not specify the `ResourceTypes` property.

Required: No

Type: Boolean

`IncludeGlobalResourceTypes`

Indicates whether AWS Config records all supported global resource types. When AWS Config supports new global resource types, AWS Config will automatically start recording them if you enable this property.

Note
If you set this property to `true`, you must set the `AllSupported` property to `true`.

Required: No

Type: Boolean

`ResourceTypes`

A list of valid AWS resource types to include in this recording group, such as **`AWS::EC2::Instance`** or **`AWS::CloudTrail::Trail`**. If you specify this property, do not specify the `AllSupported` property. For a list of supported resource types, see Supported resource types in the *AWS Config Developer Guide*.

Required: No

Type: List of strings

AWS Config DeliveryChannel ConfigSnapshotDeliveryProperties

`ConfigSnapshotDeliveryProperties` is a property of the AWS::Config::DeliveryChannel (p. 533) resource that specifies how AWS Config delivers configuration snapshots to the S3 bucket in your delivery channel.

Syntax

JSON

```
{
  "DeliveryFrequency" : String
}
```

YAML

```
DeliveryFrequency: String
```

Properties

`DeliveryFrequency`

The frequency with which AWS Config delivers configuration snapshots. For valid values, see ConfigSnapshotDeliveryProperties in the *AWS Config API Reference*.

Required: No

Type: String

AWS Data Pipeline Pipeline ParameterObjects

`ParameterObjects` is a property of the AWS::DataPipeline::Pipeline (p. 535) resource that describes parameters that are used in a pipeline definition.

Syntax

JSON

```
{
  "Attributes" : [ Attribute, ... ],
  "Id" : String
}
```

YAML

```
Attributes:
```

```
  - Attribute
Id: String
```

Properties

`Attributes`

Key-value pairs that define the attributes of the parameter object.

Required: Yes

Type: AWS Data Pipeline Parameter Objects Attributes (p. 1090)

`Id`

The identifier of the parameter object.

Required: Yes

Type: String

AWS Data Pipeline Parameter Objects Attributes

`Attribute` is a property of the AWS Data Pipeline Pipeline ParameterObjects (p. 1089) property that defines the attributes of a parameter object as key-value pairs.

Syntax

JSON

```
{
  "Key" : String,
  "StringValue" : String
}
```

YAML

```
Key: String
StringValue: String
```

Properties

`Key`

Specifies the name of a parameter attribute. To view parameter attributes, see Creating a Pipeline Using Parameterized Templates in the *AWS Data Pipeline Developer Guide*.

Required: Yes

Type: String

`StringValue`

A parameter attribute value.

Required: Conditional if the key that you are using requires it.

Type: String

AWS Data Pipeline Pipeline ParameterValues

`ParameterValues` is a property of the AWS::DataPipeline::Pipeline (p. 535) resource that sets values for parameters that are used in a pipeline definition.

Syntax

JSON

```
{
  "Id" : String,
  "StringValue" : String
}
```

YAML

```
Id: String
StringValue: String
```

Properties

`Id`

The ID of a parameter object.

Required: Yes

Type: String

`StringValue`

A value to associate with the parameter object.

Required: Yes

Type: String

AWS Data Pipeline PipelineObjects

`PipelineObjects` is a property of the AWS::DataPipeline::Pipeline (p. 535) resource that describes a data pipeline object.

Syntax

JSON

```
{
  "Fields" : [ Field type ],
  "Id" : String,
  "Name" : String
}
```

YAML

```
Fields:
```

```
  - Field type
Id: String
Name: String
```

Properties

`Fields`

Key-value pairs that define the properties of the object.

Required: Yes

Type: AWS Data Pipeline Data Pipeline Object Fields (p. 1092)

`Id`

Identifier of the object.

Required: Yes

Type: String

`Name`

Name of the object.

Required: Yes

Type: String

AWS Data Pipeline Data Pipeline Object Fields

Key-value pairs that describe the properties of a data pipeline object (p. 1091).

Syntax

JSON

```
{
  "Key" : String,
  "RefValue" : String,
  "StringValue" : String
}
```

YAML

```
Key: String
RefValue: String
StringValue: String
```

Properties

`Key`

Specifies the name of a field for a particular object. To view fields for a data pipeline object, see Pipeline Object Reference in the *AWS Data Pipeline Developer Guide*.

Required: Yes

Type: String

`RefValue`

A field value that you specify as an identifier of another object in the same pipeline definition.

Note
You can specify the field value as either a string value (`StringValue`) or a reference to another object (`RefValue`), but not both.

Required: Conditional if the key that you are using requires it.

Type: String

`StringValue`

A field value that you specify as a string. To view valid values for a particular field, see Pipeline Object Reference in the *AWS Data Pipeline Developer Guide*.

Note
You can specify the field value as either a string value (`StringValue`) or a reference to another object (`RefValue`), but not both.

Required: Conditional if the key that you are using requires it.

Type: String

AWS Data Pipeline Pipeline PipelineTags

`PipelineTags` is a property of the AWS::DataPipeline::Pipeline (p. 535) resource that defines arbitrary key-value pairs for a pipeline.

Syntax

JSON

```
{
  "Key" : String,
  "Value" : String
}
```

YAML

```
Key: String
Value: String
```

Properties

`Key`

The key name of a tag.

Required: Yes

Type: String

`Value`

The value to associate with the key name.

Required: Yes

Type: String

AWS Directory Service MicrosoftAD VpcSettings

`VpcSettings` is a property of the AWS::DirectoryService::MicrosoftAD (p. 544) resource that specifies the VPC settings for a Microsoft directory server.

Syntax

JSON

```
{
  "SubnetIds" : [ String, ... ],
  "VpcId" : String
}
```

YAML

```
SubnetIds:
  - String
VpcId: String
```

Properties

`SubnetIds`

A list of two subnet IDs for the directory servers. Each subnet must be in different Availability Zones (AZs). AWS Directory Service creates a directory server and a DNS server in each subnet.

Required: Yes

Type: List of strings

`VpcId`

The VPC ID in which to create the Microsoft Active Directory server.

Required: Yes

Type: String

AWS Directory Service SimpleAD VpcSettings

`VpcSettings` is a property of the AWS::DirectoryService::SimpleAD (p. 547) resource that specifies the VPC settings for a directory server.

Syntax

JSON

```
{
```

```
  "SubnetIds" : [ String, ... ],
  "VpcId" : String
}
```

YAML

```
SubnetIds:
  - String
VpcId: String
```

Properties

`SubnetIds`

A list of two subnet IDs for the directory servers. Each subnet must be in different Availability Zones (AZ). AWS Directory Service creates a directory server and a DNS server in each subnet.

Required: Yes

Type: List of strings

`VpcId`

The VPC ID in which to create the Simple AD directory.

Required: Yes

Type: String

DynamoDB Attribute Definitions

A list of attribute definitions for the AWS::DynamoDB::Table (p. 550) resource. Each element is composed of an `AttributeName` and `AttributeType`.

Syntax

JSON

```
{
  "AttributeName" : String,
  "AttributeType" : String
}
```

YAML

```
AttributeName: String
AttributeType: String
```

Properties

`AttributeName`

The name of an attribute. Attribute names can be 1 – 255 characters long and have no character restrictions.

Required: Yes

Type: String

`AttributeType`

The data type for the attribute. You can specify `S` for string data, `N` for numeric data, or `B` for binary data.

Required: Yes

Type: String

DynamoDB Global Secondary Indexes

Describes global secondary indexes for the AWS::DynamoDB::Table (p. 550) resource.

Syntax

JSON

```
{
  "IndexName" : String,
  "KeySchema" : [ KeySchema, ... ],
  "Projection" : { Projection },
  "ProvisionedThroughput" : { ProvisionedThroughput }
}
```

YAML

```
IndexName: String
KeySchema:
  - KeySchema
Projection:
  Projection
ProvisionedThroughput:
  ProvisionedThroughput
```

Properties

`IndexName`

The name of the global secondary index. The index name can be 3 – 255 characters long and have no character restrictions.

Required: Yes

Type: String

`KeySchema`

The complete index key schema for the global secondary index, which consists of one or more pairs of attribute names and key types.

Required: Yes

Type: DynamoDB Key Schema (p. 1097)

`Projection`

Attributes that are copied (projected) from the source table into the index. These attributes are in addition to the primary key attributes and index key attributes, which are automatically projected.

Required: Yes

Type: DynamoDB Projection Object (p. 1099)

`ProvisionedThroughput`

The provisioned throughput settings for the index.

Required: Yes

Type: DynamoDB Provisioned Throughput (p. 1100)

DynamoDB Key Schema

Describes a primary key for the AWS::DynamoDB::Table (p. 550) resource or a key schema for an index. Each element is composed of an `AttributeName` and `KeyType`.

For the primary key of an Amazon DynamoDB table that consists of only a hash attribute, specify one element with a `KeyType` of `HASH`. For the primary key of an Amazon DynamoDB table that consists of a hash and range attributes, specify two elements: one with a `KeyType` of `HASH` and one with a `KeyType` of `RANGE`.

For a complete discussion of DynamoDB primary keys, see Primary Key in the *Amazon DynamoDB Developer Guide*.

Syntax

JSON

```
{
  "AttributeName" : String,
  "KeyType" : "HASH or RANGE"
}
```

YAML

```
AttributeName: String
KeyType: HASH or RANGE
```

Properties

`AttributeName`

The attribute name that is used as the primary key for this table. Primary key element names can be 1 – 255 characters long and have no character restrictions.

Required: Yes

Type: String

`KeyType`

Represents the attribute data, consisting of the data type and the attribute value itself. You can specify `HASH` or `RANGE`.

Required: Yes

Type: String

Examples

For an example of a declared key schema, see AWS::DynamoDB::Table (p. 550).

DynamoDB Local Secondary Indexes

Describes local secondary indexes for the AWS::DynamoDB::Table (p. 550) resource. Each index is scoped to a given hash key value. Tables with one or more local secondary indexes are subject to an item collection size limit, where the amount of data within a given item collection cannot exceed 10 GB.

Syntax

JSON

```
{
  "IndexName" : String,
  "KeySchema" : [ KeySchema, ...],
  "Projection" : { Projection }
}
```

YAML

```
IndexName: String
KeySchema:
  KeySchema
Projection:
  Projection
```

Properties

`IndexName`

The name of the local secondary index. The index name can be 3 – 255 characters long and have no character restrictions.

Required: Yes

Type: String

`KeySchema`

The complete index key schema for the local secondary index, which consists of one or more pairs of attribute names and key types. For local secondary indexes, the hash key must be the same as that of the source table.

Required: Yes

Type: DynamoDB Key Schema (p. 1097)

`Projection`

Attributes that are copied (projected) from the source table into the index. These attributes are additions to the primary key attributes and index key attributes, which are automatically projected.

Required: Yes

Type: DynamoDB Projection Object (p. 1099)

Examples

For an example of a declared local secondary index, see AWS::DynamoDB::Table (p. 550).

DynamoDB Projection Object

Attributes that are copied (projected) from the source table into the index. These attributes are additions to the primary key attributes and index key attributes, which are automatically projected.

Syntax

JSON

```
{
  "NonKeyAttributes" : [ String, ... ],
  "ProjectionType" : String
}
```

YAML

```
NonKeyAttributes:
  - String
ProjectionType: String
```

Properties

`NonKeyAttributes`

The non-key attribute names that are projected into the index.

For local secondary indexes, the total count of `NonKeyAttributes` summed across all of the local secondary indexes must not exceed 20. If you project the same attribute into two different indexes, this counts as two distinct attributes in determining the total.

Required: No

Type: List of strings

`ProjectionType`

The set of attributes that are projected into the index:

`KEYS_ONLY`

Only the index and primary keys are projected into the index.

`INCLUDE`

Only the specified table attributes are projected into the index. The list of projected attributes are in `NonKeyAttributes`.

`ALL`

All of the table attributes are projected into the index.

Required: No

Type: String

DynamoDB Provisioned Throughput

Describes a set of provisioned throughput values for an AWS::DynamoDB::Table (p. 550) resource. DynamoDB uses these capacity units to allocate sufficient resources to provide the requested throughput.

For a complete discussion of DynamoDB provisioned throughput values, see Specifying Read and Write Requirements in the *DynamoDB Developer Guide*.

Syntax

JSON

```
{
   "ReadCapacityUnits (p. 1100)" : Number,
   "WriteCapacityUnits (p. 1100)" : Number
}
```

YAML

```
ReadCapacityUnits (p. 1100): Number
WriteCapacityUnits (p. 1100): Number
```

Parameters

`ReadCapacityUnits`

Sets the desired minimum number of consistent reads of items (up to 1KB in size) per second for the specified table before Amazon DynamoDB balances the load.

Required: Yes

Type: Number

`WriteCapacityUnits`

Sets the desired minimum number of consistent writes of items (up to 1KB in size) per second for the specified table before Amazon DynamoDB balances the load.

Required: Yes

Type: Number

Note

For detailed information about the limits of provisioned throughput values in DynamoDB, see Limits in Amazon DynamoDB in the *DynamoDB Developer Guide*.

DynamoDB Table StreamSpecification

`StreamSpecification` is a property of the AWS::DynamoDB::Table (p. 550) resource that defines the settings of a DynamoDB table's stream.

Syntax

JSON

```
{
  "StreamViewType" : String
}
```

YAML

```
StreamViewType: String
```

Parameters

`StreamViewType`

Determines the information that the stream captures when an item in the table is modified. For valid values, see StreamSpecification in the *Amazon DynamoDB API Reference*.

Required: Yes

Type: String

Amazon EC2 Block Device Mapping Property

The Amazon EC2 block device mapping property is an embedded property of the AWS::EC2::Instance (p. 574) resource. For block device mappings for an Auto Scaling launch configuration, see AutoScaling Block Device Mapping (p. 1029).

Syntax

JSON

```
{
  "DeviceName (p. 1101)" : String,
  "Ebs (p. 1102)" : EC2 EBS Block Device,
  "NoDevice (p. 1102)" : {},
  "VirtualName (p. 1102)" : String
}
```

YAML

```
DeviceName (p. 1101): String
Ebs (p. 1102):
  EC2 EBS Block Device
NoDevice (p. 1102): {}
VirtualName (p. 1102): String
```

Properties

`DeviceName`

The name of the device within Amazon EC2.

Required: Yes

Type: String

`Ebs`

Required: Conditional You can specify either `VirtualName` or `Ebs`, but not both.

Type: Amazon Elastic Block Store Block Device Property (p. 1103).

`NoDevice`

This property can be used to unmap a defined device.

Required: No

Type: an empty map: {}.

`VirtualName`

The name of the virtual device. The name must be in the form `ephemeralX` where *X* is a number starting from zero (0); for example, `ephemeral0`.

Required: Conditional You can specify either `VirtualName` or `Ebs`, but not both.

Type: String

Examples

Block Device Mapping with two EBS Volumes

This example sets the EBS-backed root device (/dev/sda1) size to 50 GiB, and another EBS-backed device mapped to /dev/sdm that is 100 GiB in size.

```
"BlockDeviceMappings" : [
   {
      "DeviceName" : "/dev/sda1",
      "Ebs" : { "VolumeSize" : "50" }
   },
   {
      "DeviceName" : "/dev/sdm",
      "Ebs" : { "VolumeSize" : "100" }
   }
]
```

Block Device Mapping with an Ephemeral Drive

This example maps an ephemeral drive to device /dev/sdc.

```
"BlockDeviceMappings" : [
   {
      "DeviceName"  : "/dev/sdc",
      "VirtualName" : "ephemeral0"
   }
]
```

Unmapping an AMI-defined Device

To unmap a device defined in the AMI, set the `NoDevice` property to an empty map, as shown here:

```
{
   "DeviceName":"/dev/sde",
   "NoDevice": {}
}
```

See Also

- Amazon EC2 Instance Store in the *Amazon Elastic Compute Cloud User Guide*

Amazon Elastic Block Store Block Device Property

The Amazon Elastic Block Store block device type is an embedded property of the Amazon EC2 Block Device Mapping Property (p. 1101) property.

Syntax

JSON

```
{
   "DeleteOnTermination (p. 1103)" : Boolean,
   "Encrypted" : Boolean,
   "Iops (p. 1104)" : Number,
   "SnapshotId (p. 1104)" : String,
   "VolumeSize (p. 1104)" : String,
   "VolumeType (p. 1104)" : String
}
```

YAML

```
DeleteOnTermination (p. 1103): Boolean
Encrypted: Boolean
Iops (p. 1104): Number
SnapshotId (p. 1104): String
VolumeSize (p. 1104): String
VolumeType (p. 1104): String
```

Properties

`DeleteOnTermination`

Determines whether to delete the volume on instance termination. The default value is `true`.

Required: No

Type: Boolean

`Encrypted`

Indicates whether the volume is encrypted. Encrypted Amazon EBS volumes can only be attached to instance types that support Amazon EBS encryption. Volumes that are created from encrypted snapshots are automatically encrypted. You cannot create an encrypted volume from an unencrypted snapshot or vice versa. If your AMI uses encrypted volumes, you can only launch the AMI on supported instance types. For more information, see Amazon EBS encryption in the *Amazon EC2 User Guide for Linux Instances*.

Required: No

Type: Boolean

`Iops`

The number of I/O operations per second (IOPS) that the volume supports. This can be an integer from 100 – 2000.

Required: Conditional Required when the volume type (p. 1104) is `io1`; not used with other volume types.

Type: Number

`SnapshotId`

The snapshot ID of the volume to use to create a block device.

Required: Conditional If you specify both `SnapshotId` and `VolumeSize`, `VolumeSize` must be equal or greater than the size of the snapshot.

Type: String

`VolumeSize`

The volume size, in gibibytes (GiB). For valid values, see the `Size` parameter for the CreateVolume action in the *Amazon EC2 API Reference*.

Required: Conditional If you specify both `SnapshotId` and `VolumeSize`, `VolumeSize` must be equal or greater than the size of the snapshot.

Type: String

Update requires: Some interruptions (p. 90)

`VolumeType`

The volume type. If you set the type to `io1`, you must also set the `Iops` property. For valid values, see the `VolumeType` parameter for the CreateVolume action in the *Amazon EC2 API Reference*.

Required: No

Type: String

Example

```
{
   "DeviceName":"/dev/sdc",
   "Ebs":{
      "SnapshotId":"snap-xxxxxx",
      "VolumeSize":"50",
      "VolumeType":"io1",
      "Iops":"1000",
      "DeleteOnTermination":"false"
   }
}
```

See Also

- CreateVolume in the *Amazon Elastic Compute Cloud API Reference*

Amazon EC2 Instance SsmAssociations

`SsmAssociations` is a property of the AWS::EC2::Instance (p. 574) resource that specifies the Amazon EC2 Systems Manager (SSM) document and parameter values to associate with an instance.

Syntax

JSON

```
{
  "AssociationParameters" : [ Parameters, ... ],
  "DocumentName" : String
}
```

YAML

```
AssociationParameters:
  - Parameters
DocumentName: String
```

Properties

`AssociationParameters`

The input parameter values to use with the associated SSM document.

Required: No

Type: List of Amazon EC2 Instance SsmAssociations AssociationParameters (p. 1105)

`DocumentName`

The name of an SSM document to associate with the instance.

Required: Yes

Type: String

Amazon EC2 Instance SsmAssociations AssociationParameters

`AssociationParameters` is a property of the Amazon EC2 Instance SsmAssociations (p. 1105) property that specifies input parameter values for an Amazon EC2 Systems Manager (SSM) document.

Syntax

JSON

```
{
  "Key" : String,
  "Value" : [ String, ... ]
}
```

YAML

```
Key: String
Value:
  - String
```

Properties

`Key`

The name of an input parameter that is in the associated SSM document.

Required: Yes

Type: String

`Value`

The value of an input parameter.

Required: Yes

Type: List of strings

EC2 MountPoint Property Type

The EC2 MountPoint property is an embedded property of the AWS::EC2::Instance (p. 574) type.

Syntax

JSON

```
{
  "Device (p. 1106)" : String,
  "VolumeId (p. 1106)" : String
}
```

YAML

```
Device (p. 1106): String,
  VolumeId (p. 1106): String
```

Properties

`Device`

How the device is exposed to the instance (such as /dev/sdh, or xvdh).

Required: Yes

Type: String

`VolumeId`

The ID of the Amazon EBS volume. The volume and instance must be within the same Availability Zone and the instance must be running.

Required: Yes

Type: String

Example

This mount point (specified in the `Volumes` property in the EC2 instance) refers to a named EBS volume, "NewVolume".

```
"Ec2Instance" : {
   "Type" : "AWS::EC2::Instance",
   "Properties" : {
      "AvailabilityZone" : {
         "Fn::FindInMap" : [ "RegionMap", { "Ref" : "AWS::Region" }, "TestAz" ]
      },
      "SecurityGroups" : [ { "Ref" : "InstanceSecurityGroup" } ],
      "KeyName" : { "Ref" : "KeyName" },
      "ImageId" : {
         "Fn::FindInMap" : [ "RegionMap", { "Ref" : "AWS::Region" }, "AMI" ]
      },
      "Volumes" : [
         { "VolumeId" : { "Ref" : "NewVolume" }, "Device" : "/dev/sdk" }
      ]
   }
},
"NewVolume" : {
   "Type" : "AWS::EC2::Volume",
   "Properties" : {
      "Size" : "100",
      "AvailabilityZone" : {
         "Fn::FindInMap" : [ "RegionMap", { "Ref" : "AWS::Region" }, "TestAz" ]
      }
   }
}
```

See Also

- AWS::EC2::Instance (p. 574)
- AWS::EC2::Volume (p. 633)

EC2 NetworkInterface Embedded Property Type

The `EC2 Network Interface` type is an embedded property of the AWS::EC2::Instance (p. 574) type. It specifies a network interface that is to be attached.

Syntax

JSON

```
{
  "AssociatePublicIpAddress (p. 1108)" : Boolean,
  "DeleteOnTermination (p. 1108)" : Boolean,
  "Description (p. 1108)" : String,
  "DeviceIndex (p. 1108)" : String,
  "GroupSet (p. 1109)" : [ String, ... ],
  "NetworkInterfaceId (p. 1109)" : String,
```

```
  "Ipv6AddressCount" : Integer,
  "Ipv6Addresses" : [ IPv6 Address Type, ... ],
  "PrivateIpAddress (p. 1109)" : String,
  "PrivateIpAddresses (p. 1109)" : [ PrivateIpAddressSpecification, ... ],
  "SecondaryPrivateIpAddressCount (p. 1110)" : Integer,
  "SubnetId (p. 1110)" : String
}
```

YAML

```
AssociatePublicIpAddress (p. 1108): Boolean
DeleteOnTermination (p. 1108): Boolean
Description (p. 1108): String
DeviceIndex (p. 1108): String
GroupSet (p. 1109):
  - String
NetworkInterfaceId (p. 1109): String
Ipv6AddressCount: Integer
Ipv6Addresses:
  - IPv6 Address Type
PrivateIpAddress (p. 1109): String
PrivateIpAddresses (p. 1109):
  - PrivateIpAddressSpecification
SecondaryPrivateIpAddressCount (p. 1110): Integer
SubnetId (p. 1110): String
```

Properties

`AssociatePublicIpAddress`

Indicates whether the network interface receives a public IP address. You can associate a public IP address with a network interface only if it has a device index of `eth0` and if it is a new network interface (not an existing one). In other words, if you specify true, don't specify a network interface ID. For more information, see Amazon EC2 Instance IP Addressing.

Required: No

Type: Boolean.

`DeleteOnTermination`

Whether to delete the network interface when the instance terminates.

Required: No

Type: Boolean.

`Description`

The description of this network interface.

Required: No

Type: String

`DeviceIndex`

The network interface's position in the attachment order.

Required: Yes

Type: String

`GroupSet`

A list of security group IDs associated with this network interface.

Required: No

Type: List of strings.

`NetworkInterfaceId`

An existing network interface ID.

Required: Conditional. If you don't specify the `SubnetId` property, you must specify this property.

Type: String

`Ipv6AddressCount`

The number of IPv6 addresses to associate with the network interface. Amazon EC2 automatically selects the IPv6 addresses from the subnet range. To specify specific IPv6 addresses, use the `Ipv6Addresses` property and don't specify this property.

For restrictions on which instance types support IPv6 addresses, see the RunInstances action in the *Amazon EC2 API Reference*.

Required: No

Type: Integer

`Ipv6Addresses`

One or more specific IPv6 addresses from the IPv6 CIDR block range of your subnet to associate with the network interface. To specify a number of IPv6 addresses, use the `Ipv6AddressCount` property and don't specify this property.

For information about restrictions on which instance types support IPv6 addresses, see the RunInstances action in the *Amazon EC2 API Reference*.

Required: No

Type: List of EC2 NetworkInterface Ipv6Addresses (p. 1111)

`PrivateIpAddress`

Assigns a single private IP address to the network interface, which is used as the primary private IP address. If you want to specify multiple private IP address, use the `PrivateIpAddresses` property.

Required: No

Type: String

`PrivateIpAddresses`

Assigns a list of private IP addresses to the network interface. You can specify a primary private IP address by setting the value of the `Primary` property to `true` in the `PrivateIpAddressSpecification` property. If you want Amazon EC2 to automatically assign private IP addresses, use the `SecondaryPrivateIpCount` property and do not specify this property.

For information about the maximum number of private IP addresses, see Private IP Addresses Per ENI Per Instance Type in the *Amazon EC2 User Guide for Linux Instances*.

Required: No

Type: list of PrivateIpAddressSpecification (p. 1112)

`SecondaryPrivateIpAddressCount`

The number of secondary private IP addresses that Amazon EC2 auto assigns to the network interface. Amazon EC2 uses the value of the `PrivateIpAddress` property as the primary private IP address. If you don't specify that property, Amazon EC2 auto assigns both the primary and secondary private IP addresses.

If you want to specify your own list of private IP addresses, use the `PrivateIpAddresses` property and do not specify this property.

For information about the maximum number of private IP addresses, see Private IP Addresses Per ENI Per Instance Type in the *Amazon EC2 User Guide for Linux Instances*.

Required: No

Type: Integer.

`SubnetId`

The ID of the subnet to associate with the network interface.

Required: Conditional. If you don't specify the `NetworkInterfaceId` property, you must specify this property.

Type: String

EC2 NetworkAclEntry Icmp

The `Icmp` property is an embedded property of the AWS::EC2::NetworkAclEntry (p. 590) type.

Syntax

JSON

```
{
   "Code" : Integer,
   "Type" : Integer
}
```

YAML

```
Code: Integer
Type: Integer
```

Properties

`Code`

The Internet Control Message Protocol (ICMP) code. You can use -1 to specify all ICMP codes for the given ICMP type.

Required: Conditional. Required if you specify `1` (ICMP) for the CreateNetworkAclEntry protocol parameter.

Type: Integer

`Type`

The Internet Control Message Protocol (ICMP) type. You can use -1 to specify all ICMP types.

Required: Conditional. Required if you specify `1` (ICMP) for the CreateNetworkAclEntry protocol parameter.

Type: Integer

EC2 NetworkAclEntry PortRange

The `PortRange` property is an embedded property of the AWS::EC2::NetworkAclEntry (p. 590) type.

Syntax

JSON

```
{
   "From" : Integer,
   "To" : Integer
}
```

YAML

```
From: Integer
To: Integer
```

Properties

`From`

The first port in the range.

Required: Conditional. Required if you specify 6 (TCP) or 17 (UDP) for the protocol parameter.

Type: Integer

`To`

The last port in the range.

Required: Conditional. Required if you specify 6 (TCP) or 17 (UDP) for the protocol parameter.

Type: Integer

EC2 NetworkInterface Ipv6Addresses

`Ipv6Addresses` is a property of the AWS::EC2::NetworkInterface (p. 594) resource that specifies an IPv6 address to associate with the network interface.

Syntax

JSON

```
{
   "Ipv6Address" : String
}
```

YAML

```
Ipv6Address: String
```

Properties

`Ipv6Address`

The IPv6 address to associate with the network interface.

Required: Yes

Type: String

EC2 Network Interface Private IP Specification

The `PrivateIpAddressSpecification` type is an embedded property of the AWS::EC2::NetworkInterface (p. 594) type.

Syntax

JSON

```
{
  "PrivateIpAddress" : String,
  "Primary" : Boolean
}
```

YAML

```
PrivateIpAddress: String
Primary: Boolean
```

Properties

`PrivateIpAddress`

The private IP address of the network interface.
Required: Yes
Type: String

`Primary`

Sets the private IP address as the primary private address. You can set only one primary private IP address. If you don't specify a primary private IP address, Amazon EC2 automatically assigns a primary private IP address.
Required: Yes
Type: Boolean

EC2 Security Group Rule Property Type

The EC2 Security Group Rule is an embedded property of the AWS::EC2::SecurityGroup (p. 608) type.

Syntax SecurityGroupIngress

JSON

```
{
  "CidrIp (p. 1113)" : String,
  "CidrIpv6 (p. 1114)" : String,
  "FromPort (p. 1114)" : Integer,
  "IpProtocol (p. 1114)" : String,
  "SourceSecurityGroupId (p. 1114)" : String,
  "SourceSecurityGroupName (p. 1114)" : String,
  "SourceSecurityGroupOwnerId (p. 1115)" : String,
  "ToPort (p. 1115)" : Integer
}
```

YAML

```
CidrIp (p. 1113): String
FromPort (p. 1114): Integer
IpProtocol (p. 1114): String
SourceSecurityGroupId (p. 1114): String
SourceSecurityGroupName (p. 1114): String
SourceSecurityGroupOwnerId (p. 1115): String
ToPort (p. 1115): Integer
```

Syntax SecurityGroupEgress

JSON

```
{
  "CidrIp (p. 1113)" : String,
  "CidrIpv6 (p. 1114)" : String,
  "DestinationPrefixListId (SecurityGroupEgress only)" : String,
  "FromPort (p. 1114)" : Integer,
  "IpProtocol (p. 1114)" : String,
  "DestinationSecurityGroupId (p. 1114)" : String,
  "ToPort (p. 1115)" : Integer
}
```

YAML

```
CidrIp (p. 1113): String
DestinationPrefixListId (SecurityGroupEgress only): String
FromPort (p. 1114): Integer
IpProtocol (p. 1114): String
DestinationSecurityGroupId (p. 1114): String
ToPort (p. 1115): Integer
```

Properties

`CidrIp`

Specifies an IPv4 CIDR range.

Required: Conditional. You must specify only one of the following properties: `CidrIp`, `CidrIpv6`, `DestinationPrefixListId`, `DestinationSecurityGroupId`, or `SourceSecurityGroupId`.

Type: String

`CidrIpv6`

Specifies an IPv6 CIDR range.

Required: Conditional. You must specify only one of the following properties: `CidrIp`, `CidrIpv6`, `DestinationPrefixListId`, `DestinationSecurityGroupId`, or `SourceSecurityGroupId`.

Type: String

`DestinationPrefixListId (SecurityGroupEgress only)`

The AWS service prefix of an Amazon VPC endpoint. For more information, see VPC Endpoints in the *Amazon VPC User Guide*.

Required: Conditional. You must specify only one of the following properties: `CidrIp`, `CidrIpv6`, `DestinationPrefixListId`, `DestinationSecurityGroupId`, or `SourceSecurityGroupId`.

Type: String

`DestinationSecurityGroupId (SecurityGroupEgress only)`

Specifies the GroupId of the destination Amazon VPC security group.

Required: Conditional. You must specify only one of the following properties: `CidrIp`, `CidrIpv6`, `DestinationPrefixListId`, `DestinationSecurityGroupId`, or `SourceSecurityGroupId`.

Type: String

`FromPort`

The start of port range for the TCP and UDP protocols, or an ICMP type number. An ICMP type number of -1 indicates a wildcard (i.e., any ICMP type number).

Required: No

Type: Integer

`IpProtocol`

An IP protocol name or number. For valid values, go to the IpProtocol parameter in AuthorizeSecurityGroupIngress

Required: Yes

Type: String

`SourceSecurityGroupId (SecurityGroupIngress only)`

For VPC security groups only. Specifies the ID of the Amazon EC2 Security Group to allow access. You can use the `Ref` intrinsic function to refer to the logical ID of a security group defined in the same template.

Required: Conditional. You must specify only one of the following properties: `CidrIp`, `CidrIpv6`, `DestinationPrefixListId`, `DestinationSecurityGroupId`, or `SourceSecurityGroupId`.

Type: String

`SourceSecurityGroupName (SecurityGroupIngress only)`

For non-VPC security groups only. Specifies the name of the Amazon EC2 Security Group to use for access. You can use the `Ref` intrinsic function to refer to the logical name of a security group that is defined in the same template.

Required: Conditional. If you specify `CidrIp`, do not specify `SourceSecurityGroupName`.

Type: String

`SourceSecurityGroupOwnerId (SecurityGroupIngress only)`

Specifies the AWS Account ID of the owner of the Amazon EC2 Security Group that is specified in the `SourceSecurityGroupName` property.

Required: Conditional. If you specify `SourceSecurityGroupName` and that security group is owned by a different account than the account creating the stack, you must specify the `SourceSecurityGroupOwnerId`; otherwise, this property is optional.

Type: String

`ToPort`

The end of port range for the TCP and UDP protocols, or an ICMP code. An ICMP code of -1 indicates a wildcard (i.e., any ICMP code).

Required: No

Type: Integer

Examples

Security Group with CidrIp

JSON

```
"InstanceSecurityGroup" : {
   "Type" : "AWS::EC2::SecurityGroup",
   "Properties" : {
      "GroupDescription" : "Enable SSH access via port 22",
      "SecurityGroupIngress" : [ {
         "IpProtocol" : "tcp",
         "FromPort" : "22",
         "ToPort" : "22",
         "CidrIp" : "0.0.0.0/0"
      } ]
   }
}
```

YAML

```
InstanceSecurityGroup:
  Type: "AWS::EC2::SecurityGroup"
  Properties:
    GroupDescription: "Enable SSH access via port 22"
    SecurityGroupIngress:
      -
        IpProtocol: "tcp"
        FromPort: "22"
        ToPort: "22"
        CidrIp: "0.0.0.0/0"
```

Security Group with Security Group Id

JSON

```
"InstanceSecurityGroup" : {
   "Type" : "AWS::EC2::SecurityGroup",
   "Properties" : {
      "GroupDescription" : "Enable HTTP access on the configured port",
```

```
      "VpcId" : { "Ref" : "VpcId" },
      "SecurityGroupIngress" : [ {
         "IpProtocol" : "tcp",
         "FromPort" : { "Ref" : "WebServerPort" },
         "ToPort" : { "Ref" : "WebServerPort" },
         "SourceSecurityGroupId" : { "Ref" : "LoadBalancerSecurityGroup" }
      } ]
   }
}
```

YAML

```
InstanceSecurityGroup:
  Type: "AWS::EC2::SecurityGroup"
  Properties:
    GroupDescription: "Enable HTTP access on the configured port"
    VpcId:
      Ref: "VpcId"
    SecurityGroupIngress:
      -
        IpProtocol: "tcp"
        FromPort:
          Ref: "WebServerPort"
        ToPort:
          Ref: "WebServerPort"
        SourceSecurityGroupId:
          Ref: "LoadBalancerSecurityGroup"
```

Security Group with Multiple Ingress Rules

This snippet grants SSH access with CidrIp, and HTTP access with `SourceSecurityGroupName`. `Fn::GetAtt` is used to derive the values for `SourceSecurityGroupName` and `SourceSecurityGroupOwnerId` from the elastic load balancer.

JSON

```
"ElasticLoadBalancer" : {
   "Type" : "AWS::ElasticLoadBalancing::LoadBalancer",
   "Properties" : {
      "AvailabilityZones" : { "Fn::GetAZs" : "" },
      "Listeners" : [ {
         "LoadBalancerPort" : "80",
         "InstancePort" : { "Ref" : "WebServerPort" },
         "Protocol" : "HTTP"
      } ],
      "HealthCheck" : {
         "Target" : { "Fn::Join" : [ "", ["HTTP:", { "Ref" : "WebServerPort" }, "/"]]},
         "HealthyThreshold" : "3",
         "UnhealthyThreshold" : "5",
         "Interval" : "30",
         "Timeout" : "5"
      }
   }
},

"InstanceSecurityGroup" : {
   "Type" : "AWS::EC2::SecurityGroup",
   "Properties" : {
      "GroupDescription" : "Enable SSH access and HTTP from the load balancer only",
      "SecurityGroupIngress" : [ {
         "IpProtocol" : "tcp",
         "FromPort" : "22",
```

```
          "ToPort" : "22",
          "CidrIp" : "0.0.0.0/0"
       }, {
          "IpProtocol" : "tcp",
          "FromPort" : { "Ref" : "WebServerPort" },
          "ToPort" : { "Ref" : "WebServerPort" },
          "SourceSecurityGroupOwnerId" : {"Fn::GetAtt" : ["ElasticLoadBalancer",
 "SourceSecurityGroup.OwnerAlias"]},
          "SourceSecurityGroupName" : {"Fn::GetAtt" : ["ElasticLoadBalancer",
 "SourceSecurityGroup.GroupName"]}
       } ]
    }
}
```

YAML

```
ElasticLoadBalancer:
  Type: "AWS::ElasticLoadBalancing::LoadBalancer"
  Properties:
    AvailabilityZones:
      Fn::GetAZs: ""
    Listeners:
      -
        LoadBalancerPort: "80"
        InstancePort:
          Ref: "WebServerPort"
        Protocol: "HTTP"
    HealthCheck:
      Target:
        Fn::Join:
          - ""
          -
            - "HTTP:"
            -
              Ref: "WebServerPort"
            - "/"
      HealthyThreshold: "3"
      UnhealthyThreshold: "5"
      Interval: "30"
      Timeout: "5"
InstanceSecurityGroup:
  Type: "AWS::EC2::SecurityGroup"
  Properties:
    GroupDescription: "Enable SSH access and HTTP from the load balancer only"
    SecurityGroupIngress:
      -
        IpProtocol: "tcp"
        FromPort: "22"
        ToPort: "22"
        CidrIp: "0.0.0.0/0"
      -
        IpProtocol: "tcp"
        FromPort:
          Ref: "WebServerPort"
        ToPort:
          Ref: "WebServerPort"
        SourceSecurityGroupOwnerId:
          Fn::GetAtt:
            - "ElasticLoadBalancer"
            - "SourceSecurityGroup.OwnerAlias"
        SourceSecurityGroupName:
          Fn::GetAtt:
            - "ElasticLoadBalancer"
            - "SourceSecurityGroup.GroupName"
```

See Also

- Amazon EC2 Security Groups in the *Amazon EC2 User Guide*

Amazon EC2 SpotFleet SpotFleetRequestConfigData

`SpotFleetRequestConfigData` is a property of the AWS::EC2::SpotFleet (p. 622) resource that defines the configuration of a Spot fleet request.

Syntax

JSON

```
{
  "AllocationStrategy" : String,
  "ExcessCapacityTerminationPolicy" : String,
  "IamFleetRole" : String,
  "LaunchSpecifications" : [ LaunchSpecifications, ... ],
  "SpotPrice" : String,
  "TargetCapacity" : Integer,
  "TerminateInstancesWithExpiration" : Boolean,
  "ValidFrom" : String,
  "ValidUntil" : String
}
```

YAML

```
AllocationStrategy: String
ExcessCapacityTerminationPolicy: String
IamFleetRole: String
LaunchSpecifications:
  - LaunchSpecifications
SpotPrice: String
TargetCapacity: Integer
TerminateInstancesWithExpiration: Boolean
ValidFrom: String
ValidUntil: String
```

Properties

`AllocationStrategy`

Indicates how to allocate the target capacity across the Spot pools that you specified in the Spot fleet request. For valid values, see SpotFleetRequestConfigData in the *Amazon EC2 API Reference*.

Required: No

Type: String

`ExcessCapacityTerminationPolicy`

Indicates whether running Spot instances are terminated if you decrease the target capacity of the Spot fleet request below the current size of the Spot fleet. For valid values, see SpotFleetRequestConfigData in the *Amazon EC2 API Reference*.

Required: No

Type: String

`IamFleetRole`

The Amazon Resource Name (ARN) of an AWS Identity and Access Management (IAM) role that grants the Spot fleet the ability to bid on, launch, and terminate instances on your behalf. For more information, see Spot Fleet Prerequisites in the *Amazon EC2 User Guide for Linux Instances*.

Required: Yes

Type: String

`LaunchSpecifications`

The launch specifications for the Spot fleet request.

Required: Yes

Type: List of Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications (p. 1120)

`SpotPrice`

The bid price per unit hour. For more information, see How Spot Fleet Works in the *Amazon EC2 User Guide for Linux Instances*.

Required: Yes

Type: String

`TargetCapacity`

The number of units to request for the spot fleet. You can choose to set the target capacity as the number of instances or as a performance characteristic that is important to your application workload, such as vCPUs, memory, or I/O. For more information, see How Spot Fleet Works in the *Amazon EC2 User Guide for Linux Instances*.

Required: Yes

Type: Integer

`TerminateInstancesWithExpiration`

Indicates whether running Spot instances are terminated when the Spot fleet request expires.

Required: No

Type: Boolean

`ValidFrom`

The start date and time of the request, in UTC format (*YYYY*-*MM*-*DD*T*HH*:*MM*:*SS*Z). By default, Amazon Elastic Compute Cloud (Amazon EC2) starts fulfilling the request immediately.

Required: No

Type: String

`ValidUntil`

The end date and time of the request, in UTC format (*YYYY*-*MM*-*DD*T*HH*:*MM*:*SS*Z). After the end date and time, Amazon EC2 doesn't request new Spot instances or enable them to fulfill the request.

Required: No

Type: String

Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications

`LaunchSpecifications` is a property of the Amazon EC2 SpotFleet SpotFleetRequestConfigData (p. 1118) property that defines the launch specifications for the Spot fleet request.

Syntax

JSON

```
{
  "BlockDeviceMappings" : [ BlockDeviceMapping, ... ],
  "EbsOptimized" : Boolean,
  "IamInstanceProfile" : IamInstanceProfile,
  "ImageId" : String,
  "InstanceType" : String,
  "KernelId" : String,
  "KeyName" : String,
  "Monitoring" : Boolean,
  "NetworkInterfaces" : [ NetworkInterface, ... ],
  "Placement" : Placement,
  "RamdiskId" : String,
  "SecurityGroups" : [ SecurityGroup, ... ],
  "SpotPrice" : String,
  "SubnetId" : String,
  "UserData" : String,
  "WeightedCapacity" : Number
}
```

YAML

```
BlockDeviceMappings:
  - BlockDeviceMapping
EbsOptimized: Boolean
IamInstanceProfile:
  IamInstanceProfile
ImageId: String
InstanceType: String
KernelId: String
KeyName: String
Monitoring: Boolean
NetworkInterfaces:
  NetworkInterface
Placement:
  Placement
RamdiskId: String
SecurityGroups:
  SecurityGroup
SpotPrice: String
SubnetId: String
UserData: String
WeightedCapacity: Number
```

Properties

`BlockDeviceMappings`

Defines the block devices that are mapped to the Spot instances.

Required: No

Type: List of Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications BlockDeviceMappings (p. 1123)

`EbsOptimized`

Indicates whether the instances are optimized for Amazon Elastic Block Store (Amazon EBS) I/O. This optimization provides dedicated throughput to Amazon EBS and an optimized configuration stack to provide optimal EBS I/O performance. This optimization isn't available with all instance types. Additional usage charges apply when you use an Amazon EBS-optimized instance.

Required: No

Type: Boolean

`IamInstanceProfile`

Defines the AWS Identity and Access Management (IAM) instance profile to associate with the instances.

Required: No

Type: Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications IamInstanceProfile (p. 1125)

`ImageId`

The unique ID of the Amazon Machine Image (AMI) to launch on the instances.

Required: Yes

Type: String

`InstanceType`

Specifies the instance type of the EC2 instances.

Required: Yes

Type: String

`KernelId`

The ID of the kernel that is associated with the Amazon Elastic Compute Cloud (Amazon EC2) AMI.

Required: No

Type: String

`KeyName`

An Amazon EC2 key pair to associate with the instances.

Required: No

Type: String

`Monitoring`

Enable or disable monitoring for the instances.

Required: No

Type: Amazon EC2 SpotFleet SpotFleetRequestConfigData LaunchSpecifications Monitoring (p. 1126)

`NetworkInterfaces`

The network interfaces to associate with the instances.

Required: No

Type: List of Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications NetworkInterfaces (p. 1127)

`Placement`

Defines a placement group, which is a logical grouping of instances within a single Availability Zone (AZ).

Required: No

Type: Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications Placement (p. 1130)

`RamdiskId`

The ID of the RAM disk to select. Some kernels require additional drivers at launch. Check the kernel requirements for information about whether you need to specify a RAM disk. To find kernel requirements, refer to the AWS Resource Center and search for the kernel ID.

Required: No

Type: String

`SecurityGroups`

One or more security group IDs to associate with the instances.

Required: No

Type: List of Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications SecurityGroups (p. 1130)

`SpotPrice`

The bid price per unit hour for the specified instance type. If you don't specify a value, Amazon EC2 uses the Spot bid price for the fleet. For more information, see How Spot Fleet Works in the *Amazon EC2 User Guide for Linux Instances*.

Required: No

Type: String

`SubnetId`

The ID of the subnet in which to launch the instances.

Required: No

Type: String

`UserData`

Base64-encoded MIME user data that instances use when starting up.

Required: No

Type: String

`WeightedCapacity`

The number of units provided by the specified instance type. These units are the same units that you chose to set the target capacity in terms of instances or a performance characteristic, such as vCPUs, memory, or I/O. For more information, see How Spot Fleet Works in the *Amazon EC2 User Guide for Linux Instances*.

If the target capacity divided by this value is not a whole number, Amazon EC2 rounds the number of instances to the next whole number.

Required: No

Type: Number

Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications BlockDeviceMappings

`BlockDeviceMappings` is a property of the Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications (p. 1120) property that defines the block devices that are mapped to an instance.

Syntax

JSON

```
{
  "DeviceName" : String,
  "Ebs" : EBSBlockDevice,
  "NoDevice" : Boolean,
  "VirtualName" : String
}
```

YAML

```
DeviceName: String
Ebs:
  EBSBlockDevice
NoDevice: Boolean
VirtualName: String
```

Properties

`DeviceName`

The name of the device within the EC2 instance, such as `/dev/dsh` or `xvdh`.

Required: Yes

Type: String

`Ebs`

The Amazon Elastic Block Store (Amazon EBS) volume information.

Required: Conditional You can specify either the `VirtualName` or `Ebs`, but not both.

Type: Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications BlockDeviceMappings Ebs (p. 1124)

`NoDevice`

Suppresses the specified device that is included in the block device mapping of the Amazon Machine Image (AMI).

Required: No

Type: Boolean

`VirtualName`

The name of the virtual device. The name must be in the form `ephemeralX` where *X* is a number equal to or greater than zero (0), for example, `ephemeral0`.

Required: Conditional You can specify either the `VirtualName` or `Ebs`, but not both.

Type: String

Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications BlockDeviceMappings Ebs

`Ebs` is a property of the Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications BlockDeviceMappings (p. 1123) property that defines a block device for an Amazon Elastic Block Store (Amazon EBS) volume.

Syntax

JSON

```
{
  "DeleteOnTermination" : Boolean,
  "Encrypted" : Boolean,
  "Iops" : Integer,
  "SnapshotId" : String,
  "VolumeSize" : Integer,
  "VolumeType" : String
}
```

YAML

```
DeleteOnTermination: Boolean
Encrypted: Boolean
Iops: Integer
SnapshotId: String
VolumeSize: Integer
VolumeType: String
```

Properties

`DeleteOnTermination`

Indicates whether to delete the volume when the instance is terminated.

Required: No

Type: Boolean

`Encrypted`

Indicates whether the EBS volume is encrypted. Encrypted Amazon EBS volumes can be attached only to instances that support Amazon EBS encryption.

Required: No

Type: Boolean

`Iops`

The number of I/O operations per second (IOPS) that the volume supports. For more information, see Iops for the `EbsBlockDevice` action in the *Amazon EC2 API Reference*.

Required: No

Type: Integer

`SnapshotId`

The snapshot ID of the volume that you want to use. If you specify both the `SnapshotId` and `VolumeSize` properties, `VolumeSize` must be equal to or greater than the size of the snapshot.

Required: No

Type: String

`VolumeSize`

The volume size, in Gibibytes (GiB). If you specify both the `SnapshotId` and `VolumeSize` properties, `VolumeSize` must be equal to or greater than the size of the snapshot. For more information about specifying the volume size, see VolumeSize for the `EbsBlockDevice` action in the *Amazon EC2 API Reference*.

Required: No

Type: Integer

`VolumeType`

The volume type. For more information about specifying the volume type, see VolumeType for the `EbsBlockDevice` action in the *Amazon EC2 API Reference*.

Required: No

Type: String

Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications IamInstanceProfile

`IamInstanceProfile` is a property of the Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications (p. 1120) property that specifies the IAM instance profile to associate with the instances.

Syntax

JSON

```
{
  "Arn" : String
}
```

YAML

```
Arn: String
```

Properties

`Arn`

The Amazon Resource Name (ARN) of the instance profile to associate with the instances. The instance profile contains the IAM role that is associated with the instances.

Required: No

Type: String

Amazon EC2 SpotFleet SpotFleetRequestConfigData LaunchSpecifications Monitoring

`Monitoring` is a property of the Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications (p. 1120) property that enables instance monitoring.

Syntax

JSON

```
{
  "Enabled" : Boolean
}
```

YAML

```
Enabled: Boolean
```

Properties

`Enabled`

Indicates whether monitoring is enabled for the instances.

Required: No

Type: Boolean

Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications NetworkInterfaces

`NetworkInterfaces` is a property of the Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications (p. 1120) property that defines the network interface of the instances in a Spot fleet.

Syntax

JSON

```
{
  "AssociatePublicIpAddress" : Boolean,
  "DeleteOnTermination" : Boolean,
  "Description" : String,
  "DeviceIndex" : Integer,
  "Groups" : [ String, ... ],
  "Ipv6AddressCount" : Integer,
  "Ipv6Addresses" : [ IPv6 Address Type, ... ],
  "NetworkInterfaceId" : String,
  "PrivateIpAddresses" : [ PrivateIpAddresses, ... ],
  "SecondaryPrivateIpAddressCount" : Integer,
  "SubnetId" : String
}
```

YAML

```
AssociatePublicIpAddress: Boolean
DeleteOnTermination: Boolean
Description: String
DeviceIndex: Integer
Groups:
  - String
Ipv6AddressCount: Integer
Ipv6Addresses:
  - IPv6 Address Type
NetworkInterfaceId: String
PrivateIpAddresses:
  - PrivateIpAddresses
SecondaryPrivateIpAddressCount: Integer
SubnetId: String
```

Properties

`AssociatePublicIpAddress`

Indicates whether to assign a public IP address to an instance that you launch in a VPC. You can assign the public IP address can only to a network interface for eth0, and only to a new network interface, not an existing one.

Required: No

Type: Boolean

`DeleteOnTermination`

Indicates whether to delete the network interface when the instance terminates.

Required: No

Type: Boolean

`Description`

The description of this network interface.

Required: No

Type: String

`DeviceIndex`

The network interface's position in the attachment order.

Required: Yes

Type: Integer

`Groups`

A list of security group IDs to associate with this network interface.

Required: No

Type: List of strings

`Ipv6AddressCount`

The number of IPv6 addresses to associate with the network interface. Amazon Elastic Compute Cloud automatically selects the IPv6 addresses from the subnet range. To specify specific IPv6 addresses, use the `Ipv6Addresses` property and don't specify this property.

Required: No

Type: Integer

`Ipv6Addresses`

One or more specific IPv6 addresses from the IPv6 CIDR block range of your subnet to associate with the network interface. To specify a number of IPv6 addresses, use the `Ipv6AddressCount` property and don't specify this property.

Required: No

Type: List of EC2 NetworkInterface Ipv6Addresses (p. 1111)

`NetworkInterfaceId`

A network interface ID.

Required: No

Type: String

`PrivateIpAddresses`

One or more private IP addresses to assign to the network interface.

Required: No

Type: List of Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications NetworkInterfaces PrivateIpAddresses (p. 1129)

`SecondaryPrivateIpAddressCount`

The number of secondary private IP addresses that Amazon EC2 automatically assigns to the network interface.

Required: No

Type: Integer

`SubnetId`

The ID of the subnet to associate with the network interface.

Required: Conditional. If you don't specify the `NetworkInterfaceId` property, you must specify this property.

Type: String

Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications NetworkInterfaces PrivateIpAddresses

`PrivateIpAddresses` is a property of the Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications NetworkInterfaces (p. 1127) property that specifies the private IP address that you want to assign to the network interface.

Syntax

JSON

```
{
  "Primary" : Boolean,
  "PrivateIpAddress" : String
}
```

YAML

```
Primary: Boolean
PrivateIpAddress: String
```

Properties

`Primary`

Indicates whether the private IP address is the primary private IP address. You can designate only one IP address as primary.

Required: No

Type: Boolean

`PrivateIpAddress`

The private IP address.

Required: Yes

Type: String

Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications Placement

`Placement` is a property of the Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications (p. 1120) property that defines the placement group for the Spot instances.

Syntax

JSON

```
{
  "AvailabilityZone" : String,
  "GroupName" : String
}
```

YAML

```
AvailabilityZone: String
GroupName: String
```

Properties

`AvailabilityZone`

The Availability Zone (AZ) of the placement group.

Required: No

Type: String

`GroupName`

The name of the placement group (for cluster instances).

Required: No

Type: String

Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications SecurityGroups

`SecurityGroups` is a property of the Amazon Elastic Compute Cloud SpotFleet SpotFleetRequestConfigData LaunchSpecifications (p. 1120) property that specifies a security group to associate with the instances.

Syntax

JSON

```
{
  "GroupId" : String
}
```

YAML

```
GroupId: String
```

Properties

`GroupId`

The ID of a security group.

Required: No

Type: String

Amazon EC2 Container Service Service DeploymentConfiguration

`DeploymentConfiguration` is a property of the AWS::ECS::Service (p. 671) resource that configures how many tasks run when you update a running Amazon EC2 Container Service (Amazon ECS) service.

Syntax

JSON

```
{
  "MaximumPercent" : Integer,
  "MinimumHealthyPercent" : Integer
}
```

YAML

```
MaximumPercent: Integer
MinimumHealthyPercent: Integer
```

Properties

`MaximumPercent`

The maximum number of tasks, specified as a percentage of the Amazon ECS service's `DesiredCount` value, that can run in a service during a deployment. To calculate the maximum number of tasks, Amazon ECS uses this formula: the value of `DesiredCount` * (the value of the `MaximumPercent`/100), rounded down to the nearest integer value.

Required: No

Type: Integer

`MinimumHealthyPercent`

The minimum number of tasks, specified as a percentage of the Amazon ECS service's `DesiredCount` value, that must continue to run and remain healthy during a deployment. To calculate the minimum number of tasks, Amazon ECS uses this formula: the value of `DesiredCount` * (the value of the `MinimumHealthyPercent`/100), rounded up to the nearest integer value.

Required: No

Type: Integer

Amazon EC2 Container Service Service LoadBalancers

`LoadBalancers` is a property of the AWS::ECS::Service (p. 671) resource that specifies the load balancer to associate with an Amazon EC2 Container Service (Amazon ECS) service.

Syntax

JSON

```
{
  "ContainerName" : String,
  "ContainerPort" : Integer,
  "LoadBalancerName" : String,
  "TargetGroupArn" : String
}
```

YAML

```
ContainerName: String
ContainerPort: Integer
LoadBalancerName: String
TargetGroupArn: String
```

Properties

`ContainerName`

The name of a container to use with the load balancer.

Required: Yes

Type: String

`ContainerPort`

The port number on the container to direct load balancer traffic to. Your container instances must allow ingress traffic on this port.

Required: Yes

Type: Integer

`LoadBalancerName`

The name of a Classic Load Balancer to associate with the Amazon ECS service.

Required: No

Type: String

`TargetGroupArn`

An Application load balancer target group Amazon Resource Name (ARN) to associate with the Amazon ECS service.

Required: No

Type: String

Amazon EC2 Container Service TaskDefinition ContainerDefinitions

`ContainerDefinitions` is a property of the AWS::ECS::TaskDefinition (p. 675) resource that describes the configuration of an Amazon EC2 Container Service (Amazon ECS) container. The container definitions are passed to the Docker daemon.

Syntax

JSON

```
{
  "Command" : [ String, ... ],
  "Cpu" : Integer,
  "DisableNetworking" : Boolean,
  "DnsSearchDomains" : [ String, ... ],
  "DnsServers" : [ String, ... ],
  "DockerLabels" : { String:String, ... },
  "DockerSecurityOptions" : [ String, ... ],
  "EntryPoint" : [ String, ... ],
  "Environment" : [ Environment Variable, ... ],
  "Essential" : Boolean,
  "ExtraHosts" : [ Host Entry, ... ],
  "Hostname" : String,
  "Image" : String,
  "Links" : [ String, ... ],
  "LogConfiguration" : Log Configuration,
  "Memory" : Integer,
  "MemoryReservation" : Integer,
  "MountPoints" : [ Mount Point, ... ],
  "Name" : String,
  "PortMappings" : [ Port Map, ... ],
  "Privileged" : Boolean,
  "ReadonlyRootFilesystem" : Boolean,
  "Ulimits" : [ Ulimit, ... ],
  "User" : String,
  "VolumesFrom" : [ Volume From, ... ],
  "WorkingDirectory" : String
}
```

YAML

```
Command:
  - String
Cpu: Integer
DisableNetworking: Boolean
DnsSearchDomains:
  - String
DnsServers:
  - String
DockerLabels:
  String: String
DockerSecurityOptions:
  - String
EntryPoint:
  - String
Environment:
  - Environment Variable
Essential: Boolean
ExtraHosts:
  - Host Entry
Hostname: String
Image: String
Links:
  - String
LogConfiguration:
  Log Configuration
Memory: Integer
MemoryReservation: Integer
MountPoints:
  - Mount Point
Name: String
PortMappings:
  - Port Map
Privileged: Boolean
ReadonlyRootFilesystem: Boolean
Ulimits:
  - Ulimit
User: String
VolumesFrom:
  - Volume From
WorkingDirectory: String
```

Properties

For more information about each property, see Task Definition Parameters in the *Amazon EC2 Container Service Developer Guide*.

`Command`

The `CMD` value to pass to the container. For more information about the Docker `CMD` parameter, see https://docs.docker.com/engine/reference/builder/#cmd.

Required: No

Type: List of strings

`Cpu`

The minimum number of CPU units to reserve for the container. Containers share unallocated CPU units with other containers on the instance by using the same ratio as their allocated CPU units. For more information, see the `cpu` content for the ContainerDefinition data type in the *Amazon EC2 Container Service API Reference*.

Required: No

Type: Integer

`DisableNetworking`

Indicates whether networking is disabled within the container.

Required: No

Type: Boolean

`DnsSearchDomains`

A list of DNS search domains that are provided to the container. The domain names that the DNS logic looks up when a process attempts to access a bare unqualified hostname.

Required: No

Type: List of strings

`DnsServers`

A list of DNS servers that Amazon ECS provides to the container.

Required: No

Type: List of strings

`DockerLabels`

A key-value map of labels for the container.

Required: No

Type: Key-value pairs, with the name of the label as the key and the label value as the value.

`DockerSecurityOptions`

A list of custom labels for SELinux and AppArmor multi-level security systems. For more information, see the `dockerSecurityOptions` content for the ContainerDefinition data type in the *Amazon EC2 Container Service API Reference*.

Required: No

Type: List of strings

`EntryPoint`

The `ENTRYPOINT` value to pass to the container. For more information about the Docker `ENTRYPOINT` parameter, see https://docs.docker.com/engine/reference/builder/#entrypoint.

Required: No

Type: List of strings

`Environment`

The environment variables to pass to the container.

Required: No

Type: List of Amazon EC2 Container Service TaskDefinition ContainerDefinitions Environment (p. 1138)

`Essential`

Indicates whether the task stops if this container fails. If you specify **`true`** and the container fails, all other containers in the task stop. If you specify **`false`** and the container fails, none of the other containers in the task is affected. This value is **`true`** by default.

You must have at least one essential container in a task.

Required: No

Type: Boolean

`ExtraHosts`

A list of hostnames and IP address mappings to append to the `/etc/hosts` file on the container.

Required: No

Type: List of Amazon EC2 Container Service TaskDefinition ContainerDefinitions HostEntry (p. 1139)

`Hostname`

The name that Docker will use for the container's hostname.

Required: No

Type: String

`Image`

The image to use for a container, which is passed directly to the Docker daemon. You can use images in the Docker Hub registry or specify other repositories (*`repository-url`*/*`image`*:*`tag`*).

Required: No

Type: String

`Links`

The name of another container to connect to. With links, containers can communicate with each other without using port mappings.

Required: No

Type: List of strings

`LogConfiguration`

Configures a custom log driver for the container. For more information, see the `logConfiguration` content for the ContainerDefinition data type in the *Amazon EC2 Container Service API Reference*.

Required: No

Type: Amazon EC2 Container Service TaskDefinition ContainerDefinitions LogConfiguration (p. 1139)

`Memory`

The number of MiB of memory to reserve for the container. If your container attempts to exceed the allocated memory, the container is terminated.

Required: Conditional. You must specify one or both of the `Memory` or `MemoryReservation` properties. If you specify both, the value for the `Memory` property must be greater than the value of the `MemoryReservation` property.

Type: Integer

`MemoryReservation`

The number of MiB of memory to reserve for the container. When system memory is under contention, Docker attempts to keep the container memory within the limit. If the container requires more memory, it can consume up to the value specified by the `Memory` property or all of the available memory on the container instance, whichever comes first. This is called a soft limit.

Required: Conditional. You must specify one or both of the `Memory` or `MemoryReservation` properties. If you specify both, the value for the `Memory` property must be greater than the value of the `MemoryReservation` property.

Type: Integer

`MountPoints`

The mount points for data volumes in the container.

Required: No

Type: List of Amazon EC2 Container Service TaskDefinition ContainerDefinitions MountPoints (p. 1140)

`Name`

A name for the container.

Required: No

Type: String

`PortMappings`

A mapping of the container port to a host port. Port mappings enable containers to access ports on the host container instance to send or receive traffic.

Required: No

Type: List of Amazon EC2 Container Service TaskDefinition ContainerDefinitions PortMappings (p. 1141)

`Privileged`

Indicates whether the container is given full access to the host container instance.

Required: No

Type: Boolean

`ReadonlyRootFilesystem`

Indicates whether the container's root file system is mounted as read only.

Required: No

Type: Boolean

`Ulimits`

A list of ulimits to set in the container. The ulimits set constraints on how much resources a container can consume so that it doesn't deplete all available resources on the host.

Required: No

Type: List of Amazon EC2 Container Service TaskDefinition ContainerDefinitions Ulimit (p. 1142)

`User`

The user name to use inside the container.

Required: No

Type: String

`VolumesFrom`

The data volumes to mount from another container.

Required: No

Type: List of Amazon EC2 Container Service TaskDefinition ContainerDefinitions VolumesFrom (p. 1143)

`WorkingDirectory`

The working directory in the container in which to run commands.

Required: No

Type: String

Amazon EC2 Container Service TaskDefinition ContainerDefinitions Environment

`Environment` is a property of the Amazon EC2 Container Service TaskDefinition ContainerDefinitions (p. 1133) property that specifies environment variables for a container.

Syntax

JSON

```
{
  "Name" : String,
  "Value" : String
}
```

YAML

```
Name: String
Value: String
```

Properties

For more information about each property, see Task Definition Parameters in the *Amazon EC2 Container Service Developer Guide*.

`Name`

The name of the environment variable.

Required: Yes

Type: String

`Value`

The value of the environment variable.

Required: Yes

Type: String

Amazon EC2 Container Service TaskDefinition ContainerDefinitions HostEntry

`HostEntry` is a property of the Amazon EC2 Container Service TaskDefinition ContainerDefinitions (p. 1133) property that specifies the hostnames and IP address entries to add to the Amazon EC2 Container Service (Amazon ECS) container's `/etc/hosts` file.

Syntax

JSON

```
{
  "Hostname" : String,
  "IpAddress" : String
}
```

YAML

```
Hostname: String
IpAddress: String
```

Properties

`Hostname`

The hostname to use in the `/etc/hosts` file.

Required: Yes

Type: String

`IpAddress`

The IP address to use in the `/etc/hosts` file.

Required: Yes

Type: String

Amazon EC2 Container Service TaskDefinition ContainerDefinitions LogConfiguration

`LogConfiguration` is a property of the Amazon EC2 Container Service TaskDefinition ContainerDefinitions (p. 1133) property that configures a custom log driver for an Amazon EC2 Container Service (Amazon ECS) container.

Syntax

JSON

```
{
  "LogDriver" : String,
  "Options" : { String:String, ... }
}
```

YAML

```
LogDriver: String
Options:
  String: String
```

Properties

`LogDriver`

The log driver to use for the container. This parameter requires that your container instance uses Docker Remote API Version 1.18 or greater. For more information, see the `logDriver` content for the LogConfiguration data type in the *Amazon EC2 Container Service API Reference*.

Required: Yes

Type: String

`Options`

The configuration options to send to the log driver. This parameter requires that your container instance uses Docker Remote API Version 1.18 or greater.

Required: No

Type: Key-value pairs, with the option name as the key and the option value as the value.

Amazon EC2 Container Service TaskDefinition ContainerDefinitions MountPoints

`MountPoints` is a property of the Amazon EC2 Container Service TaskDefinition ContainerDefinitions (p. 1133) property that specifies the mount points for data volumes in a container.

Syntax

JSON

```
{
  "ContainerPath" : String,
  "SourceVolume" : String,
  "ReadOnly" : Boolean
}
```

YAML

```
ContainerPath: String
```

```
SourceVolume: String
ReadOnly: Boolean
```

Properties

For more information about each property, see Task Definition Parameters in the *Amazon EC2 Container Service Developer Guide*.

`ContainerPath`

The path on the container that indicates where you want to mount the volume.

Required: Yes

Type: String

`SourceVolume`

The name of the volume to mount.

Required: Yes

Type: String

`ReadOnly`

Indicates whether the container can write to the volume. If you specify **`true`**, the container has read-only access to the volume. If you specify **`false`**, the container can write to the volume. By default, the value is **`false`**.

Required: No

Type: Boolean

Amazon EC2 Container Service TaskDefinition ContainerDefinitions PortMappings

`PortMappings` is a property of the Amazon EC2 Container Service TaskDefinition ContainerDefinitions (p. 1133) property that maps a container port to a host port.

Syntax

JSON

```
{
  "ContainerPort" : Integer,
  "HostPort" : Integer,
  "Protocol" : String
}
```

YAML

```
ContainerPort: Integer
HostPort: Integer
Protocol: String
```

Properties

For more information about each property, see Task Definition Parameters in the *Amazon EC2 Container Service Developer Guide*.

`ContainerPort`

The port number on the container bound to the host port.

Required: Yes

Type: Integer

`HostPort`

The host port number on the container instance that you want to reserve for your container. You can specify a non-reserved host port for your container port mapping, omit the host port, or set the host port to `0`. If you specify a container port but no host port, your container host port is assigned automatically .

Don't specify a host port in the `49153` to `65535` port range; these ports are reserved for automatic assignment. Other reserved ports include `22` for SSH, `2375` and `2376` for Docker, and `51678` for the Amazon EC2 Container Service container agent. Don't specify a host port that is being used for a task —that port is reserved while the task is running.

Required: No

Type: Integer

`Protocol`

The protocol used for the port mapping. For valid values, see the `protocol` parameter in the *Amazon EC2 Container Service Developer Guide*. By default, AWS CloudFormation specifies `tcp`.

Required: No

Type: String

Amazon EC2 Container Service TaskDefinition ContainerDefinitions Ulimit

`Ulimit` is a property of the Amazon EC2 Container Service TaskDefinition ContainerDefinitions (p. 1133) property that specifies resource limits for an Amazon EC2 Container Service (Amazon ECS) container.

Syntax

JSON

```
{
  "HardLimit" : Integer,
  "Name" : String,
  "SoftLimit" : Integer
}
```

YAML

```
HardLimit: Integer
```

```
Name: String
SoftLimit: Integer
```

Properties

`HardLimit`

The hard limit for the ulimit type.

Required: Yes

Type: Integer

`Name`

The type of ulimit. For valid values, see the `name` content for the Ulimit data type in the *Amazon EC2 Container Service API Reference*.

Required: No

Type: String

`SoftLimit`

The soft limit for the ulimit type.

Required: Yes

Type: Integer

Amazon EC2 Container Service TaskDefinition ContainerDefinitions VolumesFrom

`VolumesFrom` is a property of the Amazon EC2 Container Service TaskDefinition ContainerDefinitions (p. 1133) property that mounts data volumes from other containers.

Syntax

JSON

```
{
  "SourceContainer" : String,
  "ReadOnly" : Boolean
}
```

YAML

```
SourceContainer: String
ReadOnly: Boolean
```

Properties

For more information about each property, see Task Definition Parameters in the *Amazon EC2 Container Service Developer Guide*.

`SourceContainer`

The name of the container that has the volumes to mount.

Required: Yes

Type: String

`ReadOnly`

Indicates whether the container can write to the volume. If you specify **`true`**, the container has read-only access to the volume. If you specify **`false`**, the container can write to the volume. By default, the value is **`false`**.

Required: No

Type: Boolean

Amazon EC2 Container Service TaskDefinition Volumes

`Volumes` is a property of the AWS::ECS::TaskDefinition (p. 675) resource that specifies a list of data volumes, which your containers can then access.

Syntax

JSON

```
{
  "Name" : String,
  "Host" : Host
}
```

YAML

```
Name: String
Host:
  Host
```

Properties

For more information about each property, see Task Definition Parameters in the *Amazon EC2 Container Service Developer Guide*.

`Name`

The name of the volume. To specify mount points in your container definitions, use the value of this property.

Required: Yes

Type: String

`Host`

Determines whether your data volume persists on the host container instance and at the location where it is stored.

Required: No

Type: Amazon EC2 Container Service TaskDefinition Volumes Host (p. 1145)

Amazon EC2 Container Service TaskDefinition Volumes Host

`Host` is a property of the Amazon EC2 Container Service TaskDefinition Volumes (p. 1144) property that specifies the data volume path on the host container instance.

Syntax

JSON

```
{
  "SourcePath" : String
}
```

YAML

```
SourcePath: String
```

Properties

For more information about each property, see Task Definition Parameters in the *Amazon EC2 Container Service Developer Guide*.

`SourcePath`

The data volume path on the host container instance.

If you don't specify this parameter, the Docker daemon assigns a path for you, but the data volume might not persist after the associated container stops running. If you do specify a path, the data volume persists at that location on the host container instance until you manually delete it.

Required: No

Type: String

Amazon Elastic File System FileSystem FileSystemTags

`FileSystemTags` is a property of the AWS::EFS::FileSystem (p. 679) resource that associates key-value pairs with a file system. You can use any of the following Unicode characters for keys and values: letters, digits, whitespace, _, ., /, =, +, and -.

Syntax

JSON

```
{
```

```
  "Key" : String,
  "Value" : String
}
```

YAML

```
Key: String
Value: String
```

Properties

`Key`

The key name of the tag. You can specify a value that is from 1 to 128 Unicode characters in length, but you cannot use the prefix `aws:`.

Required: No

Type: String

`Value`

The value of the tag key. You can specify a value that is from 0 to 128 Unicode characters in length.

Required: No

Type: String

Elastic Beanstalk Environment Tier Property Type

Describes the environment tier for an AWS::ElasticBeanstalk::Environment (p. 714) resource. For more information, see Environment Tiers in the *AWS Elastic Beanstalk Developer Guide*.

Syntax

JSON

```
{
   "Name" : String,
   "Type" : String,
   "Version" : String
}
```

YAML

```
Name: String
Type: String
Version: String
```

Members

`Name`

The name of the environment tier. You can specify `WebServer` or `Worker`.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Type`

The type of this environment tier. You can specify `Standard` for the `WebServer` tier or `SQS/HTTP` for the `Worker` tier.

Required: No

Type: String

Update requires: Replacement (p. 90)

`Version`

The version of this environment tier.

Required: No

Type: String

Update requires: No interruption (p. 90)

Elastic Beanstalk OptionSettings Property Type

`OptionSettings` is an embedded property of the AWS::ElasticBeanstalk::Environment (p. 714) and AWS::ElasticBeanstalk::ConfigurationTemplate (p. 711) resources. You use the `OptionSettings` property to specify an array of options for the Elastic Beanstalk environment.

Syntax

JSON

```
{
   "Namespace (p. 1147)" : String,
   "OptionName (p. 1148)" : String,
   "Value (p. 1148)" : String
}
```

YAML

```
Namespace (p. 1147): String
OptionName (p. 1148): String
Value (p. 1148): String
```

Members

`Namespace`

A unique namespace identifying the option's associated AWS resource. For a list of namespaces that you can use, see Configuration Options in the *AWS Elastic Beanstalk Developer Guide*.

Required: Yes

Type: String

`OptionName`

The name of the configuration option. For a list of options that you can use, see Configuration Options in the *AWS Elastic Beanstalk Developer Guide*.

Required: Yes

Type: String

`Value`

The value of the setting.

Required: Yes

Type: String

See Also

- ConfigurationOptionSetting in the *AWS Elastic Beanstalk Developer Guide*
- Option Values in the *AWS Elastic Beanstalk Developer Guide*

Elastic Beanstalk SourceBundle Property Type

The `SourceBundle` property is an embedded property of the AWS::ElasticBeanstalk::ApplicationVersion (p. 709) resource.

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
   "S3Bucket (p. 1148)" : String,
   "S3Key (p. 1149)" : String
}
```

YAML

```
S3Bucket (p. 1148): String
S3Key (p. 1149): String
```

Members

`S3Bucket`

The Amazon S3 bucket where the data is located.

Required: Yes

Type: String

`S3Key`

The Amazon S3 key where the data is located.

Required: Yes

Type: String

Elastic Beanstalk SourceConfiguration Property Type

Use settings from another Elastic Beanstalk configuration template for the AWS::ElasticBeanstalk::ConfigurationTemplate (p. 711) resource type.

Syntax

JSON

```
{
   "ApplicationName" : String,
   "TemplateName" : String
}
```

YAML

```
ApplicationName: String
TemplateName: String
```

Members

`ApplicationName`

The name of the Elastic Beanstalk application that contains the configuration template that you want to use.

Required: Yes

Type: String

`TemplateName`

The name of the configuration template.

Required: Yes

Type: String

Amazon ElastiCache ReplicationGroup NodeGroupConfiguration

`NodeGroupConfiguration` is a property of the AWS::ElastiCache::ReplicationGroup (p. 693) resource that configures an Amazon ElastiCache (ElastiCache) Redis cluster node group.

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "PrimaryAvailabilityZone" : String,
  "ReplicaAvailabilityZones" : [ String, ... ],
  "ReplicaCount" : Integer,
  "Slots" : String
}
```

YAML

```
PrimaryAvailabilityZone: String
ReplicaAvailabilityZones:
  - String
ReplicaCount: Integer
Slots: String
```

Properties

`PrimaryAvailabilityZone`

The Availability Zone where ElastiCache launches the node group's primary node.

Required: No

Type: String

`ReplicaAvailabilityZones`

A list of Availability Zones where ElastiCache launches the read replicas. The number of Availability Zones must match the value of the `ReplicaCount` property or, if you don't specify the `ReplicaCount` property, the replication group's `ReplicasPerNodeGroup` property.

Required: No

Type: List of strings

`ReplicaCount`

The number of read replica nodes in the node group.

Required: No

Type: Integer

`Slots`

A string of comma-separated values where the first set of values are the slot numbers (zero based), and the second set of values are the keyspaces for each slot. The following example specifies three slots (numbered 0, 1, and 2): `0,1,2,0-4999,5000-9999,10000-16,383`.

If you don't specify a value, ElastiCache allocates keys equally among each slot.

Required: No

Type: String

Elastic Load Balancing AccessLoggingPolicy

The `AccessLoggingPolicy` property describes where and how access logs are stored for the AWS::ElasticLoadBalancing::LoadBalancer (p. 719) resource.

Syntax

JSON

```
{
  "EmitInterval" : Integer,
  "Enabled" : Boolean,
  "S3BucketName" : String,
  "S3BucketPrefix" : String
}
```

YAML

```
EmitInterval: Integer
Enabled: Boolean
S3BucketName: String
S3BucketPrefix: String
```

Properties

`EmitInterval`

The interval for publishing access logs in minutes. You can specify an interval of either 5 minutes or 60 minutes.

Required: No

Type: Integer

`Enabled`

Whether logging is enabled for the load balancer.

Required: Yes

Type: Boolean

`S3BucketName`

The name of an Amazon S3 bucket where access log files are stored.

Required: Yes

Type: String

`S3BucketPrefix`

A prefix for the all log object keys, such as `my-load-balancer-logs/prod`. If you store log files from multiple sources in a single bucket, you can use a prefix to distinguish each log file and its source.

Required: No

Type: String

ElasticLoadBalancing AppCookieStickinessPolicy Type

The AppCookieStickinessPolicy type is an embedded property of the AWS::ElasticLoadBalancing::LoadBalancer (p. 719) type.

Syntax

JSON

```
{
   "CookieName (p. 1152)" : String,
   "PolicyName (p. 1152)" : String
}
```

YAML

```
CookieName (p. 1152): String
PolicyName (p. 1152): String
```

Properties

`CookieName`

Name of the application cookie used for stickiness.

Required: Yes

Type: String

`PolicyName`

The name of the policy being created. The name must be unique within the set of policies for this Load Balancer.

Note
To associate this policy with a listener, include the policy name in the listener's PolicyNames (p. 1156) property.

Required: Yes

Type: String

See Also

- Sample template snippets in the Examples section of AWS::ElasticLoadBalancing::LoadBalancer (p. 719).
- CreateAppCookieStickinessPolicyin the *Elastic Load Balancing API Reference version 2012-06-01*

Elastic Load Balancing ConnectionDrainingPolicy

The `ConnectionDrainingPolicy` property describes how deregistered or unhealthy instances handle in-flight requests for the AWS::ElasticLoadBalancing::LoadBalancer (p. 719) resource. Connection draining ensures that the load balancer completes serving all in-flight requests made to a registered instance when the instance is deregistered or becomes unhealthy. Without connection draining, the load balancer closes connections to deregistered or unhealthy instances, and any in-flight requests are not completed.

For more information about connection draining and default values, see Enable or Disable Connection Draining for Your Load Balancer in the *Elastic Load Balancing User Guide*.

Syntax

JSON

```
{
   "Enabled" : Boolean,
   "Timeout" : Integer
}
```

YAML

```
Enabled: Boolean
Timeout: Integer
```

Properties

`Enabled`

Whether or not connection draining is enabled for the load balancer.

Required: Yes

Type: Boolean

`Timeout`

The time in seconds after the load balancer closes all connections to a deregistered or unhealthy instance.

Required: No

Type: Integer

Elastic Load Balancing ConnectionSettings

`ConnectionSettings` is a property of the AWS::ElasticLoadBalancing::LoadBalancer (p. 719) resource that describes how long the front-end and back-end connections of your load balancer can remain idle. For more information, see Configure Idle Connection Timeout in the *Elastic Load Balancing User Guide*.

Syntax

JSON

```
{
  "IdleTimeout" : Integer
}
```

YAML

```
IdleTimeout: Integer
```

Properties

`IdleTimeout`

The time (in seconds) that a connection to the load balancer can remain idle, which means no data is sent over the connection. After the specified time, the load balancer closes the connection.

Required: Yes

Type: Integer

ElasticLoadBalancing HealthCheck Type

The ElasticLoadBalancing HealthCheck is an embedded property of the AWS::ElasticLoadBalancing::LoadBalancer (p. 719) type.

Syntax

JSON

```
{
   "HealthyThreshold (p. 1154)" : String,
   "Interval (p. 1154)" : String,
   "Target (p. 1155)" : String,
   "Timeout (p. 1155)" : String,
   "UnhealthyThreshold (p. 1155)" : String
}
```

YAML

```
HealthyThreshold (p. 1154): String
Interval (p. 1154): String
Target (p. 1155): String
Timeout (p. 1155): String
UnhealthyThreshold (p. 1155): String
```

Properties

`HealthyThreshold`

Specifies the number of consecutive health probe successes required before moving the instance to the Healthy state.

Required: Yes

Type: String

`Interval`

Specifies the approximate interval, in seconds, between health checks of an individual instance.

Required: Yes

Type: String

`Target`

Specifies the instance's protocol and port to check. The protocol can be TCP, HTTP, HTTPS, or SSL. The range of valid ports is 1 through 65535.

Required: Yes

Type: String

> **Note**
> For TCP and SSL, you specify a port pair. For example, you can specify `TCP:5000` or `SSL:5000`. The health check attempts to open a TCP or SSL connection to the instance on the port that you specify. If the health check fails to connect within the configured timeout period, the instance is considered unhealthy.
> For HTTP or HTTPS, you specify a port and a path to ping (*`HTTP or HTTPS`*:*`port`*/*`PathToPing`*). For example, you can specify `HTTP:80/weather/us/wa/seattle`. In this case, an HTTP GET request is issued to the instance on the given port and path. If the health check receives any response other than `200 OK` within the configured timeout period, the instance is considered unhealthy. The total length of the HTTP or HTTPS ping target cannot be more than 1024 16-bit Unicode characters.

`Timeout`

Specifies the amount of time, in seconds, during which no response means a failed health probe. This value must be less than the value for `Interval`.

Required: Yes

Type: String

`UnhealthyThreshold`

Specifies the number of consecutive health probe failures required before moving the instance to the Unhealthy state.

Required: Yes

Type: String

ElasticLoadBalancing LBCookieStickinessPolicy Type

The LBCookieStickinessPolicy type is an embedded property of the AWS::ElasticLoadBalancing::LoadBalancer (p. 719) type.

Syntax

JSON

```
{
  "CookieExpirationPeriod (p. 1156)" : String,
  "PolicyName (p. 1156)" : String
}
```

YAML

```
CookieExpirationPeriod (p. 1156): String
```

```
PolicyName (p. 1156): String
```

Properties

`CookieExpirationPeriod`

The time period, in seconds, after which the cookie should be considered stale. If this parameter isn't specified, the sticky session will last for the duration of the browser session.

Required: No

Type: String

`PolicyName`

The name of the policy being created. The name must be unique within the set of policies for this load balancer.

> **Note**
> To associate this policy with a listener, include the policy name in the listener's PolicyNames (p. 1156) property.

See Also

- Sample template snippets in the Examples section of AWS::ElasticLoadBalancing::LoadBalancer (p. 719).
- CreateLBCookieStickinessPolicy in the *Elastic Load Balancing API Reference version 2012-06-01*

ElasticLoadBalancing Listener Property Type

The `Listener` property is an embedded property of the `AWS::ElasticLoadBalancing::LoadBalancer (p. 719)` type.

Syntax

JSON

```
{
  "InstancePort (p. 1157)" : String,
  "InstanceProtocol (p. 1157)" : String,
  "LoadBalancerPort (p. 1157)" : String,
  "PolicyNames (p. 1157)" :  [ String, ... ],
  "Protocol (p. 1157)" : String,
  "SSLCertificateId (p. 1158)" : String
}
```

YAML

```
InstancePort (p. 1157): String
InstanceProtocol (p. 1157): String
LoadBalancerPort (p. 1157): String
PolicyNames (p. 1157):
  - String
Protocol (p. 1157): String
SSLCertificateId (p. 1158): String
```

Properties

`InstancePort`

Specifies the TCP port on which the instance server listens. You can't modify this property during the life of the load balancer.

Required: Yes

Type: String

`InstanceProtocol`

Specifies the protocol to use for routing traffic to back-end instances: HTTP, HTTPS, TCP, or SSL. You can't modify this property during the life of the load balancer.

Required: No

Type: String

Note

- If the front-end protocol is HTTP or HTTPS, `InstanceProtocol` must be on the same protocol layer (HTTP or HTTPS). Likewise, if the front-end protocol is TCP or SSL, `InstanceProtocol` must be TCP or SSL. By default, Elastic Load Balancing sets the instance protocol to HTTP or TCP.
- If there is another `Listener` with the same `InstancePort` whose `InstanceProtocol` is secure, (using HTTPS or SSL), the `InstanceProtocol` of the `Listener` must be secure (using HTTPS or SSL). If there is another `Listener` with the same `InstancePort` whose `InstanceProtocol` is HTTP or TCP, the `InstanceProtocol` of the `Listener` must be either HTTP or TCP.

`LoadBalancerPort`

Specifies the external load balancer port number. You can't modify this property during the life of the load balancer.

Required: Yes

Type: String

`PolicyNames`

A list of `ElasticLoadBalancing` policy (p. 1158) names to associate with the `Listener`. Specify only policies that are compatible with a `Listener`. For more information, see `DescribeLoadBalancerPolicyTypes` in the *Elastic Load Balancing API Reference version 2012-06-01*.

Note

By default, Elastic Load Balancing associates the latest predefined policy with your load balancer. When a new predefined policy is added, we recommend that you update your load balancer to use the new predefined policy. Alternatively, you can select a different predefined security policy or create a custom policy. To create a security policy, use the `Policies` property of the AWS::ElasticLoadBalancing::LoadBalancer (p. 719) resource.

Required: No

Type: List of strings

`Protocol`

Specifies the load balancer transport protocol to use for routing: HTTP, HTTPS, TCP or SSL. You can't modify this property during the life of the load balancer.

Required: Yes

Type: String

`SSLCertificateId`

The ARN of the SSL certificate to use. For more information about SSL certificates, see Managing Server Certificates in the *AWS Identity and Access Management User Guide*.

Required: No

Type: String

ElasticLoadBalancing Policy Type

The ElasticLoadBalancing policy type is an embedded property of the AWS::ElasticLoadBalancing::LoadBalancer (p. 719) resource. You associate policies with a listener (p. 1156) by referencing a policy's name in the listener's `PolicyNames` property.

Syntax

JSON

```
{
  "Attributes (p. 1158)" : [ { "Name" : String, "Value" : String }, ... ],
  "InstancePorts (p. 1158)" : [ String, ... ],
  "LoadBalancerPorts (p. 1159)" : [ String, ... ],
  "PolicyName (p. 1159)" : String,
  "PolicyType (p. 1159)" : String
}
```

YAML

```
Attributes (p. 1158):
  -
    "Name" : String
    "Value" : String
InstancePorts (p. 1158):
  - String
LoadBalancerPorts (p. 1159):
  - String
PolicyName (p. 1159): String
PolicyType (p. 1159): String
```

Properties

`Attributes`

A list of arbitrary attributes for this policy. If you don't need to specify any policy attributes, specify an empty list (`[]`).

Required: Yes

Type: List of JSON name-value pairs.

`InstancePorts`

A list of instance ports for the policy. These are the ports associated with the back-end server.

Required: No

Type: List of String

`LoadBalancerPorts`

A list of external load balancer ports for the policy.

Required: Only for some policies. For more information, see the *Elastic Load Balancing Developer Guide*.

Type: List of String

`PolicyName`

A name for this policy that is unique to the load balancer.

Required: Yes

Type: String

`PolicyType`

The name of the policy type for this policy. This must be one of the types reported by the Elastic Load Balancing DescribeLoadBalancerPolicyTypes action.

Required: Yes

Type: String

Examples

This example shows a snippet of the policies section of an elastic load balancer listener.

```
"Policies" : [
   {
      "PolicyName" : "MySSLNegotiationPolicy",
      "PolicyType" : "SSLNegotiationPolicyType",
      "Attributes" : [
         { "Name" : "Protocol-TLSv1", "Value" : "true" },
         { "Name" : "Protocol-SSLv3", "Value" : "false" },
         { "Name" : "DHE-RSA-AES256-SHA", "Value" : "true" } ]
   }, {
      "PolicyName" : "MyAppCookieStickinessPolicy",
      "PolicyType" : "AppCookieStickinessPolicyType",
      "Attributes" : [
         { "Name" : "CookieName", "Value" : "MyCookie"} ]
   }, {
      "PolicyName" : "MyPublicKeyPolicy",
      "PolicyType" : "PublicKeyPolicyType",
      "Attributes" : [ {
         "Name" : "PublicKey",
         "Value" : { "Fn::Join" : [
            "\n", [
               "MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDh/51Aohx5VrpmlfGHZCzciMBa",
               "fkHve+MQYYJcxmNUKMdsWnz9WtVfKxxWUU7Cfor4lorYmENGCG8FWqCoLDMFs7pN",
               "yGEtpsrlKhzZWtgY1d7eGrUrBil03bI90E2KW0j4qAwGYAC8xixOkNClicojeEz4",
               "f4rr3sUf+ZBSsuMEuwIDAQAB" ]
         ] }
      } ]
   }, {
      "PolicyName" : "MyBackendServerAuthenticationPolicy",
      "PolicyType" : "BackendServerAuthenticationPolicyType",
      "Attributes" : [
         { "Name" : "PublicKeyPolicyName", "Value" : "MyPublicKeyPolicy" } ],
```

```
      "InstancePorts" : [ "8443" ]
   }
]
```

This example shows a snippet of the policies section of an elastic load balancer using proxy protocol.

```
"Policies" : [{
   "PolicyName" : "EnableProxyProtocol",
   "PolicyType" : "ProxyProtocolPolicyType",
   "Attributes" : [{
      "Name"  : "ProxyProtocol",
      "Value" : "true"
   }],
   "InstancePorts" : [{"Ref" : "WebServerPort"}]
}]
```

In the following snippet, the load balancer uses a predefined security policy. These predefined policies are provided by Elastic Load Balancing. For more information, see SSL Security Policies in the *Elastic Load Balancing User Guide*.

```
"Policies" : [{
   "PolicyName" : "ELBSecurityPolicyName",
   "PolicyType" : "SSLNegotiationPolicyType",
   "Attributes" : [{
      "Name"  : "Reference-Security-Policy",
      "Value" : "ELBSecurityPolicy-2014-10"
   }]
}]
```

See Also

- AWS::ElasticLoadBalancing::LoadBalancer (p. 719)
- ElasticLoadBalancing AppCookieStickinessPolicy Type (p. 1152)
- ElasticLoadBalancing LBCookieStickinessPolicy Type (p. 1155)

Elastic Load Balancing Listener Certificates

`Certificates` is a property of the AWS::ElasticLoadBalancingV2::Listener (p. 731) resource that specifies the SSL server certificate that Elastic Load Balancing will deploy on an listener. For more information, see Create an HTTPS Listener for Your Application Load Balancer in the *Application Load Balancers Guide*.

Syntax

JSON

```
{
  "CertificateArn" : String
}
```

YAML

```
CertificateArn: String
```

Properties

`CertificateArn`

The Amazon Resource Name (ARN) of the certificate to associate with the listener.

Required: No

Type: String

Elastic Load Balancing Listener DefaultActions

`DefaultActions` is a property of the AWS::ElasticLoadBalancingV2::Listener (p. 731) resource that specifies the default actions the Elastic Load Balancing listener takes when handling incoming requests.

Syntax

JSON

```
{
  "TargetGroupArn" : String,
  "Type" : String
}
```

YAML

```
TargetGroupArn: String
Type: String
```

Properties

`TargetGroupArn`

The Amazon Resource Name (ARN) of the target group to which Elastic Load Balancing routes the traffic.

Required: Yes

Type: String

`Type`

The type of action. For valid values, see the `Type` contents for the Action data type in the *Elastic Load Balancing API Reference version 2015-12-01*.

Required: Yes

Type: String

Elastic Load Balancing ListenerRule Actions

`Actions` is a property of the AWS::ElasticLoadBalancingV2::ListenerRule (p. 733) resource that specifies the actions an Elastic Load Balancing listener takes when an incoming request meets a listener rule's condition.

Syntax

JSON

```
{
  "TargetGroupArn" : String,
  "Type" : String
}
```

YAML

```
TargetGroupArn: String
Type: String
```

Properties

`TargetGroupArn`

The Amazon Resource Name (ARN) of the target group to which Elastic Load Balancing routes the traffic.

Required: Yes

Type: String

`Type`

The type of action. For valid values, see the `Type` contents for the Action data type in the *Elastic Load Balancing API Reference version 2015-12-01*.

Required: Yes

Type: String

Elastic Load Balancing ListenerRule Conditions

`Conditions` is a property of the AWS::ElasticLoadBalancingV2::ListenerRule (p. 733) resource that specifies the conditions when an Elastic Load Balancing listener rule takes effect.

Syntax

JSON

```
{
  "Field" : String,
  "Values" : [ String, ... ]
}
```

YAML

```
Field: String
Values:
  - String
```

Properties

`Field`

The name of the condition that you want to define, such as `path-pattern` (which forwards requests based on the URL of the request).

For valid values, see the `Field` contents for the RuleCondition data type in the *Elastic Load Balancing API Reference version 2015-12-01*.

Required: No

Type: String

`Values`

The value for the field that you specified in the `Field` property.

Required: No

Type: List of strings

Elastic Load Balancing LoadBalancer LoadBalancerAttributes

`LoadBalancerAttributes` is a property of the AWS::ElasticLoadBalancingV2::LoadBalancer (p. 736) resource that configures settings for an Elastic Load Balancing Application load balancer. For more information, see Load Balancer Attributes in the *Application Load Balancers Guide*.

Syntax

JSON

```
{
  "Key" : String,
  "Value" : String
}
```

YAML

```
Key: String
Value: String
```

Properties

`Key`

The name of an attribute that you want to configure. For the list of attributes that you can configure, see the `Key` contents for the LoadBalancerAttribute data type in the *Elastic Load Balancing API Reference version 2015-12-01*.

Required: No

Type: String

`Value`

A value for the attribute.

Required: No

Type: String

Elastic Load Balancing TargetGroup Matcher

`Matcher` is a property of the AWS::ElasticLoadBalancingV2::TargetGroup (p. 739) resource that specifies the HTTP codes that healthy targets must use when responding to an Elastic Load Balancing health check.

Syntax

JSON

```
{
  "HttpCode" : String
}
```

YAML

```
HttpCode: String
```

Properties

`HttpCode`

The HTTP codes that a healthy target must use when responding to a health check, such as `200,202` or `200-399`.

For valid and default values, see the `HttpCode` contents for the Matcher data type in the *Elastic Load Balancing API Reference version 2015-12-01*.

Required: No

Type: String

Elastic Load Balancing TargetGroup TargetDescription

`TargetDescription` is a property of the AWS::ElasticLoadBalancingV2::TargetGroup (p. 739) resource that specifies a target to add to an Elastic Load Balancing target group.

Syntax

JSON

```
{
  "Id" : String,
  "Port" : Integer
}
```

YAML

```
Id: String
Port: Integer
```

Properties

`Id`

The ID of the target, such as an EC2 instance ID.

Required: Yes

Type: String

`Port`

The port number on which the target is listening for traffic.

Required: No

Type: Integer

Elastic Load Balancing TargetGroup TargetGroupAttributes

`TargetGroupAttributes` is a property of the AWS::ElasticLoadBalancingV2::TargetGroup (p. 739) resource that configures settings for an Elastic Load Balancing target group. For more information, see Target Group Attributes in the *Application Load Balancers Guide*.

Syntax

JSON

```
{
  "Key" : String,
  "Value" : String
}
```

YAML

```
Key: String
Value: String
```

Properties

`Key`

The name of the attribute that you want to configure. For the list of attributes that you can configure, see the `Key` contents for the TargetGroupAttribute data type in the *Elastic Load Balancing API Reference version 2015-12-01*.

Required: No

Type: String

`Value`

A value for the attribute.

Required: No

Type: String

Amazon Elasticsearch Service Domain EBSOptions

`EBSOptions` is a property of the (p.) resource that configures the Amazon Elastic Block Store (Amazon EBS) volumes that are attached to data nodes in the Amazon Elasticsearch Service (Amazon ES) domain.

Syntax

JSON

```
{
  "EBSEnabled" : Boolean,
  "Iops" : Integer,
  "VolumeSize" : Integer,
  "VolumeType" : String
}
```

YAML

```
EBSEnabled: Boolean
Iops: Integer
VolumeSize: Integer
VolumeType: String
```

Properties

`EBSEnabled`

Specifies whether Amazon EBS volumes are attached to data nodes in the Amazon ES domain.

Required: No

Type: Boolean

`Iops`

The number of I/O operations per second (IOPS) that the volume supports. This property applies only to the Provisioned IOPS (SSD) EBS volume type.

Required: No

Type: Integer

`VolumeSize`

The size of the EBS volume for each data node. The minimum and maximum size of an EBS volume depends on the EBS volume type and the instance type to which it is attached. For more information, see Configuring EBS-based Storage in the *Amazon Elasticsearch Service Developer Guide*.

Required: No

Type: Integer

`VolumeType`

The EBS volume type to use with the Amazon ES domain, such as `standard`, `gp2`, or `io1`. For more information about each type, see Amazon EBS Volume Types in the *Amazon EC2 User Guide for Linux Instances*.

Required: No

Type: String

Amazon Elasticsearch Service Domain ElasticsearchClusterConfig

`ElasticsearchClusterConfig` is a property of the (p.) resource that configures the cluster of an Amazon Elasticsearch Service (Amazon ES) domain.

Syntax

JSON

```
{
  "DedicatedMasterCount" : Integer,
  "DedicatedMasterEnabled" : Boolean,
  "DedicatedMasterType" : String,
  "InstanceCount" : Integer,
  "InstanceType" : String,
  "ZoneAwarenessEnabled" : Boolean
}
```

YAML

```
DedicatedMasterCount: Integer
DedicatedMasterEnabled: Boolean
DedicatedMasterType: String
InstanceCount: Integer
InstanceType: String
ZoneAwarenessEnabled: Boolean
```

Properties

`DedicatedMasterCount`

The number of instances to use for the master node.

If you specify this property, you must specify `true` for the `DedicatedMasterEnabled` property

Required: No

Type: Integer

`DedicatedMasterEnabled`

Indicates whether to use a dedicated master node for the Amazon ES domain. A dedicated master node is a cluster node that performs cluster management tasks, but doesn't hold data or respond to data upload requests. Dedicated master nodes offload cluster management tasks to increase the stability of your search clusters.

Required: No

Type: Boolean

`DedicatedMasterType`

The hardware configuration of the computer that hosts the dedicated master node, such as `m3.medium.elasticsearch`. For valid values, see Configuring Amazon ES Domains in the *Amazon Elasticsearch Service Developer Guide*.

If you specify this property, you must specify `true` for the `DedicatedMasterEnabled` property

Required: No

Type: String

`InstanceCount`

The number of data nodes (instances) to use in the Amazon ES domain.

Required: No

Type: Integer

`InstanceType`

The instance type for your data nodes, such as `m3.medium.elasticsearch`. For valid values, see Configuring Amazon ES Domains in the *Amazon Elasticsearch Service Developer Guide*.

Required: No

Type: String

`ZoneAwarenessEnabled`

Indicates whether to enable zone awareness for the Amazon ES domain. When you enable zone awareness, Amazon ES allocates the nodes and replica index shards that belong to a cluster across two Availability Zones (AZs) in the same region to prevent data loss and minimize downtime in the event of node or data center failure. Don't enable zone awareness if your cluster has no replica index shards or is a single-node cluster. For more information, see Enabling Zone Awareness in the *Amazon Elasticsearch Service Developer Guide*.

Required: No

Type: Boolean

Amazon Elasticsearch Service Domain SnapshotOptions

`SnapshotOptions` is a property of the (p.) resource that configures the automated snapshot of Amazon Elasticsearch Service (Amazon ES) domain indices.

Syntax

JSON

```
{
  "AutomatedSnapshotStartHour" : Integer
```

```
}
```

YAML

```
AutomatedSnapshotStartHour: Integer
```

Properties

`AutomatedSnapshotStartHour`

The hour in UTC during which the service takes an automated daily snapshot of the indices in the Amazon ES domain. For example, if you specify `0`, Amazon ES takes an automated snapshot everyday between midnight and 1 am. You can specify a value between `0` and `23`.

Required: No

Type: Integer

Amazon EMR Cluster Application

`Application` is a property of the AWS::EMR::Cluster (p. 750) resource that adds an Amazon EMR (Amazon EMR) application bundle or third-party software to an Amazon EMR cluster.

Syntax

JSON

```
{
  "AdditionalInfo" : { String:String, ... },
  "Args" : [ String, ... ],
  "Name" : String,
  "Version" : String
}
```

YAML

```
AdditionalInfo:
  String: String
Args:
  - String
Name: String
Version: String
```

Properties

`AdditionalInfo`

Metadata about third-party applications that third-party vendors use for testing purposes.

Required: No

Type: String-to-string map

`Args`

Arguments that Amazon EMR passes to the application.

Required: No

Type: List of strings

`Name`

The name of the application to add to your cluster, such as `Hadoop` or `Hive`. For valid values, see the Applications parameter in the *Amazon EMR API Reference*.

Required: No

Type: String

`Version`

The version of the application.

Required: No

Type: String

Amazon EMR Cluster BootstrapActionConfig

`BootstrapActionConfig` is a property of the AWS::EMR::Cluster (p. 750) resource that specifies bootstrap actions that Amazon EMR (Amazon EMR) runs before it installs applications on the cluster nodes.

Syntax

JSON

```
{
  "Name" : String,
  "ScriptBootstrapAction" : ScriptBootstrapAction
}
```

YAML

```
Name: String
ScriptBootstrapAction: ScriptBootstrapAction
```

Properties

`Name`

The name of the bootstrap action to add to your cluster.

Required: Yes

Type: String

`ScriptBootstrapAction`

The script that the bootstrap action runs.

Required: Yes

Type: Amazon EMR Cluster BootstrapActionConfig ScriptBootstrapActionConfig (p. 1171)

Amazon EMR Cluster BootstrapActionConfig ScriptBootstrapActionConfig

`ScriptBootstrapActionConfig` is a property of the Amazon EMR Cluster BootstrapActionConfig (p. 1170) property that specifies the arguments and location of the bootstrap script that Amazon EMR (Amazon EMR) runs before it installs applications on the cluster nodes.

Syntax

JSON

```
{
  "Args" : [ String, ... ],
  "Path" : String
}
```

YAML

```
Args:
  - String
Path: String
```

Properties

`Args`

A list of command line arguments to pass to the bootstrap action script.

Required: No

Type: List of strings

`Path`

The location of the script that Amazon EMR runs during a bootstrap action. Specify a location in an S3 bucket or your local file system.

Required: Yes

Type: String

Amazon EMR Cluster Configuration

`Configuration` is a property of the AWS::EMR::Cluster (p. 750) resource that specifies the software configuration of an Amazon EMR (Amazon EMR) cluster. For example configurations, see Configuring Applications in the *Amazon EMR Release Guide*.

Syntax

JSON

```
{
  "Classification" : String,
  "ConfigurationProperties" : { String:String, ... },
```

```
  "Configurations" : [ Configuration, ... ]
}
```

YAML

```
Classification: String
ConfigurationProperties:
  String: String
Configurations:
  - Configuration
```

Properties

`Classification`

The name of an application-specific configuration file. For more information see, Configuring Applications in the *Amazon EMR Release Guide*.

Required: No

Type: String

`ConfigurationProperties`

The settings that you want to change in the application-specific configuration file. For more information see, Configuring Applications in the *Amazon EMR Release Guide*.

Required: No

Type: String-to-string map

`Configurations`

A list of configurations to apply to this configuration. You can nest configurations so that a single configuration can have its own configurations. In other words, you can configure a configuration. For more information see, Configuring Applications in the *Amazon EMR Release Guide*.

Required: No

Type: List of Amazon EMR Cluster Configuration (p. 1171)

Amazon EMR Cluster JobFlowInstancesConfig

`JobFlowInstancesConfig` is a property of the AWS::EMR::Cluster (p. 750) resource that configures the EC2 instances (nodes) that will run jobs in an Amazon EMR (Amazon EMR) cluster.

Syntax

JSON

```
{
  "AdditionalMasterSecurityGroups" : [ String, ... ],
  "AdditionalSlaveSecurityGroups" : [ String, ... ],
  "CoreInstanceGroup" : InstanceGroupConfig,
  "Ec2KeyName" : String,
  "Ec2SubnetId" : String,
  "EmrManagedMasterSecurityGroup" : String,
  "EmrManagedSlaveSecurityGroup" : String,
```

```
  "HadoopVersion" : String,
  "MasterInstanceGroup" : InstanceGroupConfig,
  "Placement" : Placement,
  "ServiceAccessSecurityGroup" : String,
  "TerminationProtected" : Boolean
}
```

YAML

```
AdditionalMasterSecurityGroups:
  - String
AdditionalSlaveSecurityGroups:
  - String
CoreInstanceGroup:
  InstanceGroupConfig
Ec2KeyName: String
Ec2SubnetId: String
EmrManagedMasterSecurityGroup: String
EmrManagedSlaveSecurityGroup: String
HadoopVersion: String
MasterInstanceGroup:
  InstanceGroupConfig
Placement:
  Placement
ServiceAccessSecurityGroup: String
TerminationProtected: Boolean
```

Properties

`AdditionalMasterSecurityGroups`

A list of additional EC2 security group IDs to assign to the master instance (master node) in your Amazon EMR cluster. Use this property to supplement the rules specified by the Amazon EMR managed master security group.

Required: No

Type: List of strings

`AdditionalSlaveSecurityGroups`

A list of additional EC2 security group IDs to assign to the slave instances (slave nodes) in your Amazon EMR cluster. Use this property to supplement the rules specified by the Amazon EMR managed slave security group.

Required: No

Type: List of strings

`CoreInstanceGroup`

The settings for the core instances in your Amazon EMR cluster.

Required: Yes

Type: Amazon EMR Cluster JobFlowInstancesConfig InstanceGroupConfig (p. 1175)

`Ec2KeyName`

The name of an Amazon Elastic Compute Cloud (Amazon EC2) key pair, which you can use to access the instances in your Amazon EMR cluster.

Required: No

Type: String

`Ec2SubnetId`

The ID of a subnet where you want to launch your instances.

Required: No

Type: String

`EmrManagedMasterSecurityGroup`

The ID of an EC2 security group (managed by Amazon EMR) that is assigned to the master instance (master node) in your Amazon EMR cluster.

Required: No

Type: String

`EmrManagedSlaveSecurityGroup`

The ID of an EC2 security group (managed by Amazon EMR) that is assigned to the slave instances (slave nodes) in your Amazon EMR cluster.

Required: No

Type: String

`HadoopVersion`

The Hadoop version for the job flow. For valid values, see the HadoopVersion parameter in the *Amazon EMR API Reference*.

Required: No

Type: String

`MasterInstanceGroup`

The settings for the master instance (master node).

Required: Yes

Type: Amazon EMR Cluster JobFlowInstancesConfig InstanceGroupConfig (p. 1175)

`Placement`

The Availability Zone (AZ) in which the job flow runs.

Required: No

Type: Amazon EMR Cluster JobFlowInstancesConfig PlacementType (p. 1176)

`ServiceAccessSecurityGroup`

The ID of an EC2 security group (managed by Amazon EMR) that services use to access clusters in private subnets.

Required: No

Type: String

`TerminationProtected`

Indicates whether to prevent the EC2 instances from being terminated by an API call or user intervention. If you want to delete a stack with protected instances, update this value to `false` before you delete the stack. By default, AWS CloudFormation sets this property to `false`.

Required: No

Type: Boolean

Amazon EMR Cluster JobFlowInstancesConfig InstanceGroupConfig

`InstanceGroupConfig` is a property of the `CoreInstanceGroup` and `MasterInstanceGroup` properties of the job flow instances configuration (p. 1172). The `InstanceGroupConfig` property specifies the settings for instances (nodes) in the core and master instance groups of an Amazon EMR (Amazon EMR) cluster.

Syntax

JSON

```
{
  "BidPrice" : String,
  "Configurations" : [ Configuration, ... ],
  "EbsConfiguration" : EBSConfiguration,
  "InstanceCount" : Integer,
  "InstanceType" : String,
  "Market" : String,
  "Name" : String
}
```

YAML

```
BidPrice: String
Configurations:
  - Configuration
EbsConfiguration:
  EBSConfiguration
InstanceCount: Integer
InstanceType: String
Market: String
Name: String
```

Properties

`BidPrice`

When launching instances as Spot Instances, the bid price in USD for each EC2 instance in the instance group.

Required: No

Type: String

`Configurations`

A list of configurations to apply to this instance group. For more information see, Configuring Applications in the *Amazon EMR Release Guide*.

Required: No

Type: List of Amazon EMR Cluster Configuration (p. 1171)

`EbsConfiguration`

Configures Amazon Elastic Block Store (Amazon EBS) storage volumes to attach to your instances.

Required: No

Type: Amazon EMR EbsConfiguration (p. 1177)

Update requires: Replacement (p. 90)

`InstanceCount`

The number of instances to launch in the instance group.

Required: Yes

Type: Integer

`InstanceType`

The EC2 instance type for all instances in the instance group. For more information, see Instance Configurations in the *Amazon EMR Management Guide*.

Required: Yes

Type: String

`Market`

The type of marketplace from which your instances are provisioned into this group, either `ON_DEMAND` or `SPOT`. For more information, see Amazon EC2 Purchasing Options.

Required: No

Type: String

`Name`

A name for the instance group.

Required: No

Type: String

Amazon EMR Cluster JobFlowInstancesConfig PlacementType

`PlacementType` is a property of the Amazon EMR Cluster JobFlowInstancesConfig (p. 1172) property that specifies the Availability Zone (AZ) in which the job flow runs.

Syntax

JSON

```
{
  "AvailabilityZone" : String
}
```

YAML

```
AvailabilityZone: String
```

Properties

`AvailabilityZone`

The Amazon Elastic Compute Cloud (Amazon EC2) AZ for the job flow. For more information, see http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-regions-availability-zones.html in the *Amazon EC2 User Guide for Linux Instances*.

Required: Yes

Type: String

Amazon EMR EbsConfiguration

`EbsConfiguration` is a property of the Amazon EMR Cluster JobFlowInstancesConfig InstanceGroupConfig (p. 1175) property and the AWS::EMR::InstanceGroupConfig (p. 755) resource that defines Amazon Elastic Block Store (Amazon EBS) storage volumes to attach to your Amazon EMR (Amazon EMR) instances.

Syntax

JSON

```
{
  "EbsBlockDeviceConfigs" : [ EbsBlockDeviceConfig, ... ],
  "EbsOptimized" : Boolean
}
```

YAML

```
EbsBlockDeviceConfigs:
  - EbsBlockDeviceConfig
EbsOptimized: Boolean
```

Properties

`EbsBlockDeviceConfigs`

Configures the block storage devices that are associated with your EMR instances.

Required: No

Type: List of Amazon EMR EbsConfiguration EbsBlockDeviceConfigs (p. 1178)

`EbsOptimized`

Indicates whether the instances are optimized for Amazon EBS I/O. This optimization provides dedicated throughput to Amazon EBS and an optimized configuration stack to provide optimal EBS I/O performance. For more information about fees and supported instance types, see EBS-Optimized Instances in the *Amazon EC2 User Guide for Linux Instances*.

Required: No

Type: Boolean

Amazon EMR EbsConfiguration EbsBlockDeviceConfigs

`EbsBlockDeviceConfigs` is a property of the Amazon EMR EbsConfiguration (p. 1177) property that defines the settings for the Amazon Elastic Block Store (Amazon EBS) volumes that Amazon EMR (Amazon EMR) associates with your instances.

Syntax

JSON

```
{
  "VolumeSpecification" : VolumeSpecification,
  "VolumesPerInstance" : Integer
}
```

YAML

```
VolumeSpecification:
  VolumeSpecification
VolumesPerInstance: Integer
```

Properties

`VolumeSpecification`

The settings for the Amazon EBS volumes.

Required: Yes

Type: Amazon EMR EbsConfiguration EbsBlockDeviceConfig VolumeSpecification (p. 1178)

`VolumesPerInstance`

The number of Amazon EBS volumes that you want to create for each instance in the EMR cluster or instance group.

Required: No

Type: Integer

Amazon EMR EbsConfiguration EbsBlockDeviceConfig VolumeSpecification

`VolumeSpecification` is a property of the Amazon EMR EbsConfiguration (p. 1177) property that configures the Amazon Elastic Block Store (Amazon EBS) volumes that Amazon EMR (Amazon EMR) associates with your instances.

Syntax

JSON

```
{
  "Iops" : Integer,
```

```
  "SizeInGB" : Integer,
  "VolumeType" : String
}
```

YAML

```
Iops: Integer
SizeInGB: Integer
VolumeType: String
```

Properties

`Iops`

The number of I/O operations per second (IOPS) that the volume supports. For more information, see Iops for the `EbsBlockDevice` action in the *Amazon EC2 API Reference*.

Required: No

Type: Integer

`SizeInGB`

The volume size, in Gibibytes (GiB). For more information about specifying the volume size, see VolumeSize for the `EbsBlockDevice` action in the *Amazon EC2 API Reference*.

Required: Yes

Type: Integer

`VolumeType`

The volume type, such as `standard` or `io1`. For more information about specifying the volume type, see VolumeType for the `EbsBlockDevice` action in the *Amazon EC2 API Reference*.

Required: Yes

Type: String

Amazon EMR Step HadoopJarStepConfig

`HadoopJarStepConfig` is a property of the AWS::EMR::Step (p. 758) resource that specifies a JAR file and runtime settings that Amazon EMR (Amazon EMR) executes.

Syntax

JSON

```
{
  "Args" : [ String, ... ],
  "Jar" : String,
  "MainClass" : String,
  "StepProperties" : [ KeyValue, ... ]
}
```

YAML

```
Args:
```

```
  - String
Jar: String
MainClass: String
StepProperties:
  - KeyValue
```

Properties

`Args`

A list of command line arguments passed to the JAR file's main function when the function is executed.

Required: No

Type: List of strings

`Jar`

A path to the JAR file that Amazon EMR runs for the job flow step.

Required: Yes

Type: String

`MainClass`

The name of the main class in the specified JAR file. If you don't specify a value, you must specify a main class in the JAR file's manifest file.

Required: No

Type: String

`StepProperties`

A list of Java properties that are set when the job flow step runs. You can use these properties to pass key-value pairs to your main function in the JAR file.

Required: No

Type: List of Amazon EMR Step HadoopJarStepConfig KeyValue (p. 1180)

Amazon EMR Step HadoopJarStepConfig KeyValue

`KeyValue` is a property of the Amazon EMR Step HadoopJarStepConfig (p. 1179) property that specifies key-value pairs, which are passed to a JAR file that Amazon EMR (Amazon EMR) executes.

Syntax

JSON

```
{
  "Key" : String,
  "Value" : String
}
```

YAML

```
Key: String
Value: String
```

Properties

`Key`

The unique identifier of a key-value pair.

Required: No

Type: String

`Value`

The value part of the identified key.

Required: No

Type: String

Amazon GameLift Alias RoutingStrategy

`RoutingStrategy` is a property of the AWS::GameLift::Alias (p. 766) resource that configures the routing strategy for an Amazon GameLift (GameLift) alias. For more information, see the RoutingStrategy data type in the *Amazon GameLift API Reference*.

Syntax

JSON

```
{
  "FleetId" : String,
  "Message" : String,
  "Type" : String
}
```

YAML

```
FleetId: String
Message: String
Type: String
```

Properties

`FleetId`

A unique identifier of a GameLift fleet to associate with the alias.

Required: Conditional. If you specify `SIMPLE` for the `Type` property, you must specify this property.

Type: String

`Message`

A text message that GameLift displays for the `Terminal` routing type.

Required: Conditional. If you specify `TERMINAL` for the `Type` property, you must specify this property.

Type: String

`Type`

The type of routing strategy. For the `SIMPLE` type, traffic is routed to an active GameLift fleet. For the `Terminal` type, GameLift returns an exception with the message that you specified in the `Message` property.

Required: Yes

Type: String

Amazon GameLift Build StorageLocation

`StorageLocation` is a property of the AWS::GameLift::Build (p. 768) resource that specifies the location of an Amazon GameLift (GameLift) build package files, such as the game server binaries. For more information, see Uploading a Build to Amazon GameLift in the *Amazon GameLift Developer Guide*.

Syntax

JSON

```
{
  "Bucket" : String,
  "Key" : String,
  "RoleArn" : String
}
```

YAML

```
Bucket: String
Key: String
RoleArn: String
```

Properties

`Bucket`

The S3 bucket where the GameLift build package files are stored.

Required: Yes

Type: String

`Key`

The prefix (folder name) where the GameLift build package files are located.

Required: Yes

Type: String

`RoleArn`

An AWS Identity and Access Management (IAM) role Amazon Resource Name (ARN) that GameLift can assume to retrieve the build package files from Amazon Simple Storage Service (Amazon S3).

Required: Yes

Type: String

Amazon GameLift Fleet EC2InboundPermission

`EC2InboundPermission` is a property of the AWS::GameLift::Fleet (p. 771) resource that specifies the traffic that is permitted to access your game servers in an Amazon GameLift (GameLift) fleet.

Syntax

JSON

```
{
  "FromPort" : Integer,
  "IpRange" : String,
  "Protocol" : String,
  "ToPort" : Integer
}
```

YAML

```
FromPort: Integer
IpRange: String
Protocol: String
ToPort: Integer
```

Properties

`FromPort`

The starting value for a range of allowed port numbers. This value must be lower than the `ToPort` value.

Required: Yes

Type: Integer

`IpRange`

The range of allowed IP addresses in CIDR notation.

Required: Yes

Type: String

`Protocol`

The network communication protocol that is used by the fleet. For valid values, see the IpPermission data type in the *Amazon GameLift API Reference*.

Required: Yes

Type: String

`ToPort`

The ending value for a range of allowed port numbers. This value must be higher than the `FromPort` value.

Required: Yes

Type: Integer

IAM Policies

`Policies` is a property of the AWS::IAM::Role (p. 787), AWS::IAM::Group (p. 777), and AWS::IAM::User (p. 793) resources. The `Policies` property describes what actions are allowed on what resources. For more information about IAM policies, see Overview of Policies in *IAM User Guide*.

Syntax

JSON

```
{
  "PolicyDocument" : JSON,
  "PolicyName" : String
}
```

YAML

```
PolicyDocument: JSON
PolicyName: String
```

Properties

`PolicyDocument`

A policy document that describes what actions are allowed on which resources.

Required: Yes

Type: JSON object

Update requires: No interruption (p. 90)

`PolicyName`

The name of the policy.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

IAM User LoginProfile

`LoginProfile` is a property of the AWS::IAM::User (p. 793) resource that creates a login profile for users so that they can access the AWS Management Console.

Syntax

JSON

```
{
  "Password" : String,
  "PasswordResetRequired" : Boolean
}
```

YAML

```
Password: String
PasswordResetRequired: Boolean
```

Properties

`Password`

The password for the user.

Required: Yes

Type: String

`PasswordResetRequired`

Specifies whether the user is required to set a new password the next time the user logs in to the AWS Management Console.

Required: No

Type: Boolean

AWS IoT Actions

`Actions` is a property of the `TopicRulePayload` property that describes the actions associated with an AWS IoT rule. For more information, see Rules for AWS IoT.

Syntax

JSON

```
{
  "CloudwatchAlarm": CloudwatchAlarm Action,
  "CloudwatchMetric": CloudwatchMetric Action,
  "DynamoDB": DynamoDB Action,
  "Elasticsearch": Elasticsearch Action,
  "Firehose": Firehose Action,
  "Kinesis": Kinesis Action,
  "Lambda": Lambda Action,
  "Republish": Republish Action,
  "S3": S3 Action,
  "Sns": Sns Action,
  "Sqs": Sqs Action
}
```

YAML

```
CloudwatchAlarm:
  CloudwatchAlarm Action
CloudwatchMetric:
  CloudwatchMetric Action
DynamoDB:
  DynamoDB Action
Elasticsearch:
  Elasticsearch Action
Firehose:
```

```
  Firehose Action
Kinesis:
  Kinesis Action
Lambda:
  Lambda Action
Republish:
  Republish Action
S3:
  S3 Action
Sns:
  Sns Action
Sqs:
  Sqs Action
```

Properties

`CloudwatchAlarm`

Changes the state of a CloudWatch alarm.

Required: No

Type: `CloudWatchAlarm` (p. 1187) action object

`CloudwatchMetric`

Captures a CloudWatch metric.

Required: No

Type: `CloudWatchMetric` (p. 1188) action object

`DynamoDB`

Writes data to a DynamoDB table.

Required: No

Type: `DynamoDB` (p. 1189) action object

`Elasticsearch`

Writes data to an Elasticsearch domain.

Required: No

Type: `Elasticsearch` (p. 1191) action object

`Firehose`

Writes data to a Firehose stream.

Required: No

Type: `Firehose` (p. 1192) action object

`Kinesis`

Writes data to an Amazon Kinesis stream.

Required: No

Type: `Kinesis` (p. 1193) action object

`Lambda`

Invokes a Lambda function.

Required: No

Type: `Lambda` (p. 1194) action object

`Republish`

Publishes data to an MQ Telemetry Transport (MQTT) topic different from the one currently specified.

Required: No

Type: `Republish` (p. 1194) action object

`S3`

Writes data to an S3 bucket.

Required: No

Type: `S3` (p. 1195) action object

`Sns`

Publishes data to an SNS topic.

Required: No

Type: `Sns` (p. 1196) action object

`Sqs`

Publishes data to an SQS queue.

Required: No

Type: `Sqs` (p. 1197) action object

AWS IoT CloudwatchAlarm Action

`CloudwatchAlarm` is a property of the `Actions` property that describes an action that updates a CloudWatch alarm.

Syntax

JSON

```
{
  "AlarmName": String,
  "RoleArn": String,
  "StateReason": String,
  "StateValue": String
}
```

YAML

```
AlarmName: String
RoleArn: String
StateReason: String
StateValue: String
```

Properties

`AlarmName`

The CloudWatch alarm name.

Required: Yes

Type: String

`RoleArn`

The IAM role that allows access to the CloudWatch alarm.

Required: Yes

Type: String

`StateReason`

The reason for the change of the alarm state.

Required: Yes

Type: String

`StateValue`

The value of the alarm state.

Required: Yes

Type: String

AWS IoT CloudwatchMetric Action

`CloudwatchMetric` is a property of the `Actions` property that describes an action that captures a CloudWatch metric.

Syntax

JSON

```
{
  "MetricName": String,
  "MetricNamespace": String,
  "MetricTimestamp": String,
  "MetricUnit": String,
  "MetricValue": String,
  "RoleArn": String
}
```

YAML

```
MetricName: String
MetricNamespace: String
MetricTimestamp: String
MetricUnit: String
MetricValue: String
RoleArn: String
```

Properties

`MetricName`

The name of the CloudWatch metric.

Required: Yes

Type: String

`MetricNamespace`

The name of the CloudWatch metric namespace.

Required: Yes

Type: String

`MetricTimestamp`

An optional Unix timestamp.

Required: No

Type: String

`MetricUnit`

The metric unit supported by Amazon CloudWatch.

Required: Yes

Type: String

`MetricValue`

The value to publish to the metric. For example, if you count the occurrences of a particular term such as `Error`, the value will be `1` for each occurrence.

Required: Yes

Type: String

`RoleArn`

The ARN of the IAM role that grants access to the CloudWatch metric.

Required: Yes

Type: String

AWS IoT DynamoDB Action

`DynamoDB` is a property of the `Actions` property that describes an AWS IoT action that writes data to a DynamoDB table.

The `HashKeyField`, `RangeKeyField`, and `TableName` values must match the values you used when you initially created the table.

The `HashKeyValue` and `RangeKeyValue` fields use the `${sql-expression}` substitution template syntax. You can specify any valid expression in a `WHERE` or `SELECT` clause. This expression can include JSON properties, comparisons, calculations, and functions, for example:

- The `"HashKeyValue" : "${topic(3)}` field uses the third level of the topic.
- The `"RangeKeyValue" : "${timestamp()}` field uses the timestamp.

Syntax

JSON

```
{
  "HashKeyField": String,
  "HashKeyValue": String,
  "PayloadField": String,
  "RangeKeyField": String,
  "RangeKeyValue": String,
  "RoleArn": String,
  "TableName": String
}
```

YAML

```
HashKeyField: String
HashKeyValue: String
PayloadField: String
RangeKeyField: String
RangeKeyValue: String
RoleArn: String
TableName: String
```

Properties

`HashKeyField`

The name of the hash key.

Required: Yes

Type: String

`HashKeyValue`

The value of the hash key.

Required: Yes

Type: String

`PayloadField`

The name of the column in the DynamoDB table that contains the result of the query. You can customize this name.

Required: No

Type: String

`RangeKeyField`

The name of the range key.

Required: Yes

Type: String

`RangeKeyValue`

The value of the range key.

Required: Yes

Type: String

`RoleArn`

The ARN of the IAM role that grants access to the DynamoDB table.

Required: Yes

Type: String

`TableName`

The name of the DynamoDB table.

Required: Yes

Type: String

AWS IoT Elasticsearch Action

`Elasticsearch` is a property of the `Actions` property that describes an action that writes data to an Elasticsearch domain.

Syntax

JSON

```
{
  "Endpoint": String,
  "Id": String,
  "Index": String,
  "RoleArn": String,
  "Type": String
}
```

YAML

```
Endpoint: String
Id": String
Index": String
RoleArn": String
Type": String
```

Properties

`Endpoint`

The endpoint of your Elasticsearch domain.

Required: Yes

Type: String

`Id`

A unique identifier for the stored data.

Required: Yes

Type: String

`Index`

The Elasticsearch index where the data is stored.

Required: Yes

Type: String

`RoleArn`

The ARN of the IAM role that grants access to Elasticsearch.

Required: Yes

Type: String

`Type`

The type of stored data.

Required: Yes

Type: String

AWS IoT Firehose Action

`Firehose` is a property of the `Actions` property that describes an action that writes data to a Firehose stream.

Syntax

JSON

```
{
  "DeliveryStreamName": String,
  "RoleArn": String,
  "Separator": String
}
```

YAML

```
DeliveryStreamName: String
RoleArn: String
Separator: String
```

Properties

`DeliveryStreamName`

The delivery stream name.

Required: Yes

Type: String

`RoleArn`

The Amazon Resource Name (ARN) of the IAM role that grants access to the Firehose stream.

Required: Yes

Type: String

`Separator`

A character separator that's used to separate records written to the Firehose stream. For valid values, see Firehose Action in the *AWS IoT Developer Guide*.

Required: No

Type: String

AWS IoT Kinesis Action

`Kinesis` is a property of the `Actions` property that describes an action that writes data to an Amazon Kinesis stream.

Syntax

JSON

```
{
  "PartitionKey": String,
  "RoleArn": String,
  "StreamName": String
}
```

YAML

```
PartitionKey: String
RoleArn: String
StreamName: String
```

Properties

`PartitionKey`

The partition key (the grouping of data by shard within an an Amazon Kinesis stream).

Required: No

Type: String

`RoleArn`

The ARN of the IAM role that grants access to an Amazon Kinesis stream.

Required: Yes

Type: String

`StreamName`

The name of the Amazon Kinesis stream.

Required: Yes

Type: String

AWS IoT Lambda Action

`Lambda` is a property of the `Actions` property that describes an action that invokes a Lambda function.

Syntax

JSON

```
{
  "FunctionArn": String
}
```

YAML

```
FunctionArn: String
```

Properties

`FunctionArn`

The ARN of the Lambda function.

Required: Yes

Type: String

AWS IoT Republish Action

`Republish` is a property of the `Actions` property that describes an action that publishes data to an MQ Telemetry Transport (MQTT) topic different from the one currently specified.

Syntax

JSON

```
{
  "RoleArn": String,
  "Topic": String
}
```

YAML

```
RoleArn: String
Topic: String
```

Properties

`RoleArn`

The ARN of the IAM role that grants publishing access.

Required: Yes

Type: String

`Topic`

The name of the MQTT topic topic different from the one currently specified.

Required: Yes

Type: String

AWS IoT S3 Action

`S3` is a property of the `Actions` property that describes an action that writes data to an S3 bucket.

Syntax

JSON

```
{
  "BucketName": String,
  "Key": String,
  "RoleArn": String
}
```

YAML

```
BucketName: String
Key: String
RoleArn: String
```

Properties

`BucketName`

The name of the S3 bucket.

Required: Yes

Type: String

`Key`

The object key (the name of an object in the S3 bucket).

Required: Yes

Type: String

`RoleArn`

The ARN of the IAM role that grants access to Amazon S3.

Required: Yes

Type: String

AWS IoT Sns Action

`Sns` is a property of the `Actions` property that describes an action that publishes data to an SNS topic.

Syntax

JSON

```
{
  "MessageFormat": String,
  "RoleArn": String,
  "TargetArn": String
}
```

YAML

```
MessageFormat: String
RoleArn: String
TargetArn: String
```

Properties

`MessageFormat`

The format of the published message. Amazon SNS uses this setting to determine whether it should parse the payload and extract the platform-specific bits from the payload.

For more information, see Appendix: Message and JSON Formats in the *Amazon Simple Notification Service Developer Guide*.

Required: No

Type: String

`RoleArn`

The ARN of the IAM role that grants access to Amazon SNS.

Required: Yes

Type: String

`TargetArn`

The ARN of the Amazon SNS topic.

Required: Yes

Type: String

AWS IoT Sqs Action

`Sqs` is a property of the `Actions` property that describes an action that publishes data to an SQS queue.

Syntax

JSON

```
{
  "QueueUrl": String,
  "RoleArn": String,
  "UseBase64": Boolean
}
```

YAML

```
QueueUrl: String
RoleArn: String
UseBase64: Boolean
```

Properties

`QueueUrl`

The URL of the Amazon Simple Queue Service (Amazon SQS) queue.

Required: Yes

Type: String

`RoleArn`

The ARN of the IAM role that grants access to Amazon SQS.

Required: Yes

Type: String

`UseBase64`

Specifies whether Base64 encoding should be used.

Required: No

Type: Boolean

AWS IoT TopicRulePayload

`TopicRulePayload` is a property of the `AWS::IoT::TopicRule` resource that describes the payload of an AWS IoT rule.

Syntax

JSON

```
{
```

```
  "Actions": [ Action, ... ],
  "AwsIotSqlVersion": String,
  "Description": String,
  "RuleDisabled": Boolean,
  "Sql": String
}
```

YAML

```
Actions:
  - Action
AwsIotSqlVersion: String
Description: String
RuleDisabled: Boolean
Sql: String
```

Properties

`Actions`

The actions associated with the rule.

Required: Yes

Type: Array of `Action` (p. 1185) objects

Update requires: No interruption (p. 90)

`AwsIotSqlVersion`

The version of the SQL rules engine to use when evaluating the rule.

Required: No

Type: String

Update requires: No interruption (p. 90)

`Description`

The description of the rule.

Required: No

Type: String

Update requires: No interruption (p. 90)

`RuleDisabled`

Specifies whether the rule is disabled.

Required: Yes

Type: Boolean

Update requires: No interruption (p. 90)

`Sql`

The SQL statement that queries the topic. For more information, see Rules for AWS IoT in the *AWS IoT Developer Guide*.

Required: Yes

Type: String

Update requires: No interruption (p. 90)

Amazon Kinesis Firehose DeliveryStream Destination CloudWatchLoggingOptions

`CloudWatchLoggingOptions` is a property of the Amazon Kinesis Firehose DeliveryStream ElasticsearchDestinationConfiguration (p. 1200), Amazon Kinesis Firehose DeliveryStream RedshiftDestinationConfiguration (p. 1203), and Amazon Kinesis Firehose DeliveryStream S3DestinationConfiguration (p. 1206) properties that specifies Amazon CloudWatch Logs (CloudWatch Logs) logging options that Amazon Kinesis Firehose (Firehose) uses for the delivery stream.

Syntax

JSON

```
{
  "Enabled" : Boolean,
  "LogGroupName" : String,
  "LogStreamName" : String
}
```

YAML

```
Enabled: Boolean
LogGroupName: String
LogStreamName: String
```

Properties

`Enabled`

Indicates whether CloudWatch Logs logging is enabled.

Required: No

Type: Boolean

`LogGroupName`

The name of the CloudWatch Logs log group that contains the log stream that Firehose will use.

Required: Conditional. If you enable logging, you must specify this property.

Type: String

`LogStreamName`

The name of the CloudWatch Logs log stream that Firehose uses to send logs about data delivery.

Required: Conditional. If you enable logging, you must specify this property.

Type: String

Amazon Kinesis Firehose DeliveryStream ElasticsearchDestinationConfiguration

`ElasticsearchDestinationConfiguration` is a property of the AWS::KinesisFirehose::DeliveryStream (p. 811) resource that specifies an Amazon Elasticsearch Service (Amazon ES) domain that Amazon Kinesis Firehose (Firehose) delivers data to.

Syntax

JSON

```
{
  "BufferingHints" : BufferingHints (p. 1202),
  "CloudWatchLoggingOptions" : CloudWatchLoggingOptions (p. 1199),
  "DomainARN" : String,
  "IndexName" : String,
  "IndexRotationPeriod" : String,
  "RetryOptions" : RetryOptions (p. 1203),
  "RoleARN" : String,
  "S3BackupMode" : String,
  "S3Configuration" : S3Configuration (p. 1206),
  "TypeName" : String
}
```

YAML

```
BufferingHints:
  BufferingHints (p. 1202)
CloudWatchLoggingOptions:
  CloudWatchLoggingOptions (p. 1199)
DomainARN: String
IndexName: String
IndexRotationPeriod: String
RetryOptions:
  RetryOptions (p. 1203)
RoleARN: String
S3BackupMode: String
S3Configuration:
  S3Configuration (p. 1206)
TypeName: String
```

Properties

`BufferingHints`

Configures how Firehose buffers incoming data while delivering it to the Amazon ES domain.

Required: Yes

Type: Amazon Kinesis Firehose DeliveryStream ElasticsearchDestinationConfiguration BufferingHints (p. 1202)

`CloudWatchLoggingOptions`

The Amazon CloudWatch Logs logging options for the delivery stream.

Required: No

Type: Amazon Kinesis Firehose DeliveryStream Destination CloudWatchLoggingOptions (p. 1199)

`DomainARN`

The Amazon Resource Name (ARN) of the Amazon ES domain that Firehose delivers data to.

Required: Yes

Type: String

`IndexName`

The name of the Elasticsearch index to which Firehose adds data for indexing.

Required: Yes

Type: String

`IndexRotationPeriod`

The frequency of Elasticsearch index rotation. If you enable index rotation, Firehose appends a portion of the UTC arrival timestamp to the specified index name, and rotates the appended timestamp accordingly. For more information, see Index Rotation for the Amazon ES Destination in the *Amazon Kinesis Firehose Developer Guide*.

Required: Yes

Type: String

`RetryOptions`

The retry behavior when Firehose is unable to deliver data to Amazon ES.

Type: Amazon Kinesis Firehose DeliveryStream ElasticsearchDestinationConfiguration RetryOptions (p. 1203)

Type: String

`RoleARN`

The ARN of the AWS Identity and Access Management (IAM) role that grants Firehose access to your S3 bucket, AWS KMS (if you enable data encryption), and Amazon CloudWatch Logs (if you enable logging).

For more information, see Grant Firehose Access to an Amazon Elasticsearch Service Destination in the *Amazon Kinesis Firehose Developer Guide*.

Required: Yes

Type: String

`S3BackupMode`

The condition under which Firehose delivers data to Amazon Simple Storage Service (Amazon S3). You can send Amazon S3 all documents (all data) or only the documents that Firehose could not deliver to the Amazon ES destination. For more information and valid values, see the `S3BackupMode` content for the ElasticsearchDestinationConfiguration data type in the *Amazon Kinesis Firehose API Reference*.

Required: Yes

Type: String

`S3Configuration`

The S3 bucket where Firehose backs up incoming data.

Type: Amazon Kinesis Firehose DeliveryStream S3DestinationConfiguration (p. 1206)

Type: String

`TypeName`

The Elasticsearch type name that Amazon ES adds to documents when indexing data.

Required: Yes

Type: String

Amazon Kinesis Firehose DeliveryStream ElasticsearchDestinationConfiguration BufferingHints

`BufferingHints` is a property of the Amazon Kinesis Firehose DeliveryStream ElasticsearchDestinationConfiguration (p. 1200) property that specifies how Amazon Kinesis Firehose (Firehose) buffers incoming data while delivering it to the destination. The first buffer condition that is satisfied triggers Firehose to deliver the data.

Syntax

JSON

```
{
  "IntervalInSeconds" : Integer,
  "SizeInMBs" : Integer
}
```

YAML

```
IntervalInSeconds: Integer
SizeInMBs: Integer
```

Properties

`IntervalInSeconds`

The length of time, in seconds, that Firehose buffers incoming data before delivering it to the destination. For valid values, see the `IntervalInSeconds` content for the BufferingHints data type in the *Amazon Kinesis Firehose API Reference*.

Required: Yes

Type: Integer

`SizeInMBs`

The size of the buffer, in MBs, that Firehose uses for incoming data before delivering it to the destination. For valid values, see the `SizeInMBs` content for the BufferingHints data type in the *Amazon Kinesis Firehose API Reference*.

Required: Yes

Type: Integer

Amazon Kinesis Firehose DeliveryStream ElasticsearchDestinationConfiguration RetryOptions

`RetryOptions` is a property of the Amazon Kinesis Firehose DeliveryStream ElasticsearchDestinationConfiguration (p. 1200) property that configures the retry behavior when Amazon Kinesis Firehose (Firehose) can't deliver data to Amazon Elasticsearch Service (Amazon ES).

Syntax

JSON

```
{
  "DurationInSeconds" : Integer
}
```

YAML

```
DurationInSeconds: Integer
```

Properties

`DurationInSeconds`

After an initial failure to deliver to Amazon ES, the total amount of time during which Firehose re-attempts delivery (including the first attempt). If Firehose can't deliver the data within the specified time, it writes the data to the backup S3 bucket. For valid values, see the `DurationInSeconds` content for the ElasticsearchRetryOptions data type in the *Amazon Kinesis Firehose API Reference*.

Required: Yes

Type: Integer

Amazon Kinesis Firehose DeliveryStream RedshiftDestinationConfiguration

`RedshiftDestinationConfiguration` is a property of the AWS::KinesisFirehose::DeliveryStream (p. 811) resource that specifies an Amazon Redshift cluster to which Amazon Kinesis Firehose (Firehose) delivers data.

Syntax

JSON

```
{
  "CloudWatchLoggingOptions" : CloudWatchLoggingOptions (p. 1199),
  "ClusterJDBCURL" : String,
  "CopyCommand" : CopyCommand (p. 1205),
  "Password" : String,
  "RoleARN" : String,
  "S3Configuration" : S3Configuration (p. 1206),
  "Username" : String
}
```

YAML

```
CloudWatchLoggingOptions:
  CloudWatchLoggingOptions (p. 1199)
ClusterJDBCURL: String
CopyCommand:
  CopyCommand (p. 1205)
Password: String
RoleARN: String
S3Configuration:
  S3Configuration (p. 1206)
Username: String
```

Properties

`CloudWatchLoggingOptions`

The Amazon CloudWatch Logs logging options for the delivery stream.

Required: No

Type: Amazon Kinesis Firehose DeliveryStream Destination CloudWatchLoggingOptions (p. 1199)

`ClusterJDBCURL`

The connection string that Firehose uses to connect to the Amazon Redshift cluster.

Required: Yes

Type: String

`CopyCommand`

Configures the Amazon Redshift `COPY` command that Firehose uses to load data into the cluster from the S3 bucket.

Required: Yes

Type: Amazon Kinesis Firehose DeliveryStream RedshiftDestinationConfiguration CopyCommand (p. 1205)

`Password`

The password for the Amazon Redshift user that you specified in the `Username` property.

Required: Yes

Type: String

`RoleARN`

The ARN of the AWS Identity and Access Management (IAM) role that grants Firehose access to your S3 bucket and AWS KMS (if you enable data encryption).

For more information, see Grant Firehose Access to an Amazon Redshift Destination in the *Amazon Kinesis Firehose Developer Guide*.

Required: Yes

Type: String

`S3Configuration`

The S3 bucket where Firehose first delivers data. After the data is in the bucket, Firehose uses the `COPY` command to load the data into the Amazon Redshift cluster. For the S3 bucket's compression format, don't specify `SNAPPY` or `ZIP` because the Amazon Redshift `COPY` command doesn't support them.

Required: Yes

Type: Amazon Kinesis Firehose DeliveryStream S3DestinationConfiguration (p. 1206)

`Username`

The Amazon Redshift user that has permission to access the Amazon Redshift cluster. This user must have `INSERT` privileges for copying data from the S3 bucket to the cluster.

Required: Yes

Type: String

Amazon Kinesis Firehose DeliveryStream RedshiftDestinationConfiguration CopyCommand

`CopyCommand` is a property of the Amazon Kinesis Firehose DeliveryStream RedshiftDestinationConfiguration (p. 1203) property that configures the Amazon Redshift`COPY` command that Amazon Kinesis Firehose (Firehose) uses to load data into an Amazon Redshift cluster from an S3 bucket.

Syntax

JSON

```
{
  "CopyOptions" : String,
  "DataTableColumns" : String,
  "DataTableName" : String
}
```

YAML

```
CopyOptions: String
DataTableColumns: String
DataTableName: String
```

Properties

`CopyOptions`

Parameters to use with the Amazon Redshift `COPY` command. For examples, see the `CopyOptions` content for the CopyCommand data type in the *Amazon Kinesis Firehose API Reference*.

Required: No

Type: String

`DataTableColumns`

A comma-separated list of the column names in the table that Firehose copies data to.

Required: No

Type: String

`DataTableName`

The name of the table where Firehose adds the copied data.

Required: Yes

Type: String

Amazon Kinesis Firehose DeliveryStream S3DestinationConfiguration

`S3DestinationConfiguration` is a property of the AWS::KinesisFirehose::DeliveryStream (p. 811) resource and the Amazon Kinesis Firehose DeliveryStream ElasticsearchDestinationConfiguration (p. 1200) and Amazon Kinesis Firehose DeliveryStream RedshiftDestinationConfiguration (p. 1203) properties that specifies an Amazon Simple Storage Service (Amazon S3) destination to which Amazon Kinesis Firehose (Firehose) delivers data.

Syntax

JSON

```
{
  "BucketARN" : String,
  "BufferingHints" : BufferingHints (p. 1207),
  "CloudWatchLoggingOptions" : CloudWatchLoggingOptions (p. 1199),
  "CompressionFormat" : String,
  "EncryptionConfiguration" : EncryptionConfiguration (p. 1209),
  "Prefix" : String,
  "RoleARN" : String
}
```

YAML

```
BucketARN: String
BufferingHints:
  BufferingHints (p. 1207)
CloudWatchLoggingOptions:
  CloudWatchLoggingOptions (p. 1199)
CompressionFormat: String
EncryptionConfiguration:
  EncryptionConfiguration (p. 1209)
Prefix: String
RoleARN: String
```

Properties

`BucketARN`

The Amazon Resource Name (ARN) of the S3 bucket to send data to.

Required: Yes

Type: String

`BufferingHints`

Configures how Firehose buffers incoming data while delivering it to the S3 bucket.

Required: Yes

Type: Amazon Kinesis Firehose DeliveryStream S3DestinationConfiguration BufferingHints (p. 1207)

`CloudWatchLoggingOptions`

The Amazon CloudWatch Logs logging options for the delivery stream.

Required: No

Type: Amazon Kinesis Firehose DeliveryStream Destination CloudWatchLoggingOptions (p. 1199)

`CompressionFormat`

The type of compression that Firehose uses to compress the data that it delivers to the S3 bucket. For valid values, see the `CompressionFormat` content for the S3DestinationConfiguration data type in the *Amazon Kinesis Firehose API Reference*.

Required: Yes

Type: String

`EncryptionConfiguration`

Configures Amazon Simple Storage Service (Amazon S3) server-side encryption. Firehose uses AWS Key Management Service (AWS KMS) to encrypt the data that it delivers to your S3 bucket.

Required: No

Type: Amazon Kinesis Firehose DeliveryStream S3DestinationConfiguration EncryptionConfiguration (p. 1209)

`Prefix`

A prefix that Firehose adds to the files that it delivers to the S3 bucket. The prefix helps you identify the files that Firehose delivered.

Required: Yes

Type: String

`RoleARN`

The ARN of an AWS Identity and Access Management (IAM) role that grants Firehose access to your S3 bucket and AWS KMS (if you enable data encryption).

For more information, see Grant Firehose Access to an Amazon S3 Destination in the *Amazon Kinesis Firehose Developer Guide*.

Required: Yes

Type: String

Amazon Kinesis Firehose DeliveryStream S3DestinationConfiguration BufferingHints

`BufferingHints` is a property of the Amazon Kinesis Firehose DeliveryStream S3DestinationConfiguration (p. 1206) property that specifies how Amazon Kinesis Firehose (Firehose) buffers incoming data before delivering it to the destination. The first buffer condition that is satisfied triggers Firehose to deliver the data..

Syntax

JSON

```
{
  "IntervalInSeconds" : Integer,
  "SizeInMBs" : Integer
}
```

YAML

```
IntervalInSeconds: Integer
SizeInMBs: Integer
```

Properties

`IntervalInSeconds`

The length of time, in seconds, that Firehose buffers incoming data before delivering it to the destination. For valid values, see the `IntervalInSeconds` content for the BufferingHints data type in the *Amazon Kinesis Firehose API Reference*.

Required: Yes

Type: Integer

`SizeInMBs`

The size of the buffer, in MBs, that Firehose uses for incoming data before delivering it to the destination. For valid values, see the `SizeInMBs` content for the BufferingHints data type in the *Amazon Kinesis Firehose API Reference*.

Required: Yes

Type: Integer

Amazon Kinesis Firehose DeliveryStream S3DestinationConfiguration EncryptionConfiguration KMSEncryptionConfig

`KMSEncryptionConfig` is a property of the Amazon Kinesis Firehose DeliveryStream S3DestinationConfiguration EncryptionConfiguration KMSEncryptionConfig (p. 1208) property that specifies the AWS Key Management Service (AWS KMS) encryption key that Amazon Simple Storage Service (Amazon S3) uses to encrypt data delivered by the Amazon Kinesis Firehose (Firehose) stream.

Syntax

JSON

```
{
  "AWSKMSKeyARN" : String
}
```

YAML

```
AWSKMSKeyARN: String
```

Properties

`AWSKMSKeyARN`

The Amazon Resource Name (ARN) of the AWS KMS encryption key that Amazon S3 uses to encrypt data delivered by the Firehose stream. The key must belong to the same region as the destination S3 bucket.

Required: Yes

Type: String

Amazon Kinesis Firehose DeliveryStream S3DestinationConfiguration EncryptionConfiguration

`EncryptionConfiguration` is a property of the Amazon Kinesis Firehose DeliveryStream S3DestinationConfiguration (p. 1206) property that specifies the encryption settings that Amazon Kinesis Firehose (Firehose) uses when delivering data to Amazon Simple Storage Service (Amazon S3).

Syntax

JSON

```
{
  "KMSEncryptionConfig" : KMSEncryptionConfig (p. 1208),
  "NoEncryptionConfig" : String
}
```

YAML

```
KMSEncryptionConfig:
  KMSEncryptionConfig (p. 1208)
NoEncryptionConfig: String
```

Properties

`KMSEncryptionConfig`

The AWS Key Management Service (AWS KMS) encryption key that Amazon S3 uses to encrypt your data.

Required: No

Type: Amazon Kinesis Firehose DeliveryStream S3DestinationConfiguration EncryptionConfiguration KMSEncryptionConfig (p. 1208)

`NoEncryptionConfig`

Disables encryption. For valid values, see the `NoEncryptionConfig` content for the EncryptionConfiguration data type in the *Amazon Kinesis Firehose API Reference*.

Required: No

Type: String

AWS Lambda Function DeadLetterConfig

`DeadLetterConfig` is a property of the AWS::Lambda::Function (p. 824) resource that specifies a Dead Letter Queue (DLQ) that AWS Lambda (Lambda) sends events to when it can't process them. For example, you can send unprocessed events to an Amazon Simple Notification Service (Amazon SNS) topic, where you can take further action.

Syntax

JSON

```
{
  "TargetArn" : String
}
```

YAML

```
TargetArn: String
```

Properties

`TargetArn`

The Amazon Resource Name (ARN) of a resource where Lambda delivers unprocessed events, such as an Amazon SNS topic or Amazon Simple Queue Service (Amazon SQS) queue. For the Lambda function execution role, you must explicitly provide the relevant permissions so that access to your DLQ resource is part of the execution role for your Lambda function.

Required: No

Type: String

AWS Lambda Function Environment

`Environment` is a property of the AWS::Lambda::Function (p. 824) resource that specifies key-value pairs that the AWS Lambda (Lambda) function can access so that you can apply configuration changes, such as test and production environment configurations, without changing the function code.

Syntax

JSON

```
{
  "Variables" : { String:String, ... }
}
```

YAML

```
Variables:
```

```
  String: String
```

Properties

`Variables`

A map of key-value pairs that the Lambda function can access.

Required: No

Type: Mapping of key-value pairs

AWS Lambda Function Code

`Code` is a property of the AWS::Lambda::Function (p. 824) resource that enables you to specify the source code of an AWS Lambda function. Your source code can be located in either the template or a file in an S3 bucket. For `nodejs4.3` and `python2.7` runtime environments only, you can provide source code as inline text in your template.

> **Note**
> To update a Lambda function whose source code is in an S3 bucket, you must trigger an update by updating the `S3Bucket`, `S3Key`, or `S3ObjectVersion` property. Updating the source code alone doesn't update the function.

Syntax

JSON

```
{
  "S3Bucket" : String,
  "S3Key" : String,
  "S3ObjectVersion" : String,
  "ZipFile" : String
}
```

YAML

```
S3Bucket: String
S3Key: String
S3ObjectVersion: String
ZipFile: String
```

Properties

`S3Bucket`

The name of the S3 bucket that contains the source code of your Lambda function. The S3 bucket must be in the same region as the stack.

> **Note**
> The `cfn-response` module isn't available for source code stored in S3 buckets. To send responses, write your own functions.

Required: Conditional Specify both the `S3Bucket` and `S3Key` properties or specify the `ZipFile` property.

Type: String

`S3Key`

The location and name of the `.zip` file that contains your source code. If you specify this property, you must also specify the `S3Bucket` property.

Required: Conditional You must specify both the `S3Bucket` and `S3Key` properties or specify the `ZipFile` property.

Type: String

`S3ObjectVersion`

If you have S3 versioning enabled, the version ID of the`.zip` file that contains your source code. You can specify this property only if you specify the `S3Bucket` and `S3Key` properties.

Required: No

Type: String

`ZipFile`

For `nodejs4.3` and `python2.7` runtime environments, the source code of your Lambda function. You can't use this property with other runtime environments.

You can specify up to 4096 characters. You must precede certain special characters in your source code, such as quotation marks ("), newlines (`\n`), and tabs (`\t`), with a backslash (`\`). For a list of special characters, see http://json.org/.

If you specify a function that interacts with an AWS CloudFormation custom resource, you don't have to write your own functions to send responses to the custom resource that invoked the function. AWS CloudFormation provides a response module that simplifies sending responses. For more information, see `cfn-response` Module (p. 1212).

Required: Conditional You must specify both the `S3Bucket` and `S3Key` properties or specify the `ZipFile` property.

Type: String

`cfn-response` Module

When you use the `ZipFile` property to specify your function's source code and that function interacts with an AWS CloudFormation custom resource, you can load the `cfn-response` module to send responses to those resources. The module contains a `send` method, which sends a response object (p. 375) to a custom resource by way of an Amazon S3 pre-signed URL (the `ResponseURL`).

After executing the `send` method, the Lambda function terminates, so anything you write after that method is ignored.

Note
The `cfn-response` module is available only when you use the `ZipFile` property to write your source code. It isn't available for source code stored in S3 buckets. For code in S3 buckets, you must write your own functions to send responses.

Loading the `cfn-response` Module

For the `nodejs4.3` runtime environment, use the `require()` function to load the `cfn-response` module. For example, the following code example creates a `cfn-response` object with the name `response`:

```
var response = require('cfn-response');
```

For `python2.7` environments, use the `import` statement to load the `cfnresponse` module, as shown in the following example:

Note

Use this exact import statement. If you use other variants of the import statement, AWS CloudFormation won't include the response module.

```
import cfnresponse
```

`send` Method Parameters

You can use the following parameters with the `send` method.

`event`

The fields in a custom resource request (p. 376).

`context`

An object, specific to Lambda functions, that you can use to specify when the function and any callbacks have completed execution or to access information from within the Lambda execution environment. For more information, see Programming Model (Node.js) in the *AWS Lambda Developer Guide*.

`responseStatus`

Whether the function successfully completed. Use the `cfnresponse` module constants to specify the status: `SUCCESS` for successful executions and `FAILED` for failed executions.

`responseData`

The `Data` field of a custom resource response object (p. 375). The data is a list of name-value pairs.

`physicalResourceId`

Optional. The unique identifier of the custom resource that invoked the function. By default, the module uses the name of the Amazon CloudWatch Logs log stream that is associated with the Lambda function.

Examples

Node.js

In the following Node.js example, the inline Lambda function takes an input value and multiplies it by 5. Inline functions are especially useful for smaller functions because they allow you to specify the source code directly in the template instead of creating a package and uploading it to an Amazon S3 bucket. The function uses the `cfn-response send` method to send the result back to the custom resource that invoked it.

JSON

```
"ZipFile": { "Fn::Join": ["", [
  "var response = require('cfn-response');",
  "exports.handler = function(event, context) {",
  "  var input = parseInt(event.ResourceProperties.Input);",
  "  var responseData = {Value: input * 5};",
  "  response.send(event, context, response.SUCCESS, responseData);",
  "};"
]]}
```

YAML

```
ZipFile: >
```

```
  var response = require('cfn-response');
  exports.handler = function(event, context) {
    var input = parseInt(event.ResourceProperties.Input);
    var responseData = {Value: input * 5};
    response.send(event, context, response.SUCCESS, responseData);
  };
```

Python

As in the preceding example, in the following Python 2.7 example, the inline Lambda function takes an integer value and multiplies it by 5.

JSON

```
"ZipFile" : { "Fn::Join" : ["\n", [
  "import json",
  "import cfnresponse",
  "def handler(event, context):",
  "   responseValue = int(event['ResourceProperties']['Input']) * 5",
  "   responseData = {}",
  "   responseData['Data'] = responseValue",
  "   cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
 \"CustomResourcePhysicalID\")"
]]}
```

YAML

```
ZipFile: |
  import json
  import cfnresponse
  def handler(event, context):
    responseValue = int(event['ResourceProperties']['Input']) * 5
    responseData = {}
    responseData['Data'] = responseValue
    cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
 "CustomResourcePhysicalID")
```

Module Source Code

The response module source code for the `nodejs4.3` runtime environment follows. Review it to understand what the module does and for help with implementing your own response functions.

```
/* Copyright 2015 Amazon Web Services, Inc. or its affiliates. All Rights Reserved.
   This file is licensed to you under the AWS Customer Agreement (the "License").
   You may not use this file except in compliance with the License.
   A copy of the License is located at http://aws.amazon.com/agreement/.
   This file is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
 KIND, express or implied.
   See the License for the specific language governing permissions and limitations under
 the License. */

exports.SUCCESS = "SUCCESS";
exports.FAILED = "FAILED";

exports.send = function(event, context, responseStatus, responseData, physicalResourceId) {

    var responseBody = JSON.stringify({
        Status: responseStatus,
        Reason: "See the details in CloudWatch Log Stream: " + context.logStreamName,
        PhysicalResourceId: physicalResourceId || context.logStreamName,
        StackId: event.StackId,
```

```
        RequestId: event.RequestId,
        LogicalResourceId: event.LogicalResourceId,
        Data: responseData
    });

    console.log("Response body:\n", responseBody);

    var https = require("https");
    var url = require("url");

    var parsedUrl = url.parse(event.ResponseURL);
    var options = {
        hostname: parsedUrl.hostname,
        port: 443,
        path: parsedUrl.path,
        method: "PUT",
        headers: {
            "content-type": "",
            "content-length": responseBody.length
        }
    };

    var request = https.request(options, function(response) {
        console.log("Status code: " + response.statusCode);
        console.log("Status message: " + response.statusMessage);
        context.done();
    });

    request.on("error", function(error) {
        console.log("send(..) failed executing https.request(..): " + error);
        context.done();
    });

    request.write(responseBody);
    request.end();
}
```

The response module source code for the `python2.7` environment follows:

```
#  Copyright 2016 Amazon Web Services, Inc. or its affiliates. All Rights Reserved.
#  This file is licensed to you under the AWS Customer Agreement (the "License").
#  You may not use this file except in compliance with the License.
#  A copy of the License is located at http://aws.amazon.com/agreement/ .
#  This file is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
 KIND, express or implied.
#  See the License for the specific language governing permissions and limitations under
 the License.

from botocore.vendored import requests
import json

SUCCESS = "SUCCESS"
FAILED = "FAILED"

def send(event, context, responseStatus, responseData, physicalResourceId):
    responseUrl = event['ResponseURL']

    print responseUrl

    responseBody = {}
    responseBody['Status'] = responseStatus
    responseBody['Reason'] = 'See the details in CloudWatch Log Stream: ' +
 context.log_stream_name
    responseBody['PhysicalResourceId'] = physicalResourceId or context.log_stream_name
    responseBody['StackId'] = event['StackId']
```

```
    responseBody['RequestId'] = event['RequestId']
    responseBody['LogicalResourceId'] = event['LogicalResourceId']
    responseBody['Data'] = responseData

    json_responseBody = json.dumps(responseBody)

    print "Response body:\n" + json_responseBody

    headers = {
        'content-type' : '',
        'content-length' : str(len(json_responseBody))
    }

    try:
        response = requests.put(responseUrl,
                                data=json_responseBody,
                                headers=headers)
        print "Status code: " + response.reason
    except Exception as e:
        print "send(..) failed executing requests.put(..): " + str(e)
```

AWS Lambda Function VPCConfig

`VpcConfig` is a property of the AWS::Lambda::Function (p. 824) resource that enables to your AWS Lambda (Lambda) function to access resources in a VPC. For more information, see Configuring a Lambda Function to Access Resources in an Amazon VPC in the *AWS Lambda Developer Guide*.

Syntax

JSON

```
{
  "SecurityGroupIds" : [ String, ... ],
  "SubnetIds" : [ String, ... ]
}
```

YAML

```
SecurityGroupIds:
  - String
SubnetIds:
  - String
```

Properties

`SecurityGroupIds`

A list of one or more security groups IDs in the VPC that includes the resources to which your Lambda function requires access.

Required: Yes

Type: List of strings

`SubnetIds`

A list of one or more subnet IDs in the VPC that includes the resources to which your Lambda function requires access.

Required: Yes

Type: List of strings

Name Type

For some resources, you can specify a custom name. By default, AWS CloudFormation generates a unique physical ID to name a resource. For example, AWS CloudFormation might name an Amazon S3 bucket with the following physical ID `stack123123123123-s3bucket-abcdefghijk1`. With custom names, you can specify a name that's easier to read and identify, such as `production-app-logs` or `business-metrics`.

Resource names must be unique across all of your active stacks. If you reuse templates to create multiple stacks, you must change or remove custom names from your template. If you don't specify a name, AWS CloudFormation generates a unique physical ID to name the resource. Names must begin with a letter; contain only ASCII letters, digits, and hyphens; and not end with a hyphen or contain two consecutive hyphens.

Also, do not manage stack resources outside of AWS CloudFormation. For example, if you rename a resource that's part of a stack without using AWS CloudFormation, you might get an error any time you try to update or delete that stack.

Important
You can't perform an update that causes a custom-named resource to be replaced. If you must replace the resource, specify a new name.

Example

If you want to use a custom name, specify a name property for that resource in your AWS CloudFormation template. Each resource that supports custom names has its own property that you specify. For example, to name an DynamoDB table, you use the `TableName` property, as shown in the following sample:

JSON

```
"myDynamoDBTable" : {
   "Type" : "AWS::DynamoDB::Table",
   "Properties" : {
      "KeySchema" : {
         "HashKeyElement": {
            "AttributeName" : "AttributeName1",
            "AttributeType" : "S"
         },
         "RangeKeyElement" : {
            "AttributeName" : "AttributeName2",
            "AttributeType" : "N"
         }
      },
      "ProvisionedThroughput" : {
         "ReadCapacityUnits" : "5",
         "WriteCapacityUnits" : "10"
      },
      "TableName" : "SampleTable"
   }
}
```

YAML

```
myDynamoDBTable:
```

```
Type: "AWS::DynamoDB::Table"
Properties:
  KeySchema:
    HashKeyElement:
      AttributeName: "AttributeName1"
      AttributeType: "S"
    RangeKeyElement:
      AttributeName: "AttributeName2"
      AttributeType: "N"
  ProvisionedThroughput:
    ReadCapacityUnits: "5"
    WriteCapacityUnits: "10"
  TableName: "SampleTable"
```

Supported Resources

The following resource types support custom names:

- AWS::ApiGateway::ApiKey (p. 397)
- AWS::ApiGateway::Model (p. 412)
- AWS::CloudWatch::Alarm (p. 498)
- AWS::DynamoDB::Table (p. 550)
- AWS::ElasticBeanstalk::Application (p. 708)
- AWS::ElasticBeanstalk::Environment (p. 714)
- AWS::CodeDeploy::Application (p. 509)
- AWS::CodeDeploy::DeploymentConfig (p. 510)
- AWS::CodeDeploy::DeploymentGroup (p. 512)
- AWS::Config::ConfigRule (p. 525)
- AWS::Config::DeliveryChannel (p. 533)
- AWS::Config::ConfigurationRecorder (p. 531)
- AWS::ElasticLoadBalancing::LoadBalancer (p. 719)
- AWS::ElastiCache::CacheCluster (p. 684)
- AWS::ECR::Repository (p. 667)
- AWS::ECS::Cluster (p. 669)
- AWS::Elasticsearch::Domain (p. 745)
- AWS::Events::Rule (p. 761)
- AWS::IAM::Group (p. 777)
- AWS::IAM::Role (p. 787)
- AWS::IAM::User (p. 793)
- AWS::Lambda::Function (p. 824)
- AWS::RDS::DBInstance (p. 881)
- AWS::S3::Bucket (p. 937)
- AWS::SNS::Topic (p. 954)
- AWS::SQS::Queue (p. 958)

DataSource

`DataSource` is a property of the AWS::OpsWorks::App (p. 843) resource that specifies a database to associate with an AWS OpsWorks app.

Syntax

JSON

```
{
  "Arn" : String,
  "DatabaseName" : String,
  "Type" : String
}
```

YAML

```
Arn: String
DatabaseName: String
Type: String
```

Properties

`Arn`

The ARN of the data source.

Required: No

Type: String

`DatabaseName`

The name of the database.

Required: No

Type: String

`Type`

The type of the data source, such as `AutoSelectOpsworksMysqlInstance`, `OpsworksMysqlInstance`, or `RdsDbInstance`. For valid values, see the DataSource type in the *AWS OpsWorks Stacks API Reference*.

Required: No

Type: String

AWS OpsWorks App Environment

`Environment` is a property of the AWS::OpsWorks::App (p. 843) resource that specifies the environment variable to associate with the AWS OpsWorks app.

Syntax

JSON

```
{
  "Key" : String,
```

```
  "Secure" : Boolean,
  "Value" : String
}
```

YAML

```
Key: String
Secure: Boolean
Value: String
```

Properties

`Key`

The name of the environment variable, which can consist of up to 64 characters. You can use upper and lowercase letters, numbers, and underscores (_), but the name must start with a letter or underscore.

Required: Yes

Type: String

`Secure`

Indicates whether the value of the environment variable is concealed, such as with a DescribeApps response. To conceal an environment variable's value, set the value to `true`.

Required: No

Type: Boolean

`Value`

The value of the environment variable, which can be empty. You can specify a value of up to 256 characters.

Required: Yes

Type: String

AWS OpsWorks AutoScalingThresholds Type

Describes the scaling thresholds for the AWS OpsWorks LoadBasedAutoScaling Type (p. 1223) property. For more information, see AutoScalingThresholds in the *AWS OpsWorks Stacks API Reference*.

Syntax

JSON

```
{
  "CpuThreshold" : Number,
  "IgnoreMetricsTime" : Integer,
  "InstanceCount" : Integer,
  "LoadThreshold" : Number,
  "MemoryThreshold" : Number,
  "ThresholdsWaitTime" : Integer
}
```

YAML

```
CpuThreshold: Number
IgnoreMetricsTime: Integer
InstanceCount: Integer
LoadThreshold: Number
MemoryThreshold: Number
ThresholdsWaitTime: Integer
```

Properties

`CpuThreshold`

The percentage of CPU utilization that triggers the starting or stopping of instances (scaling).

Required: No

Type: Number

`IgnoreMetricsTime`

The amount of time (in minutes) after a scaling event occurs that AWS OpsWorks should ignore metrics and not start any additional scaling events.

Required: No

Type: Integer

`InstanceCount`

The number of instances to add or remove when the load exceeds a threshold.

Required: No

Type: Integer

`LoadThreshold`

The degree of system load that triggers the starting or stopping of instances (scaling). For more information about how load is computed, see Load (computing).

Required: No

Type: Number

`MemoryThreshold`

The percentage of memory consumption that triggers the starting or stopping of instances (scaling).

Required: No

Type: Number

`ThresholdsWaitTime`

The amount of time, in minutes, that the load must exceed a threshold before instances are added or removed.

Required: No

Type: Integer

AWS OpsWorks ChefConfiguration Type

Describes the Chef configuration for the AWS::OpsWorks::Stack (p. 862) resource type. For more information, see ChefConfiguration in the *AWS OpsWorks Stacks API Reference*.

Syntax

JSON

```
{
  "BerkshelfVersion" : String,
  "ManageBerkshelf" : Boolean
}
```

YAML

```
BerkshelfVersion: String
ManageBerkshelf: Boolean
```

Properties

`BerkshelfVersion`

The Berkshelf version.

Required: No

Type: String

`ManageBerkshelf`

Whether to enable Berkshelf.

Required: No

Type: Boolean

AWS OpsWorks Layer LifeCycleConfiguration

`LifeCycleConfiguration` is property of the AWS::OpsWorks::Layer (p. 856) resource that specifies the lifecycle event configuration for the layer.

Syntax

JSON

```
{
  "ShutdownEventConfiguration" : ShutdownEventConfiguration
}
```

YAML

```
ShutdownEventConfiguration:
  ShutdownEventConfiguration
```

Properties

`ShutdownEventConfiguration`

Specifies the shutdown event configuration for a layer.

Required: No

Type: AWS OpsWorks Layer LifeCycleConfiguration ShutdownEventConfiguration (p. 1223)

AWS OpsWorks Layer LifeCycleConfiguration ShutdownEventConfiguration

`ShutdownEventConfiguration` is a property of the AWS OpsWorks Layer LifeCycleConfiguration (p. 1222) property that specifies the shutdown event configuration for a lifecycle event.

Syntax

JSON

```
{
  "DelayUntilElbConnectionsDrained" : Boolean,
  "ExecutionTimeout" : Integer
}
```

YAML

```
DelayUntilElbConnectionsDrained: Boolean
ExecutionTimeout: Integer
```

Properties

`DelayUntilElbConnectionsDrained`

Indicates whether to wait for connections to drain from the Elastic Load Balancing load balancers.

Required: No

Type: Boolean

`ExecutionTimeout`

The time, in seconds, that AWS OpsWorks waits after a shutdown event has been triggered before shutting down an instance.

Required: No

Type: Integer

AWS OpsWorks LoadBasedAutoScaling Type

Describes the load-based automatic scaling configuration for an AWS::OpsWorks::Layer (p. 856) resource type. For more information, see SetLoadBasedAutoScaling in the *AWS OpsWorks Stacks API Reference*.

Syntax

JSON

```
{
  "DownScaling" : { AutoScalingThresholds },
  "Enable" : Boolean,
  "UpScaling" : { AutoScalingThresholds }
}
```

YAML

```
DownScaling:
  AutoScalingThresholds
Enable: Boolean
UpScaling:
  AutoScalingThresholds
```

Properties

`DownScaling`

The threshold below which the instances are scaled down (stopped). If the load falls below this threshold for a specified amount of time, AWS OpsWorks stops a specified number of instances.

Required: No

Type: AWS OpsWorks AutoScalingThresholds Type (p. 1220)

`Enable`

Whether to enable automatic load-based scaling for the layer.

Required: No

Type: Boolean

`UpScaling`

The threshold above which the instances are scaled up (added). If the load exceeds this thresholds for a specified amount of time, AWS OpsWorks starts a specified number of instances.

Required: No

Type: AWS OpsWorks AutoScalingThresholds Type (p. 1220)

AWS OpsWorks Instance BlockDeviceMapping

`BlockDeviceMappings` is a property of the AWS::OpsWorks::Instance (p. 848) resource that defines the block devices that are mapped to an AWS OpsWorks instance.

Syntax

JSON

```
{
```

```
  "DeviceName" : String,
  "Ebs" : EbsBlockDevice (p. 1225),
  "NoDevice" : String,
  "VirtualName" : String
}
```

YAML

```
DeviceName: String
Ebs:
EbsBlockDevice (p. 1225)
NoDevice: String
VirtualName: String
```

Properties

`DeviceName`

The name of the device that is exposed to the instance, such as `/dev/dsh` or `xvdh`. For the root device, you can use the explicit device name or you can set this parameter to `ROOT_DEVICE`. If you set the parameter to `ROOT_DEVICE`, AWS OpsWorks provides the correct device name.

Required: No

Type: String

`Ebs`

Configuration information about the Amazon Elastic Block Store (Amazon EBS) volume.

Required: Conditional You can specify either the `VirtualName` or `Ebs`, but not both.

Type: AWS OpsWorks Instance BlockDeviceMapping EbsBlockDevice (p. 1225)

`NoDevice`

Suppresses the device that is specified in the block device mapping of the AWS OpsWorks instance Amazon Machine Image (AMI).

Required: No

Type: String

`VirtualName`

The name of the virtual device. The name must be in the form `ephemeralX`, where *X* is a number equal to or greater than zero (0), for example, `ephemeral0`.

Required: Conditional You can specify either the `VirtualName` or `Ebs`, but not both.

Type: String

AWS OpsWorks Instance BlockDeviceMapping EbsBlockDevice

`EbsBlockDevice` is a property of the AWS OpsWorks Instance BlockDeviceMapping (p. 1224) property that defines a block device for an Amazon Elastic Block Store (Amazon EBS) volume.

Syntax

JSON

```
{
  "DeleteOnTermination" : Boolean,
  "Iops" : Integer,
  "SnapshotId" : String,
  "VolumeSize" : Integer,
  "VolumeType" : String
}
```

YAML

```
DeleteOnTermination: Boolean
Iops: Integer
SnapshotId: String
VolumeSize: Integer
VolumeType: String
```

Properties

`DeleteOnTermination`

Indicates whether to delete the volume when the instance is terminated.

Required: No

Type: Boolean

`Iops`

The number of I/O operations per second (IOPS) that the volume supports. For more information, see Iops for the `EbsBlockDevice` action in the *Amazon EC2 API Reference*.

Required: No

Type: Integer

`SnapshotId`

The snapshot ID of the volume that you want to use. If you specify both the `SnapshotId` and `VolumeSize` properties, `VolumeSize` must be equal to or greater than the size of the snapshot.

Required: No

Type: String

`VolumeSize`

The volume size, in Gibibytes (GiB). If you specify both the `SnapshotId` and `VolumeSize` properties, `VolumeSize` must be equal to or greater than the size of the snapshot. For more information about specifying volume size, see VolumeSize for the `EbsBlockDevice` action in the *Amazon EC2 API Reference*.

Required: No

Type: Integer

`VolumeType`

The volume type. For more information about specifying the volume type, see VolumeType for the `EbsBlockDevice` action in the *Amazon EC2 API Reference*.

Required: No

Type: String

AWS OpsWorks Recipes Type

Describes custom event recipes for the AWS::OpsWorks::Layer (p. 856) resource type that AWS OpsWorks runs after the standard event recipes. For more information, see AWS OpsWorks Lifecycle Events in the *AWS OpsWorks User Guide*.

Syntax

JSON

```
{
  "Configure" : [ String, ... ],
  "Deploy" : [ String, ... ],
  "Setup" : [ String, ... ],
  "Shutdown" : [ String, ... ],
  "Undeploy" : [ String, ... ]
}
```

YAML

```
Configure:
  - String
Deploy:
  - String
Setup:
  - String
Shutdown:
  - String
Undeploy:
  - String
```

Properties

`Configure`

Custom recipe names to be run following a Configure event. The event occurs on all of the stack's instances when an instance enters or leaves the online state.

Required: No

Type: List of strings

`Deploy`

Custom recipe names to be run following a Deploy event. The event occurs when you run a `deploy` command, typically to deploy an application to a set of application server instances.

Required: No

Type: List of strings

`Setup`

Custom recipe names to be run following a Setup event. This event occurs on a new instance after it successfully boots.

Required: No

Type: List of strings

`Shutdown`

Custom recipe names to be run following a Shutdown event. This event occurs after you direct AWS OpsWorks to shut an instance down before the associated Amazon EC2 instance is actually terminated.

Required: No

Type: List of strings

`Undeploy`

Custom recipe names to be run following a Undeploy event. This event occurs when you delete an app or run an `undeploy` command to remove an app from a set of application server instances.

Required: No

Type: List of strings

AWS OpsWorks Source Type

Describes the information required to retrieve a cookbook or app from a repository for the AWS::OpsWorks::Stack (p. 862) or AWS::OpsWorks::App (p. 843) resource types.

For more information and valid values, see Source in the *AWS OpsWorks Stacks API Reference*.

Syntax

JSON

```
{
  "Password" : String,
  "Revision" : String,
  "SshKey" : String,
  "Type" : String,
  "Url" : String,
  "Username" : String
}
```

YAML

```
Password: String
Revision: String
SshKey: String
Type: String
Url: String
Username: String
```

Properties

`Password`

This parameter depends on the repository type. For Amazon S3 bundles, set `Password` to the appropriate IAM secret access key. For HTTP bundles, Git repositories, and Subversion repositories, set `Password` to the appropriate password.

Required: No

Type: String

`Revision`

The application's version. With AWS OpsWorks, you can deploy new versions of an application. One of the simplest approaches is to have branches or revisions in your repository that represent different versions that can potentially be deployed.

Required: No

Type: String

`SshKey`

The repository's SSH key. For more information, see Using Git Repository SSH Keys in the *AWS OpsWorks User Guide*.

To pass in an SSH key as a parameter, see the following example:

```
"Parameters" : {
  "GitSSHKey" : {
    "Description" : "Change SSH key newlines to commas.",
    "Type" : "CommaDelimitedList",
    "NoEcho" : "true"
  },

...

  "CustomCookbooksSource": {
    "Revision" : { "Ref": "GitRevision"},
    "SshKey" : { "Fn::Join" : [ "\n", { "Ref": "GitSSHKey"} ] },
    "Type": "git",
    "Url": { "Ref": "GitURL"}
  }

...
```

Required: No

Type: String

`Type`

The repository type.

Required: No

Type: String

`Url`

The source URL.

Required: No

Type: String

`Username`

This parameter depends on the repository type. For Amazon S3 bundles, set `Username` to the appropriate IAM access key ID. For HTTP bundles, Git repositories, and Subversion repositories, set `Username` to the appropriate user name.

Required: No

Type: String

AWS OpsWorks SslConfiguration Type

Describes an SSL configuration for the AWS::OpsWorks::App (p. 843) resource type.

Syntax

JSON

```
{
  "Certificate" : String,
  "Chain" : String,
  "PrivateKey" : String
}
```

YAML

```
Certificate: String
Chain: String
PrivateKey: String
```

Properties

`Certificate`

The contents of the certificate's `domain.crt` file.

Required: Yes

Type: String

`Chain`

An intermediate certificate authority key or client authentication.

Required: No

Type: String

`PrivateKey`

The private key; the contents of the certificate's `domain.kex` file.

Required: Yes

Type: String

AWS OpsWorks Stack ElasticIp

`ElasticIps` is a property of the AWS::OpsWorks::Stack (p. 862) resource that registers an Elastic IP address with an AWS OpsWorks stack.

Syntax

JSON

```
{
  "Ip" : String,
  "Name" : String
}
```

YAML

```
Ip: String
Name: String
```

Properties

`Ip`

The Elastic IP address.

Required: Yes

Type: String

`Name`

A name for the Elastic IP address.

Required: No

Type: String

AWS OpsWorks Stack RdsDbInstance

`RdsDbInstance` is a property of the AWS::OpsWorks::Stack (p. 862) resource that registers an Amazon Relational Database Service (Amazon RDS) DB instance with an AWS OpsWorks stack.

Syntax

JSON

```
{
  "DbPassword" : String,
  "DbUser" : String,
  "RdsDbInstanceArn" : String
}
```

YAML

```
DbPassword: String
```

```
DbUser: String
RdsDbInstanceArn: String
```

Properties

`DbPassword`

The password of the registered database.

Required: Yes

Type: String

`DbUser`

The master user name of the registered database.

Required: Yes

Type: String

`RdsDbInstanceArn`

The Amazon Resource Name (ARN) of the Amazon RDS DB instance to register with the AWS OpsWorks stack.

Required: Yes

Type: String

AWS OpsWorks StackConfigurationManager Type

Describes the stack configuration manager for the AWS::OpsWorks::Stack (p. 862) resource type. For more information, see StackConfigurationManager in the *AWS OpsWorks Stacks API Reference*.

Syntax

JSON

```
{
  "Name" : String,
  "Version" : String
}
```

YAML

```
Name: String
Version: String
```

Properties

`Name`

The name of the configuration manager.

Required: No

Type: String

`Version`

The Chef version.

Required: No

Type: String

AWS OpsWorks TimeBasedAutoScaling Type

Describes the automatic time-based scaling configuration for an AWS::OpsWorks::Instance (p. 848) resource type. For more information, see SetTimeBasedAutoScaling in the *AWS OpsWorks Stacks API Reference*.

Syntax

JSON

```
{
  "Friday" : { Integer : String, ... },
  "Monday" : { Integer : String, ... },
  "Saturday" : { Integer : String, ... },
  "Sunday" : { Integer : String, ... },
  "Thursday" : { Integer : String, ... },
  "Tuesday" : { Integer : String, ... },
  "Wednesday" : { Integer : String, ... }
}
```

YAML

```
Friday:
  - Integer: String
Monday:
  - Integer: String
Saturday:
  - Integer: String
Sunday:
  - Integer: String
Thursday:
  - Integer: String
Tuesday:
  - Integer: String
Wednesday:
  - Integer: String
```

Properties

For each day of the week, the schedule consists of a set of key–value pairs, where the key is the time period (a UTC hour) of `0` – `23` and the value indicates whether the instance should be online (`on`) or offline (`off`) for the specified period.

`Friday`

The schedule for Friday.

Required: No

Type: String to string map

`Monday`

The schedule for Monday.

Required: No

Type: String to string map

`Saturday`

The schedule for Saturday.

Required: No

Type: String to string map

`Sunday`

The schedule for Sunday.

Required: No

Type: String to string map

`Thursday`

The schedule for Thursday.

Required: No

Type: String to string map

`Tuesday`

The schedule for Tuesday.

Required: No

Type: String to string map

`Wednesday`

The schedule for Wednesday.

Required: No

Type: String to string map

AWS OpsWorks VolumeConfiguration Type

Describes the Amazon EBS volumes for the AWS::OpsWorks::Layer (p. 856) resource type.

Syntax

JSON

```
{
  "Iops" : Integer,
  "MountPoint" : String,
  "NumberOfDisks" : Integer,
  "RaidLevel" : Integer,
```

```
  "Size" : Integer,
  "VolumeType" : String
}
```

YAML

```
Iops: Integer
MountPoint: String
NumberOfDisks: Integer
RaidLevel: Integer
Size: Integer
VolumeType: String
```

Properties

`Iops`

The number of I/O operations per second (IOPS) to provision for the volume.

Required: Conditional. If you specify `io1` for the volume type, you must specify this property.

Type: Integer

`MountPoint`

The volume mount point, such as `/dev/sdh`.

Required: Yes

Type: String

`NumberOfDisks`

The number of disks in the volume.

Required: Yes

Type: Integer

`RaidLevel`

The volume RAID level.

Required: No

Type: Integer

`Size`

The volume size.

Required: Yes

Type: Integer

`VolumeType`

The type of volume, such as magnetic or SSD. For valid values, see VolumeConfiguration in the *AWS OpsWorks Stacks API Reference*.

Required: No

Type: String

Amazon Redshift Parameter Type

Describes parameters for the AWS::Redshift::ClusterParameterGroup (p. 917) resource type.

Syntax

JSON

```
{
  "ParameterName" : String,
  "ParameterValue" : String
}
```

YAML

```
ParameterName: String
ParameterValue: String
```

Properties

`ParameterName`

The name of the parameter.

Required: Yes

Type: String

`ParameterValue`

The value of the parameter.

Required: Yes

Type: String

AWS CloudFormation Resource Tags Type

You can use the AWS CloudFormation Resource Tags property to apply tags to resources, which can help you identify and categorize those resources. You can tag only resources for which AWS CloudFormation supports tagging. For information about which resources you can tag with AWS CloudFormation, see the individual resources in AWS Resource Types Reference (p. 390).

> **Note**
> Tagging implementations might vary by resource. For example, AWS::AutoScaling::AutoScalingGroup provides an additional, required `PropagateAtLaunch` property as part of its tagging scheme.

In addition to any tags you define, AWS CloudFormation automatically creates the following stack-level tags with the prefix `aws:`:

- `aws:cloudformation:`*`logical-id`*
- `aws:cloudformation:`*`stack-id`*
- `aws:cloudformation:`*`stack-name`*

All stack-level tags, including automatically created tags, are propagated to resources that AWS CloudFormation supports. Currently, tags are not propagated to Amazon EBS volumes that are created from block device mappings.

Syntax

JSON

```
{
  "Key (p. 1237)" : String,
  "Value (p. 1237)" : String
}
```

YAML

```
Key (p. 1237): String
Value (p. 1237): String
```

Properties

`Key`
: The key name of the tag. You can specify a value that is 1 to 127 Unicode characters in length and cannot be prefixed with `aws:`. You can use any of the following characters: the set of Unicode letters, digits, whitespace, `_`, `.`, `/`, `=`, `+`, and `-`.

  Required: Yes

  Type: String

`Value`
: The value for the tag. You can specify a value that is 1 to 255 Unicode characters in length and cannot be prefixed with `aws:`. You can use any of the following characters: the set of Unicode letters, digits, whitespace, `_`, `.`, `/`, `=`, `+`, and `-`.

  Required: Yes

  Type: String

See Also

- Setting Stack Options (p. 76)
- Viewing Stack Data and Resources (p. 78)

Amazon RDS OptionGroup OptionConfigurations

Use the `OptionConfigurations` property to configure an option and its settings for an AWS::RDS::OptionGroup (p. 907) resource.

Syntax

JSON

```
{
  "DBSecurityGroupMemberships" : [ String, ... ],
  "OptionName" : String,
```

```
  "OptionSettings" : [ OptionSettings, ... ],
  "Port" : Integer,
  "VpcSecurityGroupMemberships" : [ String, ... ]
}
```

YAML

```
DBSecurityGroupMemberships:
  - String
OptionName: String
OptionSettings:
  - OptionSettings
Port: Integer
VpcSecurityGroupMemberships:
  - String
```

Properties

`DBSecurityGroupMemberships`

A list of database security group names for this option. If the option requires access to a port, the security groups must allow access to that port. If you specify this property, don't specify the `VPCSecurityGroupMemberships` property.

Required: No

Type: List of strings

`OptionName`

The name of the option. For more information about options, see Working with Option Groups in the *Amazon Relational Database Service User Guide*.

Required: Yes

Type: String

`OptionSettings`

The settings for this option.

Required: No

Type: Amazon RDS OptionGroup OptionConfigurations OptionSettings (p. 1239)

`Port`

The port number that this option uses.

Required: No

Type: Integer

`VpcSecurityGroupMemberships`

A list of VPC security group IDs for this option. If the option requires access to a port, the security groups must allow access to that port. If you specify this property, don't specify the `DBSecurityGroupMemberships` property.

Required: No

Type: List of strings

Amazon RDS OptionGroup OptionConfigurations OptionSettings

Use the `OptionSettings` property to specify settings for an option in the `OptionConfigurations` (p. 1237) property.

Syntax

JSON

```
{
  "Name" : String,
  "Value" : String
}
```

YAML

```
Name: String
Value: String
```

Properties

For more information about option settings, see Working with Option Groups in the *Amazon Relational Database Service User Guide*

`Name`

The name of the option setting that you want to specify.

Required: No

Type: String

`Value`

The value of the option setting.

Required: No

Type: String

Amazon RDS Security Group Rule

The Amazon RDS security group rule is an embedded property of the AWS::RDS::DBSecurityGroup (p. 898) type.

Syntax

JSON

```
{
  "CIDRIP (p. 1240)": String,
  "EC2SecurityGroupId (p. 1240)": String,
  "EC2SecurityGroupName (p. 1240)": String,
```

```
  "EC2SecurityGroupOwnerId (p. 1240)": String
}
```

YAML

```
CIDRIP (p. 1240): String
EC2SecurityGroupId (p. 1240): String
EC2SecurityGroupName (p. 1240): String
EC2SecurityGroupOwnerId (p. 1240): String
```

Properties

`CIDRIP`

The IP range to authorize.

For an overview of CIDR ranges, go to the Wikipedia Tutorial.
Type: String

Required: No

Update requires: Replacement (p. 90)

`EC2SecurityGroupId`

Id of the VPC or EC2 Security Group to authorize.

For VPC DB Security Groups, use EC2SecurityGroupId. For EC2 Security Groups, use EC2SecurityGroupOwnerId and either EC2SecurityGroupName or EC2SecurityGroupId.
Type: String

Required: No

Update requires: Replacement (p. 90)

`EC2SecurityGroupName`

Name of the EC2 Security Group to authorize.

For VPC DB Security Groups, use EC2SecurityGroupId. For EC2 Security Groups, use EC2SecurityGroupOwnerId and either EC2SecurityGroupName or EC2SecurityGroupId.
Type: String

Required: No

Update requires: Replacement (p. 90)

`EC2SecurityGroupOwnerId`

AWS Account Number of the owner of the EC2 Security Group specified in the EC2SecurityGroupName parameter. The AWS Access Key ID is not an acceptable value.

For VPC DB Security Groups, use EC2SecurityGroupId. For EC2 Security Groups, use EC2SecurityGroupOwnerId and either EC2SecurityGroupName or EC2SecurityGroupId.
Type: String

Required: No

Update requires: Replacement (p. 90)

Route 53 AliasTarget Property

`AliasTarget` is a property of the AWS::Route53::RecordSet (p. 930) resource.

For more information about alias resource record sets, see Creating Alias Resource Record Sets in the *Amazon Route 53 Developer Guide*.

Syntax

JSON

```
{
  "DNSName" : String,
  "EvaluateTargetHealth" : Boolean,
  "HostedZoneId" : String
}
```

YAML

```
DNSName: String
EvaluateTargetHealth: Boolean
HostedZoneId: String
```

Properties

`DNSName`

The DNS name of the load balancer, the domain name of the CloudFront distribution, the website endpoint of the Amazon S3 bucket, or another record set in the same hosted zone that is the target of the alias.

Type: String

Required: Yes

`EvaluateTargetHealth`

Whether Amazon Route 53 checks the health of the resource record sets in the alias target when responding to DNS queries. For more information about using this property, see EvaluateTargetHealth in the *Amazon Route 53 API Reference*.

Type: Boolean

Required: No

`HostedZoneId`

The hosted zone ID. For load balancers, use the canonical hosted zone ID of the load balancer. For Amazon S3, use the hosted zone ID for your bucket's website endpoint. For CloudFront, use `Z2FDTNDATAQYW2`. For a list of hosted zone IDs of other services, see the relevant service in the AWS Regions and Endpoints.

Type: String

Required: Yes

Amazon Route 53 Record Set GeoLocation Property

The `GeoLocation` property is part of the AWS::Route53::RecordSet (p. 930) resource that describes how Amazon Route 53 responds to DNS queries based on the geographic location of the query.

Syntax

JSON

```
{
  "ContinentCode" : String,
  "CountryCode" : String,
  "SubdivisionCode" : String
}
```

YAML

```
ContinentCode: String
CountryCode: String
SubdivisionCode: String
```

Properties

`ContinentCode`

All DNS queries from the continent that you specified are routed to this resource record set. If you specify this property, omit the `CountryCode` and `SubdivisionCode` properties.

For valid values, see the ContinentCode element in the *Amazon Route 53 API Reference*.

Type: String

Required: Conditional. You must specify this or the `CountryCode` property.

`CountryCode`

All DNS queries from the country that you specified are routed to this resource record set. If you specify this property, omit the `ContinentCode` property.

For valid values, see the CountryCode element in the *Amazon Route 53 API Reference*.

Type: String

Required: Conditional. You must specify this or the `ContinentCode` property.

`SubdivisionCode`

If you specified `US` for the country code, you can specify a state in the United States. All DNS queries from the state that you specified are routed to this resource record set. If you specify this property, you must specify `US` for the `CountryCode` and omit the `ContinentCode` property.

For valid values, see the SubdivisionCode element in the *Amazon Route 53 API Reference*.

Type: String

Required: No

Amazon Route 53 HealthCheckConfig

The `HealthCheckConfig` property is part of the AWS::Route53::HealthCheck (p. 925) resource that describes a health check that Amazon Route 53 uses before responding to a DNS query.

Syntax

JSON

```
{
  "AlarmIdentifier" : AlarmIdentifier,
  "ChildHealthChecks" : [ String, ... ],
  "EnableSNI" : Boolean,
  "FailureThreshold" : Integer,
  "FullyQualifiedDomainName" : String,
  "HealthThreshold" : Integer,
  "InsufficientDataHealthStatus" : String,
  "Inverted" : Boolean,
  "IPAddress" : String,
  "MeasureLatency" : Boolean,
  "Port" : Integer,
  "RequestInterval" : Integer,
  "ResourcePath" : String,
  "SearchString" : String,
  "Type" : String
}
```

YAML

```
AlarmIdentifier:  AlarmIdentifier
ChildHealthChecks:
   - String
EnableSNI: Boolean
FailureThreshold: Integer
FullyQualifiedDomainName: String
HealthThreshold: Integer
InsufficientDataHealthStatus: String
Inverted: Boolean
IPAddress: String
MeasureLatency: Boolean
Port: Integer
RequestInterval: Integer
ResourcePath: String
SearchString: String
Type: String
```

Properties

`AlarmIdentifier`

Identifies the CloudWatch alarm that you want Amazon Route 53 health checkers to use to determine whether this health check is healthy.

Type: Amazon Route 53 AlarmIdentifier (p. 1246)

Required: No

`ChildHealthChecks`

(CALCULATED Health Checks Only) A complex type that contains one `ChildHealthCheck` element for each health check that you want to associate with a `CALCULATED` health check.

Required: No

Type: List of strings

`EnableSNI`

Specifies whether you want Amazon Route 53 to send the value of `FullyQualifiedDomainName` to the endpoint in the `client_hello` message during TLS negotiation. This allows the endpoint to respond to `HTTPS` health check requests with the applicable SSL/TLS certificate. For more information, see http://docs.aws.amazon.com/Route53/latest/APIReference/API_HealthCheckConfig.html.

Required: No

Type: Boolean

`FailureThreshold`

The number of consecutive health checks that an endpoint must pass or fail for Amazon Route 53 to change the current status of the endpoint from unhealthy to healthy or healthy to unhealthy. For more information, see How Amazon Route 53 Determines Whether an Endpoint Is Healthy in the *Amazon Route 53 Developer Guide*.

Required: No

Type: Integer

`FullyQualifiedDomainName`

If you specified the `IPAddress` property, the value that you want Amazon Route 53 to pass in the host header in all health checks except for TCP health checks. If you don't specify an IP address, the domain that Amazon Route 53 sends a DNS request to. Amazon Route 53 uses the IP address that the DNS returns to check the health of the endpoint.

Required: Conditional

Type: String

`HealthThreshold`

The number of child health checks that are associated with a `CALCULATED` health that Amazon Route 53 must consider healthy for the `CALCULATED` health check to be considered healthy.

Required: No

Type: Integer

`InsufficientDataHealthStatus`

When Amazon CloudWatch has insufficient data about the metric to determine the alarm state, the status that you want Amazon Route 53 to assign to the health check (`Healthy`, `Unhealthy`, or `LastKnownStatus`).

Required: No

Type: String

`Inverted`

Specifies whether you want Amazon Route 53 to invert the status of a health check, for example, to consider a health check unhealthy when it otherwise would be considered healthy.

Required: No

Type: Boolean

`IPAddress`

The IPv4 IP address of the endpoint on which you want Amazon Route 53 to perform health checks. If you don't specify an IP address, Amazon Route 53 sends a DNS request to resolve the domain name that you specify in the `FullyQualifiedDomainName` property.

Required: No

Type: String

`MeasureLatency`

Specifies whether you want Amazon Route 53 to measure the latency between health checkers in multiple AWS regions and your endpoint and display CloudWatch latency graphs on the **Health Checks** page in the Amazon Route 53 console.

Required: No

Type: Boolean

Update requires: Replacement (p. 90)

`Port`

The port on the endpoint on which you want Amazon Route 53 to perform health checks.

Required: Conditional. Required when you specify `TCP` for the `Type` property.

Type: Integer

`RequestInterval`

The number of seconds between the time that Amazon Route 53 gets a response from your endpoint and the time that it sends the next health check request. Each Amazon Route 53 health checker makes requests at this interval. For valid values, see the RequestInterval element in the *Amazon Route 53 API Reference*.

Required: No

Type: Integer

Update requires: Replacement (p. 90)

`ResourcePath`

The path that you want Amazon Route 53 to request when performing health checks. The path can be any value for which your endpoint returns an HTTP status code of `2`*xx* or `3`*xx* when the endpoint is healthy, such as `/docs/route53-health-check.html`.

Required: No

Type: String

`SearchString`

If the value of the `Type` property is `HTTP_STR_MATCH` or `HTTPS_STR_MATCH`, the string that you want Amazon Route 53 to search for in the response body from the specified resource. If the string appears in the response body, Amazon Route 53 considers the resource healthy.

Required: No

Type: String

`Type`

The type of health check that you want to create. This indicates how Amazon Route 53 determines whether an endpoint is healthy. You can specify `HTTP`, `HTTPS`, `HTTP_STR_MATCH`, `HTTPS_STR_MATCH`, or `TCP`. For information about the different types, see the `Type` element in the *Amazon Route 53 API Reference*.

Required: Yes

Type: String

Update requires: Replacement (p. 90)

Amazon Route 53 AlarmIdentifier

The `AlarmIdentifier` subproperty describes the name and Region that are associated with an Amazon Route 53 HealthCheckConfig (p. 1242) property.

Syntax

JSON

```
{
  "Name" : String,
  "Region" : String
}
```

YAML

```
Name: String
Region: String
```

Properties

`Name`

The name of the Amazon CloudWatch alarm that you want Amazon Route 53 health checkers to use to determine whether this health check is healthy.

Required: Yes

Type: String

`Region`

A complex type that identifies the CloudWatch alarm that you want Amazon Route 53 health checkers to use to determine whether this health check is healthy. For example, `us-west-2`.

Required: Yes

Type: String

Amazon Route 53 HealthCheckTags

The `HealthCheckTags` property describes key-value pairs that are associated with an AWS::Route53::HealthCheck (p. 925) resource.

Syntax

JSON

```
{
```

```
  "Key" : String,
  "Value" : String
}
```

YAML

```
Key: String
Value: String
```

Properties

`Key`

The key name of the tag.

Required: Yes

Type: String

`Value`

The value for the tag.

Required: Yes

Type: String

Amazon Route 53 HostedZoneConfig Property

The `HostedZoneConfig` property is part of the AWS::Route53::HostedZone (p. 927) resource that can contain a comment about the hosted zone.

Syntax

JSON

```
{
  "Comment" : String
}
```

YAML

```
Comment: String
```

Properties

`Comment`

Any comments that you want to include about the hosted zone.

Type: String

Required: No

Amazon Route 53 HostedZoneTags

The `HostedZoneTags` property describes key-value pairs that are associated with an AWS::Route53::HostedZone (p. 927) resource.

Syntax

JSON

```
{
  "Key" : String,
  "Value" : String
}
```

YAML

```
Key: String
Value: String
```

Properties

`Key`

The key name of the tag.

Required: Yes

Type: String

`Value`

The value for the tag.

Required: Yes

Type: String

Amazon Route 53 HostedZoneVPCs

The `HostedZoneVPCs` property is part of the AWS::Route53::HostedZone (p. 927) resource that specifies the VPCs to associate with the hosted zone.

Syntax

JSON

```
{
  "VPCId" : String,
  "VPCRegion" : String
}
```

YAML

```
VPCId: String
VPCRegion: String
```

Properties

`VPCId`

The ID of the Amazon VPC that you want to associate with the hosted zone.

Required: Yes

Type: String

`VPCRegion`

The region in which the Amazon VPC was created as specified in the `VPCId` property.

Required: Yes

Type: String

Amazon S3 Cors Configuration

Describes the cross-origin access configuration for objects in an AWS::S3::Bucket (p. 937) resource.

Syntax

JSON

```
{
  "CorsRules" : [ CorsRules, ... ]
}
```

YAML

```
CorsRules:
  - CorsRules
```

Properties

`CorsRules`

A set of origins and methods that you allow.

Required: Yes

Type: Amazon S3 Cors Configuration Rule (p. 1249)

Amazon S3 Cors Configuration Rule

Describes cross-origin access rules for the Amazon S3 Cors Configuration (p. 1249) property.

Syntax

JSON

```
{
```

```
  "AllowedHeaders" : [ String, ... ],
  "AllowedMethods" : [ String, ... ],
  "AllowedOrigins" : [ String, ... ],
  "ExposedHeaders" : [ String, ... ],
  "Id" : String,
  "MaxAge" : Integer
}
```

YAML

```
AllowedHeaders:
  - String
AllowedMethods:
  - String
AllowedOrigins:
  - String
ExposedHeaders:
  - String
Id: String
MaxAge: Integer
```

Properties

`AllowedHeaders`

Headers that are specified in the `Access-Control-Request-Headers` header. These headers are allowed in a preflight OPTIONS request. In response to any preflight OPTIONS request, Amazon S3 returns any requested headers that are allowed.

Required: No

Type: List of strings

`AllowedMethods`

An HTTP method that you allow the origin to execute. The valid values are `GET`, `PUT`, `HEAD`, `POST`, and `DELETE`.

Required: Yes

Type: List of strings

`AllowedOrigins`

An origin that you allow to send cross-domain requests.

Required: Yes

Type: List of strings

`ExposedHeaders`

One or more headers in the response that are accessible to client applications (for example, from a JavaScript XMLHttpRequest object).

Required: No

Type: List of strings

`Id`

A unique identifier for this rule. The value cannot be more than 255 characters.

Required: No

Type: String

`MaxAge`

The time in seconds that your browser is to cache the preflight response for the specified resource.

Required: No

Type: Integer

Amazon S3 Lifecycle Configuration

Describes the lifecycle configuration for objects in an AWS::S3::Bucket (p. 937) resource.

Syntax

JSON

```
{
  "Rules" : [ Lifecycle Rule, ... ]
}
```

YAML

```
Rules:
  - Lifecycle Rule
```

Properties

`Rules`

A lifecycle rule for individual objects in an S3 bucket.

Required: Yes

Type: Amazon S3 Lifecycle Rule (p. 1251)

Amazon S3 Lifecycle Rule

Describes lifecycle rules for the Amazon S3 Lifecycle Configuration (p. 1251) property.

Syntax

JSON

```
{
  "ExpirationDate" : String,
  "ExpirationInDays" : Integer,
  "Id" : String,
  "NoncurrentVersionExpirationInDays" : Integer,
  "NoncurrentVersionTransition (deprecated)" : NoncurrentVersionTransition type,
  "NoncurrentVersionTransitions" : [ NoncurrentVersionTransition type, ... ],
  "Prefix" : String,
  "Status" : String,
```

```
  "Transition (deprecated)" : Transition type,
  "Transitions" : [ Transition type, ... ]
}
```

YAML

```
ExpirationDate: String
ExpirationInDays: Integer
Id: String
NoncurrentVersionExpirationInDays: Integer
NoncurrentVersionTransition (deprecated):
  NoncurrentVersionTransition type
NoncurrentVersionTransitions:
  - NoncurrentVersionTransition type
Prefix: String
Status: String
Transition (deprecated):
  Transition type
Transitions:
  - Transition type
```

Properties

`ExpirationDate`

Indicates when objects are deleted from Amazon S3 and Amazon Glacier. The date value must be in ISO 8601 format. The time is always midnight UTC. If you specify an expiration and transition time, you must use the same time unit for both properties (either in days or by date). The expiration time must also be later than the transition time.

Required: Conditional. You must specify at least one of the following properties: `ExpirationDate`, `ExpirationInDays`, `NoncurrentVersionExpirationInDays`, `NoncurrentVersionTransition`, `NoncurrentVersionTransitions`, `Transition`, or `Transitions`.

Type: String

`ExpirationInDays`

Indicates the number of days after creation when objects are deleted from Amazon S3 and Amazon Glacier. If you specify an expiration and transition time, you must use the same time unit for both properties (either in days or by date). The expiration time must also be later than the transition time.

Required: Conditional. You must specify at least one of the following properties: `ExpirationDate`, `ExpirationInDays`, `NoncurrentVersionExpirationInDays`, `NoncurrentVersionTransition`, `NoncurrentVersionTransitions`, `Transition`, or `Transitions`.

Type: Integer

`Id`

A unique identifier for this rule. The value cannot be more than 255 characters.

Required: No

Type: String

`NoncurrentVersionExpirationInDays`

For buckets with versioning enabled (or suspended), specifies the time, in days, between when a new version of the object is uploaded to the bucket and when old versions of the object expire. When object versions expire, Amazon S3 permanently deletes them. If you specify a transition and expiration time, the expiration time must be later than the transition time.

Required: Conditional. You must specify at least one of the following properties: `ExpirationDate`, `ExpirationInDays`, `NoncurrentVersionExpirationInDays`, `NoncurrentVersionTransition`, `NoncurrentVersionTransitions`, `Transition`, or `Transitions`.

Type: Integer

`NoncurrentVersionTransition` (deprecated)

For buckets with versioning enabled (or suspended), specifies when non-current objects transition to a specified storage class. If you specify a transition and expiration time, the expiration time must be later than the transition time. If you specify this property, don't specify the `NoncurrentVersionTransitions` property.

Required: Conditional. You must specify at least one of the following properties: `ExpirationDate`, `ExpirationInDays`, `NoncurrentVersionExpirationInDays`, `NoncurrentVersionTransition`, `NoncurrentVersionTransitions`, `Transition`, or `Transitions`.

Type: Amazon S3 Lifecycle Rule NoncurrentVersionTransition (p. 1254)

`NoncurrentVersionTransitions`

For buckets with versioning enabled (or suspended), one or more transition rules that specify when non-current objects transition to a specified storage class. If you specify a transition and expiration time, the expiration time must be later than the transition time. If you specify this property, don't specify the `NoncurrentVersionTransition` property.

Required: Conditional. You must specify at least one of the following properties: `ExpirationDate`, `ExpirationInDays`, `NoncurrentVersionExpirationInDays`, `NoncurrentVersionTransition`, `NoncurrentVersionTransitions`, `Transition`, or `Transitions`.

Type: List of Amazon S3 Lifecycle Rule NoncurrentVersionTransition (p. 1254)

`Prefix`

Object key prefix that identifies one or more objects to which this rule applies.

Required: No

Type: String

`Status`

Specify either `Enabled` or `Disabled`. If you specify `Enabled`, Amazon S3 executes this rule as scheduled. If you specify `Disabled`, Amazon S3 ignores this rule.

Required: Yes

Type: String

`Transition` (deprecated)

Specifies when an object transitions to a specified storage class. If you specify an expiration and transition time, you must use the same time unit for both properties (either in days or by date). The expiration time must also be later than the transition time. If you specify this property, don't specify the `Transitions` property.

Required: Conditional. You must specify at least one of the following properties: `ExpirationDate`, `ExpirationInDays`, `NoncurrentVersionExpirationInDays`, `NoncurrentVersionTransition`, `NoncurrentVersionTransitions`, `Transition`, or `Transitions`.

Type: Amazon S3 Lifecycle Rule Transition (p. 1254)

`Transitions`

One or more transition rules that specify when an object transitions to a specified storage class. If you specify an expiration and transition time, you must use the same time unit for both properties (either in days or by date). The expiration time must also be later than the transition time. If you specify this property, don't specify the `Transition` property.

Required: Conditional. You must specify at least one of the following properties: `ExpirationDate`, `ExpirationInDays`, `NoncurrentVersionExpirationInDays`, `NoncurrentVersionTransition`, `NoncurrentVersionTransitions`, `Transition`, or `Transitions`.

Type: List of Amazon S3 Lifecycle Rule Transition (p. 1254)

Amazon S3 Lifecycle Rule NoncurrentVersionTransition

`NoncurrentVersionTransition` is a property of the Amazon S3 Lifecycle Rule (p. 1251) property that describes when noncurrent objects transition to a specified storage class.

Syntax

JSON

```
{
  "StorageClass" : String,
  "TransitionInDays" : Integer
}
```

YAML

```
StorageClass: String
TransitionInDays: Integer
```

Properties

`StorageClass`

The storage class to which you want the object to transition, such as `GLACIER`. For valid values, see the `StorageClass` request element of the PUT Bucket lifecycle action in the *Amazon Simple Storage Service API Reference*.

Required: Yes

Type: String

`TransitionInDays`

The number of days between the time that a new version of the object is uploaded to the bucket and when old versions of the object are transitioned to the specified storage class.

Required: Yes

Type: Integer

Amazon S3 Lifecycle Rule Transition

Describes when an object transitions to a specified storage class for the Amazon S3 Lifecycle Rule (p. 1251) property.

Syntax

JSON

```
{
  "StorageClass" : String,
  "TransitionDate" : String,
  "TransitionInDays" : Integer
}
```

YAML

```
StorageClass: String
TransitionDate: String
TransitionInDays: Integer
```

Properties

`StorageClass`

The storage class to which you want the object to transition, such as `GLACIER`. For valid values, see the `StorageClass` request element of the PUT Bucket lifecycle action in the *Amazon Simple Storage Service API Reference*.

Required: Yes

Type: String

`TransitionDate`

Indicates when objects are transitioned to the specified storage class. The date value must be in ISO 8601 format. The time is always midnight UTC.

Required: Conditional

Type: String

`TransitionInDays`

Indicates the number of days after creation when objects are transitioned to the specified storage class.

Required: Conditional

Type: Integer

Amazon S3 Logging Configuration

Describes where logs are stored and the prefix that Amazon S3 assigns to all log object keys for an AWS::S3::Bucket (p. 937) resource. These logs track requests to an Amazon S3 bucket. For more information, see PUT Bucket logging in the *Amazon Simple Storage Service API Reference*.

Syntax

JSON

```
{
  "DestinationBucketName" : String,
```

```
  "LogFilePrefix" : String
}
```

YAML

```
DestinationBucketName: String
LogFilePrefix: String
```

Properties

`DestinationBucketName`

The name of an Amazon S3 bucket where Amazon S3 store server access log files. You can store log files in any bucket that you own. By default, logs are stored in the bucket where the `LoggingConfiguration` property is defined.

Required: No

Type: String

`LogFilePrefix`

A prefix for the all log object keys. If you store log files from multiple Amazon S3 buckets in a single bucket, you can use a prefix to distinguish which log files came from which bucket.

Required: No

Type: String

Amazon S3 NotificationConfiguration

Describes the notification configuration for an AWS::S3::Bucket (p. 937) resource.

Note
If you create the target resource and related permissions in the same template, you might have a circular dependency.
For example, you might use the AWS::Lambda::Permission resource to grant the S3 bucket to invoke a Lambda function. However, AWS CloudFormation can't create the S3 bucket until the bucket has permission to invoke the function (AWS CloudFormation checks if the S3 bucket can invoke the function). If you're using Refs to pass the bucket name, this leads to a circular dependency.
To avoid this dependency, you can create all resources without specifying the notification configuration. Then, update the stack with a notification configuration.

Syntax

JSON

```
{
  "LambdaConfigurations" : [ Lambda Configuration, ... ],
  "QueueConfigurations" : [ Queue Configuration, ... ],
  "TopicConfigurations" : [ Topic Configuration, ... ]
}
```

YAML

```
LambdaConfigurations:
```

```
  - Lambda Configuration
QueueConfigurations:
  - Queue Configuration
TopicConfigurations:
  - Topic Configuration
```

Properties

`LambdaConfigurations`

The AWS Lambda functions to invoke and the events for which to invoke the functions.

Required: No

Type: Amazon Simple Storage Service NotificationConfiguration LambdaConfigurations (p. 1259)

`QueueConfigurations`

The Amazon Simple Queue Service queues to publish messages to and the events for which to publish messages.

Required: No

Type: Amazon Simple Storage Service NotificationConfiguration QueueConfigurations (p. 1261)

`TopicConfigurations`

The topic to which notifications are sent and the events for which notification are generated.

Required: No

Type: Amazon S3 NotificationConfiguration TopicConfigurations (p. 1262)

Amazon S3 NotificationConfiguration Config Filter

`Filter` is a property of the `LambdaConfigurations (p. 1259)`, `QueueConfigurations (p. 1261)`, and `TopicConfigurations (p. 1262)` properties that describes the filtering rules that determine the Amazon Simple Storage Service (Amazon S3) objects for which to send notifications.

Syntax

JSON

```
{
  "S3Key" : S3 Key
}
```

YAML

```
S3Key:
  S3 Key
```

Properties

`S3Key`

Amazon S3 filtering rules that describe for which object key names to send notifications.

Required: Yes

Type: Amazon S3 NotificationConfiguration Config Filter S3Key (p. 1258)

Amazon S3 NotificationConfiguration Config Filter S3Key

`S3Key` is a property of the Amazon S3 NotificationConfiguration Config Filter (p. 1257) property that specifies the key names of Amazon Simple Storage Service (Amazon S3) objects for which to send notifications.

Syntax

JSON

```
{
  "Rules" : [ Rule, ... ]
}
```

YAML

```
Rules:
  - Rule
```

Properties

`Rules`

The object key name to filter on and whether to filter on the suffix or prefix of the key name.

Required: Yes

Type: List of Amazon S3 NotificationConfiguration Config Filter S3Key Rules (p. 1258)

Amazon S3 NotificationConfiguration Config Filter S3Key Rules

`Rules` is a property of the Amazon S3 NotificationConfiguration Config Filter S3Key (p. 1258) property that describes the Amazon Simple Storage Service (Amazon S3) object key name to filter on and whether to filter on the suffix or prefix of the key name.

Syntax

JSON

```
{
  "Name" : String,
  "Value" : String
}
```

YAML

```
Name: String
Value: String
```

Properties

`Name`

Whether the filter matches the prefix or suffix of object key names. For valid values, see the `Name` request element of the PUT Bucket notification action in the *Amazon Simple Storage Service API Reference*.

Required: Yes

Type: String

`Value`

The value that the filter searches for in object key names.

Required: Yes

Type: String

Amazon Simple Storage Service NotificationConfiguration LambdaConfigurations

`LambdaConfigurations` is a property of the Amazon S3 NotificationConfiguration (p. 1256) property that describes the AWS Lambda (Lambda) functions to invoke and the events for which to invoke them.

Syntax

JSON

```
{
  "Event" : String,
  "Filter" : Filter,
  "Function" : String
}
```

YAML

```
Event: String
Filter:
  Filter
Function: String
```

Properties

`Event`

The S3 bucket event for which to invoke the Lambda function. For more information, see Supported Event Types in the *Amazon Simple Storage Service Developer Guide*.

Required: Yes

Type: String

`Filter`

The filtering rules that determine which objects invoke the Lambda function. For example, you can create a filter so that only image files with a `.jpg` extension invoke the function when they are added to the S3 bucket.

Required: No

Type: Amazon S3 NotificationConfiguration Config Filter (p. 1257)

`Function`

The Amazon Resource Name (ARN) of the Lambda function that Amazon S3 invokes when the specified event type occurs.

Required: Yes

Type: String

Example

The following example creates a `NotificationConfiguration` for Lambda using an S3 bucket named `EncryptionServiceBucket`.

Note
The `BucketName` is unique and the `Value` contains a file extension without a period (`.`).

JSON

```
"EncryptionServiceBucket" : {
  "Type" : "AWS::S3::Bucket",
  "Properties" : {
    "BucketName" : { "Fn::Sub" : "${User}-encryption-service" },
    "NotificationConfiguration" : {
      "LambdaConfigurations" : [{
        "Function" : { "Ref" : "LambdaDeploymentArn" },
        "Event" : "s3:ObjectCreated:*",
        "Filter" : {
          "S3Key" : {
            "Rules" : [{
              "Name" : "suffix",
              "Value" : "zip"
            }]
          }
        }
      }]
    }
  }
}
```

YAML

```
EncryptionServiceBucket:
  Type: "AWS::S3::Bucket"
  Properties:
    BucketName: !Sub ${User}-encryption-service
    NotificationConfiguration:
      LambdaConfigurations:
        -
          Function: !Ref LambdaDeploymentArn
```

```
          Event: "s3:ObjectCreated:*"
          Filter:
            S3Key:
              Rules:
                -
                  Name: suffix
                  Value: zip
```

Amazon Simple Storage Service NotificationConfiguration QueueConfigurations

`QueueConfigurations` is a property of the Amazon S3 NotificationConfiguration (p. 1256) property that describes the S3 bucket events about which you want to send messages to Amazon SQS and the queues to which you want to send them.

Syntax

JSON

```
{
  "Event" : String,
  "Filter" : Filter,
  "Queue" : String
}
```

YAML

```
Event: String
Filter:
  Filter
Queue: String
```

Properties

`Event`

The S3 bucket event about which you want to publish messages to Amazon Simple Queue Service (Amazon SQS). For more information, see Supported Event Types in the *Amazon Simple Storage Service Developer Guide*.

Required: Yes

Type: String

`Filter`

The filtering rules that determine for which objects to send notifications. For example, you can create a filter so that Amazon Simple Storage Service (Amazon S3) sends notifications only when image files with a `.jpg` extension are added to the bucket.

Required: No

Type: Amazon S3 NotificationConfiguration Config Filter (p. 1257)

`Queue`

The Amazon Resource Name (ARN) of the Amazon SQS queue that Amazon S3 publishes messages to when the specified event type occurs.

Required: Yes

Type: String

Amazon S3 NotificationConfiguration TopicConfigurations

Describes the topic and events for the Amazon S3 NotificationConfiguration (p. 1256) property.

Syntax

JSON

```
{
  "Event" : String,
  "Filter" : Filter,
  "Topic" : String
}
```

YAML

```
Event: String
Filter:
  Filter
Topic: String
```

Properties

`Event`

The Amazon Simple Storage Service (Amazon S3) bucket event about which to send notifications. For more information, see Supported Event Types in the *Amazon Simple Storage Service Developer Guide*.

Required: Yes

Type: String

`Filter`

The filtering rules that determine for which objects to send notifications. For example, you can create a filter so that Amazon Simple Storage Service (Amazon S3) sends notifications only when image files with a `.jpg` extension are added to the bucket.

Required: No

Type: Amazon S3 NotificationConfiguration Config Filter (p. 1257)

`Topic`

The Amazon SNS topic Amazon Resource Name (ARN) to which Amazon S3 reports the specified events.

Required: Yes

Type: String

Amazon S3 ReplicationConfiguration

`ReplicationConfiguration` is a property of the AWS::S3::Bucket (p. 937) resource that specifies replication rules and the AWS Identity and Access Management (IAM) role Amazon Simple Storage Service (Amazon S3) uses to replicate objects.

Syntax

JSON

```
{
  "Role" : String,
  "Rules" : [ Rule, ... ]
}
```

YAML

```
Role: String
Rules:
  - Rule
```

Properties

`Role`

The Amazon Resource Name (ARN) of an AWS Identity and Access Management (IAM) role that Amazon S3 assumes when replicating objects. For more information, see How to Set Up Cross-Region Replication in the *Amazon Simple Storage Service Developer Guide*.

Required: Yes

Type: String

`Rules`

A replication rule that specifies which objects to replicate and where they are stored.

Required: Yes

Type: List of Amazon S3 ReplicationConfiguration Rules (p. 1263)

Amazon S3 ReplicationConfiguration Rules

`Rules` is a property of the Amazon S3 ReplicationConfiguration (p. 1263) property that specifies which Amazon Simple Storage Service (Amazon S3) objects to replicate and where to store them.

Syntax

JSON

```
{
  "Destination" : String,
  "Id" : String,
  "Prefix" : String,
  "Status" : String
}
```

YAML

```
Destination: String
Id: String
Prefix: String
Status: String
```

Properties

`Destination`

Defines the destination where Amazon S3 stores replicated objects.

Required: Yes

Type: Amazon S3 ReplicationConfiguration Rules Destination (p. 1264)

`Id`

A unique identifier for the rule. If you don't specify a value, AWS CloudFormation generates a random ID.

Required: No

Type: String

`Prefix`

An object prefix. This rule applies to all Amazon S3 objects with this prefix. To specify all objects in an S3 bucket, specify an empty string.

Required: Yes

Type: String

`Status`

Whether the rule is enabled. For valid values, see the `Status` element of the PUT Bucket replication action in the *Amazon Simple Storage Service API Reference*.

Required: Yes

Type: String

Amazon S3 ReplicationConfiguration Rules Destination

`Destination` is a property of the Amazon S3 ReplicationConfiguration Rules (p. 1263) property that specifies which Amazon Simple Storage Service (Amazon S3) bucket to store replicated objects and their storage class.

Syntax

JSON

```
{
  "Bucket" : String,
  "StorageClass" : String
}
```

YAML

```
Bucket: String
StorageClass: String
```

Properties

`Bucket`

The Amazon resource name (ARN) of an S3 bucket where Amazon S3 stores replicated objects. This destination bucket must be in a different region than your source bucket.

If you have multiple rules in your replication configuration, specify the same destination bucket for all of the rules.

Required: Yes

Type: String

`StorageClass`

The storage class to use when replicating objects, such as standard or reduced redundancy. By default, Amazon S3 uses the storage class of the source object to create object replica. For valid values, see the `StorageClass` element of the PUT Bucket replication action in the *Amazon Simple Storage Service API Reference*.

Required: No

Type: String

Amazon S3 Versioning Configuration

Describes the versioning state of an AWS::S3::Bucket (p. 937) resource. For more information, see PUT Bucket versioning in the *Amazon Simple Storage Service API Reference*.

Syntax

JSON

```
{
  "Status" : String
}
```

YAML

```
Status: String
```

Properties

`Status`

The versioning state of an Amazon S3 bucket. If you enable versioning, you must suspend versioning to disable it.

Valid values include `Enabled` and `Suspended`. The default is `Suspended`.

Required: Yes

Type: String

Amazon S3 Website Configuration Property

`WebsiteConfiguration` is an embedded property of the AWS::S3::Bucket (p. 937) resource.

Syntax

JSON

```
{
  "ErrorDocument" : String,
  "IndexDocument" : String,
  "RedirectAllRequestsTo" : Redirect all requests rule,
  "RoutingRules" : [ Routing rule, ... ]
}
```

YAML

```
ErrorDocument: String
IndexDocument: String
RedirectAllRequestsTo:
  Redirect all requests rule
RoutingRules:
  - Routing rule
```

Properties

`ErrorDocument`

The name of the error document for the website.

Required: No

Type: String

`IndexDocument`

The name of the index document for the website.

Required: Yes

Type: String

`RedirectAllRequestsTo`

The redirect behavior for every request to this bucket's website endpoint.

Important
If you specify this property, you cannot specify any other property.

Required: No

Type: Amazon S3 Website Configuration Redirect All Requests To Property (p. 1267)

`RoutingRules`

Rules that define when a redirect is applied and the redirect behavior.

Required: No

Type: List of Amazon S3 Website Configuration Routing Rules Property (p. 1268)

Example

```
"S3Bucket" : {
   "Type" : "AWS::S3::Bucket",
   "Properties" : {
      "AccessControl" : "PublicRead",
      "WebsiteConfiguration" : {
         "IndexDocument" : "index.html",
         "ErrorDocument" : "error.html"
      }
   }
}
```

See Also

- Custom Error Document Support in the *Amazon Simple Storage Service Developer Guide*
- Index Document Support in the *Amazon Simple Storage Service Developer Guide*

Amazon S3 Website Configuration Redirect All Requests To Property

The `RedirectAllRequestsTo` code is an embedded property of the Amazon S3 Website Configuration Property (p. 1266) property that describes the redirect behavior of all requests to a website endpoint of an Amazon S3 bucket.

Syntax

JSON

```
{
  "HostName" : String,
  "Protocol" : String
}
```

YAML

```
HostName: String
Protocol: String
```

Properties

`HostName`

Name of the host where requests are redirected.

Required: Yes

Type: String

Protocol

Protocol to use (`http` or `https`) when redirecting requests. The default is the protocol that is used in the original request.

Required: No

Type: String

Amazon S3 Website Configuration Routing Rules Property

The `RoutingRules` property is an embedded property of the Amazon S3 Website Configuration Property (p. 1266) property. This property describes the redirect behavior and when a redirect is applied.

Syntax

JSON

```
{
  "RedirectRule" : Redirect rule,
  "RoutingRuleCondition" : Routing rule condition
}
```

YAML

```
RedirectRule:
  Redirect rule
RoutingRuleCondition:
  Routing rule condition
```

Properties

RedirectRule

Redirect requests to another host, to another page, or with another protocol.

Required: Yes

Type: Amazon S3 Website Configuration Routing Rules Redirect Rule Property (p. 1268)

RoutingRuleCondition

Rules that define when a redirect is applied.

Required: No

Type: Amazon S3 Website Configuration Routing Rules Routing Rule Condition Property (p. 1270)

Amazon S3 Website Configuration Routing Rules Redirect Rule Property

The `RedirectRule` property is an embedded property of the Amazon S3 Website Configuration Routing Rules Property (p. 1268) that describes how requests are redirected. In the event of an error, you can specify a different error code to return.

Syntax

JSON

```
{
  "HostName" : String,
  "HttpRedirectCode" : String,
  "Protocol" : String,
  "ReplaceKeyPrefixWith" : String,
  "ReplaceKeyWith" : String
}
```

YAML

```
HostName: String
HttpRedirectCode: String
Protocol: String
ReplaceKeyPrefixWith: String
ReplaceKeyWith: String
```

Properties

HostName

Name of the host where requests are redirected.

Required: No

Type: String

HttpRedirectCode

The HTTP redirect code to use on the response.

Required: No

Type: String

Protocol

The protocol to use in the redirect request.

Required: No

Type: String

ReplaceKeyPrefixWith

The object key prefix to use in the redirect request. For example, to redirect requests for all pages with the prefix `docs/` (objects in the `docs/` folder) to the `documents/` prefix, you can set the `KeyPrefixEquals` property in routing condition property to `docs/`, and set the *ReplaceKeyPrefixWith* property to `documents/`.

> **Important**
> If you specify this property, you cannot specify the `ReplaceKeyWith` property.

Required: No

Type: String

ReplaceKeyWith

The specific object key to use in the redirect request. For example, redirect request to `error.html`.

Important
If you specify this property, you cannot specify the `ReplaceKeyPrefixWith` property.

Required: No

Type: String

Amazon S3 Website Configuration Routing Rules Routing Rule Condition Property

The `RoutingRuleCondition` property is an embedded property of the Amazon S3 Website Configuration Routing Rules Property (p. 1268) that describes a condition that must be met for a redirect to apply.

Syntax

JSON

```
{
  "HttpErrorCodeReturnedEquals" : String,
  "KeyPrefixEquals" : String
}
```

YAML

```
HttpErrorCodeReturnedEquals: String
KeyPrefixEquals: String
```

Properties

HttpErrorCodeReturnedEquals

Applies this redirect if the error code equals this value in the event of an error.

Required: Conditional. You must specify at least one condition property.

Type: String

KeyPrefixEquals

The object key name prefix when the redirect is applied. For example, to redirect requests for `ExamplePage.html`, set the key prefix to `ExamplePage.html`. To redirect request for all pages with the prefix `docs/`, set the key prefix to `docs/`, which identifies all objects in the `docs/` folder.

Required: Conditional. You must at least one condition property.

Type: String

Amazon EC2 Systems Manager Association Targets

`Targets` is a property of the AWS::SSM::Association (p. 966) resource that specifies the targets for an Amazon EC2 Systems Manager (SSM) document.

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

JSON

```
{
  "Key" : String,
  "Values" : [ String, ... ]
}
```

YAML

```
Key: String
Values:
  - String
```

Properties

`Key`

The name of the criteria that EC2 instances must meet. For valid keys, see the Target data type in the *Amazon EC2 Systems Manager API Reference*.

Required: Yes

Type: String

`Values`

The value of the criteria. SSM runs targeted commands on EC2 instances that match the criteria. For more information, see the Target data type in the *Amazon EC2 Systems Manager API Reference*.

Required: Yes

Type: List of strings

Amazon SNS Subscription Property Type

`Subscription` is an embedded property of the `AWS::SNS::Topic` (p. 954) resource that describes the subscription endpoints for an Amazon Simple Notification Service (Amazon SNS) topic.

Syntax

JSON

```
{
  "Endpoint" : String,
  "Protocol" : String
}
```

YAML

```
Endpoint: String
```

```
Protocol: String
```

Properties

`Endpoint`

The subscription's endpoint (format depends on the protocol). For more information, see the Subscribe Endpoint parameter in the *Amazon Simple Notification Service API Reference*.

Required: Yes

Type: String

`Protocol`

The subscription's protocol. For more information, see the Subscribe Protocol parameter in the *Amazon Simple Notification Service API Reference*.

Required: Yes

Type: String

Amazon SQS RedrivePolicy

The RedrivePolicy type is a property of the AWS::SQS::Queue (p. 958) resource.

Syntax

JSON

```
{
  "deadLetterTargetArn" : String,
  "maxReceiveCount" : Integer
}
```

YAML

```
deadLetterTargetArn: String
maxReceiveCount: Integer
```

Properties

`deadLetterTargetArn`

The Amazon Resource Name (ARN) of the dead letter queue to which the messages are sent to after the `maxReceiveCount` value has been exceeded.

Required: Yes

Type: String

`maxReceiveCount`

The number of times a message is delivered to the source queue before being sent to the dead letter queue.

Required: Yes

Type: Integer

AWS WAF ByteMatchSet ByteMatchTuples

`ByteMatchTuples` is a property of the AWS::WAF::ByteMatchSet (p. 979) resource that specifies settings for an AWS WAF `ByteMatchSet` resource, such as the bytes (typically a string that corresponds with ASCII characters) that you want AWS WAF to search for in web requests.

Syntax

JSON

```
{
  "FieldToMatch" : Field to match,
  "PositionalConstraint" : String,
  "TargetString" : String,
  "TargetStringBase64" : String,
  "TextTransformation" : String
}
```

YAML

```
FieldToMatch:
  Field to match
PositionalConstraint: String
TargetString: String
TargetStringBase64: String
TextTransformation: String
```

Properties

`FieldToMatch`

The part of a web request that you want AWS WAF to search, such as a specific header or a query string.

Required: Yes

Type: AWS WAF ByteMatchSet ByteMatchTuples FieldToMatch (p. 1274)

`PositionalConstraint`

How AWS WAF finds matches within the web request part in which you are searching. For valid values, see the `PositionalConstraint` content for the ByteMatchTuple data type in the *AWS WAF API Reference*.

Required: Yes

Type: String

`TargetString`

The value that AWS WAF searches for. AWS CloudFormation base64 encodes this value before sending it to AWS WAF.

AWS WAF searches for this value in a specific part of web requests, which you define in the `FieldToMatch` property.

Valid values depend on the `Type` value in the `FieldToMatch` property. For example, for a `METHOD` type, you must specify HTTP methods such as `DELETE`, `GET`, `HEAD`, `OPTIONS`, `PATCH`, `POST`, and `PUT`. For more information, see the `TargetString` content for the ByteMatchTuple data type in the *AWS WAF API Reference*.

Required: Conditional. You must specify this property or the `TargetStringBase64` property.

Type: String

`TargetStringBase64`

The base64-encoded value that AWS WAF searches for. AWS CloudFormation sends this value to AWS WAF without encoding it.

AWS WAF searches for this value in a specific part of web requests, which you define in the `FieldToMatch` property.

Valid values depend on the `Type` value in the `FieldToMatch` property. For example, for a `METHOD` type, you must specify HTTP methods such as `DELETE`, `GET`, `HEAD`, `OPTIONS`, `PATCH`, `POST`, and `PUT`. For more information, see the `TargetString` content for the ByteMatchTuple data type in the *AWS WAF API Reference*.

Required: Conditional. You must specify this property or the `TargetString` property.

Type: String

`TextTransformation`

Specifies how AWS WAF processes the target string value. Text transformations eliminate some of the unusual formatting that attackers use in web requests in an effort to bypass AWS WAF. If you specify a transformation, AWS WAF transforms the target string value before inspecting a web request for a match.

For example, AWS WAF can replace whitespace characters (such as `\t` and `\n`) with a single space. For valid values, see the `TextTransformation` content for the ByteMatchTuple data type in the *AWS WAF API Reference*.

Required: Yes

Type: String

AWS WAF ByteMatchSet ByteMatchTuples FieldToMatch

`FieldToMatch` is a property of the AWS WAF ByteMatchSet ByteMatchTuples (p. 1273) property that specifies the part of a web request that you want AWS WAF to search, such as a specific header or a query string.

Syntax

JSON

```
{
  "Data" : String,
  "Type" : String
}
```

YAML

```
Data: String
Type: String
```

Properties

`Data`

If you specify `HEADER` for the `Type` property, the name of the header that AWS WAF searches for, such as `User-Agent` or `Referer`. If you specify any other value for the `Type` property, do not specify this property.

Required: Conditional

Type: String

`Type`

The part of the web request in which AWS WAF searches for the target string. For valid values, see FieldToMatch in the *AWS WAF API Reference*.

Required: Yes

Type: String

AWS WAF IPSet IPSetDescriptors

`IPSetDescriptors` is a property of the AWS::WAF::IPSet (p. 983) resource that specifies the IP address type and IP address range (in CIDR notation) from which web requests originate.

Syntax

JSON

```
{
  "Type" : String,
  "Value" : String
}
```

YAML

```
Type: String
Value: String
```

Properties

`Type`

The IP address type, such as `IPV4`. For valid values, see the `Type` contents of the IPSetDescriptor data type in the *AWS WAF API Reference*.

Required: Yes

Type: String

`Value`

An IP address (in CIDR notation) that AWS WAF permits, blocks, or counts. For example, to specify a single IP address such as `192.0.2.44`, specify `192.0.2.44/32`. To specify a range of IP addresses such as `192.0.2.0` to `192.0.2.255`, specify `192.0.2.0/24`.

Required: Yes

Type: String

AWS WAF Rule Predicates

`Predicates` is a property of the AWS::WAF::Rule (p. 986) resource that specifies the `ByteMatchSet`, `IPSet`, `SizeConstraintSet`, `SqlInjectionMatchSet`, or `XssMatchSet` objects to include in an AWS WAF rule. If you add more than one predicate to a rule, an incoming request must match all of the specifications in the predicates to be allowed or blocked.

Syntax

JSON

```
{
  "DataId" : String,
  "Negated" : Boolean,
  "Type" : String
}
```

YAML

```
DataId: String
Negated: Boolean
Type: String
```

Properties

`DataId`

The unique identifier of a predicate, such as the ID of a `ByteMatchSet` or `IPSet`.

Required: Yes

Type: String

`Negated`

Whether to use the settings or the negated settings that you specified in the `ByteMatchSet`, `IPSet`, `SizeConstraintSet`, `SqlInjectionMatchSet`, or `XssMatchSet` objects.

Specify `false` if you want AWS WAF to allow, block, or count requests based on the settings in the specified `ByteMatchSet`, `IPSet`, `SizeConstraintSet`, `SqlInjectionMatchSet`, or `XssMatchSet` objects. For example, if an `IPSet` object includes the IP address `192.0.2.44`, AWS WAF allows, blocks, or counts requests originating from that IP address.

Specify `true` if you want AWS WAF to allow, block, or count requests based on the negated settings in the `ByteMatchSet`, `IPSet`, `SizeConstraintSet`, `SqlInjectionMatchSet`, or `XssMatchSet` objects. For example, if an `IPSet` object includes the IP address `192.0.2.44`, AWS WAF allows, blocks, or counts requests originating from all IP addresses except 192.0.2.44.

Required: Yes

Type: Boolean

`Type`

The type of predicate in a rule, such as an `IPSet` (`IPMatch`). For valid values, see the `Type` contents of the Predicate data type in the *AWS WAF API Reference*.

Required: Yes

Type: String

AWS WAF SizeConstraintSet SizeConstraint

`SizeConstraint` is a property of the AWS::WAF::SizeConstraintSet (p. 988) resource that specifies a size constraint and which part of a web request that you want AWS WAF to constrain.

Syntax

JSON

```
{
  "ComparisonOperator" : String,
  "FieldToMatch" : Field to match,
  "Size" : String,
  "TextTransformation" : String
}
```

YAML

```
ComparisonOperator: String
FieldToMatch:
  Field to match
Size: String
TextTransformation: String
```

Properties

`ComparisonOperator`

The type of comparison that you want AWS WAF to perform. AWS WAF uses this value in combination with the `Size` and `FieldToMatch` property values to check if the size constraint is a match. For more information and valid values, see the `ComparisonOperator` content for the SizeConstraint data type in the *AWS WAF API Reference*.

Required: Yes

Type: String

`FieldToMatch`

The part of a web request that you want AWS WAF to search, such as a specific header or a query string.

Required: Yes

Type: AWS WAF SizeConstraintSet SizeConstraint FieldToMatch (p. 1278)

`Size`

The size in bytes that you want AWS WAF to compare against the size of the specified `FieldToMatch`. AWS WAF uses `Size` in combination with the `ComparisonOperator` and `FieldToMatch` property values to check if the size constraint of a web request is a match. For more information and valid values, see the `Size` content for the SizeConstraint data type in the *AWS WAF API Reference*.

Required: Yes

Type: Integer

`TextTransformation`

Specifies how AWS WAF processes the `FieldToMatch` property before inspecting a request for a match. Text transformations eliminate some of the unusual formatting that attackers use in web requests in an effort to bypass AWS WAF. If you specify a transformation, AWS WAF transforms the `FieldToMatch` before inspecting a web request for a match.

For example, AWS WAF can replace white space characters (such as `\t` and `\n`) with a single space. For valid values, see the `TextTransformation` content for the SizeConstraint data type in the *AWS WAF API Reference*.

Required: Yes

Type: String

AWS WAF SizeConstraintSet SizeConstraint FieldToMatch

`FieldToMatch` is a property of the AWS WAF SizeConstraintSet SizeConstraint (p. 1277) property that specifies the part of a web request that you want AWS WAF to check for a size constraint, such as a specific header or a query string.

Syntax

JSON

```
{
  "Data" : String,
  "Type" : String
}
```

YAML

```
Data: String
Type: String
```

Properties

`Data`

If you specify `HEADER` for the `Type` property, the name of the header that AWS WAF searches for, such as `User-Agent` or `Referer`. If you specify any other value for the `Type` property, do not specify this property.

Required: Conditional

Type: String

`Type`

The part of the web request in which AWS WAF searches for the target string. For valid values, see FieldToMatch in the *AWS WAF API Reference*.

Required: Yes

Type: String

AWS WAF SqlInjectionMatchSet SqlInjectionMatchTuples

`SqlInjectionMatchTuples` is a property of the AWS::WAF::SqlInjectionMatchSet (p. 991) resource that specifies the parts of web requests that AWS WAF inspects for SQL code.

Syntax

JSON

```
{
  "FieldToMatch" : Field to match,
  "TextTransformation" : String
}
```

YAML

```
FieldToMatch:
  Field to match
TextTransformation: String
```

Properties

`FieldToMatch`

The part of a web request that you want AWS WAF to search, such as a specific header or a query string.

Required: Yes

Type: AWS WAF ByteMatchSet ByteMatchTuples FieldToMatch (p. 1274)

`TextTransformation`

Text transformations eliminate some of the unusual formatting that attackers use in web requests in an effort to bypass AWS WAF. If you specify a transformation, AWS WAF transforms the target string value before inspecting a web request for a match. For valid values, see the `TextTransformation` content for the SqlInjectionMatchTuple data type in the *AWS WAF API Reference*.

Required: Yes

Type: String

AWS WAF SqlInjectionMatchSet SqlInjectionMatchTuples FieldToMatch

`FieldToMatch` is a property of the AWS WAF ByteMatchSet ByteMatchTuples (p. 1273) property that specifies the part of a web request that you want AWS WAF to search, such as a specific header or a query string.

Syntax

JSON

```
{
  "Data" : String,
  "Type" : String
}
```

YAML

```
Data: String
Type: String
```

Properties

`Data`

If you specify `HEADER` for the `Type` property, the name of the header that AWS WAF searches for, such as `User-Agent` or `Referer`. If you specify any other value for the `Type` property, do not specify this property.

Required: Conditional

Type: String

`Type`

The part of the web request in which AWS WAF searches for the target string. For valid values, see FieldToMatch in the *AWS WAF API Reference*.

Required: Yes

Type: String

AWS WAF XssMatchSet XssMatchTuple

`XssMatchTuple` is a property of the AWS::WAF::XssMatchSet (p. 999) resource that specifies the part of a web request that you want AWS WAF to inspect for cross-site scripting attacks.

Syntax

JSON

```
{
  "FieldToMatch" : Field to match,
```

```
  "TextTransformation" : String
}
```

YAML

```
FieldToMatch:
  Field to match
TextTransformation: String
```

Properties

`FieldToMatch`

The part of a web request that you want AWS WAF to search, such as a specific header or a query string.

Required: Yes

Type: AWS WAF XssMatchSet XssMatchTuple FieldToMatch (p. 1281)

`TextTransformation`

Specifies how AWS WAF processes the `FieldToMatch` property before inspecting a request for a match. Text transformations eliminate some of the unusual formatting that attackers use in web requests in an effort to bypass AWS WAF. If you specify a transformation, AWS WAF transforms the`FieldToMatch` parameter before inspecting a web request for a match.

For example, AWS WAF can replace white space characters (such as `\t` and `\n`) with a single space. For valid values, see the `TextTransformation` content for the XssMatchTuple data type in the *AWS WAF API Reference*.

Required: Yes

Type: String

AWS WAF XssMatchSet XssMatchTuple FieldToMatch

`FieldToMatch` is a property of the AWS WAF XssMatchSet XssMatchTuple FieldToMatch (p. 1281) property that specifies the part of a web request that you want AWS WAF to search, such as a specific header or a query string.

Syntax

JSON

```
{
  "Data" : String,
  "Type" : String
}
```

YAML

```
Data: String
```

```
Type: String
```

Properties

`Data`

If you specify `HEADER` for the `Type` property, the name of the header that AWS WAF searches for, such as `User-Agent` or `Referer`. If you specify any other value for the `Type` property, do not specify this property.

Required: Conditional

Type: String

`Type`

The part of the web request in which AWS WAF searches for the target string. For valid values, see FieldToMatch in the *AWS WAF API Reference*.

Required: Yes

Type: String

AWS WAF WebACL Action

`Action` is a property of the AWS::WAF::WebACL (p. 994) resource and the AWS WAF WebACL Rules (p. 1283) property that specifies the action AWS WAF takes when a web request matches or doesn't match all rule conditions.

Syntax

JSON

```
{
  "Type" : String
}
```

YAML

```
Type: String
```

Properties

`Type`

For actions that are associated with a rule, the action that AWS WAF takes when a web request matches all conditions in a rule.

For the default action of a web access control list (ACL), the action that AWS WAF takes when a web request doesn't match all conditions in any rule.

For valid value, see the `Type` contents of the WafAction data type in the *AWS WAF API Reference*.

Required: Yes

Type: String

AWS WAF WebACL Rules

`Rules` is a property of the AWS::WAF::WebACL (p. 994) resource that specifies the rule to associate with an AWS WAF web access control list (ACL) and the rule's settings.

Syntax

JSON

```
{
  "Action" : String,
  "Priority" : Integer,
  "RuleId" : String
}
```

YAML

```
Action: String
Priority: Integer
RuleId: String
```

Properties

`Action`

The action that Amazon CloudFront (CloudFront) or AWS WAF takes when a web request matches all conditions in the rule, such as allow, block, or count the request.

Required: Yes

Type: AWS WAF WebACL Action (p. 1282)

`Priority`

The order in which AWS WAF evaluates the rules in a web ACL. AWS WAF evaluates rules with a lower value before rules with a higher value. The value must be a unique integer. If you have multiple rules in a web ACL, the priority numbers do not need to be consecutive.

Required: Yes

Type: Integer

`RuleId`

The ID of an AWS WAF rule (p. 986) to associate with a web ACL.

Required: Yes

Type: String

AWS CloudFormation Resource Specification

The AWS CloudFormation resource specification is a JSON-formatted text file that defines the resources and properties that AWS CloudFormation supports. The document is a machine-readable, strongly typed specification that you can use to build tools for creating AWS CloudFormation templates. For example, you can use the specification to build auto completion and validation functionality for AWS CloudFormation templates in your IDE (integrated development environment).

The resource specification is organized as both a single file and as a series of files, where each file contains the definition of one resource type. The single and separated files contain identical information. Depending on the tool and your implementation, use the file or files that work for you.

To download the resource specification, see the following table. Currently, the specifications are identical for every region, except for US East (N. Virginia), US East (Ohio), US West (Oregon), and EU (Ireland) Region. These regions contain a resource specification for AWS CodeCommit.

Resource Specification

Region	Single File	All Files
Asia Pacific (Mumbai) Region	CloudFormationResourceSpecification.json	CloudFormationResourceSpecification.zip
Asia Pacific (Seoul) Region	CloudFormationResourceSpecification.json	CloudFormationResourceSpecification.zip
Asia Pacific (Sydney) Region	CloudFormationResourceSpecification.json	CloudFormationResourceSpecification.zip
Asia Pacific (Singapore) Region	CloudFormationResourceSpecification.json	CloudFormationResourceSpecification.zip
Asia Pacific (Tokyo) Region	CloudFormationResourceSpecification.json	CloudFormationResourceSpecification.zip
Canada (Central) Region	CloudFormationResourceSpecification.json	CloudFormationResourceSpecification.zip
EU (Frankfurt) Region	CloudFormationResourceSpecification.json	CloudFormationResourceSpecification.zip
EU (London) Region	CloudFormationResourceSpecification.json	CloudFormationResourceSpecification.zip
EU (Ireland) Region	CloudFormationResourceSpecification.json	CloudFormationResourceSpecification.zip
South America (São Paulo)	CloudFormationResourceSpecification.json	CloudFormationResourceSpecification.zip
US East (N. Virginia)	CloudFormationResourceSpecification.json	CloudFormationResourceSpecification.zip
US East (Ohio)	CloudFormationResourceSpecification.json	CloudFormationResourceSpecification.zip
US West (N. California)	CloudFormationResourceSpecification.json	CloudFormationResourceSpecification.zip
US West (Oregon)	CloudFormationResourceSpecification.json	CloudFormationResourceSpecification.zip

The following example shows the specification for an AWS Key Management Service key resource (`AWS::KMS::Key`). It shows the properties for the `AWS::KMS::Key` resource, which properties are required, the type of allowed value for each property, and their update behavior. For details about the specification, see Specification Format (p. 1285).

```
    "AWS::KMS::Key": {
      "Attributes": {
        "Arn": {
          "PrimitiveType": "String"
        }
      },
      "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-
resource-kms-key.html",
      "Properties": {
        "Description": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-resource-kms-key.html#cfn-kms-key-description",
```

```
          "PrimitiveType": "String",
          "Required": false,
          "UpdateType": "Mutable"
        },
        "EnableKeyRotation": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-resource-kms-key.html#cfn-kms-key-enablekeyrotation",
          "PrimitiveType": "Boolean",
          "Required": false,
          "UpdateType": "Mutable"
        },
        "Enabled": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-resource-kms-key.html#cfn-kms-key-enabled",
          "PrimitiveType": "Boolean",
          "Required": false,
          "UpdateType": "Mutable"
        },
        "KeyPolicy": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-resource-kms-key.html#cfn-kms-key-keypolicy",
          "PrimitiveType": "Json",
          "Required": true,
          "UpdateType": "Mutable"
        },
        "KeyUsage": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-resource-kms-key.html#cfn-kms-key-keyusage",
          "PrimitiveType": "String",
          "Required": false,
          "UpdateType": "Immutable"
        }
      }
    }
```

Specification Format

AWS CloudFormation creates a specification for each resource type (p. 390), such as `AWS::S3::Bucket` or `AWS::EC2::Instance`. The following sections describe the format and each field within the specification.

Topics

Specification Sections

The formal definition for each resource type is organized into three main sections: `PropertyTypes`, `ResourceSpecificationVersion`, and `ResourceTypes`, as shown in the following example:

```
{
  "PropertyTypes": {
    Property specifications (p. 1286)
  },
  "ResourceSpecificationVersion": "Specification version number",
  "ResourceTypes": {
    Resource specification (p. 1287)
  }
```

```
}
```

`PropertyTypes`

For resources that have properties within a property (also known as subproperties), a list of subproperty specifications, such as which properties are required, the type of allowed value for each property, and their update behavior. For more information, see Property Specification (p. 1286).

If a resource doesn't have subproperties, this section is omitted.

`ResourceSpecificationVersion`

The version of the resource specification. The version format is *`majorVersion`*`.`*`minorVersion`*`.`*`patch`*, where each release increments the version number. All resources have the same version number regardless of whether the resource was updated.

AWS CloudFormation increments the patch number when the service makes a backwards-compatible bug fix, such as fixing a broken documentation link. When AWS CloudFormation adds resources or properties that are backwards compatible, it increments the minor version number. For example, later versions of a specification might add additional resource properties to support new features of an AWS service.

Backwards incompatible changes increment the major version number. A backwards incompatible change can result from a change in the resource specification, such as a name change to a field, or a change to a resource, such as the making an optional resource property required.

`ResourceTypes`

The list of resources and information about each resource's properties, such as its property names, which properties are requires, and their update behavior. For more information, see Resource Specification (p. 1287).

> **Note**
> If you view a file that contains the definition of one resource type, this property name is `ResourceType` (singular).

Property Specification

The specification for each property includes the following fields. For subproperties, the property name uses the *`resourceType`*`.`*`subpropertyName`* format.

```
"Property name": {
  "Documentation": "Link to the relevant documentation"
  "DuplicatesAllowed": "true or false",
  "ItemType": "Type of list or map (non-primitive)",
  "PrimitiveItemType": "Type of list or map (primitive)",
  "PrimitiveType": "Type of value (primitive)",
  "Required": "true or false",
  "Type": "Type of value (non-primitive)",
  "UpdateType": "Mutable, Immutable, or Conditional",
}
```

`Documentation`

A link to the *AWS CloudFormation User Guide* that provides information about the property.

`DuplicatesAllowed`

If the value of the `Type` field is `List`, indicates whether AWS CloudFormation allows duplicate values. If the value is `true`, AWS CloudFormation ignores duplicate values. If the value is `false`, AWS CloudFormation returns an error if you submit duplicate values.

`ItemType`

If the value of the `Type` field is `List` or `Map`, indicates the type of list or map if they contain non-primitive types. Otherwise, this field is omitted. For lists or maps that contain primitive types, the `PrimitiveItemType` property indicates the valid value type.

A subproperty name is a valid item type. For example, if the type value is `List` and the item type value is `PortMapping`, you can specify a list of port mapping properties.

`PrimitiveItemType`

If the value of the `Type` field is `List` or `Map`, indicates the type of list or map if they contain primitive types. Otherwise, this field is omitted. For lists or maps that contain non-primitive types, the `ItemType` property indicates the valid value type.

The valid primitive types for lists and maps are `String`, `Long`, `Integer`, `Double`, `Boolean`, or `Timestamp`.

For example, if the type value is `List` and the item type value is `String`, you can specify a list of strings for the property. If the type value is `Map` and the item type value is `Boolean`, you can specify a string to Boolean mapping for the property.

`PrimitiveType`

For primitive values, the valid primitive type for the property. A primitive type is a basic data type for resource property values. The valid primitive types are `String`, `Long`, `Integer`, `Double`, `Boolean`, `Timestamp` or `Json`. If valid values are a non-primitive type, this field is omitted and the `Type` field indicates the valid value type.

`Required`

Indicates whether the property is required.

`Type`

For non-primitive types, valid values for the property. The valid types are a subproperty name, `List` or `Map`. If valid values are a primitive type, this field is omitted and the `PrimitiveType` field indicates the valid value type.

A list is a comma-separated list of values. A map is a set of key-value pairs, where the keys are always strings. The value type for lists and maps are indicated by the `ItemType` or `PrimitiveItemType` field.

`UpdateType`

During a stack update, the update behavior when you add, remove, or modify the property. AWS CloudFormation replaces the resource when you change `Immutable` properties. AWS CloudFormation doesn't replace the resource when you change `mutable` properties. Conditional updates can be mutable or immutable, depending on, for example, which other properties you updated. For more information, see the relevant resource type (p. 390) documentation.

Resource Specification

The specification for each resource type includes the following fields.

```
"Resource type name": {
  "Attributes": {
    "AttributeName": {
      "ItemType": "Return list or map type (non-primitive)",
      "PrimitiveItemType": "Return list or map type (primitive)",
      "PrimitiveType": "Return value type (primitive)",
      "Type": "Return value type (non-primitive)",
    }
  },
  "Documentation": "Link to the relevant documentation",
  "Properties": {
```

```
    Property specifications (p. 1286)
  }
}
```

`Attributes`

A list of resource attributes that you can use in an `Fn::GetAtt` (p. 1324) function. For each attribute, this section provides the attribute name and the type of value that AWS CloudFormation returns.

`ItemType`

If the value of the `Type` field is `List`, indicates the type of list that the `Fn::GetAtt` function returns for the attribute if the list contains non-primitive types. The valid type is a name of a property.

`PrimitiveItemType`

If the value of the `Type` field is `List`, indicates the type of list that the `Fn::GetAtt` function returns for the attribute if the list contains primitive types. For lists that contain non-primitive types, the `ItemType` property indicates the valid value type. The valid primitive types for lists are `String`, `Long`, `Integer`, `Double`, `Boolean`, or `Timestamp`.

For example, if the type value is `List` and the primitive item type value is `String`, the `Fn::GetAtt` function returns a list of strings.

`PrimitiveType`

For primitive return values, the type of primitive value that the `Fn::GetAtt` function returns for the attribute. A primitive type is a basic data type for resource property values. The valid primitive types are `String`, `Long`, `Integer`, `Double`, `Boolean`, `Timestamp` or `Json`.

`Type`

For non-primitive return values, the type of value that the `Fn::GetAtt` function returns for the attribute. The valid types are a property name or `List`.

A list is a comma-separated list of values. The value type for lists are indicated by the `ItemType` or `PrimitiveItemType` field.

`Documentation`

A link to the *AWS CloudFormation User Guide* for information about the resource.

`Properties`

A list of property specifications for the resource. For details, see Property Specification (p. 1286).

Example Resource Specification

The following examples highlight and explain parts of the `AWS::Elasticsearch::Domain` (p. 745) resource specification.

The `AWS::Elasticsearch::Domain` resource type contains subproperties, so the specification includes a `PropertyTypes` section. This section is followed by the `ResourceSpecificationVersion` section, which shows the specification version as `1.0.0`. After the specification version is the `ResourceType` section that specifies the resource type, provides a documentation link, and details the resource's properties.

```
{
  "PropertyTypes": {
    ...

  },
  "ResourceSpecificationVersion": "1.0.0",
  "ResourceType": {
    "AWS::Elasticsearch::Domain": {
```

```
      "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-
resource-elasticsearch-domain.html",
      "Properties": {
        ...

        }
      }
    }
  }
}
```

Focusing on the `ResourceType` section, the following example shows two properties of the `AWS::Elasticsearch::Domain` resource type. The `AdvancedOptions` property is not required and accepts a string to string map. A map is a collection of key-value pairs, where the keys are always strings. The value type is indicated by the `ItemType` field, which is `String`. Therefore, the type is a string to string map. The update behavior for this property is mutable. If update this property, AWS CloudFormation keeps the resource instead of creating a new one and then deleting the old one (an immutable update).

The `SnapshotOptions` property is not required and accepts a subproperty named `SnapshotOptions`. Details of the `SnapshotOptions` subproperty is provided in the `PropertyTypes` section.

```
{
  "PropertyTypes": {
    ...

  },
  "ResourceSpecificationVersion": "1.0.0",
  "ResourceType": {
    "AWS::Elasticsearch::Domain": {
      "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-
resource-elasticsearch-domain.html",
      "Properties": {
        ...

        "AdvancedOptions": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-resource-elasticsearch-domain.html#cfn-elasticsearch-domain-advancedoptions",
          "DuplicatesAllowed": false,
          "ItemType": "String",
          "Required": false,
          "Type": "Map",
          "UpdateType": "Mutable"
        },
        ...

        "SnapshotOptions": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-resource-elasticsearch-domain.html#cfn-elasticsearch-domain-snapshotoptions",
          "Required": false,
          "Type": "SnapshotOptions",
          "UpdateType": "Mutable"
        },
        ...

      }
    }
  }
}
```

In the `PropertyTypes`, the specification lists all of the subproperties of a resource (including nested subproperties). The following example details the `AWS::Elasticsearch::Domain.SnapshotOptions` subproperty. It contains one property named `AutomatedSnapshotStartHour`, which is not required and accepts integer value types.

```
"PropertyTypes": {
  ...

  "AWS::Elasticsearch::Domain.SnapshotOptions": {
    "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-
properties-elasticsearch-domain-snapshotoptions.html",
    "Properties": {
      "AutomatedSnapshotStartHour": {
        "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-properties-elasticsearch-domain-snapshotoptions.html#cfn-elasticsearch-domain-
snapshotoptions-automatedsnapshotstarthour",
        "Required": false,
        "Type": "Integer",
        "UpdateType": "Mutable"
      }
    }
  },
  ...

}
```

For your reference, the following example provides the entire `AWS::Elasticsearch::Domain` resource specification.

```
{
  "PropertyTypes": {
    "AWS::Elasticsearch::Domain.EBSOptions": {
      "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-
properties-elasticsearch-domain-ebsoptions.html",
      "Properties": {
        "EBSEnabled": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-properties-elasticsearch-domain-ebsoptions.html#cfn-elasticsearch-domain-ebsoptions-
ebsenabled",
          "Required": false,
          "Type": "Boolean",
          "UpdateType": "Mutable"
        },
        "Iops": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-properties-elasticsearch-domain-ebsoptions.html#cfn-elasticsearch-domain-ebsoptions-
iops",
          "Required": false,
          "Type": "Integer",
          "UpdateType": "Mutable"
        },
        "VolumeSize": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-properties-elasticsearch-domain-ebsoptions.html#cfn-elasticsearch-domain-ebsoptions-
volumesize",
          "Required": false,
          "Type": "Integer",
          "UpdateType": "Mutable"
        },
        "VolumeType": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-properties-elasticsearch-domain-ebsoptions.html#cfn-elasticsearch-domain-ebsoptions-
volumetype",
          "Required": false,
          "Type": "String",
          "UpdateType": "Mutable"
        }
      }
    },
```

```
    "AWS::Elasticsearch::Domain.ElasticsearchClusterConfig": {
      "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-
properties-elasticsearch-domain-elasticsearchclusterconfig.html",
      "Properties": {
        "DedicatedMasterCount": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-properties-elasticsearch-domain-elasticsearchclusterconfig.html#cfn-elasticsearch-
domain-elasticseachclusterconfig-dedicatedmastercount",
          "Required": false,
          "Type": "Integer",
          "UpdateType": "Mutable"
        },
        "DedicatedMasterEnabled": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-properties-elasticsearch-domain-elasticsearchclusterconfig.html#cfn-elasticsearch-
domain-elasticseachclusterconfig-dedicatedmasterenabled",
          "Required": false,
          "Type": "Boolean",
          "UpdateType": "Mutable"
        },
        "DedicatedMasterType": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-properties-elasticsearch-domain-elasticsearchclusterconfig.html#cfn-elasticsearch-
domain-elasticseachclusterconfig-dedicatedmastertype",
          "Required": false,
          "Type": "String",
          "UpdateType": "Mutable"
        },
        "InstanceCount": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-properties-elasticsearch-domain-elasticsearchclusterconfig.html#cfn-elasticsearch-
domain-elasticseachclusterconfig-instancecount",
          "Required": false,
          "Type": "Integer",
          "UpdateType": "Mutable"
        },
        "InstanceType": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-properties-elasticsearch-domain-elasticsearchclusterconfig.html#cfn-elasticsearch-
domain-elasticseachclusterconfig-instnacetype",
          "Required": false,
          "Type": "String",
          "UpdateType": "Mutable"
        },
        "ZoneAwarenessEnabled": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-properties-elasticsearch-domain-elasticsearchclusterconfig.html#cfn-elasticsearch-
domain-elasticseachclusterconfig-zoneawarenessenabled",
          "Required": false,
          "Type": "Boolean",
          "UpdateType": "Mutable"
        }
      }
    },
    "AWS::Elasticsearch::Domain.SnapshotOptions": {
      "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-
properties-elasticsearch-domain-snapshotoptions.html",
      "Properties": {
        "AutomatedSnapshotStartHour": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-properties-elasticsearch-domain-snapshotoptions.html#cfn-elasticsearch-domain-
snapshotoptions-automatedsnapshotstarthour",
          "Required": false,
          "Type": "Integer",
          "UpdateType": "Mutable"
        }
```

```
      }
    },
    "Tag": {
      "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-
properties-resource-tags.html",
      "Properties": {
        "Key": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-properties-resource-tags.html#cfn-resource-tags-key",
          "Required": true,
          "UpdateType": "Immutable"
        },
        "Value": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-properties-resource-tags.html#cfn-resource-tags-value",
          "Required": true,
          "UpdateType": "Immutable"
        }
      }
    }
  },
  "ResourceSpecificationVersion": "1.0.0",
  "ResourceType": {
    "AWS::Elasticsearch::Domain": {
      "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-
resource-elasticsearch-domain.html",
      "Properties": {
        "AccessPolicies": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-resource-elasticsearch-domain.html#cfn-elasticsearch-domain-accesspolicies",
          "Required": false,
          "Type": "Json",
          "UpdateType": "Mutable"
        },
        "AdvancedOptions": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-resource-elasticsearch-domain.html#cfn-elasticsearch-domain-advancedoptions",
          "DuplicatesAllowed": false,
          "ItemType": "String",
          "Required": false,
          "Type": "Map",
          "UpdateType": "Mutable"
        },
        "DomainName": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-resource-elasticsearch-domain.html#cfn-elasticsearch-domain-domainname",
          "Required": false,
          "Type": "String",
          "UpdateType": "Immutable"
        },
        "EBSOptions": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-resource-elasticsearch-domain.html#cfn-elasticsearch-domain-ebsoptions",
          "Required": false,
          "Type": "EBSOptions",
          "UpdateType": "Mutable"
        },
        "ElasticsearchClusterConfig": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/
UserGuide/aws-resource-elasticsearch-domain.html#cfn-elasticsearch-domain-
elasticsearchclusterconfig",
          "Required": false,
          "Type": "ElasticsearchClusterConfig",
          "UpdateType": "Mutable"
        },
        "ElasticsearchVersion": {
```

```
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-resource-elasticsearch-domain.html#cfn-elasticsearch-domain-elasticsearchversion",
          "Required": false,
          "Type": "String",
          "UpdateType": "Immutable"
        },
        "SnapshotOptions": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-resource-elasticsearch-domain.html#cfn-elasticsearch-domain-snapshotoptions",
          "Required": false,
          "Type": "SnapshotOptions",
          "UpdateType": "Mutable"
        },
        "Tags": {
          "Documentation": "http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-resource-elasticsearch-domain.html#cfn-elasticsearch-domain-tags",
          "DuplicatesAllowed": true,
          "ItemType": "Tag",
          "Required": false,
          "Type": "List",
          "UpdateType": "Mutable"
        }
      }
    }
  }
}
```

Resource Attribute Reference

This section details the attributes that you can add to a resource to control additional behaviors and relationships.

Topics

CreationPolicy Attribute

Associate the `CreationPolicy` attribute with a resource to prevent its status from reaching create complete until AWS CloudFormation receives a specified number of success signals or the timeout period is exceeded. To signal a resource, you can use the cfn-signal (p. 1358) helper script or `SignalResource` API. AWS CloudFormation publishes valid signals to the stack events so that you track the number of signals sent.

The creation policy is invoked only when AWS CloudFormation creates the associated resource. Currently, the only AWS CloudFormation resources that support creation policies are AWS::AutoScaling::AutoScalingGroup (p. 433), AWS::EC2::Instance (p. 574), and AWS::CloudFormation::WaitCondition (p. 487).

Use the `CreationPolicy` attribute when you want to wait on resource configuration actions before stack creation proceeds. For example, if you install and configure software applications on an EC2 instance, you might want those applications to be running before proceeding. In such cases, you can add a `CreationPolicy` attribute to the instance, and then send a success signal to the instance after the applications are installed and configured. For a detailed example, see Deploying Applications on Amazon EC2 with AWS CloudFormation (p. 212).

Syntax

JSON

```
"CreationPolicy" : {
  "AutoScalingCreationPolicy" : {
    "MinSuccessfulInstancesPercent" : Integer
  },
  "ResourceSignal" : {
    "Count" : Integer,
    "Timeout" : String
  }
}
```

YAML

```
CreationPolicy:
  AutoScalingCreationPolicy:
    MinSuccessfulInstancesPercent: Integer
  ResourceSignal:
    Count: Integer
    Timeout: String
```

CreationPolicy Properties

`AutoScalingCreationPolicy`

For an Auto Scaling group replacement update (p. 1304), specifies how many instances must signal success for the update to succeed.

`MinSuccessfulInstancesPercent`

Specifies the percentage of instances in an Auto Scaling replacement update that must signal success for the update to succeed. You can specify a value from `0` to `100`. AWS CloudFormation rounds to the nearest tenth of a percent. For example, if you update five instances with a minimum successful percentage of `50`, three instances must signal success. If an instance doesn't send a signal within the time specified by the `Timeout` property, AWS CloudFormation assumes that the instance wasn't created.

Default: `100`

Type: Integer

Required: No

`ResourceSignal`

When AWS CloudFormation creates the associated resource, configures the number of required success signals and the length of time that AWS CloudFormation waits for those signals.

`Count`

The number of success signals AWS CloudFormation must receive before it sets the resource status as `CREATE_COMPLETE`. If the resource receives a failure signal or doesn't receive the specified number of signals before the timeout period expires, the resource creation fails and AWS CloudFormation rolls the stack back.

Default: `1`

Type: Integer

Required: No

`Timeout`

The length of time that AWS CloudFormation waits for the number of signals that was specified in the `Count` property. The timeout period starts after AWS CloudFormation starts creating the resource, and the timeout expires no sooner than the time you specify but can occur shortly thereafter. The maximum time that you can specify is 12 hours.

The value must be in ISO8601 duration format, in the form: "PT#H#M#S", where each # is the number of hours, minutes, and seconds, respectively. For best results, specify a period of time that gives your instances plenty of time to get up and running. A shorter timeout can cause a rollback.

Default: `PT5M` (5 minutes)

Type: String

Required: No

Examples

Auto Scaling Group

The following example shows how to add a creation policy to an Auto Scaling group. The creation policy requires three success signals and times out after 15 minutes.

To have instances wait for an Elastic Load Balancing health check before they signal success, add a health-check verification by using the cfn-init helper script. For an example, see the `verify_instance_health` command in the Auto Scaling rolling updates sample template.

JSON

```
"AutoScalingGroup": {
  "Type": "AWS::AutoScaling::AutoScalingGroup",
  "Properties": {
    "AvailabilityZones": { "Fn::GetAZs": "" },
    "LaunchConfigurationName": { "Ref": "LaunchConfig" },
    "DesiredCapacity": "3",
    "MinSize": "1",
    "MaxSize": "4"
  },
  "CreationPolicy": {
    "ResourceSignal": {
      "Count": "3",
      "Timeout": "PT15M"
    }
  },
  "UpdatePolicy" : {
    "AutoScalingScheduledAction" : {
      "IgnoreUnmodifiedGroupSizeProperties" : "true"
    },
    "AutoScalingRollingUpdate" : {
      "MinInstancesInService" : "1",
      "MaxBatchSize" : "2",
      "PauseTime" : "PT1M",
      "WaitOnResourceSignals" : "true"
    }
  }
},
```

```
"LaunchConfig": {
  "Type": "AWS::AutoScaling::LaunchConfiguration",
  "Properties": {
    "ImageId": "ami-16d18a7e",
    "InstanceType": "t2.micro",
    "UserData": {
      "Fn::Base64": {
        "Fn::Join" : [ "", [
          "#!/bin/bash -xe\n",
          "yum install -y aws-cfn-bootstrap\n",
          "/opt/aws/bin/cfn-signal -e 0 --stack ", { "Ref": "AWS::StackName" },
          " --resource AutoScalingGroup ",
          " --region ", { "Ref" : "AWS::Region" }, "\n"
        ] ]
      }
    }
  }
}
```

YAML

```
AutoScalingGroup:
  Type: AWS::AutoScaling::AutoScalingGroup
  Properties:
    AvailabilityZones:
      Fn::GetAZs: ''
    LaunchConfigurationName:
      Ref: LaunchConfig
    DesiredCapacity: '3'
    MinSize: '1'
    MaxSize: '4'
  CreationPolicy:
    ResourceSignal:
      Count: '3'
      Timeout: PT15M
  UpdatePolicy:
    AutoScalingScheduledAction:
      IgnoreUnmodifiedGroupSizeProperties: 'true'
    AutoScalingRollingUpdate:
      MinInstancesInService: '1'
      MaxBatchSize: '2'
      PauseTime: PT1M
      WaitOnResourceSignals: 'true'
LaunchConfig:
  Type: AWS::AutoScaling::LaunchConfiguration
  Properties:
    ImageId: ami-16d18a7e
    InstanceType: t2.micro
    UserData:
      "Fn::Base64":
        !Sub |
          #!/bin/bash -xe
          yum update -y aws-cfn-bootstrap
          /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --resource
 AutoScalingGroup --region ${AWS::Region}
```

WaitCondition

The following example shows how to add a creation policy to a wait condition.

JSON

```
"WaitCondition" : {
```

```
    "Type" : "AWS::CloudFormation::WaitCondition",
    "CreationPolicy" : {
        "ResourceSignal" : {
            "Timeout" : "PT15M",
            "Count" : "5"
        }
    }
}
```

YAML

```
WaitCondition:
  Type: AWS::CloudFormation::WaitCondition
  CreationPolicy:
    ResourceSignal:
      Timeout: PT15M
      Count: 5
```

DeletionPolicy Attribute

With the DeletionPolicy attribute you can preserve or (in some cases) backup a resource when its stack is deleted. You specify a DeletionPolicy attribute for each resource that you want to control. If a resource has no DeletionPolicy attribute, AWS CloudFormation deletes the resource by default.

To keep a resource when its stack is deleted, specify `Retain` for that resource. You can use retain for any resource. For example, you can retain a nested stack, S3 bucket, or EC2 instance so that you can continue to use or modify those resources after you delete their stacks.

> **Note**
> If you want to modify resources outside of AWS CloudFormation, use a retain policy and then delete the stack. Otherwise, your resources might get out of sync with your AWS CloudFormation template and cause stack errors.

For resources that support snapshots, such as `AWS::RDS::DBInstance` and `AWS::EC2::Volume`, you can specify `Snapshot` to have AWS CloudFormation create a snapshot before deleting the resource.

The following snippet contains an Amazon S3 bucket resource with a `Retain` deletion policy. When this stack is deleted, AWS CloudFormation leaves the bucket without deleting it.

JSON

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Resources" : {
    "myS3Bucket" : {
      "Type" : "AWS::S3::Bucket",
      "DeletionPolicy" : "Retain"
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  myS3Bucket:
    Type: AWS::S3::Bucket
```

```
    DeletionPolicy: Retain
```

DeletionPolicy Options

Delete
: AWS CloudFormation deletes the resource and all its content if applicable during stack deletion. You can add this deletion policy to any resource type. By default, if you don't specify a DeletionPolicy, AWS CloudFormation deletes your resources.

> **Note**
> For Amazon S3 buckets, you must delete all objects in the bucket for deletion to succeed.

Retain
: AWS CloudFormation keeps the resource without deleting the resource or its contents when its stack is deleted. You can add this deletion policy to any resource type. Note that when AWS CloudFormation completes the stack deletion, the stack will be in `Delete_Complete` state; however, resources that are retained continue to exist and continue to incur applicable charges until you delete those resources.

Snapshot
: For resources that support snapshots (`AWS::EC2::Volume`, `AWS::ElastiCache::CacheCluster`, `AWS::ElastiCache::ReplicationGroup`, `AWS::RDS::DBInstance`, `AWS::RDS::DBCluster`, and `AWS::Redshift::Cluster`), AWS CloudFormation creates a snapshot for the resource before deleting it. Note that when AWS CloudFormation completes the stack deletion, the stack will be in the `Delete_Complete` state; however, the snapshots that are created with this policy continue to exist and continue to incur applicable charges until you delete those snapshots.

DependsOn Attribute

With the `DependsOn` attribute you can specify that the creation of a specific resource follows another. When you add a `DependsOn` attribute to a resource, that resource is created only after the creation of the resource specified in the `DependsOn` attribute.

> **Important**
> Dependent stacks also have implicit dependencies. For example, if the properties of resource A use a `!Ref` to resource B, the following rule apply:
>
> - Resource A is created before resource B.
> - Resource B is deleted before resource A.

You can use the `DependsOn` attribute with any resource. Here are some typical uses:

- Determine when a wait condition goes into effect. For more information, see Creating Wait Conditions in a Template (p. 228).
- Declare dependencies for resources that must be created or deleted in a specific order. For example, you must explicitly declare dependencies on gateway attachments for some resources in a VPC. For more information, see When a DependsOn attribute is required (p. 1300).
- Override default parallelism when creating, updating, or deleting resources. AWS CloudFormation creates, updates, and deletes resources in parallel to the extent possible. It automatically determines which resources in a template can be parallelized and which have dependencies that require other operations to finish first. You can use `DependsOn` to explicitly specify dependencies, which overrides the default parallelism and directs CloudFormation to operate on those resources in a specified order.

> **Note**
> During a stack update, resources that depend on updated resources are updated automatically. AWS CloudFormation makes no changes to the automatically-updated resources, but, if a stack policy is associated with these resources, your account must have the permissionsto update them.

Syntax

The `DependsOn` attribute can take a single string or list of strings.

```
"DependsOn" : [ String, ... ]
```

Example

The following template contains an AWS::EC2::Instance (p. 574) resource with a `DependsOn` attribute that specifies myDB, an AWS::RDS::DBInstance (p. 881). When AWS CloudFormation creates this stack, it first creates myDB, then creates Ec2Instance.

JSON

```
{
    "AWSTemplateFormatVersion" : "2010-09-09",
    "Mappings" : {
        "RegionMap" : {
            "us-east-1" : { "AMI" : "ami-76f0061f" },
            "us-west-1" : { "AMI" : "ami-655a0a20" },
            "eu-west-1" : { "AMI" : "ami-7fd4e10b" },
            "ap-northeast-1" : { "AMI" : "ami-8e08a38f" },
            "ap-southeast-1" : { "AMI" : "ami-72621c20" }
        }
    },
    "Resources" : {
        "Ec2Instance" : {
            "Type" : "AWS::EC2::Instance",
            "Properties" : {
                "ImageId" : {
                   "Fn::FindInMap" : [ "RegionMap", { "Ref" : "AWS::Region" }, "AMI" ]
                }
            },
            "DependsOn" : "myDB"
        },
        "myDB" : {
            "Type" : "AWS::RDS::DBInstance",
            "Properties" : {
               "AllocatedStorage" : "5",
               "DBInstanceClass" : "db.m1.small",
               "Engine" : "MySQL",
               "EngineVersion" : "5.5",
               "MasterUsername" : "MyName",
               "MasterUserPassword" : "MyPassword"
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Mappings:
  RegionMap:
    us-east-1:
      AMI: ami-76f0061f
    us-west-1:
      AMI: ami-655a0a20
    eu-west-1:
```

```
      AMI: ami-7fd4e10b
    ap-northeast-1:
      AMI: ami-8e08a38f
    ap-southeast-1:
      AMI: ami-72621c20
Resources:
  Ec2Instance:
    Type: AWS::EC2::Instance
    Properties:
      ImageId:
        Fn::FindInMap:
        - RegionMap
        - Ref: AWS::Region
        - AMI
    DependsOn: myDB
  myDB:
    Type: AWS::RDS::DBInstance
    Properties:
      AllocatedStorage: '5'
      DBInstanceClass: db.m1.small
      Engine: MySQL
      EngineVersion: '5.5'
      MasterUsername: MyName
      MasterUserPassword: MyPassword
```

When a DependsOn attribute is required

VPC-gateway attachment

Some resources in a VPC require a gateway (either an Internet or VPN gateway). If your AWS CloudFormation template defines a VPC, a gateway, and a gateway attachment, any resources that require the gateway are dependent on the gateway attachment. For example, an Amazon EC2 instance with a public IP address is dependent on the VPC-gateway attachment if the `VPC` and `InternetGateway` resources are also declared in the same template.

Currently, the following resources depend on a VPC-gateway attachment when they have an associated public IP address and are in a VPC:

- Auto Scaling groups
- Amazon EC2 instances
- Elastic Load Balancing load balancers
- Elastic IP addresses
- Amazon RDS database instances
- Amazon VPC routes that include the Internet gateway

A VPN gateway route propagation depends on a VPC-gateway attachment when you have a VPN gateway

The following snippet shows a sample gateway attachment and an Amazon EC2 instance that depends on a gateway attachment:

JSON

```
"GatewayToInternet" : {
  "Type" : "AWS::EC2::VPCGatewayAttachment",
  "Properties" : {
    "VpcId" : { "Ref" : "VPC" },
    "InternetGatewayId" : { "Ref" : "InternetGateway" }
  }
```

```
},

"EC2Host" : {
  "Type" : "AWS::EC2::Instance",
  "DependsOn" : "GatewayToInternet",
    "Properties" : {
      "InstanceType" : { "Ref" : "EC2InstanceType" },
      "KeyName"  : { "Ref" : "KeyName" },
      "ImageId"  : { "Fn::FindInMap" : [ "AWSRegionArch2AMI", { "Ref" : "AWS::Region" },
        { "Fn::FindInMap" : [ "AWSInstanceType2Arch", { "Ref" : "EC2InstanceType" },
 "Arch" ] } ] },
      "NetworkInterfaces" : [{
        "GroupSet"                 : [{ "Ref" : "EC2SecurityGroup" }],
        "AssociatePublicIpAddress" : "true",
        "DeviceIndex"              : "0",
        "DeleteOnTermination"      : "true",
        "SubnetId"                 : { "Ref" : "PublicSubnet" }
      }]
    }
}
```

YAML

```
GatewayToInternet:
  Type: AWS::EC2::VPCGatewayAttachment
  Properties:
    VpcId:
      Ref: VPC
    InternetGatewayId:
      Ref: InternetGateway
EC2Host:
  Type: AWS::EC2::Instance
  DependsOn: GatewayToInternet
  Properties:
    InstanceType:
      Ref: EC2InstanceType
    KeyName:
      Ref: KeyName
    ImageId:
      Fn::FindInMap:
      - AWSRegionArch2AMI
      - Ref: AWS::Region
      - Fn::FindInMap:
        - AWSInstanceType2Arch
        - Ref: EC2InstanceType
        - Arch
    NetworkInterfaces:
    - GroupSet:
      - Ref: EC2SecurityGroup
      AssociatePublicIpAddress: 'true'
      DeviceIndex: '0'
      DeleteOnTermination: 'true'
      SubnetId:
        Ref: PublicSubnet
```

Amazon ECS Service and Auto Scaling Group

When you use Auto Scaling or Amazon Elastic Compute Cloud (Amazon EC2) to create container instances for an Amazon ECS cluster, the Amazon ECS service resource must have a dependency on the Auto Scaling group or Amazon EC2 instances, as shown in the following snippet. That way the container instances are available and associated with the Amazon ECS cluster before AWS CloudFormation creates the Amazon ECS service.

JSON

```
"service": {
  "Type": "AWS::ECS::Service",
  "DependsOn": ["ECSAutoScalingGroup"],
  "Properties" : {
    "Cluster": {"Ref": "ECSCluster"},
    "DesiredCount": "1",
    "LoadBalancers": [
      {
        "ContainerName": "simple-app",
        "ContainerPort": "80",
        "LoadBalancerName" : { "Ref" : "EcsElasticLoadBalancer" }
      }
    ],
    "Role" : {"Ref":"ECSServiceRole"},
    "TaskDefinition" : {"Ref":"taskdefinition"}
  }
}
```

YAML

```
service:
  Type: AWS::ECS::Service
  DependsOn:
  - ECSAutoScalingGroup
  Properties:
    Cluster:
      Ref: ECSCluster
    DesiredCount: 1
    LoadBalancers:
    - ContainerName: simple-app
      ContainerPort: 80
      LoadBalancerName:
        Ref: EcsElasticLoadBalancer
    Role:
      Ref: ECSServiceRole
    TaskDefinition:
      Ref: taskdefinition
```

IAM Role Policy

Resources that make additional calls to AWS require a service role, which permits a service to make calls to AWS on your behalf. For example, the `AWS::CodeDeploy::DeploymentGroup` resource requires a service role so that AWS CodeDeploy has permissions to deploy applications to your instances. When you have a single template that defines a service role, the role's policy (by using the `AWS::IAM::Policy` or `AWS::IAM::ManagedPolicy` resource), and a resource that uses the role, add a dependency so that the resource depends on the role's policy. This dependency ensures that the policy is available throughout the resource's lifecycle.

For example, imagine that you have a template with a deployment group resource, a service role, and the role's policy. When you create a stack, AWS CloudFormation won't create the deployment group until it creates the role's policy. Without the dependency, AWS CloudFormation can create the deployment group resource before it creates the role's policy. If that happens, the deployment group will fail to create because of insufficient permissions.

If the role has an embedded policy, don't specify a dependency. AWS CloudFormation creates the role and its policy at the same time.

Metadata Attribute

The Metadata attribute enables you to associate structured data with a resource. By adding a Metadata attribute to a resource, you can add data in JSON or YAML to the resource declaration. In addition, you can use intrinsic functions (such as GetAtt (p. 1324) and Ref (p. 1343)), parameters, and pseudo parameters within the Metadata attribute to add those interpreted values.

> **Note**
> AWS CloudFormation does not validate the syntax within the Metadata attribute.

You can retrieve this data using the AWS command `aws cloudformation describe-stack-resource` or the DescribeStackResource action.

Example

The following template contains an Amazon S3 bucket resource with a Metadata attribute.

JSON

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "MyS3Bucket" : {
         "Type" : "AWS::S3::Bucket",
         "Metadata" : { "Object1" : "Location1",  "Object2" : "Location2" }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  MyS3Bucket:
    Type: AWS::S3::Bucket
    Metadata:
      Object1: Location1
      Object2: Location2
```

UpdatePolicy Attribute

Use the `UpdatePolicy` attribute to specify how AWS CloudFormation handles updates to the `AWS::AutoScaling::AutoScalingGroup` resource. AWS CloudFormation invokes one of three update policies depending on the type of change you make or whether a scheduled action is associated with the Auto Scaling group.

- The `AutoScalingReplacingUpdate` and `AutoScalingRollingUpdate` policies apply *only* when you do one or more of the following:
  - Change the Auto Scaling group's `AWS::AutoScaling::LaunchConfiguration`.
  - Change the Auto Scaling group's `VPCZoneIdentifier` property
  - Update an Auto Scaling group that contains instances that don't match the current `LaunchConfiguration`.

  If both the `AutoScalingReplacingUpdate` and `AutoScalingRollingUpdate` policies are specified, setting the `WillReplace` property to `true` gives `AutoScalingReplacingUpdate` precedence.

- The `AutoScalingScheduledAction` policy applies when you update a stack that includes an Auto Scaling group with an associated scheduled action.

AutoScalingReplacingUpdate Policy

To specify how AWS CloudFormation handles replacement updates for an Auto Scaling group, use the `AutoScalingReplacingUpdate` policy. This policy enables you to specify whether AWS CloudFormation replaces an Auto Scaling group with a new one or replaces only the instances in the Auto Scaling group.

> **Important**
> Before attempting an update, ensure that you have sufficient Amazon EC2 capacity for both your old and new Auto Scaling groups.

Syntax

JSON

```
"UpdatePolicy" : {
  "AutoScalingReplacingUpdate (p. 1304)" : {
    "WillReplace" : Boolean
  }
}
```

YAML

```
UpdatePolicy:
  AutoScalingReplacingUpdate (p. 1304):
    WillReplace: Boolean
```

Properties

`WillReplace`

Specifies whether an Auto Scaling group and the instances it contains are replaced during an update. During replacement, AWS CloudFormation retains the old group until it finishes creating the new one. If the update fails, AWS CloudFormation can roll back to the old Auto Scaling group and delete the new Auto Scaling group.

While AWS CloudFormation creates the new group, it doesn't detach or attach any instances. After successfully creating the new Auto Scaling group, AWS CloudFormation deletes the old Auto Scaling group during the cleanup process.

When you set the `WillReplace` parameter, remember to specify a matching `CreationPolicy`. If the minimum number of instances (specified by the `MinSuccessfulInstancesPercent` property) don't signal success within the `Timeout` period (specified in the `CreationPolicy` policy), the replacement update fails and AWS CloudFormation rolls back to the old Auto Scaling group.

Type: Boolean

Required: No

AutoScalingRollingUpdate Policy

To specify how AWS CloudFormation handles rolling updates for an Auto Scaling group, use the `AutoScalingRollingUpdate` policy. Rolling updates enable you to specify whether AWS CloudFormation updates instances that are in an Auto Scaling group in batches or all at once.

Important
During a rolling update, some Auto Scaling processes might make changes to the Auto Scaling group before AWS CloudFormation completes the rolling update. These changes might cause the rolling update to fail. To prevent Auto Scaling from running processes during a rolling update, use the `SuspendProcesses` property. For more information, see What are some recommended best practices for performing Auto Scaling group rolling updates?

Syntax

JSON

```
"UpdatePolicy" : {
  "AutoScalingRollingUpdate (p. 1304)" : {
    "MaxBatchSize" : Integer,
    "MinInstancesInService" : Integer,
    "MinSuccessfulInstancesPercent" : Integer
    "PauseTime" : String,
    "SuspendProcesses" : [ List of processes ],
    "WaitOnResourceSignals" : Boolean
  }
}
```

YAML

```
UpdatePolicy:
  AutoScalingRollingUpdate (p. 1304):
    MaxBatchSize: Integer
    MinInstancesInService: Integer
    MinSuccessfulInstancesPercent: Integer
    PauseTime: String
    SuspendProcesses:
      - List of processes
    WaitOnResourceSignals: Boolean
```

Properties

`MaxBatchSize`

Specifies the maximum number of instances that AWS CloudFormation updates.

Default: `1`

Type: Integer

Required: No

`MinInstancesInService`

Specifies the minimum number of instances that must be in service within the Auto Scaling group while AWS CloudFormation updates old instances.

Default: `0`

Type: Integer

Required: No

`MinSuccessfulInstancesPercent`

Specifies the percentage of instances in an Auto Scaling rolling update that must signal success for an update to succeed. You can specify a value from `0` to `100`. AWS CloudFormation rounds to the nearest tenth of a percent. For example, if you update five instances with a minimum successful percentage of `50`, three instances must signal success.

If an instance doesn't send a signal within the time specified in the `PauseTime` property, AWS CloudFormation assumes that the instance wasn't updated.

If you specify this property, you must also enable the `WaitOnResourceSignals` and `PauseTime` properties.

Default: `100`

Type: Integer

Required: No

`PauseTime`

The amount of time that AWS CloudFormation pauses after making a change to a batch of instances to give those instances time to start software applications. For example, you might need to specify `PauseTime` when scaling up the number of instances in an Auto Scaling group.

If you enable the `WaitOnResourceSignals` property, `PauseTime` is the amount of time that AWS CloudFormation should wait for the Auto Scaling group to receive the required number of valid signals from added or replaced instances. If the `PauseTime` is exceeded before the Auto Scaling group receives the required number of signals, the update fails. For best results, specify a time period that gives your applications sufficient time to get started. If the update needs to be rolled back, a short `PauseTime` can cause the rollback to fail.

Specify `PauseTime` in the ISO8601 duration format (in the format `PT#H#M#S`, where each `#` is the number of hours, minutes, and seconds, respectively). The maximum `PauseTime` is one hour (`PT1H`).

Default: `PT0S` (zero seconds). If the `WaitOnResourceSignals` property is set to `true`, the default is `PT5M`.

Type: String

Required: No

`SuspendProcesses`

Specifies the Auto Scaling processes to suspend during a stack update. Suspending processes prevents Auto Scaling from interfering with a stack update. For example, you can suspend alarming so that Auto Scaling doesn't execute scaling policies associated with an alarm. For valid values, see the `ScalingProcesses.member.N` parameter for the SuspendProcesses action in the *Auto Scaling API Reference*.

Default: Not specified

Type: List of Auto Scaling processes

Required: No

`WaitOnResourceSignals`

Specifies whether the Auto Scaling group waits on signals from new instances during an update. Use this property to ensure that instances have completed installing and configuring applications before the Auto Scaling group update proceeds. AWS CloudFormation suspends the update of an Auto Scaling group after new EC2 instances are launched into the group. AWS CloudFormation must receive a signal from each new instance within the specified `PauseTime` before continuing the update. To signal the Auto Scaling group, use the cfn-signal helper script or `SignalResource` API.

To have instances wait for an Elastic Load Balancing health check before they signal success, add a health-check verification by using the cfn-init helper script. For an example, see the `verify_instance_health` command in the Auto Scaling rolling updates sample template.

Default: `false`

Type: Boolean

Required: Conditional. If you specify the `MinSuccessfulInstancesPercent` property, you must also enable the `WaitOnResourceSignals` and `PauseTime` properties.

AutoScalingScheduledAction Policy

To specify how AWS CloudFormation handles updates for the `MinSize`, `MaxSize`, and `DesiredCapacity` properties when the `AWS::AutoScaling::AutoScalingGroup` resource has an associated scheduled action, use the `AutoScalingScheduledAction` policy.

With scheduled actions, the group size properties of an Auto Scaling group can change at any time. When you update a stack with an Auto Scaling group and scheduled action, AWS CloudFormation always sets the group size property values of your Auto Scaling group to the values that are defined in the `AWS::AutoScaling::AutoScalingGroup` resource of your template, even if a scheduled action is in effect.

If you do not want AWS CloudFormation to change any of the group size property values when you have a scheduled action in effect, use the `AutoScalingScheduledAction` update policy to prevent AWS CloudFormation from changing the `MinSize`, `MaxSize`, or `DesiredCapacity` properties unless you have modified these values in your template.

Syntax

JSON

```
"UpdatePolicy" : {
  "AutoScalingScheduledAction (p. 1307)" : {
    "IgnoreUnmodifiedGroupSizeProperties" : Boolean
  }
}
```

YAML

```
UpdatePolicy:
  AutoScalingScheduledAction (p. 1307):
    IgnoreUnmodifiedGroupSizeProperties: Boolean
```

Properties

`IgnoreUnmodifiedGroupSizeProperties`

Specifies whether AWS CloudFormation ignores differences in group size properties between your current Auto Scaling group and the Auto Scaling group described in the `AWS::AutoScaling::AutoScalingGroup` resource of your template during a stack update. If you modify any of the group size property values in your template, AWS CloudFormation uses the modified values and updates your Auto Scaling group.

Default: `false`

Type: Boolean

Required: No

Examples

The following examples show how to add an update policy to an Auto Scaling group and how to maintain availability when updating metadata.

Add an UpdatePolicy to an Auto Scaling Group

The following example shows how to add an update policy. During an update, the Auto Scaling group updates instances in batches of two and keeps a minimum of one instance in service. Because the `WaitOnResourceSignals` flag is set, the Auto Scaling group waits for new instances that are added to the group. The new instances must signal the Auto Scaling group before it updates the next batch of instances.

JSON

```
"ASG" : {
   "Type" : "AWS::AutoScaling::AutoScalingGroup",
   "Properties" : {
      "AvailabilityZones" : [
         "us-east-1a",
         "us-east-1b"
      ],
      "DesiredCapacity" : "1",
      "LaunchConfigurationName" : {
         "Ref" : "LaunchConfig"
      },
      "MaxSize" : "4",
      "MinSize" : "1"
   },
   "UpdatePolicy" : {
      "AutoScalingScheduledAction" : {
         "IgnoreUnmodifiedGroupSizeProperties" : "true"
      },
      "AutoScalingRollingUpdate" : {
         "MinInstancesInService" : "1",
         "MaxBatchSize" : "2",
         "WaitOnResourceSignals" : "true",
         "PauseTime" : "PT10M"
      }
   }
 },
"ScheduledAction" : {
   "Type" : "AWS::AutoScaling::ScheduledAction",
   "Properties" : {
      "AutoScalingGroupName" : {
         "Ref" : "ASG"
      },
      "DesiredCapacity" : "2",
      "StartTime" : "2017-06-02T20 : 00 : 00Z"
   }
}
```

YAML

```
ASG:
  Type: 'AWS::AutoScaling::AutoScalingGroup'
  Properties:
    AvailabilityZones:
    - us-east-1a
    - us-east-1b
    DesiredCapacity: '1'
    LaunchConfigurationName:
      Ref: LaunchConfig
    MaxSize: '4'
    MinSize: '1'
  UpdatePolicy:
    AutoScalingScheduledAction:
      IgnoreUnmodifiedGroupSizeProperties: 'true'
    AutoScalingRollingUpdate:
```

```
      MinInstancesInService: '1'
      MaxBatchSize: '2'
      WaitOnResourceSignals: 'true'
      PauseTime: PT10M
ScheduledAction:
  Type: 'AWS::AutoScaling::ScheduledAction'
  Properties:
    AutoScalingGroupName:
      Ref: ASG
    DesiredCapacity: '2'
    StartTime: '2017-06-02T20 : 00 : 00Z'
```

AutoScalingReplacingUpdate Policy

The following example declares a policy that forces an associated Auto Scaling group to be replaced during an update. For the update to succeed, a percentage of instances (specified by the `MinSuccessfulPercentParameter` parameter) must signal success within the `Timeout` period.

JSON

```
"UpdatePolicy" : {
  "AutoScalingReplacingUpdate" : {
    "WillReplace" : "true"
  }
},
"CreationPolicy" : {
  "ResourceSignal" : {
    "Count" : { "Ref" : "ResourceSignalsOnCreate"},
    "Timeout" : "PT10M"
  },
  "AutoScalingCreationPolicy" : {
    "MinSuccessfulInstancesPercent" : { "Ref" : "MinSuccessfulPercentParameter" }
  }
}
```

YAML

```
UpdatePolicy:
  AutoScalingReplacingUpdate:
    WillReplace: 'true'
CreationPolicy:
  ResourceSignal:
    Count: !Ref 'ResourceSignalsOnCreate'
    Timeout: PT10M
  AutoScalingCreationPolicy:
    MinSuccessfulInstancesPercent: !Ref 'MinSuccessfulPercentParameter'
```

Maintain Availability When Updating the Metadata for the cfn-init Helper Script

When you install software applications on your instances, you might use the `AWS::CloudFormation::Init` metadata key and the `cfn-init` helper script to bootstrap the instances in your Auto Scaling group. AWS CloudFormation installs the packages, runs the commands, and performs other bootstrapping actions described in the metadata.

When you update only the metadata (for example, when updating a package to another version), you can use the `cfn-hup` helper daemon to detect and apply the updates. However, the `cfn-hup` daemon runs independently on each instance. If the daemon happens to runs at the same time on all instances, your application or service might be unavailable during the update. To guarantee availability, you can force a rolling update so that AWS CloudFormation updates your instances one batch at a time.

Important

Forcing a rolling update requires AWS CloudFormation to create a new instance and then delete the old one. Any information stored on the old instance is lost.

To force a rolling update, change the logical ID of the launch configuration resource, and then update the stack and any references pointing to the original logic ID (such as the associated Auto Scaling group). AWS CloudFormation triggers a rolling update on the Auto Scaling group, replacing all instances.

Original Template

```
"LaunchConfig": {
  "Type" : "AWS::AutoScaling::LaunchConfiguration",
  "Metadata" : {
    "Comment" : "Install a simple PHP application",
    "AWS::CloudFormation::Init" : {
    ...
    }
  }
}
```

Updated Logical ID

```
"LaunchConfigUpdateRubygemsPkg": {
  "Type" : "AWS::AutoScaling::LaunchConfiguration",
  "Metadata" : {
    "Comment" : "Install a simple PHP application",
    "AWS::CloudFormation::Init" : {
    ...
    }
  }
}
```

Intrinsic Function Reference

AWS CloudFormation provides several built-in functions that help you manage your stacks. Use intrinsic functions in your templates to assign values to properties that are not available until runtime.

Note

You can use intrinsic functions only in specific parts of a template. Currently, you can use intrinsic functions in resource properties, outputs, metadata attributes, and update policy attributes.

Topics

`Fn::Base64`

The intrinsic function `Fn::Base64` returns the Base64 representation of the input string. This function is typically used to pass encoded data to Amazon EC2 instances by way of the `UserData` property.

Declaration

JSON

```
{ "Fn::Base64" : valueToEncode }
```

YAML

Syntax for the full function name:

```
Fn::Base64: valueToEncode
```

Syntax for the short form:

```
!Base64 valueToEncode
```

Note
If you use the short form and immediately include another function in the `valueToEncode` parameter, use the full function name for at least one of the functions. For example, the following syntax is invalid:

```
!Base64 !Sub string
!Base64 !Ref logical_ID
```

Instead, use the full function name for at least one of the functions, as shown in the following examples:

```
!Base64
  "Fn::Sub": string

Fn::Base64:
  !Sub string
```

Parameters

valueToEncode
    The string value you want to convert to Base64.

Return Value:

The original string, in Base64 representation.

Example

JSON

```
{ "Fn::Base64" : "AWS CloudFormation" }
```

YAML

```
Fn::Base64: AWS CloudFormation
```

Supported Functions

You can use any function that returns a string inside the `Fn::Base64` function.

See Also

- Intrinsic Function Reference (p. 1310)

Condition Functions

You can use intrinsic functions, such as `Fn::If`, `Fn::Equals`, and `Fn::Not`, to conditionally create stack resources. These conditions are evaluated based on input parameters that you declare when you create or update a stack. After you define all your conditions, you can associate them with resources or resource properties in the Resources and Outputs sections of a template.

You define all conditions in the Conditions section of a template except for `Fn::If` conditions. You can use the `Fn::If` condition in the metadata attribute, update policy attribute, and property values in the Resources section and Outputs sections of a template.

You might use conditions when you want to reuse a template that can create resources in different contexts, such as a test environment versus a production environment. In your template, you can add an `EnvironmentType` input parameter, which accepts either **`prod`** or **`test`** as inputs. For the production environment, you might include Amazon EC2 instances with certain capabilities; however, for the test environment, you want to use less capabilities to save costs. With conditions, you can define which resources are created and how they're configured for each environment type.

For more information about the Conditions section, see Conditions (p. 144).

> **Note**
> You can only reference other conditions and values from the Parameters and Mappings sections of a template. For example, you can reference a value from an input parameter, but you cannot reference the logical ID of a resource in a condition.

Topics

Associating a Condition

To conditionally create resources, resource properties, or outputs, you must associate a condition with them. Add the `Condition:` key and the logical ID of the condition as an attribute to associate a condition, as shown in the following snippet. AWS CloudFormation creates the `NewVolume` resource only when the `CreateProdResources` condition evaluates to true.

Example JSON

```
"NewVolume" : {
  "Type" : "AWS::EC2::Volume",
  "Condition" : "CreateProdResources",
  "Properties" : {
     "Size" : "100",
     "AvailabilityZone" : { "Fn::GetAtt" : [ "EC2Instance", "AvailabilityZone" ]}
}
```

Example YAML

```
NewVolume:
  Type: "AWS::EC2::Volume"
  Condition: CreateProdResources
  Properties:
    Size: 100
    AvailabilityZone: !GetAtt EC2Instance.AvailabilityZone
```

For the `Fn::If` function, you only need to specify the condition name. The following snippet shows how to use `Fn::If` to conditionally specify a resource property. If the `CreateLargeSize` condition is true, AWS CloudFormation sets the volume size to `100`. If the condition is false, AWS CloudFormation sets the volume size to `10`.

Example JSON

```
"NewVolume" : {
  "Type" : "AWS::EC2::Volume",
  "Properties" : {
    "Size" : {
      "Fn::If" : [
        "CreateLargeSize",
        "100",
        "10"
      ]},
    "AvailabilityZone" : { "Fn::GetAtt" : [ "Ec2Instance", "AvailabilityZone" ]}
  },
  "DeletionPolicy" : "Snapshot"
}
```

Example YAML

```
NewVolume:
  Type: "AWS::EC2::Volume"
  Properties:
    Size:
      !If [CreateLargeSize, 100, 10]
    AvailabilityZone: !GetAtt: Ec2Instance.AvailabilityZone
  DeletionPolicy: Snapshot
```

You can also use conditions inside other conditions. The following snippet is from the `Conditions` section of a template. The `MyAndCondition` condition includes the `SomeOtherCondition` condition:

Example JSON

```
"MyAndCondition": {
```

```
   "Fn::And": [
      {"Fn::Equals": ["sg-mysggroup", {"Ref": "ASecurityGroup"}]},
      {"Condition": "SomeOtherCondition"}
   ]
}
```

Example YAML

```
MyAndCondition: !And
  - !Equals ["sg-mysggroup", !Ref "ASecurityGroup"]
  - !Condition SomeOtherCondition
```

Fn::And

Returns `true` if all the specified conditions evaluate to true, or returns `false` if any one of the conditions evaluates to false. `Fn::And` acts as an AND operator. The minimum number of conditions that you can include is 2, and the maximum is 10.

Declaration

JSON

```
"Fn::And": [{condition}, {...}]
```

YAML

Syntax for the full function name:

```
Fn::And: [condition]
```

Syntax for the short form:

```
!And [condition]
```

Parameters

`condition`

A condition that evaluates to `true` or `false`.

Example

The following `MyAndCondition` evaluates to true if the referenced security group name is equal to `sg-mysggroup` and if `SomeOtherCondition` evaluates to true:

JSON

```
"MyAndCondition": {
   "Fn::And": [
      {"Fn::Equals": ["sg-mysggroup", {"Ref": "ASecurityGroup"}]},
      {"Condition": "SomeOtherCondition"}
   ]
```

```
}
```

YAML

```
MyAndCondition: !And
  - !Equals ["sg-mysggroup", !Ref ASecurityGroup]
  - !Condition SomeOtherCondition
```

Fn::Equals

Compares if two values are equal. Returns `true` if the two values are equal or `false` if they aren't.

Declaration

JSON

```
"Fn::Equals" : ["value_1", "value_2"]
```

YAML

Syntax for the full function name:

```
Fn::Equals: [value_1, value_2]
```

Syntax for the short form:

```
!Equals [value_1, value_2]
```

Parameters

`value`

A value of any type that you want to compare.

Example

The following `UseProdCondition` condition evaluates to true if the value for the `EnvironmentType` parameter is equal to `prod`:

JSON

```
"UseProdCondition" : {
   "Fn::Equals": [
      {"Ref": "EnvironmentType"},
      "prod"
   ]
}
```

YAML

```
UseProdCondition:
  !Equals [!Ref EnvironmentType, prod]
```

Fn::If

Returns one value if the specified condition evaluates to `true` and another value if the specified condition evaluates to `false`. Currently, AWS CloudFormation supports the `Fn::If` intrinsic function in the metadata attribute, update policy attribute, and property values in the Resources section and Outputs sections of a template. You can use the `AWS::NoValue` pseudo parameter as a return value to remove the corresponding property.

Declaration

JSON

```
"Fn::If": [condition_name, value_if_true, value_if_false]
```

YAML

Syntax for the full function name:

```
Fn::If: [condition_name, value_if_true, value_if_false]
```

Syntax for the short form:

```
!If [condition_name, value_if_true, value_if_false]
```

Parameters

`condition_name`

A reference to a condition in the Conditions section. Use the condition's name to reference it.

`value_if_true`

A value to be returned if the specified condition evaluates to `true`.

`value_if_false`

A value to be returned if the specified condition evaluates to `false`.

Examples

To view additional samples, see Sample Templates (p. 1321).

Example 1

The following snippet uses an `Fn::If` function in the `SecurityGroups` property for an Amazon EC2 resource. If the `CreateNewSecurityGroup` condition evaluates to true, AWS CloudFormation uses the referenced value of `NewSecurityGroup` to specify the `SecurityGroups` property; otherwise, AWS CloudFormation uses the referenced value of `ExistingSecurityGroup`.

JSON

```
"SecurityGroups" : [{
  "Fn::If" : [
    "CreateNewSecurityGroup",
    {"Ref" : "NewSecurityGroup"},
    {"Ref" : "ExistingSecurityGroup"}
  ]
}]
```

YAML

```
SecurityGroups:
  - !If [CreateNewSecurityGroup, !Ref NewSecurityGroup, !Ref ExistingSecurityGroup]
```

Example 2

In the Output section of a template, you can use the `Fn::If` function to conditionally output information. In the following snippet, if the `CreateNewSecurityGroup` condition evaluates to true, AWS CloudFormation outputs the security group ID of the `NewSecurityGroup` resource. If the condition is false, AWS CloudFormation outputs the security group ID of the `ExistingSecurityGroup` resource.

JSON

```
"Outputs" : {
  "SecurityGroupId" : {
    "Description" : "Group ID of the security group used.",
    "Value" : {
      "Fn::If" : [
        "CreateNewSecurityGroup",
        {"Ref" : "NewSecurityGroup"},
        {"Ref" : "ExistingSecurityGroup"}
      ]
    }
  }
}
```

YAML

```
Outputs:
  SecurityGroupId:
    Description: Group ID of the security group used.
    Value: !If [CreateNewSecurityGroup, !Ref NewSecurityGroup, !Ref ExistingSecurityGroup]
```

Example 3

The following snippet uses the `AWS::NoValue` pseudo parameter in an `Fn::If` function. The condition uses a snapshot for an Amazon RDS DB instance only if a snapshot ID is provided. If the `UseDBSnapshot` condition evaluates to true, AWS CloudFormation uses the `DBSnapshotName` parameter value for the `DBSnapshotIdentifier` property. If the condition evaluates to false, AWS CloudFormation removes the `DBSnapshotIdentifier` property.

JSON

```
"MyDB" : {
  "Type" : "AWS::RDS::DBInstance",
  "Properties" : {
    "AllocatedStorage" : "5",
    "DBInstanceClass" : "db.m1.small",
    "Engine" : "MySQL",
    "EngineVersion" : "5.5",
    "MasterUsername" : { "Ref" : "DBUser" },
    "MasterUserPassword" : { "Ref" : "DBPassword" },
    "DBParameterGroupName" : { "Ref" : "MyRDSParamGroup" },
    "DBSnapshotIdentifier" : {
      "Fn::If" : [
        "UseDBSnapshot",
        {"Ref" : "DBSnapshotName"},
        {"Ref" : "AWS::NoValue"}
```

```
      ]
    }
  }
}
```

YAML

```
MyDB:
  Type: "AWS::RDS::DBInstance"
  Properties:
    AllocatedStorage: 5
    DBInstanceClass: db.m1.small
    Engine: MySQL
    EngineVersion: 5.5
    MasterUsername: !Ref DBUser
    MasterUserPassword: !Ref DBPassword
    DBParameterGroupName: !Ref MyRDSParamGroup
    DBSnapshotIdentifier:
      !If [UseDBSnapshot, !Ref DBSnapshotName, !Ref "AWS::NoValue"]
```

Example 4

The following snippet provides an auto scaling update policy only if the `RollingUpdates` condition evaluates to true. If the condition evaluates to false, AWS CloudFormation removes the `AutoScalingRollingUpdate` update policy.

JSON

```
"UpdatePolicy": {
  "AutoScalingRollingUpdate": {
    "Fn::If": [
      "RollingUpdates",
      {
        "MaxBatchSize": "2",
        "MinInstancesInService": "2",
        "PauseTime": "PT0M30S"
      },
      {
        "Ref" : "AWS::NoValue"
      }
    ]
  }
}
```

YAML

```
UpdatePolicy:
  AutoScalingRollingUpdate:
    !If
      - RollingUpdates
      -
        MaxBatchSize: 2
        MinInstancesInService: 2
        PauseTime: PT0M30S
      - !Ref "AWS::NoValue"
```

Fn::Not

Returns `true` for a condition that evaluates to `false` or returns `false` for a condition that evaluates to `true`. `Fn::Not` acts as a NOT operator.

Declaration

JSON

```
"Fn::Not": [{condition}]
```

YAML

Syntax for the full function name:

```
Fn::Not: [condition]
```

Syntax for the short form:

```
!Not [condition]
```

Parameters

`condition`

A condition such as `Fn::Equals` that evaluates to `true` or `false`.

Example

The following `EnvCondition` condition evaluates to true if the value for the `EnvironmentType` parameter is not equal to `prod`:

JSON

```
"MyNotCondition" : {
   "Fn::Not" : [{
      "Fn::Equals" : [
         {"Ref" : "EnvironmentType"},
         "prod"
      ]
   }]
}
```

YAML

```
MyNotCondition:
  !Not [!Equals [!Ref EnvironmentType, prod]]
```

Fn::Or

Returns `true` if any one of the specified conditions evaluate to true, or returns `false` if all of the conditions evaluates to false. `Fn::Or` acts as an OR operator. The minimum number of conditions that you can include is 2, and the maximum is 10.

Declaration

JSON

```
"Fn::Or": [{condition}, {...}]
```

YAML

Syntax for the full function name:

```
Fn::Or: [condition, ...]
```

Syntax for the short form:

```
!Or [condition, ...]
```

Parameters

`condition`

A condition that evaluates to `true` or `false`.

Example

The following `MyOrCondition` evaluates to true if the referenced security group name is equal to `sg-mysggroup` or if `SomeOtherCondition` evaluates to true:

JSON

```
"MyOrCondition" : {
   "Fn::Or" : [
      {"Fn::Equals" : ["sg-mysggroup", {"Ref" : "ASecurityGroup"}]},
      {"Condition" : "SomeOtherCondition"}
   ]
}
```

YAML

```
MyOrCondition:
  !Or [!Equals [sg-mysggroup, !Ref ASecurityGroup], Condition: SomeOtherCondition]
```

Supported Functions

You can use the following functions in the `Fn::If` condition:

- `Fn::Base64`
- `Fn::FindInMap`
- `Fn::GetAtt`
- `Fn::GetAZs`
- `Fn::If`
- `Fn::Join`
- `Fn::Select`
- `Ref`

You can use the following functions in all other condition functions, such as `Fn::Equals` and `Fn::Or`:

- `Fn::FindInMap`
- `Ref`
- Other condition functions

Sample Templates

Conditionally create resources for a production, development, or test stack

In some cases, you might want to create stacks that are similar but with minor tweaks. For example, you might have a template that you use for production applications. You want to create the same production stack so that you can use it for development or testing. However, for development and testing, you might not require all the extra capacity that's included in a production-level stack. Instead, you can use an environment type input parameter in order to conditionally create stack resources that are specific to production, development, or testing, as shown in the following sample:

You can specify `prod`, `dev`, or `test` for the `EnvType` parameter. For each environment type, the template specifies a different instance type. The instance types can range from a large, compute-optimized instance type to a small general purpose instance type. In order to conditionally specify the instance type, the template defines two conditions in the Conditions section of the template: `CreateProdResources`, which evaluates to true if the `EnvType` parameter value is equal to `prod` and `CreateDevResources`, which evaluates to true if the parameter value is equal to `dev`.

In the `InstanceType` property, the template nests two `Fn::If` intrinsic functions to determine which instance type to use. If the `CreateProdResources` condition is true, the instance type is `c1.xlarge`. If the condition is false, the `CreateDevResources` condition is evaluated. If the `CreateDevResources` condition is true, the instance type is `m1.large` or else the instance type is `m1.small`.

In addition to the instance type, the production environment creates and attaches an Amazon EC2 volume to the instance. The `MountPoint` and `NewVolume` resources are associated with the `CreateProdResources` condition so that the resources are created only if the condition evaluates to true.

Conditionally assign a resource property

In this example, you can create an Amazon RDS DB instance from a snapshot. If you specify the `DBSnapshotName` parameter, AWS CloudFormation uses the parameter value as the snapshot name when creating the DB instance. If you keep the default value (empty string), AWS CloudFormation removes the `DBSnapshotIdentifier` property and creates a DB instance from scratch.

The `UseDBSnapshot` condition evaluates to true only if the `DBSnapshotName` is not an empty string. If the `UseDBSnapshot` condition evaluates to true, AWS CloudFormation uses the `DBSnapshotName` parameter value for the `DBSnapshotIdentifier` property. If the condition evaluates to false, AWS CloudFormation removes the `DBSnapshotIdentifier` property. The `AWS::NoValue` pseudo parameter removes the corresponding resource property when it is used as a return value.

Conditionally use an existing resource

In this example, you can use an Amazon EC2 security group that has already been created or you can create a new security group, which is specified in the template. For the `ExistingSecurityGroup` parameter, you can specify the `default` security group name or `NONE`. If you specify `default`, AWS CloudFormation uses a security group that has already been created and is named `default`. If you specify `NONE`, AWS CloudFormation creates the security group that's defined in the template.

To determine whether to create the `NewSecurityGroup` resource, the resource is associated with the `CreateNewSecurityGroup` condition. The resource is created only when the condition is true (when the `ExistingSecurityGroup` parameter is equal to `NONE`).

In the `SecurityGroups` property, the template uses the `Fn::If` intrinsic function to determine which security group to use. If the `CreateNewSecurityGroup` condition evaluates to true, the security group property references the `NewSecurityGroup` resource. If the `CreateNewSecurityGroup` condition evaluates to false, the security group property references the `ExistingSecurityGroup` parameter (the `default` security group).

Lastly, the template conditionally outputs the security group ID. If the `CreateNewSecurityGroup` condition evaluates to true, AWS CloudFormation outputs the security group ID of the `NewSecurityGroup` resource. If the condition is false, AWS CloudFormation outputs the security group ID of the `ExistingSecurityGroup` resource.

`Fn::FindInMap`

The intrinsic function `Fn::FindInMap` returns the value corresponding to keys in a two-level map that is declared in the `Mappings` section.

Declaration

JSON

```
{ "Fn::FindInMap" : [ "MapName", "TopLevelKey", "SecondLevelKey"] }
```

YAML

Syntax for the full function name:

```
Fn::FindInMap: [ MapName, TopLevelKey, SecondLevelKey ]
```

Syntax for the short form:

```
!FindInMap [ MapName, TopLevelKey, SecondLevelKey ]
```

> **Note**
> You can't nest two instances of two functions in short form.

Parameters

MapName
: The logical name of a mapping declared in the Mappings section that contains the keys and values.

TopLevelKey
: The top-level key name. Its value is a list of key-value pairs.

SecondLevelKey
: The second-level key name, which is set to one of the keys from the list assigned to `TopLevelKey`.

Return Value:

The value that is assigned to `SecondLevelKey`.

Example

The following example shows how to use `Fn::FindInMap` for a template with a `Mappings` section that contains a single map, `RegionMap`, that associates AMIs with AWS regions.

- The map has 5 top-level keys that correspond to various AWS regions.
- Each top-level key is assigned a list with two second level keys, `"32"` and `"64"`, that correspond to the AMI's architecture.
- Each of the second-level keys is assigned an appropriate AMI name.

The example template contains an `AWS::EC2::Instance` resource whose `ImageId` property is set by the `FindInMap` function.

`MapName` is set to the map of interest, `"RegionMap"` in this example. `TopLevelKey` is set to the region where the stack is created, which is determined by using the `"AWS::Region"` pseudo parameter. `SecondLevelKey` is set to the desired architecture, `"32"` for this example.

`FindInMap` returns the AMI assigned to `FindInMap`. For a 32-bit instance in us-east-1, `FindInMap` would return `"ami-6411e20d"`.

JSON

```
{
  ...
  "Mappings" : {
    "RegionMap" : {
      "us-east-1" : { "32" : "ami-6411e20d", "64" : "ami-7a11e213" },
      "us-west-1" : { "32" : "ami-c9c7978c", "64" : "ami-cfc7978a" },
      "eu-west-1" : { "32" : "ami-37c2f643", "64" : "ami-31c2f645" },
      "ap-southeast-1" : { "32" : "ami-66f28c34", "64" : "ami-60f28c32" },
      "ap-northeast-1" : { "32" : "ami-9c03a89d", "64" : "ami-a003a8a1" }
    }
  },

  "Resources" : {
     "myEC2Instance" : {
        "Type" : "AWS::EC2::Instance",
        "Properties" : {
           "ImageId" : { "Fn::FindInMap" : [ "RegionMap", { "Ref" : "AWS::Region" },
 "32"]},
           "InstanceType" : "m1.small"
        }
     }
 }
}
```

YAML

```
Mappings:
  RegionMap:
    us-east-1:
      32: "ami-6411e20d"
      64: "ami-7a11e213"
    us-west-1:
      32: "ami-c9c7978c"
      64: "ami-cfc7978a"
    eu-west-1:
      32: "ami-37c2f643"
      64: "ami-31c2f645"
    ap-southeast-1:
      32: "ami-66f28c34"
      64: "ami-60f28c32"
    ap-northeast-1:
      32: "ami-9c03a89d"
      64: "ami-a003a8a1"
Resources:
  myEC2Instance:
    Type: "AWS::EC2::Instance"
    Properties:
      ImageId: !FindInMap [ RegionMap, !Ref "AWS::Region", 32 ]
      InstanceType: m1.small
```

Supported Functions

You can use the following functions in a `Fn::FindInMap` function:

- `Fn::FindInMap`
- `Ref`

`Fn::GetAtt`

The `Fn::GetAtt` intrinsic function returns the value of an attribute from a resource in the template.

Declaration

JSON

```
{ "Fn::GetAtt" : [ "logicalNameOfResource", "attributeName" ] }
```

YAML

Syntax for the full function name:

```
Fn::GetAtt: [ logicalNameOfResource, attributeName ]
```

Syntax for the short form:

```
!GetAtt logicalNameOfResource.attributeName
```

Parameters

`logicalNameOfResource`

The logical name of the resource that contains the attribute that you want.

`attributeName`

The name of the resource-specific attribute whose value you want. See the resource's reference page for details about the attributes available for that resource type.

Return Value

The attribute value.

Example

This example returns a string containing the DNS name of the Lload balancer with the logical name `MyLB`.

JSON

```
"Fn::GetAtt" : [ "MyLB" , "DNSName" ]
```

YAML

```
!GetAtt MyLB.DNSName
```

Supported Functions

For the `Fn::GetAtt` logical resource name, you cannot use functions. You must specify a string that is a resource's logical ID.

For the `Fn::GetAtt` attribute name, you can use the `Ref` function.

Attributes

You can retrieve the following attributes using `Fn::GetAtt`.

Resource TypeName	Attribute	Description
AWS::ApiGateway::RestApi (p. 417)	`RootResourceId`	The root resource ID for a `RestApi` resource. Example: `a0bc123d4e`
AWS::CloudFormation::WaitCondition (p. 487)	`Data`	A JSON-format string containing the `UniqueId` and `Data` values from the wait condition signal(s) for the specified wait condition. For more information about wait condition signals, see Wait Condition Signal JSON Format (p. 231). Example of a wait condition with two signals: `{"Signal1":"Step 1 complete.","Signal2":"Step 2 complete."}`
AWS::CloudFormation::Stack (p. 485)	`Outputs.NestedStackOutputName`	The output value from the nested stack that you specified, where *`NestedStackOutputName`* is the name of the output value.
AWS::CloudFront::Distribution (p. 492)	`DomainName`	Example: `d2fadu0nynjpfn.cloudfront.net`
AWS::CodeBuild::Project (p. 503)	`Arn`	Example: `arn:aws:codebuild:us-west-2:123456789012:project/myProjectName`
AWS::CodeCommit::Repository (p. 507)	`Arn`	Example: `arn:aws:codecommit:us-east-1:123456789012:MyDemoRepo`
AWS::CodeCommit::Repository (p. 507)	`CloneUrlHttp`	Example: `https://codecommit.us-east-1.amazonaws.com/v1/repos/MyDemoRepo`
AWS::CodeCommit::Repository (p. 507)	`CloneUrlSsh`	Example: `ssh://git-codecommit.us-east-1.amazonaws.com/v1/repos//v1/repos/MyDemoRepo`
AWS::CodeCommit::Repository (p. 507)	`Name`	Example: `MyDemoRepo`
AWS::Config::ConfigRule (p. 525)	`Arn`	Example: `arn:aws:config:us-east-1:123456789012:config-rule/config-rule-a1bzhi`
AWS::Config::ConfigRule (p. 525)	`ConfigRuleId`	Example: `config-rule-a1bzhi`
AWS::Config::ConfigRule (p. 525)	`Compliance.Type`	Example: `COMPLIANT`
AWS::DirectoryService::MicrosoftAD (p. 544) and AWS::DirectoryService::SimpleAD (p. 547)	`Alias`	The alias for a directory. Examples: `d-12373a053a` or `alias4-mydirectory-12345abcgmzsk` (if you have the `CreateAlias` property set to true)

Resource TypeName	Attribute	Description
AWS::DirectoryService::MicrosoftAD (p. 544) and AWS::DirectoryService::SimpleAD (p. 547)	`DnsIpAddresses`	The IP addresses of the DNS servers for the directory. Example: `[ "192.0.2.1", "192.0.2.2" ]`
AWS::DynamoDB::Table (p. 550)	`StreamArn`	The Amazon Resource Name (ARN) of the DynamoDB stream. To use this attribute, you must specify the DynamoDB table `StreamSpecification` property. Example: `arn:aws:dynamodb:us-east-1:123456789012:table/testddbstack-myDynamoDBTable-012A1SL7SMP5Q/stream/2015-11-30T20:10:00.000`
AWS::EC2::EIP (p. 564)	`AllocationId`	The ID that AWS assigns to represent the allocation of the address for use with Amazon VPC. It is returned only for VPC Elastic IP addresses. Example: `eipalloc-5723d13e`
AWS::EC2::Instance (p. 574)	`AvailabilityZone`	The Availability Zone where the instance that you specified is launched. Example: us-east-1b
AWS::EC2::Instance (p. 574)	`PrivateDnsName`	The private DNS name of the instance that you specified. Example: `ip-10-24-34-0.ec2.internal`
AWS::EC2::Instance (p. 574)	`PublicDnsName`	The public DNS name of the instance that you specified. Example: `ec2-107-20-50-45.compute-1.amazonaws.com`
AWS::EC2::Instance (p. 574)	`PrivateIp`	The private IP address of the instance that you specified. Example: `10.24.34.0`
AWS::EC2::Instance (p. 574)	`PublicIp`	The public IP address of the instance that you specified. Example: `192.0.2.0`
AWS::EC2::NetworkInterface (p. 594)	`PrimaryPrivateIpAddress`	The primary private IP address of the network interface that you specified. Example: `10.0.0.192`
AWS::EC2::NetworkInterface (p. 594)	`SecondaryPrivateIpAddresses`	The secondary private IP addresses of the network interface that you specified. Example: `["10.0.0.161", "10.0.0.162", "10.0.0.163"]`
AWS::EC2::SecurityGroup (p. 608)	`GroupId`	The group ID of the specified security group. Example: `sg-94b3a1f6`
AWS::EC2::Subnet (p. 625)	`AvailabilityZone`	The Availability Zone of the subnet. Example: `us-east-1a`

Resource TypeName	Attribute	Description
AWS::EC2::Subnet (p. 625)	`Ipv6CidrBlocks`	A list of IPv6 CIDR blocks that are associated with the subnet. Example: `[ 2001:db8:1234:1a00::/64 ]`
AWS::EC2::SubnetNetworkAclAssociation (p. 629)	`AssociationId`	The `NetworkAcl associationId` that is attached to a subnet.
AWS::EC2::VPC (p. 639)	`CidrBlock`	The set of IP addresses for the VPC. Example: `10.0.0.0/16`
AWS::EC2::VPC (p. 639)	`DefaultNetworkAcl`	The default network ACL ID that is associated with the VPC, which AWS creates when you create a VPC. Example: `acl-814dafe3`
AWS::EC2::VPC (p. 639)	`DefaultSecurityGroup`	The default security group ID that is associated with the VPC, which AWS creates when you create a VPC. Example: `sg-b178e0d3`
AWS::EC2::VPC (p. 639)	`Ipv6CidrBlocks`	A list of IPv6 CIDR blocks that are associated with the VPC. Example: `[ 2001:db8:1234:1a00::/56 ]`
AWS::ECS::Service (p. 671)	`Name`	The name of an Amazon EC2 Container Service service. Example: `sample-webapp`
AWS::ElastiCache::CacheCluster (p. 684)	`ConfigurationEndpoint.Address`	The DNS address of the configuration endpoint for the Memcached cache cluster. Example: `test.abc12a.cfg.use1.cache.amazonaws.com:11111`
AWS::ElastiCache::CacheCluster (p. 684)	`ConfigurationEndpoint.Port`	The port number of the configuration endpoint for the Memcached cache cluster.
AWS::ElastiCache::CacheCluster (p. 684)	`RedisEndpoint.Address`	The DNS address of the configuration endpoint for the Redis cache cluster. Example: `test.abc12a.cfg.use1.cache.amazonaws.com:11111`
AWS::ElastiCache::CacheCluster (p. 684)	`RedisEndpoint.Port`	The port number of the configuration endpoint for the Redis cache cluster.
AWS::ElastiCache::ReplicationGroup (p. 698)	`PrimaryEndPoint.Address`	The DNS address of the primary read-write cache node.
AWS::ElastiCache::ReplicationGroup (p. 698)	`PrimaryEndPoint.Port`	The port number that the primary read-write cache engine is listening on.

Resource TypeName	Attribute	Description
AWS::ElastiCache::ReplicationGroup (p. 693)	`ReadEndPoint.Addresses`	A string with a list of endpoints for the read-only replicas. The order of the addresses map to the order of the ports from the `ReadEndPoint.Ports` attribute. Example: `"[abc12xmy3d1w3hv6-001.rep12a.0001.use1.cache.amazonaws.com, abc12xmy3d1w3hv6-002.rep12a.0001.use1.cache.amazonaws.com, abc12xmy3d1w3hv6-003.rep12a.0001.use1.cache.amazonaws.com]"`
AWS::ElastiCache::ReplicationGroup (p. 693)	`ReadEndPoint.Ports`	A string with a list of ports for the read-only replicas. The order of the ports maps to the order of the addresses from the `ReadEndPoint.Addresses` attribute. Example: `"[6379, 6379, 6379]"`
AWS::ElastiCache::ReplicationGroup (p. 693)	`ReadEndPoint.Addresses.List`	A list of endpoints for the read-only replicas. Example: `["abc12xmy3d1w3hv6-001.rep12a.0001.use1.cache.amazonaws.com", "abc12xmy3d1w3hv6-002.rep12a.0001.use1.cache.amazonaws.com", "abc12xmy3d1w3hv6-003.rep12a.0001.use1.cache.amazonaws.com"]`
AWS::ElastiCache::ReplicationGroup (p. 693)	`ReadEndPoint.Ports.List`	A list of ports for the read-only replicas. Example: `["6379","6379","6379"]`
AWS::ElasticBeanstalk::Environment (p. 714)	`EndpointURL`	The URL to the load balancer for this environment. Example: `awseb-myst-myen-132MQC4KRLAMD-1371280482.us-east-1.elb.amazonaws.com`
AWS::ElasticLoadBalancing::LoadBalancer (p. 719)	`CanonicalHostedZoneName`	The name of the Amazon Route 53-hosted zone that is associated with the load balancer. Example: `mystack-myelb-15HMABG9ZCN57-1013119603.us-east-1.elb.amazonaws.com`
AWS::ElasticLoadBalancing::LoadBalancer (p. 719)	`CanonicalHostedZoneNameID`	The ID of the Amazon Route 53 hosted zone name that is associated with the l oad balancer. Example: `Z3DZXE0Q79N41H`
AWS::ElasticLoadBalancing::LoadBalancer (p. 719)	`DNSName`	The DNS name for the load balancer. Example: `mystack-myelb-15HMABG9ZCN57-1013119603.us-east-1.elb.amazonaws.com`
AWS::ElasticLoadBalancing::LoadBalancer (p. 719)	`SourceSecurityGroup.GroupName`	The security group that you can use as part of your inbound rules for your load balancer's back-end Amazon EC2 application instances. Example: `amazon-elb`
AWS::ElasticLoadBalancing::LoadBalancer (p. 719)	`SourceSecurityGroup.OwnerAlias`	The owner of the source security group. Example: `amazon-elb-sg`

Resource TypeName	Attribute	Description
AWS::ElasticLoadBalancingV2::LoadBalancer (p. 736)	`DNSName`	The DNS name for the application load balancer. Example: `my-load-balancer-424835706.us-west-2.elb.amazonaws.com`
AWS::ElasticLoadBalancingV2::LoadBalancer (p. 736)	`CanonicalHostedZoneID`	The ID of the Amazon Route 53-hosted zone name that is associated with the load balancer. Example: `Z2P70J7EXAMPLE`
AWS::ElasticLoadBalancingV2::LoadBalancer (p. 736)	`LoadBalancerFullName`	The full name of the application load balancer. Example: `app/my-load-balancer/50dc6c495c0c9188`
AWS::ElasticLoadBalancingV2::LoadBalancer (p. 736)	`LoadBalancerName`	The name of the application load balancer. Example: `my-load-balancer`
AWS::ElasticLoadBalancingV2::LoadBalancer (p. 736)	`SecurityGroups`	The IDs of the security groups for the application load balancer. Example: `sg-123456a`
AWS::ElasticLoadBalancingV2::TargetGroup (p. 739)	`LoadBalancerArns`	The Amazon Resource Names (ARNs) of the load balancers that route traffic to this target group. Example: `[ "arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188" ]`
AWS::ElasticLoadBalancingV2::TargetGroup (p. 739)	`TargetGroupFullName`	The full name of the target group. Example: `targetgroup/my-target-group/cbf133c568e0d028`
AWS::Elasticsearch::Domain (p. 745)	`DomainArn`	The Amazon Resource Name (ARN) of the domain. Example: `arn:aws:es:us-west-2:123456789012:domain/mystack-elasti-1ab2cdefghij`
AWS::Elasticsearch::Domain (p. 745)	`DomainEndpoint`	The domain-specific endpoint that is used to submit index, search, and data upload requests to an Amazon Elasticsearch Service domain. Example: `search-mystack-elasti-1ab2cdefghij-ab1c2deckoyb3hofw7wpqa3cm.us-west-2.es.amazonaws.com`
AWS::EMR::Cluster (p. 750)	`MasterPublicDNS`	The public DNS name of the master node (instance). Example: `ec2-12-123-123-123.us-west-2.compute.amazonaws.com`
AWS::Events::Rule (p. 761)	`Arn`	The Amazon Resource Name (ARN) of the event rule. Example: `arn:aws:events:us-east-1:123456789012:rule/example`

Resource TypeName	Attribute	Description
AWS::IAM::AccessKey (p. 775)	`SecretAccessKey`	The secret access key for the specified `Access Key`. Example: `wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY`
AWS::IAM::Group (p. 777)	`Arn`	Example: `arn:aws:iam::123456789012:group/mystack-mygroup-1DZETITOWEKVO`
AWS::IAM::InstanceProfile (p. 779)	`Arn`	Returns the Amazon Resource Name (ARN) for the instance profile. Example: `arn:aws:iam::1234567890:instance-profile/MyProfile-ASDNSDLKJ`
AWS::IAM::Role (p. 787)	`Arn`	Example: `arn:aws:iam::1234567890:role/MyRole-AJJHDSKSDF`
AWS::IAM::User (p. 793)	`Arn`	Example: `arn:aws:iam::123456789012:user/mystack-myuser-1CCXAFG2H2U4D`
AWS::IoT::Certificate (p. 797)	`Arn`	Example: `arn:aws:iot:ap-southeast-2:123456789012:cert/a1234567b89c012d3e4fg567hij8k9l01mno1p23q45678901rs234567890t1u`
AWS::Kinesis::Stream (p. 810)	`Arn`	The ARN of the Amazon Kinesis stream. Example: `arn:aws:kinesis:us-east-1:123456789012:stream/mystream`.
AWS::KMS::Key (p. 816)	`Arn`	The ARN of the AWS KMS key. Example: `arn:aws:kms:us-west-2:123456789012:key/12a34567-8c90-1defg-af84-0bf06c1747f3`.
AWS::Lambda::Function (p. 824)	`Arn`	Example: `arn:aws:lambda:us-west-2:123456789012:MyStack-AMILookUp-NT5EUXTNTXXD`
AWS::Lambda::Version (p. 831)	`Version`	The version of a Lambda function. Example: `1`
AWS::Logs::LogGroup (p. 835)	`Arn`	The ARN of the Amazon CloudWatch Logs log group. Example: `arn:aws:logs:us-east-1:123456789012:log-group:/mystack-testgroup-12ABC1AB12A1:*`
AWS::OpsWorks::Instance (p. 848)	`PublicIp`	The public IP address of an AWS OpsWorks instance. **Note** To use this attribute, the AWS OpsWorks instance must be in an AWS OpsWorks layer that auto-assigns public IP addresses. Example: `192.0.2.0`

Resource TypeName	Attribute	Description
AWS::Redshift::Cluster (p. 911)	`Endpoint.Address`	The connection endpoint for the cluster. Example: `examplecluster.cg034hpkmmjt.us-east-1.redshift.amazonaws.com`
AWS::Redshift::Cluster (p. 911)	`Endpoint.Port`	The connection port for the cluster. Example: `5439`
AWS::RDS::DBCluster (p. 872)	`Endpoint.Address`	The connection endpoint for the DB cluster. Example: `mystack-mydbcluster-1apw1j4phylrk.cg034hpkmmjt.us-east-1.rds.amazonaws.com`
AWS::RDS::DBCluster (p. 872)	`Endpoint.Port`	The port number on which the DB cluster accepts connections. Example: `3306`
AWS::RDS::DBInstance (p. 881)	`Endpoint.Address`	The connection endpoint for the database. Example: `mystack-mydb-1apw1j4phylrk.cg034hpkmmjt.us-east-1.rds.amazonaws.com`
AWS::RDS::DBInstance (p. 881)	`Endpoint.Port`	The port number on which the database accepts connections. Example: `3306`
AWS::Route53::HostedZone (p. 927)	`NameServers`	Returns the set of name servers for the specific hosted zone. Example: `ns1.example.com`
AWS::S3::Bucket (p. 937)	`DomainName`	The DNS name of the specified bucket. Example: `mystack-mybucket-kdwwxmddtr2g.s3.amazonaws.com`
AWS::S3::Bucket (p. 937)	`WebsiteURL`	The Amazon S3 website endpoint for the specified bucket. Example: `http://mystack-mybucket-kdwwxmddtr2g.s3-website-us-east-1.amazonaws.com/`
AWS::Serverless::Function (p. 151)	No attribute.	The ARN of an `AWS::Serverless::Function` resource.
AWS::SNS::Topic (p. 954)	`TopicName`	The name of an Amazon SNS topic. Example: `my-sns-topic`
AWS::StepFunctions::Activity (p. 975)	`Name`	The name of the AWS Step Functions activity.
AWS::StepFunctions::StateMachine (p. 976)	`Name`	The name of the Step Functions state machine.

Resource TypeName	Attribute	Description
AWS::SQS::Queue (p. 958)	Arn	The ARN for the specified queue. Example: `arn:aws:sqs:us-east-1:123456789012:mystack-myqueue-15PG5C2FC1CW8`
AWS::SQS::Queue (p. 958)	QueueName	The name of an Amazon SQS queue. Example: `mystack-myqueue-1VF9BKQH5BJVI`

Fn::GetAZs

The intrinsic function `Fn::GetAZs` returns an array that lists Availability Zones for a specified region. Because customers have access to different Availability Zones, the intrinsic function `Fn::GetAZs` enables template authors to write templates that adapt to the calling user's access. That way you don't have to hard-code a full list of Availability Zones for a specified region.

> **Important**
> For the EC2-Classic platform, the `Fn::GetAZs` function returns all Availability Zones for a region. For the EC2-VPC platform, the `Fn::GetAZs` function returns only Availability Zones that have a default subnet unless none of the Availability Zones has a default subnet; in that case, all Availability Zones are returned.
> Similarly to the response from the `describe-availability-zones` AWS CLI command, the order of the results from the `Fn::GetAZs` function is not guaranteed and can change when new Availability Zones are added.

IAM permissions

The permissions that you need in order to use the `Fn::GetAZs` function depend on the platform in which you're launching Amazon EC2 instances. For both platforms, you need permissions to the Amazon EC2 `DescribeAvailabilityZones` and `DescribeAccountAttributes` actions. For EC2-VPC, you also need permissions to the Amazon EC2 `DescribeSubnets` action.

Declaration

JSON

```
{ "Fn::GetAZs" : "region" }
```

YAML

Syntax for the full function name:

```
Fn::GetAZs: region
```

Syntax for the short form:

```
!GetAZs region
```

Parameters

region
: The name of the region for which you want to get the Availability Zones.

You can use the `AWS::Region` pseudo parameter to specify the region in which the stack is created. Specifying an empty string is equivalent to specifying `AWS::Region`.

Return Value

The list of Availability Zones for the region.

Examples

For these examples, AWS CloudFormation evaluates `Fn::GetAZs` to the following array—assuming that the user has created the stack in the `us-east-1` region:

```
[ "us-east-1a", "us-east-1b", "us-east-1c", "us-east-1d" ]
```

JSON

```
{ "Fn::GetAZs" : "" }
{ "Fn::GetAZs" : { "Ref" : "AWS::Region" } }
{ "Fn::GetAZs" : "us-east-1" }
```

YAML

```
Fn::GetAZs: ""
Fn::GetAZs:
  Ref: "AWS::Region"
Fn::GetAZs: us-east-1
```

Specify a Subnet's Availability Zone

The following example uses `Fn::GetAZs` to specify a subnet's Availability Zone:

JSON

```
"mySubnet" : {
  "Type" : "AWS::EC2::Subnet",
  "Properties" : {
    "VpcId" : { "Ref" : "VPC" },
    "CidrBlock" : "10.0.0.0/24",
    "AvailabilityZone" : {
      "Fn::Select" : [ "0", { "Fn::GetAZs" : "" } ]
    }
  }
}
```

YAML

```
mySubnet:
  Type: "AWS::EC2::Subnet"
  Properties:
    VpcId:
      !Ref VPC
    CidrBlock: 10.0.0.0/24
    AvailabilityZone:
      Fn::Select:
        - 0
```

```
        - Fn::GetAZs: ""
```

Supported Functions

You can use the `Ref` function in the `Fn::GetAZs` function.

Fn::ImportValue

The intrinsic function `Fn::ImportValue` returns the value of an output exported (p. 155) by another stack. You typically use this function to create cross-stack references (p. 200). In the following example template snippets, Stack A exports VPC security group values and Stack B imports them.

Note
The following restrictions apply to cross-stack references:

- For each AWS account, `Export` names must be unique within a region.
- You can't create cross-stack references across regions. You can use the intrinsic function `Fn::ImportValue` to import only values that have been exported within the same region.
- For outputs, the value of the `Name` property of an `Export` can't use `Ref` or `GetAtt` functions that depend on a resource.

  Similarly, the `ImportValue` function can't include `Ref` or `GetAtt` functions that depend on a resource.
- You can't delete a stack if another stack references one of its outputs.
- You can't modify or remove an output value that is referenced by another stack.

Stack A Export

```
"Outputs" : {
  "PublicSubnet" : {
    "Description" : "The subnet ID to use for public web servers",
    "Value" :  { "Ref" : "PublicSubnet" },
    "Export" : { "Name" : {"Fn::Sub": "${AWS::StackName}-SubnetID" }}
  },
  "WebServerSecurityGroup" : {
    "Description" : "The security group ID to use for public web servers",
    "Value" :  { "Fn::GetAtt" : ["WebServerSecurityGroup", "GroupId"] },
    "Export" : { "Name" : {"Fn::Sub": "${AWS::StackName}-SecurityGroupID" }}
  }
}
```

Stack B Import

```
"Resources" : {
  "WebServerInstance" : {
    "Type" : "AWS::EC2::Instance",
    "Properties" : {
      "InstanceType" : "t2.micro",
      "ImageId" : "ami-a1b23456",
      "NetworkInterfaces" : [{
        "GroupSet" : [{"Fn::ImportValue" : {"Fn::Sub" : "${NetworkStackNameParameter}-
SecurityGroupID"}}],
        "AssociatePublicIpAddress" : "true",
        "DeviceIndex" : "0",
        "DeleteOnTermination" : "true",
        "SubnetId" : {"Fn::ImportValue" : {"Fn::Sub" : "${NetworkStackNameParameter}-
SubnetID"}}
      }]
```

```
    }
  }
}
```

Declaration

JSON

```
{ "Fn::ImportValue" : sharedValueToImport }
```

YAML

You can use the full function name:

```
Fn::ImportValue: sharedValueToImport
```

Alternatively, you can use the short form:

```
!ImportValue sharedValueToImport
```

Important
You can't use the short form of `!ImportValue` when it contains a `!Sub`. The following example is valid for AWS CloudFormation, but *not* valid for YAML:

```
!ImportValue
  !Sub "${NetworkStack}-SubnetID"
```

Instead, you must use the full function name, for example:

```
Fn::ImportValue:
  !Sub "${NetworkStack}-SubnetID"
```

Parameters

ValueToImport
: The stack output value that you want to import.

Return Value

The stack output value.

Example

JSON

```
{ "Fn::ImportValue" : {"Fn::Sub": "${NetworkStackNameParameter}-SubnetID" } }
```

YAML

```
Fn::ImportValue:
```

```
!Sub "${NetworkStackName}-SecurityGroupID"
```

Supported Functions

You can use the following functions in the `Fn::ImportValue` function. The value of these functions can't depend on a resource.

- `Fn::Base64`
- `Fn::FindInMap`
- `Fn::If`
- `Fn::Join`
- `Fn::Select`
- `Fn::Split`
- `Fn::Sub`
- `Ref`

`Fn::Join`

The intrinsic function `Fn::Join` appends a set of values into a single value, separated by the specified delimiter. If a delimiter is the empty string, the set of values are concatenated with no delimiter.

Declaration

JSON

```
{ "Fn::Join" : [ "delimiter", [ comma-delimited list of values ] ] }
```

YAML

Syntax for the full function name:

```
Fn::Join: [ delimiter, [ comma-delimited list of values ] ]
```

Syntax for the short form:

```
!Join [ delimiter, [ comma-delimited list of values ] ]
```

Parameters

delimiter
: The value you want to occur between fragments. The delimiter will occur between fragments only. It will not terminate the final value.

ListOfValues
: The list of values you want combined.

Return Value

The combined string.

Example

The following example returns: `"a:b:c"`.

JSON

```
"Fn::Join" : [ ":", [ "a", "b", "c" ] ]
```

YAML

```
!Join [ ":", [ a, b, c ] ]
```

Supported Functions

For the `Fn::Join` delimiter, you cannot use any functions. You must specify a string value.

For the `Fn::Join` list of values, you can use the following functions:

- `Fn::Base64`
- `Fn::FindInMap`
- `Fn::GetAtt`
- `Fn::GetAZs`
- `Fn::If`
- `Fn::Join`
- `Fn::Split`
- `Fn::Select`
- `Ref`

`Fn::Select`

The intrinsic function `Fn::Select` returns a single object from a list of objects by index.

> **Important**
> Fn::Select does not check for null values or if the index is out of bounds of the array. Both conditions will result in a stack error, so you should be certain that the index you choose is valid, and that the list contains non-null values.

Declaration

JSON

```
{ "Fn::Select" : [ index, listOfObjects ] }
```

YAML

Syntax for the full function name:

```
Fn::Select: [ index, listOfObjects ]
```

Syntax for the short form:

```
!Select [ index, listOfObjects ]
```

Parameters

index

The index of the object to retrieve. This must be a value from zero to N-1, where N represents the number of elements in the array.

listOfObjects

The list of objects to select from. This list must not be null, nor can it have null entries.

Return Value

The selected object.

Examples

Basic Example

The following example returns: `"grapes"`.

JSON

```
{ "Fn::Select" : [ "1", [ "apples", "grapes", "oranges", "mangoes" ] ] }
```

YAML

```
!Select [ "1", [ "apples", "grapes", "oranges", "mangoes" ] ]
```

Comma-delimited List Parameter Type

You can use `Fn::Select` to select an object from a `CommaDelimitedList` parameter. You might use a `CommaDelimitedList` parameter to combine the values of related parameters, which reduces the total number of parameters in your template. For example, the following parameter specifies a comma-delimited list of three CIDR blocks:

To specify one of the three CIDR blocks, use `Fn::Select` in the Resources section of the same template, as shown in the following sample snippet:

JSON

```
"Subnet0": {
  "Type": "AWS::EC2::Subnet",
    "Properties": {
      "VpcId": { "Ref": "VPC" },
      "CidrBlock": { "Fn::Select" : [ "0", {"Ref": "DbSubnetIpBlocks"} ] }
    }
}
```

YAML

```
Subnet0:
```

```
  Type: "AWS::EC2::Subnet"
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [ 0, !Ref DbSubnetIpBlocks ]
```

Supported Functions

For the `Fn::Select` index value, you can use the `Ref` and `Fn::FindInMap` functions.

For the `Fn::Select` list of objects, you can use the following functions:

- `Fn::FindInMap`
- `Fn::GetAtt`
- `Fn::GetAZs`
- `Fn::If`
- `Fn::Split`
- `Ref`

Fn::Split

To split a string into a list of string values so that you can select an element from the resulting string list, use the `Fn::Split` intrinsic function. Specify the location of splits with a delimiter, such as `,` (a comma). After you split a string, use the `Fn::Select` (p. 1337) function to pick a specific element.

For example, if a comma-delimited string of subnet IDs is imported to your stack template, you can split the string at each comma. From the list of subnet IDs, use the `Fn::Select` intrinsic function to specify a subnet ID for a resource.

Declaration

JSON

```
{ "Fn::Split" : [ "delimiter", "source string" ] }
```

YAML

Syntax for the full function name:

```
Fn::Split: [ delimiter, source string ]
```

Syntax for the short form:

```
!Split [ delimiter, source string ]
```

Parameters

You must specify both parameters.

delimiter
: A string value that determines where the source string is divided.

source string
: The string value that you want to split.

Return Value

A list of string values.

Examples

The following examples demonstrate the behavior of the `Fn::Split` function.

Simple List

The following example splits a string at each vertical bar (|). The function returns `["a", "b", "c"]`.

JSON

```
{ "Fn::Split" : [ "|" , "a|b|c" ] }
```

YAML

```
!Split [ "|" , "a|b|c" ]
```

List with Empty String Values

If you split a string with consecutive delimiters, the resulting list will include an empty string. The following example shows how a string with two consecutive delimiters and an appended delimiter is split. The function returns `["a", "", "c", ""]`.

JSON

```
{ "Fn::Split" : [ "|" , "a||c|" ] }
```

YAML

```
!Split [ "|" , "a||c|" ]
```

Split an Imported Output Value

The following example splits an imported output value, and then selects the third element from the resulting list of subnet IDs, as specified by the `Fn::Select` function.

JSON

```
{ "Fn::Select" : [ "2", { "Fn::Split": [",", {"Fn::ImportValue": "AccountSubnetIDs"}]}] }
```

YAML

```
!Select [2, !Split [",", !ImportValue AccountSubnetIDs]]
```

Supported Functions

For the `Fn::Split` delimiter, you cannot use any functions. You must specify a string value.

For the `Fn::Split` list of values, you can use the following functions:

- `Fn::Base64`

- Fn::FindInMap
- Fn::GetAtt
- Fn::GetAZs
- Fn::If
- Fn::Join
- Fn::Select
- Ref

Fn::Sub

The intrinsic function `Fn::Sub` substitutes variables in an input string with values that you specify. In your templates, you can use this function to construct commands or outputs that include values that aren't available until you create or update a stack.

Declaration

The following sections show the function's syntax.

JSON

```
{ "Fn::Sub" : [ String, { Var1Name: Var1Value, Var2Name: Var2Value } ] }
```

If you're substituting only template parameters, resource logical IDs, or resource attributes in the *String* parameter, don't specify a variable map.

```
{ "Fn::Sub" : String }
```

YAML

Syntax for the full function name:

```
Fn::Sub:
  - String
  - { Var1Name: Var1Value, Var2Name: Var2Value }
```

Syntax for the short form:

```
!Sub
  - String
  - { Var1Name: Var1Value, Var2Name: Var2Value }
```

If you're substituting only template parameters, resource logical IDs, or resource attributes in the *String* parameter, don't specify a variable map.

Syntax for the full function name:

```
Fn::Sub: String
```

Syntax for the short form:

```
!Sub String
```

Parameters

`String`

A string with variables that AWS CloudFormation substitutes with their associated values at runtime. Write variables as `${MyVarName}`. Variables can be template parameter names, resource logical IDs, resource attributes, or a variable in a key-value map. If you specify only template parameter names, resource logical IDs, and resource attributes, don't specify a key-value map.

If you specify template parameter names or resource logical IDs, such as `${InstanceTypeParameter}`, AWS CloudFormation returns the same values as if you used the `Ref` intrinsic function. If you specify resource attributes, such as `${MyInstance.PublicIp}`, AWS CloudFormation returns the same values as if you used the `Fn::GetAtt` intrinsic function.

To write a dollar sign and curly braces (`${}`) literally, add an exclamation point (`!`) after the open curly brace, such as `${!Literal}`. AWS CloudFormation resolves this text as `${Literal}`.

`VarName`

The name of a variable that you included in the `String` parameter.

`VarValue`

The value that AWS CloudFormation substitutes for the associated variable name at runtime.

Return Value

AWS CloudFormation returns the original string, substituting the values for all of the variables.

Examples

The following examples demonstrate how to use the `Fn::Sub` function.

UserData Commands

The following example uses `Fn::Sub` to substitute the AWS CloudFormation stack name and the AWS region pseudo parameters for the actual stack name and region at runtime.

JSON

For readability, the JSON example uses the `Fn::Join` function to separate each command, instead of specifying the entire user data script in a single string value.

```
"UserData": { "Fn::Base64": { "Fn::Join": ["\n", [
  "#!/bin/bash -xe",
  "yum update -y aws-cfn-bootstrap",
  { "Fn::Sub": "/opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --resource LaunchConfig
 --configsets wordpress_install --region ${AWS::Region}" },
  { "Fn::Sub": "/opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --stack
 ${AWS::StackName} --resource WebServer --region ${AWS::Region}" }]]
}}
```

YAML

The YAML example uses a literal block to specify the user data script.

```
UserData:
  Fn::Base64:
    !Sub |
      #!/bin/bash -xe
```

```
      yum update -y aws-cfn-bootstrap
      /opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --resource LaunchConfig --
configsets wordpress_install --region ${AWS::Region}
      /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --resource WebServerGroup --
region ${AWS::Region}
```

Fn::Sub with a Mapping

The following example uses a mapping to substitute the `${Domain}` variable with the resulting value from the `Ref` function.

JSON

```
{ "Fn::Sub": [ "www.${Domain}", { "Domain": {"Ref" : "RootDomainName" }} ]}
```

YAML

```
Name: !Sub
  - www.${Domain}
  - { Domain: !Ref RootDomainName }
```

Supported Functions

You can use the following functions in the `Fn::Sub` function:

- `Fn::Base64`
- `Fn::FindInMap`
- `Fn::GetAtt`
- `Fn::GetAZs`
- `Fn::If`
- `Fn::Join`
- `Fn::Select`
- `Ref`

`Ref`

The intrinsic function `Ref` returns the value of the specified *parameter* or *resource*.

- When you specify a parameter's logical name, it returns the value of the parameter.
- When you specify a resource's logical name, it returns a value that you can typically use to refer to that resource, such as a physical ID (p. 153).

When you are declaring a resource in a template and you need to specify another template resource by name, you can use the `Ref` to refer to that other resource. In general, `Ref` returns the name of the resource. For example, a reference to an AWS::AutoScaling::AutoScalingGroup (p. 433) returns the name of that Auto Scaling group resource.

For some resources, an identifier is returned that has another significant meaning in the context of the resource. An AWS::EC2::EIP (p. 564) resource, for instance, returns the IP address, and an AWS::EC2::Instance (p. 574) returns the instance ID.

At the bottom of this topic, there is a table that lists the values returned for many common resource types. More information about `Ref` return values for a particular resource or property can be found in the documentation for that resource or property.

Tip
You can also use `Ref` to add values to Output messages.

Declaration

JSON

```
{ "Ref" : "logicalName" }
```

YAML

Syntax for the full function name:

```
Ref: logicalName
```

Syntax for the short form:

```
!Ref logicalName
```

Parameters

logicalName

The logical name of the resource or parameter you want to dereference.

Return Value

The physical ID of the resource or the value of the parameter.

Example

The following resource declaration for an Elastic IP address needs the instance ID of an EC2 instance and uses the `Ref` function to specify the instance ID of the MyEC2Instance resource:

JSON

```
"MyEIP" : {
   "Type" : "AWS::EC2::EIP",
   "Properties" : {
      "InstanceId" : { "Ref" : "MyEC2Instance" }
   }
}
```

YAML

```
MyEIP:
  Type: "AWS::EC2::EIP"
  Properties:
    InstanceId: !Ref MyEC2Instance
```

Supported Functions

You cannot use any functions in the `Ref` function. You must specify a string that is a resource logical ID.

Resource Return Examples

This section lists sample values returned by `Ref` for particular AWS CloudFormation resources. For more information about `Ref` return values for a particular resource or property, refer to the documentation for that resource or property.

Resource Type	Reference Value	Example Return Value
AWS::ApiGateway::Account (p. 395)	API Gateway account resource ID	`mysta-accou-01234b567890example`
AWS::ApiGateway::ApiKey (p. 397)	API key	`m2m1k7sybf`
AWS::ApiGateway::Authorizer (p. 399)	Authorizer resource ID	`abcde1`
AWS::ApiGateway::ClientCertificate (p. 404)	Client certificate name	`abc123`
AWS::ApiGateway::Deployment (p. 405)	Deployment resource ID	`abc123`
AWS::ApiGateway::Method (p. 409)	Method resource ID	`mysta-metho-01234b567890example`
AWS::ApiGateway::Model (p. 412)	Model name	`myModel`
AWS::ApiGateway::Resource (p. 415)	API Gateway resource ID	`abc123`
AWS::ApiGateway::RestApi (p. 417)	Rest API resource ID	`a1bcdef2gh`
AWS::ApiGateway::Stage (p. 420)	Stage name	`MyTestStage`
AWS::ApplicationAutoScaling::ScalableTarget (p. 427)	Scalable Target ID	`service/ecsStack-MyECSCluster-AB12CDE3F4GH/ecsStack-MyECSService-AB12CDE3F4GH|ecs:service:DesiredCount|ecs`
AWS::ApplicationAutoScaling::ScalingPolicy (p. 430)	Application Auto Scaling policy Amazon Resource Name (ARN)	`arn:aws:autoscaling:us-east-1:123456789012:scalingPolicy:12ab3c4d-5 ecs/service/ecsStack-MyECSCluster-AB12CDE3F4GH/ecsStack-MyECSService-AB12CDE3F4GH:policyName/MyStepPolicy`
AWS::AutoScaling::AutoScalingGroup (p. 433)	Name	`mystack-myasgroup-NT5EUXTNTXXD`
AWS::AutoScaling::LaunchConfiguration (p. 440)	Name	`mystack-mylaunchconfig-1DDYF1E3B3I`
AWS::AutoScaling::LifecycleHook (p. 449)	Name	`mylifecyclehookname`
AWS::AutoScaling::ScalingPolicy (p. 452)	Scaling policy Amazon Resource Name (ARN)	`arn:aws:autoscaling:us-east-1:123456789012:scalingPolicy:ab12c4d5-a1b2-a1b2-a1b2-ab12c4d56789:autoScalingGroupName/myStack-AutoScalingGroup-AB12C4D5E6:policyName/myStack-myScalingPolicy-AB12C4D5E6`
AWS::AutoScaling::ScheduledAction (p. 456)	Name	`mystack-myscheduledaction-NT5EUXTNTXXD`
AWS::CertificateManager::Certificate (p. 459)	Certificate Amazon Resource Name (ARN)	`arn:aws:acm:us-east-1:123456789012:certificate/12ab3c4d-567`

Resource Type	Reference Value	Example Return Value
AWS::CloudFormation::Stack (p. 485)	Stack ID	`arn:aws:cloudformation:us-east-1:803981987763:stack/mystack-mynestedstack-sggfrhxhum7w/f449b250-b969-11e0-a185-5081d0136786`
AWS::CloudFormation::WaitCondition (p. 487)	Name	`arn:aws:cloudformation:us-east-1:803981987763:stack/mystack/c325e210-bdf2-11e0-9638-50690880c386/mywaithandle`
AWS::CloudFormation::WaitConditionHandle (p. 491)	Wait Condition Signal URL	`https://cloudformation-waitcondition-us-east-1.s3.amazonaws.com/arn%3Aaws%3Acloudformation%3Aus-east-1%3A803981987763%3Astack%2Fwaittest%2F054a33d0-bdee-11e0-8816-5081c490a786%2FmyWaitHandle?Expires=1312475488&AWSAccessKeyId=AKIAIOSFOD%3D`
AWS::CloudFront::Distribution (p. 492)	Distribution ID	`E27LVI50CSW06W`
AWS::CloudTrail::Trail (p. 493)	Trail name	`awscloudtrail-example`
AWS::CloudWatch::Alarm (p. 498)	Name	`mystack-myalarm-3AOHFRGOXR5T`
AWS::CodeBuild::Project (p. 503)	Project name	`myProjectName`
AWS::CodeCommit::Repository (p. 507)	Repository ID	`12a345b6-bbb7-4bb6-90b0-8c9577a2d2b9`
AWS::CodeDeploy::Application (p. 509)	Application name	`myapplication-a123d0d1`
AWS::CodeDeploy::DeploymentConfig (p. 510)	Deployment configuration name	`mydeploymentconfig-a123d0d1`
AWS::CodeDeploy::DeploymentGroup (p. 512)	Deployment group name	`mydeploymentgroup-a123d0d1`
AWS::CodePipeline::CustomActionType (p. 517)	Custom action type name	`mysta-MyCus-A1BCDEFGHIJ2`
AWS::CodePipeline::Pipeline (p. 520)	Pipeline name	`mysta-MyPipeline-A1BCDEFGHIJ2`
AWS::Config::ConfigRule (p. 525)	Configuration rule name	`mystack-MyConfigRule-12ABCFPXHV4OV`
AWS::Config::ConfigurationRecorder (p. 531)	Configuration recorder name	`default`
AWS::Config::DeliveryChannel (p. 533)	Delivery channel name	`default`
AWS::DataPipeline::Pipeline (p. 535)	Pipeline ID	`df-sample322HVPGK130TOD`
AWS::DirectoryService::MicrosoftAD (p. 544)	Microsoft directory ID	`d-12345ab592`
AWS::DirectoryService::SimpleAD (p. 547)	Directory ID	`d-12345ab592`
AWS::EC2::EIP (p. 564)	Elastic IP Address	`192.0.2.0`
AWS::EC2::EIPAssociation (p. 566)	Name	`mystack-myeipa-1NU3IL8LJ313N`

Resource Type	Reference Value	Example Return Value
AWS::EC2::FlowLog (p. 570)	Flow log ID	`fl-1a23b456`
AWS::EC2::Host (p. 572)	Host ID	`h-0ab123c45d67ef89`
AWS::EC2::Instance (p. 574)	Instance ID	`i-636be302`
AWS::EC2::NatGateway (p. 586)	NAT gateway ID	`nat-0a12bc456789de0fg`
AWS::EC2::PlacementGroup (p. 601)	Placement group name	`mystack-myplacementgroup-CU6107MRVLR7`
AWS::EC2::RouteTable (p. 606)	Route table ID	`rtb-12a34567`
AWS::EC2::SecurityGroup (p. 608)	Name or security group ID (for VPC security groups that are not in a default VPC)	`mystack-mysecuritygroup-QQB406M8FISX or sg-94b3a1f6`
AWS::EC2::SecurityGroupIngress (p. 616)	Name	`mysecuritygroupingress`
AWS::EC2::Subnet (p. 625)	Subnet ID	`subnet-e19f0178`
AWS::EC2::Volume (p. 633)	Volume ID	`vol-3cdd3f56`
AWS::EC2::VolumeAttachment (p. 637)	Name	`mystack-myvola-ERXHJITXMRLT`
AWS::EC2::VPC (p. 639)	VPC ID	`vpc-18ac277d`
AWS::EC2::VPCPeeringConnection (p. 649)	VPC peering connection ID	`pcx-75de3e1d`
AWS::EC2::VPCEndpoint (p. 645)	Enpoint ID	`vpce-a123d0d1`
AWS::ECR::Repository (p. 667)	Repository name	`test-repository`
AWS::ECS::Cluster (p. 669)	Name	`MyStack-MyECSCluster-NT5EUXTNTXXD`
AWS::ECS::Service (p. 671)	Service ARN	`arn:aws:ecs:us-west-2:123456789012:service/sample-webapp`
AWS::ECS::TaskDefinition (p. 675)	Task ARN	`arn:aws:ecs:us-west-2:123456789012:task/1abf0f6d-a411-4033-b8eb-a4eed3ad252a`
AWS::EFS::FileSystem (p. 679)	File system ID	`fs-47a2c22e`
AWS::EFS::MountTarget (p. 681)	Mount target ID	`fsmt-55a4413c`
AWS::ElastiCache::ReplicationGroup (p. 693)	Name	`abc12xmy3d1w3hv6`
AWS::ElastiCache::SubnetGroup (p. 706)	Name	`myCachesubnetgroup`
AWS::ElasticLoadBalancingV2::Listener (p. 721)	Listener's Amazon Resource Name (ARN)	`arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2`

Resource Type	Reference Value	Example Return Value
AWS::ElasticLoadBalancingV2::ListenerRule (p. 733)	Listener rule's Amazon Resource Name (ARN)	`arn:aws:elasticloadbalancing:us-west-2:123456789012:listener-rule/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2/9683b2d02a6cabee`
AWS::ElasticLoadBalancingV2::LoadBalancer (p. 736)	Application load balancer's Amazon Resource Name (ARN)	`arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-internal-load-balancer/50dc6c495c0c9188`
AWS::ElasticLoadBalancingV2::TargetGroup (p. 739)	Target group's Amazon Resource Name (ARN)	`arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067`
AWS::Elasticsearch::Domain (p. 745)	Domain name	`mystack-elasticsea-abc1d2efg3h4`
AWS::EMR::Cluster (p. 750)	Cluster ID	`j-1ABCD123AB1A`
AWS::EMR::InstanceGroupConfig (p. 755)	Instance group ID	`ig-ABC12DEF3456`
AWS::EMR::Step (p. 758)	Step ID	`s-1A2BC3D4EFG56`
AWS::ElasticBeanstalk::Application (p. 708)	Name	`mystack-myapplication-FM6BIXY7U8PK`
AWS::ElasticBeanstalk::ApplicationVersion (p. 709)	Name	`mystack-myapplicationversion-iy8ptveuxjly`
AWS::ElasticBeanstalk::ConfigurationTemplate (p. 711)	Name	`mystack-myconfigurationtemplate-108RPH64J195`
AWS::ElasticBeanstalk::Environment (p. 714)	Name	`mystack-myenv-LKGNQSFHO1DB`
AWS::ElasticLoadBalancing::LoadBalancer (p. 719)	Name	`mystack-myelb-1WQN7BJGDB5YQ`
AWS::Events::Rule (p. 761)	Event rule ID	`mystack-ScheduledRule-ABCDEFGHIJK`
AWS::GameLift::Alias (p. 766)	Alias ID	`myalias-a01234b56-7890-1de2-f345-g67h8i901j2k`
AWS::GameLift::Build (p. 768)	Build ID	`mybuild-a01234b56-7890-1de2-f345-g67h8i901j2k`
AWS::GameLift::Fleet (p. 771)	Fleet ID	`myfleet-a01234b56-7890-1de2-f345-g67h8i901j2k`
AWS::IAM::AccessKey (p. 775)	AccessKeyId	`AKIAIOSFODNN7EXAMPLE`
AWS::IAM::Group (p. 777)	Group name	`mystack-mygroup-1DZETITOWEKVO`
AWS::IAM::ManagedPolicy (p. 782)	Policy ARN	`arn:aws:iam::123456789012:policy/teststack-CreateTestDBPolicy-16M23YE3CS700`
AWS::IAM::User (p. 793)	User name	`mystack-myuser-1CCXAFG2H2U4D`
AWS::IoT::Certificate (p. 797)	Certificate ID	`a1234567b89c012d3e4fg567hij8k9l01mno1p23q456`

Resource Type	Reference Value	Example Return Value
AWS::IoT::Policy (p. 799)	Policy name	`MyPolicyName`
AWS::IoT::Thing (p. 803)	Thing name	`MyStack-MyThing-AB1CDEFGHIJK`
AWS::IoT::TopicRule (p. 807)	Topic rule name	`MyStackMyTopicRule12ABC3D456EFG`
AWS::Kinesis::Stream (p. 810)	Name	`mystack-mystream-1NAOH4L1RIQ7I`
AWS::KinesisFirehose::DeliveryStream (p. 811)	Delivery stream name	`mystack-deliverystream-1ABCD2EF3GHIJ`
AWS::KMS::Alias (p. 814)	Alias name	`alias/myAlias`
AWS::KMS::Key (p. 816)	Key ID	`123ab456-a4c2-44cb-95fd-b781f32fbb37`
AWS::Lambda::Alias (p. 822)	Amazon Resource Name of the AWS Lambda alias	`arn:aws:lambda:us-west-2:123456789012:function:helloworld:BETA`
AWS::Lambda::EventSourceMapping (p. 819)	Name	`MyStack-lambdaeventsourcemapping-CU6107MRVLR7`
AWS::Lambda::Function (p. 824)	Name	`MyStack-AMILookUp-NT5EUXTNTXXD`
AWS::Lambda::Version (p. 831)	Amazon Resource Name of the AWS Lambda version	`arn:aws:lambda:us-west-2:123456789012:function:helloworld:1`
AWS::Logs::Destination (p. 833)	Destination name	`TestDestination`
AWS::Logs::LogGroup (p. 835)	Name	`mystack-myLogGroup-1341JS4M96031`
AWS::Logs::LogStream (p. 837)	Log stream name	`MyAppLogStream`
AWS::OpsWorks::App (p. 843)	AWS OpsWorks Application ID	`4fee5b96-0d10-4af1-bcc5-25f92e3c6acf`
AWS::OpsWorks::Instance (p. 848)	AWS OpsWorks Instance ID	`aa2e9ae2-2b4b-491c-aeb6-8bf3ce9400fe`
AWS::OpsWorks::Layer (p. 856)	AWS OpsWorks Layer ID	`730b238b-f7c4-461d-b7c0-3feb7ef1152a`
AWS::OpsWorks::Stack (p. 862)	AWS OpsWorks Stack ID	`5c9f04e8-370e-4bd3-ae09-a4bbcc2998bb`
AWS::OpsWorks::UserProfile (p. 869)	IAM user Amazon Resource Name	`arn:aws:iam::123456789012:user/opsworksuser`
AWS::OpsWorks::Volume (p. 870)	AWS OpsWorks Volume ID	`1ab23cd4-92ff-4501-b37c-example`
AWS::RDS::DBCluster (p. 872)	Cluster name	`test-rdscluster-pdedtss0mfqr`
AWS::RDS::DBClusterParameterGroup (p. 879)	Parameter group name	`test-dbparamgroup-4l8qqx46vjby`
AWS::RDS::DBInstance (p. 881)	Name	`mystack-mydb-ea5ugmfvuaxg`
AWS::RDS::DBSecurityGroup (p. 898)	Name	`mystack-mydbsecuritygroup-1k5u5dxjb0nxs`

Resource Type	Reference Value	Example Return Value
AWS::RDS::DBSubnetGroup (p. 903)	DB subnet group name	mystack-mydbsubnetgroup-1k5u5dxjb0nxs
AWS::RDS::OptionGroup (p. 907)	Name	mystack-myoptiongroup-1qmfawfea4vmz
AWS::Redshift::Cluster (p. 911)	Name	mystack-myredshiftcluster-ranmiv3f0mad
AWS::Redshift::ClusterParameterGroup (p. 917)	Name	mysta-mypar-1AJYM1FL3WQBW
AWS::Redshift::ClusterSecurityGroup (p. 920)	Name	mystack-myredshiftclustersecuritygroup-bjy2afmhy3ee
AWS::Redshift::ClusterSubnetGroup (p. 923)	Name	mystack-myredshiftclustersubnetgroup-aq6rsdq8rp71
AWS::Route53::HealthCheck (p. 925)	Amazon Route 53 health check ID	e0a123b4-4dba-4650-935e-example
AWS::Route53::HostedZone (p. 927)	Hosted zone ID	Z23ABC4XYZL05B
AWS::S3::Bucket (p. 937)	Name	mystack-mys3bucket-1hbsmonr9mytq
AWS::SDB::Domain (p. 952)	Name	mystack-mysdbdomain-IVNAOZTDFVXL
AWS::SNS::Topic (p. 954)	Topic ARN	arn:aws:sns:us-east-1:123456789012:mystack-mytopic-NZJ5JSMVGFIE
AWS::SQS::Queue (p. 958)	Queue URL	https://sqs.us-east-1.amazonaws.com/803981987763/aa4-MyQueue-Z5NOSZO2PZE9
AWS::SSM::Document (p. 968)	SSM document name	ssm-myinstanceconfig-ABCNPH3XCAO6
AWS::StepFunctions::Activity (p. 975)	Amazon Resource Name (ARN) of the AWS Step Functions activity	arn:aws:states:us-east-1:111122223333:activity:myActivity
AWS::StepFunctions::StateMachine (p. 976)	ARN of the created Step Functions state machine	arn:aws:states:us-east-1:111122223333:stateMachine:MyStateMach ABCDEFGHIJ1K
AWS::WAF::ByteMatchSet (p. 979)	Byte match ID	aabc123a-fb4f-4fc6-becb-2b00831cadcf
AWS::WAF::IPSet (p. 983)	IP set ID	aabc123a-fb4f-4fc6-becb-2b00831cadcf
AWS::WAF::Rule (p. 986)	Rule ID	aabc123a-fb4f-4fc6-becb-2b00831cadcf
AWS::WAF::SizeConstraintSet (p. 988)	Size constraint set ID	aabc123a-fb4f-4fc6-becb-2b00831cadcf

Resource Type	Reference Value	Example Return Value
AWS::WAF::SqlInjectionMatchSet (p. 991)	SQL match set ID	`aabc123a-fb4f-4fc6-becb-2b00831cadcf`
AWS::WAF::WebACL (p. 994)	Web ACL ID	`aabc123a-fb4f-4fc6-becb-2b00831cadcf`
AWS::WAF::XssMatchSet (p. 998)	XSS match set ID	`aabc123a-fb4f-4fc6-becb-2b00831cadcf`
AWS::WorkSpaces::Workspace (p. 1002)	Workspace ID	`ws-cdd1gggh7`
Pseudo Parameter (p. 1351)	AWS::AccountId	`123456789012`
Pseudo Parameter (p. 1351)	AWS::NotificationARNs	`[arn:aws:sns:us-east-1:123456789012:MyTopic]`
Pseudo Parameter (p. 1351)	AWS::NoValue	Does not return a value.
Pseudo Parameter (p. 1351)	AWS::Region	`us-east-1`
Pseudo Parameter (p. 1351)	AWS::StackId	`arn:aws:cloudformation:us-east-1:123456789012:stack/MyStack/1c2fa620-982a-11e3-aff7-50e2416294e0`
Pseudo Parameter (p. 1351)	AWS::StackName	`MyStack`

Pseudo Parameters Reference

Pseudo Parameters are parameters that are predefined by AWS CloudFormation. You do not declare them in your template. Use them the same way as you would a parameter, as the argument for the `Ref` function.

Example

The following snippet assigns the value of the `AWS::Region` pseudo parameter to an output value:

JSON

```
"Outputs" : {
   "MyStacksRegion" : { "Value" : { "Ref" : "AWS::Region" } }
}
```

YAML

```
Outputs:
  MyStacksRegion:
    Value: !Ref "AWS::Region"
```

AWS::AccountId

Returns the AWS account ID of the account in which the stack is being created, such as `123456789012`.

AWS::NotificationARNs

Returns the list of notification Amazon Resource Names (ARNs) for the current stack.

To get a single ARN from the list, use Fn::Select (p. 1337).

JSON

```
"myASGrpOne" : {
   "Type" : "AWS::AutoScaling::AutoScalingGroup",
   "Version" : "2009-05-15",
   "Properties" : {
      "AvailabilityZones" : [ "us-east-1a" ],
      "LaunchConfigurationName" : { "Ref" : "MyLaunchConfiguration" },
      "MinSize" : "0",
      "MaxSize" : "0",
      "NotificationConfigurations" : [{
         "TopicARN" : { "Fn::Select" : [ "0", { "Ref" : "AWS::NotificationARNs" } ] },
         "NotificationTypes" : [ "autoscaling:EC2_INSTANCE_LAUNCH",
 "autoscaling:EC2_INSTANCE_LAUNCH_ERROR" ]
      }]
   }
}
```

YAML

```
myASGrpOne:
  Type: AWS::AutoScaling::AutoScalingGroup
  Version: '2009-05-15'
  Properties:
    AvailabilityZones:
    - "us-east-1a"
    LaunchConfigurationName:
      Ref: MyLaunchConfiguration
    MinSize: '0'
    MaxSize: '0'
    NotificationConfigurations:
    - TopicARN:
        Fn::Select:
        - '0'
        - Ref: AWS::NotificationARNs
      NotificationTypes:
      - autoscaling:EC2_INSTANCE_LAUNCH
      - autoscaling:EC2_INSTANCE_LAUNCH_ERROR
```

AWS::NoValue

Removes the corresponding resource property when specified as a return value in the `Fn::If` intrinsic function.

For example, you can use the `AWS::NoValue` parameter when you want to use a snapshot for an Amazon RDS DB instance only if a snapshot ID is provided. If the `UseDBSnapshot` condition evaluates to true, AWS CloudFormation uses the `DBSnapshotName` parameter value for the `DBSnapshotIdentifier` property. If the condition evaluates to false, AWS CloudFormation removes the `DBSnapshotIdentifier` property.

JSON

```
"MyDB" : {
  "Type" : "AWS::RDS::DBInstance",
  "Properties" : {
    "AllocatedStorage" : "5",
    "DBInstanceClass" : "db.m1.small",
    "Engine" : "MySQL",
    "EngineVersion" : "5.5",
    "MasterUsername" : { "Ref" : "DBUser" },
    "MasterUserPassword" : { "Ref" : "DBPassword" },
    "DBParameterGroupName" : { "Ref" : "MyRDSParamGroup" },
    "DBSnapshotIdentifier" : {
      "Fn::If" : [
        "UseDBSnapshot",
        {"Ref" : "DBSnapshotName"},
        {"Ref" : "AWS::NoValue"}
      ]
    }
  }
}
```

YAML

```
MyDB:
  Type: AWS::RDS::DBInstance
  Properties:
    AllocatedStorage: '5'
    DBInstanceClass: db.m1.small
    Engine: MySQL
    EngineVersion: '5.5'
    MasterUsername:
      Ref: DBUser
    MasterUserPassword:
      Ref: DBPassword
    DBParameterGroupName:
      Ref: MyRDSParamGroup
    DBSnapshotIdentifier:
      Fn::If:
      - UseDBSnapshot
      - Ref: DBSnapshotName
      - Ref: AWS::NoValue
```

AWS::Region

Returns a string representing the AWS Region in which the encompassing resource is being created, such as `us-west-2`.

AWS::StackId

Returns the ID of the stack as specified with the `aws cloudformation create-stack` command, such as `arn:aws:cloudformation:us-west-2:123456789012:stack/teststack/51af3dc0-da77-11e4-872e-1234567db123`.

AWS::StackName

Returns the name of the stack as specified with the `aws cloudformation create-stack` command, such as `teststack`.

CloudFormation Helper Scripts Reference

Topics

AWS CloudFormation provides a set of Python helper scripts that you can use to install software and start services on an Amazon EC2 instance that you create as part of your stack. You can call the helper scripts directly from your template. The scripts work in conjunction with resource metadata that you define in the same template. The helper scripts run on the Amazon EC2 instance as part of the stack creation process.

The helper scripts are pre-installed on the latest versions of the Amazon Linux AMI. The helper scripts are also available from the Amazon Linux yum repository for use with other UNIX/Linux AMIs.

Currently, AWS CloudFormation provides the following helpers:

- cfn-init (p. 1355): Used to retrieve and interpret the resource metadata, installing packages, creating files and starting services.
- cfn-signal (p. 1358): A simple wrapper to signal an AWS CloudFormation CreationPolicy or WaitCondition, enabling you to synchronize other resources in the stack with the application being ready.
- cfn-get-metadata (p. 1361): A wrapper script making it easy to retrieve either all metadata defined for a resource or path to a specific key or subtree of the resource metadata.
- cfn-hup (p. 1363): A daemon to check for updates to metadata and execute custom hooks when the changes are detected.

These scripts are installed by default on the latest Amazon Linux AMI in /opt/aws/bin. They are also available in the Amazon Linux AMI yum repository for previous versions of the Amazon Linux AMI as well as via RPM for other Linux/Unix distributions. You can also install the scripts on Microsoft Windows (2008 or later) by using Python for Windows.

The scripts are not executed by default. You must include calls to execute specific helper scripts.

The AWS CloudFormation helper scripts are available from the following locations:

- The latest version of the Amazon Linux AMI has the AWS CloudFormation helper scripts installed by default in /opt/aws/bin.
- The AWS helper scripts are available in the Amazon Linux AMI yum repository (the package name is aws-cfn-bootstrap) for previous versions of the Amazon Linux AMI.
- The helpers are also available in other formats:
  - https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-latest.amzn1.noarch.rpm
  - https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-latest.tar.gz to install the helper scripts via the Python easy-install tools. For Ubuntu, to complete installation, you must create a symlink: `ln -s /root/aws-cfn-bootstrap-latest/init/ubuntu/cfn-hup /etc/init.d/cfn-hup`.
  - https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-latest.zip
  - 32 bit: https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-latest.msi or 64 bit: https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-win64-latest.msi for installation on Microsoft Windows.
- The source for the scripts is available at https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-latest.src.rpm, which can be used for Linux distributions other than the Amazon Linux AMI.

The helper scripts are updated periodically. If you use the helper scripts, ensure your launched instances are using the latest version of the scripts:

- Include the following command in the `UserData` property of your template before you call the scripts. This command ensures that you get the latest version:

  ```
  yum install -y aws-cfn-bootstrap
  ```
- If you don't include the `yum install` command and you use the `cfn-init`, `cfn-signal`, or `cfn-get-metadata` scripts, then you'll need to manually update the scripts in each Amazon EC2 Linux instance using this command:

  ```
  sudo yum install -y aws-cfn-bootstrap
  ```
- If you don't include the `yum install` command and you use the `cfn-hup` script, then you'll need to manually update the script in each Amazon EC2 Linux instance using these commands:

  ```
  sudo yum install -y aws-cfn-bootstrap
  ```

  ```
  sudo /sbin/service cfn-hup restart
  ```
- If you use the source code for the scripts to work with another version of Linux or a different platform, and you have created your own certificate trust store, you'll also need to keep the trust store updated.

cfn-init

Description

The cfn-init helper script reads template metadata from the AWS::CloudFormation::Init key and acts accordingly to:

- Fetch and parse metadata from CloudFormation
- Install packages
- Write files to disk
- Enable/disable and start/stop services

> **Note**
> If you use cfn-init to update an existing file, it creates a backup copy of the original file in the same directory with a .bak extension. For example, if you update `/path/to/file_name`, the action produces two files: `/path/to/file_name.bak` contains the original file's contents and `/path/to/file_name` contains the updated contents.

For information about the template metadata, see AWS::CloudFormation::Init (p. 470).

> **Note**
> cfn-init does not require credentials, so you do not need to use the `--access-key`, `--secret-key`, `--role`, or `--credential-file` options.

Syntax

```
cfn-init --stack|-s stack.name.or.id \
         --resource|-r logical.resource.id \
         --region region
         --access-key access.key \
         --secret-key secret.key \
         --role rolename\
         --credential-file|-f credential.file \
         --configsets|-c config.sets \
         --url|-u service.url \
```

```
        --http-proxy HTTP.proxy \
        --https-proxy HTTPS.proxy \
        --verbose|-v
```

Options

Name	Description	Required
`-s, --stack`	Name of the Stack. *Type*: String *Default*: None *Example*: `-s { "Ref" : "AWS::StackName" },`	Yes
`-r, --resource`	The logical resource ID of the resource that contains the metadata. *Type*: String *Example*: `-r WebServerHost`	Yes
`--region`	The AWS CloudFormation regional endpoint to use. *Type*: String *Default*: `us-east-1` *Example*: `--region ", { "Ref" : "AWS::Region" },`	No
`--access-key`	AWS access key for an account with permission to call DescribeStackResource on CloudFormation. The credential file parameter supersedes this parameter. *Type*: String	No
`--secret-key`	AWS secret access key that corresponds to the specified AWS access key. *Type*: String	No
`--role`	The name of an IAM role that is associated with the instance. *Type*: String Condition: The credential file parameter supersedes this parameter.	No
`-f, --credential-file`	A file that contains both a secret access key and an access key. The credential file parameter supersedes the --role, --access-key, and --secret-key parameters. *Type*: String	No
`-c, --configsets`	A comma-separated list of configsets to run (in order). *Type*: String *Default*: `default`	No

Name	Description	Required
`-u, --url`	The AWS CloudFormation endpoint to use. *Type*: String	No
`--http-proxy`	An HTTP proxy (non-SSL). Use the following format: `http://user:password@host:port` *Type*: String	No
`--https-proxy`	An HTTPS proxy. Use the following format: `https://user:password@host:port` *Type*: String	No
`-v`	Verbose output. This is useful for debugging cases where cfn-init is failing to initialize. **Note** To debug initialization events, you should turn DisableRollback on. You can do this by using the CloudFormation console, selecting *Show Advanced Options*, and then setting "Rollback on failure" to "No". You can then SSH into the console and read the logs at /var/log/cfn-init.log.	No

Example

Amazon Linux Example

The following snippet shows the `UserData` property of an EC2 instance, which runs the `InstallAndRun` configset that is associated with the `WebServerInstance` resource.

For a complete example template, see Deploying Applications on Amazon EC2 with AWS CloudFormation (p. 212).

JSON

```
"UserData" : { "Fn::Base64" :
  { "Fn::Join" : ["", [
     "#!/bin/bash -xe\n",
     "# Install the files and packages from the metadata\n",
     "/opt/aws/bin/cfn-init -v ",
     "         --stack ", { "Ref" : "AWS::StackName" },
     "         --resource WebServerInstance ",
     "         --configsets InstallAndRun ",
     "         --region ", { "Ref" : "AWS::Region" }, "\n"
  ]]}
}
```

YAML

```
UserData:
  "Fn::Base64":
    !Sub |
      #!/bin/bash -xe
```

```
      /opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --resource LaunchConfig --
configsets wordpress_install --region ${AWS::Region}
```

cfn-signal

Description

The cfn-signal helper script signals AWS CloudFormation to indicate whether Amazon EC2 instances have been successfully created or updated. If you install and configure software applications on instances, you can signal AWS CloudFormation when those software applications are ready.

You use the cfn-signal script in conjunction with a CreationPolicy (p. 1293) or an Auto Scaling group with a `WaitOnResourceSignals` (p. 1303) update policy. When AWS CloudFormation creates or updates resources with those policies, it suspends work on the stack until the resource receives the requisite number of signals or until the timeout period is exceeded. For each valid signal that AWS CloudFormation receives, AWS CloudFormation publishes the signals to the stack events so that you track each signal. For a walkthrough that uses a creation policy and cfn-signal, see Deploying Applications on Amazon EC2 with AWS CloudFormation (p. 212).

Syntax for Resource Signaling (Recommended)

If you want to signal AWS CloudFormation resources, use the following syntax.

> **Note**
> cfn-signal does not require credentials, so you do not need to use the `--access-key`, `--secret-key`, `--role`, or `--credential-file` options.

```
cfn-signal --success|-s signal.to.send \
        --access-key access.key \
        --credential-file|-f credential.file \
        --exit-code|-e exit.code \
        --http-proxy HTTP.proxy \
        --https-proxy HTTPS.proxy \
        --id|-i unique.id \
        --region AWS.region \
        --resource resource.logical.ID \
        --role IAM.role.name \
        --secret-key secret.key \
        --stack stack.name.or.stack.ID \
        --url AWS CloudFormation.endpoint
```

Syntax for Use with Wait Condition Handle

If you want to signal a wait condition handle, use the following syntax.

```
cfn-signal --success|-s signal.to.send \
        --reason|-r resource.status.reason \
        --data|-d data \
        --id|-i unique.id \
        --exit-code|-e exit.code \
        waitconditionhandle.url
```

Options

The options that you can use depend on whether you're signaling a creation policy or a wait condition handle. Some options that apply to a creation policy might not apply to a wait condition handle.

Name	Description	Required
`--access-key` (resource signaling only)	AWS access key for an account with permission to call the AWS CloudFormation `SignalResource` API. The credential file parameter supersedes this parameter. *Type*: String	No
`-d, --data` (wait condition handle only)	Data to send back with the `waitConditionHandle`. Defaults to blank. *Type*: String *Default*: blank	No
`-e, --exit-code`	The error code from a process that can be used to determine success or failure. If specified, the `--success` option is ignored. *Type*: String *Examples*: `-e $?` (for Linux), `-e %ERRORLEVEL%` (for Windows cmd.exe), and `-e $lastexitcode` (for Windows PowerShell).	No
`-f, --credential-file` (resource signaling only)	A file that contains both a secret access key and an access key. The credential file parameter supersedes the --role, --access-key, and --secret-key parameters. *Type*: String	No
`--http-proxy`	An HTTP proxy (non-SSL). Use the following format: `http://user:password@host:port` *Type*: String	No
`--https-proxy`	An HTTPS proxy. Use the following format: `https://user:password@host:port` *Type*: String	No
`-i, --id`	The unique ID to send. *Type*: String *Default*: The ID of the Amazon EC2 instance. If the ID cannot be resolved, the machine's Fully Qualified Domain Name (FQDN) is returned.	No
`-r, --reason` (wait condition handle only)	A status reason for the resource event (currently only used on failure) - defaults to 'Configuration failed' if success is false. *Type*: String	No
`--region` (resource signaling only)	The AWS CloudFormation regional endpoint to use. *Type*: String *Default*: `us-east-1`	No

Name	Description	Required
`--resource` (resource signaling only)	The logical ID (p. 153) of the resource that contains the creations policy you want to signal. *Type*: String	Yes
`--role` (resource signaling only)	The name of an IAM role that is associated with the instance. *Type*: String Condition: The credential file parameter supersedes this parameter.	No
`-s, --success`	if true, signal SUCCESS, else FAILURE. *Type*: Boolean *Default*: `true`	No
`--secret-key` (resource signaling only)	AWS secret access key that corresponds to the specified AWS access key. *Type*: String	No
`--stack` (resource signaling only)	The stack name or stack ID that contains the resource you want to signal. *Type*: String	Yes
`-u, --url` (resource signaling only)	The AWS CloudFormation endpoint to use. *Type*: String	No
`waitconditionhandle.url` (wait condition handle only)	A presigned URL that you can use to signal success or failure to an associated `WaitCondition` *Type*: String	Yes

Example

Amazon Linux Example

A common usage pattern is to use cfn-init and cfn-signal together. The cfn-signal call uses the return status of the call to cfn-init (using the $? shell construct). If the application fails to install, the instance will fail to create and the stack will rollback. For Windows stacks, see Bootstrapping AWS CloudFormation Windows Stacks (p. 122).

JSON

```
"MyInstance": {
  "Type": "AWS::EC2::Instance",
  "Metadata": {
    "AWS::CloudFormation::Init" : {
      cfn-init information
    }
  },
```

```
  "Properties": {
    "ImageId" : "ami-12345678",
    "UserData" : {
      "Fn::Base64" : {
        "Fn::Join" : ["", [
          "#!/bin/bash\n",
          "/opt/aws/bin/cfn-init -s ", { "Ref" : "AWS::StackName" },
          "         -r MyInstance ",
          "         --region ", { "Ref" : "AWS::Region" },
          "\n",
          "/opt/aws/bin/cfn-signal -e $? --stack ", { "Ref" : "AWS::StackName" },
          "         --resource MyInstance \n"
        ] ]
      }
    }
  },
  "CreationPolicy" : {
    "ResourceSignal" : {
      "Timeout" : "PT5M"
    }
  }
}
```

YAML

```
MyInstance:
  Type: AWS::EC2::Instance
  Metadata:
    AWS::CloudFormation::Init:
      cfn-init information
  Properties:
    ImageId: ami-12345678
    UserData:
      "Fn::Base64":
        !Sub |
          #!/bin/bash -xe
          yum update -y aws-cfn-bootstrap
          /opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --resource LaunchConfig --
configsets wordpress_install --region ${AWS::Region}
          /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --resource WebServerGroup
 --region ${AWS::Region}
  CreationPolicy:
    ResourceSignal:
      Timeout: PT5M
```

Examples

Several AWS CloudFormation sample templates use cfn-signal, including the following templates.

- LAMP: Single EC2 Instance with local MySQL database
- WordPress: Single EC2 Instance with local MySQL database

cfn-get-metadata

Description

You can use the cfn-get-metadata helper script to fetch a metadata block from CloudFormation and print it to standard out. You can also print a sub-tree of the metadata block if you specify a key. However, only top-level keys are supported.

Note

cfn-get-metadata does not require credentials, so you do not need to use the `--access-key`, `--secret-key`, or `--credential-file` options.

Syntax

```
cfn-get-metadata --access-key access.key \
                 --secret-key secret.key \
                 --credential-file|f credential.file \
                 --key|k key \
                 --stack|-s stack.name.or.id \
                 --resource|-r logical.resource.id \
                 --url|-u service.url \
                 --region region
```

Options

Name	Description	Required
`-k, --key`	For a key-value pair, returns the name of the key for the value that you specified. *Type*: String *Example*: For `{ "SampleKey1" : "Key1", "SampleKey2" : "Key2" }`, `cfn-get-metadata -k Key2` returns `SampleKey2`.	No
`-s, --stack`	Name of the Stack. *Type*: String *Default*: None *Example*: `-s { "Ref" : "AWS::StackName" },`	Yes
`-r, --resource`	The logical resource ID of the resource that contains the metadata. *Type*: String *Example*: `-r WebServerHost`	Yes
`--region`	The region to derive the CloudFormation URL from. *Type*: String *Default*: None *Example*: `--region ", { "Ref" : "AWS::Region" },`	No
`--access-key`	AWS Access Key for an account with permission to call DescribeStackResource on CloudFormation. *Type*: String Condition: The credential file parameter supersedes this parameter.	Conditional

Name	Description	Required
`--secret-key`	AWS Secret Key that corresponds to the specified AWS Access Key. *Type*: String Condition: The credential file parameter supersedes this parameter.	Conditional
`-f, --credential-file`	A file that contains both a secret key and an access key. *Type*: String Condition: The credential file parameter supersedes the --access-key and --secret-key parameters.	Conditional

cfn-hup

Description

The cfn-hup helper is a daemon that detects changes in resource metadata and runs user-specified actions when a change is detected. This allows you to make configuration updates on your running Amazon EC2 instances through the UpdateStack API action.

Syntax

```
cfn-hup --config|-c config.dir \
        --no-daemon \
        --verbose|-v
```

Options

Name	Description	Required
`--config|-c config.dir`	Specifies the path that the cfn-hup script looks for the `cfn-hup.conf` and the `hooks.d` directories. On Windows, the default path is *`system_drive`*`\cfn`. On Linux, the default path is `/etc/cfn`.	No
`--no-daemon`	Specify this option to run the cfn-hup script once and exit.	No
`-v, --verbose`	Specify this option to use verbose mode.	No

cfn-hup.conf Configuration File

The cfn-hup.conf file stores the name of the stack and the AWS credentials that the cfn-hup daemon targets. The cfn-hup.conf file uses the following format:

```
[main]
stack=<stack-name-or-id>
```

Name	Description	Required
`stack`	A stack name or ID. *Type*: String	Yes
`credential-file`	An owner-only credential file, in the same format used for the command line tools. Example: **Note** cfn-hup does not require credentials, so you do not need to use the `--credential-file` option.	No
`region`	The name of the AWS region containing the stack. *Example*: `us-east-1`	No
`interval`	The interval used to check for changes to the resource metadata in minutes Type: Number *Default*: `15`	No
`verbose`	Specifies whether to use verbose logging. Type: Boolean *Default*: `false`	No

hooks.conf Configuration File

The user actions that the cfn-hup daemon calls periodically are defined in the hooks.conf configuration file. The hooks.conf file uses the following *format*:

```
[hookname]
triggers=post.add or post.update or post.remove
path=Resources.<logicalResourceId> (.Metadata or .PhysicalResourceId)
(.<optionalMetadatapath>)
action=<arbitrary shell command>
runas=<runas user>
```

When the action is run, it is run in a copy of the current environment (that cfn-hup is in), with CFN_OLD_METADATA set to the previous value of path, and CFN_NEW_METADATA set to the current value.

The hooks configuration file is loaded at cfn-hup daemon startup only, so new hooks will require the daemon to be restarted. A cache of previous metadata values is stored at /var/lib/cfn-hup/data/metadata_db (not human readable)—you can delete this cache to force cfn-hup to run all post.add actions again.

Name	Description	Required
`hookname`	A unique name for this hook *Type*: String	Yes

Name	Description	Required
`triggers`	A comma-delimited list of conditions to detect. *Valid values*: `post.add`, `post.update`, or `post.remove` *Example*: `post.add, post.update`	Yes
`path`	The path to the metadata object. Supports an arbitrarily deep path within the Metadata block. **Path format options** • Resources.*<LogicalResourceId>*—monitor the last updated time of the resource, triggering on any change to the resource. • Resources.*<LogicalResourceId>*.PhysicalResourceId—monitor the physical ID of the resource, triggering only when the associated resource identity changes (such as a new EC2 instance). • Resources.*<LogicalResourceId>*.Metadata(*.optional path*)—monitor the metadata of a resource for changes (a metadata subpath may be specified to an arbitrarily deep level to monitor specific values).	Yes
`action`	An arbitrary shell command that is run as given.	Yes
`runas`	A user to run the commands as. Cfn-hup uses the su command to switch to the user.	Yes

hooks.d Directory

To support composition of several applications deploying change notification hooks, cfn-hup supports a directory named hooks.d that is located in the hooks configuration directory. You can place one or more additional hooks configuration files in the hooks.d directory. The additional hooks files must use the same layout as the hooks.conf file.

The cfn-hup daemon parses and loads each file in this directory. If any hooks in the hooks.d directory have the same name as a hook in hooks.conf, the hooks will be merged (meaning hooks.d will overwrite hooks.conf for any values that both files specify).

Example

In the following template snippet, AWS CloudFormation triggers the `cfn-auto-reloader.conf` hooks file when you change the `AWS::CloudFormation::Init` resource that is associated with the `LaunchConfig` resource.

JSON

```
...
    "LaunchConfig": {
      "Type" : "AWS::AutoScaling::LaunchConfiguration",
      "Metadata" : {
        "QBVersion": {"Ref": "paramQBVersion"},
        "AWS::CloudFormation::Init" : {
...
```

```
              "/etc/cfn/hooks.d/cfn-auto-reloader.conf": {
                "content": { "Fn::Join": [ "", [
                  "[cfn-auto-reloader-hook]\n",
                  "triggers=post.update\n",
                  "path=Resources.LaunchConfig.Metadata.AWS::CloudFormation::Init\n",
                  "action=/opt/aws/bin/cfn-init -v ",
                          "         --stack ", { "Ref" : "AWS::StackName" },
                          "         --resource LaunchConfig ",
                          "         --configsets wordpress_install ",
                          "         --region ", { "Ref" : "AWS::Region" }, "\n"
                ]]},
                "mode"  : "000400",
                "owner" : "root",
                "group" : "root"
              }
...
```

YAML

```
...
  LaunchConfig:
    Type: "AWS::AutoScaling::LaunchConfiguration"
    Metadata:
      QBVersion: !Ref paramQBVersion
      AWS::CloudFormation::Init:
...
            /etc/cfn/hooks.d/cfn-auto-reloader.conf:
              content: !Sub |
                [cfn-auto-reloader-hook]
                triggers=post.update
                path=Resources.LaunchConfig.Metadata.AWS::CloudFormation::Init
                action=/opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --resource
 LaunchConfig --configsets wordpress_install --region ${AWS::Region}
              mode: "000400"
              owner: "root"
              group: "root"
...
```

Additional Example

For a sample template, see Deploying Applications on Amazon EC2 with AWS CloudFormation (p. 212).

Sample Templates

AWS CloudFormation sample templates demonstrate how you can create templates for various uses. For example, one sample template describes a load-balancing, auto scaling WordPress blog in an Amazon VPC. We recommend that you use these sample templates as a starting point for creating your own templates and not to launch production-level environments.

To view the sample templates, go to http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/cfn-sample-templates.html

> **Note**
> The AWS Quick Starts use AWS CloudFormation templates to automate software deployments, such as a Chef Server or MongoDB, on AWS. You can use these templates to learn how to deploy your own solution on AWS. For more information, see AWS Quick Start Reference Deployments.

Troubleshooting AWS CloudFormation

When you use AWS CloudFormation, you might encounter issues when you create, update, or delete AWS CloudFormation stacks. The following sections can help you troubleshoot some common issues that you might encounter.

For general questions about AWS CloudFormation, see the AWS CloudFormation FAQs. You can also search for answers and post questions in the AWS CloudFormation forums.

Topics

Troubleshooting Guide

If AWS CloudFormation fails to create, update, or delete your stack, you can view error messages or logs to help you learn more about the issue. The following tasks describe general methods for troubleshooting a AWS CloudFormation issue. For information about specific errors and solutions, see the Troubleshooting Errors (p. 1369) section.

- Use the AWS CloudFormation console to view the status of your stack. In the console, you can view a list of stack events while your stack is being created, updated, or deleted. From this list, find the failure event and then view the status reason for that event. The status reason might contain an error message from AWS CloudFormation or from a particular service that can help you troubleshoot your problem. For more information about viewing stack events, see Viewing Stack Data and Resources (p. 78).
- For Amazon EC2 issues, view the cloud-init and cfn logs. These logs are published on the Amazon EC2 instance in the `/var/log/` directory. These logs capture processes and command outputs while AWS CloudFormation is setting up your instance. For Windows, view the EC2Configure service and cfn logs in `%ProgramFiles%\Amazon\EC2ConfigService` and `C:\cfn\log`.

  You can also configure your AWS CloudFormation template so that the logs are published to Amazon CloudWatch, which displays logs in the AWS Management Console so you don't have to connect to your Amazon EC2 instance. For more information, see View CloudFormation Logs in the Console in the Application Management Blog.

Troubleshooting Errors

When you come across the following errors with your AWS CloudFormation stack, you can use the following solutions to help you find the source of the problems and fix them.

Topics

Delete Stack Fails

To resolve this situation, try the following:

- Some resources must be empty before they can be deleted. For example, you must delete all objects in an Amazon S3 bucket or remove all instances in an Amazon EC2 security group before you can delete the bucket or security group.
- Ensure that you have the necessary IAM permissions to delete the resources in the stack. In addition to AWS CloudFormation permissions, you must be allowed to use the underlying services, such as Amazon S3 or Amazon EC2.
- When stacks are in the `DELETE_FAILED` state because AWS CloudFormation couldn't delete a resource, rerun the deletion with the `RetainResources` parameter and specify the resource that AWS CloudFormation can't delete. AWS CloudFormation deletes the stack without deleting the retained resource. Retaining resources is useful when you can't delete a resource, such as an S3 bucket that contains objects that you want to keep, but you still want to delete the stack.

  After you delete the stack, you can manually delete retained resources by using their associated AWS service.
- For all other issues, if you have AWS Premium Support, you can create a Technical Support case. See Contacting Support (p. 1373).

Dependency Error

To resolve a dependency error, add a `DependsOn` attribute to resources that depend on other resources in your template. In some cases, you must explicitly declare dependencies so that AWS CloudFormation can create or delete resources in the correct order. For example, if you create an Elastic IP and a VPC with an

Internet gateway in the same stack, the Elastic IP must depend on the Internet gateway attachment. For additional information, see DependsOn Attribute (p. 1298).

Error Parsing Parameter When Passing a List

When you use the AWS Command Line Interface or AWS CloudFormation to pass in a list, add the escape character (\) before each comma. The following sample shows how you specify an input parameter when using the CLI.

```
ParameterKey=CIDR,ParameterValue='10.10.0.0/16\,10.10.0.0/24\,10.10.1.0/24'
```

Insufficient IAM Permissions

When you work with an AWS CloudFormation stack, you not only need permissions to use AWS CloudFormation, you must also have permission to use the underlying services that are described in your template. For example, if you're creating an Amazon S3 bucket or starting an Amazon EC2 instance, you need permissions to Amazon S3 or Amazon EC2. Review your IAM policy and verify that you have the necessary permissions before you work with AWS CloudFormation stacks. For more information see, Controlling Access with AWS Identity and Access Management (p. 8).

Invalid Value or Unsupported Resource Property

When you create or update an AWS CloudFormation stack, your stack can fail due to invalid input parameters, unsupported resource property names, or unsupported resource property values. For input parameters, verify that the resource exists. For example, when you specify an Amazon EC2 key pair or VPC ID, the resource must exist in your account and in the region in which you are creating or updating your stack. You can use AWS-specific parameter types (p. 133) to ensure that you use valid values.

For resource property names and values, update your template to use valid names and values. For a list of all the resources and their property names, see AWS Resource Types Reference (p. 390).

Limit Exceeded

Verify that you didn't reach a resource limit. For example, the default number Amazon EC2 instances that you can launch is 20. If try to create more Amazon EC2 instances than your account limit, the instance creation fails and you receive the error `Status=start_failed`. To view the default AWS limits by service, see AWS Service Limits in the *AWS General Reference*.

For AWS CloudFormation limits and tweaking strategies, see AWS CloudFormation Limits (p. 16).

Also, during an update, if a resource is replaced, AWS CloudFormation creates new resource before it deletes the old one. This replacement might put your account over the resource limit, which would cause your update to fail. You can delete excess resources or request a limit increase.

Nested Stacks are Stuck in `UPDATE_COMPLETE_CLEANUP_IN_PROGRESS`, `UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS`, or `UPDATE_ROLLBACK_IN_PROGRESS`

A nested stack failed to roll back. Because of potential resource dependencies between nested stacks, AWS CloudFormation doesn't start cleaning up nested stack resources until all nested stacks have been updated or have rolled back. When a nested stack fails to roll back, AWS CloudFormation cancels all operations, regardless of the state that the other nested stacks are in. A nested stack that completed updating or rolling back but did not receive a signal from AWS CloudFormation to start cleaning up because another nested failed to roll back is in an `UPDATE_COMPLETE_CLEANUP_IN_PROGRESS` or `UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS` state. A nested stack that failed to update but did not receive a signal to start rolling back is in an `UPDATE_ROLLBACK_IN_PROGRESS` state.

A nested stack might fail to roll back because of changes that were made outside of AWS CloudFormation, when the stack template doesn't accurately reflect the state of the stack. A nested stack might also fail if an Auto Scaling group in a nested stack had an insufficient resource signal timeout period when the group was created or updated.

To fix the stack, contact AWS customer support (p. 1373).

No Updates to Perform

To update an AWS CloudFormation stack, you must submit template or parameter value changes to AWS CloudFormation. However, AWS CloudFormation won't recognize some template changes as an update, such as changes to a deletion policy, update policy, condition declaration, or output declaration. If you need to make such changes without making any other change, you can add or modify a metadata (p. 1303) attribute for any of your resources.

For more information about modifying templates during an update, see Modifying a Stack Template (p. 90).

Resource Failed to Stabilize During a Create, Update, or Delete Stack Operation

A resource did not respond because the operation exceeded the AWS CloudFormation timeout period or an AWS service was interrupted. For service interruptions, check that the relevant AWS service is running, and then retry the stack operation.

If the AWS services have been running successfully, check if your stack contains one of the following resources:

- `AWS::AutoScaling::AutoScalingGroup` for create, update, and delete operations
- `AWS::CertificateManager::Certificate` for create operations
- `AWS::CloudFormation::Stack` for create, update, and delete operations
- `AWS::ElasticSearch::Domain` for update operations
- `AWS::RDS::DBCluster` for create and update operations
- `AWS::RDS::DBInstance` for create, update, and delete operations
- `AWS::Redshift::Cluster` for update operations

Operations for these resources might take longer than the default timeout period. The timeout period depends on the resource and credentials that you use. To extend the timeout period, specify a service role (p. 13) when you perform the stack operation. If you're already using a service role, or if your stack contains a resource that isn't listed, contact AWS customer support (p. 1373).

If your stack is in the `UPDATE_ROLLBACK_FAILED` state, see Update Rollback Failed (p. 1372).

Security Group Does Not Exist in VPC

Verify that the security group exists in the VPC that you specified. If the security group exists, ensure that you specify the security group ID and not the security group name. For example, the `AWS::EC2::SecurityGroupIngress` resource has a `SourceSecurityGroupName` and

`SourceSecurityGroupId` properties. For VPC security groups, you must use the `SourceSecurityGroupId` property and specify the security group ID.

Update Rollback Failed

A dependent resource cannot return to its original state, causing the rollback to fail (`UPDATE_ROLLBACK_FAILED` state). For example, you might have a stack that is rolling back to an old database instance that was deleted outside of AWS CloudFormation. Because AWS CloudFormation doesn't know the database was deleted, it assumes that the database instance still exists and attempts to roll back to it, causing the update rollback to fail.

Depending on the cause of the failure, you can manually fix the error and continue the rollback. By continuing the rollback, you can return your stack to a working state (the `UPDATE_ROLLBACK_COMPLETE` state), and then try to update the stack again. The following list describes solutions to common errors that cause update rollback failures:

- Failed to receive the required number of signals

  Use the signal-resource command to manually send the required number of successful signals to the resource that is waiting for them, and then continue rolling back the update. For example, during an update rollback, instances in an Auto Scaling group might fail to signal success within the specified timeout duration. Manually send success signals to the Auto Scaling group. When you continue the update rollback, AWS CloudFormation sees your signals and proceeds with the rollback.
- Changes to a resource were made outside of AWS CloudFormation

  Manually sync resources so that they match the original stack's template, and then continue rolling back the update. For example, if you manually deleted a resource that AWS CloudFormation is attempting to roll back to, you must manually create that resource with the same name and properties it had in the original stack.
- Insufficient permissions

  Check that you have sufficient IAM permissions to modify resources, and then continue the update rollback. For example, your IAM policy might allow you to create an S3 bucket, but not modify the bucket. Add the modify actions to your policy.
- Invalid security token

  AWS CloudFormation requires a new set of credentials. No change is required. Continue rolling back the update, which refreshes the credentials.
- Limitation error

  Delete resources that you don't need or request a limit increase, and then continue rolling back the update. For example, if your account limit for the number of EC2 instances is 20 and the update rollback exceeds that limit, it will fail.
- Resource did not stabilize

  A resource did not respond because the operation might have exceeded the AWS CloudFormation timeout period or an AWS service might have been interrupted. No change is required. After the resource operation is complete or the AWS service is back in operation, continue rolling back the update.

To continue rolling back an update, you can use the AWS CloudFormation console or AWS command line interface (CLI). For more information, see Continue Rolling Back an Update (p. 119).

If none of these solutions work, you can skip the resources that AWS CloudFormation can't successfully roll back. For more information, see the `ResourcesToSkip` parameter for the ContinueUpdateRollback action in the *AWS CloudFormation API Reference*. AWS CloudFormation sets the status of the specified resources to UPDATE_COMPLETE and continues to roll back the stack. After the rollback is complete, the state of the skipped resources will be inconsistent with the state of the resources in the stack template. Before you perform another stack update, you must modify the resources or update the stack to be consistent with each other. If you don't, subsequent stack updates might fail and make your stack unrecoverable.

Wait Condition Didn't Receive the Required Number of Signals from an Amazon EC2 Instance

To resolve this situation, try the following:

- Ensure that the AMI you're using has the AWS CloudFormation helper scripts installed. If the AMI doesn't include the helper scripts, you can also download them to your instance. For more information, see CloudFormation Helper Scripts Reference (p. 1354).
- Verify that the `cfn-signal` command was successfully run on the instance. You can view logs, such as `/var/log/cloud-init.log` or `/var/log/cfn-init.log`, to help you debug the instance launch. You can retrieve the logs by logging in to your instance, but you must disable rollback on failure (p. 76) or else AWS CloudFormation deletes the instance after your stack fails to create. You can also publish the logs to Amazon CloudWatch. For Windows, you can view cfn logs in `C:\cfn\log` and EC2Config service logs in `%ProgramFiles%\Amazon\EC2ConfigService`.
- Verify that the instance has a connection to the Internet. If the instance is in a VPC, the instance should be able to connect to the Internet through a NAT device if it's is in a private subnet or through an Internet gateway if it's in a public subnet. To test the instance's Internet connection, try to access a public web page, such as `http://aws.amazon.com`. For example, you can run the following command on the instance. It should return an HTTP 200 status code.

```
curl -I https://aws.amazon.com
```

  For information about configuring a NAT device, see NAT in the *Amazon VPC User Guide*.

Contacting Support

If you have AWS Premium Support, you can create a technical support case at https://console.aws.amazon.com/support/home#/. Before you contact support, gather the following information:

- The ID of the stack. You can find the stack ID in the **Overview** tab of the AWS CloudFormation console. For more information, see Viewing Stack Data and Resources (p. 78).

  Important
  Do not make changes to the stack outside of AWS CloudFormation. Making changes to your stack outside of AWS CloudFormation might put your stack in an unrecoverable state.
- Any stack error messages. For information about viewing stack error messages, see the Troubleshooting Guide (p. 1368) section.
- For Amazon EC2 issues, gather the cloud-init and cfn logs. These logs are published on the Amazon EC2 instance in the `/var/log/` directory. These logs capture processes and command outputs while your instance is setting up. For Windows, gather the EC2Configure service and cfn logs in `%ProgramFiles%\Amazon\EC2ConfigService` and `C:\cfn\log`.

You can also search for answers and post questions in the AWS CloudFormation forums.

Release History

The following table describes the important changes to the documentation since the preceding release of AWS CloudFormation.

Change	Release Date	Description	API Version
Edit templates in YAML and JSON using AWS CloudFormation Designer	April 6, 2017	When you create AWS CloudFormation templates using Designer, you can now edit your template in both YAML and JSON in the integrated editor. You can also convert JSON templates to YAML and vice-versa, depending on your preferred template authoring language. For more information, see What Is AWS CloudFormation Designer? (p. 157).	2010-05-15
New resource	April 6, 2017	AWS::SSM::Parameter (p. 972) Use the `AWS::SSM::Parameter` resource to create an SSM parameter in Parameter Store.	2010-05-15
`AWS::Include` transform	March 28, 2017	Use the `AWS::Include` transform to reference reusable snippets stored in an Amazon S3 bucket. For more information, see AWS::Include Transform (p. 149).	2010-05-15
Peer your Amazon VPC with another account	March 28, 2017	You can now use AWS CloudFormation to peer your Amazon VPC with a VPC in another AWS account. For more information, see Walkthrough: Peer with an Amazon VPC in Another AWS Account (p. 195).	2010-05-15
New resource	March 28, 2017	AWS::ApiGateway::UsagePlanKey (p. 425) Use the `AWS::ApiGateway::UsagePlanKey` resource to associate a usage plan key and determine which users the usage plan is applied to.	2010-05-15
Updated resources	March 28, 2017	AWS::EC2::VPCPeeringConnection (p. 649) Use the `PeerOwnerId` property and the `PeerRoleArn` property to peer with a VPC in another AWS account. For more information, see Walkthrough: Peer with an Amazon VPC in Another AWS Account (p. 195).	2010-05-15

Change	Release Date	Description	API Version
		AWS::IAM::InstanceProfile (p. 779) Use the `InstanceProfileName` property to configure an instance profile. AWS::Lambda::Function (p. 824) Use the `DeadLetterConfig` property to configure how AWS Lambda handles events that it can't process. Node.js v0.10 is no longer supported for the `Runtime` property. AWS::Route53::HealthCheck (p. 925) There are seven new resource subproperty types for the Amazon Route 53 HealthCheckConfig (p. 1242) `HealthCheckConfig` property: `AlarmIdentifier`, `ChildHealthChecks`, `EnableSNI`, `HealthThreshold`, `InsufficientDataHealthStatus`, `Inverted`, and `MeasureLatency`. AWS::SQS::Queue (p. 958) Use the `ContentBasedDeduplication` and `FifoQueue` properties to create First-In-First-Out (FIFO) Amazon Simple Queue Service queues. AWS::S3::Bucket (p. 937) You can now specify IPv6 domain names for your Amazon S3 buckets.	
New resources	February 10, 2017	AWS::StepFunctions::Activity (p. 975) Use the `AWS::StepFunctions::Activity` resource to create an AWS Step Functions activity. AWS::StepFunctions::StateMachine (p. 976) Use the `AWS::StepFunctions::StateMachine` resource to create a Step Functions state machine.	2010-05-15
New intrinsic function	January 17, 2017	Use the `Fn::Split` function to split a string into a list of string values. For more information, see Fn::Split (p. 1339).	2010-05-15
Console support for listing imports	January 17, 2017	Use the AWS CloudFormation console to see all of the stacks that are importing an exported output value. For more information, see Listing Stacks That Import an Exported Output Value (p. 121).	2010-05-15

Change	Release Date	Description	API Version
Updated resources	January 17, 2017	AWS::AutoScaling::AutoScalingGroup (p. 433) The `LoadBalancerNames` property can be updated without replacing the Auto Scaling group. AWS::ECS::TaskDefinition (p. 675) Added the `NetworkMode` and `MemoryReservation` properties. AWS::RDS::DBCluster (p. 872) AWS CloudFormation supports updates to the `Tags` property. AWS::RDS::DBInstance (p. 881) Added the `Timezone` property. AWS IoT Firehose Action (p. 1192) Added the `Separator` property. AWS::OpsWorks::Instance (p. 848) Added the `PublicIp` attribute for the `Fn::GetAtt` intrinsic function.	2010-05-15
New resources	December 01, 2016	AWS::CodeBuild::Project (p. 503) Use the `AWS::CodeBuild::Project` resource to create an AWS CodeBuild project that defines how AWS CodeBuild builds your source code. AWS::SSM::Association (p. 966) Use the `AWS::SSM::Association` resource to associate an Amazon EC2 Systems Manager document with EC2 instances. AWS::EC2::SubnetCidrBlock (p. 628) Use the `AWS::EC2::SubnetCidrBlock` resource to associate a single IPv6 CIDR block with an Amazon VPC subnet. AWS::EC2::VPCCidrBlock (p. 642) Use the `AWS::EC2::VPCCidrBlock` resource to associate a single Amazon-provided IPv6 CIDR block with an Amazon VPC VPC.	2010-05-15

Change	Release Date	Description	API Version
Updated resources for IPv6 support	December 01, 2016	AWS::EC2::Instance (p. 574) Added the `Ipv6AddressCount` and `Ipv6Addresses` properties. AWS::EC2::NetworkAclEntry (p. 590) Added the `Ipv6CidrBlock` property. AWS::EC2::NetworkInterface (p. 594) Added the `Ipv6AddressCount` and `Ipv6Addresses` properties. AWS::EC2::Route (p. 602) Added the `DestinationIpv6CidrBlock` property. AWS::EC2::SecurityGroupEgress (p. 611) Added the `CidrIpv6` property. AWS::EC2::SecurityGroupIngress (p. 616) Added the `CidrIpv6` property. AWS::EC2::SpotFleet (p. 622) Added the `Ipv6AddressCount` and `Ipv6Addresses` properties for the launch specification network interfaces. AWS::EC2::Subnet (p. 625) Added the `Ipv6CidrBlocks` attribute for the `Fn::GetAtt` function. AWS::EC2::VPC (p. 639) Added the `Ipv6CidrBlocks` attribute for the `Fn::GetAtt` function. AWS::SSM::Document (p. 968) Added the `DocumentType` property.	2010-05-15
Resource specification	November 22, 2016	Use the AWS CloudFormation resource specification to builds tools that help you create AWS CloudFormation templates. The specification is a machine-readable, JSON-formatted text file. For more information, see AWS CloudFormation Resource Specification (p. 1283).	2010-05-15
New resources	November 22, 2016	AWS::OpsWorks::UserProfile (p. 869) Use the `AWS::OpsWorks::UserProfile` resource to configure SSH access for users who require access to instances in an AWS OpsWorks stack. AWS::OpsWorks::Volume (p. 870) Use the `AWS::OpsWorks::Volume` resource to register an Amazon Elastic Block Store volume with an AWS OpsWorks stack.	2010-05-15

Change	Release Date	Description	API Version
Updated resources	November 22, 2016	AWS::OpsWorks::App (p. 843) Added the `DataSources` property. AWS::OpsWorks::Instance (p. 848) Added the `BlockDeviceMappings`, `AgentVersion`, `ElasticIps`, `Hostname`, `Tenancy`, and `Volumes` properties. AWS::OpsWorks::Layer (p. 856) Added the `CustomJson` and `VolumeConfigurations` properties. AWS::OpsWorks::Stack (p. 862) Added the `ElasticIps`, `EcsClusterArn`, `RdsDbInstances`, `CloneAppIds`, `ClonePermissions`, and `SourceStackId` properties. AWS::RDS::DBInstance (p. 881) Added the `CopyTagsToSnapshot` property.	2010-05-15
List imports	November 22, 2016	List imports of an exported output value to track which AWS CloudFormation stacks are importing the value. For more information, see Listing Stacks That Import an Exported Output Value (p. 121).	2010-05-15
Transforms	November 17, 2016	Specify the AWS Serverless Application Model (AWS SAM) that AWS CloudFormation uses to process AWS SAM syntax for serverless applications. For more information, see Transform (p. 148).	2010-05-15
New resource	November 17, 2016	AWS::SNS::Subscription (p. 953) Use the `AWS::SNS::Subscription` resource to subscribe an endpoint to an Amazon Simple Notification Service topic.	2010-05-15
Updated resource	November 17, 2016	AWS::Lambda::Function (p. 824) Use the `Environment` property to specify key-value pairs (environment variables) that your AWS Lambda function can access. Use the `KmsKeyArn` property to specify an AWS Key Management Service key that AWS Lambda uses to encrypt and decrypt environment variables.	2010-05-15

Change	Release Date	Description	API Version
New CLI commands	November 17, 2016	Uploading Local Artifacts to an S3 Bucket (p. 88) Use the `aws cloudformation package` command to upload local artifacts that are referenced in an AWS CloudFormation template to an S3 bucket. Quickly Deploying Templates with Transforms (p. 88) Use the `aws cloudformation deploy` command to combine the create and execute change set actions into a single command. This command is useful for quickly creating or updating stacks that contain transforms.	2010-05-15
Updated resource	November 03, 2016	AWS::CloudFront::Distribution (p. 492) For the CloudFront DistributionConfig (p. 1040) property, use the `HttpVersion` property to specify the latest HTTP version that viewers can use to communicate with Amazon CloudFront. For the CloudFront ForwardedValues (p. 1057) property, use the `QueryStringCacheKeys` property to specify the query string parameters that CloudFront uses to determine which content to cache.	2010-05-15
List stack exports	November 03, 2016	Use the AWS CloudFormation console, API, or AWS CLI to see a list of all the exported output values for a region. For more information, see Exporting Stack Output Values (p. 120).	2010-05-15
Continuous delivery with stacks	November 03, 2016	Use AWS CodePipeline to build continuous delivery workflows with AWS CloudFormation stacks. For more information, see Continuous Delivery with AWS CodePipeline (p. 56).	2010-05-15
Skip resources during rollback	November 03, 2016	If you have a stack in the `UPDATE_ROLLBACK_FAILED` state, use the `ResourcesToSkip` parameter for the `ContinueUpdateRollback` action to skip resources that AWS CloudFormation can't rollback. For more information, see the Troubleshooting section in Update Rollback Failed (p. 1372).	2010-05-15
Change sets enhancement	November 03, 2016	You can create a new stack using a change set (p. 72).	2010-05-15
Updated resource	October 12, 2016	AWS::ElastiCache::CacheCluster (p. 684) Update the `CacheNodeType` property without replacing the cluster. AWS::ElastiCache::ReplicationGroup (p. 693) You can create a Redis (cluster mode enabled) replication group that can contain multiple node groups (shards), each with a primary cluster and read replicas. AWS::ElastiCache::SubnetGroup (p. 706) Use the `CacheSubnetGroupName` property to specify a name for an Amazon ElastiCache subnet group.	2010-05-15

Change	Release Date	Description	API Version
New resources	October 06, 2016	AWS::ApiGateway::UsagePlan (p. 423) Use the `AWS::ApiGateway::UsagePlan` resource to specify a usage plan for deployed Amazon API Gateway APIs. AWS::CodeCommit::Repository (p. 507) Use the `AWS::CodeCommit::Repository` resource to create an AWS CodeCommit repository that is hosted by Amazon Web Services.	2010-05-15
Updated resources	October 06, 2016	AWS::ApiGateway::Authorizer (p. 399) Use the `ProviderARNs` property to use Amazon Cognito user pools as Amazon API Gateway API authorizers. AWS::ApiGateway::Deployment (p. 405) The `StageName` property is no longer required. AWS::ElasticLoadBalancingV2::TargetGroup (p. 739) For the `GetAtt` function, use the `LoadBalancerArns` attribute to retrieve the Amazon Resource Names (ARNs) of the load balancers that route traffic to the target group. AWS::RDS::DBInstance (p. 881) Use the `Domain` and `DomainIAMRoleName` properties to use Windows Authentication when users connect to the RDS DB instance. AWS::EC2::SecurityGroupEgress (p. 611) Use the `DestinationPrefixListId` property to specify the AWS service prefix of an Amazon VPC endpoint.	2010-05-15
Cross-stack reference enhancement	October 06, 2016	Use intrinsic functions to customize the `Name` value of an export (p. 155) or to refer to a value in the `ImportValue (p. 1334)` function.	2010-05-15
AWS CloudFormation service role	September 26, 2016	Use an AWS Identity and Access Management (IAM) service role for AWS CloudFormation stack operations. AWS CloudFormation uses the role's credentials to make calls to stack resources on your behalf. For more information, see AWS CloudFormation Service Role (p. 13).	2010-05-15
New feature	September 19, 2016	You can use the `Export` output field and the `Fn::ImportValue` intrinsic function to have one stack refer to resource outputs in another stack. For more information, see Outputs (p. 155), Fn::ImportValue (p. 1334), and Walkthrough: Refer to Resource Outputs in Another AWS CloudFormation Stack (p. 200).	2010-05-15
YAML support	September 19, 2016	You can use the YAML format to author AWS CloudFormation templates. YAML also allows you to, for example, add comments to your templates or use the short form for intrinsic functions. For more information, see AWS CloudFormation Template Formats (p. 127).	2010-05-15

Change	Release Date	Description	API Version
New intrinsic function	September 19, 2016	Use the `Fn::Sub` function to substitute variables in an input string with values that you specify. For more information, see `Fn::Sub` (p. 1341).	2010-05-15
New resources	September 19, 2016	AWS::KMS::Alias (p. 814) Use the `AWS::KMS::Alias` resource to create an alias for an AWS Key Management Service customer master key.	
Updated resources	September 19, 2016	AWS::EC2::SpotFleet (p. 622) For the `LaunchSpecifications` property, use the `SpotPrice` property to specify a bid price for a specific instance type. AWS::ECS::Cluster (p. 669) Use the `ClusterName` property to specify a name for an Amazon EC2 Container Service cluster. AWS::ECS::TaskDefinition (p. 675) Use the `TaskRoleArn` property to specify an AWS Identity and Access Management role that Amazon EC2 Container Service containers use to make AWS calls on your behalf. Use the `Family` property to register a task definition to a specific family. AWS::Elasticsearch::Domain (p. 745) Use the `ElasticsearchVersion` property to specify which version of Elasticsearch to use.	2010-05-15
New resources	August 11, 2016	Use the following Elastic Load Balancing Application load balancer resources to distribute incoming application traffic to multiple targets, such as EC2 instances, in multiple Availability Zones: • AWS::ElasticLoadBalancingV2::Listener (p. 731) • AWS::ElasticLoadBalancingV2::ListenerRule (p. 733) • AWS::ElasticLoadBalancingV2::LoadBalancer (p. 736) • AWS::ElasticLoadBalancingV2::TargetGroup (p. 739)	2010-05-15
Updated resource	August 11, 2016	AWS::AutoScaling::AutoScalingGroup (p. 433) Use the `TargetGroupARNs` property to associate the Auto Scaling group with one or more Application load balancer target groups. AWS::ECS::Service (p. 671) For the load `LoadBalancers` property, use the `TargetGroupArn` property to associate an Amazon EC2 Container Service service with an Application load balancer target group.	2010-05-15

Change	Release Date	Description	API Version
New resources	August 09, 2016	AWS CloudFormation added the following resources: AWS::ApplicationAutoScaling::ScalableTarget (p. 427) and AWS::ApplicationAutoScaling::ScalingPolicy (p. 430) Use an Application Auto Scaling scaling policy to define when and how a target resource scales. AWS::CertificateManager::Certificate (p. 459) Provision an AWS Certificate Manager certificate that you can use with other AWS services to enable secure connections.	2010-05-15
Updated resources	August 09, 2016	AWS CloudFormation updated the following resources: AWS::CloudFront::Distribution (p. 492) For the distribution configuration `ViewerCertificate` property, you can specify an AWS Certificate Manager certificate. For the distribution configuration `Origin` property, you can specify custom headers and the SSL protocols for custom origins. AWS::EFS::FileSystem (p. 679) You can specify the performance mode for an Amazon Elastic File System file system.	2010-05-15
New resources	July 20, 2016	AWS IoT Use AWS IoT to declare an AWS IoT policy, an X.509 certificate, an association between a policy and a principal (an X.509 certificate or other credential), an AWS IoT thing, an association between a principal and a thing, or an AWS IoT rule. • AWS::IoT::Certificate (p. 797) • AWS::IoT::Policy (p. 799) • AWS::IoT::PolicyPrincipalAttachment (p. 801) • AWS::IoT::Thing (p. 803) • AWS::IoT::ThingPrincipalAttachment (p. 805) • AWS::IoT::TopicRule (p. 807)	2010-05-15

Change	Release Date	Description	API Version
Updated resources	July 20, 2016	AWS CloudFormation updated the following resources: AWS::IAM::Group (p. 777), AWS::IAM::Role (p. 787), AWS::IAM::User (p. 793) Use the name properties to specify a custom name for AWS Identity and Access Management (IAM) resources. AWS::ApiGateway::Method (p. 408) For the `Integration` property, you can use the `PassthroughBehavior` property to specify when Amazon API Gateway passes requests to the targeted back end. AWS::ApiGateway::Model (p. 412) and AWS::ApiGateway::RestApi (p. 417) You can specify JSON objects for the `Schema` and `Body` properties.	2010-05-15
Auto Scaling group UpdatePolicy	June 9, 2016	For the `UpdatePolicy` attribute, use the `AutoScalingReplacingUpdate` property to specify whether an Auto Scaling group and the instances it contains are replaced when you update the Auto Scaling group. During a replacement, AWS CloudFormation retains the old Auto Scaling group until it creates the new one successfully so that AWS CloudFormation can roll back to the old Auto Scaling group if the update fails. For more information, see UpdatePolicy (p. 1303).	2010-05-15
New resource	June 9, 2016	AWS CloudFormation added the following resources: AWS::EC2::FlowLog (p. 570) Creates an Amazon Elastic Compute Cloud flow log that captures IP traffic for a specified network interface, subnet, or VPC. AWS::KinesisFirehose::DeliveryStream (p. 811) Creates a delivery stream that delivers real-time streaming data to a destination, such as Amazon Simple Storage Service, Amazon Redshift, or Amazon Elasticsearch Service.	2010-05-15
Updated resources	June 9, 2016	AWS CloudFormation updated the following resources: AWS::Kinesis::Stream (p. 810) Use the `Name` property to specify a name for an Amazon Kinesis stream. AWS::Lambda::Function (p. 824) For the `Code` property, you can use the `ZipFile` property and cfn response module for `nodejs4.3` runtime environments. AWS::SNS::Topic (p. 954) AWS CloudFormation enabled updates for the Amazon Simple Notification Service topic resource.	2010-05-15

Change	Release Date	Description	API Version
New resource	April 25, 2016	Use the AWS::EC2::Host (p. 572) resource to allocate a fully dedicated physical server for launching EC2 instances.	2010-05-15
Updated resources	April 25, 2016	AWS::EC2::Instance (p. 574) Use the `Affinity` and `HostId` properties to launch instances onto an Amazon Elastic Compute Cloud dedicated host. AWS::ECS::Service (p. 671) Use the `DeploymentConfiguration` property to configure how many tasks can run during a deployment. AWS::ECS::TaskDefinition (p. 675) AWS CloudFormation added support for additional Amazon EC2 Container Service container definition properties. AWS::GameLift::Fleet (p. 771) Use the `MaxSize` and `MinSize` properties to specify the maximum and minimum number of EC2 instances allowed in your Amazon GameLift fleet. AWS::Lambda::Function (p. 824) Use the `FunctionName` property to specify a name for your AWS Lambda function. You can also use Python 2.7 to specify an inline function.	2010-05-15

Change	Release Date	Description	API Version
New resources	April 18, 2016	Amazon API Gateway Use the Amazon API Gateway resources to publish, maintain, and monitor APIs at any scale. You can create APIs that clients can call to access your back-end services, such as applications running EC2 instances or code running on AWS Lambda. • AWS::ApiGateway::Account (p. 395) • AWS::ApiGateway::ApiKey (p. 397) • AWS::ApiGateway::Authorizer (p. 399) • AWS::ApiGateway::BasePathMapping (p. 403) • AWS::ApiGateway::ClientCertificate (p. 404) • AWS::ApiGateway::Deployment (p. 405) • AWS::ApiGateway::Method (p. 408) • AWS::ApiGateway::Model (p. 412) • AWS::ApiGateway::Resource (p. 415) • AWS::ApiGateway::RestApi (p. 417) • AWS::ApiGateway::Stage (p. 420) AWS::Events::Rule (p. 761) Create an Amazon CloudWatch Events rule that monitors changes to AWS resources in your account (events). If an incoming event matches the conditions that you described in the rule, Amazon CloudWatch Events sends messages to and activates your specified targets, such as AWS Lambda functions or Amazon Simple Notification Service topics. AWS::WAF::SizeConstraintSet (p. 988) and AWS::WAF::XssMatchSet (p. 999) Use the two AWS WAF rules to check the size of a web request or to prevent cross-site scripting attacks.	2010-05-15
New resources	March 31, 2016	Use the AWS::Lambda::Alias (p. 822) resource to create aliases for your AWS Lambda functions and the AWS::Lambda::Version (p. 831) resource to create versions of your functions.	2010-05-15

Change	Release Date	Description	API Version
Updated resources	March 31, 2016	AWS CloudFormation updated the following resources: AWS::EMR::Cluster (p. 750) and AWS::EMR::InstanceGroupConfig (p. 755) Use the `EbsConfiguration` property to configure Amazon Elastic Block Store storage volumes for your Amazon EMR clusters or instance groups. AWS::Lambda::Function (p. 824) Use the `VpcConfig` property to enable AWS Lambda functions to access resources in a VPC. AWS::S3::Bucket (p. 937) For the Amazon Simple Storage Service life cycle rules, you can specify multiple transition rules that specify when objects transition to a specified storage class.	2010-05-15
Change sets	March 29, 2016	Before updating stacks, use change sets to see how your changes might affect your running resources. For more information, see Updating Stacks Using Change Sets (p. 93).	2010-05-15
New resources	March 15, 2016	Use the AWS::GameLift::Alias (p. 766), AWS::GameLift::Build (p. 768), and AWS::GameLift::Fleet (p. 771) resources to deploy multiplayer game servers in AWS.	2010-05-15
New resources	February 26, 2016	AWS CloudFormation added the following resources: AWS::ECR::Repository (p. 667) Create Amazon EC2 Container Registry repositories where users can push and pull Docker images. AWS::EC2::NatGateway (p. 586) Use the network address translator (NAT) gateway to enable EC2 instances in a private subnet to connect to the Internet. AWS::Elasticsearch::Domain (p. 745) Create Amazon Elasticsearch Service (Amazon ES) domains that contain the Amazon ES engine instances, which process Amazon ES requests. AWS::EMR::Cluster (p. 750), AWS::EMR::InstanceGroupConfig (p. 755), AWS::EMR::Step (p. 758) Use the Amazon EMR resources to help you analyze and process vast amounts of data. You can create clusters and then run jobs on them.	2010-05-15

Change	Release Date	Description	API Version
Updated resources	February 26, 2016	AWS CloudFormation updated the following resources: AWS::CloudTrail::Trail (p. 493) Use the `IsMultiRegionTrail` property to specify whether to create an AWS CloudTrail trail in the region in which you create a stack or in all regions. AWS::Config::ConfigurationRecorder (p. 531) For the recording group, use the `IncludeGlobalResourceTypes` property to record all global resource types. AWS::RDS::DBCluster (p. 872) Use the `KmsKeyId` and `StorageEncrypted` properties to encrypt database instances in the cluster.	2010-05-15
Retain resources	February 26, 2016	For stacks in the `DELETE_FAILED` state, use the `RetainResources` parameter to retain resources that AWS CloudFormation can't delete. For more information, see Delete Stack Fails (p. 1369).	2010-05-15
Update stack tags	February 26, 2016	You can add, modify, or remove stack tags when you update a stack. For more information, see AWS CloudFormation Stacks Updates (p. 89).	2010-05-15
Continue rolling back failed update rollbacks	January 25, 2016	For a stack in the `UPDATE_ROLLBACK_FAILED` state, you can continue rolling back the update to get your stack in a working state. That way, you can return the stack to its original settings and try to update it again. For more information, see Continue Rolling Back an Update (p. 119).	2010-05-15
New sample templates available for the Asia Pacific (Seoul) region.	January 7, 2016	The following collection of AWS CloudFormation sample templates are for the ap-northeast-2 region: • Sample Solutions • Application Frameworks • Services For more information, see Sample Templates (p. 1367).	2010-05-15

Change	Release Date	Description	API Version
New resources	December 28, 2015	AWS CloudFormation added the following resources: AWS::DirectoryService::MicrosoftAD (p. 544) Use the Microsoft Active Directory resource to create a Microsoft Active Directory directory in AWS. AWS::Logs::Destination (p. 833) and AWS::Logs::LogStream (p. 837) Use the Amazon CloudWatch Logs resources to create a destination for real-time processing of log data or to create log streams, respectively. AWS::WAF::ByteMatchSet (p. 979), AWS::WAF::IPSet (p. 983), AWS::WAF::Rule (p. 986), AWS::WAF::SqlInjectionMatchSet (p. 991), and AWS::WAF::WebACL (p. 994) Use the AWS WAF resources to control and monitor web requests to your content.	2010-05-15
Resource updates	December 28, 2015	AWS CloudFormation updated the following resources: AWS::CloudFront::Distribution (p. 492) For the distribution configuration, use the `WebACLId` property to associate an AWS WAF web access control list (ACL) with an Amazon CloudFront distribution. For the cache behavior and default cache behavior, you can specify a default and maximum Time to Live (TTL) value. AWS::DynamoDB::Table (p. 550) You can create, update, or delete a global secondary index without replacing your Amazon DynamoDB table. AWS::S3::Bucket (p. 937) Use the `ReplicationConfiguration` property to specify which objects to replicate and where they are stored. Use the properties in the `NotificationConfiguration` property to specify filters so that Amazon Simple Storage Service sends notifications for objects that you specify.	2010-05-15
Parameter grouping and sorting	December 3, 2015	Use the AWS::CloudFormation::Interface (p. 483) metadata key to group and sort parameters in the AWS CloudFormation console when users create or update a stack with your template.	2010-05-15
Update policy attribute	December 3, 2015	For an Auto Scaling update policy attribute (p. 1303), use the `MinSuccessfulInstancesPercent` property to specify the percentage of instances that must signal success for a successful update.	2010-05-15

Change	Release Date	Description	API Version
New resources	December 3, 2015	AWS CloudFormation added the following resources: AWS::CodePipeline::Pipeline (p. 520) and AWS::CodePipeline::CustomActionType (p. 517) Use the AWS CodePipeline resources to create a pipeline that describes how software changes go through a release process. AWS::Config::ConfigurationRecorder (p. 531), AWS::Config::DeliveryChannel (p. 533), and AWS::Config::ConfigRule (p. 525) Use the AWS Config resources to monitor configuration changes to specific AWS resources. AWS::KMS::Key (p. 816) Use the AWS Key Management Service (AWS KMS) resource to create customer master keys in AWS KMS that users can use to encrypt small amounts of data. AWS::SSM::Document (p. 968) Use the Amazon EC2 Systems Manager to create a document that specifies on-instance configurations.	2010-05-15

Change	Release Date	Description	API Version
Resources update	December 3, 2015	AWS CloudFormation updated the following resources: AWS::AutoScaling::LaunchConfiguration (p. 440) Specify whether EBS volumes are encrypted. AWS::AutoScaling::ScalingPolicy (p. 452) You can use two different policy types (simple and step scaling) to specify how an Auto Scaling group scales when an Amazon CloudWatch (CloudWatch) alarm is breached. AWS::CloudTrail::Trail (p. 493) Use the CloudWatch properties to send logs to a CloudWatch log group. You can add tags to a trail and specify an AWS KMS key that you want to use to encrypt logs. AWS::CodeDeploy::Application (p. 509), AWS::CodeDeploy::DeploymentConfig (p. 510), and AWS::CodeDeploy::DeploymentGroup (p. 512) Use the `ApplicationName`, `DeploymentConfigName`, and `DeploymentGroupName` properties to specify custom names for AWS CodeDeploy resources. AWS::DynamoDB::Table (p. 550) Use the `StreamSpecification` property to specify settings for capturing changes to items stored in an Amazon DynamoDB (DynamoDB) table. AWS::EC2::Instance (p. 574) Use the `SsmAssociations` property to associate an Amazon EC2 Systems Manager document with an instance. AWS::EC2::SpotFleet (p. 622) Use the `AllocationStrategy` property to specify how to allocate target capacity across Spot pools. Use the `ExcessCapacityTerminationPolicy` property to specify how instances are terminated if the target capacity is below the size of the Spot fleet. AWS::Redshift::Cluster (p. 911) Use the `KmsKeyId` property to specify an AWS KMS key to encrypt data in an Amazon Redshift cluster. AWS::WorkSpaces::Workspace (p. 1002) Use the encryption properties to encrypt data stored on volumes.	2010-05-15
Resource update	November 4, 2015	For the AWS::EC2::Volume (p. 633) resource, use the `AutoEnableIO` property to automatically resume I/O operations if a volume's data becomes inconsistent.	2010-05-15

Change	Release Date	Description	API Version
New resources	October 1, 2015	AWS CloudFormation added the following resources: AWS::CodeDeploy::Application (p. 509), AWS::CodeDeploy::DeploymentGroup (p. 512), and AWS::CodeDeploy::DeploymentConfig (p. 510) Use the AWS CodeDeploy resources to create and apply deployments to EC2 or on-premises instances. AWS::DirectoryService::SimpleAD (p. 547) Use the Simple Active Directory resource to create an AWS Directory Service Simple AD, which is a Microsoft Active Directory-compatible directory. AWS::EC2::PlacementGroup (p. 601) Use a placement group to create a cluster of instances in a low-latency network. AWS::EC2::SpotFleet (p. 622) Use a Spot fleet to launch a collection of Spot instances that run interruptible tasks. AWS::Lambda::EventSourceMapping (p. 819) Use the event source mapping resource to specify a stream as an event source for an AWS Lambda (Lambda) function. AWS::Lambda::Permission (p. 829) Use a Lambda permission to add a statement to a Lambda function's policy. AWS::Logs::SubscriptionFilter (p. 841) Use the subscription filter to define which log events are delivered to your Amazon Kinesis stream. AWS::RDS::DBCluster (p. 872) and AWS::RDS::DBClusterParameterGroup (p. 879) Use the cluster and cluster parameter group resources to create an Amazon Aurora DB cluster. AWS::WorkSpaces::Workspace (p. 1002) Use Amazon WorkSpaces to create cloud-based desktop experiences.	2010-05-15

Change	Release Date	Description	API Version
Resource updates	October 1, 2015	AWS CloudFormation updated the following resources: AWS::ElastiCache::ReplicationGroup (p. 693) Use the `Fn::GetAtt` intrinsic function to get a list of read-only replica addresses and ports. AWS::OpsWorks::Stack (p. 862) Use the `AgentVersion` property to specify a particular AWS OpsWorks agent. AWS::OpsWorks::App (p. 843) Use the `Environment` property to specify environment variables for an AWS OpsWorks app. AWS::S3::Bucket (p. 937) For the NotificationConfiguration (p. 1256) property, you can configure notification settings for Lambda functions and Amazon Simple Queue Service (Amazon SQS) queues.	2010-05-15
IAM condition keys	October 1, 2015	For AWS Identity and Access Management (IAM) policies, use AWS CloudFormation-specific condition keys to specify when an IAM policy takes effect. For more information, see Controlling Access with AWS Identity and Access Management (p. 8).	2010-05-15
AWS CloudFormation Designer	October 1, 2015	Use AWS CloudFormation Designer (p. 157) to create and modify templates using a drag-and-drop interface.	2010-05-15
New resource	August 24, 2015	Use the AWS::EC2::VPCEndpoint (p. 645) resource to establish a private connection between your VPC and another AWS service.	2010-05-15

Change	Release Date	Description	API Version
Resource updates	August 24, 2015	AWS CloudFormation updated the following resources: AWS::ElasticBeanstalk::Environment (p. 714) Use the `Tags` property to specify tags (key-value pairs) for an AWS Elastic Beanstalk (Elastic Beanstalk) environment. AWS::Lambda::Function (p. 824) For the `Code` (p. 1211) property, use the `ZipFile` property to write the source code of your Lambda function directly in a template. Currently, you can use the `ZipFile` property only for `nodejs` runtime environments. You can still point to a file in an S3 bucket for all runtime environments, such as `java8` and `nodejs`. AWS::OpsWorks::Instance (p. 848) Use the `EbsOptimized` property to indicate whether an instance is optimized for Amazon Elastic Block Store (Amazon EBS) I/O. AWS::RDS::DBInstance (p. 881) For the `SourceDBInstanceIdentifier` property, you can specify a database instance in another region to create a cross-region read replica.	2010-05-15
Amazon S3 template URL	August 24, 2015	For versioning-enabled buckets, you can specify a version ID in an Amazon S3 template URL when you create or update a stack, such as `https://s3.amazonaws.com/templates/myTemplate.template?versionId=123ab1cdeKdOW5IH4GAcYbEngcpTJTDW`.	2010-05-15
New resource	August 3, 2015	Use the AWS::EFS::FileSystem (p. 679) resource to create an Amazon Elastic File System (Amazon EFS) file system and the AWS::EFS::MountTarget (p. 681) resource to create a mount point for a file system.	2010-05-15
Permission requirement change	June 11, 2015	When you create or update an AWS::RDS::DBInstance (p. 881) resource, you must now also have permission to call the `ec2:DescribeAccountAttributes` action.	2010-05-15

Change	Release Date	Description	API Version
New resources	June 11, 2015	AWS CloudFormation added the following resources: AWS::DataPipeline::Pipeline (p. 535) Use data pipelines to automate the movement and transformation of data. Amazon EC2 Container Service resources Use the AWS::ECS::Service (p. 671), AWS::ECS::Cluster (p. 669), and AWS::ECS::TaskDefinition (p. 675) resources to create Docker containers on a cluster of EC2 instances. AWS::ElastiCache::ReplicationGroup (p. 693) Use replication groups to create a collection of nodes with one primary read-write cluster and a maximum of five secondary read-only clusters. AWS::IAM::ManagedPolicy (p. 780) Use managed policies to create policies in your AWS account that you can use to apply permissions to IAM users, groups, and roles. AWS::Lambda::Function (p. 824) Use Lambda functions to run code in response to events. AWS::RDS::OptionGroup (p. 907) Use option groups to help you create and manage Amazon Relational Database Service (Amazon RDS) databases.	2010-05-15

Change	Release Date	Description	API Version
Resource updates	June 11, 2015	AWS CloudFormation updated the following resources: AWS::EC2::Subnet (p. 625) Use the MapPublicIpOnLaunch property to automatically assign public IP addresses to instances in a subnet. AWS::ElastiCache::CacheCluster (p. 684) Use the SnapshotName property to restore snapshot data into a new Redis cache cluster. AWS::IAM::User (p. 793) For the LoginProfile property, use the PasswordResetRequired property so that users are required to set a new password when they log in to the AWS Management Console. AWS::OpsWorks::Layer (p. 856) Use the LifecycleEventConfiguration property to configure lifecycle events for an AWS OpsWorks layer. AWS::S3::Bucket (p. 937) For the LifecycleConfiguration property, use the NoncurrentVersionExpirationInDays and NoncurrentVersionTransition properties to specify lifecycle rules for non-current object versions.	2010-05-15
New parameter types	May 19, 2015	Whenever you use the AWS CloudFormation console to create or update a stack, you can search for AWS-specific parameter type values by ID, name, or Name tag value. AWS CloudFormation also added support for the following AWS-specific parameter types. For more information, see Parameters (p. 132). • AWS::EC2::AvailabilityZone::Name • List<AWS::EC2::AvailabilityZone::Name> • AWS::EC2::Instance::Id • List<AWS::EC2::Instance::Id> • AWS::EC2::Image::Id • List<AWS::EC2::Image::Id> • AWS::EC2::SecurityGroup::GroupName • List<AWS::EC2::SecurityGroup::GroupName> • AWS::EC2::Volume::Id • List<AWS::EC2::Volume::Id> • AWS::Route53::HostedZone::Id • List<AWS::Route53::HostedZone::Id>	2010-05-15

Change	Release Date	Description	API Version
New resources	April 16, 2015	AWS CloudFormation added the following resources: AWS::AutoScaling::LifecycleHook (p. 449) Use Auto Scaling lifecycle hooks to control the state of an instance after it is launched or terminated. AWS::RDS::EventSubscription (p. 905) Use event subscriptions to get notifications about Amazon RDS events.	2010-05-15

Change	Release Date	Description	API Version
Resource updates	April 16, 2015	AWS CloudFormation updated the following resources: AWS::AutoScaling::AutoScalingGroup (p. 433) Use the `NotificationConfigurations` property to specify multiple notifications. AWS::AutoScaling::LaunchConfiguration (p. 440) Use the `PlacementTenancy` property to specify the tenancy of instances. Use the `ClassicLinkVPCId` and `ClassicLinkVPCSecurityGroups` properties to link EC2-Classic instances to a ClassicLink-enabled VPC. AWS::AutoScaling::ScalingPolicy (p. 452) Use the `MinAdjustmentStep` property to specify the minimum number of instances that are added or removed during a scaling event. AWS::CloudFront::Distribution (p. 492) For viewer certificates, use the `MinimumProtocolVersion` property to specify a minimum protocol version. For cache behaviors, use the `CachedMethods` property to specify which methods Amazon CloudFront (CloudFront) caches responses for. For origins, use the `OriginPath` to specify a path that CloudFront uses to request content. AWS::ElastiCache::CacheCluster (p. 684) For Memcached cache clusters, use the `AZMode` and `PreferredAvailabilityZones` properties to specify nodes in multiple Availability Zones (AZs). AWS::EC2::Volume (p. 633) Use the `KmsKeyId` property to specify a master key for encrypted volumes. AWS::OpsWorks::Instance (p. 848) Use the `TimeBasedAutoScaling` property to automatically scale instances based on a schedule that you specify. AWS::OpsWorks::Layer (p. 856) Use the `LoadBasedAutoScaling` property to specify load-based scaling policies. For volume configurations, use the `VolumeType` and `Iops` properties to specify a volume type and the number of I/O operations per second, respectively. AWS::RDS::DBInstance (p. 881) Use the `CharacterSetName` property to specify a character set for supported database engines. Use the `StorageEncrypted` property to indicate whether database instances will be encrypted and the `KmsKeyId` to specify a master key for encrypted database instances.	2010-05-15

Change	Release Date	Description	API Version
		AWS::Route53::HealthCheck (p. 925) Use the `HealthCheckTags` property to associate tags with health checks. AWS::Route53::HostedZone (p. 927) Use the `VPCs` property to create private hosted zones. Use the `HostedZoneTags` property to associate tags with hosted zones.	
New template section	April 16, 2015	Add the Metadata (p. 131) section to your templates to include arbitrary JSON objects that describe your templates, such as the design or implementation details.	2010-05-15
Resource update	April 8, 2015	For the AWS::CloudFormation::CustomResource (p. 467) resource, you can specify Lambda function Amazon Resource Names (ARNs) in the `ServiceToken` property.	2010-05-15
Amazon RDS update	December 24, 2014	AWS CloudFormation added two new properties for RDS DB instances. You can associate an option group with a DB instance and specify the DB instance storage type. For more information, see AWS::RDS::DBInstance (p. 881).	2010-05-15
Elastic Load Balancing update	December 24, 2014	You can use the `ConnectionSettings` property to specify how long connections can remain idle. For more information, see AWS::ElasticLoadBalancing::LoadBalancer (p. 719).	2010-05-15
Amazon Route 53 update	November 6, 2014	You can now provision and manage Amazon Route 53 hosted zones (p. 927), health checks (p. 925), failover record sets (p. 930), and geolocation record sets (p. 1241).	2010-05-15
Auto Scaling rolling update enhancement	November 6, 2014	During an update, you can use the `WaitOnResourceSignals` flag to instruct AWS CloudFormation to wait for instances to signal success. That way, AWS CloudFormation won't update the next batch of instances until the current batch is ready. For more information, see UpdatePolicy (p. 1303).	2010-05-15
New VPC Fn:GetAtt attributes	November 6, 2014	Given a VPC ID, you can retrieve the default security group and network ACL for that VPC. For more information, see `Fn::GetAtt` (p. 1324).	2010-05-15
New AWS-specific parameter types	November 6, 2014	You can specify AWS-specific parameter types in your AWS CloudFormation templates. In the AWS CloudFormation console, these parameter types provide a drop-down list of valid values. With the API or CLI, AWS CloudFormation can quickly validate values for these parameter types before creating or updating a stack. For more information, see Parameters (p. 132).	2010-05-15
CreationPolicy attribute	November 6, 2014	With the CreationPolicy attribute, you can instruct AWS CloudFormation to wait until applications are ready on EC2 instances before proceeding with stack creation. You can use a creation policy instead of a wait condition and wait condition handle. For more information, see CreationPolicy (p. 1293).	2010-05-15

Change	Release Date	Description	API Version
Amazon CloudFront forwarded values	September 29, 2014	For cache behaviors, you can forward headers to the origin. See CloudFront ForwardedValues (p. 1057).	2010-05-15
AWS OpsWorks update	September 29, 2014	For Chef 11.10, you can use the `ChefConfiguration` property to enable Berkshelf. You can also use the AWS OpsWorks built-in security groups with your AWS OpsWorks stacks. For more information, see AWS::OpsWorks::Stack (p. 862).	2010-05-15
Elastic Load Balancing tagging support	September 29, 2014	AWS CloudFormation tags Elastic Load Balancing load balancers with stack-level tags. You can also add your own tags to a load balancer. See AWS::ElasticLoadBalancing::LoadBalancer (p. 719).	2010-05-15
Amazon Simple Notification Service topic policy update	September 29, 2014	You can now update Amazon SNS topic policies. For more information, see AWS::SNS::TopicPolicy (p. 957).	2010-05-15
RDS DB instance update	September 5, 2014	You can specify whether a DB instance is Internet-facing by using the `PubliclyAccessible` property in the AWS::RDS::DBInstance (p. 881) resource.	2010-05-15
UpdatePolicy attribute update	September 05, 2014	You can specify an update policy for an Auto Scaling group that has an associated scheduled action. For more information, see UpdatePolicy (p. 1303).	2010-05-15
Amazon CloudWatch support	July 10, 2014	You can use AWS CloudFormation to provision and manage Amazon CloudWatch Logs (CloudWatch Logs) log groups and metric filters. For more information, see AWS::Logs::LogGroup (p. 835) or AWS::Logs::MetricFilter (p. 839).	2010-05-15
Amazon CloudFront distribution configuration update	June 17, 2014	You can specify additional CloudFront distribution configuration properties: • Custom error responses define custom error messages for 4xx and 5xx HTTP status codes. • Price class defines the maximum price that you want to pay for the CloudFront service. • Restrictions define who can view your content. • Viewer certificate specifies the certificate to use when viewers use HTTPS. • For cache behaviors, you can specify allowed HTTP methods and indicate whether to forward cookies. For more information, see AWS::CloudFront::Distribution (p. 492).	2010-05-15
EC2 instance update	June 17, 2014	You can specify whether an instance stops or terminates when you invoke the instance's operating system shutdown command. For more information, see AWS::EC2::Instance (p. 574).	2010-05-15

Change	Release Date	Description	API Version
EBS volume update	June 17, 2014	You can use encrypted EBS volumes with supported instance types. For more information, see AWS::EC2::Volume (p. 633).	2010-05-15
New Amazon VPC peering connection	June 17, 2014	You can use AWS CloudFormation to create an Amazon Virtual Private Cloud (Amazon VPC) peering connection, which establishes a network connection between two VPCs. For more information, see AWS::EC2::VPCPeeringConnection (p. 649).	2010-05-15
Auto Scaling group update	June 17, 2014	You can specify an existing cluster placement group in which to launch instances for an Auto Scaling group. For more information, see AWS::AutoScaling::AutoScalingGroup (p. 433).	2010-05-15
AWS CloudTrail support	June 17, 2014	AWS CloudFormation supports AWS CloudTrail, which can capture API calls made from your AWS account and publish the logs at a location you designate. For more information, see AWS::CloudTrail::Trail (p. 493).	2010-05-15
Update stack enhancements	May 12, 2014	AWS CloudFormation supports additional features for updating stacks: • You can update AWS CloudFormation stack parameters without resubmitting the stack's template. • You can add or remove Amazon SNS notification topics for an AWS CloudFormation stack. For more information, see AWS CloudFormation Stacks Updates (p. 89).	2010-05-15
Amazon Kinesis support	May 6, 2014	You can use AWS CloudFormation to create Amazon Kinesis streams that capture and transport data records from data sources. For more information, see AWS::Kinesis::Stream (p. 810).	2010-05-15
New S3 bucket properties	May 5, 2014	AWS CloudFormation supports additional S3 bucket properties: • Cross-origin resource sharing (CORS) defines cross-origin resource sharing of objects in a bucket. • Lifecycle defines how Amazon S3 manages objects during their lifetime. • Access logging policy captures information about requests made to your bucket. • Notifications define which events to report and which Amazon SNS topic to send messages to. • Versioning enables multiple variants of all objects in a bucket. • Redirect and routing rules govern redirect behavior for requests made to a bucket's website endpoint. For more information, see AWS::S3::Bucket (p. 937).	2010-05-15

Change	Release Date	Description	API Version
Auto Scaling support	May 5, 2014	AWS CloudFormation supports metrics collection for an Auto Scaling group. For more information, see AWS::AutoScaling::AutoScalingGroup (p. 433).	2010-05-15
`Fn::If` update	May 5, 2014	You can use the `Fn::If` intrinsic function in the output section of a template. For more information, see Condition Functions (p. 1312).	2010-05-15
API logging with AWS CloudTrail	April 2, 2014	You can use AWS CloudTrail (CloudTrail) to log AWS CloudFormation requests. With CloudTrail you can get a history of AWS CloudFormation API calls for your account. For more information, see Logging AWS CloudFormation API Calls in AWS CloudTrail (p. 13).	2010-05-15
Elastic Load Balancing update	March 20, 2014	You can specify an access logging policy to capture information about requests made to your load balancer. You can also specify a connection draining policy that describes how to handle in-flight requests when instances are deregistered or become unhealthy. For more information, see AWS::ElasticLoadBalancing::LoadBalancer (p. 719).	2010-05-15
AWS OpsWorks support	March 3, 2014	You can use AWS CloudFormation to provision and manage AWS OpsWorks stacks. For more information, see AWS::OpsWorks::Stack (p. 862) or AWS OpsWorks Template Snippets (p. 334).	2010-05-15
Amazon S3 template size limit increase	February 18, 2014	You can specify template sizes up to 460,800 bytes in Amazon S3.	2010-05-15
Amazon Redshift support	February 10, 2014	You can use AWS CloudFormation to provision and manage Amazon Redshift clusters. For more information, see Amazon Redshift Template Snippets (p. 340) or AWS::Redshift::Cluster (p. 911).	2010-05-15
S3 buckets and bucket policies update	February 10, 2014	You can update some properties of the S3 bucket and bucket policy resources. For more information, see AWS::S3::Bucket (p. 937) or AWS::S3::BucketPolicy (p. 950).	2010-05-15
Elastic Beanstalk environments and application versions update	February 10, 2014	You can update Elastic Beanstalk environment configurations and application versions. For more information, see AWS::ElasticBeanstalk::Environment (p. 714), AWS::ElasticBeanstalk::ConfigurationTemplate (p. 711), or AWS::ElasticBeanstalk::ApplicationVersion (p. 709).	2010-05-15
Amazon SQS update	January 29, 2014	You can specify a dead letter queue for an Amazon SQS queue. For more information, see AWS::SQS::Queue (p. 958).	2010-05-15
Auto Scaling scheduled actions	January 27, 2014	You can scale the number of EC2 instances in an Auto Scaling group based on a schedule. By using a schedule, you can scale applications in response to predictable load changes. For more information, see AWS::AutoScaling::ScheduledAction (p. 456).	2010-05-15

Change	Release Date	Description	API Version
DynamoDB secondary indexes	January 27, 2014	You can create local and global secondary indexes for DynamoDB databases. By using secondary indexes, you can efficiently access data with attributes other than the primary key. For more information, see AWS::DynamoDB::Table (p. 550).	2010-05-15
Auto Scaling update	January 2, 2014	You can specify an instance ID for an Auto Scaling group or launch configuration. You can also specify additional Auto Scaling block device properties. For more information, see AWS::AutoScaling::AutoScalingGroup (p. 433) or AWS::AutoScaling::LaunchConfiguration (p. 440).	2010-05-15
Amazon SQS update	January 2, 2014	You can update SQS queues and specify additional properties. For more information, see AWS::SQS::Queue (p. 958).	2010-05-15
Limit increases	January 2, 2014	You can specify up to 60 parameters and 60 outputs in your AWS CloudFormation templates.	2010-05-15
New console	December 19, 2013	The new AWS CloudFormation console adds features like auto-refreshing stack events and alphabetical ordering of stack parameters.	2010-05-15
Cross-zone load balancing	December 19, 2013	With cross-zone load balancing, you can route traffic to back-end instances across all Avalibility Zones (AZs). For more information, see AWS::ElasticLoadBalancing::LoadBalancer (p. 719).	2010-05-15
AWS Elastic Beanstalk environment tiers	December 19, 2013	You can specify whether AWS Elastic Beanstalk provisions resources to support a web server or to handle background processing tasks. For more information, see AWS::ElasticBeanstalk::Environment (p. 714).	2010-05-15
Resource names	December 19, 2013	You can assign names (physical IDs) to the following resources: • ElastiCache clusters • Elastic Load Balancing load balancers • RDS DB instances For more information, see Name Type (p. 1217).	2010-05-15
VPN support	November 22, 2013	You can enable a virtual private gateway (VGW) to propagate routes to the routing tables of a VPC. For more information, see AWS::EC2::VPNGatewayRoutePropagation (p. 665).	2010-05-15
Conditionally create resources and assign properties	November 8, 2013	Using input parameters, you can control the creation and settings of designated stack resources by defining conditions in your AWS CloudFormation templates. For example, you can use conditions to create stack resources for a production environment. Using the same template, you can create similar stack resources with lower capacity for a test environment. For more information, see Condition Functions (p. 1312).	2010-05-15

Change	Release Date	Description	API Version
Prevent accidental updates to stack resources	November 8, 2013	You can prevent stack updates that might result in unintentional changes to stack resources. For example, if you have a stack with a database layer that should rarely be updated, you can set a stack policy that prevents most users from updating that database layer. For more information, see Prevent Updates to Stack Resources (p. 110).	2010-05-15
Name resources	November 8, 2013	Instead of using AWS CloudFormation-generated physical IDs, you can assign names to certain resources. The following AWS CloudFormation resources support naming: • CloudWatch alarms • DynamoDB tables • Elastic Beanstalk applications and environments • S3 buckets • SNS topics • Amazon SQS queues For more information, see Name Type (p. 1217).	2010-05-15
Assign custom resource types	November 8, 2013	In your templates, you can specify your own resource type for AWS CloudFormation custom resources (`AWS::CloudFormation::CustomResource`). By using your own custom resource type name, you can quickly identify the type of custom resources that you have in your stack. For example, you can specify `"Type": "Custom::MyCustomResource"`. For more information, see AWS::CloudFormation::CustomResource (p. 467).	2010-05-15
Add pseudo parameter	November 8, 2013	You can now refer to the AWS AccountID inside AWS CloudFormation templates by referring to the `AWS::AccountID` pseudo parameter. For more information, see Pseudo Parameters Reference (p. 1351).	2010-05-15
Specify stacks in IAM policies	November 8, 2013	You can allow or deny IAM users, groups, or roles to operate on specific AWS CloudFormation stacks. For example, you can deny the delete stack action on a specific stack ID. For more information, see Controlling Access with AWS Identity and Access Management (p. 8).	2010-05-15
Federation support	October 14, 2013	AWS CloudFormation supports temporary security credentials from IAM roles, which enable scenarios such as federation and single sign-on to the AWS Management Console. You can also make calls to AWS CloudFormation from EC2 instances without embedding long-term security credentials by using IAM roles. For more information about AWS CloudFormation and IAM, see Controlling Access with AWS Identity and Access Management (p. 8).	2010-05-15
Amazon RDS read replica support	September 24, 2013	You can now create Amazon RDS read replicas from a source DB instance. For more information, see the `SourceDBInstanceIdentifier` property in the AWS::RDS::DBInstance (p. 881) resource.	2010-05-15

Change	Release Date	Description	API Version
Associate public IP address with instances in an Auto Scaling group	September 19, 2013	You can now associate public IP addresses with instances in an Auto Scaling group. For more information, see AWS::AutoScaling::LaunchConfiguration (p. 440).	2010-05-15
Additional VPC support	September 17, 2013	AWS CloudFormation adds several enhancements to support VPC and VPN functionality: • You can associate a public IP address and multiple private IP addresses to Amazon EC2 network interfaces. For more information, see AWS::EC2::NetworkInterface (p. 594). You can also associate a primary private IP address to an elastic IP address (EIP). • You can enable DNS support and specify DNS host names. For more information, see AWS::EC2::VPC (p. 639). • You can specify a static route between a virtual private gateway to your VPN gateway. For more information, see AWS::EC2::VPNConnectionRoute (p. 662).	2010-05-15
Redis and VPC security groups support for Amazon ElastiCache	September 3, 2013	You can now specify Redis as the cache engine for an Amazon ElastiCache (ElastiCache) cluster. You can also now assign VPC security groups to ElastiCache clusters. For more information, see AWS::ElastiCache::CacheCluster (p. 684).	2010-05-15
Parallel stack creation, update and deletion, and nested stack updates	August 12, 2013	AWS CloudFormation now creates, updates, and deletes resources in parallel, improving the operations' performance. If you update a top-level template, AWS CloudFormation automatically updates nested stacks that have changed. For more information, see AWS CloudFormation Stacks Updates (p. 89).	2010-05-15
VPC security groups can now be set in RDS DB instances	February 28, 2013	You can now assign VPC security groups to an RDS DB instance with AWS CloudFormation. For more information, see the VPCSecurityGroups (p. 892) property in AWS::RDS::DBInstance (p. 881).	2010-05-15
Rolling deployments for Auto Scaling groups	February 20, 2013	AWS CloudFormation now supports update policies on Auto Scaling groups, which describe how instances in the Auto Scaling group are replaced or modified when the Auto Scaling group adds or removes instances. You can modify these settings at stack creation or during a stack update. For more information and an example, see UpdatePolicy (p. 1303).	2010-05-15

Change	Release Date	Description	API Version
Cancel and rollback action for stack updates	February 20, 2013	AWS CloudFormation supports the ability to cancel a stack update. The stack must be in the UPDATE_IN_PROGRESS state when the update request is made. More information is available in the following topics: • Canceling a Stack Update (p. 109) • `aws cloudformation cancel-update-stack` • CancelUpdateStack in the *AWS CloudFormation API Reference*	2010-05-15
EBS-optimized instances for Auto Scaling groups	February 20, 2013	You can now provision EBS-optimized instances in Auto Scaling groups for dedicated throughput to Amazon Elastic Block Store (Amazon EBS) in autoscaled instances. The implementation is similar to that of the previously released support for optimized Amazon EBS EC2 instances. For more information, see the new `EbsOptimized` property in AWS::AutoScaling::LaunchConfiguration (p. 440).	2010-05-15
New documentation	December 21, 2012	AWS::EC2::Instance (p. 574) now provides a `BlockDeviceMappings` property to allow you to set block device mappings for your EC2 instance. With this change, two new types have been added: • Amazon EC2 Block Device Mapping Property (p. 1101) • Amazon Elastic Block Store Block Device Property (p. 1103)	2010-05-15
New documentation	December 21, 2012	New sections have been added to describe the procedures for creating and viewing stacks using the recently redesigned AWS Management Console. You can find them here: • Creating a Stack (p. 72) • Viewing Stack Data and Resources (p. 78)	2010-05-15
New documentation	November 15, 2012	Information about custom resources is provided in the following topics: • Custom Resources (p. 361) • AWS::CloudFormation::CustomResource (p. 467) • Custom Resource Reference (p. 372)	2010-05-15
Updated documentation	November 15, 2012	AWS CloudFormation now supports specifying provisioned I/O operations per second (IOPS) for RDS DB instances. You can set this value from 1000–10,000 in 1000 IOPS increments by using the new Iops (p. 888) property in AWS::RDS::DBInstance (p. 881). For more information about specifying IOPS for RDS DB instances, see Provisioned IOPS in the *Amazon Relational Database Service User Guide*.	2010-05-15

Change	Release Date	Description	API Version
New and updated documentation	August 27, 2012	Topics have been reorganized to more clearly provide specific information about using the AWS Management Console and using the AWS CloudFormation command-line interface (CLI). Information about tagging AWS CloudFormation stacks has been added, including new guides and updated reference topics: • New topic in Using the Console: Setting Stack Options (p. 76). • New information about tags in the *AWS CloudFormation API reference*: CreateStack, Stack, and Tag. New information about working with Windows stacks (p. 122): • Microsoft Windows Amazon Machine Images (AMIs) and AWS CloudFormation Templates (p. 122) • Bootstrapping AWS CloudFormation Windows Stacks (p. 122) New topic: Using Regular Expressions in AWS CloudFormation Templates (p. 383).	2010-05-15

Change	Release Date	Description	API Version
New feature	April 25, 2012	AWS CloudFormation now provides full support for Virtual Private Cloud (VPC) security with Amazon EC2 You can now create and populate an entire VPC with every type of VPC resource (subnets, gateways, network ACLs, route tables, and so forth) using a single AWS CloudFormation template. Templates that demonstrate new VPC features can be downloaded: Single instance in a single subnet Multiple subnets with Elastic Load Balancing (ELB) and an Auto Scaling group Documentation for the following resource types has been updated: AWS::EC2::SecurityGroup (p. 608) AWS::EC2::SecurityGroupIngress (p. 616) AWS::EC2::SecurityGroupEgress (p. 611) AWS::EC2::Instance (p. 574) AWS::AutoScaling::AutoScalingGroup (p. 433) AWS::EC2::EIP (p. 564) AWS::EC2::EIPAssociation (p. 566) AWS::ElasticLoadBalancing::LoadBalancer (p. 719) New resource types have been added to the documentation: AWS::EC2::VPC (p. 639) AWS::EC2::InternetGateway (p. 585) AWS::EC2::DHCPOptions (p. 560) AWS::EC2::DHCPOptions (p. 606) AWS::EC2::RouteTable (p. 602) AWS::EC2::NetworkAcl (p. 589) AWS::EC2::NetworkAclEntry (p. 590) AWS::EC2::Subnet (p. 625) AWS::EC2::VPNGateway (p. 663) AWS::EC2::CustomerGateway (p. 558)	2010-05-15
New feature	April 13, 2012	AWS CloudFormation now allows you to add or remove elements from a stack when updating it. AWS CloudFormation Stacks Updates (p. 89) has been updated, and a new section has been added to the walkthrough: Change the Stack's Resources (p. 42), which describes how to add and remove resources when updating the stack.	2010-05-15

Change	Release Date	Description	API Version
New feature	February 2, 2012	AWS CloudFormation now provides support for resources in an existing Amazon Virtual Private Cloud (Amazon VPC). With this release, you can: • Launch an EC2 Dedicated instance into an existing Amazon VPC. For more information, see AWS::EC2::Instance (p. 574). • Set the `SourceDestCheck` attribute of an EC2 instance that resides in an existing Amazon VPC. For more information, see AWS::EC2::Instance (p. 574). • Create Elastic IP addresses in an existing Amazon VPC. For more information, see AWS::EC2::EIP (p. 564). • Use AWS CloudFormation to create Amazon VPC security groups and ingress/egress rules in an existing VPC. For more information, see AWS::EC2::SecurityGroup (p. 608). • Associate an Auto Scaling group with an existing Amazon VPC by setting the `VPCZoneIdentifier` property of your `AWS::AutoScaling::AutoScalingGroup` resource. For more information, see AWS::AutoScaling::AutoScalingGroup (p. 433). • Attach an Elastic Load Balancing load balancer to a Amazon VPC subnet and create security groups for the load balancer. For more information, see AWS::ElasticLoadBalancing::LoadBalancer (p. 719). • Create an RDS DB instance in an existing Amazon VPC. For more information, see AWS::RDS::DBInstance (p. 881).	2010-05-15
New feature	February 2, 2012	You can now update properties for the following resources in an existing stack: • AWS::EC2::SecurityGroupIngress (p. 616) • AWS::EC2::SecurityGroupEgress (p. 611) • AWS::EC2::EIPAssociation (p. 566) • AWS::RDS::DBSubnetGroup (p. 903) • AWS::RDS::DBSecurityGroup (p. 898) • AWS::RDS::DBSecurityGroupIngress (p. 901) • AWS::Route53::RecordSetGroup (p. 935) For a complete list of updateable resources and details about what to consider when updating a stack, see AWS CloudFormation Stacks Updates (p. 89).	2010-05-15
Restructured guide	February 2, 2012	Reorganized existing sections into new sections: Working with AWS CloudFormation Templates (p. 127) and **Managing Stacks**. Moved Template Reference (p. 390) to the top level of the Table of Contents. Moved Estimating the Cost of Your AWS CloudFormation Stack (p. 78) to the Getting Started section.	2010-05-15

Change	Release Date	Description	API Version
New content	February 2, 2012	Added three new sections: • Walkthrough: Updating a Stack (p. 29) is a tutorial that walks through the process of updating a LAMP stack. • Deploying Applications on Amazon EC2 with AWS CloudFormation (p. 212) describes how to use AWS CloudFormation helper scripts to deploy applications using metadata stored in your template. • CloudFormation Helper Scripts Reference (p. 1354) provides reference material for the AWS CloudFormation helper scripts (cfn-init, cfn-get-metadata, cfn-signal, and cfn-hup).	2010-05-15
New feature	May 26, 2011	AWS CloudFormation now provides the aws cloudformation list-stacks command, which enables you to list stacks filtered by stack status. Deleted stacks can be listed for up to 90 days after they have been deleted. For more information, see Describing and Listing Your Stacks (p. 81).	2010-05-15
New features	May 26, 2011	The aws cloudformation describe-stack-resources and aws cloudformation get-template commands now enable you to get information from stacks that have been deleted for 90 days after they have been deleted. For more information, see Listing Resources (p. 86) and Retrieving a Template (p. 86).	2010-05-15
New link	March 1, 2011	AWS CloudFormation endpoint information is now located in the AWS General Reference. For more information, go to Regions and Endpoints in Amazon Web Services General Reference.	2010-05-15
Initial release	February 25, 2011	This is the initial public release of AWS CloudFormation.	2010-05-15

Supported AWS Services

AWS CloudFormation supports the following AWS services and features through the listed resources.

Topics

Analytics

Amazon EMR (Amazon EMR) (Updated in March 2016)

AWS Data Pipeline (Added in June 2015)

Amazon Elasticsearch Service (Amazon ES) (Updated in September 2016)

Amazon Kinesis (Updated in June 2016)

Application Services

Amazon API Gateway (API Gateway) (Updated in March 2017)

Amazon Simple Queue Service (Amazon SQS) (Updated in March 2017)

Compute

Elastic Load Balancing (Updated in October 2016)

AWS Lambda (Lambda) (Updated in March 2017)

Database

Amazon DynamoDB (DynamoDB) (Updated in December 2015)

Amazon ElastiCache (ElastiCache) (Updated in October 2016)

Amazon Relational Database Service (Amazon RDS) (Updated in January 2016)

Amazon Redshift (Updated in March 2017)

Amazon SimpleDB (Added in February 2011)

Developer Tools

AWS CodeBuild (Added in December 2016)

AWS CodeCommit (Added in October 2016)

AWS CodeDeploy (Updated in December 2015)

AWS CodePipeline (Added in December 2015)

Enterprise Applications

Amazon WorkSpaces (Updated in December 2015)

Game Development

Amazon GameLift (GameLift) (Updated in April 2016)

Internet of Things

AWS IoT (Updated in January 2016)

Management Tools

AWS CloudFormation (AWS CloudFormation) (Updated in April 2015)

AWS CloudTrail (CloudTrail) (Updated in February 2016)

Amazon CloudWatch (CloudWatch) (Updated in April 2016)

AWS Config (Updated in February 2016)

AWS OpsWorks (Updated in January 2016)

Mobile Services

Amazon Simple Notification Service (Amazon SNS) (Updated in November 2016)

Networking

Amazon Route 53 (Updated in March 2017)

Amazon Virtual Private Cloud (Amazon VPC) (Updated in December 2016)

Security and Identity

AWS Certificate Manager (ACM) (Added in August 2016)

AWS Directory Service (Updated in December 2015)

AWS Identity and Access Management (IAM) (Updated in March 2017)

AWS Key Management Service (AWS KMS) (Added in December 2015)

AWS WAF (Updated in April 2016)

Storage and Content Delivery

Amazon CloudFront (CloudFront) (Updated in November 2016)

Amazon Elastic Block Store (Amazon EBS) (Updated in November 2015)

Amazon Elastic File System (Amazon EFS) (Added in August 2016)

Amazon Simple Storage Service (Amazon S3) (Updated in March 2017)

AWS Glossary

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